FEATURE EXTRACTION AND KNOWLEDGE DISCOVERY IN PROCESS OPERATION ANALYSIS

by

Bing Hui CHEN

(B.Sc. M.Sc.)

Submitted in accordance with the requirements for the degree of Ph.D.

at

University of Leeds

Supervisors:

Prof. Colin McGreavy

B.Sc. (Leeds) M. Eng (Yale) D. Eng (Yale)

FIChemE C.Eng FBCS

Dr. Xue Zhong Wang

B.Sc. M. Sc. Ph.D. (China)

Department of Chemical Engineering

The University of Leeds

Leeds, LS2 9JT, UK

September 1998

The candidate confirms that the work submitted is his/her own and that appropriate credit has been given where reference has been made to the work of others.

ABSTRACT

An integrated framework for process monitoring is developed in this study which consists of three components: (1) feature extraction from dynamic transient signals using multiscale wavelet transform; (2) operational state identification using unsupervised and recursive learning methods; and (3) automatic extraction of knowledge rules from process operational data and embedding of the extracted knowledge in the structure and weights of fuzzy-neural networks. The methodologies and the prototype system which have been developed are illustrated and evaluated using data collected from a dynamic simulator of a refinery catalytic cracking process.

Methods for pre-processing dynamic transient signals for feature extraction, dimension reduction and noise removal are investigated and a new method is developed which makes use of wavelet transform to determine the singularities and irregularities of a dynamic transient signal by identifying the extrema from wavelet multiresolution analysis. The method is able to reduce the dimensionality of the data and removes noise components in a single step as well as capturing the most significant components of the dynamic response.

A modified version of the unsupervised neural network ART2, designated ARTNET, has been developed which uses wavelet feature extraction to provide a substitution of the data pre-processing part of ART2. ARTNET is shown to be more effective in avoiding the adverse effects of noise, less sensitive to user defined parameters and faster in computation, as well as still retaining the advantages of unsupervised and recursive learning.

Based on this, a fuzzy neural network is developed which is able to automatically extract knowledge rules from process data. The knowledge rules which are generated are transparent and explicit to operators. The method is therefore able to bridge the gap between numerical data and qualitative knowledge and takes advantage of the features of neural networks for capturing concepts and so provides an effective and robust method for learning knowledge from process data.

Various methods for integrating different facets of a problem, and making use of this informatin in parallel to mutually compensate for drawbacks of any single approach are also exploited.

Data obtained from a dynamic simulator of a refinery fluid catalytic cracking process (FCC) has been used to illustrate the methodologies and to evaluate a prototype system for using these new approaches. FCC provides a very useful case study because of the highly non-linear dynamics arising from the strong interactions between the reactor and fluidised bed regenerator derived from the mass and momentum balance. The use of simulation data makes it possible to look at the results in detail so that the methods can be fully tested. The case studies illustrate the potential of the methods developed.

CONTENTS

ACKNOWLEDGEMENT

I would like to thank my supervisors Prof. McGreavy and Dr. Wang, for valuable guidance, encouragement and criticism throughout the course of this work. Without their patience, faith and support, this thesis would not have been possible. I particularly indebted to them for giving me the opportunity to work on this project.

I gratefully acknowledge the financial support from Sino-British Friendship Scholarship Scheme (SBFSS)

I would also like to thank all my colleagues and friends at this Department for their help and advice. The very helpful discussions with Dr S. H. Yang are also appreciated.

I would like to dedicate the thesis to my wife, SHEN Xiao Qing and my children, Ying Hua and Ying Hao who accompanied me in such a difficult time. I also would like to dedicate the work to my parents for their care and love whatever the circumstances.

Figures

Tables

Chapter 1

INTRODUCTION

1.1 Research Motivation

In modern process plants controlled by computers, the role of operators has changed from being primarily concerned with control to a broader supervisory responsibility: analyzing operational data, identifying unusual conditions as they develop, and responding rapidly and effectively by taking corrective actions. This is a challenging task because of the overwhelming volume of data operators have to deal with: present day control rooms involve monitoring as many as thousands of process signals and hundreds of alarms. During the course of normal and steady state operations, simple observation of a smaller number of displays is sufficient to characterise process status. However, during a process transient or when a crisis occurs, the dynamic evolution of displayed data can overwhelm the operator because:

- process outputs change at different rates;
- there are transportation lags and inverse responses;
- control loops interact;
- there may be conflicting information from sensors;
- there is incomplete information due to 'lost' sensor readings.

These make it difficult for operators to carry out routine tasks such as:

- distinguishing normal from abnormal operating conditions;
- assessing current process trends and anticipating future operational states;

- identifying the cause of a process trend such as external load disturbances, process faults, operator-induced mishandling, degradation of performance due to changes in operational parameters;
- planning and scheduling operational sequences leads to the need for new procedures, e.g. recovery of process operation from a safety fallback position or to return to a desired feasible state after a process fault.

Clearly, to meet these requirements, it is necessary to improve the operational safety, reliability and productivity of process plants while reducing the load and stress on operators. This can be done by developing a process operational decision support system to assist the operators.

1.2 Process Operational Support Systems

The role of an operational support system (OPS) within the management structure is indicated in Figure 1-1. On-line signals are collected from computer control systems, which are used by the OPS for assessment and analysis and provide support to the operators by indicating the source of faults. An OPS needs to have the following capabilities: process monitoring and analysis, fault identification and diagnosis, operational planning and scheduling, intelligent control, optimisation, and advise on operational strategies, as indicated in Figure 1-2. The OPS should generally be capable of

- guaranteeing the response time
- handling efficiently large amounts of complex knowledge and data requiring real-time reasoning which may involve conflict
- learning so that the system can continuously improve in performance as well as enabling updating to be done after modifications or changes in operating conditions
- ensuring reliability of performance.

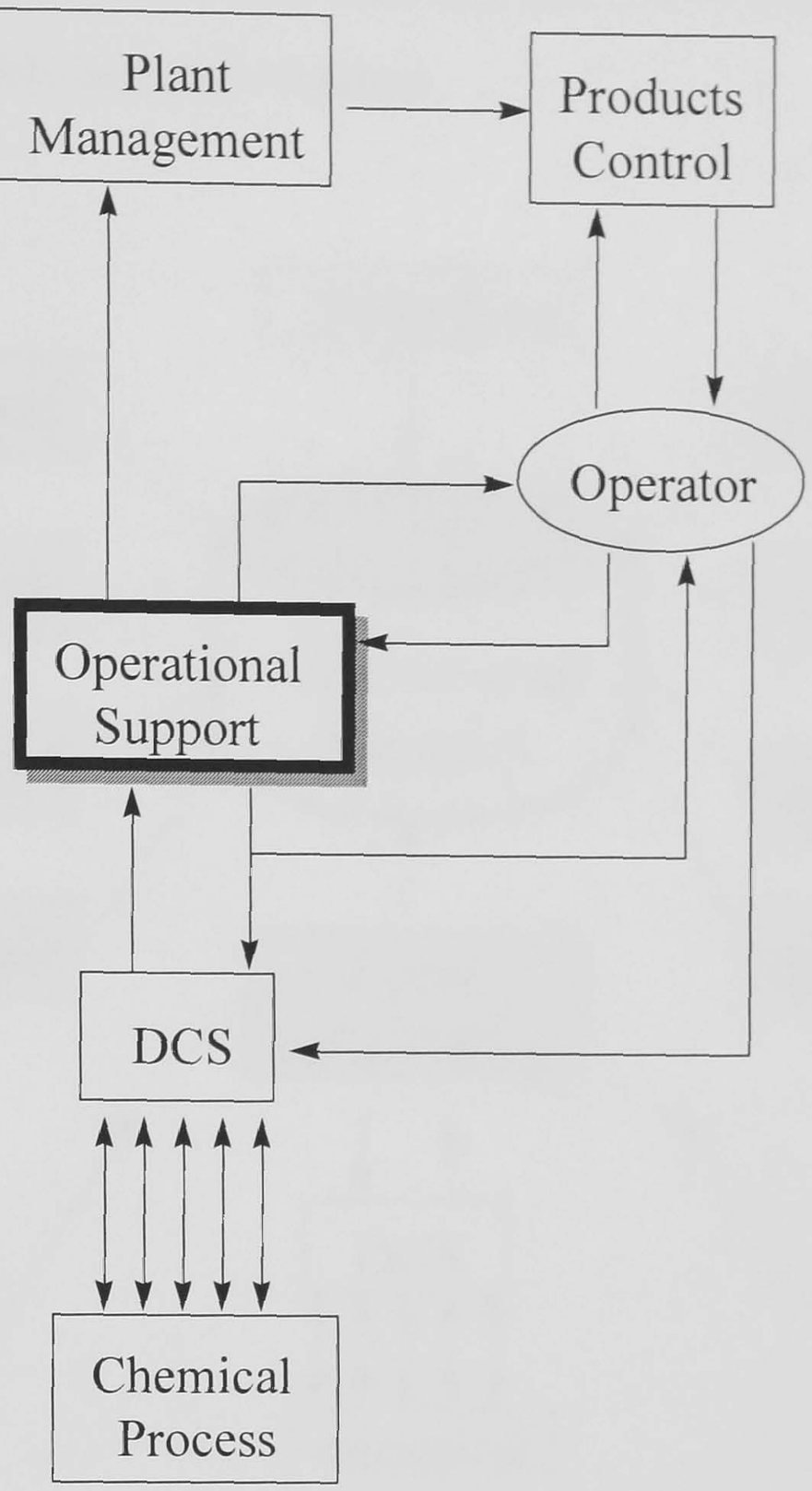


Figure 1 - 1 Role of an operational support system

Recently, Lindheim and Lien (1997) emphasised that the design and simplicity of an operator support system (OSS) and implementation, as well as the interface between the OSS and the user are all important. Flexibility is seen as an important factor that contributes to continuous learning and development (evolution). They emphasised that flexibility will be a major competitive factor in the future.

A variety of different techniques have been devised including development of qualitative (deep) models (e.g., Vinson and Ungar, 1995), knowledge-based expert systems (KBES) (e.g., Fathi et al., 1993, Saelid et al 1991), artificial neural networks, especially feedforward neural network (FFNN) (e.g., Kovio, 1994), fuzzy theory (e.g., Wang et al., 1996, 1997), graph theory (e.g., Chang and Yu, 1990) and fuzzy signed

digraphs (e.g., Wang et al, 1996, 1997, Shih and Lee 1995). Of these, KBES and FFNN have attracted considerable industrial interest.

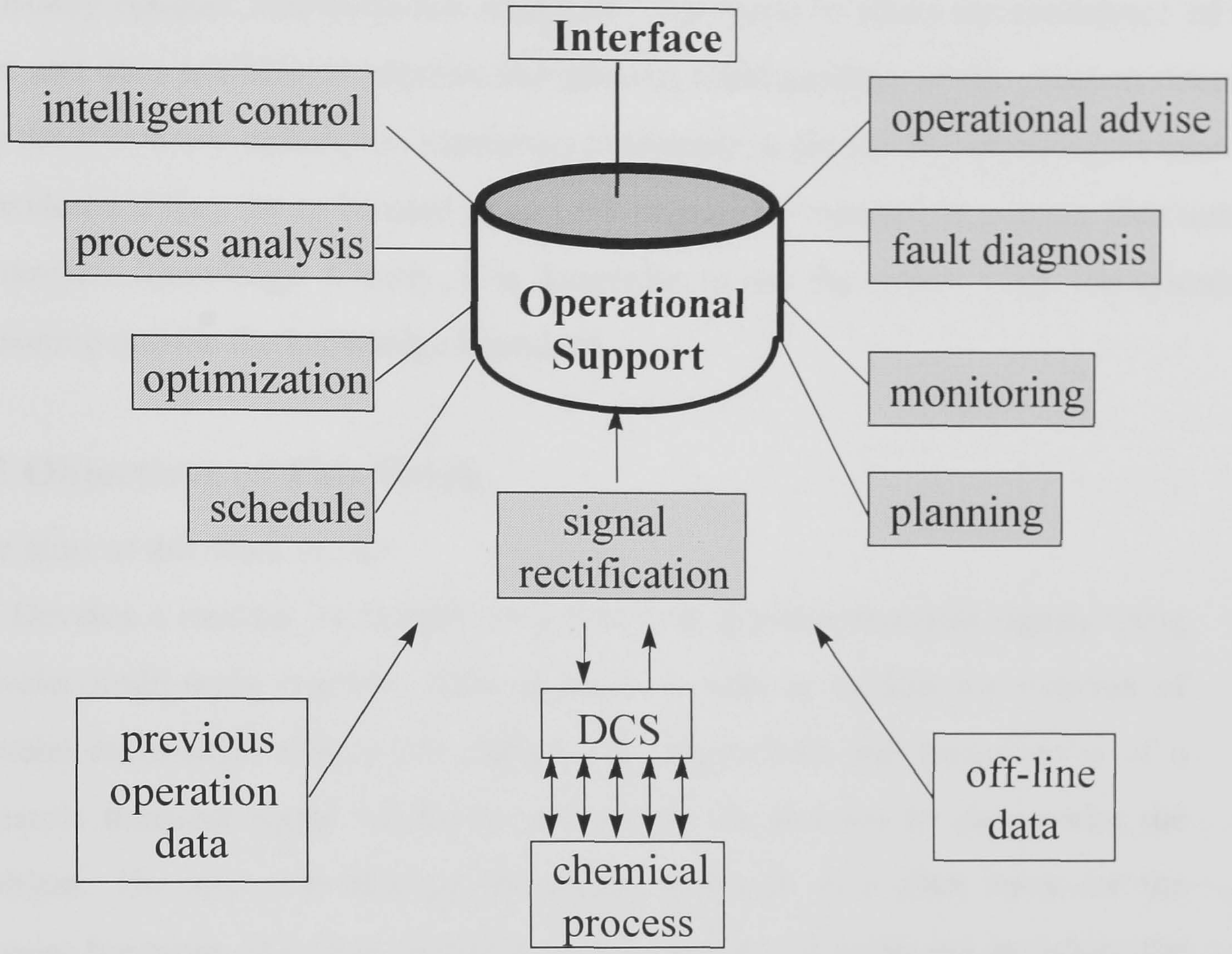


Figure 1 - 2 Content of operational support system

Expert systems make use of logic rules to carry out heuristic reasoning. Knowledge used to reach a conclusion is transparent and displayed through HOW and WHY explanation facilities. However, there are two critical issues in developing such knowledge based decision support systems for process plant operations. One is the bottleneck arising from knowledge acquisition, especially for complex processes where variables are highly interactive. The other relates to the fact that hazard and operability problems are often associated with the detailed plant topology, system properties and detailed design, which requires the systems to have the ability to model the structure and describe the behaviour of processes based on the propagation of disturbances.

Artificial neural networks (ANN) generally are able to learn the complex non-linear relationships involving multiple inputs and outputs (Morris et al., 1994). However, a major disadvantage with such an approach is that the knowledge embedded in the ANN is usually opaque. The black-box image of ANN tends to affect the confidence of the user and does not help to improve the inherent understanding of the problem domain. For the above two techniques, moreover, confidence in the reasoning procedure must be considered if they are to be used in an OPS because of incomplete training data sets or incomplete knowledge. Clearly, it is dangerous to use the results when the system is reasoning outside the knowledge boundary.

1.3 Objectives of This Work

The aims of this work are to:

(1) Develop a method for feature extraction from dynamic transient signals using wavelet multi-scale analysis. This approach is able to exploit the extrema of wavelet multi-scale analysis to capture the singularities and irregularities of a dynamic transient signal which can be used as the features to characterise the problem. The approach requires the proper selection of a filter bank for the wavelet functions. It is also necessary to take account of noise and to reduce the dimensionality of data sets

(2) As noted, the adaptive resonance theory (ART2) is not ideally suited for use with the problems described here, so there is a need to develop procedures for unsupervised recognition of operational patterns, which can combine wavelet techniques for feature extraction from dynamic transient signals with the basic algorithm of ART2. The goal is to provide a more effective way of dealing with noise contained in the transient signals while retaining an unsupervised and recursive clustering approach to identify the structure characteristics of a problem.

(3) This requires a means of generating fuzzy rules from process data which can take advantage of fuzzy concepts together with feed-forward neural networks so

as to be able to extract knowledge from numerical data. If the resulting knowledge is explicit and transparent it can provide more useful support for operators.

(4) Integrate various tools such as wavelets for feature extraction, identification of operational states using unsupervised neural network and knowledge discovery using fuzzy neural networks, to eliminate conflicts in the knowledge base and so avoid errors from extrapolation.

(5) The resulting methods will be used to develop a prototype system which can be tested by looking at a case study based on refinery fluid catalytic cracking in order to test whether it is able to deal with a representative range of operational problems effectively.

1.4 Thesis Organisation

The thesis is organised as follows:

Chapter 2 provides a comprehensive literature review on topics related to this study, including pre-processing of on-line dynamic signals for feature extraction via noise removal and dimension reduction, extrapolation issues of back-propagation neural networks, as well as knowledge discovery from data.

Chapter 3 introduces the feature extraction method using wavelet multi-scale analysis. The method identifies the extrema of a dynamic trend signal through wavelet multi-scale or multi-resolution analysis, which reduces the dimensionality and removes noise components simultaneously. The features extracted provide the inputs for identification of operational states and knowledge discovery.

Chapter 4 describes an integrated framework ARTNET which combines wavelet feature extraction and an unsupervised learning algorithm ART2, the former being used as the substitute for the data pre-processing part of ART2. ARTNET has shown advantages over ART2 in dealing with noise contained in dynamic transient signals and in improved computational performance.

Chapter 5 is devoted to the development of a fuzzy neural network method for generating production rules from numerical data. The ability to deal with conflicting data using a fuzzy-FFNN is illustrated based on the cause-effect simulation of a heater exchanger.

Chapter 6 concentrates on the application of the methods developed in previous chapters to a refinery fluid catalytic cracking (FCC) process. A customised prototype for feature extraction and knowledge discovery and knowledge base organisation is developed and applied to the FCC process. The ability of avoiding extrapolation is evaluated when the prototype is tested.

Chapter 7 presents the major conclusions and suggestions for future works.

Appendix A gives the related wavelet definitions and theorems used in Chapter 3, these are edited based on the works by Mallat and Hwang (1992) and Mallat and Zhong (1992)

Appendix B describes the ART2 architecture and its algorithm as used in Chapter 4

Chapter 2

Literature Review

2.1 Introduction

This Chapter provides a literature review on pre-processing of on-line measurements for noise removal and dimension reduction, extrapolation issues related to the use of neural networks for operational state identification, as well as knowledge discovery from numerical data. It concentrates on technologies that are relatively new and relevant to this study.

2.2 Signal Pre-processing and Data Interpretation

In many practical applications of neural networks, data pre-processing is one of the most significant factors in determining the overall system performance. One of the major issues to be resolved is that of reducing dimensionality with minimum loss of information. In this section, various methods for pre-processing on-line dynamic transient signals are considered. These include compression algorithms, transient representation using episode description, and feature extraction using principal component analysis (PCA).

2.2.1 Data compression

The data compression problem can be defined as determining the approximate representation $\hat{F}$ of a signal F so as to minimise the approximation error expressed as:

$$\left\|F-\hat{F}\right\| = \left[\int \left|F(x)-\hat{F}(x)\right|^{p} dx\right]^{1/p} \qquad \textbf{(2 - 1)}$$

$$0 < p < \infty$$

The simplest and most widely used data compression approaches in chemical manufacturing are piece-wise linear approximations. One of the earliest is the Box Car algorithm (Hale and Sellars, 1981) which records a value only when the current value differs from the last record by an amount greater than or equal to the recording limit (specified by the user) for the variable. The procedure for implementation is simple and is quite effective for processes with long spells of stable operation. However, it is not very effective when there is a drift or during a transition between steady states. Thus, in the case of rapidly increasing or decreasing trends, the algorithm does not result in good data compression. On the other hand, if the rate of change is slow, it does compress the data but at the expense of loss of information. Efforts to overcome this shortcoming have led to the development of the Backward Slope algorithm (Hale and Sellars, 1981) where the current value of a variable is predicted based on a linear extrapolation of the last two recorded values. If the actual value differs from the predicted value by an amount greater than the pre-specified recording limit, then the current value is recorded. This algorithm retains the merit of simplicity. For processes involving ramp and step changes, it achieves high compression for noiseless data but in practice the algorithm does not always improve performance, especially when there is noise present which results in poor predictions. Thus, when the process state is steady, noise can result in a non-zero slope which results in incorrect estimates of the state. In such cases, the Box Car algorithm tends to perform much better.

To try to capture the advantages of both techniques, Bader and Tucker (1987) combined them into a single algorithm that dynamically selects the technique to be applied when processing the next data point. The combined algorithm records a value when an exceptional value is indicated by both the box car and the backward slope algorithms. As long as the trend continues, only one criterion needs be checked. However, as soon as this criterion fails to be satisfied, the second criterion is tested. If that criterion also fails, a new value is recorded. This works better than either of the other two algorithms individually, but obviously requires more computation. It has been widely used in the process industry, even though its performance is far from satisfactory.

Swinging Door Trending (SDT) by Bristol (1990) is a heuristic straight-line trending and compression technique. It strives to give the longest straight-line trend possible, given the data and the maximum allowed error. It also attempts to minimise computation. In essence, the algorithm replaces a sequence of consecutive data points by a straight line, defined by initial and final points. The algorithm specifies how long a sequence or time interval should be and where the final point of that interval and the initial point of the next interval is located. The first data point is taken as the initial data point of the interval. Thereafter, the procedure is applied sequentially to each subsequent time interval. The SDT algorithm is fast in computation but is weak when dealing with noisy data and process variations which have outliers.

The PLOT algorithm based on piece-wise linear on-line trending developed by Mah et al (1995) is designed to compress noisy signals. Given a time series of process measurements, it determines all major trends or, equivalently, obtains maximum possible data compression by the use of piece-wise linear smoothing. The goals of the algorithm are:

1. to estimate the straight line as precisely as possible for any trend interval and detect any change in the trend. This determines the break point, marking the beginning of a new trend interval at the earliest opportunity;
2. detect outliers in the data and not let the trends be affected by them. For this purpose a least squares straight line is fitted to the sampled data collected during the current trend interval. The estimated straight line is updated sequentially as each new data point becomes available. The least squares (LS) method also provides an estimate of the process variance, σ^2. Since it is assumed that the variance, σ^2, is constant, it can be estimated over successive time periods.

In general, piece-wise linear algorithms approximate a process signal by representing it by a set of basis functions when $\hat{N}$ coefficients are used to represent the approximated signal:

$$\hat{F}(x) = \sum_{i=1}^{\hat{N}-1} c_i y_i(x) \qquad (2 - 2)$$

where

$$y_i(x) = m_i x + d_i$$
$$x \in [x_i, x_{i+1}]$$

and

$$m_i = \frac{\hat{F}(x_{i+1}) - \hat{F}(x_i)}{x_{i+1} - x_i}$$
$$d_i = \hat{F}(x_i) - m_i x_i$$

Generally, linear extrapolation-based data compression techniques work on the assumption that the local error between the actual and compressed signals is within given bounds. They are fast, but are unable to achieve good compression when the sensor data changes slowly. Moreover, they are not robust in dealing with process noise, so are only generally suitable for steady state analysis. To overcome these disadvantages and achieve better compression, especially when capturing dynamic trends, multi-scale techniques that use localised basis functions have been developed (Bakshi and Stephanopoulos 1996). The approach used in their work is based on decomposing the signal into wavelet basis functions, which are localised in the time-frequency domain. Criteria such as acceptable loss of information and identification of relevant features are used to select the best bases and coefficients from the wavelet packet used to represent the compressed signal. Time varying wavelet packet decomposition algorithms are applied when used on-line. These store the data in terms of contributions in the time-frequency domain and so capture the dynamic features as well as achieving greater compression than other methods. However, a disadvantage of the multi-scale techniques is that the data has to be reconstructed from the coefficients each time it is retrieved, which is computationally inefficient. Also the efficiency of compression depends on the basis functions. The selected bases have also to be classified as to whether they are concerned with storing compression information or feature extraction. Still noise influence is not taken into account in the technique.

Recently Vedam and Venkatasubramanian (1998) have developed an approach based on multi-level dyadic B-splines. The data is stored along with a time stamp. The approach, in contrast to other multi-level methods, retrieves the compressed data without requiring

time consuming computation because the output of the compression algorithm is the reconstructed data not the coefficients. Thus, the compression data obtained can be used for a variety of applications. Unfortunately, while the approach avoids inefficient computational reconstruction, it is necessary to use a time-consuming compression algorithm.

2.2.2 Episode methods

The episode method originated from B. William (1986) and is often employed, with some modification, for qualitative interpretation of transient process signals. It is based on being able to express any function totally by defining the nine primitives shown in Figure 2-1. Each primitive consists of the sign of the function value and the first and second derivatives. Each primitive therefore carries information about whether the function is positive or negative, increasing, decreasing, or not changing, together with the concavity. Combinations of episode form the trend over an interval described by a primitive and the associated time. A trend is a series of episodes that completely describe the qualitative behaviour of the system.

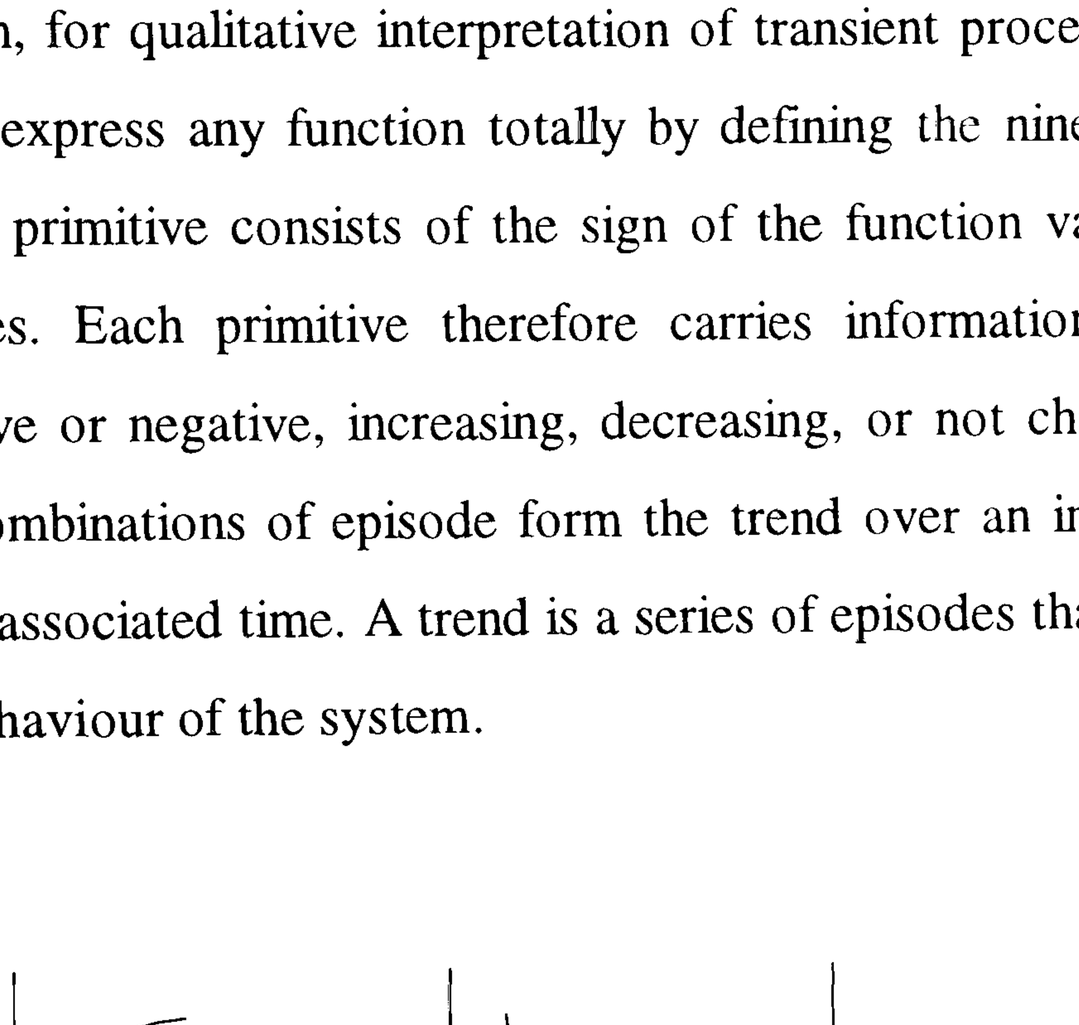


Figure 2 - 1 The primitives for the episode approach

Using this approach, Janusz and Venkatasubramanian (1991) developed a framework for automatic generation of qualitative process trend descriptions from on-line sensor data as shown in Figure 2-2. The framework is a classification tree to characterise a trend by representing it in terms of episodes. The approach is simple but is inefficient when dealing with multi-sensor signals simultaneously. Speed is mainly dependent on the classification algorithm that follows a trend point by point and then piece by piece.

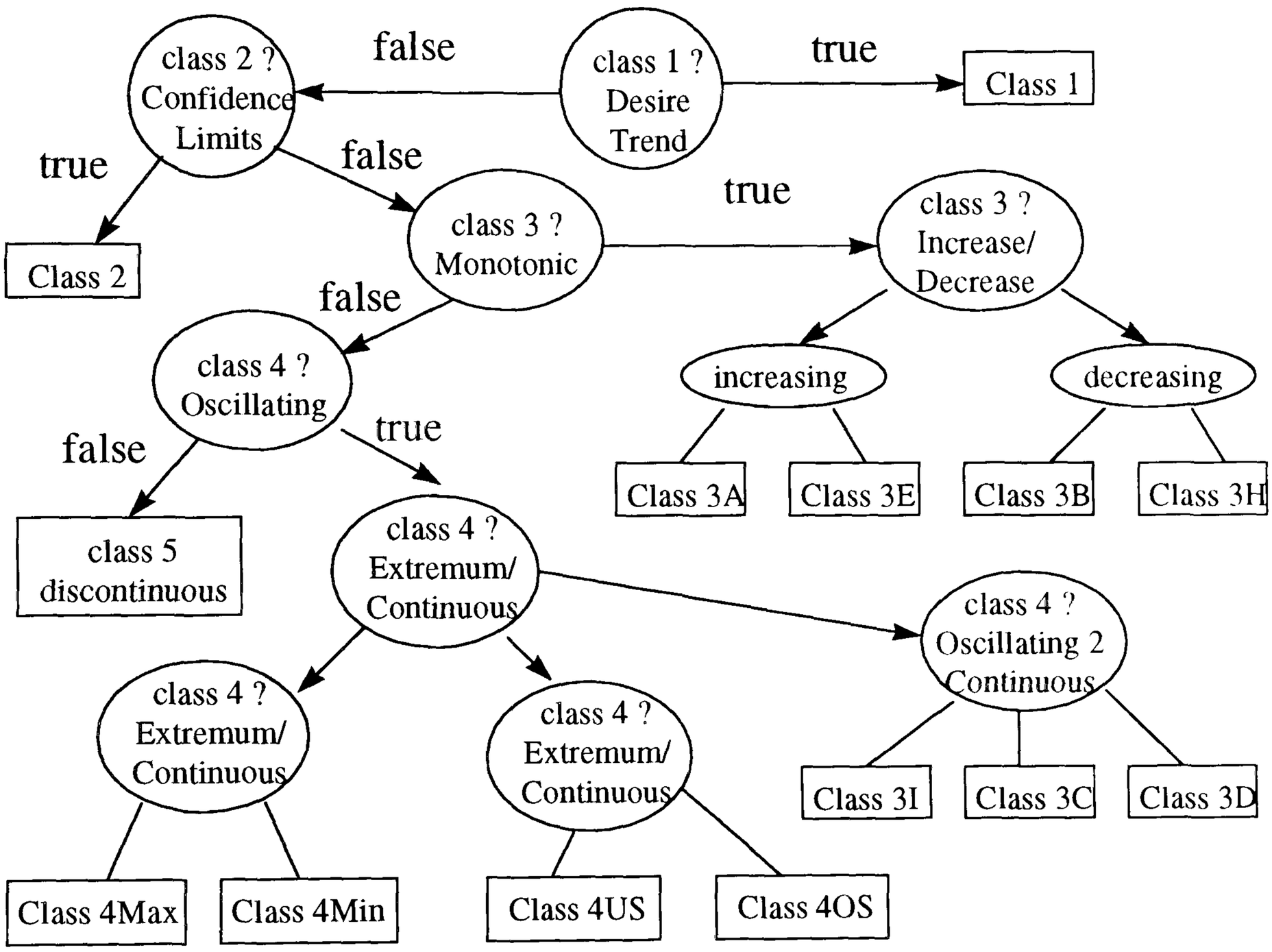


Figure 2 - 2 Trend classification tree base on episode method in Janusz and Venkatasubramanian's work

Cheung and Stephanopoulos (1990) employed a technique using seven triangle components, by dropping the C and D episodes in Figure 2-1. They then extended it to a triangular-episode representation of the process trends. The triangular-episode based description of a trend permits declarative modelling of all the important features contained in a trend and provides the basis for unambiguous inferences during various engineering activities. The speed is comparable with that quoted by Janusz and

Venkatasubramanian (1991). The process trend is the sequence of maximal scaling episodes over a time interval, where each point is strictly ordered in time (Bakshi and Stephanopoulos 1994). The points defining the maximal scaling episodes turn out to be inflexion points and are given by the scale-space image of a record of data. Wavelet decomposition is used because multi-resolution analysis by wavelets provides a scale-space image and zero-crossing of the second derivative wavelet function corresponds to the point of inflexion of the signal. Thus, signals are analysed at different resolutions by wavelet decomposition, and the points of inflexion are found by zero-crossing of the wavelet transform on the different scales. Trends are then identified by deductive learning algorithms such as decision trees.

The main idea of dynamic trend interpretation in Janusz's and Cheung's work is to classify a trend into increasing or decreasing pieces. Such an interpretation is sometimes not enough and may prove inadequate for process analysis. Furthermore, there is no noise filtering in any of the episode-based approaches, which significantly limits the identification and trend representation capability.

2.2.3 Principal component analysis

The idea of principal component analysis (PCA) was developed about 100 years ago (Sylvester, 1889), but has now re-emerged as an important technique. It employs linear mapping of multidimensional data into lower dimension spaces, with minimal loss of information. For an m dimension data set, if $X = [x_1, x_2, \cdots x_m]$, the first principal component is a linear combination of the columns of X which describes the greatest variation by $t_1 = Xp_1$. The objective is to map the m-dimension vector X in a L-dimension space of L principal components, where $L << M$. In the M-dimension space, p_1 defines the direction of the greatest variation, and t_1 represents the projection of each observation vector onto p_1. The second principal component accounts for the greatest variation of the residual data E_1 by $t_2 = E_1 p_2$. This procedure is repeated for the L principal components. If the variables in X are correlated, most of the variation in the data set X will usually have been explained by these L principal components. Hence X can be written as $X = TP' + E$ where P are eigenvectors of the covariance matrix of X.

Principal component analysis has been widely applied in many areas (Dong and McAvoy 1996), but it does have some difficulties in applications. Most significantly, it is a linear method, and most practical problems are non-linear. Applying PCA to non-linear problems is not always satisfactory (Palus and Dvrak 1992). Indeed if PCA is used for non-linear problems, the minor components not only reflect noise and minor variances, but contain important information (Xu et al 1992). If minor components are discarded, important information might therefore be lost. On the other hand, if the minor components are kept, too many components are retained for it to be useful.

Because multi-layer neural networks are auto-associative perceptrons, they can be used to perform non-linear dimensionality reduction, so they are useful in extending linear PCA to non-linear problems. Rumelhart et al (1986) considered a d-input, d-output node with M-hidden node (with $M < d$) in a three layer auto-associate neural network for non-linear principal component analysis. The targets used to train the network are the input vectors onto it. Because of the small number of units in the first layer, a perfect reconstruction of all input vectors is not generally possible. Such a network is trained by minimising a sum-of-squares error. This network can be regarded as unsupervised training, since no independent target data are provided. If the hidden units have linear activation functions, then it can be shown that the error function has a unique global minimum. At this minimum the network is a projection onto the M-dimensional sub-space spanned by the first M principal components of data.

While it might be thought that the limitation of a linear dimensionality reduction could be overcome by using a non-linear activation function for the hidden units in the network, it was shown by Bourlard and Kamp (1988) that such non-linearities make no difference and that the minimum error solution is again given by the projection onto the principal component sub-space. This means there is no advantage in using one hidden layer neural networks to perform dimensionality reduction for non-linear principal component analysis.

On the other hand, the situation is different when more hidden-layers are included in an auto-associative network. Kramer (1992) presented a non-linear principal component analysis based on a neural network. The architecture of the neural network is d-input and output linear node with a M-hidden linear node to the second hidden layers while the first and third hidden node are non-linear, as shown in Figure 2-3. This approach potentially can perform a non-linear principal component analysis because the mapping layer (the first hidden layer) and inverse mapping layer (the third hidden layer) are non-linear. The minimisation of the error function is now a non-linear optimisation problem because the error function is no longer a quadratic function of the network parameters. Computationally intensive non-linear optimisation techniques must be used and there is the risk of finding a sub-optimal local minimum of the error function. Also the dimensionality of the sub-space must be specified in advance of training the network. In fact, the hidden layers are difficult to determine and the theoretical meaning of the outputs of the bottleneck layer and the second hidden layer are not clear.

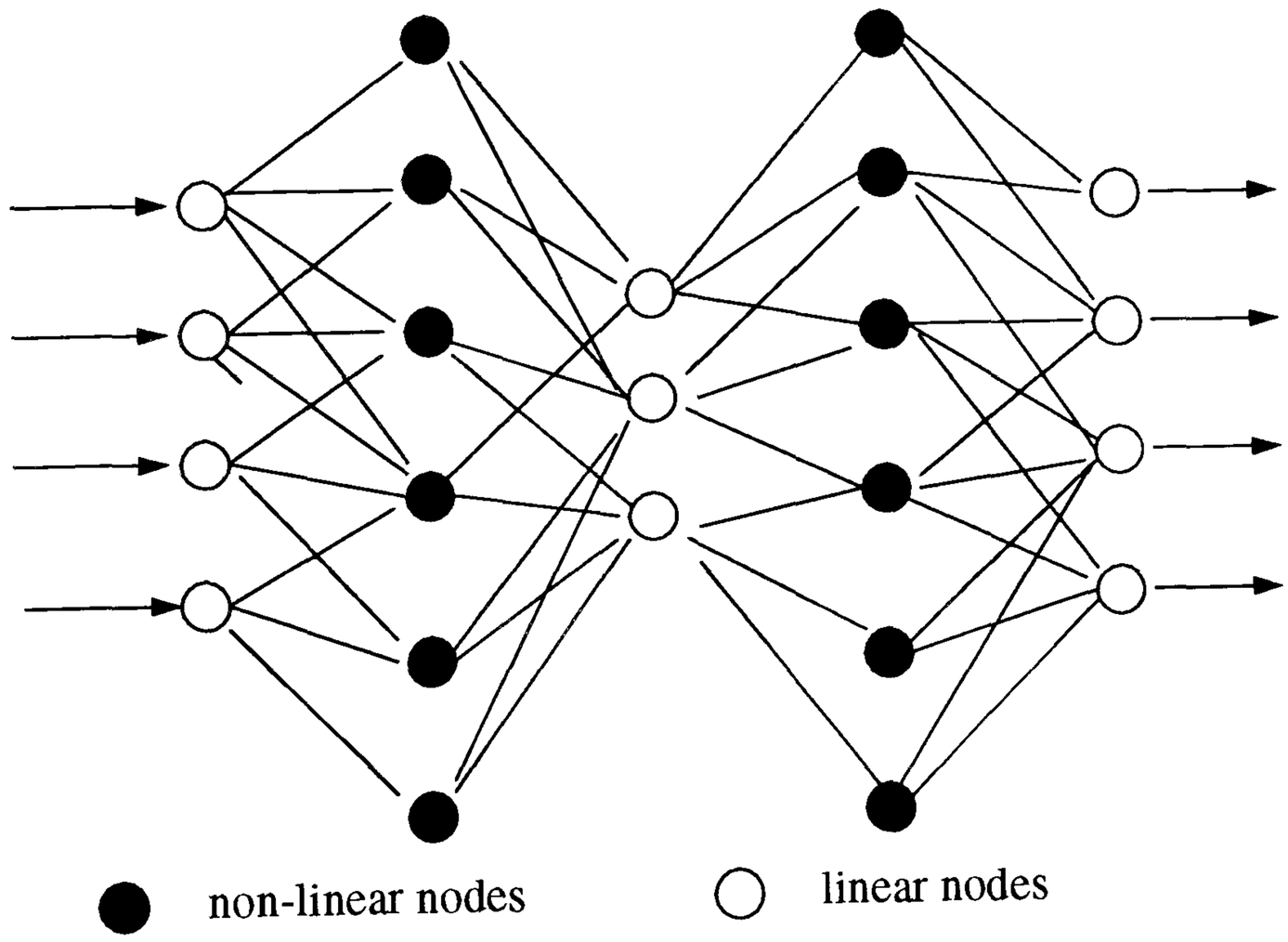


Figure 2 - 3 Auto-associative neural network in Kramer's work

Another approach to embedding non-linearities into PCA is generalised PCA (Gnanadesikian 1989). The basic idea of this approach is to extend an M-dimension variable X to include non-linear functions in the elements. The other approach is non-linear factor analysis by Etezadi-Amoli and McDonald (1993). In this approach, L-dimension polynomials are used to approximate M-dimension data. A linear least square method is used to find the coefficients of the polynomials. The difficulty in using these approaches is that it is necessary to know what kind of non-linear functions exist between variables. For high-dimensional data, these approaches become tedious because of the complicated non-linear functions.

Dong and McAvoy (1996) reported a non-linear PCA method that combines the principal curve method and a feed-forward neural network. A principal curve method is developed by Hastie and Stuetzle (1989) and is a generalisation of the first linear principal component. However, the algorithm does not offer a non-linear mapping. In Dong and McAvoy's work, a three-layer neural network with one non-linear hidden layer is used to generate a non-linear loading function. Although it proves to be a generalisation of the principal component, the method still suffers from a need to know how to determine the neural network structure although the percentage of explained variance is used to choose the number of non-linear principal components.

2.3 Extrapolation and Reliability Evaluation

Reliability of an operational support system is obviously the most important issue in system evaluation. This depends very strongly on the knowledge boundary of the system, i.e. whether the reasoning is within the knowledge base or not. Most of the methodologies used to build an operational support system, such as rule base expert systems, feed-forward neural networks and fuzzy logic are unable to automatically flag when they are extrapolating beyond the knowledge boundary. This issue is hardly discussed in rule-base expert systems because it is difficult to determine the knowledge boundary, especially when the rules are generated by interviewing experts and operators. Little effort has been devoted to determine the confidence bounds for supervised learning neural networks. Here, attention is focused on defining confidence bounds for supervised

learning neural networks as well as unsupervised learning clustering, because the work on supervised neural networks is also based on the distance metric for unsupervised learning.

2.3.1 Confidence bounds for supervised learning

Confidence bounds for supervised learning have been addressed by many researchers (Zhang et al 1997). The reliability of a neural network or any other empirical model is determined by two factors: extrapolation and local goodness of fit. The former implies that the model is being applied in a domain of independent variables where training data are available. The latter relies on the distribution of training data, i.e. the density of the data. Leonard et al (1992) introduce a novel neural network architecture known as validity index network (VI-net), which is an extension of radial basis function networks (RBFN) to estimate the reliability and the confidence of the output and indicate local regions of poor fit and extrapolation. Figure 2-4 illustrates the structure of such a VI-net with a reliability measure confidence limit (CL), local data density (ρ) and membership function for hidden units (max-act). The RBFN is indicated by a dashed line.

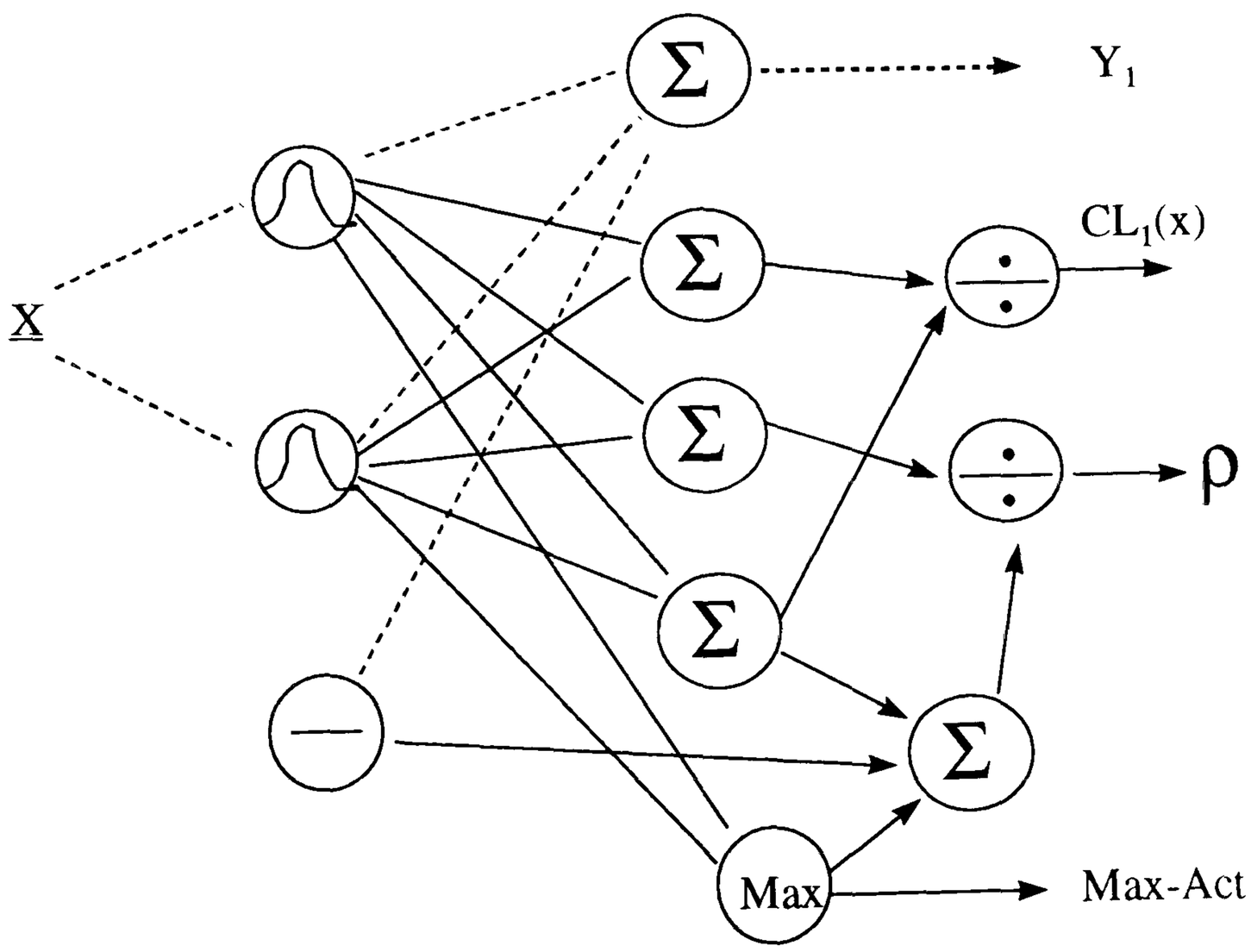


Figure 2 - 4 The VI-net (Leonard et al 1992)

After choosing a structure for the RBFN, the train-test method, known as the S-fold cross validation procedure, is used to train the VI-net. For the S-fold, the training data are randomly partitioned into S equally sized subsets. Typically S ranges from 10 to the total number of training examples. Therefore, the S-fold cross validation procedure makes maximum use of the available data while still giving an unbiased error estimate. The authors recommend that the S-fold should be repeated several times to determine the mean and variation of the unbiased error estimates for each architecture. The structure with minimum error during the train-test procedure is considered to be optimal for the VI-net. Confidence limits and local data density are automatically generated by the calculation.

The additional weights of the VI-net are based on statistics produced during the training and cross validation procedures of the basic RBFN model. VI-net involves minimal extra work because no additional training is required and all parameters in the underlying RBFN remain the same. Unfortunately, VI-net is not appropriate for the feed-forward neural network or error back propagation network, and there is no guarantee of finding the optimal net because the train-test procedure is used in the optimisation. Furthermore, the train-test is a very time-consuming procedure although no additional training is required for the RBFN.

Chryssolouis (1996) has derived a method of quantifying the confidence intervals for the prediction of neural network models using a linearised variant. The confidence intervals can be calculated for a desired level by extending standard statistical approaches which describe how well the neural network fits the training data. However, it does not account for the effect of the statistical distribution of the training data.

Shao et al (1997) have developed an approach to computing confidence bounds for predictions from a error back propagation feed-forward neural network with defined structure. The confidence intervals for parameters estimated by non-linear least squares are extended to the neural network models. A linearisation method is used which is reported to be computationally less expensive, numerically more stable and has acceptable accuracy. In this work, a constant, β, is introduced to include the influence of

distribution of the training data on the confidence interval, so the confidence bound for the error back propagation feed forward neural network model can be regarded as defining confidence intervals for compression of the data and is given by

$$\hat{y}_0 \pm ci \cdot \beta \qquad \textbf{(2 - 3)}$$

where $\hat{y}_0$ is the output of the network models, *ci* is the confidence interval and β is a coefficient which is inversely proportional to the density of the training data. They found that the confidence bounds are almost twice as wide as the standard confidence intervals when training data are not available in a given region.

The effect of extrapolation and training data distribution on the confidence bounds of the neural network models are also considered in this approach. The confidence bounds on the computation are based on the Jacobian matrix of the first order derivatives which involves a time-consuming calculation and makes this approach difficult to use for on-line applications. The density function for the training data set, ρ, is selected empirically and has no theoretical foundation.

2.3.2 Clustering Using Unsupervised Learning

The basic idea of the reliability and confidence bound approach to supervised learning is based on a distance metric which is also found in unsupervised clustering techniques. This allows unsupervised clustering to be used to avoid extrapolation of supervised learning neural network models. There are several unsupervised algorithms, among them *k*-means clustering, Kohonen's topology-preserving maps and ART (adaptive resonance theory), all of which have been widely used and are relevant to this work.

k-means clustering is widely used and has been adapted to address a range of applications. The algorithm chooses the centroid vector of a set of N training examples. according to the definition

$$\textbf{centroid vector} = \frac{1}{N}\sum S^k \qquad \textbf{(2 - 4)}$$

for the training sample set $\{S^k\}$.

Clustering operates by moving a case to the cluster from a centroid closest to it then updating the centroids for the remaining clusters. The algorithm can be summarised as:

1. Begin with any initial partition that groups the data into k clusters:
2. Take each case, S, in sequence and compute the distance from the centroid of each of the k clusters. If S is not currently in the cluster with the closest centroid, move S to the cluster and update the centroids of the cluster gaining S and the cluster losing it;
3. Repeat step 2 until a pass through the training examples causes no new assignments.

The k-means clustering algorithm is fast and efficient. However, it requires setting the number of clusters, k, in advance, which involves experimentation and is time-consuming. Another way of determining the number of clusters is to leave it to the algorithm by defining user-specified parameters. Works along these lines have been reported by MacQueen (1967), Wishart (1969) and Anderberg (1973). It is difficult to compare the variety of different clustering algorithms in the literature because comparisons are dependent on the criteria used to evaluate the final clusters. However, it seems evident that the k-means approach is good if the criterion is to minimise the sums of squares of distances between training cases and the corresponding cluster centroids. It also has the convenient property that every case is in a cluster having a centroid closest to that case (Gallant 1993).

In the k-means clustering algorithm, a case is assigned to the cluster having a representative nearest to it. This is precisely what happens in Kohonen's self-organising maps (Kohonen 1982, 1995), where the training algorithm attempts to assign some structure to the representatives. A large number of clusters are chosen, and arranged on a regular grid in one or two dimensions in the algorithm. The idea is that the representatives, called weights by Kohonen, are spatially correlated, so that representatives at nearby points on the grid are closer than those which are widely separated. This ensures that the self-organising maps are clustered according to the structure and feature similarity. Some results have been reported for one- and two-dimensional mappings that were topologically correct. However, even in the dimensional

case, Kohonen showed by simulation that an array of units starts to become sensitive to different frequencies in ascending or descending order, and there is no adaptivity (Gallant 1993).

These issues are successfully addressed by the adaptive resonance theory (ART) due to Grossberg (1976). It has evolved through three stages to ART3 dealing with autonomous learning models and is a type of competitive methodology equivalent to a winner-take-all algorithm. It is suitable for both category (pattern) formation and recall (recognition). ART1 (Carpenter and Grossberg, 1987a) depended on clustering of binary vectors, while ART2 (Carpenter and Grossberg, 1987b) extended this to continuous-valued inputs. There is no set number of clusters: these are created as needed to solve the adaptivity, especially the stability-plasticity dilemma defined by letting a system adapt and yet preventing current inputs from destroying past training.

ART1 is not suitable for process engineering application because it can only accept binary input values. Some applications of ART2 for process plant have been reported (Whiteley et al 1994, Wang and Chen 1998). The ART2 pattern clustering algorithm is a winner-take-all competition learning algorithm where recognition of an input pattern vector has to examine all existing pattern representatives i.e. weights. Clearly recognition time increases rapidly with the number of long term memory neurons used. Storing a large number of items in long term memory is a serious problem (Wang, 1993). The number of long term memory neurons is dependent on the input vector dimension and number of patterns stored in memory. Therefore, dimensionality reduction is always recommended for the input vectors. The input pattern data pre-processing in ART2 is a time-consuming procedure. The computational effort spent on this procedure is also dependent on the dimension of the input vector.

The major disadvantage of ART is the sensitivity to noise. The update rule can only reduce the co-ordinates of prototypes, so if a large number of examples are presented, each having added noise, the prototypes will shrink towards the zero vector. Prototypes that are close to the zero vectors will fail the vigilance test. Thus for enough training sets

true clusters will be divided repeatedly into groups which depend on the order of presentation of the example.

2.4 Knowledge Extraction from Numerical Data

There are several forms of learning, ranging from direct instruction to discovery. Direct instruction may require intelligent communication, including a learner's model of the teacher. At the other extreme, in learning by discovery, the learner autonomously discovers new concepts from unstructured observations or by planning and performing experiments in the environment. Between these two extremes, learning can be by example. Knowledge extraction from numerical data falls into this category.

For most processes, information can be classified into two kinds: numerical information obtained from sensor measurement, and linguistic information obtained from human experts. This section deals with reported methods for knowledge extraction from numerical data. There is a thorough review of machine learning by Michalski et al (1998). However, few examples of knowledge extraction from numerical data in chemical manufacturing have been reported so related cases are discussed. These approaches can be categorised as: inductive learning based on decision tree and rough sets theory, fuzzy sets operation, and neural networks.

2.4.1 Inductive learning

There are three most popular families of inductive learning: top-down-induction-of decision trees (TDIDT) (Quinlan 1986), attributes qualitative (AQ) (Michalski 1983), and learning from example based on rough sets (LERS) (Chan 1991, Quafafou and Chan1995).

The TDIDT approach is well suited to learning from uncertain data containing errors, usually called noisy data. This is an important aspect from the practical point of view. Examples of learning are described in terms of attributes. An attribute can be either symbolic or numerical. A symbolic attribute has an unordered set of values. Such a set is

typically small. A numerical attribute has an ordered set of values. Usually in TDIDT. several classes are learned simultaneously.

TDIDT programs produce decision trees that agree with the learning set of examples. In the case of noisy data, however, the learning procedure is allowed to generate a decision tree that only partially agrees with the learning data. That is, it does not necessarily reclassify the learning data into the classes specified in the examples. The point of this partial agreement of the synthesised tree is to eliminate noise that appears in the data. This is similar to statistical smoothing of data that reduces the effects of noise.

There are programs in the TDIDT family that can cope with several types of deficiency in the learning data. In addition to errors in the learning data, it may also be incomplete. In the case of incompleteness, the learning example can only be partially specified. A learning set can be incomplete also in the sense that it poorly represents the universe of all objects because the set may be very small compared with the complete attribute space. Noise and incompleteness make the learning task more difficult which is typical of many application areas.

In the basic TDIDT learning algorithm, a top-down decision tree is constructed by iteratively selecting the most informative attribute at the current node in the tree. The learning set is then partitioned into subsets according to the values of the selected attribute. The process of tree expansion is stopped at a given node when all the examples falling into that node belong to the same class or when at least one of the classes in the current set has a sufficient majority.

The technical issue in TDIDT is the problem of estimating probabilities from the learning data. This becomes critical when the learning sets are small. They typically become very small in the lower levels of a decision tree.

The best known family of programs that learn concepts represented by if-then rules is AQ which has many members. It is similar to TDIDT in that it uses the attribute-based framework. However, the decision tree representation of concepts is here replaced by the

more flexible if-then rule representation. Any decision tree can be easily converted into if-then rules whereas the reverse transformation is more complicated.

As in the TDIDT family, some AQ algorithms are well equipped with special mechanisms to cope with noise in the data. If-then rules are more flexible than decision trees and therefore a decision tree representation sometimes appears awkward compared with an equivalent if-then rule representation. On the other hand, TDIDT techniques are somewhat easier to implement and the corresponding programs are typically more efficient than the AQ set which suffers to some extent from combinatorial complexity.

The other inductive learning family is LERS that is based on rough set theory. Pawlak (1982, 1984, and 1985) first introduced the concept of rough set. Making use of the rough set theory for rule extraction for a dynamic system has been reported by several researchers (Chan, 1991; Srinivasan et al., 1993; Chmielewski et al., 1993; Quafafou and Chan, 1995). The basis of using rough set theory for learning from examples in LERS is illustrated in Fig 2-5 where the examples can be in the form of a decision table.

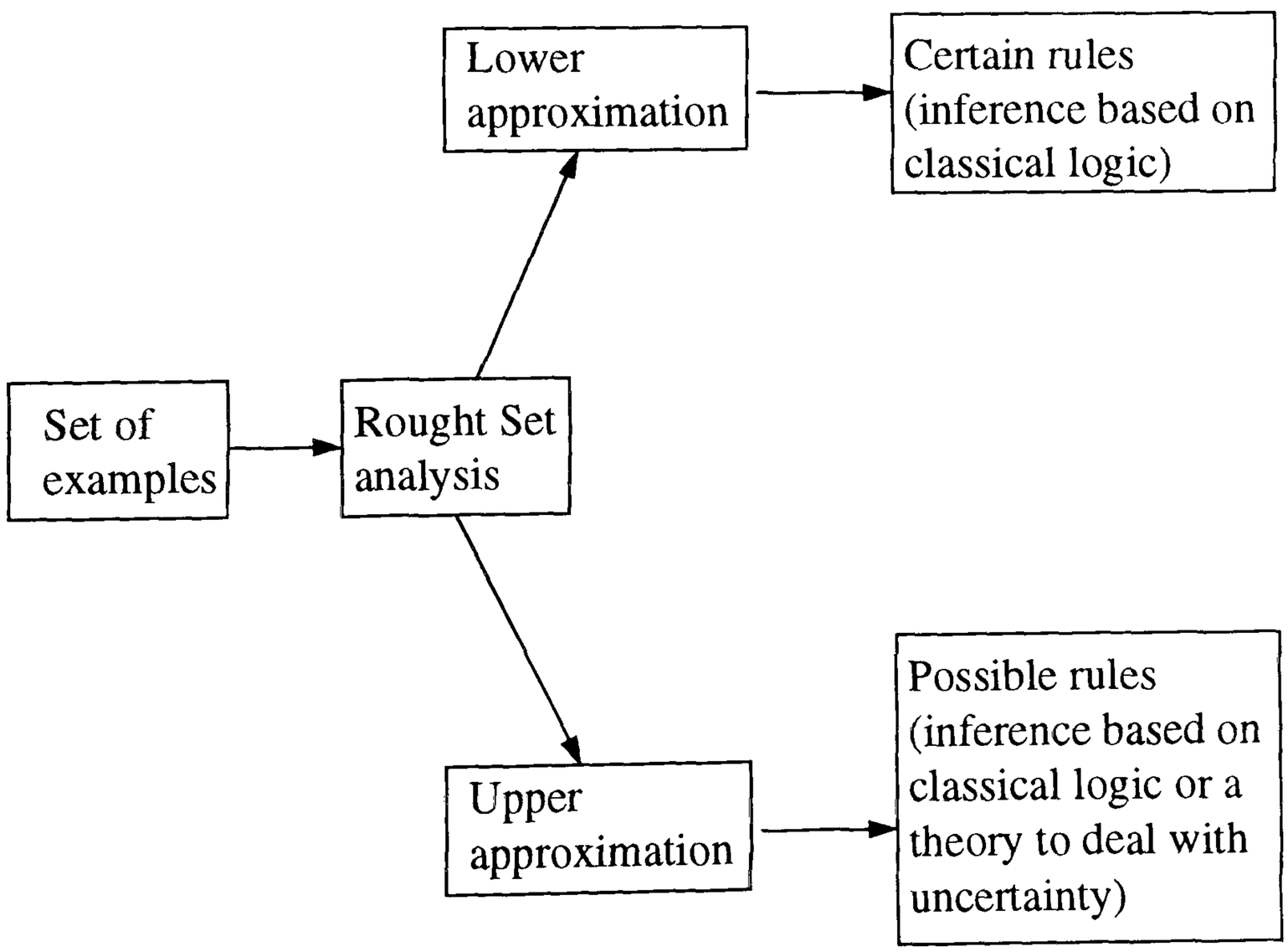


Figure 2 - 5 Principle of using rough set theory for learning from examples

The system LERS finds a minimal description of a concept, described by positive examples and excluding the remaining negative examples. Thus rules identified by LERS may be described as a minimal discriminating description of the concept. The advantage of the LERS is that it can quantise the numeric value of attributes and handle two kinds of uncertainty: missing values of attributes and inconsistent examples i.e. examples characterised by the same values of attributes, although the corresponding values of a decision are different. LERS handles inconsistencies using rough set theory that does not need any preliminary or additional information about the data.

A limitation has been pointed out by Quafafou and Chan (1995) because rough set methodology is based on fuzzy rule-based model learning. The process of building a fuzzy model starts by replacing the original values by linguistic variables that can be obtained, for instance, from elements. Any inconsistency generated by these methods is not evident and because the membership functions are considered at the beginning in the process of learning rules, the fuzzy model learned is strongly influenced by the quantising process.

2.4.2 Rule generation by fuzzy set operation

Wang and Mendel (1992) were one of the first to use fuzzy set for knowledge extraction from numerical data. In their approach, there are five steps needed to convert numerical data into fuzzy rules: (1) divide the input and output spaces of given numerical data into fuzzy regions; (2) generate fuzzy rules from the given data; (3) assign a degree of each for the generated rules; (4) create a combined fuzzy rule base; (5) determine a mapping from the input to the output space based on the combined fuzzy rule base, using a defuzzifying procedure. There are some advantages to this approach: (1) it provides a general method of combining measured numerical and human linguistic information into a common framework; (2) it is a simple and straight forward one-pass build-up procedure, so there is no time-consuming iteration; (3) there is a lot of freedom in choosing the membership function, which gives flexibility in the design of an intelligent system which will match different requirements.

One of the disadvantages in this approach is that input variable regions have to be fixed in advance. Abe and Lan (1995a) developed an approach to extract fuzzy rules directly from numerical data for pattern classification. Fuzzy rules with variable fuzzy regions were defined by activation hyperboxes which show the existence region for class data and inhibition hyperboxes which inhibit the existence of the data for that class. The rules were extracted from numerical data by recursively solving overlaps between the two classes. Then, according to the number of rules extracted, an approach is developed for the selection of optimal input variables. This was extended (Abe and Lan 1995b) to a function approximation which relates to deleting redundant input variables.

The other disadvantage of Wang and Mendel's approach is that the rule-base generated by fuzzy operators may be incomplete. A fuzzy learning algorithm provides an efficient single pass method for producing approximating functions from training data. Fuzzy algorithms are local: the decomposition of the input domains into a fuzzy partition produces a set of overlapping regions. This approximation over a region is determined solely by the example of a local connection that focuses information contained in it. Localisation introduces the possibility that regions within the domain may not cover the scope of the training data. Sudkamp and Hammell (1994) used this to establish rules and to interpolate and extend the training information to the entire rule base.

On the whole, using fuzzy sets to extract rules from numerical data is a simple and straightforward approach. However, it does have some disadvantages. First, one data pair should theoretically correspond to one rule, so the rules generated by this approach could be repeated and the set of rules can be large because the number of sensor measurements in process plant can be very large. Furthermore, the rules generated by the fuzzy set approach can be in conflict when data are from different sources.

2.4.3 Rule generation from neural networks

Feed forward connection neural networks with error back propagation algorithms, which extract rules from numerical data, were introduced by Gallant (1988). Gallant's method is able to find a single rule to explain the conclusion reached by the neural network for a given case. It involves the ordering of available attributes based on strength of inference,

i.e. the absolute magnitude of the weights. To form a rule, the remaining attribute with the greatest strength is picked. The process continues until the conjunction of these attributes is sufficiently strong to conclude the concept concerned. It generates clauses for an IF-THEN rule until the sum of absolute weights in a clause is greater than the sum of absolute weights of remaining attributes. At this point, that clause forms a rule premise. The rule must still be valid for the worst case, where all remaining positive attributes have negative activation and all remaining negative attributes are positive. In this approach, if the activation level is minus, the absolute weight for process evaluation is used. Notice that an attribute with a positive weight can contribute positively if the activation is positive, or negatively if the activation is negative. The same applies for an attribute with a negative weight. The search space for rules is much smaller in Gallant's approach since it focuses only on attributes that contribute positively to a single case.

Saito and Nakano's method (1988) finds multiple rules from a trained neural network. Their method searches through the rule space spanned by attributes selected according to given instances. The approach is empirical, and relies on observing the input-output behaviour of the trained network directly.

Hayashi (1990) describes a simple search technique for extraction of fuzzy rules from a neural network. The input units are organised as a set of groups. For each rule formation cycle, it selects one output cell and one input cell group but does not consider the interactions between different cell groups. The weights between the input and hidden layers are fixed during the learning process. Rule search is conducted directly in the space of primary attributes, without involving pattern formation and combination in the hidden layer. The overall search width is much more limited.

The rule generation method from neural networks by Fu (1994) is illustrated by reference to Figure 2-6 which is based on KT -- 'Knowledgetron' method. The word 'Knowledgetron' is defined by the author and refers to a neural network with knowledge. A rule generated has the form of

IF the premise, THEN the action (conclusion).

Specifically for the case in Figure 2-6,

$$\text{IF } A_1^+, \ldots A_i^+, \ldots \neg A_1^-, \ldots \neg A_j^-, \ldots, \text{ THEN C (or } \neg \text{C)}$$

where A_i^+ is a positive antecedent (an attribute in the positive form), $\neg A_j^-$ is a negative antecedent (an attribute in the negative form), C the concept (conclusion), and $\neg$ reads "not". Each node in the hidden or output layer is designated by a symbol that represents a concept to be confirmed or not confirmed. Confirmation or non-confirmation of a node concept is measured by the activation of the node.

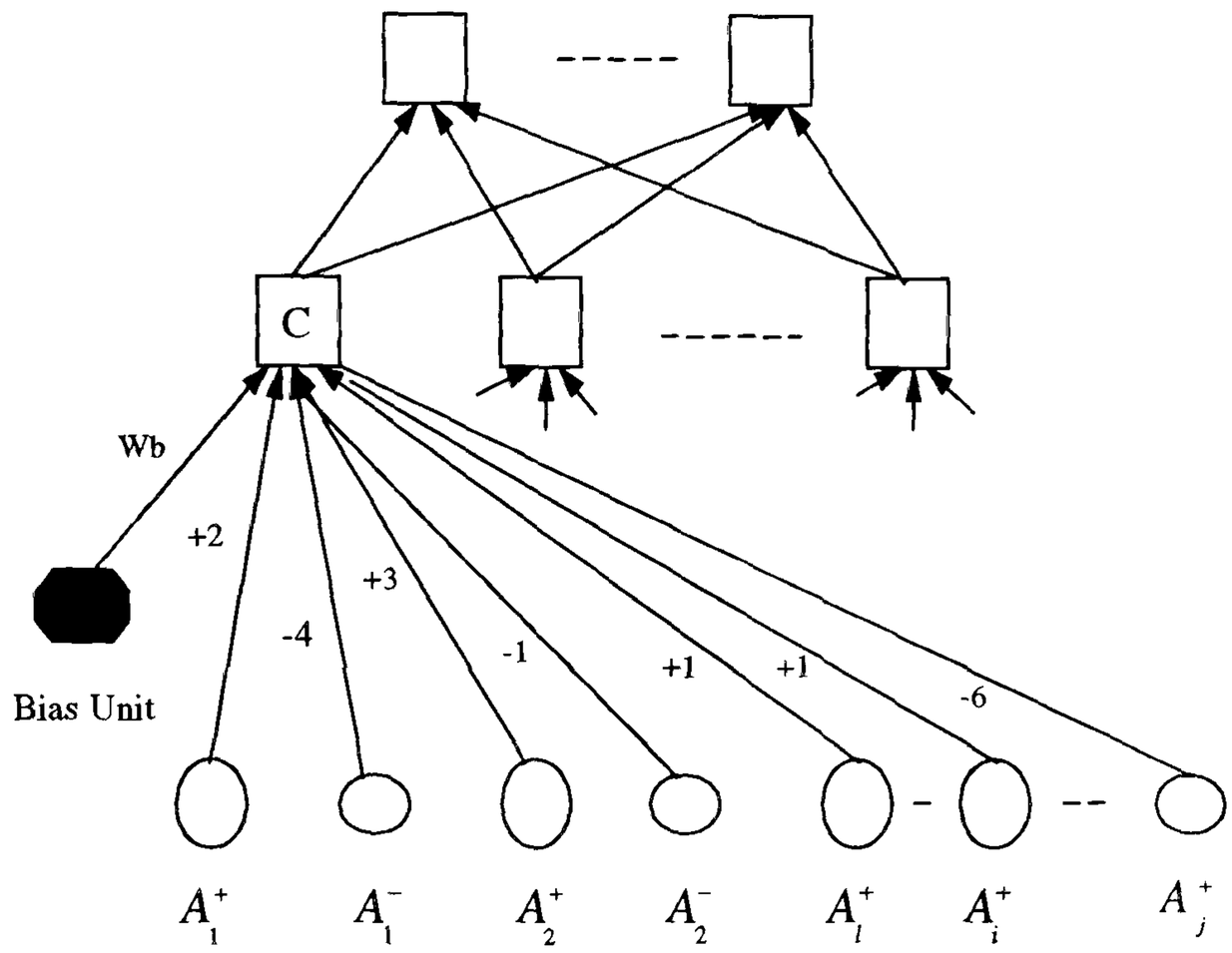


Figure 2 - 6 Network for rule generation by Fu (1994)

It is necessary to train the neural network first and then, based on the above concepts, a detailed procedure is developed for extracting rules. It overcomes the limitations of other methods (Gallant, 1988; Hayashi, 1990; Saito and Nakano, 1988) of extracting rules using trained neural networks.

2.4.4 Concluding remarks

Extracting rules from numerical data can be classified into two categories: incremental and non-incremental learning (Chan 1991). The former works on one example at a time and the latter is based on using all training examples at once. Both the fuzzy and rough set methods are incremental. Although a training neural network itself is by its nature non-incremental, the rules generated by training a neural network are still incremental. The problem with this approach is that some information contained in the numerical data

is lost during the results filtering which is a necessary step in the method. This is unlike a neural network that has to take into account all data used for training in order to obtain the weights. However, incremental learning has the advantage that it is recursive. This means that if new data is made available, it does not need to go back to train using the old data but simply updates existing knowledge. This is particularly useful for on-line applications since new data is often continuously gathered.

The method of *rule generation using fuzzy set operation* generates a rule for each data. It does not make use of information contained in the data because the procedures are independent. This becomes worse when the data pair become large.

The *rough set method* was originally not able to generate fuzzy rules. Quafafou and Chan (1995) improved the method so as to be able to generate fuzzy rules but there are still certain limitations, as noted by Quafafou and Chan (1995):

1. some elements can result in inconsistencies in the process of building a fuzzy model which starts by replacing the original values by linguistic variables,
2. the membership functions are considered early when exploring learning rules and the resulting fuzzy model is strongly influenced by the quantising process. In order to cope with this problem, the original values must be integrated into fuzzy rules related to the learning process. The membership function is used as late as possible.

The *neural network method* proposed by Fu (1994) depends on training a structured neural network properly. The nodes represent concepts and branches describe cause-effect relations. But the size of rules increases dramatically with the increase in the size of the problem.

2.5 General Observations

In the light of the above analysis, the following observations can be made.

Firstly, the episode-based approaches for qualitative interpretation of dynamic transients are easily adversely influenced by noise. Other approaches reviewed are not able to address the dynamic and time as well as local features of a transient

signal. Both the frequency and time of an event on a dynamic transient signal may carry important information. There is clearly a need to develop methods that are able to simultaneously remove noise components and reduce dimensionality. At the same time the extracted feature retains a high degree of fidelity with respect to the original signals.

It is clear that unsupervised neural networks are potentially powerful because they do not need training data and provide a means of avoiding extrapolation. These approaches should also be recursive so that they can continuously update their performance during application. Unfortunately most of the approaches are not developed specifically for data from process on-line measurements. Work needs to be done on these approaches, typically on issues of how to deal with signal noise.

Compared with the progress in knowledge discovery through automatic machine learning, little effort has been made in the process industries to generate knowledge directly from data. The major difficulty has been that existing approaches are mainly for dealing with variables that are symbolic or discrete, such as a colour being green or red. There is the need to develop methods that are able to bridge the numerical data with symbolic descriptions for the purpose of knowledge discovery.

The remaining chapters are devoted to exploring concepts using the above and to developing of a framework for integrating them. Application of the methods to processes that are at an industrial scale is also important for validation purposes.

Chapter 3

WAVELET BASED SIGNAL PRE-PROCESSING FOR NEURAL NETWORKS

3.1 Introduction

In process monitoring, dynamic trend signals may represent more important information than the instant values of variables. In fact operators spend more time on monitoring trends than the instant values. Dynamic trends are characterised by noise and high dimensionality. Firstly, with a high noise to signal ratio, it is difficult to distinguish the real change of a signal from that due to noise. Secondly, a segment of a dynamic trend may consist of tens of sampling values and a process may have several hundreds of such trends being continuously monitored. Therefore it is necessary to reduce the dimensionality by representing trend using a minimum number of features before using it with tools for process operational state identification and diagnosis.

The wavelet based signal pre-processing technique described in this work generates features for both operational state identification and knowledge extraction. The goal of feature extraction is to map the original measurements into a feature space. The approach involves:

1. extraction of features to reduce dimensionality of the original signal and retain as much of the relevant information as possible;
2. filtering out noisy components.

For wavelet applications in chemical processes, two approaches have been proposed for interpreting dynamic trends. Bakshi and Stephanopoulos (1994) used wavelet decomposition of functions to find the points of inflexion in trends. Trends are then split into episode descriptions according to the identified inflexions. Therefore, this approach can be regarded as episode-based and wavelet decomposition is only used to find the inflexion points. Based on the wavelet packet transform, Dai et al (1995) developed a framework using time-frequency phase planes to represent and analyse dynamic trends. The trends are converted into the form of visual graphics that was helpful for human experts in decision making but not suitable for computer based decision support systems. The influence of noise is not considered in above work.

The wavelet based signal pre-processing approach in this work is different from previous work. The approach picks out the wavelet transform corresponding to the turning points of trends and uses this as a feature to define the trends. Such features capture the information about process changes such as sudden disturbances, as well as increases or decreases of variable values. The approach also takes the influence of noise into account.

The rest of this Chapter starts by introducing the relevant wavelet theory. Then the feature extraction approach and its implementation are described, followed by noise component removal and piece-wise processing as a basis for dimension reduction. Finally, feature extraction using wavelet and Fourier transforms are compared and some general observations are made.

3.2 Wavelet Theory

The following is only a brief introduction to wavelet theory: detailed treatments can be found in Daubechies (1992), Hernandez and Weiss (1996), and Chui (1992).

A wavelet is a function $\psi(t) \in L^2(R)$ such that:

$$\int_{-\infty}^{+\infty} \psi(t)dt = 0 \qquad (3-1)$$

A family of functions $\{\Psi_{a,b}(t)\}$ is generated by the basis function

$$\{\psi_{a,b}(t)\} = |a|^{-1/2}\,\psi(\frac{t-b}{a}) \qquad \textbf{(3 - 2)}$$

$$a,\ b \in R \quad \text{and} \quad a \neq 0$$

where a is a dilation parameter, and b a translation parameter. The forward wavelet transform involves computation of the inner products $\langle f(t), \psi_{a,b}(t)\rangle$ for all $a,\ b \in R$ and $a \neq 0$. The inner products are called the wavelet coefficients and are defined as:

$$\langle f(t), \psi_{a,b}(t)\rangle = |a|^{-1/2} \int_{-\infty}^{\infty} f(t)\psi^*(\frac{t-b}{a})dt \qquad \textbf{(3 - 3)}$$

where the asterisk denotes the complex conjugate, and the notation $\langle \bullet \rangle$ is used for the standard inner products evaluated according to

$$\langle f, g\rangle = \int_{-\infty}^{\infty} f(t)g^*(t)dt \qquad \textbf{(3 - 4)}$$

The function *f(t)* can then be reconstructed from the wavelet coefficients:

$$f(t) = c_\Psi^{-1} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \langle f(t), \psi_{a,b}(t)\rangle \psi_{a,b}(t)\frac{dadb}{a^2} \qquad \textbf{(3 - 5)}$$

To make the wavelets useful analytic functions, the basic wavelet must possess certain desirable properties. From the inversion formula given in Equation 3-5, the condition for a function ψ(t) having a Fourier transform $\hat{\psi}(\omega)$ to be a wavelet is

$$c_\Psi = \int_{-\infty}^{\infty} \frac{|\hat{\psi}(\omega)|^2}{|\omega|} d\omega < \infty \qquad \textbf{(3 - 6)}$$

This condition is called the admissibility condition (Grossmann et al, 1985). If the wavelet ψ(t) is an absolutely integrable function, as is usually the case, then the Fourier transform $\hat{\psi}(\omega)$ is continuous. If $\hat{\psi}(\omega)$ is continuous, c_ψ can be infinite only if

$$\hat{\psi}(0) = 0 \Leftrightarrow \int_{-\infty}^{\infty} \psi(t)dt = 0 \qquad \textbf{(3 - 7)}$$

A wavelet must therefore be an oscillatory function with zero means. Equation 3-6 also suggests that $|\hat{\psi}(\omega)|^2$ decays at least as $\frac{1}{|\omega|}$. Therefore, the wavelet ψ(t) decays at least

according to $\frac{1}{|t|^{1-\varepsilon}}$, for some $\varepsilon > 0$. In practice, much stricter decay conditions are imposed on $\psi(t)$.

Associated with each wavelet there are time and frequency localisation defined, respectively, by:

$$\sigma_t^2 = \int_{-\infty}^{\infty} (t - \bar{t})^2 |\psi(t)|^2 dt \qquad \textbf{(3 - 8a)}$$

$$\sigma_\omega^2 = \int_{-\infty}^{\infty} (\omega - \bar{\omega})^2 |\hat{\psi}(\omega)|^2 d\omega \qquad \textbf{(3 - 8b)}$$

Wavelets can broadly be classified as continuous and discrete. Given a basic wavelet function $\psi(t)$, the continuous wavelet transform (CWT) coefficients of a function *f(t)* are defined by:

$$CWT_f(a,b) = \langle f, \psi_{a,b} \rangle = |a|^{-1/2} \int_{-\infty}^{\infty} f(t) \psi^* \left(\frac{t-b}{a} \right) dt \qquad \textbf{(3 - 9)}$$

$$a,\ b \in R \quad \text{and} \quad a \neq 0$$

For signal processing, a discrete wavelet transform is often used. Where the scale-time parameters a and b are restricted to the following forms:

$$a = a_0^m \ , \quad b = n b_0 a_0^m \qquad \textbf{(3 - 10)}$$

$$\psi_{m,n}(t) = 2^{-m/2} \psi(2^{-m} t - n) \qquad \textbf{(3 - 11)}$$

The discrete wavelet transform (DWT) then has the form

$$DWT_f(m,n) = \langle f, \psi_{m,n} \rangle = a_0^{-m/2} \int_{-\infty}^{\infty} f(t) \psi(a_0^{-m} t - n b_0) dt \qquad \textbf{(3 - 12)}$$

$$m,\ n \in Z$$

where m, n are integers. Typically, $a_0 = 2$ and $b_0 = 1$ is used, although a_0 and b_0 need not be restricted to these values. Figure 3-1 shows a typical non-orthogonal but symmetric wavelet (Mexican hat) which is essentially the second derivative of a Gaussian function. This figure shows how the dilation parameter, m, and translation parameter, n, affect the wavelet function shape. For small values of the dilation parameter m, the

wavelet is narrow and tends to localise in a small window defined by a short-time Fourier transform.

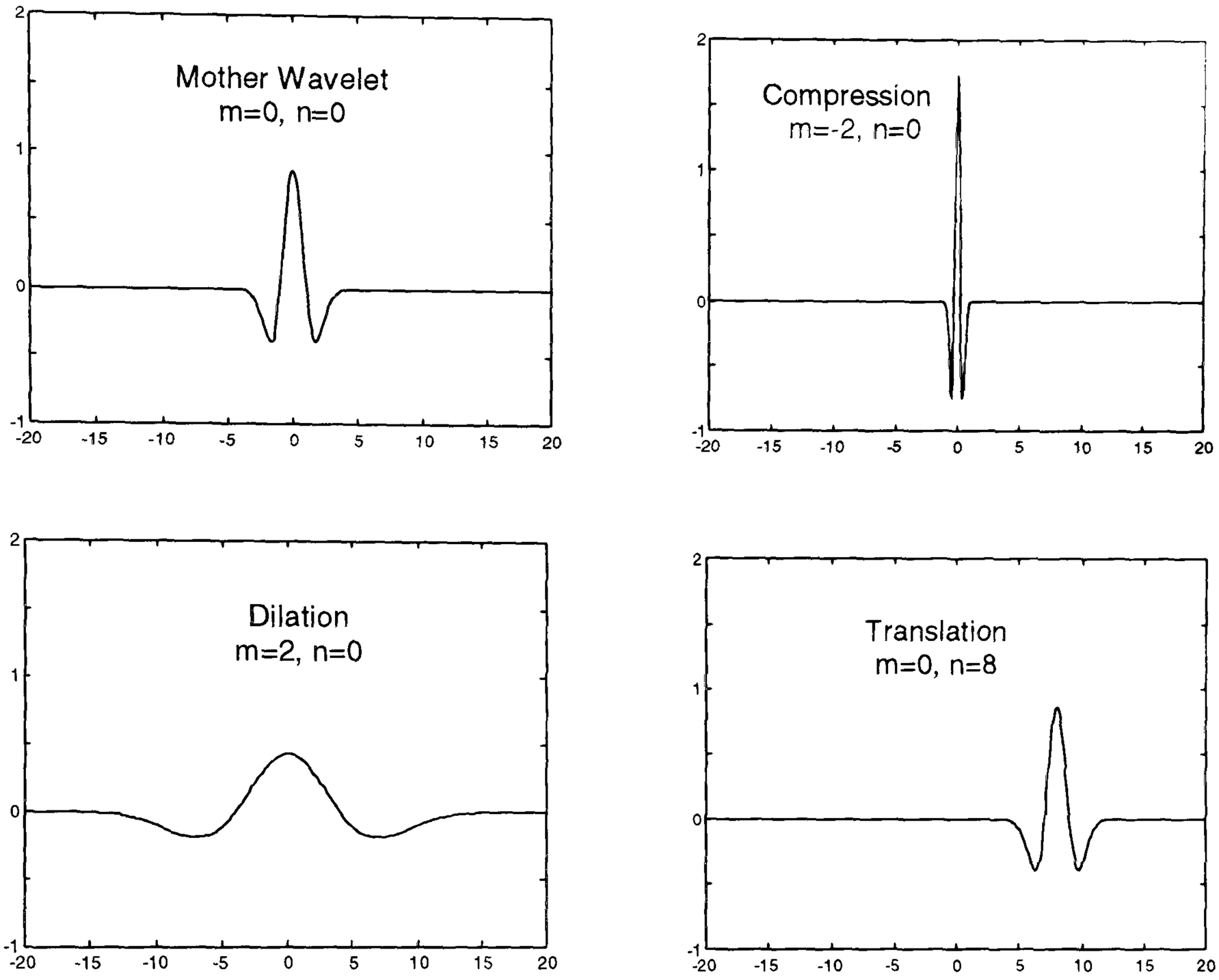


Figure 3 - 1 Mexican hat wavelet at different dilation and translation

3.3 General Description and Implementation for Data Pre-processing

The pre-processing approach, as shown in Figure 3-2, includes multi-scale wavelet decomposition of process residuals which are generated by comparing the on-line signals with a steady process trajectory. The residual generation procedure is not considered in detail. It is assumed that residuals have been appropriately generated in what follows. The decomposition divides the residual into two parts: a detailed signal and an approximate. Here, feature for pattern identification is extracted from the detailed signal while the approximate signal is used for knowledge extraction and representation by a fuzzy neural network.

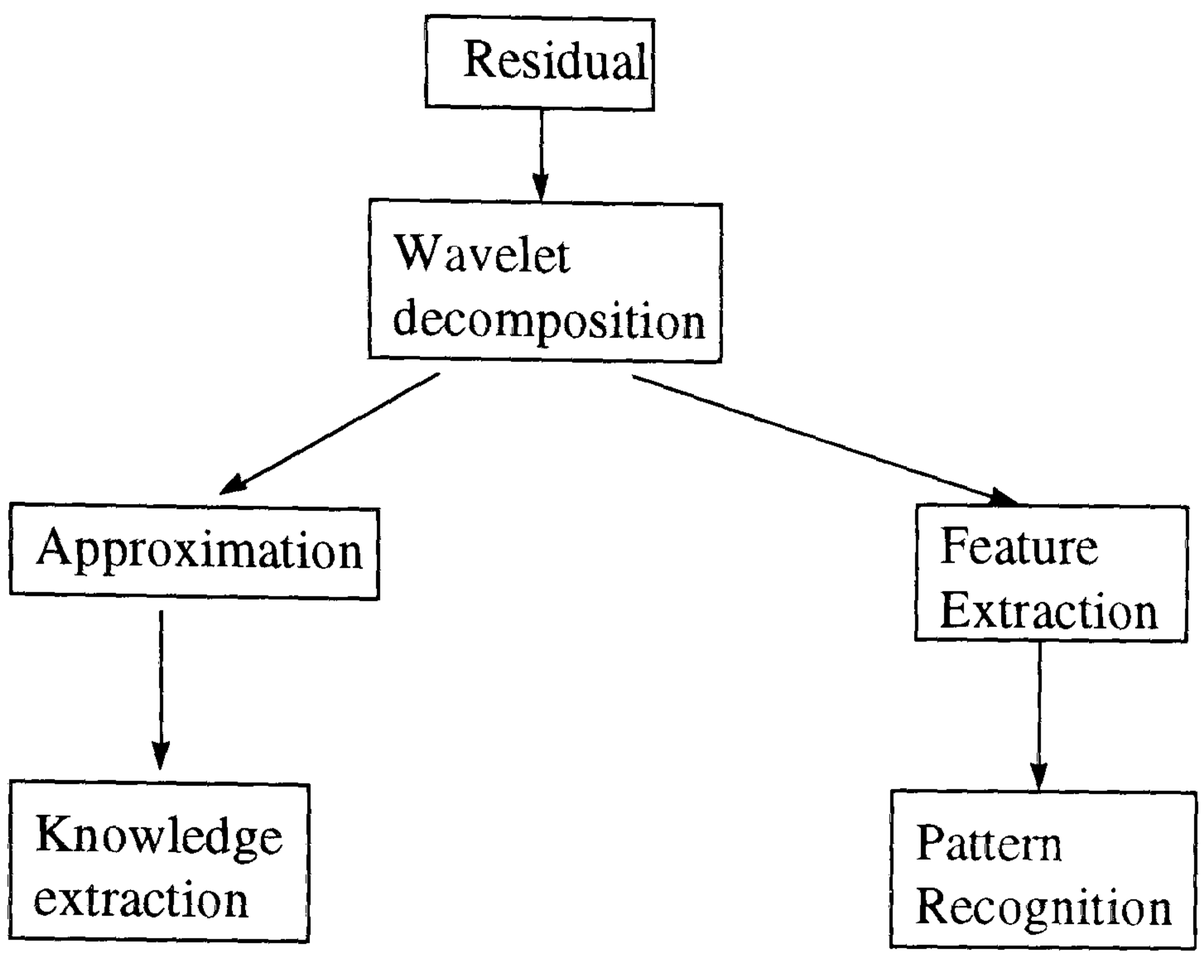


Figure 3 - 2 Wavelet based signal pre-processing

The residual generation process is essential not only for process behaviour evaluation but is also needed for wavelet transformation of signals. In chemical plant, the signal measurements have a physical meaning, e.g. reaction temperature of 450ºC. However, this cannot be directly used in a wavelet transform for two reasons:

1. the trajectory of process variables may not be constant, for example, during periodic operation;
2. the results of wavelet transform can be distorted when the process steady trajectory is included, even if it is constant. This has serious implications for the approximate part of the signal.

The second reason is best explained by an example. The temperature trend of an industrial FCC reactor is plotted in Figure 3-3(a) where the process steady trajectory is expected to be 498°C. The approximation of a four-scale multi-resolution analysis is shown in Figure 3-3(b). Obviously, the approximation cannot represent the main trend of the original signal which fluctuates between 497 and 499°C, while the approximation of the wavelet transform varies from about 200 to 550°C. There is no literature discussing this aspect of wavelet transforms. The effect is probably caused by 'end effects' because sudden changes occurred at the beginning and end of the approximation. However, the

approximation in Figure 3-3(b) still changes between 470 and 550°C even if the beginning and end of the approximation are ignored.

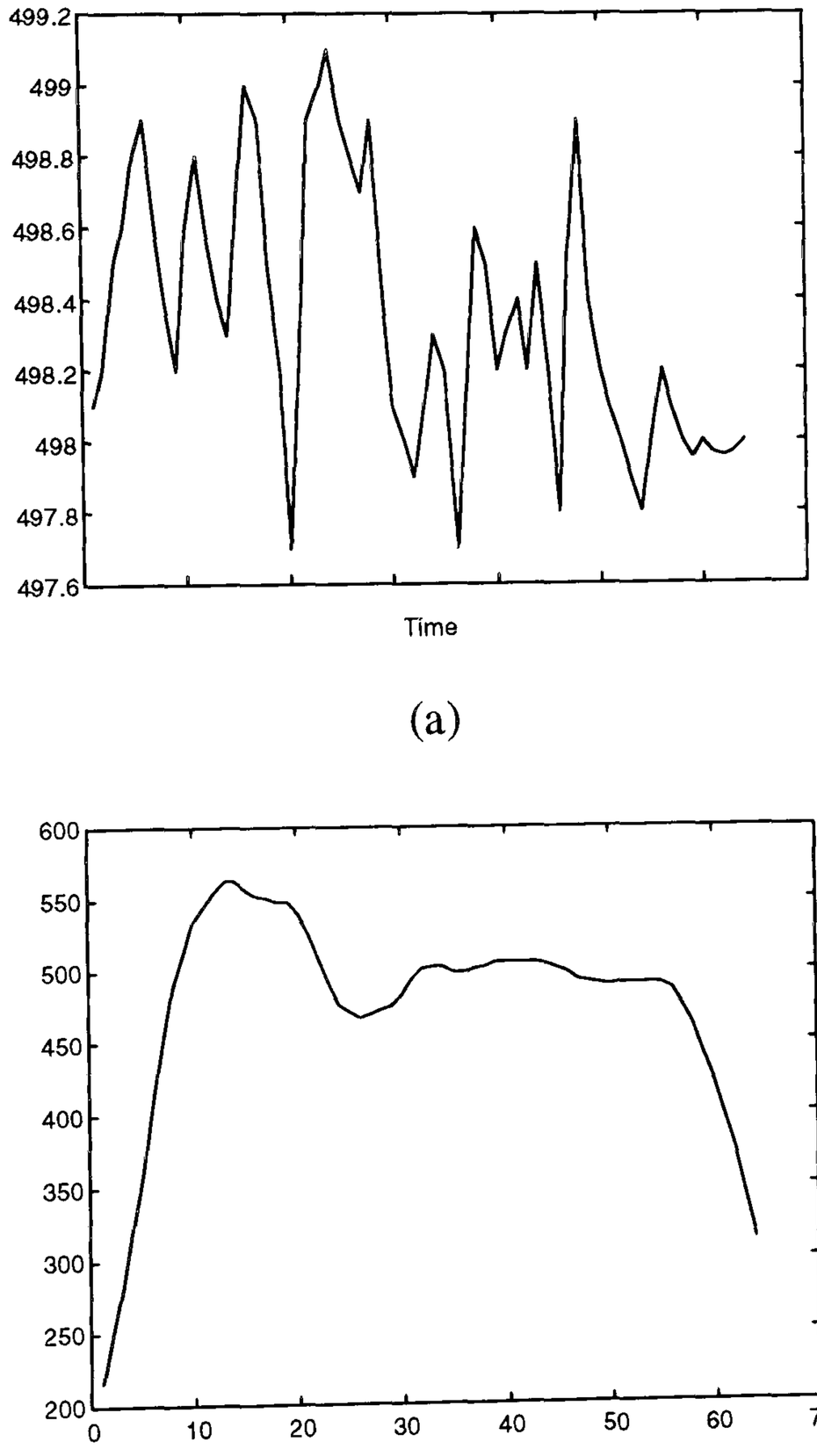


Figure 3 - 3 A process signal (a) and its low pass approximation (b)

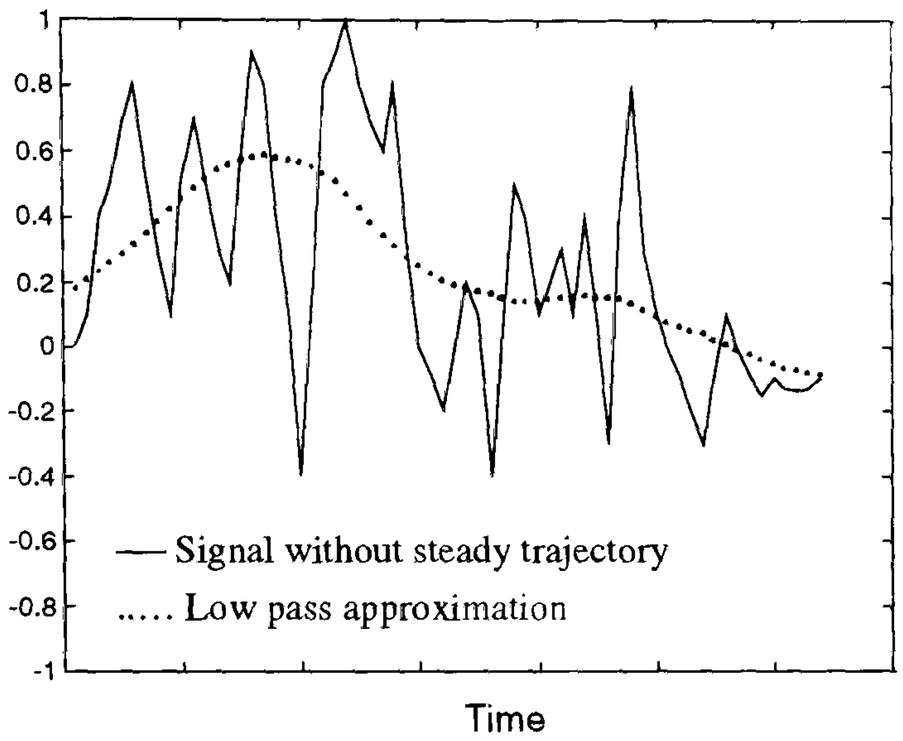
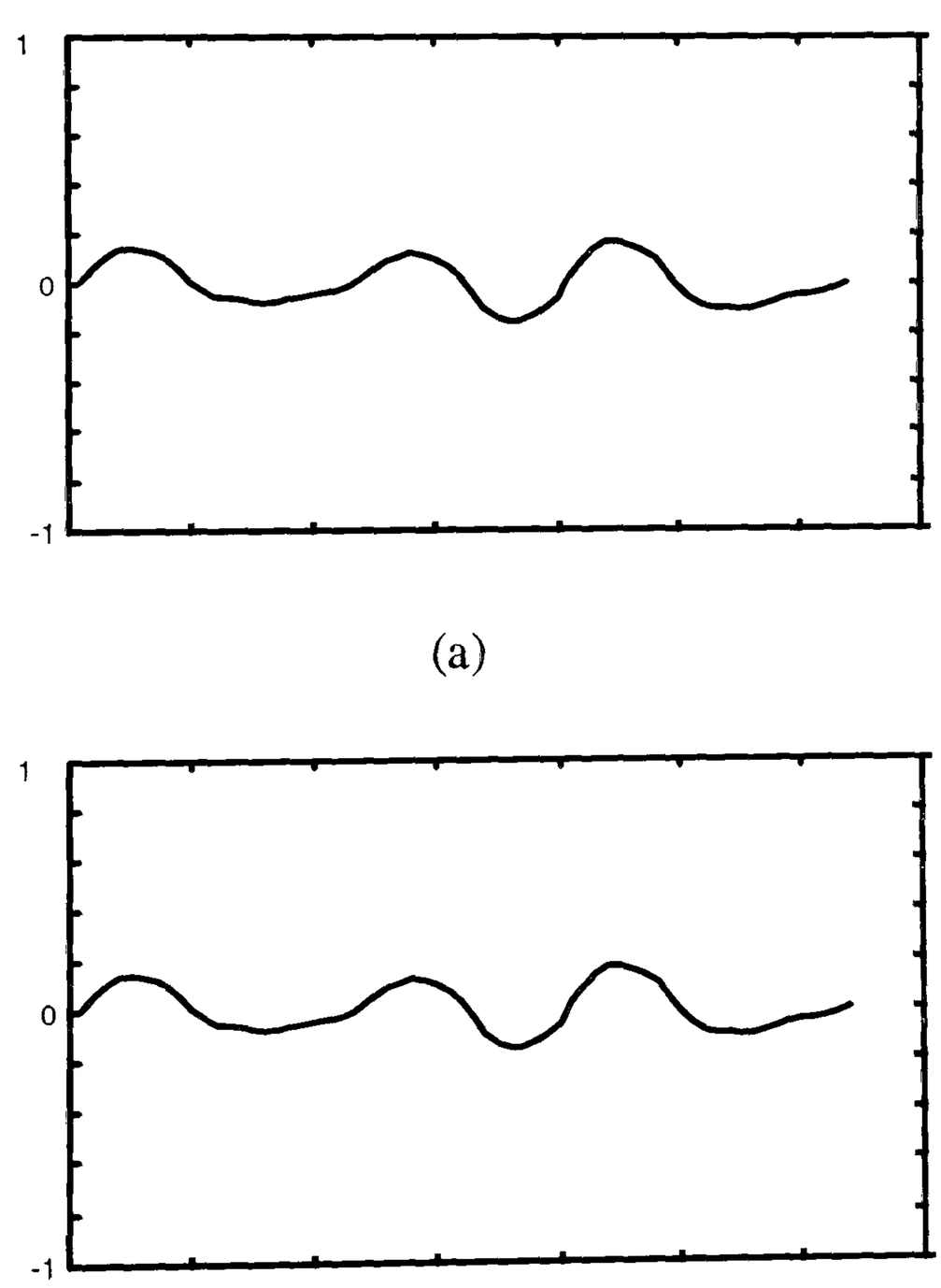


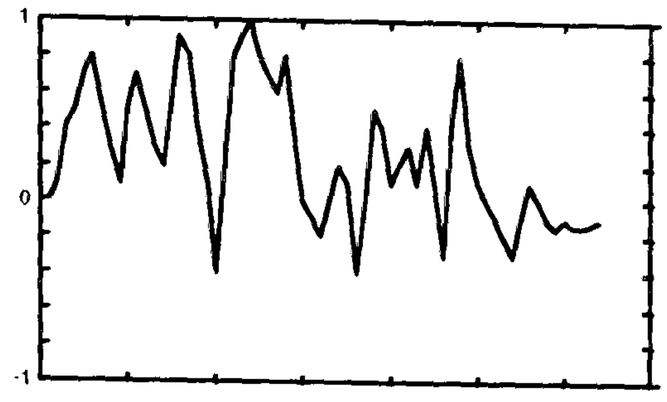
Figure 3 - 4 The signal after taking out steady state trajectory and its low pass approximation



(a)

(b)

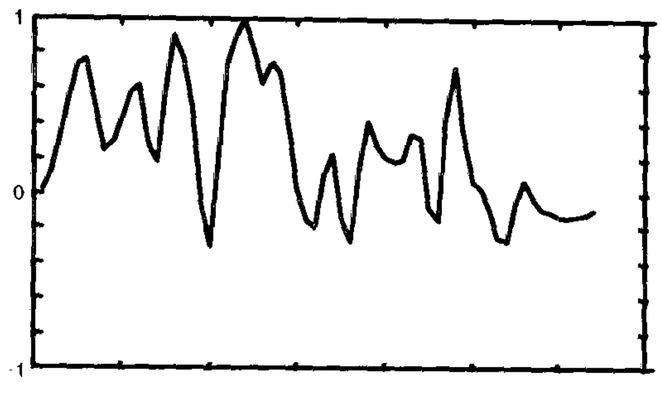
Figure 3 - 5 Detail signal of wavelet decomposition for measurement (a) and detail signal after filtering constant trajectory (b)

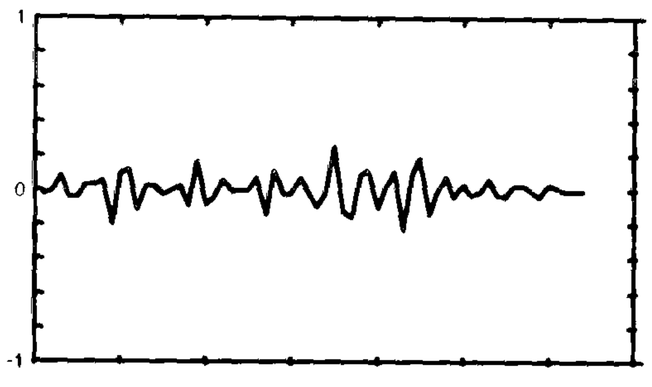


original signal

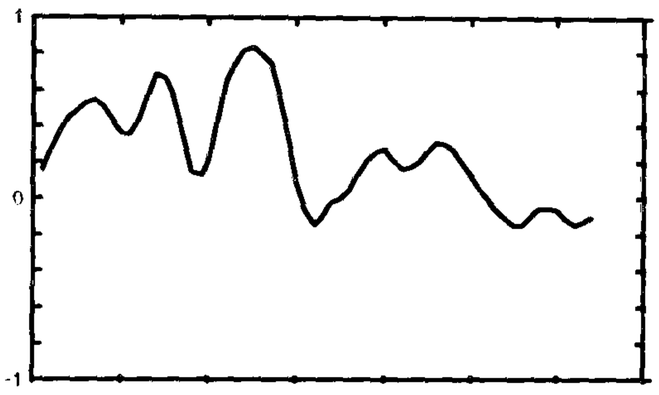
scale signal

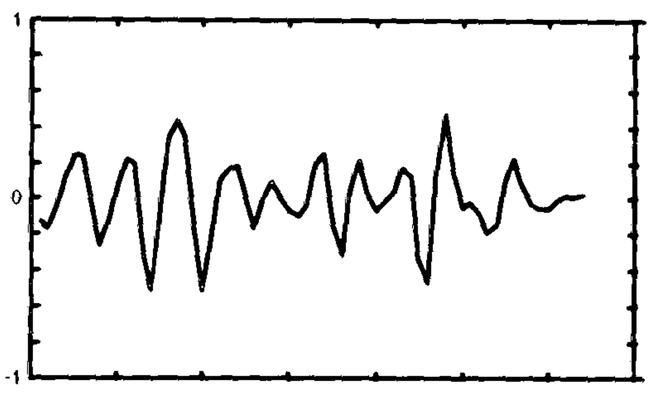
detail signal



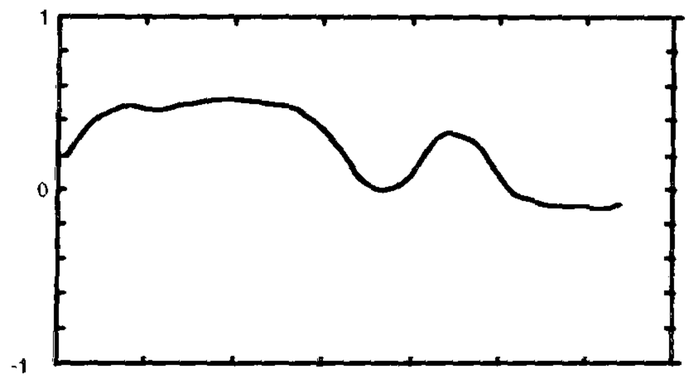


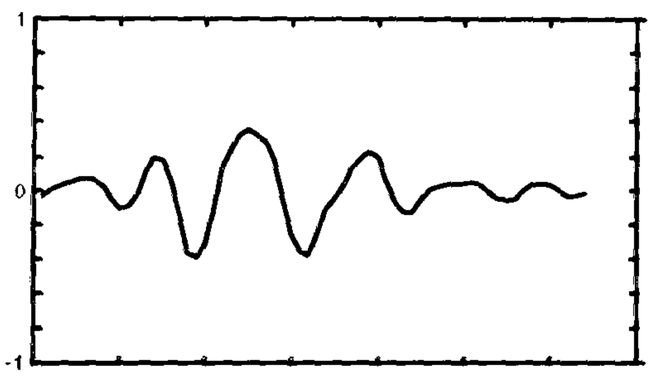
Scale 1



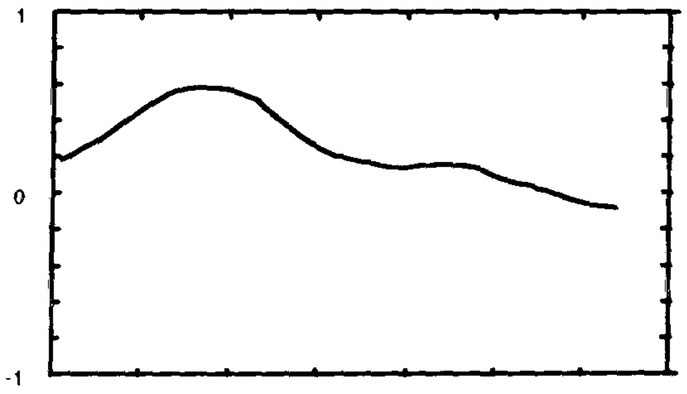


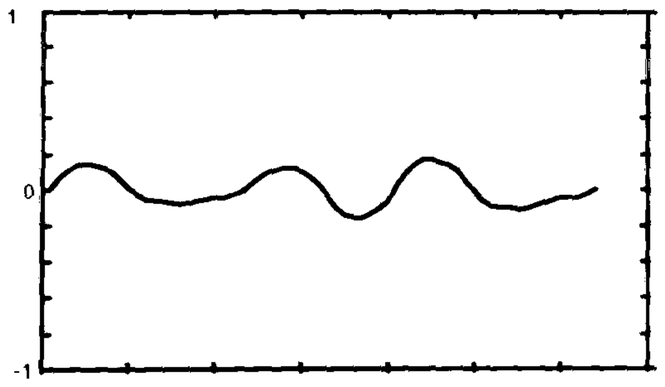
Scale 2





Scale 3





Scale 4

Figure 3 - 6 Multi-scale decomposition of a signal

Filtering the process steady trajectory, i.e. after subtracting the set point value 498°C from the measurement, gives an approximation of the wavelet transform which is much more reasonable than that derived using the raw measurement. The residual and its approximation obtained from the wavelet transform are plotted in Figure 3-4.

On the other hand, the detailed signal of wavelet decomposition is not influenced by a constant process steady trajectory. This is illustrated in Figure 3-5 where (a) the detailed signal of wavelet decomposition is obtained directly from process measurement with constant process steady trajectory and (b) the detailed signal of the wavelet decomposition is obtained by filtering out the trajectory. They are very close.

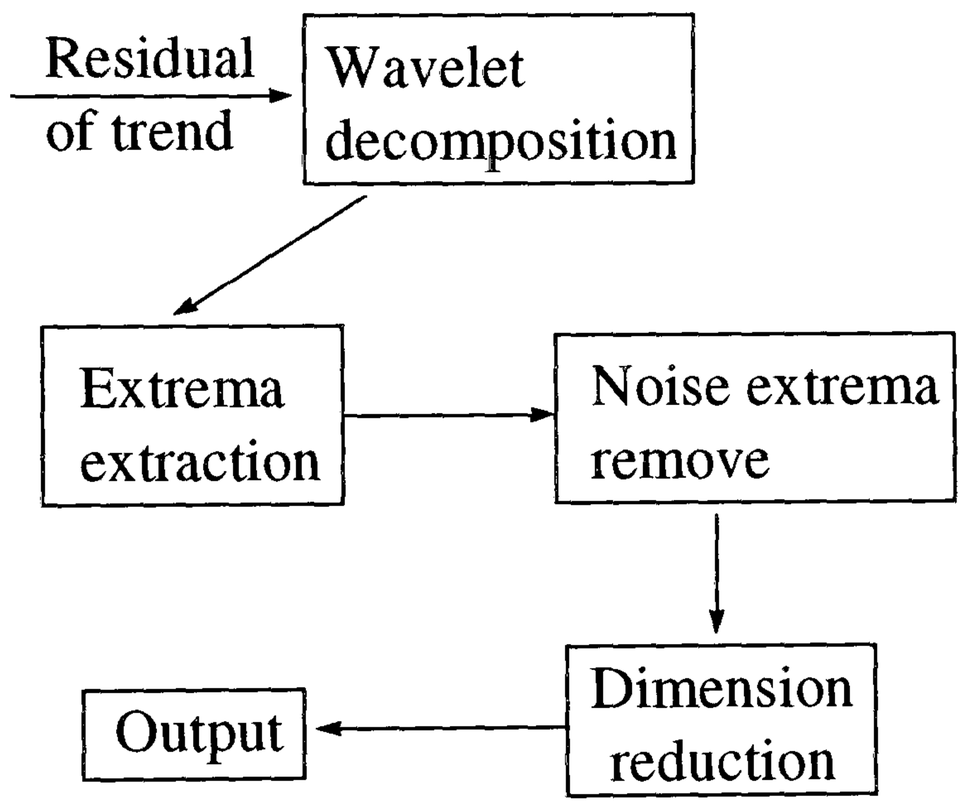


Figure 3 - 7 Details of feature extraction procedure

The residual is then decomposed into multi-scale components using wavelet multi-resolution analysis for pre-processing. Wavelet decomposition of the residual consists of two parts: detailed signals and approximations that are also called scale signals as shown in Figure 3-6. The features of the signal are obtained from the detailed signals, which is described in the next section. The high level (Scale 4 in Figure 3-6) approximation is the smoothed signal of the original measurement. The approximations are used as input to fuzzy neural networks for knowledge extraction and representation.

Feature extraction for pre-processing is illustrated in Figure 3-7, which consists of multi-scale wavelet decomposition, extrema extraction from the detail of the decomposition, noise extrema removal, and dimension reduction. These procedures are described below.

3.4 Feature Extraction

In this section, the approach using multi-scale wavelet decomposition for feature extraction to achieve reduction in dimensionality and noise filtering is described. However, the identification of process trend and features are discussed first.

3.4.1 Feature extraction and process trends

Feature extraction is basically a transformation of the measurement into data patterns, such that the space of the data pattern has a lower dimensionality than that of the measurements. An N-vector of measurements is transformed into an M-vector, $M < N$, of the features in pattern space. An important property of such a transformation is that it is information preserving. Thus, it removes redundant components while preserving, in some optimal sense, information that is crucial for pattern discrimination.

In general, if the transformation is restricted to a linear combination of measurements, then it is possible to find the optimal transformation in a least mean-squared error sense using a technique of feature selection based on an orthogonal expansion.

Let a signal $f(t)$ be represented as a point in a vector space, where the co-ordinate axes are a set of mutually orthogonal basis functions. The signal is therefore defined by its position in this space, and is represented by a set of coefficients, one for each basis function. The orthogonal expansion of a signal is to normalise the signal vector, defined by the following Equation

$$f(t) = \sum_{i=1}^{\infty} c_i \phi_i(t) \qquad \textbf{(3 - 13)}$$

where the c_i are random uncorrelated coefficients, and the $\phi_i(t)$ are the orthogonal basis functions, normalised such that

$$E\left[\phi_i(t)\phi_j(t)\right] = \begin{cases} 1 & \text{i} = \text{j} \\ 0 & \text{i} \neq \text{j} \end{cases}$$

where $E[\bullet]$ denotes the expected value.

Thus, the signal $f(t)$ is represented as a point in a finite space of n-dimensions and sampled by measuring the signal over some interval, and becomes a signal vector $F = (f_1, f_2, \cdots f_n)^t$. The orthogonal basis functions $\phi_i(t)$ are similarly sampled, so that the ith co-ordinate axis is a base vector $\Phi_i = (\phi_{i1}, \phi_{i2}, \cdots \phi_{in})^t$, and these vector are arranged as the columns of an orthogonal matrix $\Phi = (\Phi_1, \Phi_2, \cdots \Phi_n)$, normalised such that $\Phi^t\Phi = I$, where I is the identified matrix. With the coefficients represented by $c = (c_1, c_2, \cdots c_n)^t$, Equation 3-13 can be rewritten as the finite sum

$$F = \Phi c \qquad \textbf{(3 - 14)}$$

Then, the expansion is characterised by the coefficient **c**, and it is possible to calculate these coefficients from Equation 3-14 by rearranging it to give

$$c = \Phi^t F \qquad \textbf{(3 - 15)}$$

A Fourier series is a special case of the expansion that uses sinusoidal basis functions. Wavelet functions can be orthogonal, so are able to be used as basis functions for the expansion of signal and to extract features from the measurements. The wavelet transform allows the signal to be analysed in both time and frequency domains simultaneously. Wavelet transform analysis therefore is able to retain the transient parts of the signal in the transform values at the same relative position in time. In addition, selection of the wavelet used by the wavelet transform allows the user to tailor the analysis to specific features of the signal.

However, wavelet expansion is a regular sampling process that is computationally expensive when a direct wavelet transform is applied. Furthermore, using a regular sampling wavelet transform, the coefficients used as features of the signal do not significantly reduce the dimensionality. Here, a non-subsampled multi-scale wavelet transform is applied to avoid such limitations.

The process trend undoubtedly has an intuitive meaning relative to the change of process behaviour over time. A process trend is not just a smooth representation of a noise signal. Based on an explicit definition of state and trend in both continuous and discrete, qualitative state and trend are given as (Cheung and Stephanopoulos 1990)

Qualitative State (QS). Let $x{:}[a,b]\rightarrow R$ be a reasonable function. $QS(x,t)$, the qualitative state of x at $t \in [a,b]$, is defined as the triplet of qualitative values as follows:

$$QS(x,t)=\begin{cases}\text{undefined} & \text{if x is discontinuous at t}\\ \langle [x(t)],[\partial x(t)],[\partial\partial x(t)]\rangle & \text{otherwise}\end{cases}$$

where

$$[x(t]=\begin{cases}+ & \text{if } x(t)>0\\ 0 & \text{if } x(t)=0\\ - & \text{if } x(t)<0\end{cases}$$

$$[\partial x(t]=\begin{cases}+ & \text{if } x'(t)>0\\ 0 & \text{if } x'(t)=0\\ - & \text{if } x'(t)<0\end{cases}$$

$$[\partial\partial x(t]=\begin{cases}+ & \text{if } x''(t)>0\\ 0 & \text{if } x''(t)=0\\ - & \text{if } x''(t)<0\end{cases}$$

Qualitative Trend. The qualitative trend of a reasonable variable, $x{:}[a,b]\rightarrow R$, is the continuous sequence of qualitative states over $[a,b]$.

A function $x(t)$ is a reasonable variable when it satisfies the following conditions:

1. it is continuous over time interval $[a,b]$ but is allowed to have a finite number of discontinuities in the function value and/or first derivative.
2. $x'(t)$ and $x''(t)$ are continuous in (a,b), and their one-sided limits exist at a and b.
3. $x(t)$ has a finite number of extrema and inflexion points in $[a,b]$.

An episode is represented by a qualitative state and its time interval. Thus, the representation of a trend is used to find the combination of episodes of a trend. Generally, nine primitives, as shown in Figure 3-8, are used in the episode approach (Janusz and Venkatasubramanian 1991). Note that C and D are not primitives because they can be regarded as the combination of A, F and B, E. Therefore, this reduces the primitives to seven, as shown in Figure 3-9. The seven primitives can be used to describe any trend in the work described here.

A combination of episodes forms a trend over an interval and is described by a primitive and the associated time. Primitives are different for first and/or second order derivatives, so the distinguished points which are between episode segments are extrema together with inflexions which satisfy

$$\frac{\partial x}{\partial t} = 0 \quad \text{or} \quad \frac{\partial^2 x}{\partial t^2} = 0$$

Figure 3-10 gives three trends as an example. In this figure, trend 1 consists of the primitives c-d-b-a-c-d-b. The connection points of the primitives c-d, b-a, are maximum values and minimum respectively, and inflexion points are between d and b, a and c, and d and b. Trend 2 in Figure 3-10 is a similar case.

The task of identifying the episodes in a continuous signal is simply one of identifying the inflexions and/or extrema, i.e. irregularities in the signal, since these correspond to distinct points of the episode segments. For a discontinuous signal, such as obtained during batch or semi-batch operations, singularities must be included to represent the change between batches, e.g. from batch to semi-batch operation. This means that the irregularities and singularities of a signal contain the most important information about the trend. Using irregularities and singularities as representations of the feature therefore completely defines the characteristics of a signal. However, irregularities and singularities are strongly influenced by noise, as in the case of trend 3 in Figure 3-10, where a noise pulse is registered in primitives e and f, together with three singularities. Noise components must be identified and filtered from the feature representations of the trend, otherwise the representations will result in a misleading pattern classification. The issue

of feature extraction then becomes the task of finding the irregularities and singularities. and distinguishing the trend from singularities due to noise

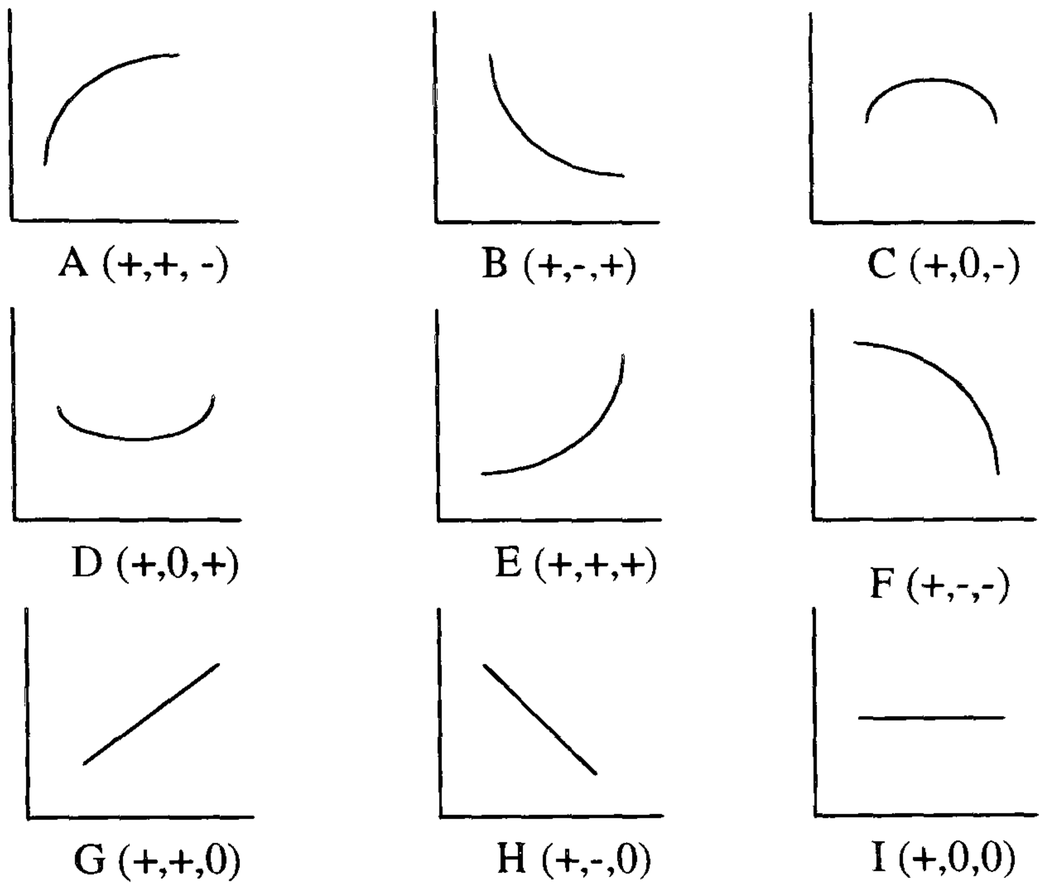


Figure 3 - 8 Primitives used in episode approach (Janusz and Venkatasubramanian 1991)

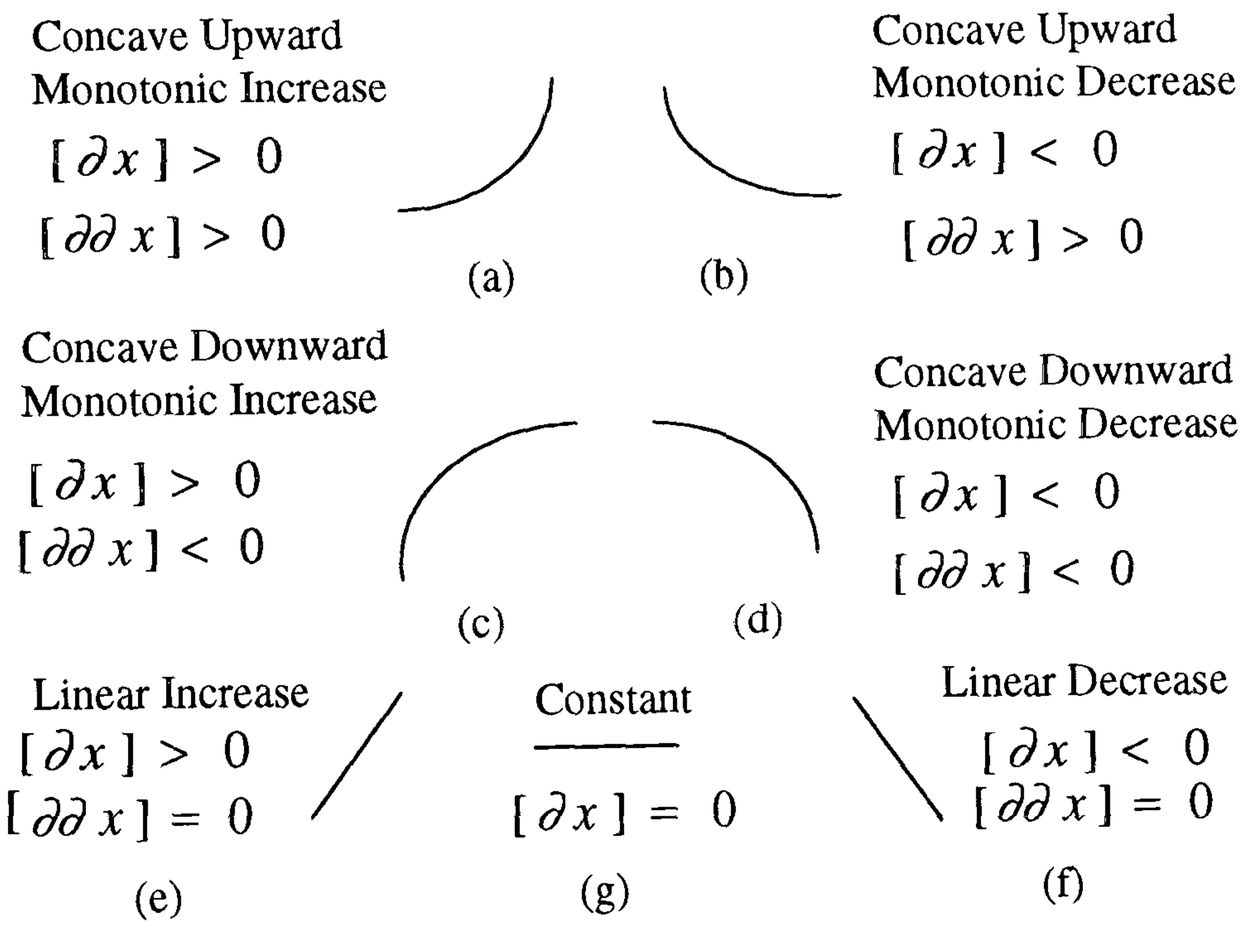


Figure 3 - 9 Seven primitives episode representation (Cheung and Stephanopoulos 1990)

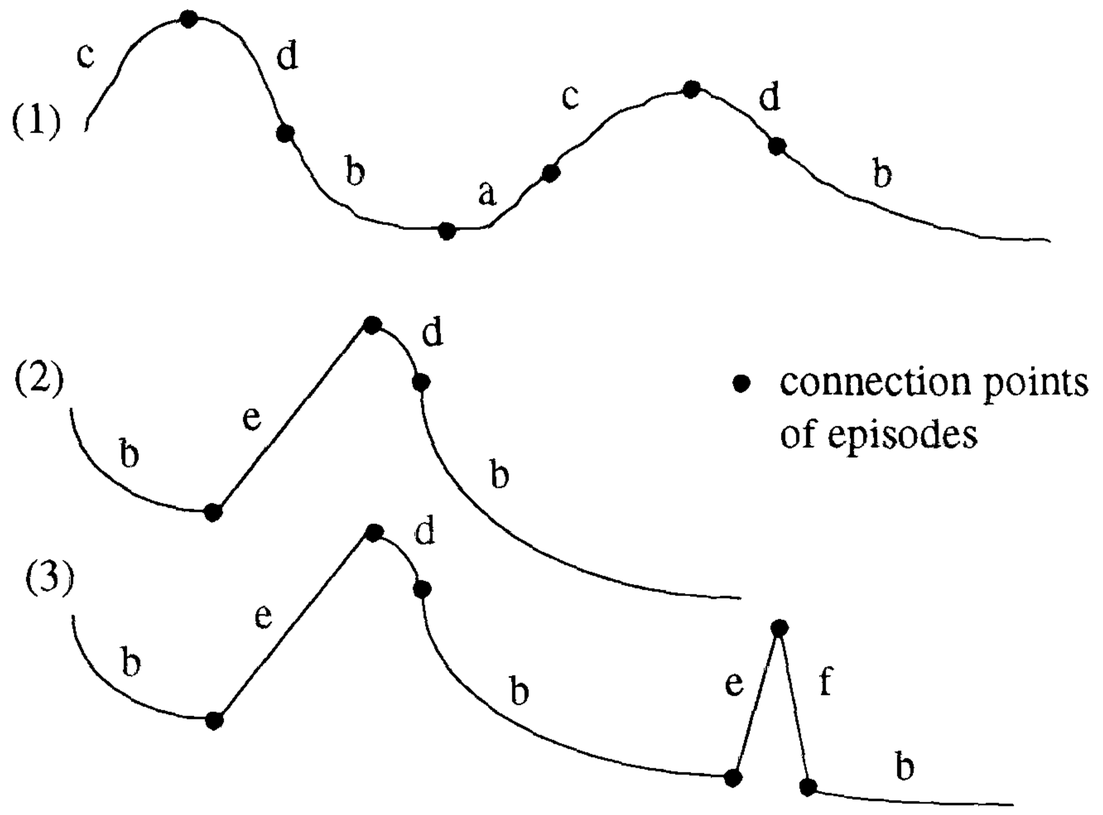


Figure 3 - 10 Singularities as connection points of episode segments

3.4.2 Wavelet for irregularity and singularity detection

Mathematically, the local irregularity and singularity of a function is often measured with Lipschitz exponents, defined as

For $0 < a < 1$, Lipschitz space is the set of all $f \in L^{\infty}(R)$ such that

$$\sup_{x \in R} |f(t+h) - f(t)| \le C|h|^{\alpha}$$

where a is the Lipschitz exponent.

In practice, the relationship between the Lipschitz exponent and the irregularity and singularity does not provide a simple and direct way of detecting and characterising the irregularities and singularities of a signal. The wavelet transform extrema representation is an efficient approach for studying irregular and singular structures of signals. Several studies have been reported on this topic (Mallat 1991, Mallat and Hwang 1992, Mallat and Zhong 1992, Berman and Baras 1993, Cvetkovic and Vetterli 1995).

The wavelet transform of a process signal represents the detailed signal at the appropriate scale, and corresponds to the difference between two scale signals. Selecting a special wavelet or filter bank, the extrema of a wavelet transform indicate the location of changes, such as turning points from increasing to decreasing and operation from one batch to another batch. The extrema in the wavelet analysis correspond to the location and character of the irregularity and singularity points of the smoothed signal. The

extrema from wavelet analysis are used as features to represent the trends and capture the most important information about them.

Mallat and Hwang (1992) and Mallat and Zhong (1992) suggest using a wavelet that is the first derivative of a scale function $\phi(t)$

$$\psi(t) = \frac{d\varphi(t)}{dt} \qquad \textbf{(3 - 16)}$$

For this, a Block spline (B-spline) wavelet and scaling function are recommended. They construct a class of one dimensional wavelets for detection of the extrema

$$|H(\omega)|^2 + G(\omega)K(\omega) = 1 \qquad \textbf{(3 - 17)}$$

Although these representations stem from the underlying continuous-time theory, implementation takes place in the discrete-time domain. Berman and Baras (1993) represent it for purely discrete-time, showing that the extrema of a wavelet transform provide a stable representation of finite length discrete-time signals. Cvetkovic and Vetterli (1995), using filter bank tools, develop a discrete-time framework for the extrema representation of a wavelet transform. They design a non-subsampled multi-resolution analysis filter bank to implement the wavelet transform for a representation.

The non-subsampled multi-resolution analysis is used here to detect irregularities and singularities of signal. An octave band non-subsampled filter bank with analysis filters $H_0(z)$ and $H_1(z)$ is shown in Figure 3-11. In this method, the wavelet transform refers to the bounded linear operators $W_j: l^2(Z) \rightarrow l^2(Z)$, j = 1, 2, .. J+1. The W_j are the convolution operators with the impulse responses of the filters:

$$V_1(z) = H_1(z)$$

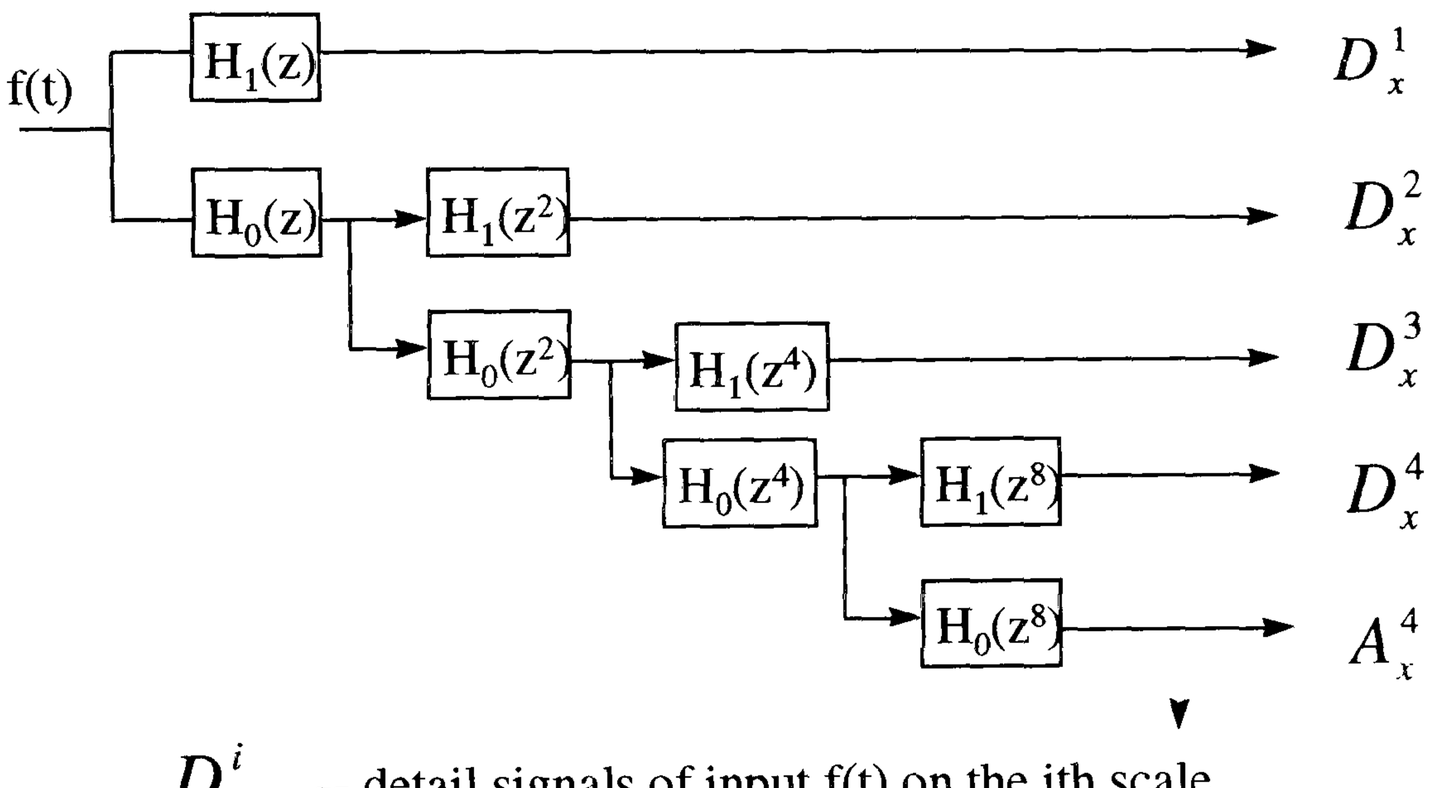
$$V_2(z) = H_0 H_1(z^2)$$

• • • • • • • •

$$V_J(z) = H_0(z)\cdots H_0(z^{2^{J-2}})H_1(z^{2^{J-1}})$$

$$V_{J0}(z) = H_0(z)\cdots H_0(z^{2^{J-2}})H_0(z^{2^{J-1}})$$

A finite impulse response (FIR) wavelet filter is recommended for use with the above operation to represent extrema. Such a filter has a sequence $\{\alpha_k : k \in Z\}$ with only K non-zero terms. A typical example of an FIR filter is the Haar wavelet, which has only two non-zero coefficients. Daubechies's wavelets (Daubechies 1992) are FIR filters and smoother than the Haar wavelet. Daubechies' wavelets having more coefficients result in a greater level of smoothness with the higher moments vanishing. They are also computationally less expensive because of being constructed by convolution of the filter.



D_x^i -- detail signals of input f(t) on the ith scale

A_x^k -- approximation signal of input f(t) on the k scale

H_0, H_1 -- analysis of low-pass and high-pass filters

Figure 3 - 11 An octave band non-subsampled filter bank

The Daubechies' scale and wavelet function are expressed as

$$\phi(t) = \sum_k h(k)\phi(2t-k) \qquad \textbf{(3 - 18)}$$

$$\psi(t) = \sum_k g(k)\phi(2t-k) \qquad \textbf{(3 - 19)}$$

where $\{h(k)\}$ is the low-pass filter coefficient and $\{g(k)\}$ the band-pass filter coefficient.

A wavelet has L vanishing moments if $\int_{-\infty}^{\infty} t^l \psi(t)dt = 0$ for l = 0, 1, ... M. Daubechies wavelets have maximum numbers of vanishing moments. Using wavelet with a larger number of vanishing moments has the advantage of being able to measure the Lipschitz regularity up to a higher order, which is of benefit in filtering out noisy components. However, the number of maxima for a given scale often increases linearly with the number of moments of the wavelet, which makes computation more expensive and increases the difficulty of reducing the dimensionality. In order to minimise computational effort, it is necessary to have the minimum number of maxima to detect interesting irregular and singular behaviour features of the signal. This means choosing a wavelet with as few vanishing moments as possible, but with sufficient to detect the Lipschitz exponents for the highest order of interest.

In this work, the eight-coefficient "Least-Asymmetric" Daubechies wavelet is used as a filter. The scale and wavelet function of the eight-coefficient "Least-Asymmetric" filter is illustrated in Figure 3-12.

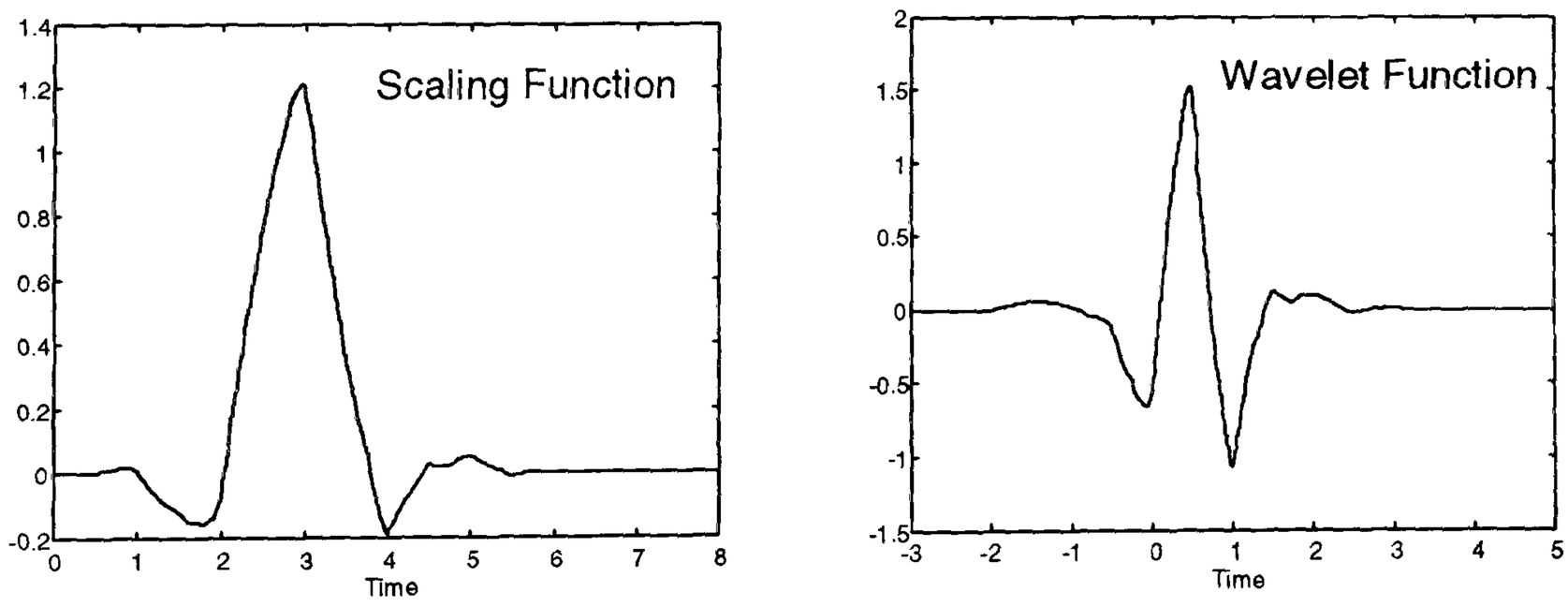
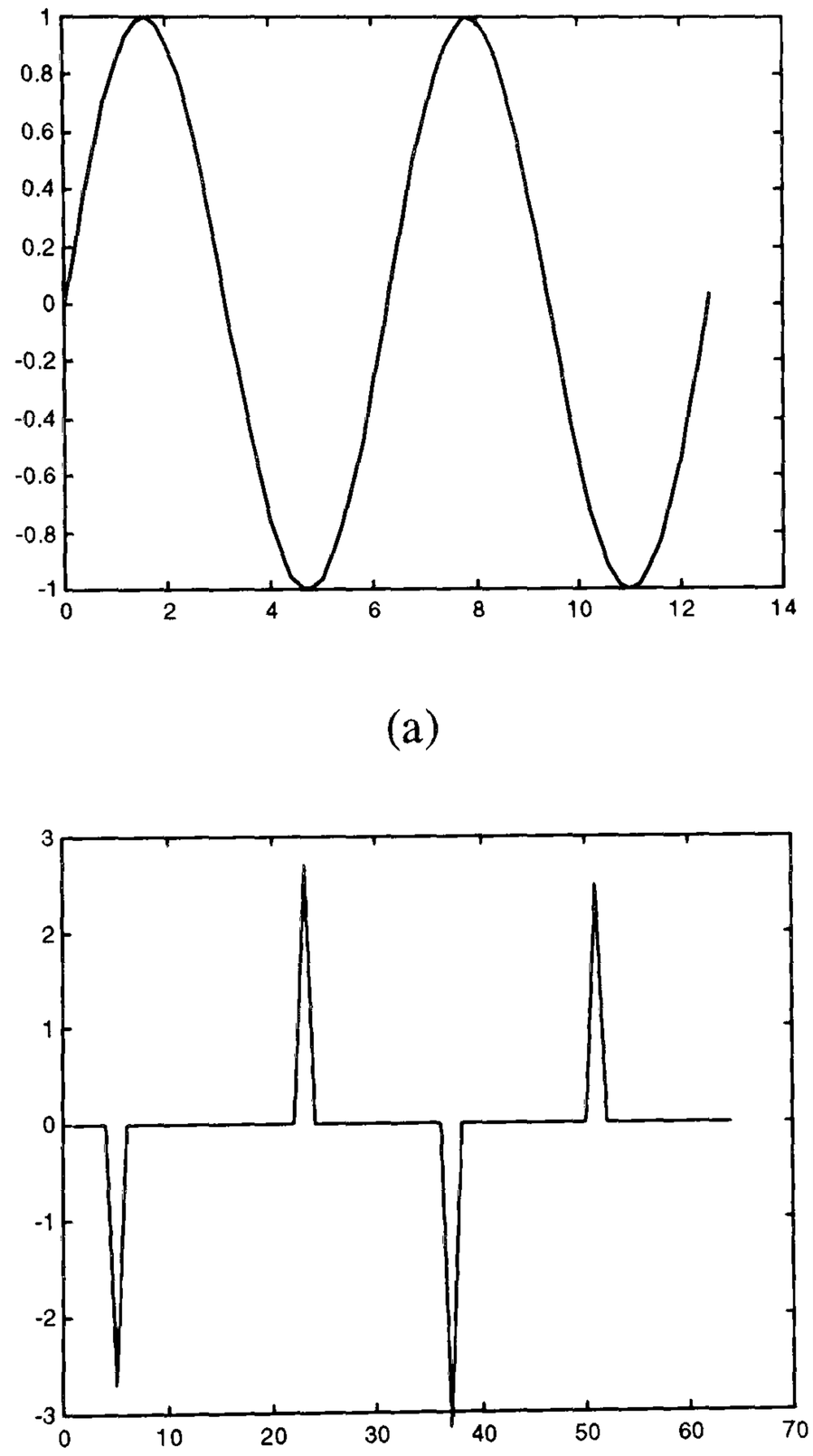


Figure 3 - 12 The "Least-Asymmetric" scale function and wavelet function

A signal such as $f(t)=sin(t)$ has extrema identified by wavelet analysis using a non-subsampled filter bank with Daubechies eight coefficient least asymmetry wavelet as illustrated in Figure 3-13. This shows that extrema correspond to the irregularities of signal. The shape of the corresponding extrema of wavelet analysis can be a maximum or minimum for the same irregularity of the signal, depending on the wavelet used. In

Figure 3-13(b), a Daubechies eight coefficient wavelet is used as the filter, and the first irregularity of the signal in Figure 3-13(a) corresponds to a minimum in the wavelet analysis, while in Figure 3-14 it becomes a maximum because a different wavelet is employed. The former approach is used in this work.



(a)

(b)

Figure 3 - 13 Signal (a) and its extrema (b) of wavelet analysis with Daubechies eight-coefficient wavelet

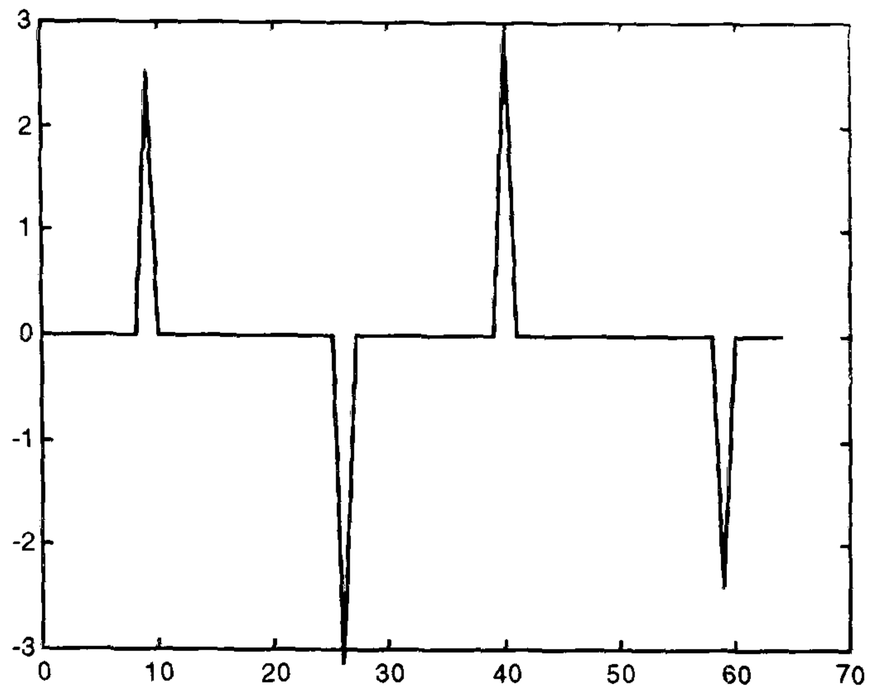


Figure 3 - 14 Extrema of signal in Figure 3-13 (a) obtained by wavelet analysis with Daubechies ten-coefficient wavelet

3.4.3 Noise extrema removal and data compression

The irregularities and singularities of a signal are strongly influenced by noise. It is necessary to filter out noise extrema from the wavelet transform before using them as input in pattern identification.

The classical technique for removing noise from a signal is to filter it. Part of the noise is removed but it may also smooth the signal irregularities and singularities at the same time. Mallat and Hwang (1992), and Mallat and Zhong (1992), have developed a noise extrema evaluation technique based on the relationship between the Lipschitz exponent and characteristics of singularities. The technique is applied here to remove the noise extrema of wavelet analysis. White and high frequency noise are considered.

Let $n(t)$ be a white noise random process and $Wn(s, t)$ the wavelet transform. Grossmann (1986) has showed that the decay of the expected value of the wavelet transform $E(|Wn(s, t)|^2)$ is proportional to $\frac{1}{s}$ i.e.

$$|Wn(s,t)|^2 = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} n(u)n(v)\psi_s(t-u)\psi_s(t-v)dudv$$

Since $n(t)$ is white noise, $E(n(u)n(v)) = \delta(u-v)$, so

$$E(|Wn(s,t)|^2) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} \delta(u-v)\psi_s(t-u)\psi_s(t-v)dudv$$

and so

$$E(|Wn(s,t)|^2) = \frac{\|\psi\|^2}{s}$$

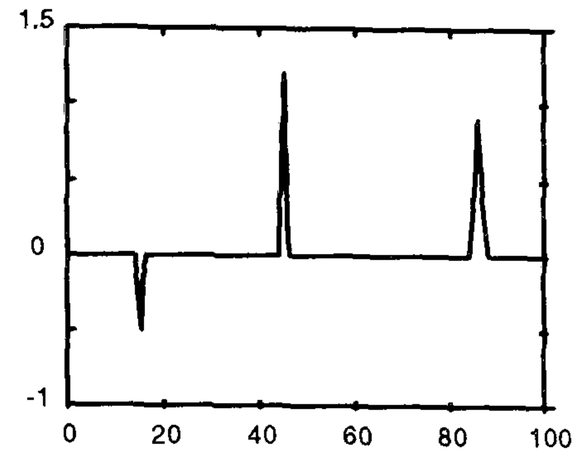
For a given scale s, the wavelet transform $Wn(s, t)$ is a random process in t. If white noise $n(t)$ is a Gaussian white noise then $Wn(s, t)$ is also a Gaussian process. Using this property, the density of local extrema of the wavelet transform is

$$d_s = \lambda \frac{\|\psi^{(2)}\|}{s\pi\|\psi^{(1)}\|}$$

where $\psi^{(n)}(t)$ is the nth derivative of wavelet $\psi(t)$ and λ a constant between 0.5 and 1. The density of local extrema is inversely proportional to the scale s. The realisation of a white noise is a distribution which is almost everywhere singular. This results in a Lipschitz $\alpha = -0.5 + \varepsilon$, for any $\varepsilon > 0$. However, evaluating the Lipschitz exponent of a function along scales is computationally expensive. According to the relationship between the extrema of wavelet analysis, the noise component can be identified based on the relationship between the Lipschitz exponent and the character of a function.

What follow is a summary of the relationship between an irregularity or singularity and the Lipschitz exponent:

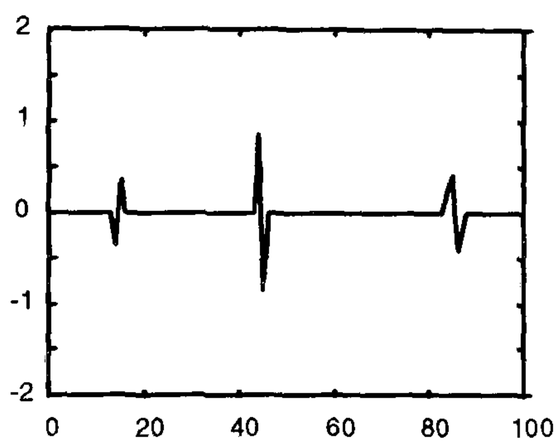
1. Lipschitz $\alpha = 1$, $f(t)$ is differentiable at t_0
2. Lipschitz $a > 0$, $f(t)$ is continuous at t_0
3. Lipschitz $a = 0$, $f(t)$ is discontinuous at t_0
4. Lipschitz $a = -1$, $f(t)$ is Dirac function at t_0
5. Lipschitz $a = -0.5 + \varepsilon$ for any $\varepsilon > 0$, $f(t)$ is white noise

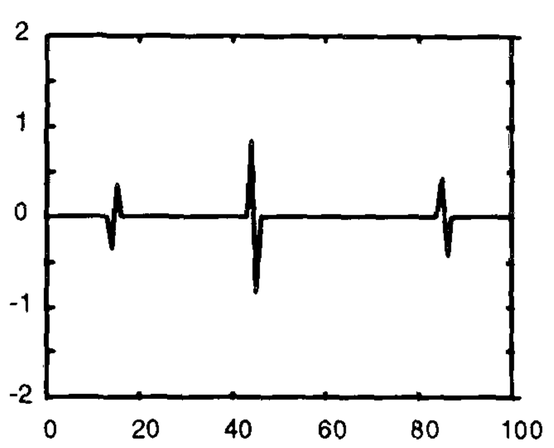


Noisy signal

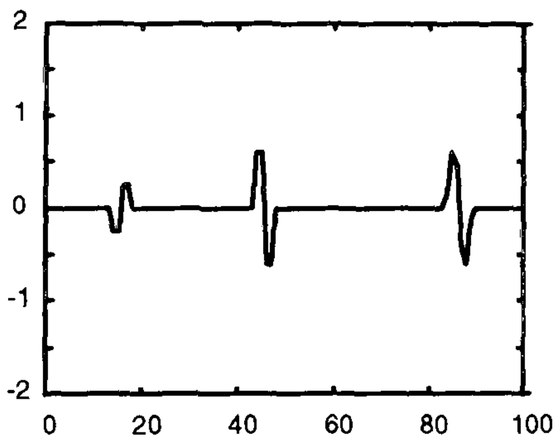
Wavelet analysis

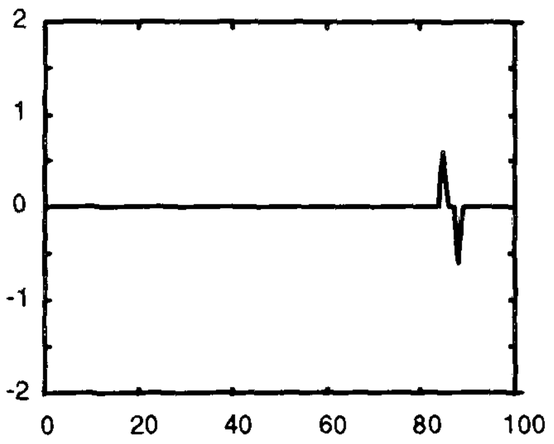
Extrema of wavelet analysis



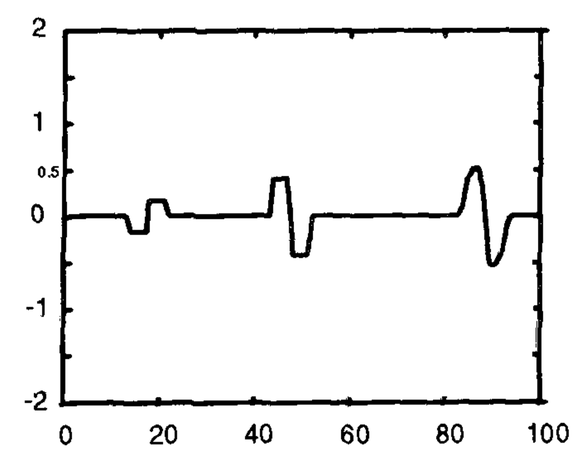


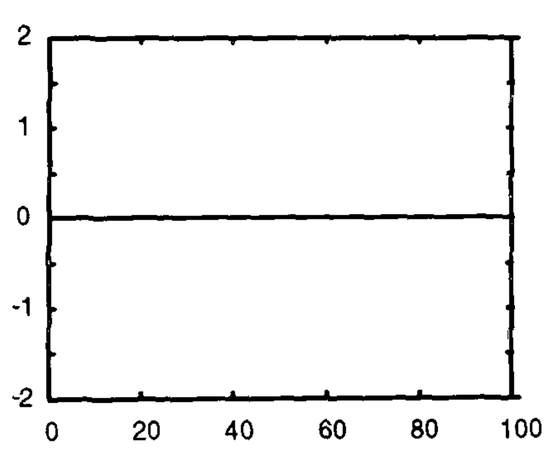
Scale 1





Scale 2





Scale 3

Figure 3 - 15 Noise signals, their wavelet transform and extrema of the wavelet transform

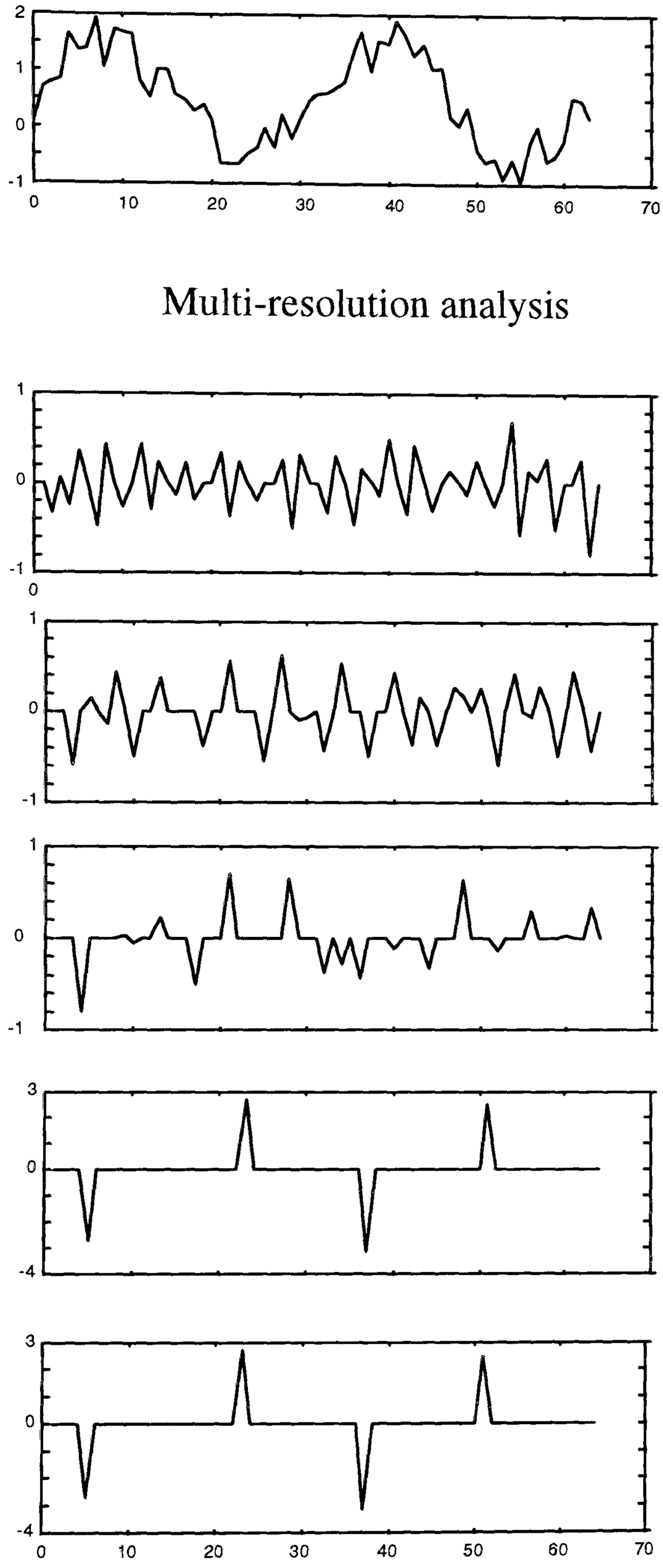


Figure 3 - 16 Using a sine wave with noise and the result of multi-resolution analysis to illustrate how many scales are needed for decomposition

Based on the definition of the Lipschitz exponent and the extrema involved in wavelet analysis, the following relationship is satisfied

$$\left|WT_f(s,t)\right| \le As^{\alpha} \qquad \textbf{(3 - 20)}$$

which is equivalent to

$$\log\left|WT_f(s,t)\right| \le \log(A) + \alpha\log(s) \qquad \textbf{(3 - 21)}$$

In Equation 3-20, $\left|WT_f(s,t)\right| \propto s^{\alpha}$ since A is constant. For a real trend, $\left|WT_f(s,t)\right|$ is increased as scale *s* increases because the Lipschitz exponent a > 0. While $\left|WT_f(s,t)\right|$ of noise decreases as the scale increases since a < 0. Hence, noise extrema die out as the scale is increased.

This makes it possible to identify extrema and distinguish them from those produced by real trend based on the fact that the noise extrema:

1. have amplitudes which decrease on average when the scale increases;
2. do not propagate to large scales.

Figure 3-15 illustrates the use of such criteria. In Figure 3-15, three different pulses are generated. The wavelet multi-resolution analyses are given on the left hand side, and extrema of wavelet analysis are on the right. Clearly, the extrema decrease and then disappear as the scale increases.

Based on such criteria, noise extrema can be removed using the following steps:

Step 1: find extrema positions and their signs, then store and flag them with largest scale detail results, assumed to be 4;

Step 2: select an extremum on the 4th scale according to the position, find the corresponding extrema with the same sign in the 3rd scale within a range. For example, if an extremum position in the 4th scale is `i`, the possible corresponding extremum must be searched within the region of `[i-2, i+2]` on the 3rd scale;

Step 3: if the same sign extremum is not found in the 3rd scale, then delete the corresponding extremum in the 4th scale and return to Step 2;

Step 4: if a same sign extremum is found on the 3rd scale (if more than one then the largest is selected), then their absolute values are compared. If the extremum value on the 4th scale is less than that in the 3rd scale. delete the corresponding extremum in the 4th scale and return to Step 2, otherwise move to Step 5;

Step 5: keep the extremum from the 3rd scale, and look for the extremum in the 2nd in a similar way as described in Step 2 to Step 4. After that, move to the extremum on the first scale;

Step 6: move to the next extremum in the 4th scale, and repeat Steps 2 to 5.

In the above noise removal algorithm, the largest scale for the decomposition has to be decided in advance. Theoretically, it should be determined according to the noise characteristics. The problem is that the characteristics of the noise are generally unknown for process measurements. One way of determining the largest scale of decomposition is to compare extrema representations for two neighbourhood scales. If they have the same non-zero item, it means no more further decomposition is needed. Figure 3-16 illustrates this method by using a signal consisting of a sine wave and white noise, and its multi-resolution wavelet analysis. Noise components are reduced and then disappear as the scale increases. Scales 4 and 5 have the same non-zero items for the extrema representations, so four scales are required for decomposition.

In fact, noise components can be filtered out based on the above criteria even if there is no similar non-zero item for two neighbouring scales because the noise components have no corresponding responses along scales. This can be seen in Figures 3-17 and 3-18. In Figure 3-17, a process trend is decomposed into four scales and Figure 3-18 shows the result obtained using the above noise components removal algorithm. It can be seen that noise components in scale 4 are identified, and not contain in feature representation as shown Figure 3-18. Generally, four or more scale decomposition is recommended for process measurements.

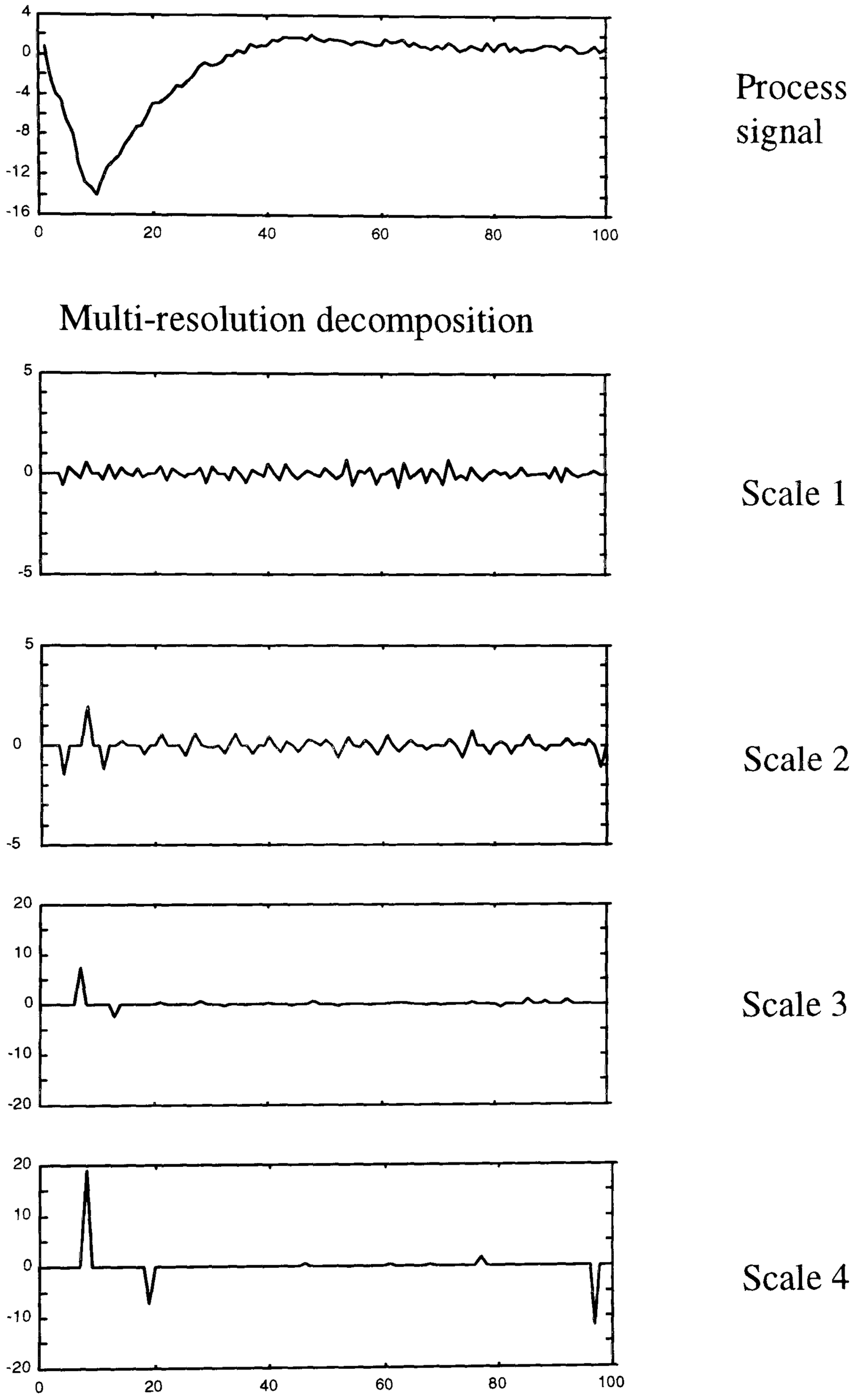


Figure 3 - 17 A process signal and its multi-resolution analysis

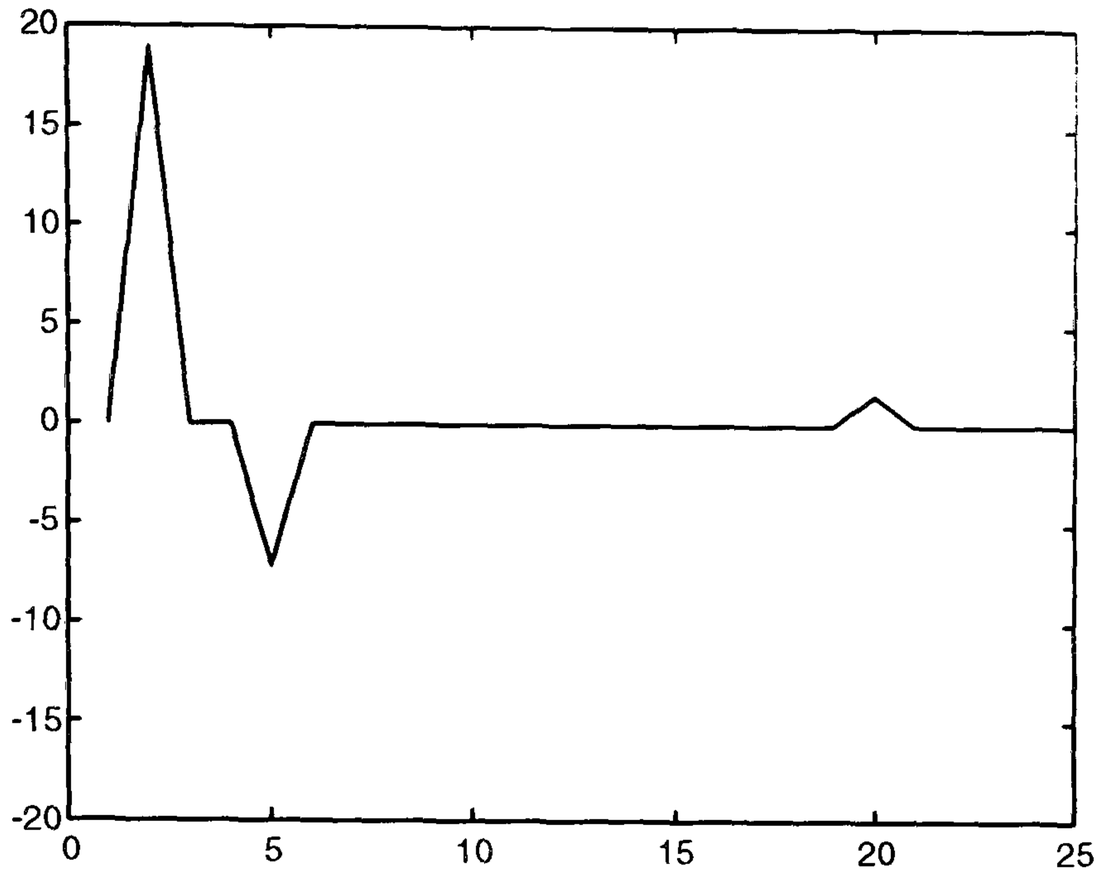


Figure 3 - 18 Features after noise components removal for the process signal in Figure 3-17

Two observations are appropriate based on these results:

1. wavelet analysis extrema of real trends remain steady whatever scales of decomposition are used, so the representation is steady;
2. the locations of extrema are shifted with time as the scale increases. In Figure 3-16, the extrema representation in scale 4 is (x_5, x_{23}, x_{37}, x_{53}), while in scale 5, it becomes (x_7, x_{22}, x_{38}, x_{54}), where x_7 stands for a non-zero datum in position 7. That is, the position of extremum of the wavelet transform is shifted along the scales. This should be considered in the noise removal algorithm and in the final results as an input of pattern classification because it will affect the clustering results.

Pattern classification is strongly dependent on the structure of input, for example, (2, 0, … 0, 3) and (2, 0, … 3, 0) should be classified as two patterns. On the other hand, the structure of extrema representation is changed at different scales because of the shift. Thus, the above two representations may come from the same pattern but from different scales, for example, one from scale 4 and the other from scale 5, as illustrated in Figure 3-16. This situation has to be considered especially when different sources of signals are used and processed. Following the piece-wise procedure deals with the influence of the shift as well as reducing dimensionality.

Normally, representations of extrema become a highly sparse vector after removing the noise components. This is true especially for trends of continuous chemical process. which are often changed at low frequency. So the linear piece-wise technique can be applied to reduce dimensionality.

A piece-wise is a function that is linear discrimination over sub-regions of the feature space. The discriminant functions are given by

$$d_k(\bar{x}) = \max_{m=1,\ldots N_k} [d_k^m(\bar{x})] \qquad \textbf{(3 - 22)}$$

k=1, ..., M

The piece-wise procedure is to find the maximum $d_k^m(\bar{x})$ along the prototype for sub-region k.

It is easy to pick out the non-zero items of features represented by extrema of the wavelet transform. This achieves a high dimensionality reduction since it is a highly sparse vector. However, this will destroy the time location of the extrema, for example the features of signal in Figure 3-16 are (x_1, x_2, x_3, x_4). If this is used for pattern identification, the clustering results only discriminate according to the magnitudes of the feature while there is a loss of information with respect to time.

In this work, the length of piece-wise sub-region is fixed in order to keep the time location information and obtain a uniform representation, as it is required for pattern classification. If a sub-region consists of four data levels, then the extrema representation of scale 4 and 5 in Figure 3-16 is a vector with 16 items and becomes a uniform representation (0, x_2, 0, 0, 0, x_6, 0, 0, 0, x_{10}, 0, 0, x_{13}, 0, 0, 0).

3.5 Comparison with Fourier Analysis

Wavelet transformation is similar to window Fourier transformation. Both the wavelet and window Fourier transforms are localised in time and frequency and may be used for approximating signals in scale-space.

The Fourier transform of a function is defined by:

$$\hat{f}(\omega) = F(\omega) = \int_{-\infty}^{\infty} e^{-i\omega x} f(x)dx \qquad \textbf{(3 - 23)}$$

The inverse Fourier transform of $\hat{f}$ is defined by

$$F^{-1}(x) = \frac{1}{2\pi}\int_{-\infty}^{\infty} e^{i\omega x}\hat{f}(\omega)d\omega \qquad \textbf{(3 - 24)}$$

with coefficients given

$$c_k = \frac{1}{2\pi}\int_{0}^{2\pi} f(x)e^{-ikx}dx \qquad \textbf{(3 - 25)}$$

The Fourier transform can be regarded as the decomposition of a function *f(t)* into a sum of frequency components, the coefficients of which are given by the inner product of *f(t)* and $e^{-i\omega t}$. This transform uses sines and cosines as the basis functions to map a time domain function into the frequency domain. Therefore the spectrum $\hat{f}(\omega)$ shows the overall strength with which any frequency ω is contained in the function *f(t)*. However, the standard Fourier transform only gives a representation of the frequency content of *f(t)*, information concerning time-localisation is not easy to obtain from $\hat{f}(\omega)$.

Time-localisation can be achieved by windowing the signal so as to cut off a localised slice of signal and then taking its Fourier transform – short-time Fourier transform (STFT) or windowed Fourier transform. The original idea comes from the work done by Gabor (1946), so sometimes it is called the Gabor transform. The general definition of the Short-time Fourier transform is,

$$F(t,\omega) = \int_{-\infty}^{\infty} f(\tau)g(\tau - t)e^{-j\omega\tau}d\tau \qquad \textbf{(3 - 26)}$$

(a)

(b)

Figure 3 - 19 Resolution of (a) window Fourier transform and (b) wavelet transform

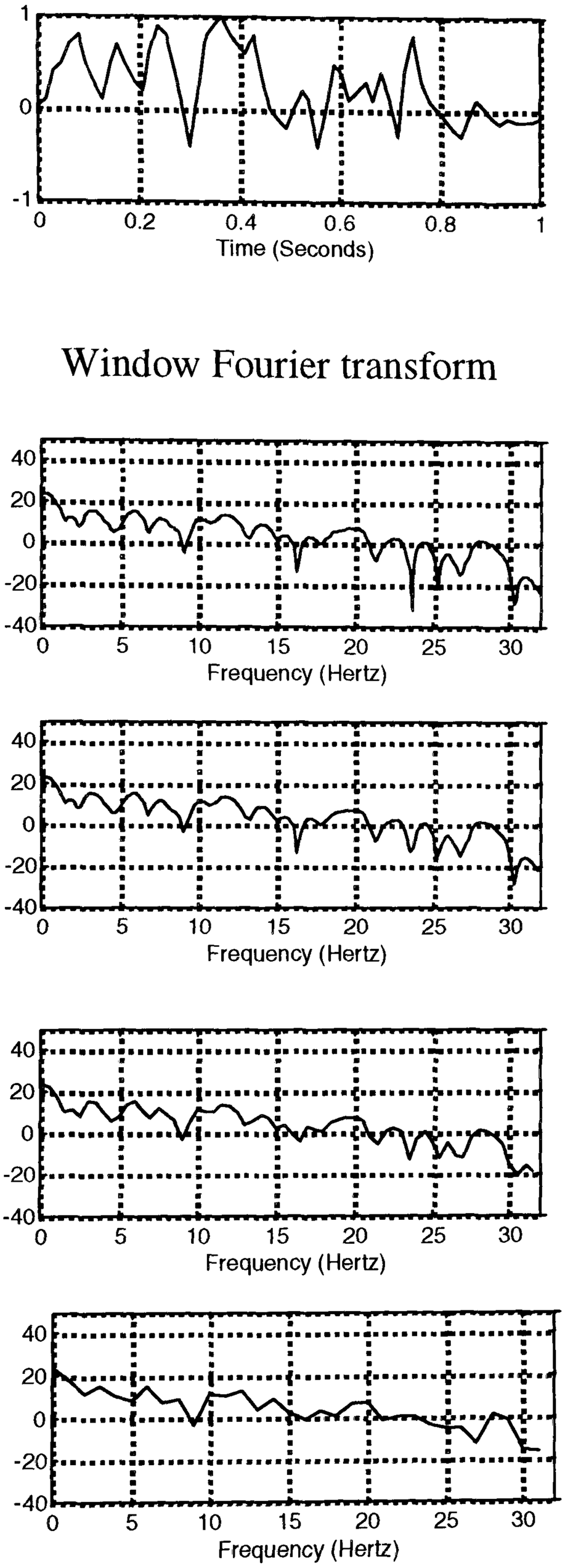


Figure 3 - 20 A process signal and its window Fourier analysis

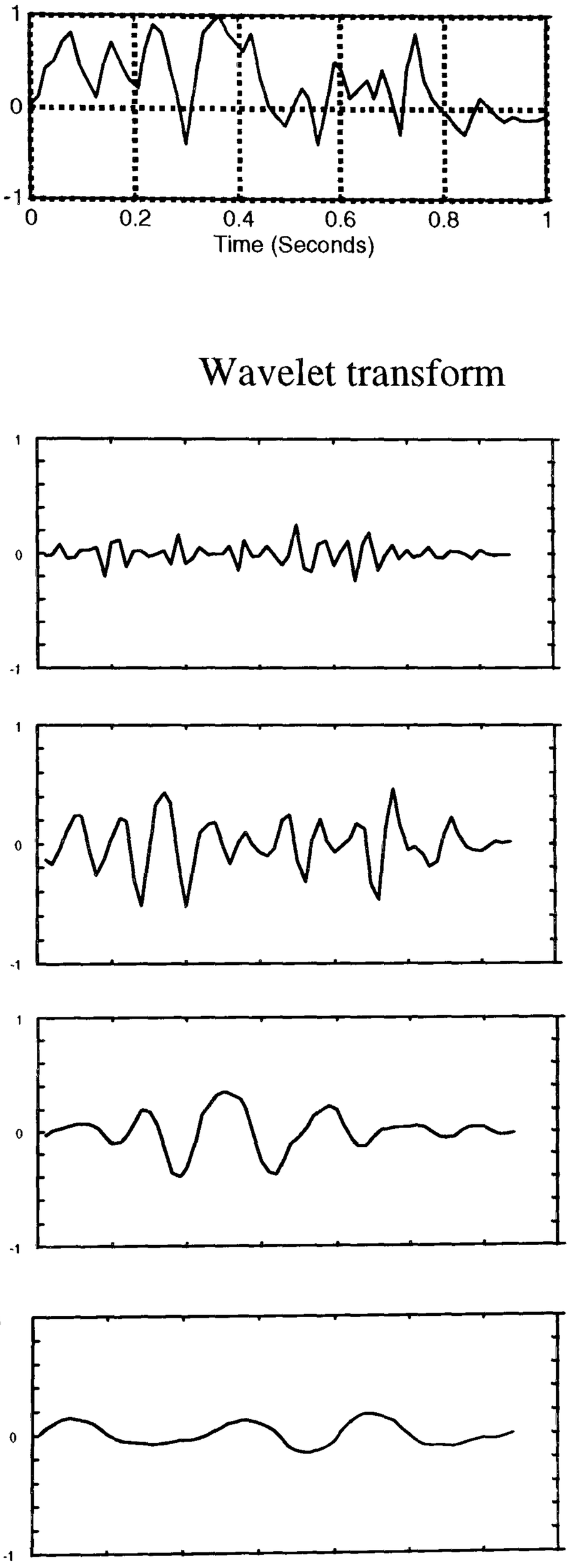


Figure 3 - 21 The signal in Figure 3-20 and its wavelet analysis

Thus, STFT maps a time domain function $f(t)$ into a time-frequency domain function $F(t, \omega)$ and is a linear mapping from the Fourier transform definition. Actually, at time t, the STFT is the Fourier transform of the function, $f(t)$, multiplied by a window $g(\tau-t)$. Since the window suppresses all function features outside the local neighbourhood around time t, the STFT is simply a "local spectrum".

However, using wavelets to represent irregularities and singularities, or sudden changes in the signal, is much more efficient than the windowed Fourier transform. The time and frequency resolution of a windowed Fourier transform is constant, as illustrated in Figure 3-19(a). If σ_t is the standard derivation of the window function, then the information provided by this decomposition is not localised within intervals of size less than σ_t. If the signal has a discontinuity, then it is difficult to locate the discontinuity with a precision better than σ_t. If the signal has important features for the different sizes, an optimum resolution cannot be defined to analyse the signal.

On the other hand, a wavelet can zoom in on such irregularities and singularities of varying size because the σ_t/s for wavelets may be varied by changing the scale parameter s, which is represented by Equations 3-8a and b. Figure 3-19(b) shows the resolution of wavelet transform.

Figure 3-20 illustrates the Fourier analysis with different window widths, and Figure 3-21 shows wavelet transform with increases in scale. Using windowed Fourier analysis, typical frequency peaks are able to represent features of the signal. Generally, process signals have a wide frequency distribution, and the feature representation based on windowed Fourier transform becomes less efficient and more difficult. More theoretical analysis and explicit explanation about the superior performance of wavelet to windowed Fourier transform is described by Daubechies (1992).

3.6 Concluding Remarks

A signal pre-processing approach using wavelet multi-scale analysis has been introduced in this Chapter. The approach is developed based on the fact that

irregularities and singularities contain the most important information about trend signals. Since the extrema of wavelet transform of signals are able to capture all the irregularities and singularities of a signal when a filter bank and wavelet function are properly selected, they are regarded as the features of the trend. The advantage of being able to capture both the frequency and time features of a transient signal makes the wavelet feature extraction approach suitable not only for continuous but also batch operations.

In addition, a wavelet-based noise component removal procedure is included in the approach, which extracts features and filters out noise in a single wavelet transform. Furthermore, a uniform sub-region piece-wise technique can be applied to obtain steady feature representation as well as reducing dimensionality for pattern identification.

Chapter 4

MEASUREMENT INTERPRETATION USING UNSUPERVISED LEARNING

4.1 Introduction

A method for feature extraction by processing dynamic transient signals using wavelet analysis was introduced in Chapter 3. In this Chapter, an integrated framework, ARTNET, is described which combines the feature extraction method with an unsupervised neural network, known as adaptive resonance theory ART2. The intention is to demonstrate that ARTNET is more effective than ART2 in dealing with noise contained in the transient signals while retaining an unsupervised and recursive clustering approach.

The remainder of this chapter is organised as follows. Section two describes the distinctive characteristics of measurement interpretation with an emphasis on the attributes that motivate application of ART type competitive learning as a possible solution. Then an unsupervised learning algorithm, ARTNET, for pattern interpretation is introduced. The algorithm adopts the basic architecture of ART2, but modifications have been made so that it is suitable for process applications by associating the wavelet-base feature extraction approach. ARTNET is compared with ART2 to demonstrate improvements when applied to chemical manufacturing problems from which some overall observations are made.

4.2 Issues in Interpretation of Measurements

The purpose of this section is to define the measurement interpretation problem and to demonstrate the motivation of using ART type unsupervised pattern recognition algorithm as a problem solving technique. The most common application – qualitative state identification is first considered to determine when the states of a process are normal, abnormal or uncertain. These concepts are then extended to multiply defined states i.e. multiple types of fault, since they are of particular interested in this work.

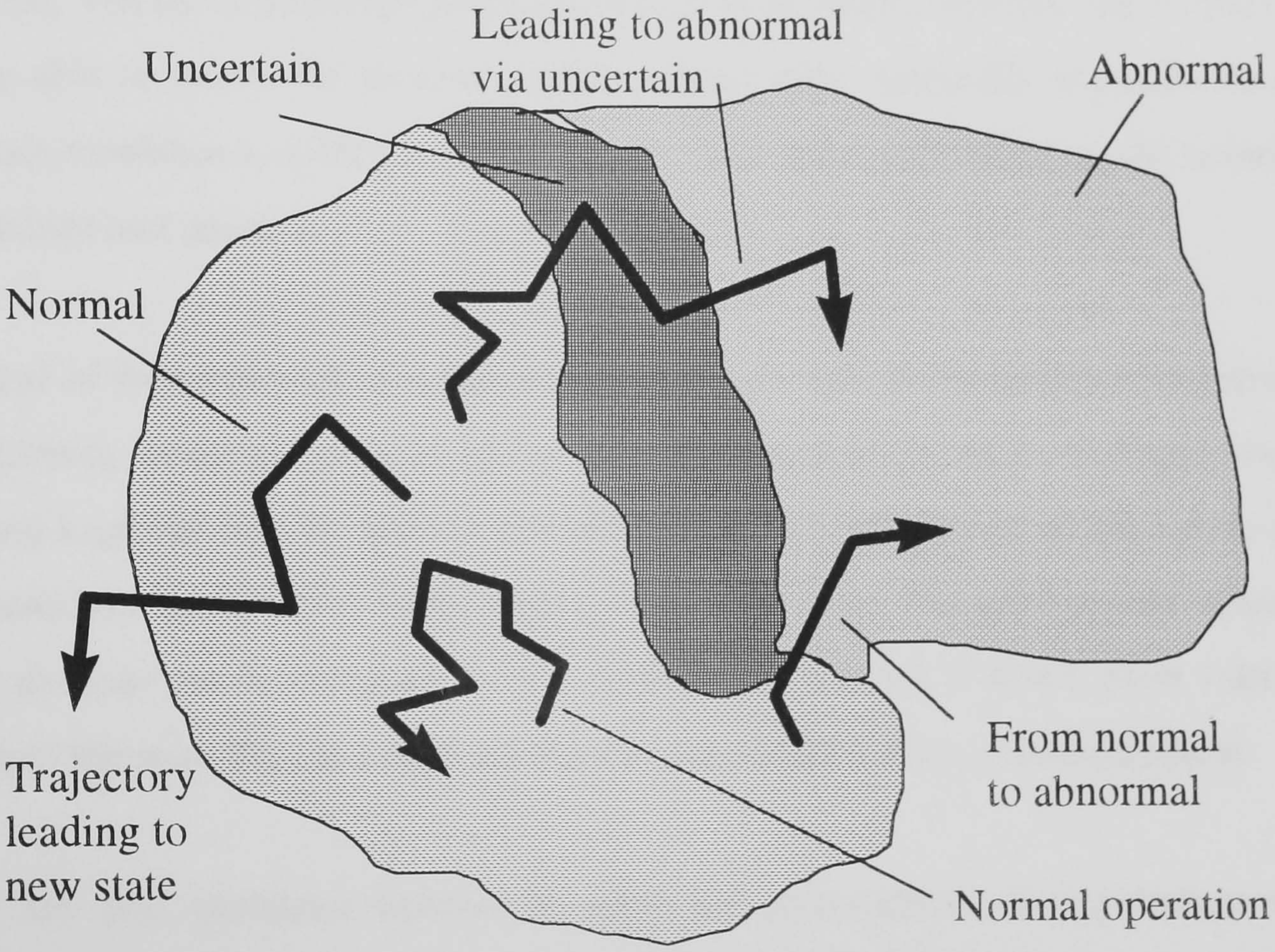


Figure 4 - 1 Pattern distribution in representation space

A process measurement can be regarded as an approximation of the vector of the process fingerprint, commonly referred to as a feature. Different vectors can be used, depending on the choice of representation for the pattern. Time and frequency or time-frequency domain representation is used in this work. Using such a representation, the set of all possible pattern slices extracted from archived plant operating data are distributed in the

manner shown in Figure 4-1. The trajectory formed by successive fingerprint vectors in the pattern representation space captures the evolving behaviour of the process.

When performing as intended, the control system constrains the process trajectory within the normal region. However, disturbances can cause the trajectory to leave normal operating conditions and may lead to uncertain or an abnormal state, or a state never seen before, as illustrated in Figure 4-1. The existence of an uncertain region reflects a typical process having dynamics measured in minutes or longer. With the exception of catastrophic events, instantaneous transitions between normal and abnormal states do not occur but involve a transition period, where it is uncertain whether the control system will be able to return the process to the original state. Generally it passes through an uncertain condition to either a normal or abnormal state, depending on the nature of the disturbance and process.

The goal of interpretation of measurements is to identify the boundaries associated with the different process states in the pattern representation space of Figure 4-1. The resulting map can then be used to classify the state of the process at any future point in time based on the location of the fingerprint on the map. The pattern data necessary to create the map can be obtained from historical plant records if future plant behaviour is similar to the past. This is a mild assumption satisfied by many chemical plants.

There are two problems needed to meet the measurement interpretation when a computer aided operational support system is used for process monitoring and fault diagnosis. Firstly, the knowledge used to form the support system is always incomplete. The second is a realisation that the data patterns will change with time due to unavoidable phenomena such as instrument degradation or catalyst deactivation.

If historical records are used as training data for a fault monitoring and diagnosis operational support system, the distribution of abnormal data is incomplete and limited to the faults that have occurred. Consequently, training data will be unavailable for many situations that should be identified as abnormal. At best, the new pattern, including

uncertain ones should be identified as unknown or new. Some nearest neighbour algorithms such as k-means unsupervised learning, and linear discriminant classifiers cannot be used in such situations, especially when a new disturbance generates a sequence of patterns not previously seen. The supervised non-linear classifier, such as a feed-forward neural network with error back propagation algorithm is prone to error in such cases. The resulting extrapolation errors occur due to the inability to define boundaries to regions lacking counter-exemplars. Likewise, it is fundamentally incorrect to conclude that previously unseen patterns are normal, based on a simple nearest neighbour analysis. The new trajectory shown in Figure 4-1 is most likely to be abnormal and should therefore never be classified as normal.

It is imperative that decision surfaces faithfully reconstruct the boundaries defined by training data so as to avoid potentially catastrophic extrapolation errors. Two types of class boundaries are evident in Figure 4-1. The first exists at the intersection between different pattern classes. This boundary type represents discrimination knowledge. The second type exists wherever the data distribution is bounded by an empty new state representation space. These boundaries represent limits on knowledge which can be extracted from the training data.

Shifting data distributions associated with natural phenomena, such as catalyst deactivation, implies the need to routinely modify class boundaries. For example, patterns that indicate conditions are too severe or abnormal for a reactor with fresh catalyst may be normal for the reactor with aged catalyst. Generally, the shift in the two distributions occurs gradually over time rather than as a discrete event. An adaptive mapping capability must be incorporated to deal with this situation. However, if the time scale associated with the shift is large enough, real-time adaptation should not be required. Discrete updates for each period should be adequate, depending on the nature of the process.

These can be extended to multi-states by dividing any abnormal region into several types of faults so that uncertain states can differentiate between various abnormal states as well

as between normal and abnormal. The problem then becomes one of multiple boundary identification or multiple state classification.

To illustrate this, consider a process having a set of measurements $(x_1, x_2, \cdots x_N)$, which reflect the value of outputs, manipulated inputs, measured external disturbances, violation of output constraints, and set points at a given time-point or over a time-interval. A pattern **p** is designated as the N-dimensional vector of these features $\mathbf{x} = (x_1, x_2, \cdots x_N)$, rather than as an object in **p**. Any pattern corresponds to a particular class of operating situations, e.g. sensor or actuator failure, process equipment failure, process parameter changes, effect of disturbances, etc. Assuming K distinct classes of operating situations such as $(C_1, C_2, \cdots, C_k)$, a pattern classification is the task which operates on a feature space S with a process to cluster the sample feature vector in S into K classes. These learned classes are then used to operate future input feature vectors to enable decisions to be taken in respect of the classes. This mapping from S to the K-dimensional vector of distinct operating situations implies the need for determining those discriminant functions that define the boundaries of the regions.

For the multi-state case, the complexity increases because more variables have to be involved to separate the states, although the types of boundaries can remain the same during the extension. This implies that an efficient algorithm is essential for state identification in real-time, which is usually required in fault monitoring and diagnosis of process plant. Above all, a successful approach to interpretation of measurements should be able to

1. distinguish different kinds of boundary to avoid extrapolation, i.e. issue an alert for a new pattern of a state which has never been seen before and denote it as don't know for a uncertain state;
2. update boundaries when there is a change in operating conditions;
3. be a computationally efficient algorithm so as to satisfy real-time requirements.

In addition to the efficiency required to meet the real-time requirements, the key to ensuring satisfactory behaviour is to use a clustering technique with closed classification

boundaries, uniform clusters, and adaptive capability. The goal of ARTNET is to provide this capability and is based on a new unsupervised competitive learning algorithm, described in detail below.

4.3 Measurement Interpretation Using ARTNET

In this section, an unsupervised competitive learning algorithm ARTNET is introduced. It adopts the basic architecture of ART2, but modifications have been made to enable it to be used for process applications. This section covers the architecture of ARTNET, similarity measures, information processing including generation of an algorithm for ARTNET, and distance threshold selection to make it possible to determine the size of the cluster.

4.3.1 Architecture of ARTNET

The ART network was developed for categorising patterns using a competitive learning paradigm. Ripley (1996) comments that ART is closely related to an adaptive version of k-means, but is expressed in a pseudo-biological language that clouds its simplicity. On the other hand, Pao (1988) believes the top-down verification step proposed is very important. In addition it introduces a gain control and a reset to make certain that learned categories are retained even while new categories are being identified. The most important contribution and the reason why ART2 is widely used is that it successfully addresses the stability-plasticity dilemma, i.e. letting a system adapt without allowing current inputs to destroy past training. This philosophy is the underlying motivation adopted for the basic architecture in this work. Two aspects in particular are of importance. Firstly, the pseudo-biological language in ART is not essential and so need not be used. This makes the algorithm simpler and more efficient. Secondly, top-down verification and the stability-plasticity capability must be retained.

As noted in section 2.3.2, the efficiency of ART2 is very strongly dependent on the size of the input vector, so a reduction in the dimension of the input vector is needed. This issue is dealt with by using the wavelet-base feature extraction approach discussed in

Chapter 3. To use the extracted features, modifications need to be made to the ART2 algorithm.

There are several adjustable parameters required in the step 1 of the ART2 algorithm as shown in Appendix B. These parameters are related to the input patterns pre-processing and are known as the internal representation. This is one of the main features of the ART2 network. The input patterns to the network are normalised with respect to unit length and correspond to the projection of the patterns on the surface of the unit hyper-sphere. The difficulty is in determining these parameters because they depend on the input data structure. The internal representation also leads to a time-consuming procedure which is not the best option for application to a process having real-time requirements. To avoid this, the pre-processing procedure in ART2 is not used in ARTNET.

Normally, the noise component filtering scheme used in ART2 discards values less than a certain threshold. Such noise suppression is not appropriate for process measurements because it can be of high frequency and significant magnitude, moreover, a small change does not always indicate noise. So the second modification is to take out noise suppression from the original ART2 algorithm and replace it with a wavelet-base feature extraction approach. By removing the biological net in ART, the metric needed to measure similarity can be substituted by a more appropriate term. Instead of counting the number of matching features, the distance between input pattern and the exemplars is used to determine the winning node during competitive learning.

The modifications can be summarised as follows. The kernel of ART, including the bottom-up scheme and, most importantly, the stability-plasticity capability is retained. Input pre-processing and noise suppression is deleted, while a defined distance replaces counting the number of matching features as the criterion for similarity. The resulting architecture of ARTNET is shown in Figure 4-2. In the ARTNET architecture, the pattern feature vector $(x_1, x_2, \cdots, x_N)$ is firstly fed to the input layer and weighted by b_{ij}, bottom-up weights. The weighted input vector is then compared with the exemplar (N-

dimension) of each existing class in the top layer by calculating the distance between input and existing exemplars. ARTNET activates the node with the smallest distance and suppresses others. The output is an active node if the smallest distance is less than a pre-set threshold, the active node is the winner and the exemplar is modified by taking the new input into account. If it is greater than the threshold a new node is created which signifies a no winner competition.

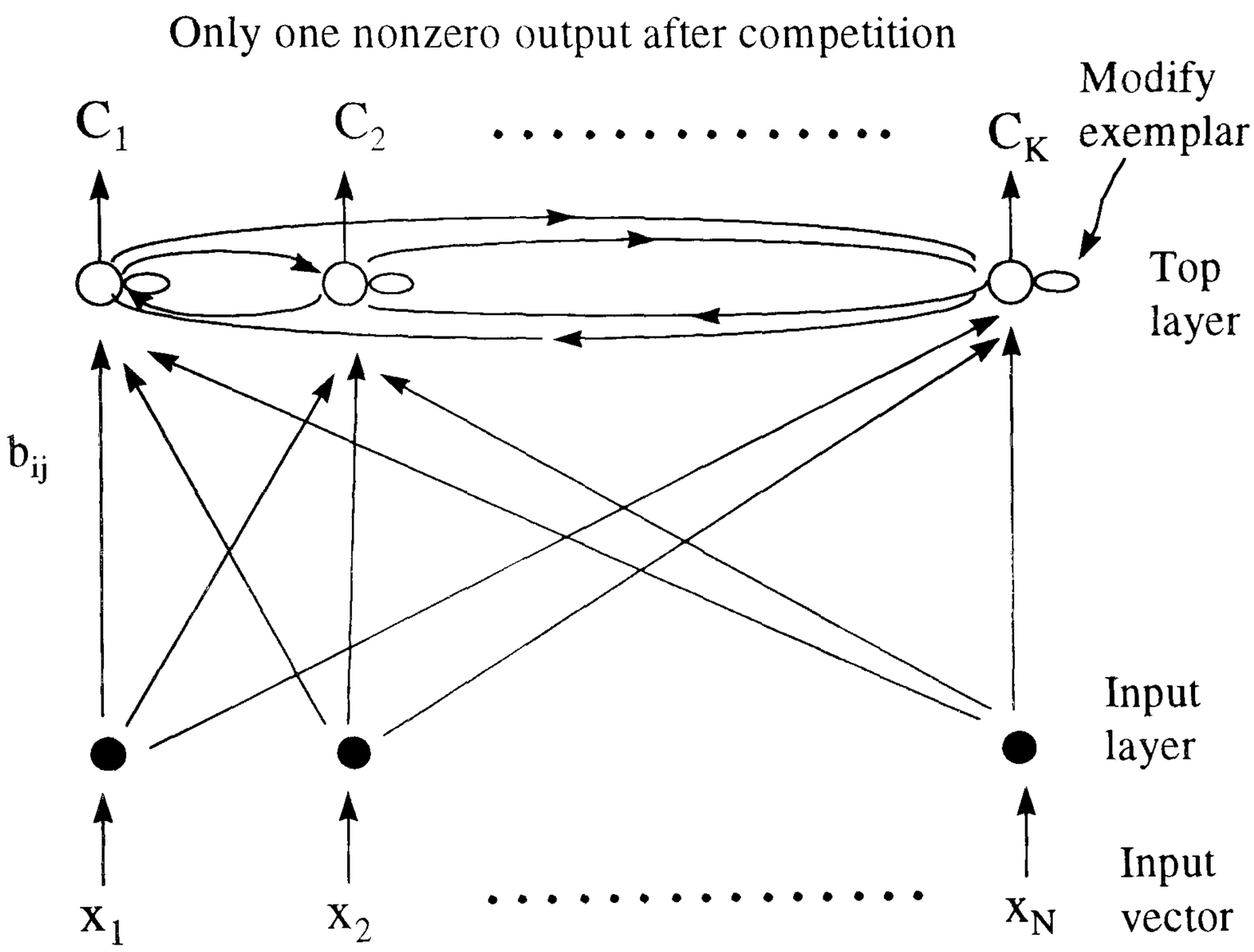


Figure 4 - 2 Basic architecture of ARTNET

4.3.2 Similarity measure in ARTNET

During unsupervised learning, it is necessary to consider how much alike any two different feature vectors are. These are sometimes called dissimilarity measures because a greater distance means greater dissimilarity. Any measure of the degree of likeness is called a similarity measure. The function giving the distance between two vectors is the most useful and is used in ARTNET as a similarity measure. There are several ways to measure the distance between the pairs of observations, such as Hamming or Euclidean distances. For continuous data, the Euclidean distance is the most commonly used to

measure similarity (Looney 1997). Formally, the Euclidean distance between two vectors x and y is defined as the root sum-squared error

$$\|x-y\|_2 = \left\{\sum_{n=1}^{N}(x_n - y_n)^2\right\}^{1/2} \qquad \textbf{(4 - 1)}$$

Each class may be represented by one or more exemplars, or prototypical, vectors forming idealisations of vectors in that class. Given any input feature vector **x** to ARTNET, the k_0 minimised norm $\|x-z^{(k_0)}\|_2$ over all K class exemplars $\{z^{(k)}\}$, for the various classes, also minimises the distance by assigning **x** to class k_0 when

$$\|x-z^{(k_0)}\|_2 = \min\left\{\|x-z^{(k)}\|_2\right\} \qquad \textbf{(4 - 2)}$$

The sum-squared error within a cluster with centre $z^{(k)}$ is the sum of the distances squared between the member points in the cluster and $z^{(k)}$, which is

$$\sigma^2(k) = \sum_{(x\in Cluster(k))}\left(\|x-z^{(k)}\|_2\right)^2 \qquad \textbf{(4 - 3)}$$

Based on this distance measure, the cluster size in ARTNET depends on a pre-set distance threshold, ρ. When the distance is less than ρ, the input will belong to the *k*th exiting class, and the elements in the class $z_i^{(k)}$ will be modified as follows:

$$z_i^{(k)} = z_i^{(k)} + \frac{1}{NP}x_i b_{ij} \qquad \textbf{(4 - 4)}$$

where NP is the number of patterns in the class. If the distance is greater than ρ for all existing classes, a new class is created which provides adaptivity and so avoids extrapolation.

4.3.3 Information processing and a computational algorithm

The architecture shown in Figure 4-2 is similar to that employed by many self-organising systems. Two layers of units or nodes are assumed with the lower layer accepting input from the environment on the basis of one input unit for each pattern feature. Each unit in the top layer corresponds to a pattern prototype defining a map cluster. The details of

the connections between the input and output units define the ARTNET model, which is dependent on the information processing procedure in ARTNET.

Information processing by the ARTNET network can be broadly summarised as follows. When an input pattern is presented to the network, it is compared to each of the existing prototypes in the top layer. The winner in the top layer is the prototype that is most similar to the input. If the similarity between the winner and input is less than a predetermined value, the distance threshold ρ, ARTNET learning is enabled and the winner is modified slightly to more closely reflect the input. If the similarity between the winner and input exceeds that required by the distance threshold, the current winner is disabled and the search process is repeated. If none of the successive winners exhibit adequate similarity, a new unit is generated in the top layer and learning is enabled to form a prototype similar to the input or takes the value of the input vector as a new exemplar.

The clusters formed by ARTNET are defined by the prototypes in the top layer and the distance threshold ρ. All clusters formed by an ARTNET model are of uniform size and provide closed classification boundaries.

In ARTNET, the first pattern of a new node in the top layer is set to the exemplar i.e. $z_i = x_i$, which implies that the bottom-up weights are a unit vector. This further simplifies the computation. The ARTNET computational programming algorithm is as follow:

1. Set initial parameters: distance threshold ρ
2. Read input patterns
3. Set initial bottom-up weights equal to the first pattern if there is no existing weight:

$$z_i^{(1)} = x_i^1, \quad K = 1, \quad NP^1 = 1$$

4. Find the next exemplar.

For $q = 2$ to N_{in} (number of input patterns)

For $i = 1$ to N_{attr} (number of attributes in a input vector)

Calculate the Euclidean distance

$$Eu = \left\| z - x^{(q)} \right\|_2$$

End

End

5. IF the distance is greater than the threshold, THEN

   form a new cluster

   ELSE

   update bottom-up weights by

$$z^{(K)} = z^{(K)} + \frac{1}{NP} x^{(q)}$$

   Update all indices and count

$$NP = NP + 1, \quad K = K + 1$$

   where NP is the number of pattern contained in the class

6. End

4.3.4 Distance threshold determination

In ARTNET, the distance threshold is a tuning parameter that can be any value greater than zero. The distance threshold is obtained by using a set of known input-output patterns. This procedure is a supervised iterative process ensuring that an appropriate distance threshold is used.

The following example is used to illustrate the procedure for determining the distance threshold. The data sets are taken from the work of Wang and Chen (1998), where ART2 is used to classify lubricating base oils in terms of their infrared spectra. Fifty-nine data patterns are included which were collected from 12 refineries representing eight different crude oils.

In general, ARTNET groups more patterns into a class as the distance threshold increases, i.e. the number of classes is reduced when the distance threshold is increased. Table 4-1 shows how the clustering relates to the distance threshold selection in

ARTNET. According to prior knowledge, the distance threshold can be any value between 0.97 to 1.2. This groups the data pattern into 7 classes.

Table 4 - 1 Lubricating base oils clustered for different distance thresholds by ARTNET

Distance threshold ρ	Number of patterns identified
0.7	16
0.75	14
0.8	11
0.9	9
0.95	8
0.97	7
1.2	6
1.38	5
1.4	4

The relationship between the number of classes and the distance threshold value is a monotone function. Therefore, the distance threshold can be determined by a searching algorithm. The searching algorithm used here is as follows.

1. Given the expected class number, N_C, which predicts the largest value of the distance threshold ρ_{max}, then let $\rho = \rho_{max}$
2. Use ρ as a distance threshold for clustering in order to get a number of the class N_{cal}
3. IF N_{cal} is less than N_C, THEN

   $\rho = \frac{\rho}{2}$ and go to step 3
4. IF N_{cal} greater than N_C, THEN

$$\rho = \rho + \frac{\rho}{2} \quad \text{go to step 3}$$

5. IF N_{cal} equals N_C, stop searching

4.4 Comparison with ART2

It is useful to compare ARTNET with ART2 in order to illustrate the benefit of the above modifications. This is conveniently done by applying it to a process engineering case study. The comparison used here relates to threshold determination, computation time, and sensitivity to noise.

The data patterns used for comparison are a set of faults and operational disturbances occurring in a refinery fluid catalytic cracking (FCC) process. Fault and disturbance data sets are from a simulator so the original data patterns are noise free. Details of the FCC process simulator are described in Chapter 6. Table 4-2 gives the fault and disturbance data sets used, which are organised into fault types and labelled with a data set number. For example, data set number 1 to 9 are grouped as fault type 1 and refer to fresh feed increasing by 10% to 90%.

The data sets consist of twelve faults or disturbances, as shown in Table 4-2. Eight operational variables are measured to characterising the faults or disturbances, namely, reaction temperature, regenerator temperature, reactor pressure, regenerator pressure, volumetric percentage of carbon dioxide in flue gas, volumetric percentage of oxygen in flue gas, catalyst recycle rate and catalyst hold-up in the reactor.

4.4.1 Threshold selection

The selection of the distance threshold in ARTNET and vigilance in ART2 are very important and are compared using the original simulation data. The distance threshold is the only tuning parameter in ARTNET, while vigilance is one of the adjustable parameters in ART2, with the same function as the distance threshold used in ARTNET. The clustering results for different thresholds for ARTNET and vigilance for ART2 are listed in Table 4-3 and 4-4 respectively.

Table 4 - 2 Description of data sets used to compare ARTNET and ART2

Data set	Type	Time t < 0 operation is at steady state, at t = 0 make the following step change
1 ~ 9	**1**	fresh feed increased by 10 20 30 40 50 60 70 80 90%
10 ~ 18	**2**	fresh feed decreased by 10 20 30 40 50 60 70 80 90%
19 ~ 24	**3**	preheat Temperature T of mixed feed increased by 5 10 15 20 °C and preheat Temperature T of mixed feed decreased by 15 10 °C
25 ~ 26	**4**	recycle slurry Flow rate F increased by 70 90%
27 ~ 28	**5**	recycle slurry Flow rate F decreased by 70 90%
29 ~ 32	**6**	opening ratio of hand-valve V20 increased by 5 10 15 24%
33 ~ 37	**7**	opening ratio of hand-valve V20 decreased by 10 15 25 35 55%
38	**8**	cooling water pump P-02 failure
39 ~ 43	**9**	air flow rate increased by 6.5 11.5 15 31.5 40.5%
44 ~ 49	**10**	air flow rate decreased by 3.5 8.5 28.5 38.5 48.5 53.5%
50	**11**	compressor failure
51 ~ 57	**12**	valve 401-ST opening from 100% decreased by 10 20 40 45 60 80 90%

Table 4 - 3 Clustering for different distance thresholds by ARTNET based on the data sets listed in Table 4-2

Distance threshold ρ	Number of classes identified	Grouping of data sets
0.8	57	
1.0	56	[56 57]
2.0	53	[5 7] [25 26] [27 28] [56 57]
3.0	50	[5 7] [19 20 23 24] [25 26] [27 28] [56 57]
4.0	48	[5 6 7 8] [19 20 21 23 24] [25 26] [27 28] [56 57]
4.5	42	[3 4 5 6 7 8 9] [19 20 21 22 23 24] [25 26] [27 28]
		[35 36] [56 57]
5.0	41	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29]
		[25 26] [27 28] [35 36] [56 57]
6.0	40	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29 52]
		[25 26] [27 28] [35 36] [56 57]
12.0	12	[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 39 40 41 42 43 44 45 46 51]
		[17 18 19] [50 52 53]

Note: [56 57] means that data sets 56 57 are clustered in one class

Table 4 - 4 Clustering for different vigilance by ART2 based on the data sets listed in Table 4-2

Vigilance	Number of classes identified	Grouping of data sets
0.9998	57	
0.9996	56	[56 57]
0.9992	53	[5 7] [25 26] [27 28] [56 57]
0.9990	50	[5 7] [19 20 23 24] [25 26] [27 28] [56 57]
0.9987	48	[5 6 7 8] [19 20 21 23 24] [25 26] [27 28] [56 57]
0.9985	42	[3 4 5 6 7 8 9] [19 20 21 22 23 24] [25 26] [27 28]
		[35 36] [56 57]
0.9982	41	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29]
		[25 26] [27 28] [35 36] [56 57]
0.9978	40	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29 52]
		[25 26] [27 28] [35 36] [56 57]
0.9965	12	[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 39 40 41 42 43 44 45 46 52]
		[16 17 18] [50 51 53] [56 57]

Note: [56 57] means that data sets 56 57 are clustered in one class

Table 4 - 5 Pairing of identified classes by ATRNET and the data sets numbered in Table 4-2 when the distance threshold ρ is 0.5

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	29	**31**	45
2	2	**17**	30	**32**	46
3	3 4 5 6 7 8 9	**18**	31	**33**	47
4	10	**19**	32	**34**	48
5	11	**20**	33	**35**	49
6	12	**21**	34	**36**	50
7	13	**22**	35 36	**37**	51
8	14	**23**	37	**38**	52
9	15	**24**	38	**39**	53
10	16	**25**	39	**40**	54
11	17	**26**	40	**41**	55
12	18	**27**	41	**42**	56 57
13	19 20 21 22 23 24	**28**	42		
14	25 26	**29**	43		
15	27 28 '	**30**	44		

Table 4 - 6 Pairing of identified classes by ART2 and the data sets numbered in Table 4-2 when the vigilance is 0.9985

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	29	**31**	45
2	2	**17**	30	**32**	46
3	3 4 5 6 7 8 9	**18**	31	**33**	47
4	10	**19**	32	**34**	48
5	11	**20**	33	**35**	49
6	12	**21**	34	**36**	50
7	13	**22**	35 36	**37**	51
8	14	**23**	37	**38**	52
9	15	**24**	38	**39**	53
10	16	**25**	39	**40**	54
11	17	**26**	40	**41**	55
12	18	**27**	41	**42**	56 57
13	19 20 21 22 23 24	**28**	42		
14	25 26	**29**	43		
15	27 28	**30**	44		

Using ARTNET, the number of classes is reduced from 57 to 12, when they are grouped according to the faults and disturbances, as the distance threshold increases from 0.8 to 12.0. On the other hand, the vigilance in ART2 decreases from 0.9998 down to 0.9965 for the same results. It is clear that the distance threshold in ARTNET varies over a wider range than the vigilance in ART2 for the same degree of clustering. This implies that the distance threshold in ARTNET is less sensitive than the vigilance in ART2.

On carefully checking of the results in Table 4-3 and 4-4, it is found that, in fact, clustering the data sets into 12 classes leads to undesirable pattern groupings. Some data sets which have significantly different fault roots are clustered into one class. For example, data sets 1 ~ 15, 19 ~ 37, 39 ~ 46, and 51, belonging to four types of fault roots are clustered into one class. It is obviously important to select a proper distance threshold when carrying out pattern clustering in ARTNET, as well as vigilance in ART2.

For ARTNET, all data in the example have been identified as individual classes with a distance threshold $\rho = 0.8$. As the threshold value increases, some data sets are assigned to few groups. When the threshold value is 4.5, the clustering result is [3,4,5,6,7,8,9], [19 20 21 22 23 24], [25 26], [27, 28], [35,36] and [56 57]. ART2 gives the same result with vigilance 0.9985. The detail of the clustering results are illustrated in Table 4-5 and 4-6. This is the best clustering result. A further change in clustering is distance threshold $\rho = 5.0$ in ARTNET, corresponding to a vigilance of 0.9982 in ART2. The span of the distance threshold is 0.5, but that of the vigilance is only 0.003. Thus a very small change of vigilance results in a change of clustering in ART2. On the other hand, the same change for ARTNET can be a much wider distance threshold selection.

Two observations can be made:

1. ARTNET is less sensitive for the distance threshold selection than vigilance in ART2
2. ARTNET has the same clustering and stability-plasticity abilities as ART2 based on noise-free simulated data sets.

4.4.2 Computation speed

Nowadays, computational speed is less of a crucial issue because of ever-increasing computer power. However, it is nevertheless useful to see whether the modifications of ART2 have increased the computational cost.

The computation time represented here is based on MATLAB programs on a Pentium MMX 166 for clustering 57 data patterns into 42 classes, as discussed in section 4.4.1. The computation speed of ARTNET and ART2 is compared in the three cases, shown in Table 4-7.

Table 4 - 7 Computation time for ARTNET and ART2 in different cases

		ARTNET	ART2
case one	simulation data for both	1m 20s	10m 17s
case two	feature by wavelet approach for both	28s	3m 12s
case three	feature by wavelet approach for ARTNET, simulation data for ART2	4m 15s	10m 17s

In case one, simulated data sets have been used directly. Eight operational variables are used to define the states, and each variable has one 64 data point sample. Thus the dimension of input to ART2 or ARTNET is 512. ART2 takes 10m 17s to achieve the clustering, while it is only 1m 20s for ARTNET.

In case two, the wavelet based approach developed in Chapter 3 is used to extract features of the trend so as to reduce the dimensionality. Feature extraction results are used as input to both ART2 and ARTNET. The dimension of each variable now corresponds to 8 features after extraction, so the input of ART2 and ARTNET is 64 rather than 512. The clustering time is therefore significantly decreased for ARTNET, and it is still faster than ART2.

In case three, the wavelet feature extraction approach is used for the pre-processing part of ARTNET. Consequently the simulation data is firstly pre-processed using wavelet

feature extraction and is then used in ARTNET for clustering. The simulation data is directly used by ART2. Thus the wavelet feature extraction approach in ARTNET acts as the internal representation and suppresses the noise scheme in ART2. Obviously. ARTNET is faster than ART2 in all the above three cases.

4.4.3 Influence of noise

The modifications of ART2 to form ARTNET include removing the noise filtering procedure from ART2 because this has been performed by the wavelet-base feature extraction approach. Here the example is used to demonstrate the effectiveness of the wavelet approach and the noise suppressing scheme in ART2 to eliminate the influence of noise.

The simulation data sets are noise free. In order to generate signal noise with different signal-to-noise ratios, the noise generator *randn* (from MATLAB) and a constant C_{noise} are used here. The noise magnitude is:

$$X_{noise} = \frac{randn(f(t))}{C_{noise}} \qquad (4-5)$$

Thus *randn(f(t))* creates normally distributed random numbers with entries chosen from a normal distribution with zero mean and variance 1.0 and numbers having the same size as *f(t)*. The constant C_{noise} controls the noise intensity so as to change the signal to noise ratio in the data sets. The data sets for pattern clustering used here are the simulation data set *f(t)* plus X_{noice}, with zero mean having been carefully checked.

The relationship of signal to noise ratio (SNR) and the constant C_{noise} are listed in Table 4-8, where $SNR = \frac{\text{signal energy}}{\text{noise variance}}$ (Candy 1988)

Table 4 - 8 The changes of SNR with different C_{noise}

C_{noise}	0.01	0.1	1	10	100
SNR	0.0026	0.2475	35.2623	2124.3	274330

Table 4 - 9 Pairing of identified classes by ATRNET when C_{noise} =0.001 ~ 100 threshold =4.5 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	29	**31**	45
2	2	**17**	30	**32**	46
3	3 4 5 6 7 8 9	**18**	31	**33**	47
4	10	**19**	32	**34**	48
5	11	**20**	33	**35**	49
6	12	**21**	34	**36**	50
7	13	**22**	35 36	**37**	51
8	14	**23**	37	**38**	52
9	15	**24**	38	**39**	53
10	16	**25**	39	**40**	54
11	17	**26**	40	**41**	55
12	18	**27**	41	**42**	56 57
13	19 20 21 22 23 24	**28**	42		
14	25 26	**29**	43		
15	27 28	**30**	44		

Table 4 - 10 Pairing of identified sets by ART2 when C_{noise} =100 and vigilance is 0.9985 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	29	**31**	45
2	2	**17**	30	**32**	46
3	3 4 5 6 7 8 9	**18**	31	**33**	47
4	10	**19**	32	**34**	48
5	11	**20**	33	**35**	49
6	12	**21**	34	**36**	50
7	13	**22**	35 36	**37**	51
8	14	**23**	37	**38**	52
9	15	**24**	38	**39**	53
10	16	**25**	39	**40**	54
11	17	**26**	40	**41**	55
12	18	**27**	41	**42**	56 57
13	19 20 21 22 23 24	**28**	42		
14	25 26	**29**	43		
15	27 28	**30**	44		

Table 4 - 11 Pairing of identified classes by ART2 when C_{noise} =10 and vigilance = 0.9985 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	27 28	**31**	44
2	2	**17**	29	**32**	45
3	3 4 5 6 7	**18**	30	**33**	46
4	8 9	**19**	31	**34**	47
5	10	**20**	32	**35**	48
6	11	**21**	33	**36**	49
7	12	**22**	34	**37**	50
8	13	**23**	35 36	**38**	51
9	14	**24**	37	**39**	52
10	15	**25**	38	**40**	53
11	16	**26**	39	**41**	54
12	17	**27**	40	**42**	55
13	18	**28**	41	**43**	56 57
14	19 20 21 22 23 24	**29**	42		
15	25 26	**30**	43		

Table 4 - 12 Pairing of identified classes by ART2 when C_{noise} =1.0 and vigilance = 0.9985 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**16**	27	**31**	44
2	2	**17**	28	**32**	45
3	3 4 5 6 7	**18**	30	**33**	46
4	8 9	**19**	31	**34**	47
5	10	**20**	32	**35**	48
6	11	**21**	33	**36**	49
7	12	**22**	34	**37**	50
8	13	**23**	35 36	**38**	29 51
9	14	**24**	37	**39**	52
10	15	**25**	38	**40**	53
11	16	**26**	39	**41**	54
12	17	**27**	40	**42**	55
13	18	**28**	41	**43**	56 57
14	19 21 22 23 24	**29**	20 42		
15	25 26	**30**	43		

Table 4 - 13 Pairing of identified classes by ART2 when C_{noise} =1.0 and vigilance =0.9991 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**17**	24	**33**	42
2	2	**18**	25 26	**34**	43
3	3 4 5 6 7	**19**	27	**35**	44
4	8 9	**20**	28	**36**	45
5	10	**21**	29	**37**	46
6	11	**22**	30	**38**	47
7	12	**23**	31	**39**	48
8	13	**24**	32	**40**	49
9	14	**25**	33	**41**	50
10	15	**26**	34	**42**	51
11	16	**27**	35 36	**43**	52
12	17	**28**	37	**44**	53
13	18	**29**	38	**45**	54
14	19 22 23	**30**	39	**46**	55
15	20	**31**	40	**47**	56
16	21	**32**	41	**48**	57

Table 4 - 14 Pairing of identified classes by ART2 when C_{noise} =0.1 and vigilance =0.9991 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**18**	21	**35**	41
2	2	**19**	23	**36**	42 25
3	3 4	**20**	24	**37**	43
4	6	**21**	26	**38**	44
5	8	**22**	27	**39**	45
6	9	**23**	28	**40**	46
7	10	**24**	29	**41**	47
8	11 5	**25**	30 7	**42**	48
9	12	**26**	31	**43**	49
10	13	**27**	32	**44**	50
11	14	**28**	33	**45**	51
12	15	**29**	34	**46**	52
13	16	**30**	36	**47**	53
14	17	**31**	37	**48**	54
15	18	**32**	38	**49**	55
16	19	**33**	39	**50**	56
17	20 35	**34**	40 22	**51**	57

Table 4 - 15 Pairing of identified classes by ART2 when C_{noise} =0.1 and vigilance =0.9999 using data in Table 4-2

Identified class	Corresponding data set	Identified class	Corresponding data set	Identified class	Corresponding data set
1	1	**20**	21	**39**	40
2	2	**21**	22	**40**	41
3	3 4	**22**	23	**41**	42
4	5	**23**	24	**42**	43
5	6	**24**	25	**43**	44
6	7	**25**	26	**44**	45
7	8	**26**	27	**45**	46
8	9	**27**	28	**46**	47
9	10	**28**	29	**47**	48
10	11	**29**	30	**48**	49
11	12	**30**	31	**49**	50
12	13	**31**	32	**50**	51
13	14	**32**	33	**51**	52
14	15	**33**	34	**52**	53
15	16	**34**	35	**53**	54
16	17	**35**	36	**54**	55
17	18	**36**	37	**55**	56
18	19	**37**	38	**56**	57
19	20	**38**	39		

Using wavelet multi-resolution analysis pre-processing to remove the noise components in the data patterns, the clustering results from ARTNET remain the same with C_{noise} =0.01 ~ 100, as shown in Table 4-9. On the other hand, the clustering by ART2 is seriously influenced by noise, especially when the signal to noise ratio of the data pattern is low, as illustrated in Tables 4-10 to 4-15.

Noise suppression in ART2 works well when the data patterns have high SNR such as C_{noise} = 100, as can be seen in Table 4-10. However, the clustering results change as the SNR decreases. As C_{noise} increases to 10, group 3 in Table 4-10 is split into two groups [3 4 5 6 7] [8 9], which means that ART2 has to create a new class to meet the vigilance criterion. The problem becomes more serious for further increases in SNR. When C_{noise} = 1.0, data patterns 20 and 29 are grouped with 42 and 51 respectively, although they are quite different. The vigilance has to be increased to separate them. As a result, other classes are split because of the increase in vigilance. Table 4-13 shows the result for C_{noise} =1.0 and vigilance =0.9991, when number of classes increases to 48 because data sets 20, 21 24, and 28 are separated as vigilance increases. When SNR is lower than 1, the clustering results of ART2 are simply disordered if the vigilance is still 0.9991, the same as it is for C_{noise} =1.0. The vigilance has to increase to 0.9999 to separate all the data sets.

Thus, the wavelet noise removal approach enables ARTNET to use a low SNR signal and retain the same clustering properties, while ART2 has to change the vigilance or/and increase the number of classes to deal with noise, especially for lower SNR. This means that ART2 may report a fault that occurred before and is archived to be a new fault because of the influence of noise. This is not harmful but is not helpful in process fault diagnosis because it makes solving the problem more difficult.

4.5 Concluding Remarks

An integrated framework ARTNET for operational state identification has been developed in this Chapter which combines wavelet feature extraction with an unsupervised neural network ARTNET. This has been applied to a case study of a

refinery fluid catalytic cracking process and found to be able to reasonably cluster the data. Compared with ART2, ARTNET is able to simultaneously reduce the dimensionality of dynamic transient signals and remove noise components through feature extraction. ARTNET has also proved to be superior to ART2 in avoiding the adverse influence of noise, selection of threshold values and in computation time.

Chapter 5

KNOWLEDGE GENERATION AND REPRESENTATION USING FUZZY NEURAL NETWORKS

5.1 Introduction

In previous Chapters, approaches to translating process trends into pattern classes by wavelet-based feature extraction and ARTNET unsupervised clustering have been developed. However, further interpretation is necessary to map the process trends to operational conditions. Because process measurements are numerical data rather than qualitative descriptions such as high temperature, it is necessary to convert the information on numerical data into qualitative form. In this Chapter, a fuzzy feed-forward neural network (fuzzy-FFNN) technique is introduced to do this, which extracts fuzzy rules automatically from numerical data. In this technique, process measurements are grouped in fuzzy regions and expressed in qualitative forms. This approach also takes account of uncertainty in process measurements and minimises loss of information because filtering of rules is not necessary. The disadvantage of using FFNN for rule generation is that it has to be retrained to reflect new knowledge. As data accumulates, such retraining becomes computationally demanding. This implies that such a system is difficult to maintain and support. Furthermore, the conflict issue is still present when the rules are generated individually, especially when they are obtained from different sorts of data.

Learning based on a FFNN involves a procedure for converting information from training data sets into the weights of a particular network architecture. Therefore the weights and architecture about the trained FFNN represent information of training data sets. A group of trained fuzzy-FFNNs is used here to provide a knowledge base which solves problems of maintenance and updating as well as resolving conflicts.

In the following, relevant fuzzy concepts are first introduced. The fuzzy-FFNN approach is then described and the problem solving ability of fuzzy-FFNN, especially in dealing with noise and conflicts in data, are illustrated using an example. A group of neural networks used to construct and represent a knowledge base is also discussed.

5.2 Fuzzy Set Theory

Some definitions relevant to fuzzy concepts and the basic operations between such sets are introduced here as they apply to the fuzzy-FFNN approach (Pao 1989, Cox 1994).

Definition 5.1 If X is a collection of objects x, then a fuzzy set $\tilde{A}$ in X is a set of ordered pairs:

$$\tilde{A} = \{(x, \mu_A(x)) | x \in X\} \qquad (5 - 1)$$

The entity $\mu_A(x)$ is called the membership function, the value of which is the grade of membership of x in $\tilde{A}$. It is also the degree to which the deterministic measurement x is compatible with the value concept of $\tilde{A}$.

Definition 5.2 The membership function $\mu_A: X \rightarrow [0,1]$ expresses the degree to which membership of a fuzzy set or region acts as a function

$$\mu_A = T(x) \qquad (5 - 2)$$

for each unique value selected from the domain. The function returns a unique degree of membership in the fuzzy region.

There are three main operators between fuzzy sets. Their definitions are based on the membership function of fuzzy sets.

Definition 5.3 The membership function $\mu_{\tilde{A}\cup\tilde{B}}$ of the union $\tilde{A}\cup\tilde{B}$ is

$$\mu_{\tilde{A}\cup\tilde{B}}(x)=\max(\mu_A(x),\mu_B(x)),\quad x\in X \qquad \textbf{(5 - 3)}$$

Definition 5.4 The membership function $\mu_{A\cap B}$ of the intersection $\tilde{A}\cap\tilde{B}$ is

$$\mu_{\tilde{A}\cap\tilde{B}}(x)=\min(\mu_A(x),\mu_B(x)),\quad x\in X \qquad \textbf{(5 - 4)}$$

Definition 5.5 The membership function of the complement of a fuzzy set A is

$$\mu_{\bar{A}}=1-\mu_A(x),\quad x\in X \qquad \textbf{(5 - 5)}$$

Definition 5.6 For a fuzzy set, a weak α-cut is defined as

$$A_\alpha\equiv\{x|\mu_A(x)>\alpha\}\quad \alpha\in[0,1) \qquad \textbf{(5 - 6)}$$

the strong α-cut is:

$$A_{\bar{\alpha}}\equiv\{x|\mu_A(x)\geq\alpha\}\quad \alpha\in(0,1] \qquad \textbf{(5 - 7)}$$

Definition 5.7 The support of a fuzzy set $\tilde{A}$, $S(\tilde{A})$ is the crisp set of all $x\in X$ such that $\mu_A(x)>0$.

Definition 5.8 The algebraic product of two fuzzy sets $\tilde{A}\bullet\tilde{B}$ is

$$\mu_{\tilde{A}\bullet\tilde{B}}(x)=\{(x,\mu_A(x)\bullet\mu_B(x))|x\in X\} \qquad \textbf{(5 - 8)}$$

According to these definitions, a proposition P may have a fuzzy truth $f=|P|$, $0\leq P\leq 1$ and $|P|$ denotes the fuzzy truth value of P. The membership function and linguistic variables are used to ascertain the fuzzy truth of a proposition. A given bounded interval on a real axis, which represents a dimension such as temperature T, may be assigned linguistic variables such as LOW, MEDIUM, and HIGH. They are relative, but for a given situation, each of the linguistic variables has a degree of truth for any input e.g. a temperature value such as T = 100°C. The linguistic variables have no hard delimiters between them. Thus consider MEDIUM and HIGH, then T = 100°C is true to some extent for MEDIUM and HIGH. Such a description in terms of linguistic variables such as medium and high permits use of propositions such as

$$P\equiv\text{"}\left(T=T_0\right)\text{ is MEDIUM,"}\quad Q\equiv\text{"}\left(T=T_0\right)\text{ is HIGH"}$$

For example, if $|P| = 0.5$ and $|Q| = 0.82$, this suggests that "$(T = T_0)$ is HIGH" is truer than "$(T = T_0)$ is MEDIUM." Thus, a process measurement can be represented by a fuzzy region and membership function $|Q| = 0.82$ can be interpreted as $T = T_0$ belongs to the HIGH region with membership degree 0.82.

5.3 A Fuzzy-FFNN Approach for Rule Extraction

Mathematically, fuzziness implies multi-valued variables or multi-valence. Multi-valued fuzziness corresponds to a degree of indeterminacy or ambiguity, partial occurrence of events or relations. Consequently, fuzziness can deal naturally with uncertainty. This means that process measurements expressed in fuzzy form become qualitative uncertainty in the value covered.

A FFNN consists of numerous neurons that provide a basis for computation. It can be trained to estimate sampled functions when the form of the function is not known. Here, it is applied to the generation of rules using non-incremental learning, which does not require rule filtering and consequently minimises the loss of information.

The discussion starts with the introduction of fuzzy rule generation using fuzzy operations based on the approximation signal by multi-resolution wavelet decomposition. A model is then constructed based on fuzzy concepts and the FFNN. A method of measuring the reliability of rules generated by the fuzzy-FFNN is then developed.

5.3.1 Fuzzy rule generation by fuzzy operation

Using the approximate signals for the trend created by wavelet multi-resolution analysis in section 3.3, the rule generation procedure based on fuzzy operation developed by Wang and Mendle (1992) is considered. The limitations of this method are discussed later.

Figure 5-1 shows the 4^{th} scale approximate signal of the wavelet multi-resolution analysis in Figure 3-6, which is the trend in a reactor temperature T_{re} for a 15% decrease in feed stock flow rate F_{fs}. Some of the extrema and points of inflexion are indicated. Other

relative trends in the operational variables are ignored for simplicity. The set of input-output data pairs can be expressed as (T_{re},...; F_{fs}).

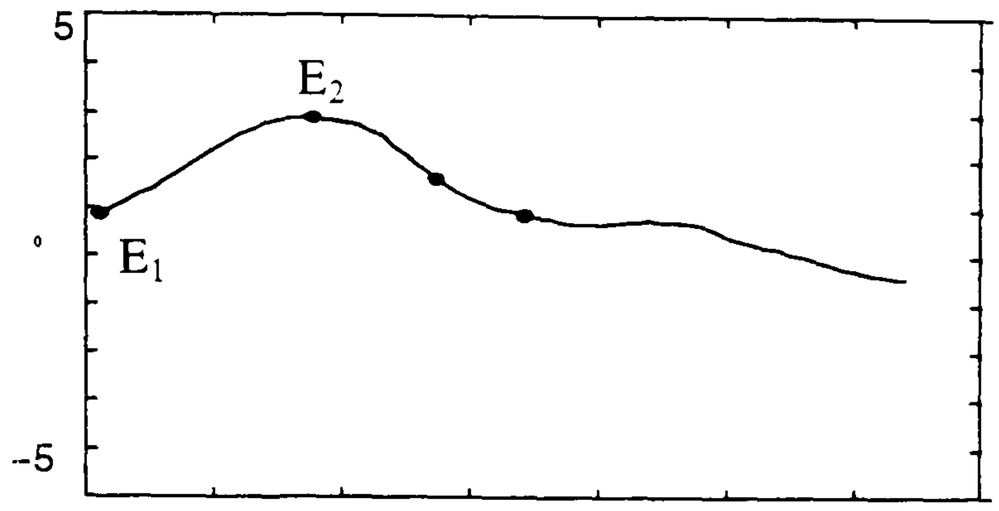


Figure 5 - 1 Approximation signal of Wavelet multi-resolution analysis

Fuzzy rule generation using fuzzy operation is based on interpreting the input-output pairs as follows:

Step 1 Divide the input and output variables into fuzzy regions. Assume that the domain intervals for T_{re} ,...; F_{fs} are $[T_{re}^-, T_{re}^+]$, ... $[F_{fs}^-, F_{fs}^+]$, respectively.

A domain interval implies that the variable can probably change in this interval, for example the set point of the reaction temperature of a FCC reactor may change by $\pm$ 5°C. Each domain interval is divided into $2K+1$ regions, for example denoted by L(Low), ML(Medium Low), Nor(Normal), MH(Medium High), and H(High) as illustrated in Figure 5-2, and each region is assigned a fuzzy membership function.

Step 2 Process the input data using the fuzzy concept. If the first maximum point E_2 in Figure 5-1 is selected for fuzzification, the value of E_2 is about 2.9, which lies in MH, i.e. medium High region with degree 0.84, or High with degree 0.16, as shown in Figure 5-3(a).

Step 3 Process the output data using the fuzzy concept. The results of F_{fs} for medium Low with degree 0.75, or Normal with degree 0.25 are shown in Figure 5-3(b).

Step 4 Assign the pair to the region having the highest values for the degree of membership.

A rule from above pair would be:

IF (reaction temperature is medium high with degree 0.84,

other operation conditions .....)

THEN (feed flow rate is medium low with degree 0.75)

Rule generation from numerical data by fuzzy operation is a straightforward one-pass procedure. As an incremental learning, a data pair will correspond to one rule. In order to obtain a compact knowledge base, a rule pruning procedure is necessary, and loss of information is unavoidable. This limitation can be overcome by combining fuzzy logic and a neural network algorithm i.e. fuzzy-FFNN.

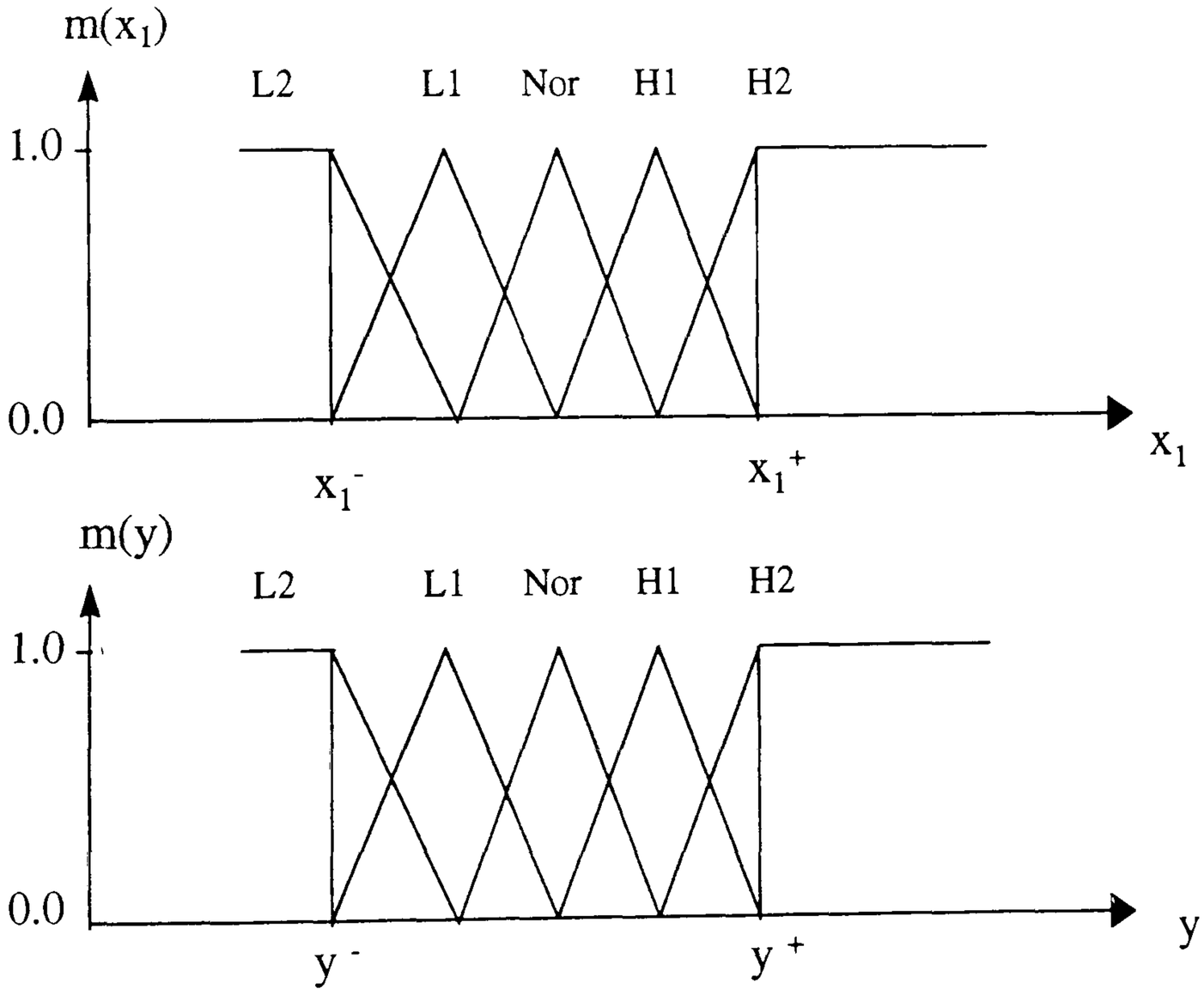
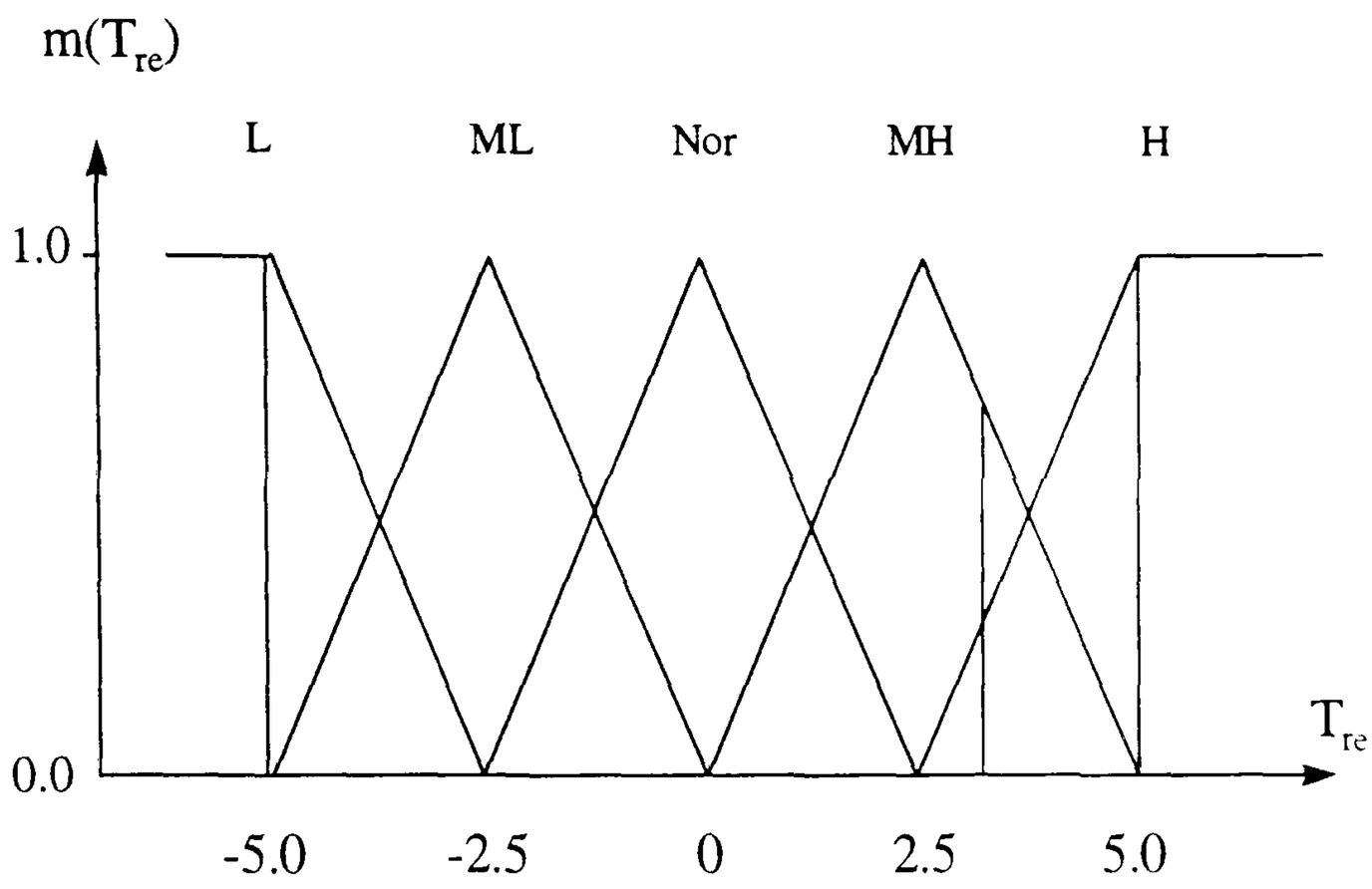


Figure 5 - 2 Divide variables in five fuzzy regions



(a)

(b)

Figure 5 - 3 Fuzzify input data (a) and output data (b)

5.3.2 Architecture and learning of a fuzzy-FFNN

Generally, a FFNN has crisp input-output values and real weight values as well. There are three possible fuzzy-FFNNs: fuzzy input-output values and crisp weights: crisp input-output values and fuzzy weights; and fuzzy input-output values and fuzzy weights. Using fuzzy weights in a fuzzy-FFNN will significantly increase the difficulty in convergence at the learning stage and it is of no benefit in rule generation to use fuzzy weights. Fuzzy input-output and a crisp weights fuzzy-FFNN structure is applied here to deal with uncertainty and convert numerical data into a qualitative representation. The training input-output data sets are fuzzified for fuzzy-FFNN learning.

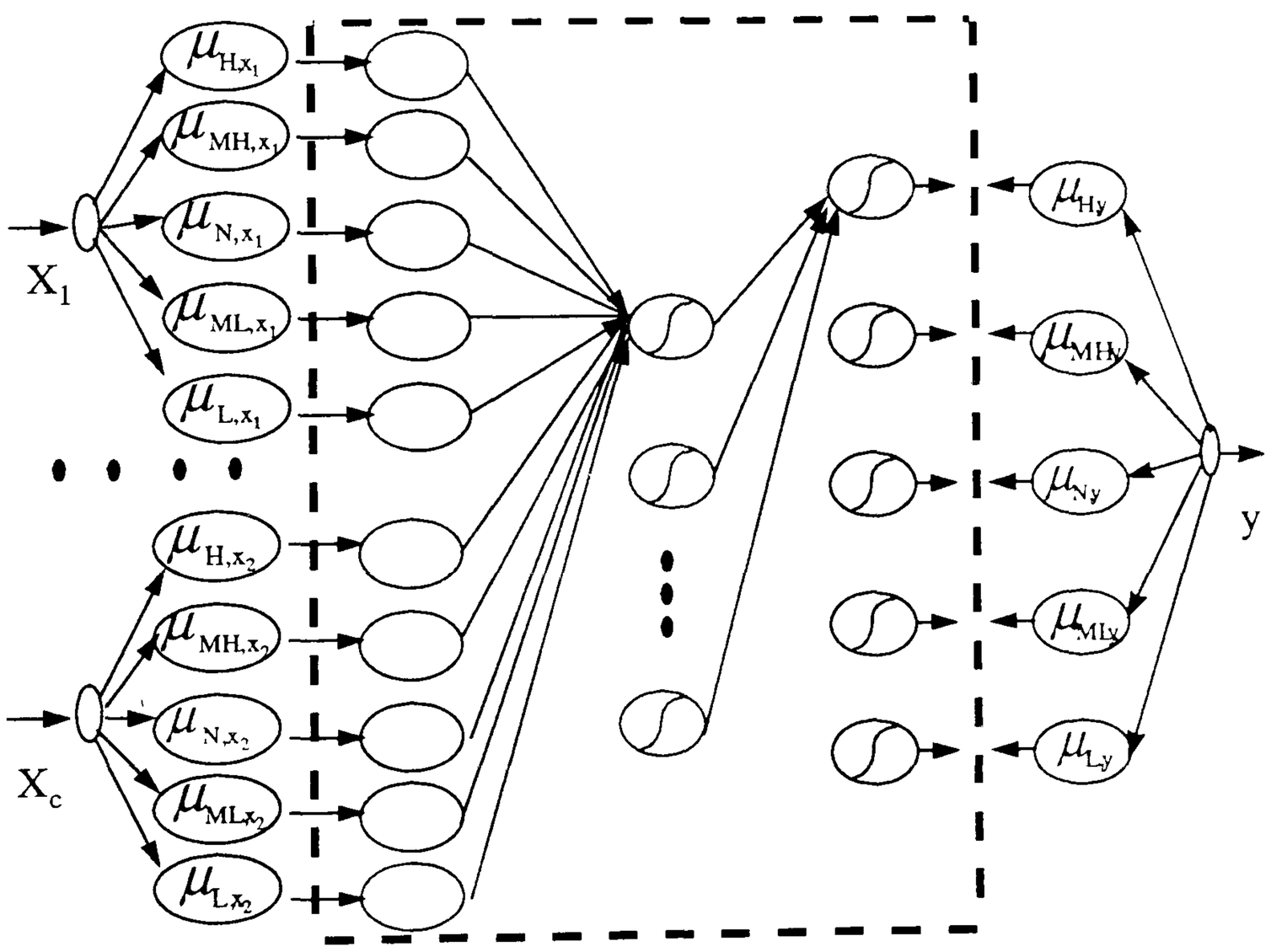


Figure 5 - 4 A fuzzy-FFNN architecture with five fuzzy regions

A fuzzy-FFNN with five fuzzy regions is illustrated in Figure 5-4. In this fuzzy-FFNN architecture, the sigmoid function

$$A(x) = \frac{1}{1+\exp(-\beta x)} \qquad (5-9)$$

is used as the activating function, and the error is measured by

$$\tilde{E} = \frac{1}{2}\sum_{k=1}^{c}(\tilde{Y}_k - \tilde{T}_k)^2 \qquad \textbf{(5 - 10)}$$

where $\tilde{Y}$ is the calculated value of fuzzy-FFNN, and $\tilde{T}$ is the target value. A fuzzy delta rule (fuzzy error back propagation) is developed for the fuzzy-FFNN training, which is illustrated below.

Given a learning rate η, the weights are adjusted using

$$w_{ji}(l+1) = w_{ji}(l) + \eta \Delta w_{ji} \qquad \textbf{(5 - 11)}$$

Δw_{ji} can be obtained from the derivatives of E

$$\frac{\partial \tilde{E}}{\partial w_{ji}} = \sum_{k=1}^{K}(\tilde{Y}_k - \tilde{T}_k)(1-\tilde{Y}_k)\tilde{Y}_k \tilde{X}_{ki} \qquad \textbf{(5 - 12)}$$

where $\tilde{Y}_k$ is calculated by

$$\tilde{Y}_k = A(\sum_{j=1}^{N} w_{ji} o_j) \qquad \textbf{(5 - 13)}$$

The hidden layer output o_j is

$$o_j = A(\sum_{i=1}^{c} w_{ij} \tilde{X}_{ki}) \qquad \textbf{(5 - 14)}$$

The criterion for stopping training depends on the overall error in Equation 5-10 being less than the current level. The following stop criterion based on fuzzy concept is applicable as well:

Suppose the support of $\tilde{T}_k$ is in the interval $[t_{k1}, t_{k2}]$ i.e. non-zero value of $\tilde{T}_k$ is only located in the interval, $1 \le k \le K$, where K is the number of fuzzification intervals. Then if $\tilde{Y}_k = \tilde{T}_k$ for all k, the support of $\tilde{E}$ is in the interval $[-\tau, \tau]$ where

$$\tau = \frac{1}{2}\sum_{k=1}^{K}(t_{k2} - t_{k1}) \qquad \textbf{(5 - 15)}$$

Let $\varepsilon > 0$ then the rule for stopping updating of the weights is when $\tilde{E}$ is inside the Θ set, where $\Theta = [-\tau - \varepsilon, \tau + \varepsilon] \times [0, 1]$.

The approach of rule generation by fuzzy-FFNN also involves the following steps:

1. Divide the input-output data into fuzzy regions according to the characteristics of the process variables;
2. Define a membership function, and fuzzify all input and output data using the membership function;
3. Construct a fuzzy neural network appropriate to the problem requirement;
4. Use a fuzzy error back propagation algorithm to train the fuzzy-FFNN.

Step 1 and 2 are similar to the step 1, 2, and 3 in 5.3.1.

This approach is non-incremental learning and takes all available data into account and generates rules immediately. Similar cases for training data sets are grouped and generate a unique rule. No redundant rule is generated if the training set is selected properly.

5.3.3 Reliability of rules

Although the above fuzzy-FFNN approach is non-incremental learning for rule generation, it is still necessary to measure the reliability of the resulting rules because the given training data set cannot be a uniform density distribution, i.e. some training regions are relatively sparse in data.

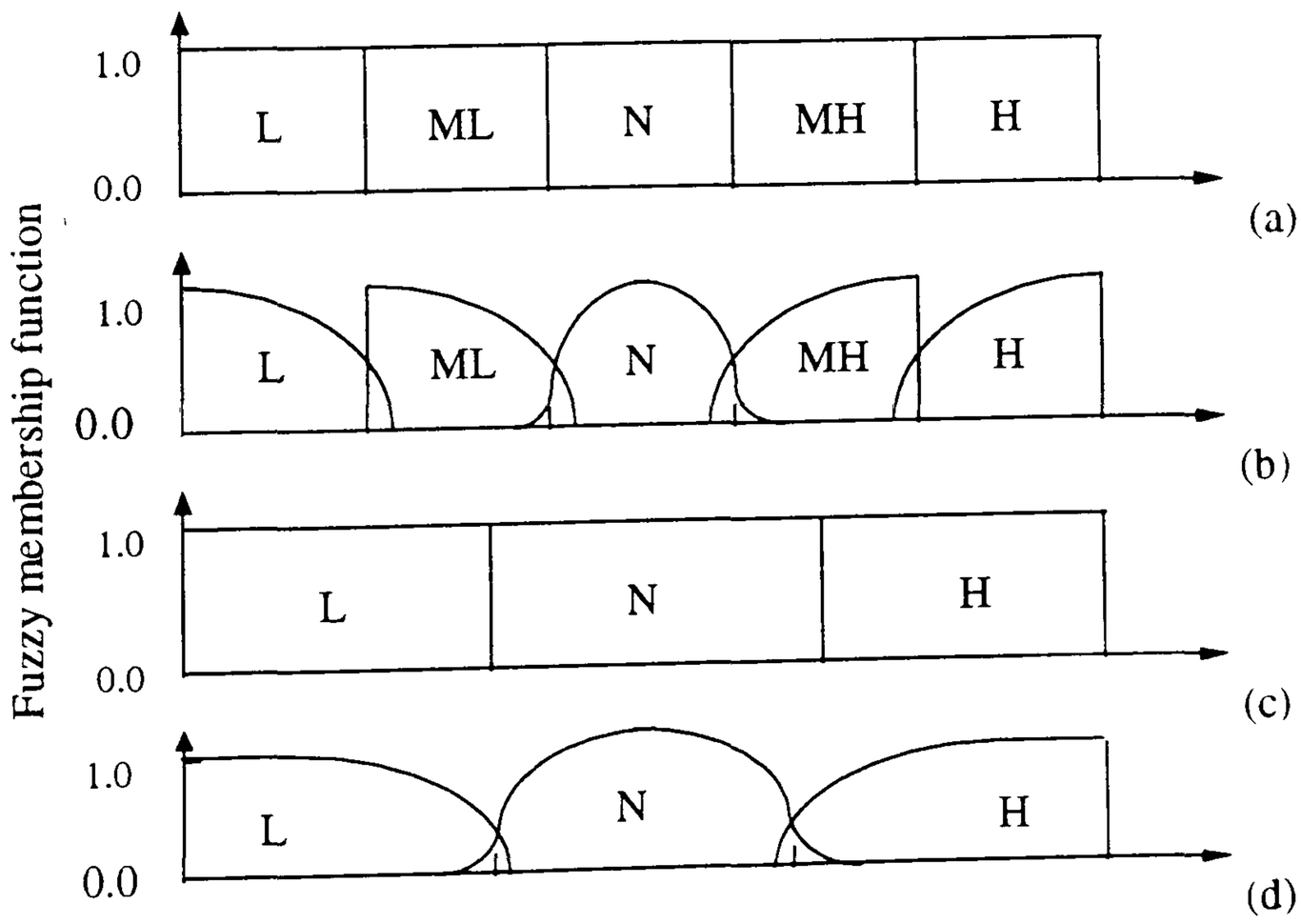


Figure 5 - 5 Typical membership functions

The fuzzy-FFNN approach can generate either of two types of rules depending on the membership functions used. The first type has the following form when the membership function is similar to (a) and (c) in Figure 5-5:

IF x_1 is High AND x_2 is Low

THEN y is medium high **(5 - 16)**

In this case, the value of a variable belonging to a fuzzy concept is unambiguous so the membership value is always one.

Another type of rule is generated when there is a membership degree, such as the functions shown in Figure 5-5 (b) and (d), for each fuzzy variable. The rules obtained in such a case are associated with a degree of membership of the following form.

IF x_1 is High ($\mu_{H,\,x_1}$) AND x_2 is Low ($\mu_{L,\,x_2}$)

THEN y is medium high ($\mu_{MH,\,y}$) **(5 - 17)**

The maximum number of rules is fixed by the fuzzy-FFNN structure. If there are NI input variables and each variable divides into NF fuzzy regions, then the maximum number of rules will be

$$\text{Number of Rules} = NF^{NI} \qquad \textbf{(5 - 18)}$$

For example, the maximum number of rules will be $5^2=25$ for a fuzzy-FFNN structure with two input variables where each variable takes five fuzzy values. This is a maximum number of rules, but the number of rules generated may be less, depending on the range of data used to derive them.

Rules generated by the fuzzy-FFNN have different reliabilities. It is necessary to discard the rules that are less reliable. As is known, the behaviour of the FFNN is strongly dependent on the distribution of the training data set. It would be expected that the prediction of the fuzzy-FFNN will also be influenced by the distribution of the training data set since the kernel is based on the FFNN. Therefore, the reliability of rules generated by the fuzzy-FFNN relies on the distribution of the training data set as well. The rules from a dense training data set space are more reliable than those from a sparse

space. This transfers to the prediction errors of the fuzzy-FFNN according to the nature of the FFNN characteristics.

On the other hand, the degree of the membership is a measure of the extent to which variables belong to the fuzzy region. The membership can be used to measure the reliability of the rule as well. The α-cut concept defined in subsection 5.2 can be applied for this purpose.

Adopting these ideas, the reliability of a rule can be measured by using the relative error of the prediction ER and its membership degree. Define a confidence factor, CF, to represent the reliability of a rule by:

$$\text{Confidence factor, CF} = \mu_{\tilde{A}_1 \bullet \tilde{A}_2 \cdots \tilde{A}_n} / \text{ER} \qquad \textbf{(5 - 19)}$$

where $\tilde{A}_i$ (i =1, 2, ...n) are fuzzy sets corresponding to variables in the rule, and the membership degree $\mu_{\tilde{A}_1 \bullet \tilde{A}_2 \cdots \tilde{A}_n}$ is calculated by Definition 5.8. An α-cut value is defined as the threshold for the rules worthy of being kept. If the $\mu_{\tilde{A}_1 \bullet \tilde{A}_2 \cdots \tilde{A}_n}$ of a rule is smaller than the α-cut value then the rule is discarded. The criterion for rule pruning can also be based on the confidence factor CF. Corresponding to the α-cut definition, a λ-cut is defined for pruning rules based on the CF value.

5.4 Case Study

The fuzzy-FFNN approach to extract fuzzy rules from numerical data has been described in section 5.3, where the algorithm and a method of measuring the reliability of a generated rule are derived. Here, operation of a heat exchanger is used as a case study to illustrate the method. It illustrates the power of the method in extracting fuzzy rules from large amounts of numerical data and also shows that it has the ability to prune weak or unreliable rules, as well as dealing with conflicting rules caused by noisy data.

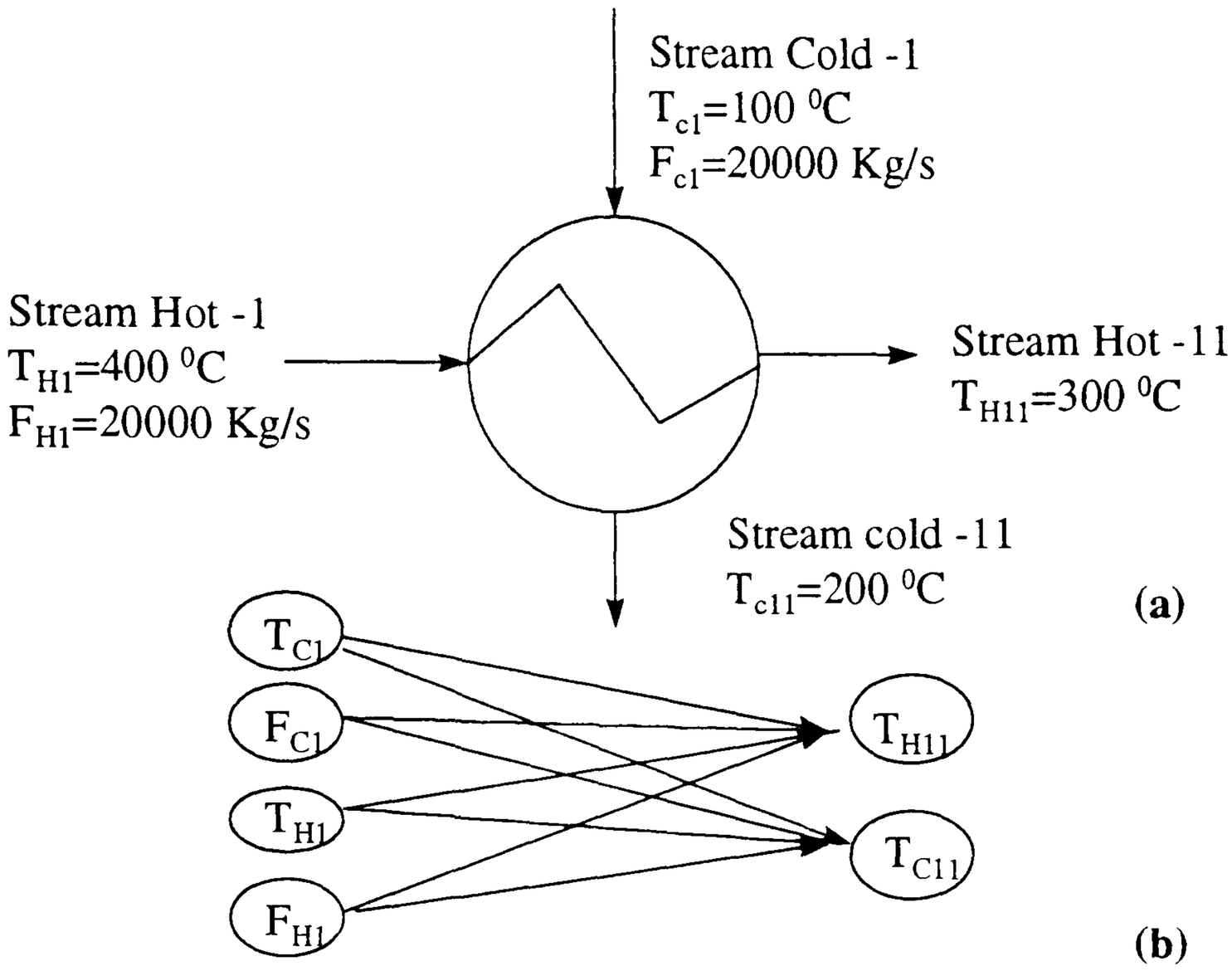


Figure 5 - 6 The shell-and-tube heat exchanger (a) and its cause-effect diagram (b)

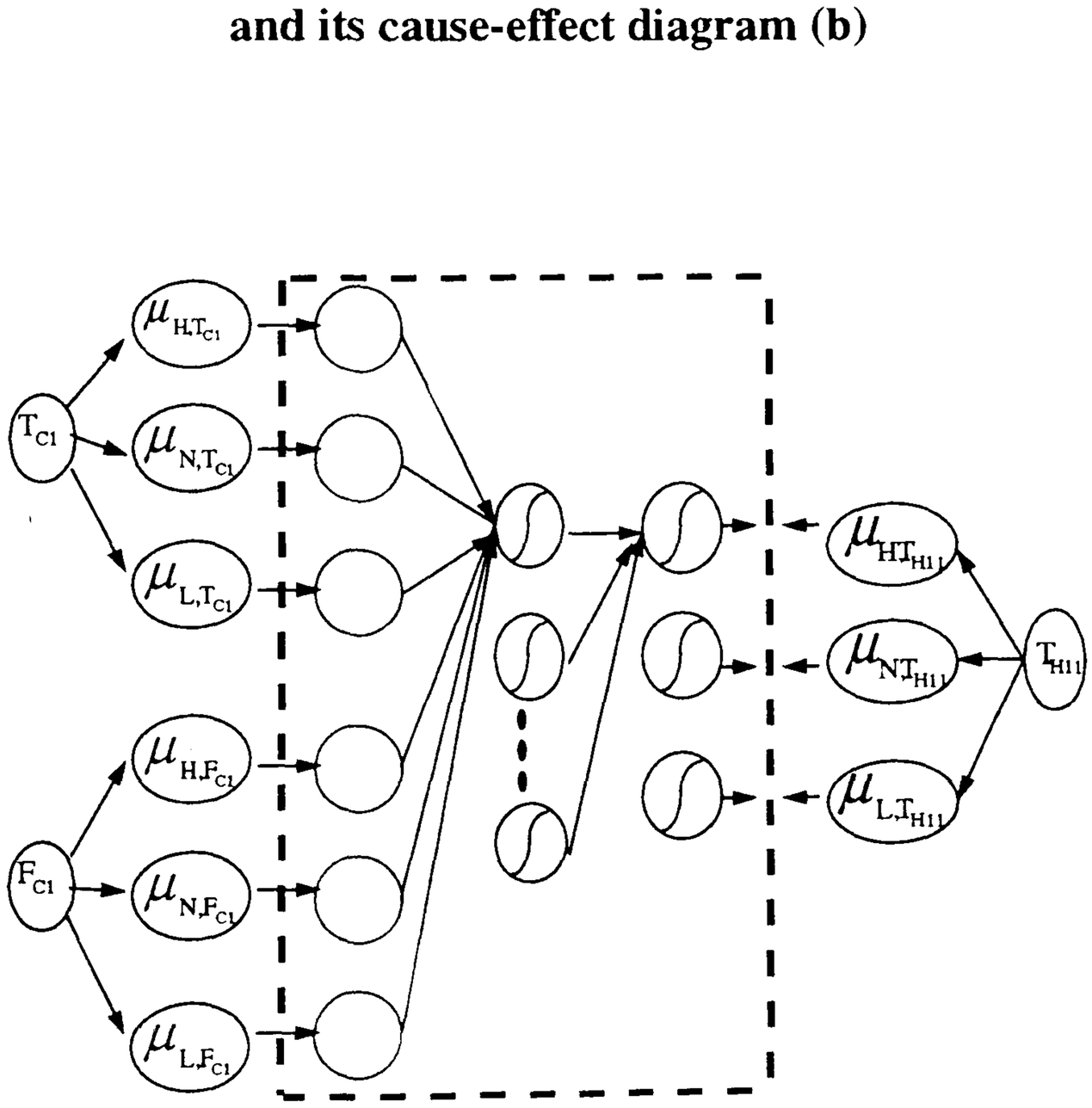


Figure 5 - 7 Fuzzy-FFNN structure with three fuzzy regions for the heat exchanger

Table 5 - 1 The ranges of low, normal and high for T_{H11}, F_{C1} and T_{C1}

	L (Low)	N (Normal)	H(High)
T_{H11} ^0C	< 285	[285, 320]	>320
F_{C1} Kg/sec	$<0.627*F_{C1,\ steady}$	$[<0.627*F_{C1,\ steady}$, $1.42*F_{C1,\ steady}]$	$>1.42*F_{C1,\ steady}$
T_{C1} ^0C	$<0.55*T_{C1,\ steady}$	$[0.55*T_{C1,\ steady}$, $1.6*T_{C1,\ steady}]$	$>1.6*T_{C1,\ steady}$

Table 5 - 2 12 groups of data patterns are designed by changing F_{C1} and/or T_{C1}

Group	Data points	$F_{C1} = F_{C1}*C1$	$T_{C1} = T_{C1}*C2$
1	(1) ~ (5)	1.0, 1.1, 1.2, 1.3, 1.4	1.0
2	(6) ~ (9)	1.45, 1.50, 1.60, 1.65	1.0
3	(10) ~ (13)	0.9, 0.8, 0.7, 0.65	1.0
4	(14) ~ (16)	0.6, 0.5, 0.45	1.0
5	(17) ~ (20)	1.0	1.2, 1.3, 1.4, 1.5
6	(21) ~ (24)	1.0	1.65, 1.7, 1.8, 1.9
7	(25) ~ (27)	1.0	0.8, 0.7, 0.6
8	(28) ~ (31)	1.0	0.5, 0.4, 0.35, 0.3
9	(32) ~ (47)	1.45, 1.5, 1.6, 1.65	1.65, 1.7, 1.8, 1.9
10	(48) ~ (59)	0.6, 0.5, 0.45	1.65, 1.7, 1.8, 1.9
11	(60) ~ (75)	1.45, 1.5, 1.6, 1.65	0.5, 0.4, 0.35. 0.3
12	(76) ~ (87)	0.6, 0.5, 0.45	0.5, 0.4, 0.35. 0.3

Table 5 - 3 Simulation data by changing F_{C1} and/or T_{C1} as shown in Table 5-2

No.	C1	C2	F_{C1}	T_{C1}	T_{H11}
1	1.0	1.0	20000	100	300.077
2	1.1	1.0	22000	100	295.955
6	1.45	1.0	29000	100	284.157
7	1.5	1.0	30000	100	282.735
10	0.9	1.0	18000	100	304.638
11	0.8	1.0	16000	100	309.720
14	0.6	1.0	12000	100	321.896
17	1.0	1.2	20000	120	306.739
18	1.0	1.3	20000	130	310.069
21	1.0	1.65	20000	165	321.727
22	1.0	1.7	20000	170	323.392
25	1.0	0.8	20000	80	293.416
28	1.0	0.5	20000	50	283.423
29	1.0	0.4	20000	40	280.092
32	1.45	1.65	29000	165	309.256
33	1.45	1.7	29000	170	311.187
41	1.6	1.7	32000	170	308.039
42	1.6	1.8	32000	180	312.037
48	0.6	1.65	12000	165	338.819
49	0.6	1.7	12000	170	340.120
50	0.6	1.8	12000	180	342.724
57	0.45	1.7	9000	170	348.987
58	0.45	1.8	9000	180	351.205
59	0.45	1.9	9000	190	353.423
60	1.45	0.5	29000	50	264.849
61	1.45	0.4	29000	40	260.988
62	1.45	0.35	29000	35	259.057
71	1.6	0.3	32000	30	252.062
72	1.65	0.5	33000	50	258.578
76	0.6	0.5	12000	50	308.879
77	0.6	0.4	12000	40	306.275
87	0.45	0.3	9000	30	317.936

Table 5 - 4 Fuzzification results of simulation data using the membership function of Figure 5-5 (c)

	F_{C1}			T_{C1}			T_{H11}		
No	L	N	H	L	N	H	L	N	H
1	0	1	0	0	1	0	0	1	0
2	0	1	0	0	1	0	0	1	0
6	0	0	1	0	1	0	1	0	0
7	0	0	1	0	1	0	1	0	0
10	0	1	0	0	1	0	0	1	0
11	0	1	0		1	0	0	1	0
14	1	0	0	0	1	0	0	0	1
17	0	1	0	0	1	0	0	1	0
18	0	1	0	0	1	0	0	1	0
21	0	1	0	0	0	1	0	0	1
22	0	1	0	0	0	1	0	0	1
25	0	1	0	0	1	0	0	1	0
28	0	1	0	1	0	0	1	0	0
29	0	1	0	1	0	0	1	0	0
32	0	0	1	0	0	1	0	1	0
33	0	0	1	0	0	1	0	1	0
41	0	0	1	0	0	1	0	1	0
42	0	0	1	0	0	1	0	1	0
48	1	0	0	0	0	1	0	0	1
49	1	0	0	0	0	1	0	0	1
50	1	0	0	0	0	1	0	0	1
57	1	0	0	0	0	1	0	0	1
58	1	0	0	0	0	1	0	0	1
59	1	0	0	0	0	1	0	0	1
60	0	0	1	1	0	0	1	0	0
61	0	0	1	1	0	0	1	0	0
62	0	0	1	1	0	0	1	0	0
71	0	0	1	1	0	0	1	0	0
72	0	0	1	1	0	0	1	0	0
76	1	0	0	1	0	0	0	1	0
77	1	0	0	1	0	0	0	1	0
87	1	0	0	1	0	0	0	1	0

Note: **L – Low, N – Normal, and H -- High**

A shell-and-tube heat exchanger is shown in Figure 5-6(a). The cause-effect diagram of variables is illustrated in Figure 5-6(b), where no leak is assumed so as to avoid complicating the discussion. Each input variable can take three possible fuzzy values, i.e. Low, Normal, and High, and F_{C1} and T_{C1} data sets are generated by fixing T_{H1} and F_{H1}. The fuzzy-FFNN architecture of this case is shown in Figure 5-7.

Firstly, fuzzy rules are generated from the simulation data using the fuzzy membership function of Figure 5-5(c). Then the fuzzy membership function with the form of Figure 5-5(d) is used to generate rules with membership degree. Using the reliability index for rules given by Equation 5-19, it is possible to prune out weak or low reliability rules. Noisy data are added to the training data set to examine how the approach deals with conflicting information.

5.4.1 Rules generated from numerical data

First, the regions covering low, normal and high are selected as indicated in Table 5-1. A simulation based on changes in F_{C1} and/or T_{C1} generates 12 groups of data patterns, as shown in Table 5-2. Simulation results for the 87 data patterns are listed in Table 5-3. Only part of the results is shown in order to illustrate the approach. Following the fuzzy rule extraction procedures described in section 5.3, the input and output variables are firstly divided into fuzzy regions based on the membership function shown in Figure 5-5(c), then all input and output variables are expressed in fuzzy terms using the selected membership functions. The data sets in Table 5-3 are converted into fuzzy form as listed in Table 5-4. Each pattern in this table is actually a different form of rule. For example, data pattern 1 represents the following rule:

IF F_{C1} is Normal and T_{C1} is Normal

THEN T_{H11} is Normal

and data pattern 57 represents the rule of

IF F_{C1} is Low and T_{C1} is High

THEN T_{H11} is High.

Patterns can be grouped when they represent the same rules, for example, data patterns 1 to 5 give the same rule so can be merged. A heat exchanger is one of the simplest operational units in chemical processes and the case has been simplified with the assumption of no leak, so the data listed in Table 5-4 are limited and it is not difficult to find rules manually by examining the table. However, this is not the case for industrial scale problem where up to millions of data sets containing noise need to be inspected. The fuzzy-FFNN rules extraction approach is valuable in such cases.

5.4.2 Simultaneous generation of rules and degree of membership

If the fuzzy membership function takes the form of Figure 5-5(b) or (d), then rules with the form of expression 5-16 are obtained. If all the data are expressed in this form and take into account the fuzzy-FFNN rule extraction approach, the number of rules obtained will be the same as the number of data patterns. This is unexpected. Using the degree of membership and FFNN training, the number of rules can be reduced.

The membership function used here has the following form:

$$y = e^{-|x|^l} \qquad (5 - 20)$$

where x is in the range [-2, 2] and l changes from 0.51 to 5.0, and $l = 1.0$ is used. The boundary values for Low, Normal and High for the three variables (F_{C1}, T_{C1} and T_{H11}) are listed in Table 5-5.

Table 5 - 5 The ranges of low, normal and high for T_{H11}, F_{C1} and T_{C1}

	L	N	H
TH11	<280	[280, 320]	>320
FC1	12000	[12000, 28000]	>28000
TC1	<40	[40, 160]	>160

Simulation results of the heat exchanger in Table 5-3 are converted to fuzzy patterns using the fuzzy membership function in Equation 5-20 and listed in Table 5-6. Again, each data pattern in Table 5-6 can be translated into a fuzzy rule. For example, data pattern 2 represents the rule: IF F_{C1} is Normal (0.61) and T_{C1} is Normal (1.0) THEN T_{H11} is Normal (0.67).

The above procedure only converts numerical data into corresponding rules, i.e. 87 data patterns become 87 rules. Data sets could be thousands when historic records are used. The above approaches are clearly difficult to use for process analysis and confuse operators. If all the rules generated from numerical data by the approach are stored, the knowledge base will become very large, which will affect reasoning speed and accuracy because of redundancy and conflicting. To overcome this, the confidence factor in Equation 5-19 is composed with a λ-cut value to filter out some less reliable rules.

For the 87 data patterns having membership functions defined by Equation 5-20, the confidence factor, CF, defined as Equation 5-19 for each pattern is obtained and shown in Table 5-7. The larger the value of CF, the more reliable the rule is. The λ-cut value described in section 5.3.2 is used to select the desired number of rules based on CF. By changing the λ-cut values, a different number of rules are obtained as shown in Table 5-8. Thus, when the λ-cut values 20, 32 rules are retained, while for 60, it is 17 as can be seen in Table 5-9. After discarding weak rules, the number is significantly reduced and a much more compact knowledge base is obtained. If there is no conflicting rule, then Table 5-9 can be considered to be the final result.

Table 5 - 6 Fuzzification results of simulation data using the membership function of Figure 5-5 (d)

	F_{C1}			T_{C1}			T_{H11}		
No	L	N	H	L	N	H	L	N	H
1	0	1	0	0	1	0	0	0.99	0
2	0	0.61	0	0	1	0	0	0.67	0
6	0	0	0.2	0	1	0	0	0.21	0
7	0	0	0.3	0	1	0	0	0.18	0
10	0	0.61	0	0	1	0	0	0.63	0
11	0	0.37	0	0	1	0	0	0.38	0
14	0	0.14	0	0	1	0	0	0	0.15
17	0	1	0	0	0.51	0	0	0.51	0
18	0	1	0	0	0.37	0	0	0.37	0
21	0	1	0	0	0	C.19	0	0	0.15
22	0	1	0	0	0	0.26	0	0	0.17
25	0	1	0	0	0.51	0	0	0.52	0
28	0	1	0	0	0.19	0	0	0.19	0
29	0	1	0	0	0.14	0	0	0.14	0
32	0	0	0.2	0	0	0.19	0	0.4	0
33	0	0	0.2	0	0	0.26	0	0.33	0
41	0	0	0.67	0	0	0.26	0	0.45	0
42	0	0	0.67	0	0	0.51	0	0.3	0
48	0	0.14	0	0	0	0.19	0	0	0.45
49	0	0.14	0	0	0	0.26	0	0	0.49
50	0	0.14	0	0	0	0.51	0	0	0.58
57	1	0	0	0	0	0.26	0	0	0.87
58	1	0	0	0	0	0.51	0	0	1
59	1	0	0	0	0	1	0	0	0.87
60	0	0	0.2	0	0.19	0	0.38	0	0
61	0	0	0.2	0	0.14	0	0.49	0	0
62	0	0	0.2	0.37	0	0	0.56	0	0
71	0	0	0.67	1	0	0	0.9	0	0
72	0	0	1	0	0.19	0	0.58	0	0
76	0	0.14	0	0	0.19	0	0	0.41	0
77	0	0.14	0	0	0.14	0	0	0.53	0
87	1	0	0	1	0	0	0	0.17	0

Table 5 - 7 Confidence factor of rules

Data points	1	2	3	4	5	...
Confidence Factor x 10	24.0147	32.2316	3.0717	.8864	.4130	...

48	49	50	51	52	53	54
0.2075	0.3547	0.7933	3.2835	0.5948	1.5431	7.9850
55	56	57	58	59	...	
9.8246	.3414	4.1989				

Table 5 - 8 Relationship between rules number and confidence factor

λ-Cut value x 10	1.0	1.5	2.0	3.0	5.0	6.0	8.0	9.0	10.0
Number of rules	48	42	32	25	21	17	11	8	6

Table 5 - 9 Fuzzy rules generated when λ-cut value is 6.0

Data patterns	**IF** F_{C1}	**and** T_{C1}	**THEN** T_{H11}
(1)	(Normal, 1.00)	(Normal, 1.00)	(Normal, 0.99)
(2)(10)	(Normal, 0.61)	(Normal, 1.00)	(Normal, 0.67)
(17)(25)	(Normal, 1.00)	(Normal, 0.51)	(Normal, 0.51)
(18)(26)	(Normal, 1.00)	(Normal, 0.37)	(Normal, 0.37)
(40)	(High, 0.67)	(High, 0.19)	(Normal, 0.55)
(47)	(High, 1.00)	(High, 1.00)	(Normal, 0.22)
(54)	(Low, 0.51)	(High, 0.51)	(High, 0.82)
(55)	(Low, 0.51)	(High, 1.00)	(High, 0.96)
(57)	(Low, 1.00)	(High, 0.26)	(High, 0.87)
(70)	(High, 0.67)	(Low, 0.37)	(Low, 0.79)
(71)	(High, 0.67)	(Low, 1.00)	(Low, 0.90)
(74)	(High, 1.00)	(Low, 0.37)	(Low, 0.87)
(75)	(High, 1.00)	(Low, 1.00)	(Low, 1.00)
(83)	(Low, 0.51)	(Low, 1.00)	(Normal, 0.28)

Table 5 - 10 Noise data are added to simulation results

	F_{C1}			T_{C1}			T_{H11}		
No	L	N	H	L	N	H	L	N	H
2	0	1	0	0	1	0	0	1	0
3	0	1	0	0	1	0	0	1	0
3*	0	1	0	0	1	0	1	0	0
4	0	1	0	0	1	0	0	1	0
5	0	1	0	0	1	0	0	1	0
6	0	0	1	0	1	0	1	0	0
7	0	0	1	0	1	0	1	0	0
7*	0	0	1	0	1	0	0	1	0
8	0	0	1	0	1	0	1	0	0
9	0	0	1	0	1	0	1	0	0
10	0	1	0	0	1	0	0	1	0
11	0	1	0	0	1	0	0	1	0
11*	0	1	0	0	1	0	0	0	1
12	0	1	0	0	1	0	0	1	0
17	0	1	0	0	1	0	0	1	0
18	0	1	0	0	1	0	0	1	0
18*	0	1	0	0	1	0	1	0	0
19	0	1	0	0	1	0	0	1	0
21	0	1	0	0	0	1	0	0	1
22	0	1	0	0	0	1	0	0	1
22*	0	1	0	0	0	1	1	0	0
23	0	1	0	0	0	1	0	0	1
28	0	1	0	1	0	0	1	0	0
29	0	1	0	1	0	0	1	0	0
29*	0	1	0	1	0	0	0	0	1
30	0	1	0	1	0	0	1	0	0
39	0	0	1	0	0	1	0	1	0
40	0	0	1	0	0	1	0	1	0
40*	0	0	1	0	0	1	1	0	0
41	0	0	1	0	0	1	0	1	0
49	1	0	0	0	0	1	0	0	1
50	1	0	0	0	0	1	0	0	1
50*	1	0	0	0	0	1	1	0	0
51	1	0	0	0	0	1	0	0	1
69	0	0	1	1	0	0	1	0	0
70	0	0	1	1	0	0	1	0	0
70*	0	0	1	1	0	0	0	1	0
71	0	0	1	1	0	0	1	0	0
79	1	0	0	1	0	0	0	1	0
80	1	0	0	1	0	0	0	1	0
80*	1	0	0	1	0	0	0	0	1
81	1	0	0	1	0	0	0	1	0

Table 5 - 11 Prediction of the fuzzy-FFNN when noise data are including in training set

No	L(Target, Predict)		N(Target, Predict)		H(Target, Predict)	
2	0	0.1	1	0.85	0	0.06
3	0	0.1	1	0.85	0	0.06
3*	1	0.1	0	0.85	0	0.06
4	0	0.1	1	0.84	0	0.06
6	1	0.81	0	0.2	0	0.01
7	1	0.82	0	0.19	0	0.01
7*	0	0.82	1	0.19	0	0.01
8	1	0.8	0	0.21	0	0.01
9	1	0.81	0	0.2	0	0.01
10	0	0.1	1	0.85	0	0.06
11	0	0.1	1	0.85	0	0.06
11*	0	0.1	0	0.85	1	0.06
12	0	0.1	1	0.84	0	0.06
17	0	0.1	1	0.84	0	0.06
18	0	0.1	1	0.85	0	0.06
18*	1	0.1	0	0.85	0	0.06
19	0	0.1	1	0.84	0	0.06
20	0	0.1	1	0.84	0	0.06
21	0	0.2	0	0.02	1	0.81
22	0	0.2	0	0.02	1	0.81
22*	1	0.2	0	0.02	0	0.81
23	0	0.2	0	0.02	1	0.8
28	1	0.8	0	0.03	0	0.06
29	1	0.8	0	0.03	0	0.06
29*	0	0.81	0	0.03	1	0.06
30	1	0.79	0	0.03	0	0.06
39	0	0.06	1	0.95	0	0.03
40	0	0.06	1	0.95	0	0.03
40*	1	0.06	0	0.95	0	0.03
41	0	0.06	1	0.94	0	0.03
49	0	0.07	0	0.01	1	0.94
50	0	0.07	0	0.01	1	0.94
50*	1	0.07	0	0.01	0	0.94
51	0	0.07	0	0.01	1	0.94
69	1	0.94	0	0.05	0	0.01
70	1	0.94	0	0.05	0	0.01
70*	0	0.94	1	0.05	0	0.01
71	1	0.94	0	0.05	0	0.01
79	0	0.01	1	0.77	0	0.24
80	0	0.01	1	0.78	0	0.24
80*	0	0.01	0	0.78	1	0.24
81	0	0.01	1	0.77	0	0.24

5.4.3 Dealing with conflicting data

Conflict is not taken into account in the above procedures for rule extraction whatever type of membership function is used. There are also no conflict rules generated because the data patterns used in rule generation are created by mathematical models. However, more often than not, data collected from processes contain noise. Therefore, conflicts, i.e. rules having the same IF part but different THEN part are likely to arise when the rules are extracted directly from a process database.

In order to examine the ability of the fuzzy-FFNN approach to deal with such conflicts arising from noisy data, some irregular patterns are deliberately introduced into the simulation data. Ten noise data patterns are introduced into the data sets in Table 5-4. The total number of cases in Table 5-4 increases from 87 to 97 and the result is summarised in Table 5-10 where noise data patterns are marked by "*". Thus, 7 and 7* have the same IF part but a different THEN, and 7* is known as a noisy data pattern. The features associated with conflicting rules are illustrated by using the data patterns from Table 5-10 as the training set and checking the predictions of the noisy data patterns against the trained fuzzy-FFNN.

Table 5-11 gives the targets and predictions of T_{H11} for the fuzzy-FFNN for all 97 cases in Table 5-10. In the case of 1, for example, the results listed in Table 5-10 are interpreted as, the target values for T_{H11} are L = 0, N = 1 and H = 0 and the predictions for T_{H11} are L = 0.1, N = 0.85 and H = 0.06.

It has been found that for all the 87 simulated data patterns, very good predictions are obtained, while in the case of all the 10 noisy data patterns the errors are large. Rules derived from such patterns need to be checked against errors based on target values and predictions. Using the λ-cut method, the number of rules can be reduced.

This means that the rules embedded in the fuzzy-FFNN do not conflict. For example, 3* has the rule obtained from Table 5-10 given by

IF F_{C1} is Normal and T_{C1} is Normal

THEN T_{H11} is Low

Although the prediction error has increased, the rule represented by the fuzzy-FFNN is still essentially correct:

IF F_{C1} is Normal and T_{C1} is Normal

THEN T_{H11} is Normal

A trained fuzzy-FFNN correctly generates rules based on either the simulation data or data with noise as shown in Table 5-12. The characteristics can be retained for a membership function of the form shown in Figure 5-5 (d).

Table 5 - 12 Rules generated by the Fuzzy-FFNN based on the simulation data or noise data

IF FC1	**and** TC1	**THEN** TH11
Normal	Normal	Normal
High	Normal	Low
Low	Normal	High
Normal	High	High
Normal	Low	Low
High	High	Normal
Low	High	High
High	Low	Low
Low	Low	Normal

5.5 Knowledge Representation and Organisation

In order to ensure that the knowledge base is in compact form, the λ-cut is used to discard weak rules and thus reduce the number of rules generated from the numerical data. Although the λ-cut can effectively select the desired number of rules, the pruning procedure does not retain all the information in the original data. Information is lost as some rules are discarded. Nor does the procedure guarantee that the knowledge base is

the most compact form, i.e. the minimum number of rules from the numerical data patterns.

If all the rules in the knowledge base are generated by a fuzzy-FFNN, no conflict rules can be guaranteed in the knowledge base as shown in section 5.4.3. However, fuzzy-FFNN training becomes computationally expensive when the volume of training data increases because the number of cases increases, since the architecture of the fuzzy-FFNN becomes very large in order to map the training data sets. In such circumstances, the fuzzy-FFNN is difficult to retrain as new cases need to be included in the knowledge base. It is difficult to maintain and update it when rules in the knowledge base are created by a unique fuzzy-FFNN. An alternative method of dealing with this issue is to divide the whole problem into several smaller items and use several fuzzy-FFNNs to map the cases. However, no-conflict rules cannot be retained if the rules in the knowledge base are from the fuzzy-FFNNs.

Rule extraction using the fuzzy-FFNN represents a fuzzy function mapping. Training a fuzzy-FFNN having a particular architecture calls for a procedure to find weights that minimise the error $\tilde{E}$ shown in Equation 5-10 so that it represents all the information in the training set with least mean square error.

The above issues can be resolved when using a group of trained fuzzy-FFNNs, rather than being based on the extracted rules used to construct the knowledge base. Thus, the domain problem is split into groups and a different architecture of fuzzy-FFNN is trained to map each group. There are several advantages of using this method:

1. A better architecture of fuzzy-FFNN can be constructed more easily although it is still a rule of thumb.
2. Using a group of smaller trained fuzzy-FFNNs to form a knowledge base makes it easier to maintain.
3. The reasoning speed can be improved.
4. Training a smaller fuzzy-FFNN is faster than a big one.

Since knowledge is represented by a group of trained fuzzy-FFNNs, the knowledge base can be organised in the form shown in Figure 5-8. ARTNET clustering is a recursive algorithm so it can be used on-line. In Figure 5-8, on-line trends or the features extracted by wavelet multi-scale analysis are used as input to ARTNET. It then derives a clustering result depending on the input. If the case is archived, the fuzzy-FFNN in the knowledge base can be found to match the reported class. If it is a new case, one of the fuzzy-FFNNs in the knowledge base can be selected to be retrained or a fuzzy-FFNN with a new architecture can be trained to accept the new information. Thus ARTNET works as a pointer in selecting a corresponding fuzzy-FFNN to map recent process trends. Only one of the fuzzy-FFNNs in the knowledge base is used to interpret the current process trend and operational conditions, therefore it is possible to guarantee that there will be no conflict because the rules are generated by a unique fuzzy-FFNN.

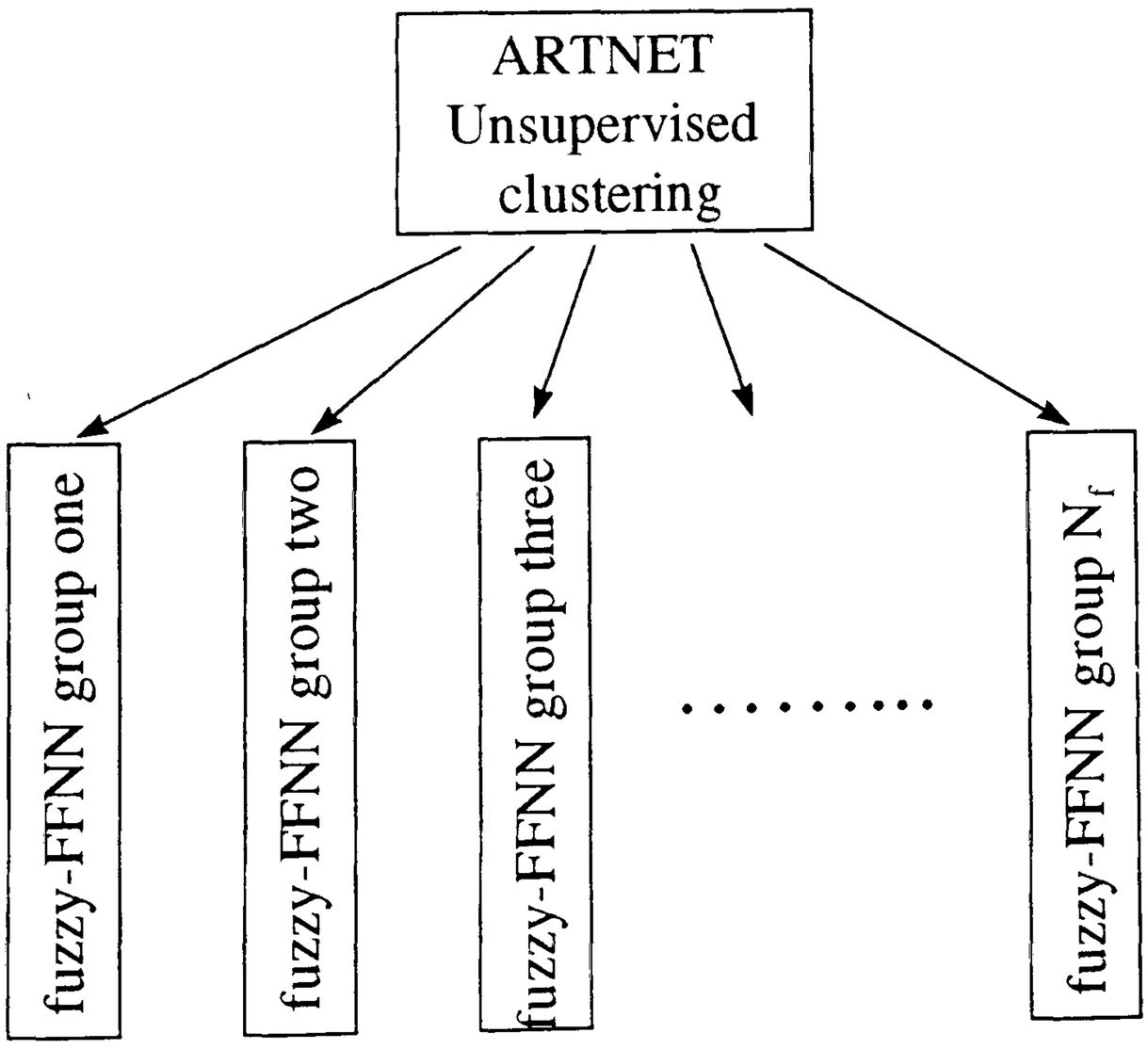


Figure 5 - 8 Knowledge base organisation

The advantages of organising a knowledge base using fuzzy-FFNNs are:

1 The knowledge base directly represented by the trained fuzzy-FFNNs ensures that it is the most compact and there are no redundant rules because there is no overlap in the fuzzy-FFNNs when the architecture is correct.

2 There is no need to consider conflict because only the corresponding fuzzy-FFNN is used for the current case when the rules are generated by an individual fuzzy-FFNN.

3 Organising the knowledge base by means of fuzzy-FFNNs to implement ARTNET clustering, the knowledge base is easy to maintain.

4 The knowledge base is easy to extend by simply adding a new trained fuzzy-FFNN which contains new information in addition to the existing knowledge base, or by retraining an existing fuzzy-FFNN which includes the new information.

5.6 Concluding Remarks

In this Chapter, a fuzzy-FFNN approach to automatically generating heuristic rules from numerical data has been developed. To keep the problem manageable, for noisy data collected over a long time, it is important for any process analysis system to be able to deal with such data effectively and efficiently. The case studies have shown that the fuzzy-FFNN is able to handle this.

By changing the fuzzy membership function, different rules can be generated including those without or with fuzzy membership values. When rules have fuzzy membership values, the λ-cut approach enables users to select the number of rules. It provides a systematic approach to generating a knowledge base from operational data of process plants.

More importantly, trained fuzzy-FFNNs are directly used to construct a knowledge base. so that it is organised by a group of fuzzy-FFNNs implemented through ARTNET clustering. The representation and organisation creates a knowledge base with the following properties:

1. compact and not redundant
2. no conflicting rules inside
3. faster reasoning
4. easy to maintain
5. easy to update

The approach has a number of novel features compared with existing methods. First of all, it uses fuzzy concepts to bridge the gap between numerical values and qualitative knowledge. Secondly, it takes advantage of back-propagation neural networks so as to make use of the large amount of data effectively. Thirdly, the feature extraction approach can be incorporated into this method.

Chapter 6

SYSTEM SYNTHESIS AND APPLICATION TO FCC

6.1 Introduction

The elements for process operation analysis have been developed and illustrated individually by simple examples in previous Chapters. Here, the operational support system is synthesised and applied to the fluid catalytic cracking (FCC) process to: (1) demonstrate the ability to interpret dynamic transients into patterns and report the location of faults in terms of rules reliably; (2) show the capability of identifying state boundaries correctly with respect to plant safety; (3) provide a well-founded representation of the intrinsic structure of the problem so as to create a knowledge base facilitating reasoning which is easy to maintain and update continuously.

In order to have a complete picture of the operational responses of the FCC process, a dynamic simulator used for training operators has been employed to generate plant data for a range of situations. The measurement values are used to synthesise the operational support system. From these individual rests, a knowledge base is completed and setting the threshold parameter in ARTNET is discussed. For new conditions, test are made to ascertain whether the fault is new or has been previously archived. The goal is to demonstrate how the system extracts features from noisy signals so that the dimensionality of the response curves can be reduced. This approach enables multiple variable inputs to be considered and avoids extrapolating the model and reasoning the

attendant uncertainty. It provides reliable operational support for processes having complicated interactions between operational variables.

6.2 The FCC Process and Associated Simulator

The fluid catalytic cracking (FCC) process shown in Figure 6-1 is a refinery process which converts heavy bottoms from the crude and vacuum distillation columns into more valuable gasoline and lighter products. It uses a preheated feed which is mixed with a high temperature slurry recycle, which comes from the bottom of the main fractionator and is injected into the riser reactor, where it is mixed with high temperature regenerated catalyst and totally vaporised. The high temperature of the regenerated catalyst provides all the sensible heat needed for vaporisation and reaction which is needed to support the endothermic cracking reactions which take place in a matter of few seconds on the catalyst. The resulting cracking reactions result in a carbonaceous coke deposit on the surface of the catalyst which deactivates it and must be burned off before the catalyst can be used again.

After the reaction has been completed, separation of catalyst and gas takes place in the disengaging zone of the reactor and any entrained catalyst is removed in the cyclones. The catalyst is returned to the stripping section of the reactor where steam is injected to remove entrained hydrocarbons. Reactor product gas is proceeded to the main fractionator for heat recovery and separation into various product streams. Wet gas from the overheads of the main fractionator (C_6 and lighter) is compressed for further separation in downstream fractionators.

Deactivated catalyst is transported from the reactor to the regenerator through the spent catalyst pipe. Air is injected into the bottom of the regenerator lift pipe to assist the circulation of catalyst. Catalyst in the regenerator is fluidised with air which is provided by the lift and combustion air blowers. Carbon and hydrogen on the catalyst react with oxygen to produce carbon monoxide, carbon dioxide and water. While most of the reactions occur in the fluidised bed, some reaction does occur in the disengaging section above the bed, where some catalyst is still present. Gas discharges from the regenerator into the cyclones where entrained catalyst is removed and returned to the bed.

The regenerator is run at a high temperature and an excess of oxygen is used to ensure that virtually all the carbon monoxide is converted to carbon dioxide before entering the cyclones. Because this is a residual FCC for a heavy feed, the amount of heat generated through burning coke in the regenerator is more than that required by the endothermic cracking reactions. Therefore it has internal and external heat exchangers to remove the excess heat.

Regenerated catalyst flows over a weir into the regenerator standpipe. The head produced by catalyst in the standpipe provides the driving force for catalyst through the regenerated catalyst pipe to the riser reactor.

The major control loops of the process are summarised in Table 6-1, they include reaction temperature, pressures of the two reactor vessels as well as flowrates of feed, air and steam and catalyst hold-up. A very important feature of the process is the very complicated dynamic interactions arising from the heat, mass and pressure transfer processes occurring in the two reactor vessels which are strongly influenced by fluidisation conditions. The control loops are safe-guarded by special routines to prevent disastrous situations arising. There are four such systems which are listed in Table 6-1. They all need the authorisation of operators, obtained by pressing one or two buttons, and are designed to prevent excessive reaction. When the safe-guard system is activated, fourteen valves have to be set closed, opened or maintained in a position which is set according to pre-defined logic.

The dynamic simulator was developed in 1992 to train process operators for commissioning the plant and has been extensively tested and shown to be very accurate, reliable and robust. It has a number of distinctive features. First, it is able to describe continuously and smoothly start-up and shutdown procedures as well as normal operation. It has a very wide range of validated operability, including conditions associated with abnormal states. Apart from faults that can be initiated randomly, twenty faults that are common to FCC processes can be reliably reproduced with high fidelity. It has been rigorously tested as part of a program of operator training over several years

and has been progressively improved. It can reproduce all operational characteristics that are commonly met.

In this study, 67 data sets generated from the simulator are used and are summarised in Table 6-2. Discussion is limited to these 67 data patterns in order to make the discussion manageable and so as not to overwhelm the user with data. The data sets include faults and disturbances arising when:

- the fresh flow rate is increased or decreased
- the preheat temperature of the mixed feed is increased or decreased
- the recycle slurry flow rate is increased or decreased
- the opening rate of hand valve V20 is increased or decreased
- the air flow rate is increased or decreased
- valve 401-ST opens from 100% decreased
- there is a cooling water pump failure
- there is a compressor failure
- there are dual faults

The data sets contain random noise. Checks have been made to ensure that the resulting noise signal has zero mean.

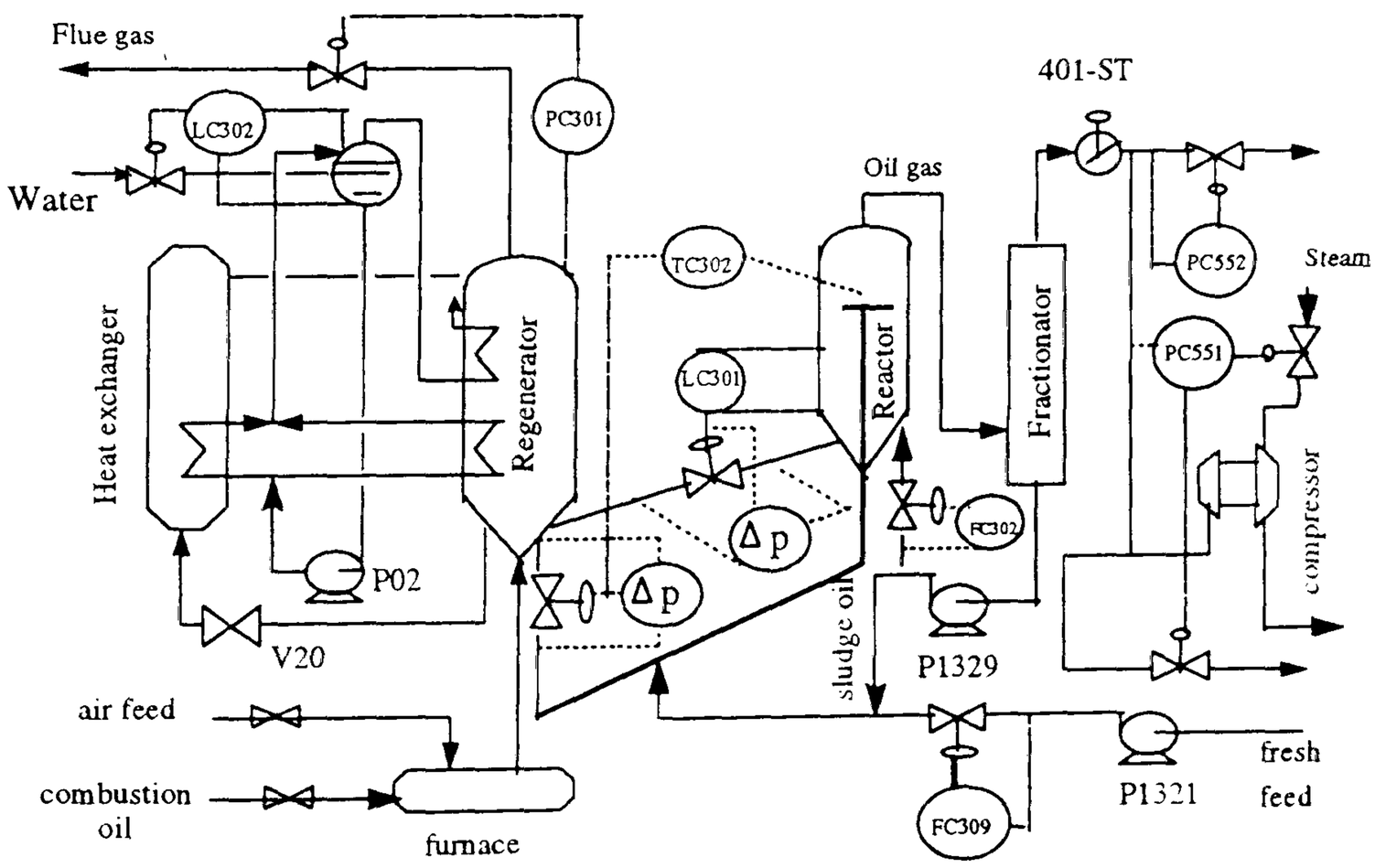


Figure 6 - 1 Schematic flow sheet of the FCC process

Table 6 - 1 The main control loops and safe-guard of the FCC process

Name of control loop or safe-guard	Note in Figure 6-2	Details
Reaction temperature control loop	TC302	Temperature control at the exit of the riser tube reactor. Set point, 513^{0}C. Manipulated variable is regenerated catalyst flow to the riser tube reactor
Regenerator pressure control loop	PC301	Pressure control of the regenerator at 1.6 kg/cm^2. Manipulated variable is the flue gas flowrate
Inlet pressure of the compressor control loop	PC551	Pressure control. This is to maintain a pressure difference of > 0.3 kg/cm^2 between the regenerator and the reactor. The manipulated variable is normally the compressor speed, but can also be switched to value on top of the fractionator. especially during start-up
Other controllers		Feed flowrates FC309 and FC305. Steam flowrate F302, water level LC302, catalyst hold-up controller LC301
Low feed flowrate (Safe-guard)		At low feed flowrate, reactions will deteriorate with increased reaction depth and secondary reactions.
Low flow of main air supply (Safe-guard)		Low flow of main air supply causes increase in regenerator temperature, poor fluidisation and even causes reverse flow of catalyst or catalyst flowing to the air compressor
Low pressure difference (Safe-guard)		A pressure difference of 0.3 kg/cm^2 between the regenerator and the reactor is normally required to prevent gas oil flowing into the regenerator because this may cause very high temperature in the regenerator and damage the regenerator. This safe-guard will activate the low feed safe guard
External heat exchanging system (Safe-guard)		When the low main air supply safe guard is activated the external heat exchanging system also needs to be cut off

Table 6 - 2 Summery of the 67 fault data sets produced on the dynamic simulator

Data sample No.	Type	Time t < 0 operation is at steady state, at t = 0 make the following step change
1 ~ 9	**1**	fresh feed increased by 10 20 30 40 50 60 70 80 90%
10 ~ 18	**2**	fresh feed decreased by 10 20 30 40 50 60 70 80 90%
19 ~ 22	**3**	preheat Temperature of mixed feed increased by 5 10 15 20 °C
23 ~ 24	**4**	preheat Temperature of mixed feed decreased by 15 10 °C
25 ~ 26	**5**	recycle slurry Flow rate increased by 70 90%
27 ~ 28	**6**	recycle slurry Flow rate decreased by 70 90%
29 ~ 32	**7**	opening ratio of hand-valve V20 increased by 5 10 15 24%
33 ~ 37	**8**	opening ratio of hand-valve V20 decreased by 10 15 25 35 55%
38	**9**	cooling water pump P-02 failure
39 ~ 43	**10**	air flow rate increased by 6.5 11.5 15 31.5 40.5%
44 ~ 49	**11**	air flow rate decreased by 3.5 8.5 28.5 38.5 48.5 53.5%
50	**12**	compressor failure
51 ~ 57	**13**	valve 401-ST opening from 100% decreased by 10 20 40 45 60 80 90%
58	**14 (1 & 8)**	fresh feed increased 65% and V20 opening rate decreased 55%
59	**15 (1 & 9)**	fresh feed flow rate increased 70% and pump p-02 failure
60	**16 (2 & 9)**	fresh feed decreased 65% and preheat of mixed feed increased 5 °C
61	**17 (10 & 12)**	air flow rate increased 9.5% and compressor failure
62	**18 (10 & 13)**	air flow rate decreased 9.5% and valve 401-ST opening decreased 35%
63	**19 (9 & 12)**	pump P-02 failure and compressor failure
64	**20 (2 & 13)**	fresh feed decreased 75% and valve 401-ST opening decreased 20%
65*	**21 (2 & 13)**	fresh feed decreased 65% and cooling water pump p-02 failure
66*	**1**	fresh feed increased by 65%
67*	**2**	preheat temperature of mixed feed increased by 5 °C

Note: * used for system evaluation

6.3 System Synthesis

Sets 1 to 64, are used for training to establish the basic capability of the operational support system. Fuzzy-FFNNs are used to represent knowledge of the data sets. This is then used by ARTNET to organise the knowledge base and fuzzy-FFNNs. The advantage of the approach is illustrated by comparing it with a single fuzzy-FFNN method model the 64 data sets. To make it possible for the system to avoid extrapolation, it is important to use the correct distance threshold. This matter is considered in some detail.

6.3.1 System description

The schematic diagram of the operational support system is shown in Figure 6-2. Essentially it comprises three parts: wavelet base pre-processing which extracts features and generates approximations of the process measurements; ARTNET pattern clustering based on the features extracted from the signals; the knowledge base represented by trained fuzzy-FFNNs using the approximations of the process measurements. The interface provides a basis for system management and displays the results of reasoning.

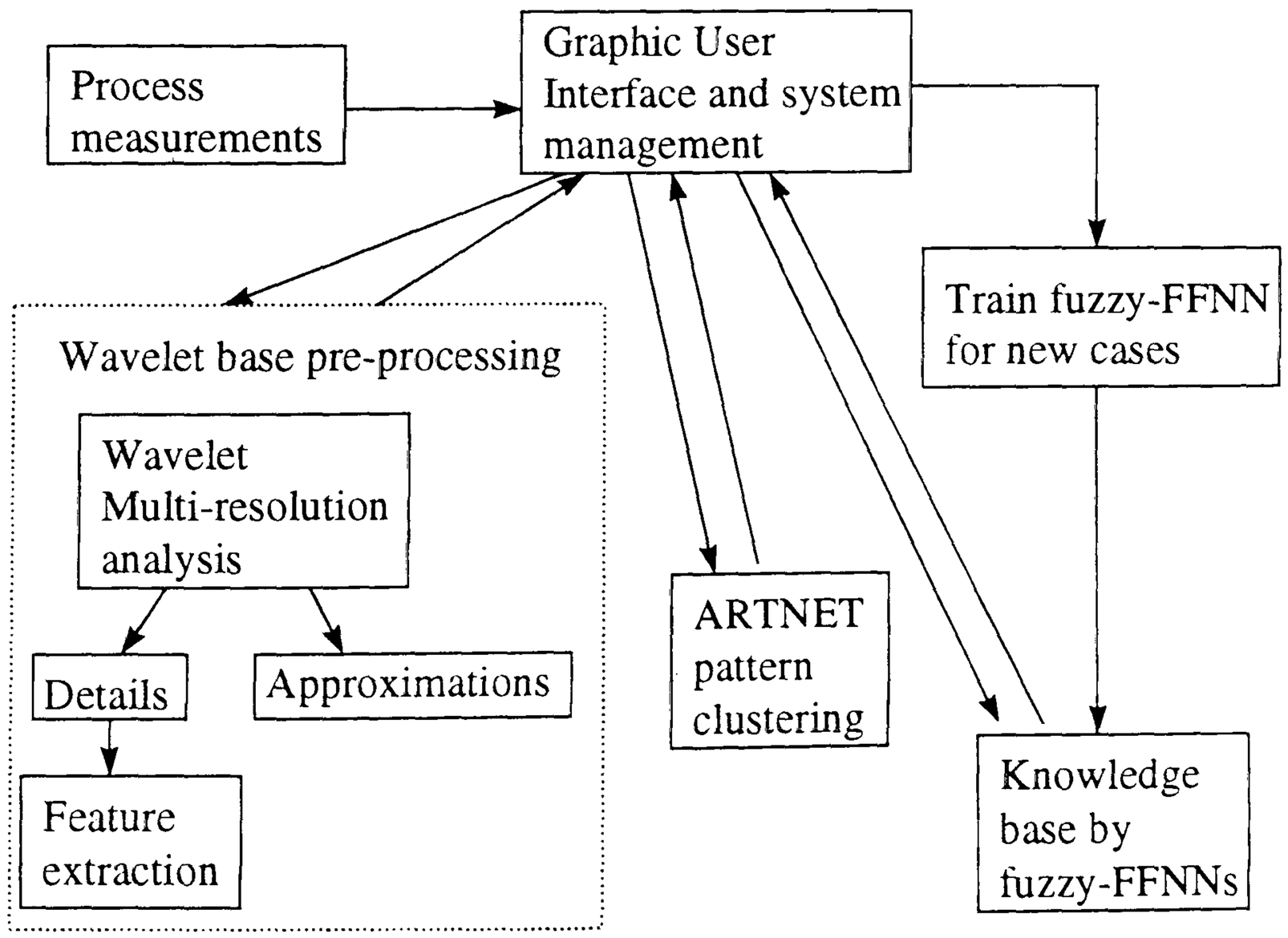


Figure 6 - 2 Schematic diagram of the system

The process measurements are input to a wavelet base signal pre-processing element in order to decompose the measurements into detailed and approximate representations. The extrema of the detail signal components correspond to irregularities and singularities in the source signal where a special wavelet basis function or filter bank is used. After removing the noise components, the remaining output of the extrema are regarded as distinctive features of the measurement. The advantage of wavelet based feature extraction is that it reduces the dimensionality of the data and removes noisy components at the same time. Thus, the signal pre-processing element simultaneously generates features and provides an approximation of the measurement based on a small finite set of components.

The features obtained by wavelet analysis are then used as inputs to ARTNET for pattern clustering. In order to take account of the interaction variables of the process, as well as enhancing the ability to discriminate between faults, multiple operating variable inputs are used. ARTNET indicates the type of class which has been archived or reports a new pattern when the input data sets are different, in some sense, from those already on file. The new pattern is then used to alert operators that the process is moving in a new direction. This ensures that plant safety is not compromised, even if the reported pattern does not ultimately result in a fault.

The knowledge base is represented by a group of trained fuzzy feed-forward neural networks (fuzzy-FFNNs). These are used to generate rules from recorded numerical values and the knowledge embedded in the structure and weights of the network. Depending on the class finally reported by ARTNET, the knowledge base is able to provide an associated cause-effect relationship. For a new pattern, it will be either included in an existing group by retraining the corresponding fuzzy-FFNN if the new data pattern is similar to the group, or it will create a new group using a new fuzzy-FFNN if it is different from existing cases.

A knowledge hierarchy in respect of process operation is then constructed by ARTNET and the fuzzy-FFNN. ARTNET provides a pointer with respect to usage, depending on

the current state of the process. It clusters cases based on the features derived from wavelet based signal pre-processing.

6.3.2 Knowledge base representation and organisation

Expert systems make use of a knowledge base made up of rules. Here, the knowledge base is created and organised by fuzzy-FFNNs which are able to generate rules from numerical data. Conflict is the vital issue from which rule-based expert systems suffer. This issue is avoided using the novel technique proposed in this study which generate and organise the knowledge using ARTNET and fuzzy-FFNNs. This particular case study clearly illustrates the advantages of this approach

Knowledge about these system can be organised by dividing the 64 data sets into six groups, where each group has similar characteristics. Table 6-3 reports the result of such a grouping where, for example, data sets No 29 to 37 are in one group because they are similar fault types related to changes of the hand valve V20 opening ratio. All equipment failure and dual fault types are also grouped because they are different from the rest. Six fuzzy-FFNNs, denoted by fuzzy-FFNN 1 to 6, have been trained using these data groups individually. For example, fuzzy-FFNN 1 is trained using data sets in group 1 which contains data Nos 1 to 18, as shown in Table 6-3. The knowledge is represented by the six trained fuzzy-FFNNs and has been organised as a hierarchy using ARTNET and fuzzy-FFNNs, as shown in Figure 6-3.

Table 6 - 3 Grouped 64 data sets for fuzzy-FFNN training

	Fault types	Data sets No.
group 1	1, 2	1 ~ 18
group 2	3, 4, 5, 6	19 ~ 28
group 3	7, 8	29 ~ 37
group 4	10, 11	39 ~ 49
group 5	13	51 ~ 57
group 6	9, 12, 14, 15, 16, 17, 18, 19, 20, 21	38, 50, 58 ~ 64

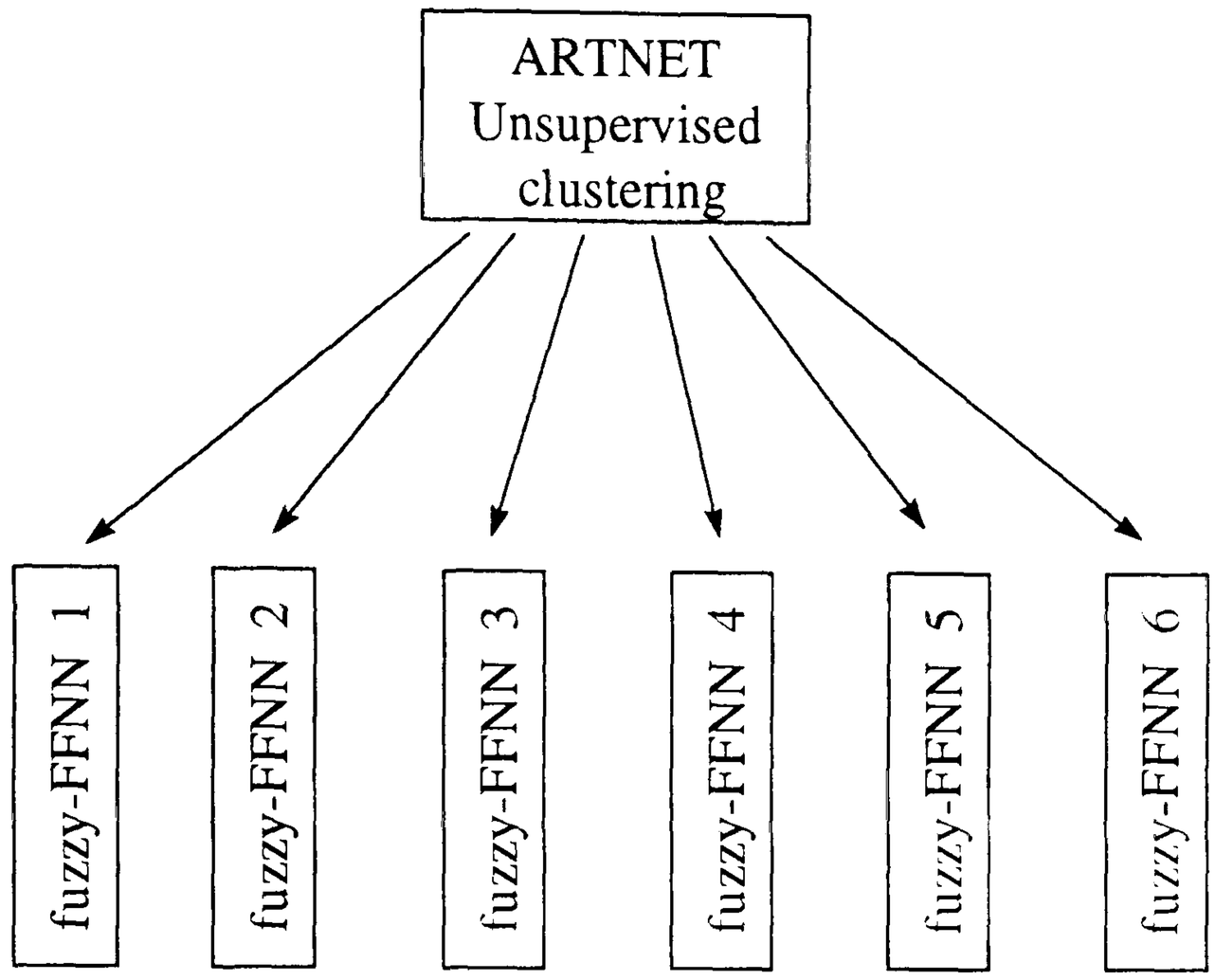


Figure 6 - 3 Knowledge base organised by ARTNET and fuzzy-FFNNs

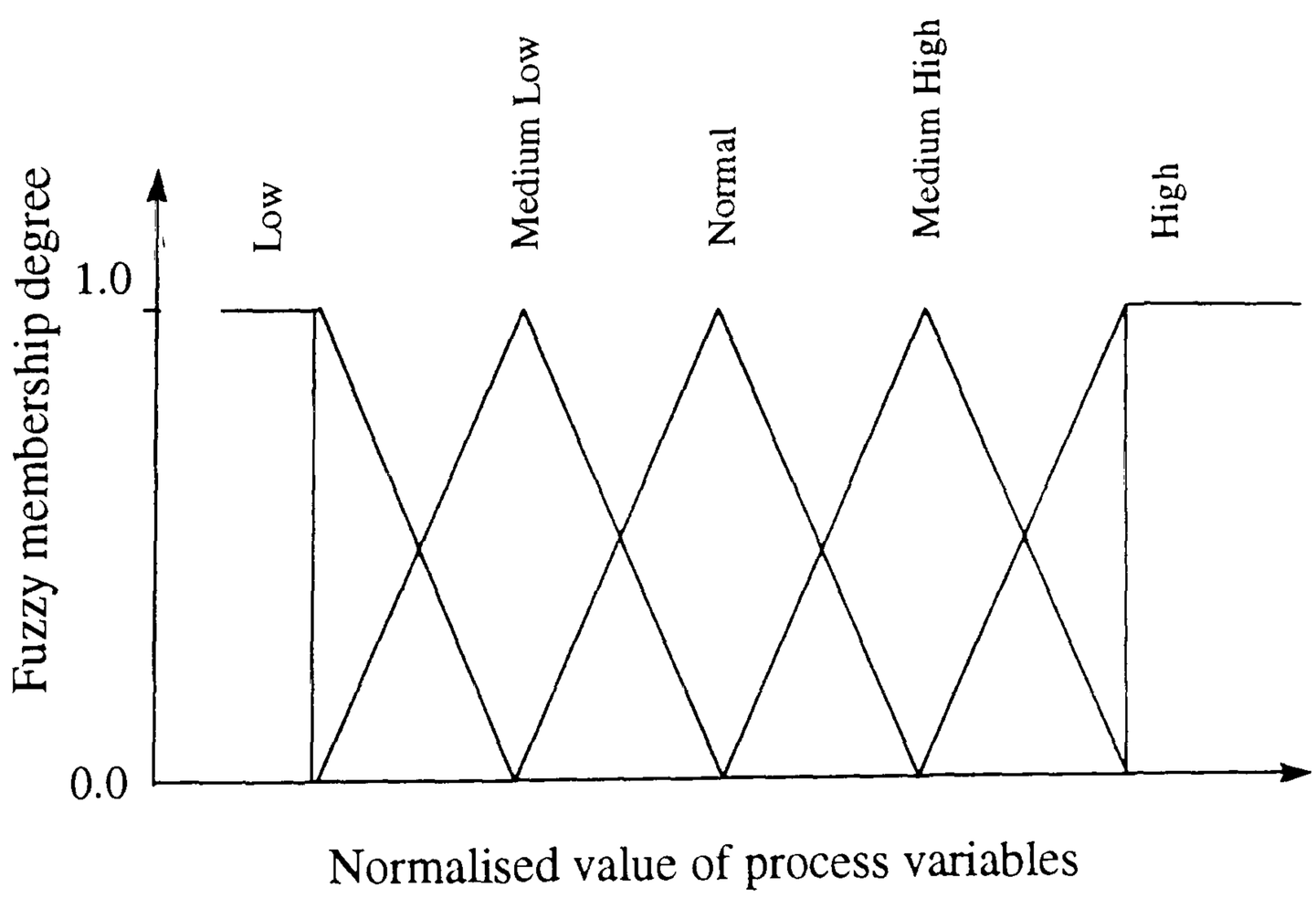


Figure 6 - 4 Fuzzy membership function used for input variable fuzzification

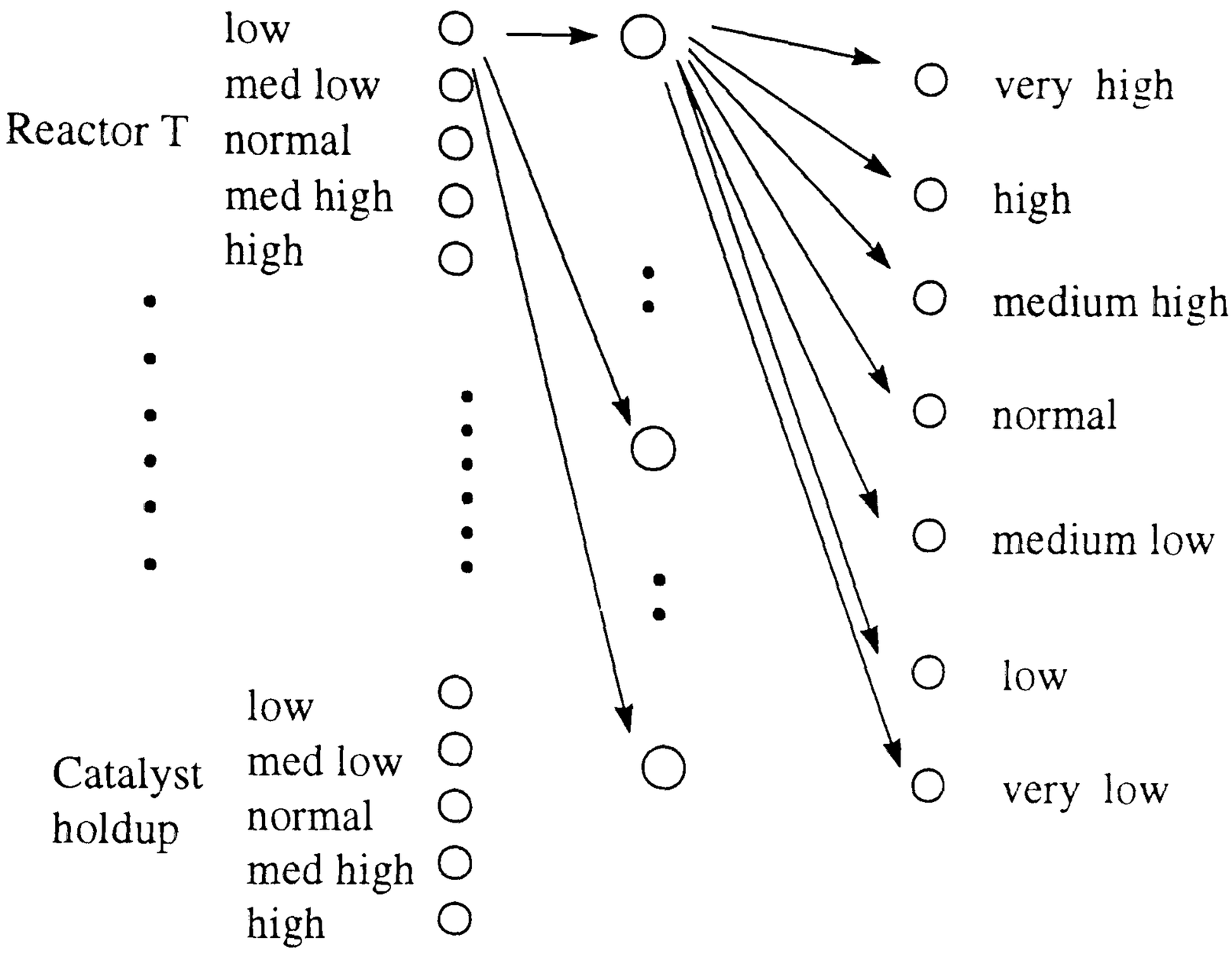


Figure 6 - 5 The fuzzy-FFNN architecture for group 1 of Table 6-4

Table 6 - 4 Divide output variables of different groups into fuzzy regions

	Fuzzy regions of output variable
group 1	fresh feed (very low, low, medium low, normal, medium high, high, very high)
group 2	preheat T (low, medium low, normal, medium high, high) slurry F (low, medium low, normal, medium high, high)
group 3	V20 open rate (low, medium low, normal, medium high, high)
group 4	air F(low, medium low, normal, medium high, high)
group 5	401-ST open rate (low, medium low, normal, medium high, high)
group 6	(P-20 failure, compressor failure, fault type 14, 15, 16, 17, 18, 19, 20, 21)

Figure 6-4 shows the fuzzy membership function used for input variable fuzzification. Each variable is split into five fuzzy regions, low, medium low, normal, medium high, and high. Input variables are represented by a vector having five elements. Each variable has only one non-zero element in the vector. For example, the reactor temperature 498°C has membership degree 0.9 in normal, 0.01 in medium high, and zero in all other regions. The variable for the region having a maximum is used to define the fuzzy vector [0 0 0.9 0 0]. Other input variables are fuzzified in the same way. Consequently, there are 40 input nodes for the fuzzy-FFNNs, because eight variables are involved for each fault and each variable is a vector having five elements.

Unlike the fuzzification used for input variables, output variables cannot be fuzzified in the same way because the assignment is problem dependent. For example, the output of fuzzy-FFNN 6 needs 10 nodes to include pump P-20 failure, compressor failure and so on, while the output node numbers of fuzzy-FFNN 3 is five because the output variable in group 3 is divided into five fuzzy regions, as can be seen by reference to Table 6-4. Based on the input and output variables fuzzification, a fuzzy-FFNN for each data group can be constructed. Figure 6-5 shows the fuzzy-FFNN architecture for group one but the others are similar. The activation functions for the neurons in both the hidden and output layers are sigmoids. The 64 data sets can then be represented by six trained fuzzy-FFNNs.

It is possible to map the 64 data sets and fault types using a single fuzzy-FFNN. However, this gives rise to difficulties especially if there is a long training time, poor mapping and inconvenient updating. This can be seen by comparing the six fuzzy-FFNNs representation with a single fuzzy-FFNN. In the following explanation, **single network** refers to mapping the 64 data sets using a single fuzzy-FFNN, while fuzzy-FFNNs 1 to 6 are the networks for data groups 1 to 6. Table 6-5 gives the architecture of the networks and the corresponding training time. Using a single network, the architecture in terms of nodes in the hidden layer has to be larger because it is used to map more information. In this case study, the hidden layer consists of 60 nodes. The output has 65 nodes because there are 13 types of faults and each fault variable has five fuzzy regions. The time for training such a network on a Sun Ultra1 was 68 minutes for 10000 learning cycles with

0.2 learning rate. On the other hand, training time is significantly reduced when using six fuzzy-FFNNs as shown in the Table 6-5.

Table 6 - 5 Training time of using single network to map 64 data sets and that of using fuzzy-FFNN groups

	Network structure	Number of training data	Training time (minutes)
Single network	40-60-65	64	68
Fuzzy-FFNN 1	40-10-7	18	3
Fuzzy-FFNN 2	40-10-10	10	5
Fuzzy-FFNN 3	40-5-5	10	2
Fuzzy-FFNN 4	40-5-5	11	2
Fuzzy-FFNN 5	40-5-5	7	2
Fuzzy-FFNN 6	40-10-10	8	6

Training: 10000 cycles with 0.2 learning rate

Table 6 - 6 Mapping of networks in Table 6-5 after training

	Convergence error	Fault incorrectly identified
Single network	0.02	24, 38
Fuzzy-FFNN 1	0.0022	None
Fuzzy-FFNN 2	0.0018	None
Fuzzy-FFNN 3	0.0017	None
Fuzzy-FFNN 4	0.0015	None
Fuzzy-FFNN 5	0.0011	None
Fuzzy-FFNN 6	0.0020	None

Table 6-6 shows the convergence error as well as the incorrect identification of faults in thetraining sets for each of the networks. Using a single network to map all 64 data sets not only takes a long time to train but also results in poor results since two faults (No 24 and 38) are incorrectly identified. The training time is much less and no wrong identification of faults occurs when the six fuzzy-FFNNs are used to represent the 64 data sets separately. It is clear that splitting the problem into groups and mapping them using fuzzy-FFNNs rather than a single network makes the representation more flexible and reliable.

Using fuzzy-FFNNs to represent the problem domain also benefits system maintenance and updating. If a new data set related to one of the groups and changes it has to be included in the knowledge base, only fuzzy-FFNN 4 needs to be retrained when using six fuzzy-FFNNs to represent the problem. If the case is different from the existing set, then a new fuzzy-FFNN may be used. For single network representation, it takes more than an hour to retrain because all data sets including archived and new data sets have to be used in retraining. This becomes worse as the number of cases increases.

As it has been shown in section 5.4.3, fuzzy-FFNNs can embed knowledge in the structure and weights and so generate no-conflict rules. However, this is only true when rules are generated by a fuzzy-FFNN. Consequently, the fuzzy-FFNNs cannot guarantee to generate no-conflict rules if they are used to represent the knowledge. The drawback can be overcome by combining ARTNET and fuzzy-FFNNs to represent and organise the knowledge base. Here, ARTNET clusters an input pattern into a class which corresponds to an trained fuzzy-FFNN, which provides rules about the input pattern. For each input of the system, only one fuzzy-FFNN is used to generate rules, therefore no-conflict rules are obtained.

6.3.3 Distance threshold setting in ARTNET

The distance threshold is a tuning parameter in ARTNET which is a sensitive factor in determining the effectiveness of the clustering. It is therefore important to find the appropriate distance threshold to achieve optimal performance of the system.

As discussed in section 4.3.4, ARTNET groups fewer patterns into a class i.e. there are more classes for the same number of patterns when the distance threshold is small. In this situations, ARTNET clusters a pattern into a new class and reduces the risk of overlapping states when classifying at operational state boundaries. However, it is necessary for ARTNET to store more exemplars to represent the archived classes. Since ARTNET is concerned with competitive learning, it clusters an input by comparing all the archived classes, so increasing the number of archived classes will slow down the classification of ARTNET.

Generally, a FFNN creates a non-linear mapping which has pattern identification abilities (Turner et al 1996) however it cannot handle extrapolation. ARTNET avoids extrapolations but is not able to deal with non-linear mapping. Clearly, the threshold value in ARTNET should be (1) small enough to form a suitable clustering boundary to avoid overlapping operational states and so avoid extrapolation based on system reasoning; (2) as large as possible to reduce the number of archived classes in ARTNET so it can share the identification tasks with fuzzy-FFNNs because knowledge is represented and organised by both ARTNET and fuzzy-FFNNs

These two considerations have been used to select a value of the threshold based on the search algorithm described in section 4.3.4. Alternatively it can be done by trial and error. For $\rho = 0.8$, all 64 data sets are identified as individual patterns and this is clearly the largest value which separates all data sets into individual classes. As the threshold value increases, some samples merge into a class. The results of increasing the threshold are shown in Table 6-7. As can be seen in Table 6-2, data sets 56 and 57 representing valve 401-ST going from 100% opening to 80% and 90% are in a single class when ρ = 1.0. Data sets No. 5 and 7 refer to fresh feed increasing by 50% and 70%, No. 25 and 26 to recycle oil flow rate increasing by 70% and 90%, and samples No. 27 and 28 to recycle oil flow rate decreasing by 70% and 90%. These are clustered into three classes, which is reasonable.

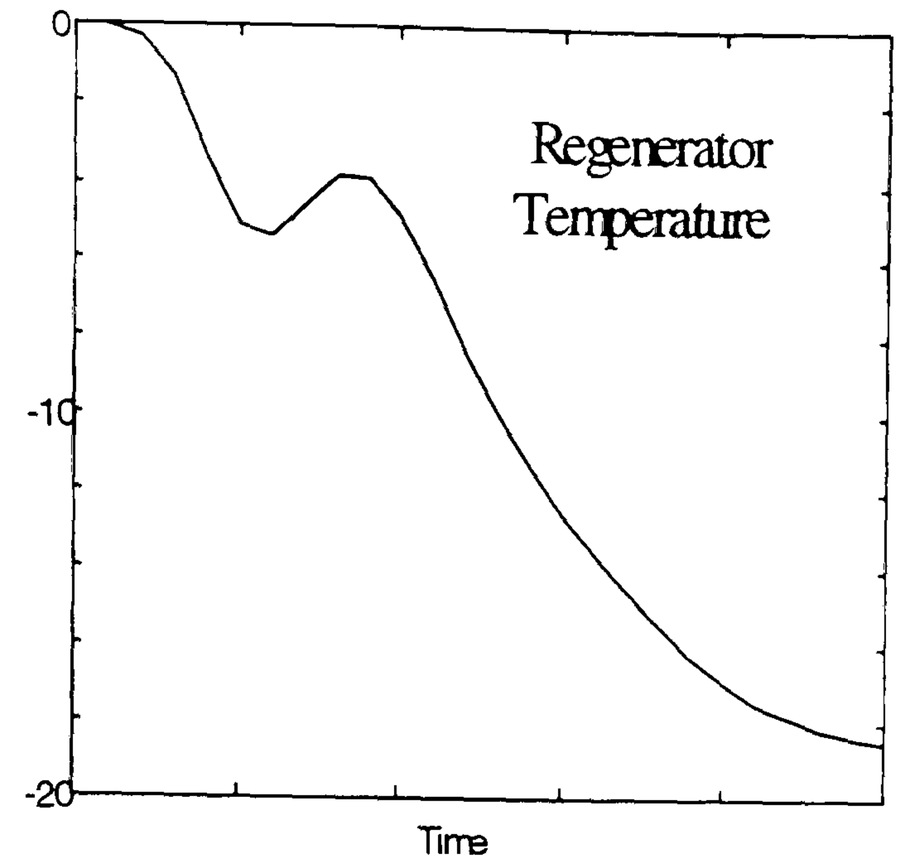
Table 6 - 7 The 64 training data sets clustered by ARTNET in different distance threshold

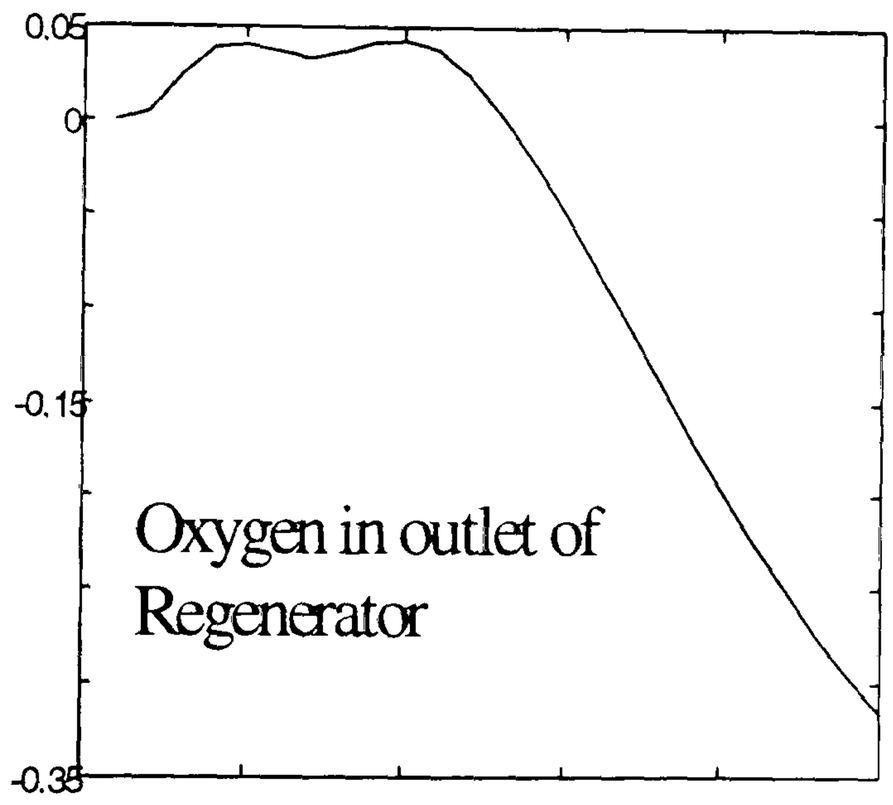
Distance threshold ρ	Number of patterns identified	Grouping of data sets
0.8	64	
1.0	63	[56 57]
2.0	60	[5 7] [25 26] [27 28] [56 57]
3.0	57	[5 7] [19 20 23 24] [25 26] [27 28] [56 57]
4.0	55	[5 6 7 8] [19 20 21 23 24] [25 26] [27 28] [56 57]
4.5	49	[3 4 5 6 7 8 9] [19 20 21 22 23 24] [25 26] [27 28]
		[35 36] [56 57]
5.0	48	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29]
		[25 26] [27 28] [35 36] [56 57]
6.0	47	[3 4 5 6 7 8 9] [19 20 21 22 23 24 29 52]
		[25 26] [27 28] [35 36] [56 57]
12.0	19	[1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 39 40 41 42 43 44 45 46 52]
		[16 17 18] [50 51 53] [56 57]

Table 6 - 8 The report of the ARTNET when the distance threshold is 4.5

Class	Corresponding data set	Class	Corresponding data set	Class	Corresponding data set
1	1	**18**	31	**35**	49
2	2	**19**	32	**36**	50
3	3 4 5 6 7 8 9	**20**	33	**37**	51
4	10	**21**	34	**38**	52
5	11	**22**	35 36	**39**	53
6	12	**23**	37	**40**	54
7	13	**24**	38	**41**	55
8	14	**25**	39	**42**	56 57
9	15	**26**	40	**43**	58
10	16	**27**	41	**44**	59
11	17	**28**	42	**45**	60
12	18	**29**	43	**46**	61
13	19 20 21 22 23 24	**30**	44	**47**	62
14	25 26	**31**	45	**48**	63
15	27 28	**32**	46	**49**	64
16	29	**33**	47		
17	30	**34**	48		

Note: Class stands for clustering pattern class; No. for data set number





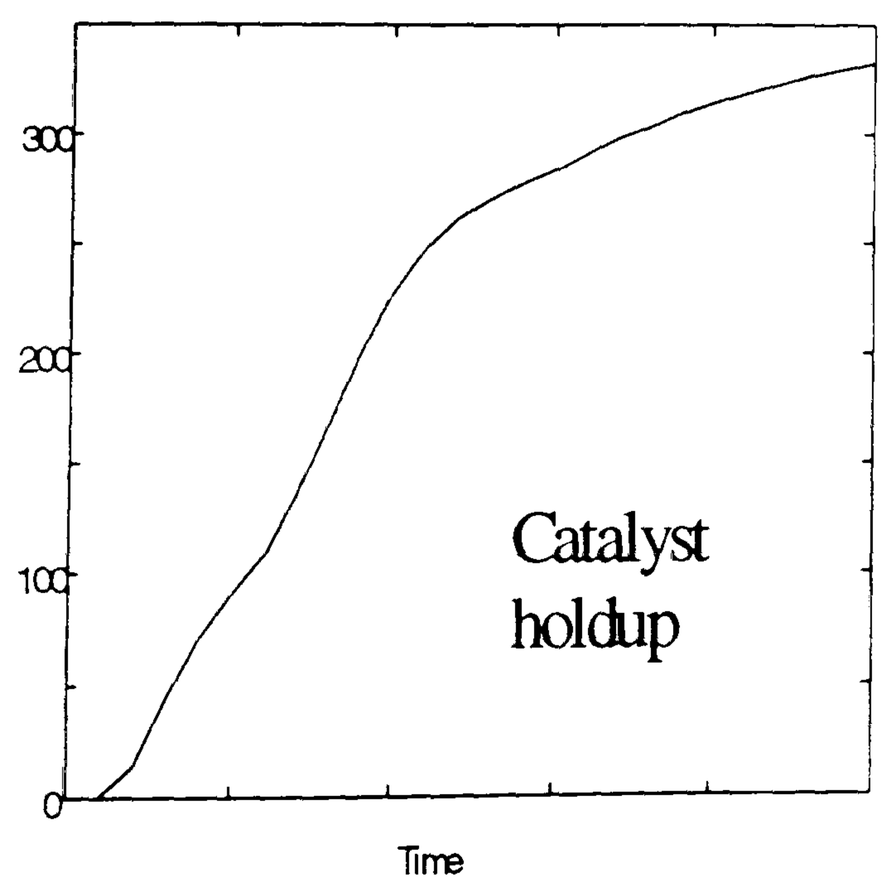


Figure 6 - 6 Dynamic trends of data set No. 5 shows an abnormal operation occurs when fresh feed is increased 50%

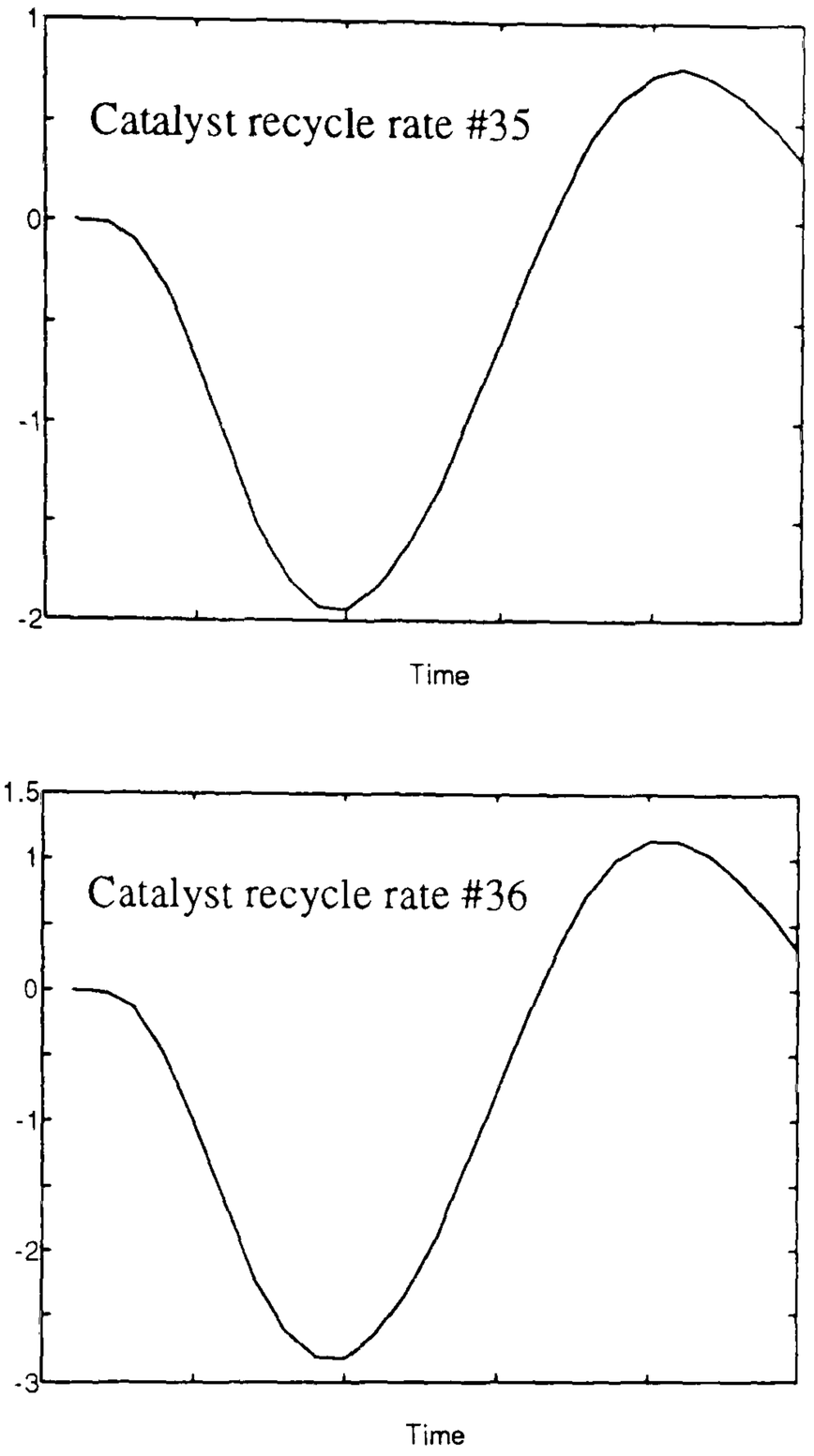


Figure 6 - 7 Dynamic trend of catalyst recycle rate for the opening ratio of hand valve V20 is decreased by 25% and 35%

When the threshold value is 4.5, the groups are [3,4,5,6,7,8,9], [19 20 21 22 23 24], [25 26], [27, 28], [35,36] and [56 57]. The pairing of identified classes and number of data sets for the threshold value are shown in Table 6-8. The clustering can be justified by inspecting the results in detail. Figure 6-6 shows the trends of three measurements taken from data set No 5. It is clear that regenerator temperature and concentration of oxygen in the regenerator flue gas fall sharply, while the catalyst hold-up in the reactor increases dramatically. This suggests an abnormal operation . A similar scenario is found for cases 3, 4, 6, 7, 8, and 9, so the result of regarding them as a single pattern is acceptable.

It is reasonable to conclude that data sets No 35 and 36 should be clustered as a single class. As can be seen in Figure 6-7, in both cases the dynamic responses of the regeneration temperature lead to a steady state and the process is still under control.

Any further increase in the threshold value is not appropriate because this would group significantly different types of behaviour into one class. For instance, data set No 29 (increasing opening of hand-valve V20 by 24%) would be grouped with the recycle slurry increase and decrease group when the threshold value is greater than 5, which is not what would be expected. Thus, a distance threshold of 4.5 is used to make the system evaluation in the next section.

6.4 System Evaluation

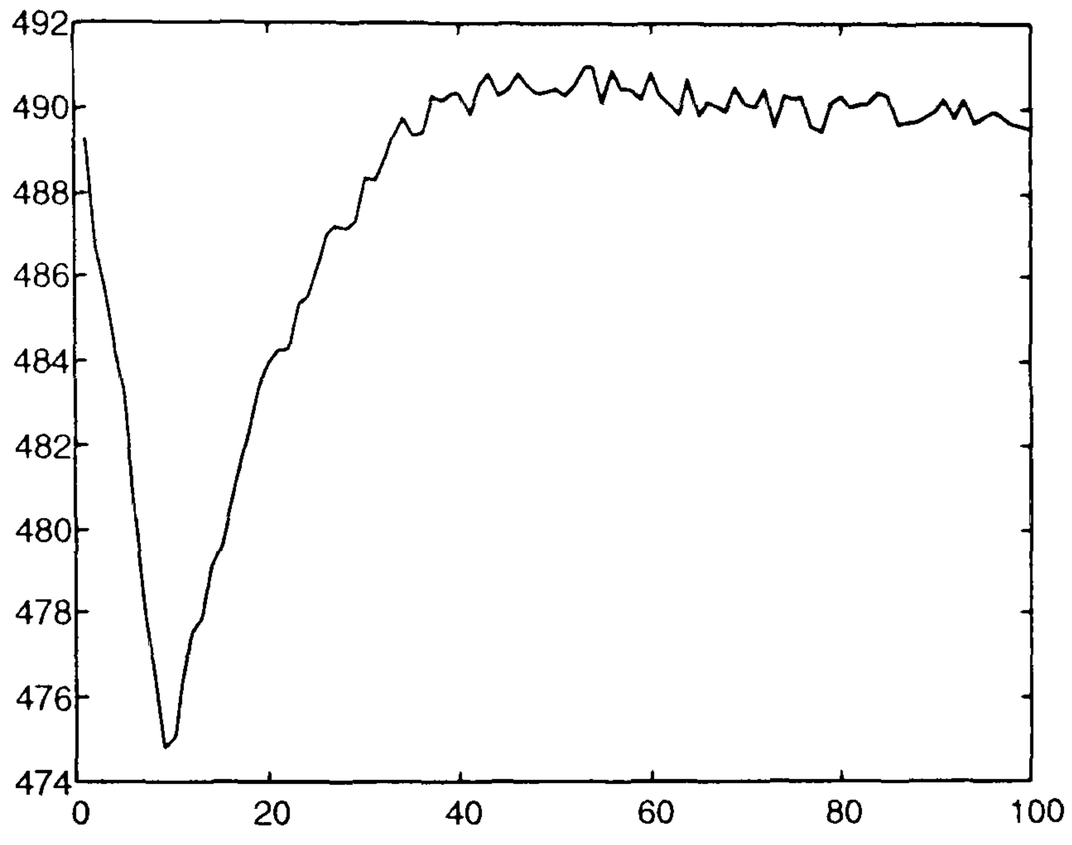
Three data sets represented by No 65 to 67 in Table 6-2, have been reserved for evaluation. Of these, the cases of No 66 and 67 are covered by the data sets used in system synthesis, i.e. within the boundary of the archived knowledge base, while No 65 is not. Data set No 66 refers to the fresh feed flow being increased by 65%, it results in a process fault. On the other hand, data set No 67 is a disturbance and relates to the preheat temperature of mixed feed being increased by 5°C. They are used to evaluate the behaviour of the process analysis system in different situations including interpreting signals and then report the location of the fault in terms of rules correctly, while avoiding extrapolation and consequently errors of interpretation.

6.4.1 Interpretation of dynamic responses

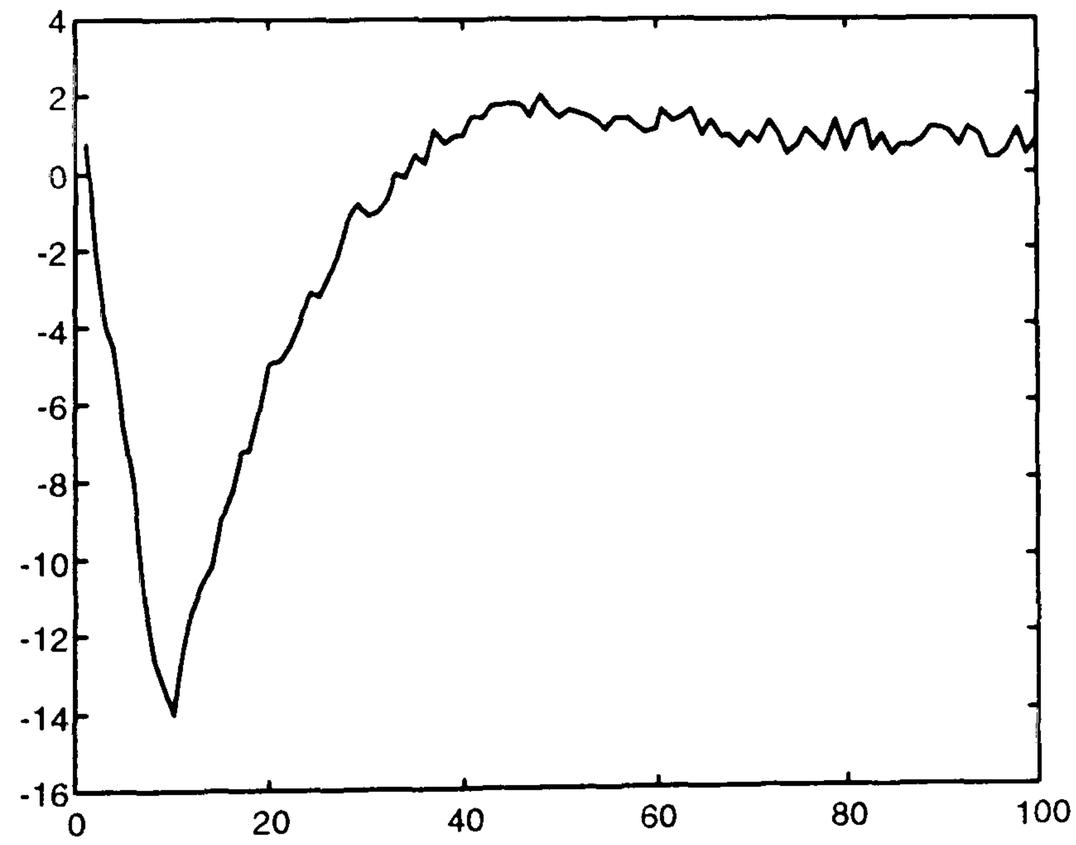
To interpret a transient input into a pattern correctly needs a well organised and constructed system which can distinguish process trends from noisy signals while capturing the most important features of the trend, recognising patterns in trends and reasoning based on a well founded knowledge base. It is the first of these points which is concentrated on here.

Figure 6-8 (a) is a reaction temperature transient based on 100 sample points of data set No 65. The signal to noise ration ($SNR = \frac{\text{signal energy}}{\text{noise variance}}$, Candy 1988) of the signal is

35.3. The residual generated by filtering out the steady process trajectory is shown in Figure 6-8 (b). This residual is then decomposed over four scales by the octave FIR non-subsampled filter bank shown in Figure 6-9.

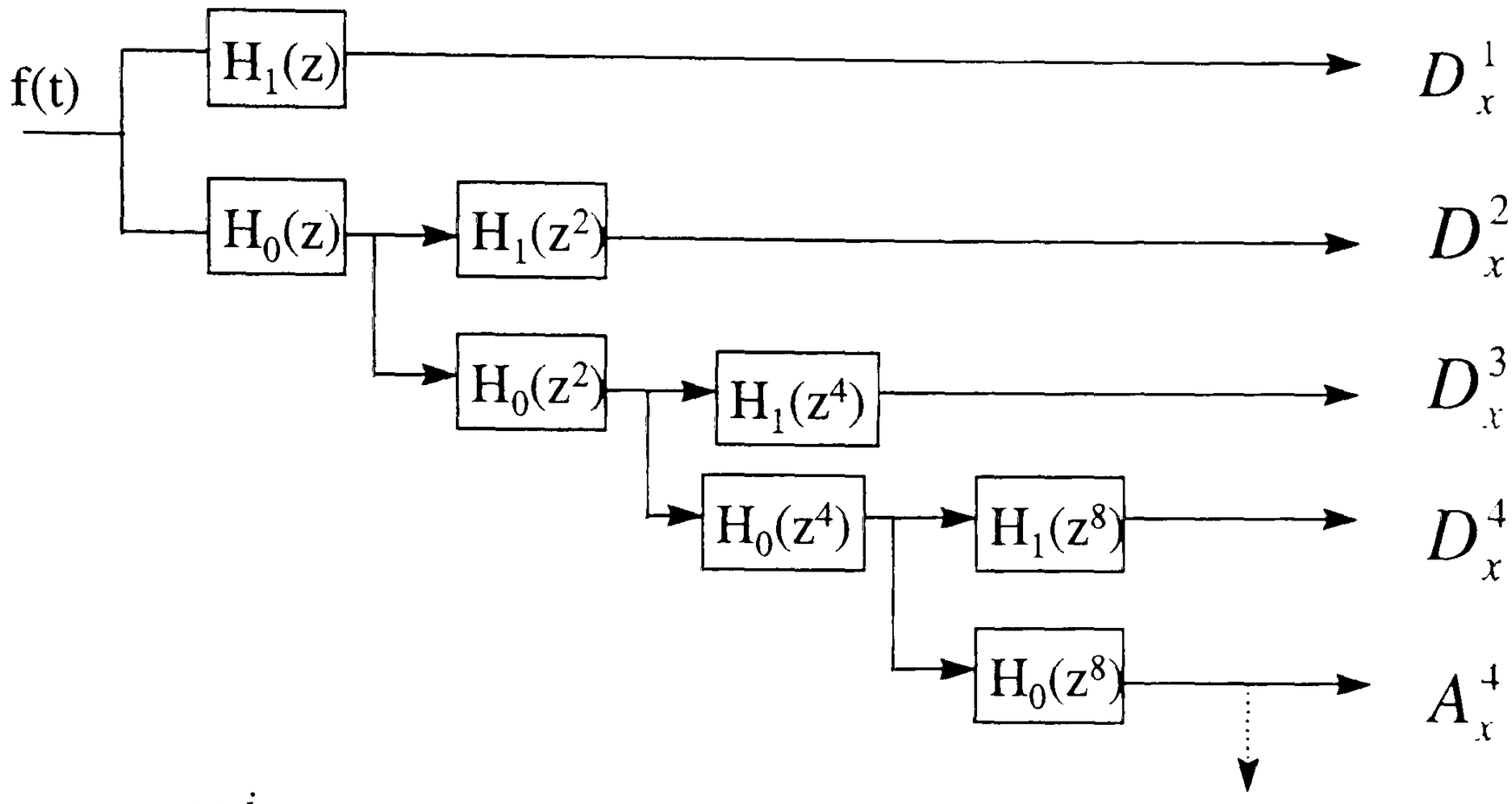


(a)



(b)

Figure 6 - 8 Reaction temperature transient (a) when feed flow rate is increased by 65%. The residual is shown in (b)



D_x^i -- detail signals of input f(t) on the ith scale

A_x^k -- approximation signal of input f(t) on the k scale

H_0, H_1 -- analysis of low-pass and high-pass filters

Figure 6 - 9 The octave band non-subsampled filter bank to decompose signal in four scales

In order to determine the noise components with respect to the scale, the wavelet function requires more moments to vanish. Unfortunately, using a wavelet function having more vanishing moment gives rise to more components in the feature, which in turn affects the compression ratio. Consequently, it is important to select a wavelet having an appropriate number of vanishing moments. In this case, the least asymmetric Daubechies's wavelets with eight coefficients has been selected as the basis functions for decomposing the residual into detailed and approximate parts.

The detail of a four scale multi-resolution analysis of the residual is shown in Figure 6-10(a) together with the corresponding extrema in Figure 6-10(b) denoted as extrema analysis. Following the derivation in section 3.4.3, the extrema influenced by the noise

fluctuations satisfy the following criteria (1) the amplitude of the extrema decrease on average as the scale of decomposition increases; (2) they do not propagate to the large scale. Extrema satisfying either of these criteria are regarded as noise components and so are filtered out.

When an infinite signal is truncated to a finite range in order to evaluate the discrete Fourier transform, spurious end effects are introduced. In practice, such a truncation is necessary, especially for on-line applications. The discrete wavelet transform of part of a signal also introduces the same end effect (Joshi et al 1995), as can be seen in Figure 6-10. The last extremum on scale 4 of Figure 6-10 is caused by an end effect rather than by any irregularity or singularity in the dynamic transient because there is no such irregularity or singularity in the process measurement signal or residual in Figure 6-8. The end effect extremum is automatically identified as a noisy component and removed from the representation because extrema end effects do not propagate along the scale and so satisfy the second criterion for noise removal. This demonstrates how feature extraction is achieved by generating the extrema of the trends and filtering out noise components, while also removing end effects. This can be seen by comparing Figures 6-10 and 6-11

Most of the elements in the detail are caused by noise and are removed after filtering the components. This is especially true for measurements in large continuous processes where the process trends often change infrequently (Dong and McAvoy 1996). This means that the extrema representation is a very sparse vector after removing the noisy components, so compression is easily achieved using piece-wise methods.

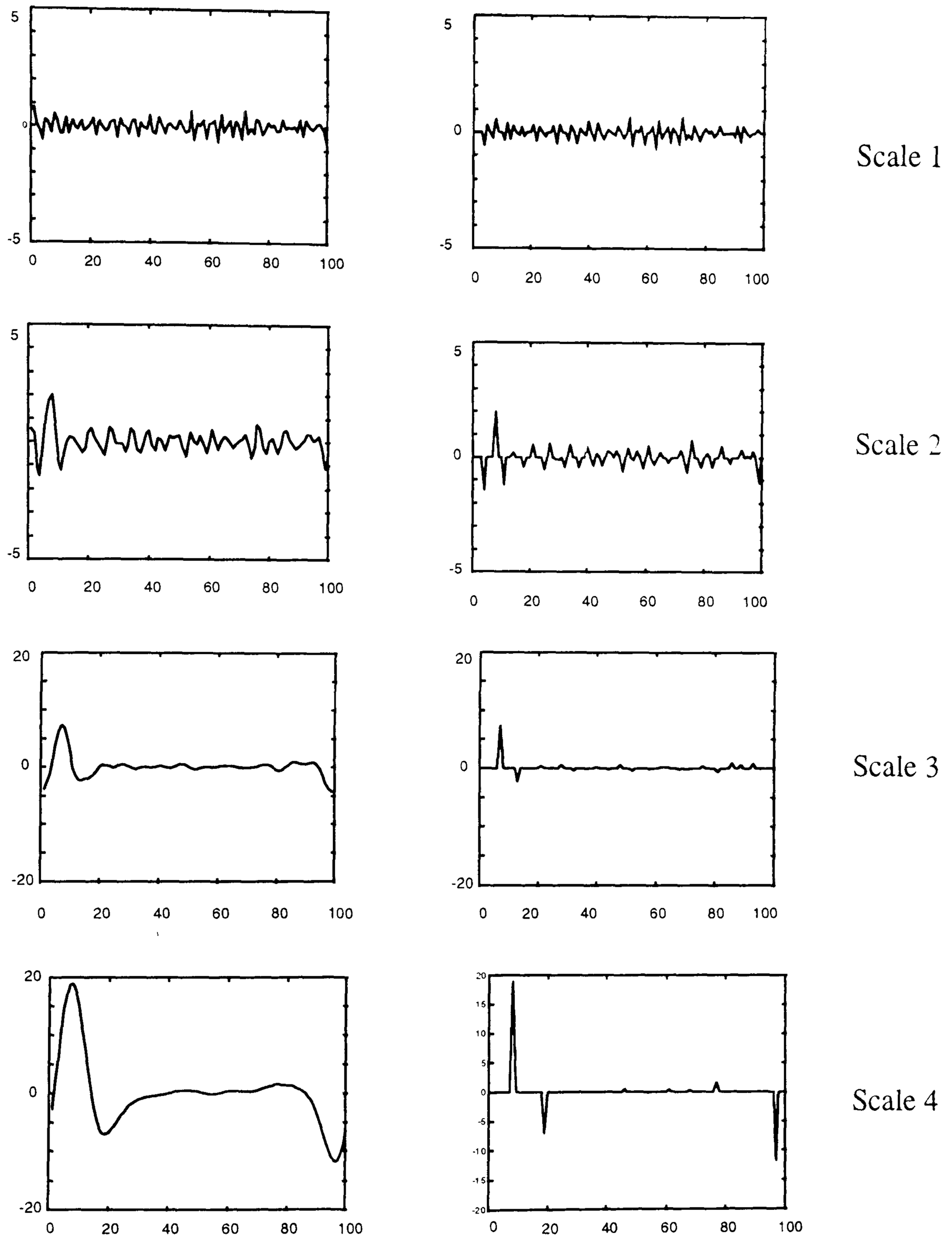
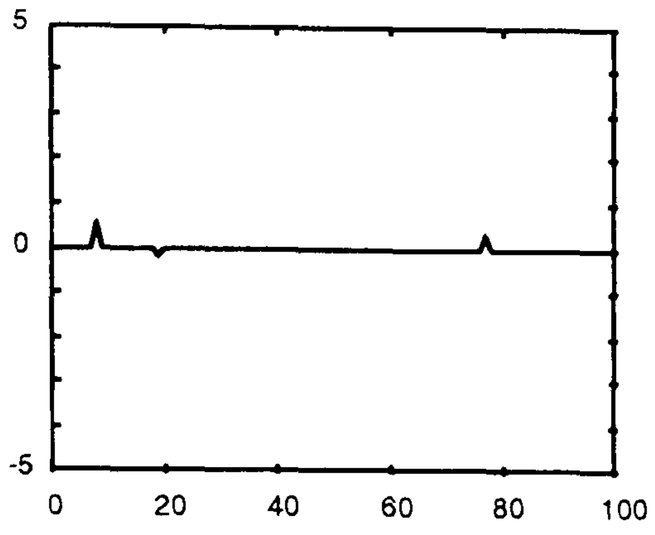
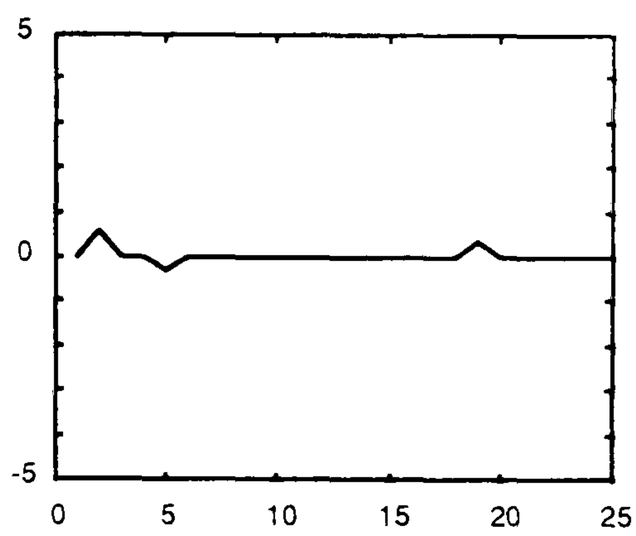
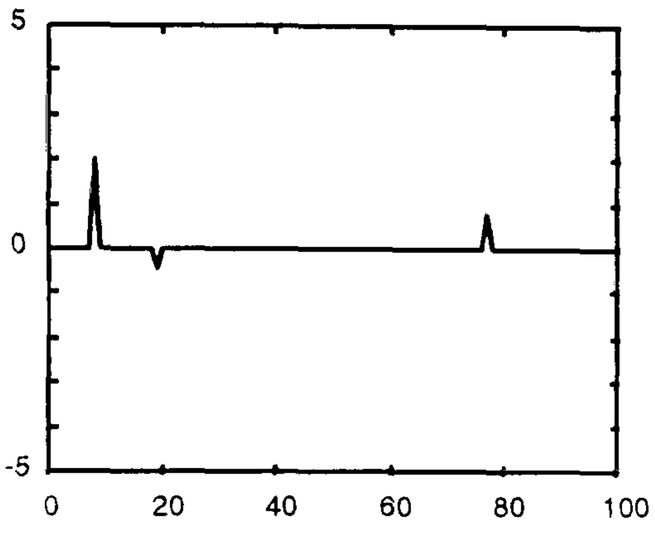


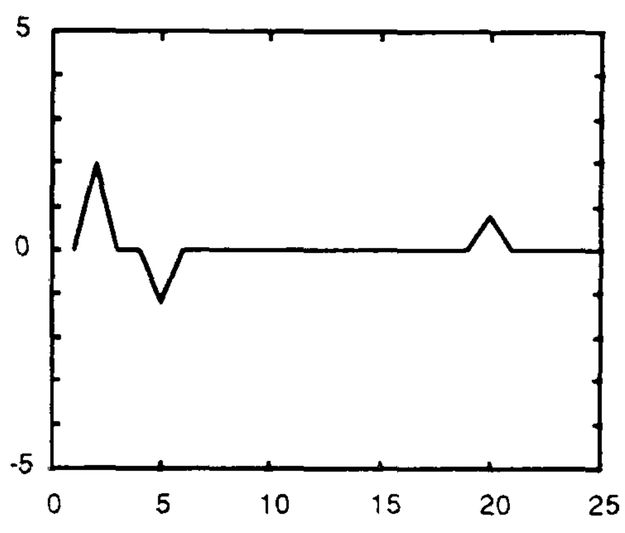
Figure 6 - 10 Multi-resolution analysis of the residual of reaction temperature (a) and its extrema representation (b)



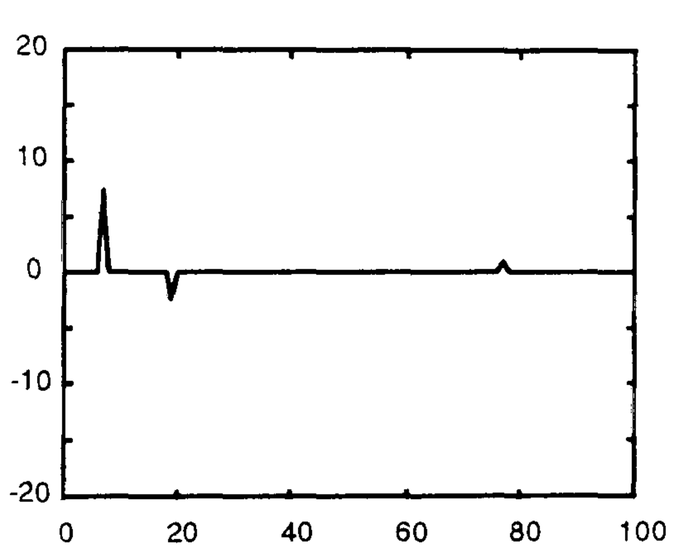


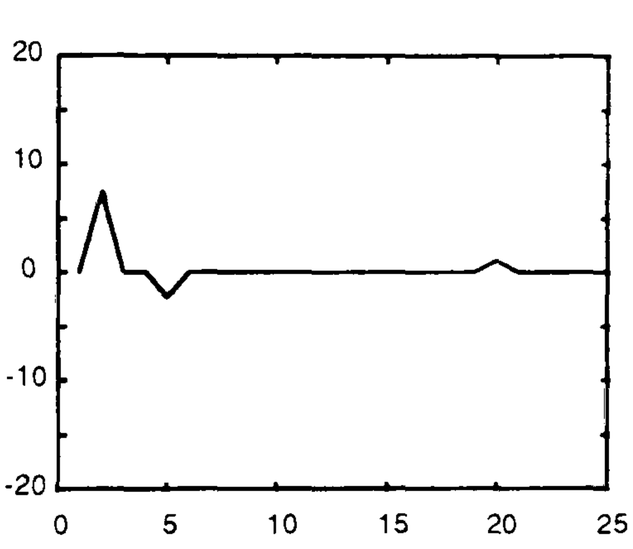
Scale 1



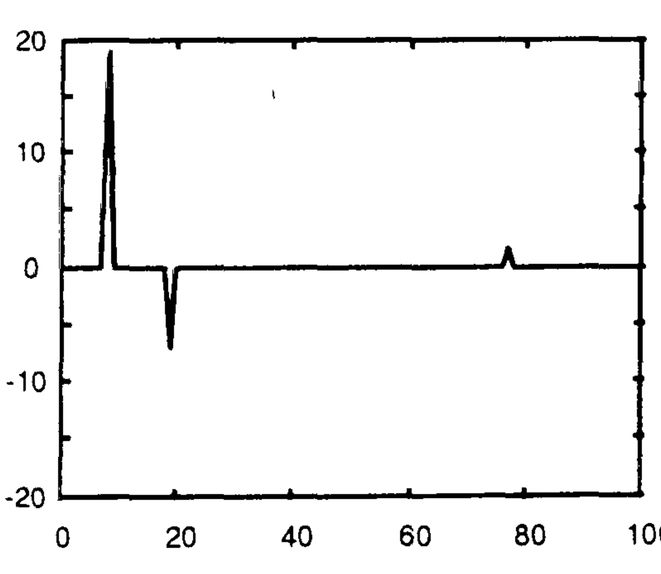


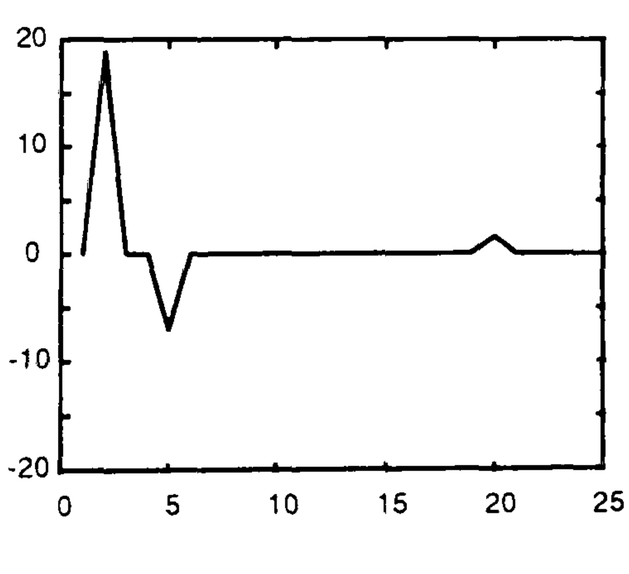
Scale 2





Scale 3



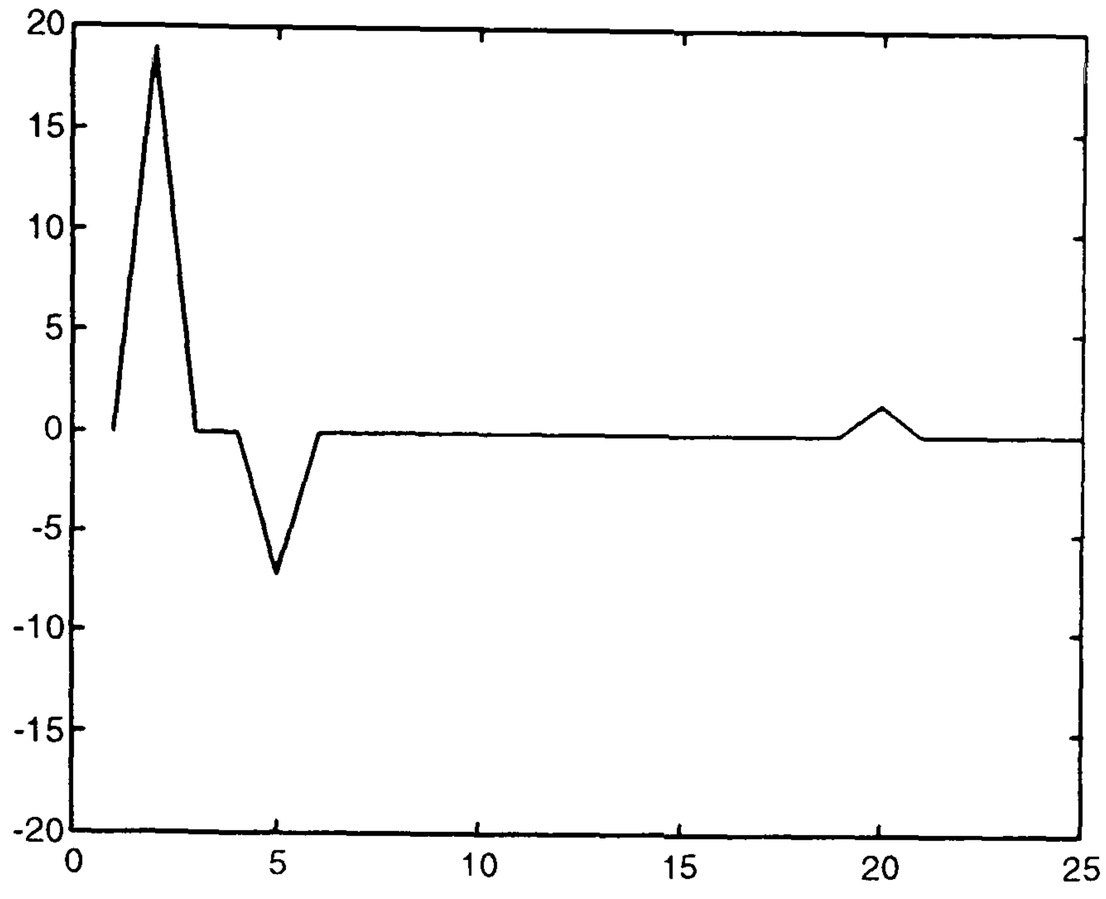


Scale 4

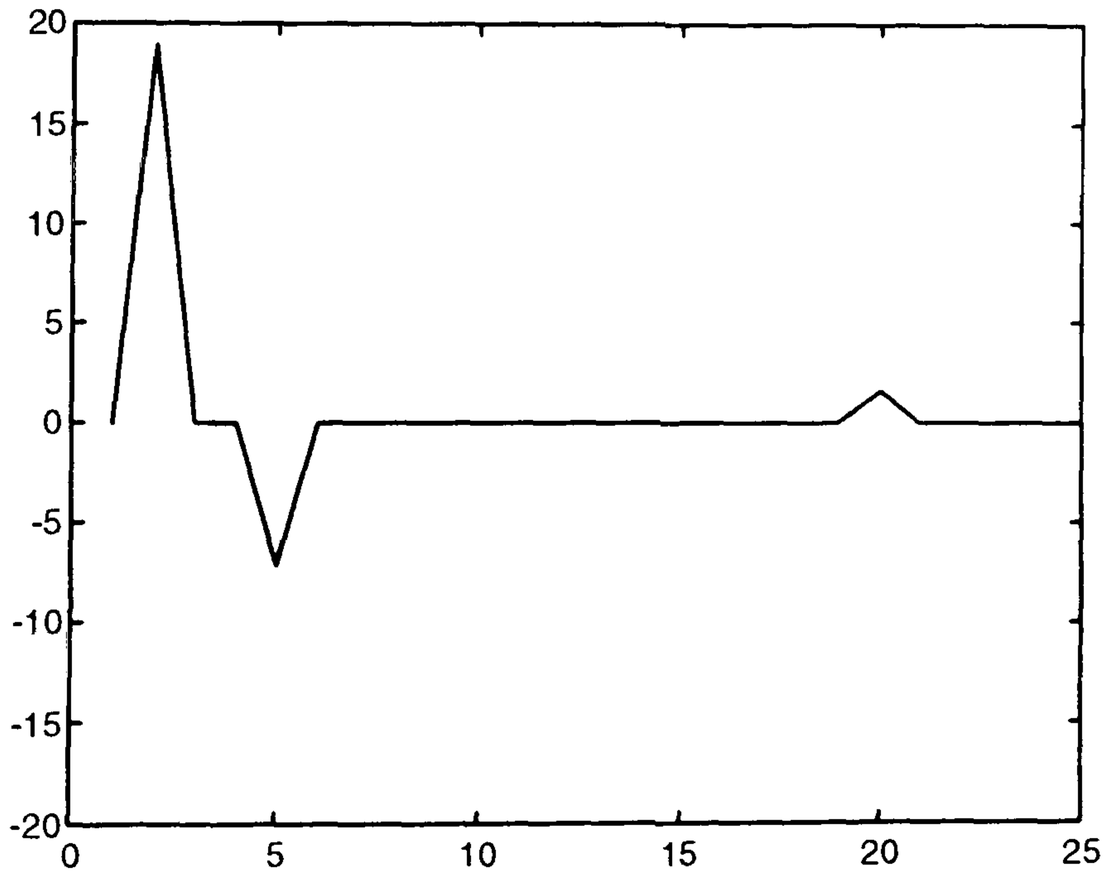
Extrema representation after removing noisy components (a)

Extrema representation after compressing (b)

Figure 6 - 11 Extrema after removing noisy components (a) and extrema representation after compressing (b) based on the results of Figure 6-10

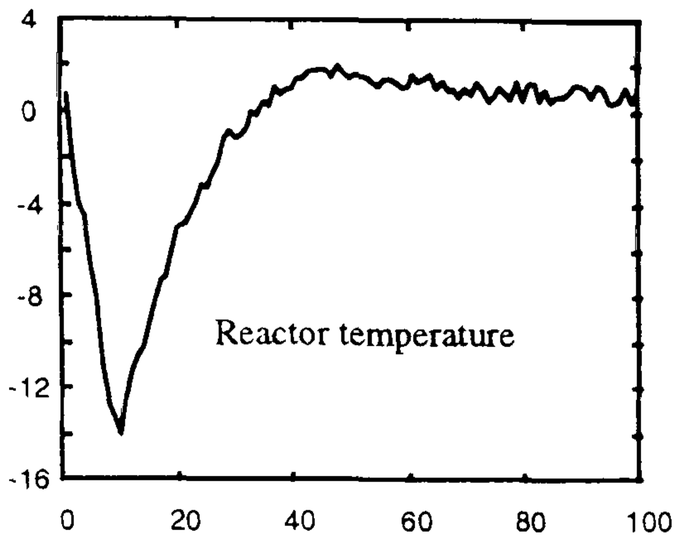


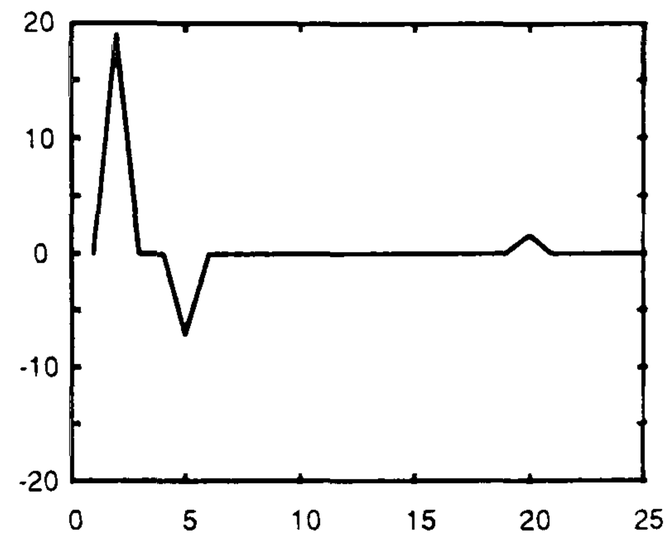
(a)

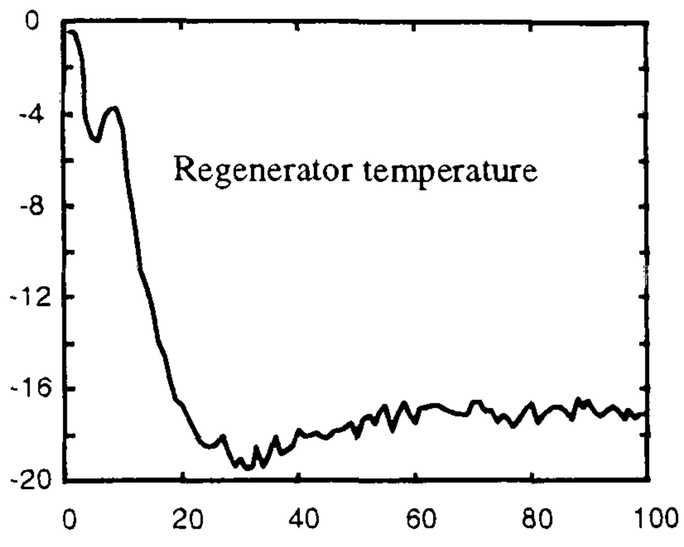


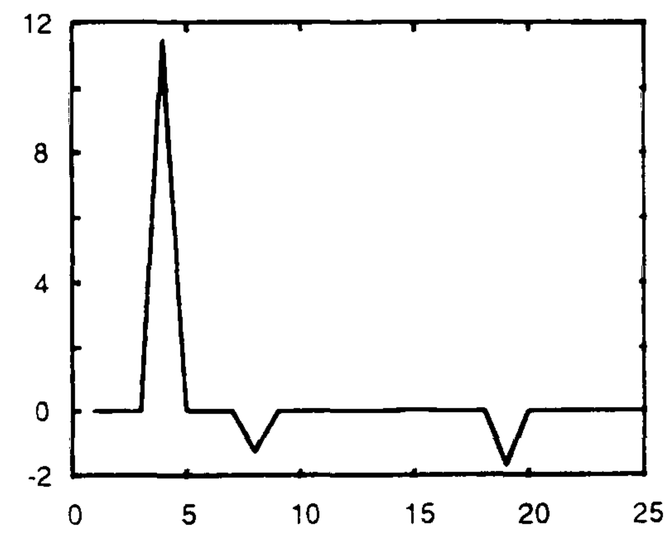
(b)

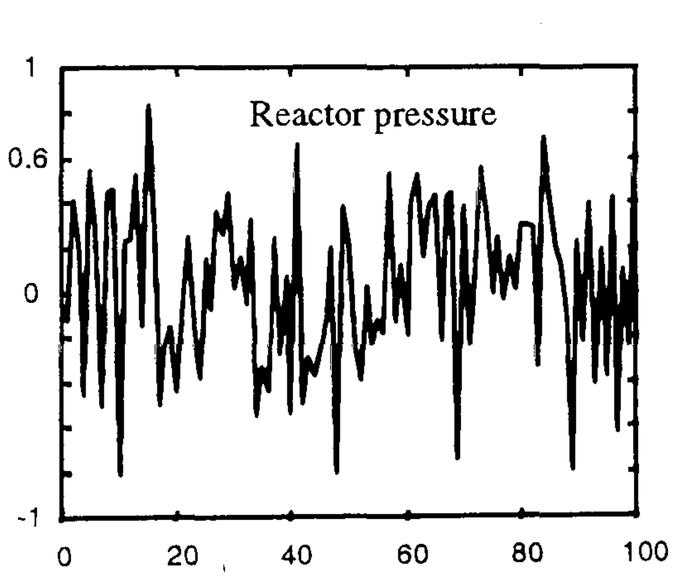
Figure 6 - 12 Comparison of extrema of noise free reaction temperature signal (a) and extrema of that based on noise filtering (b)

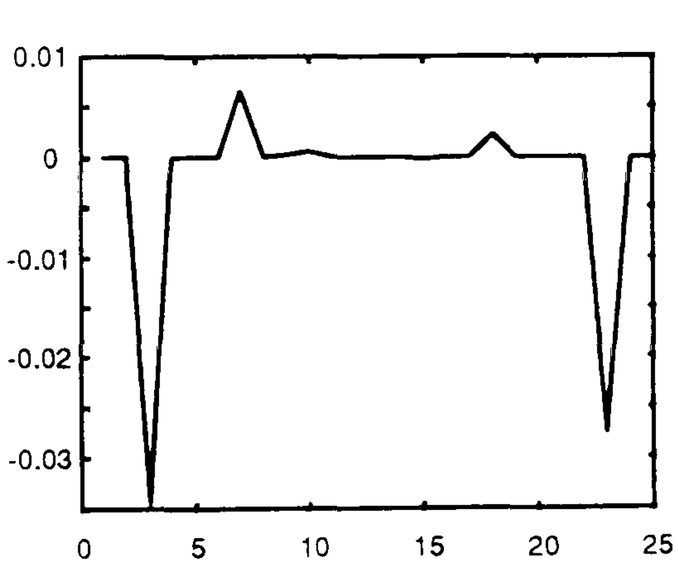


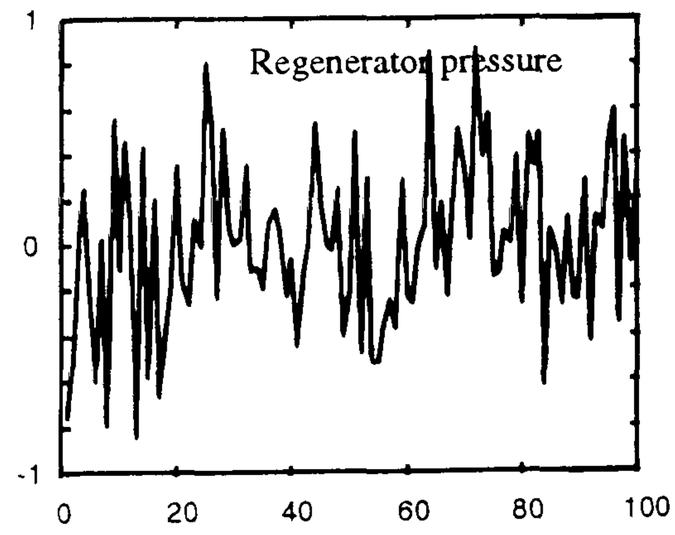












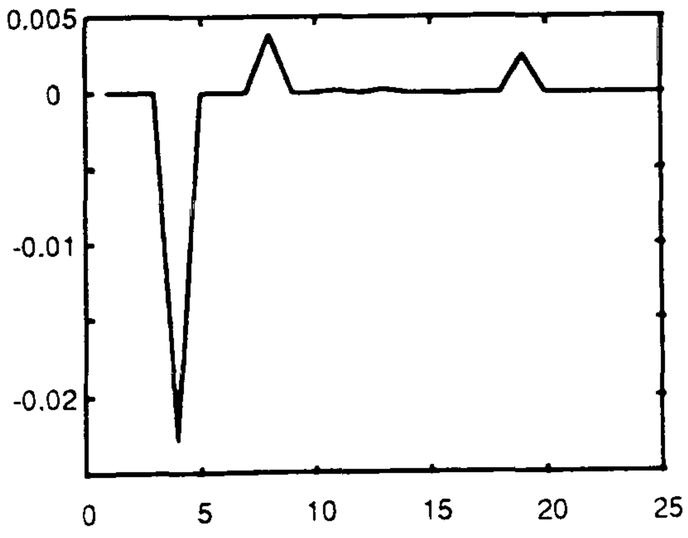
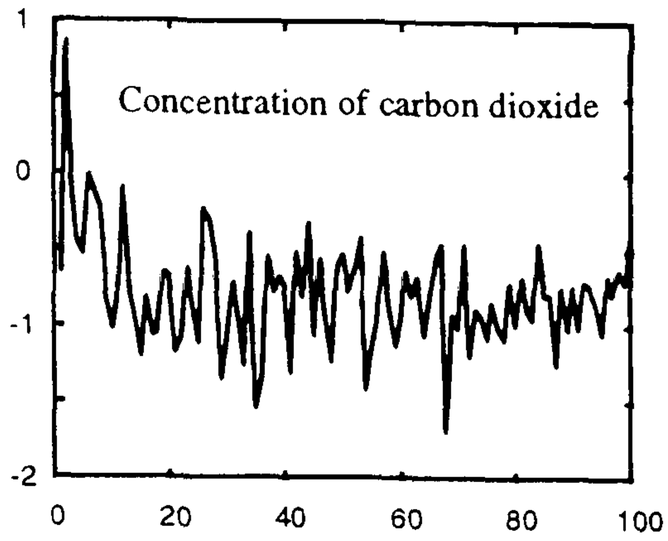
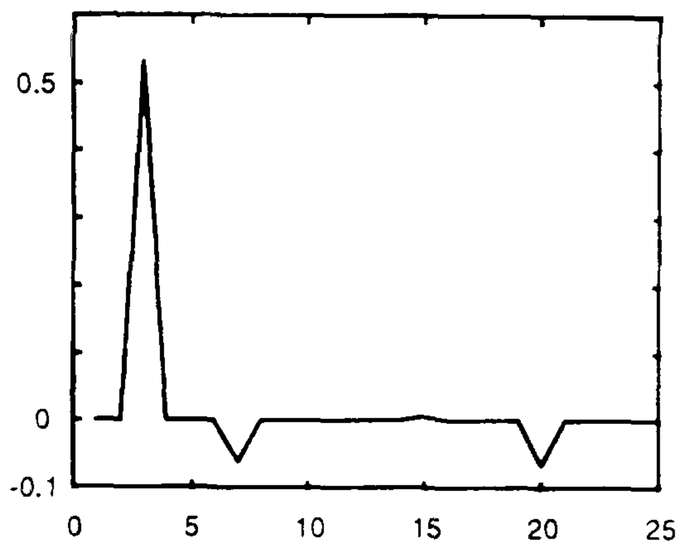
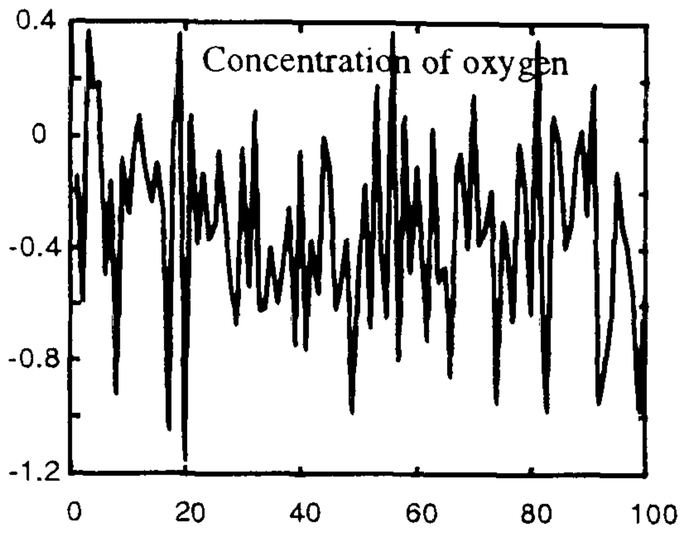
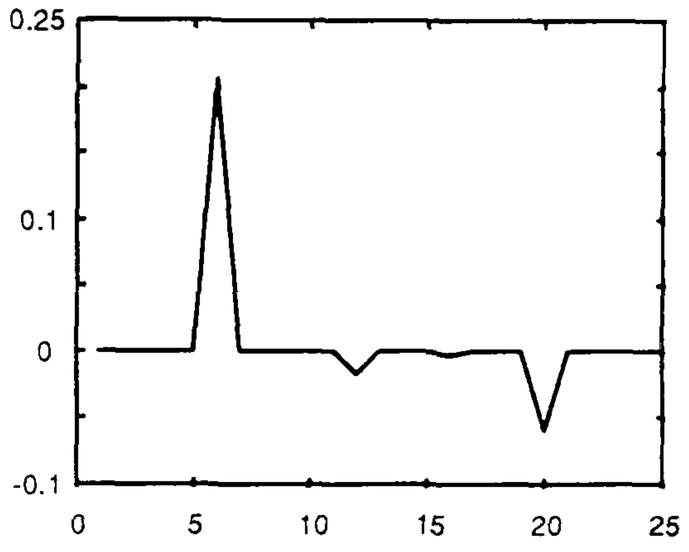


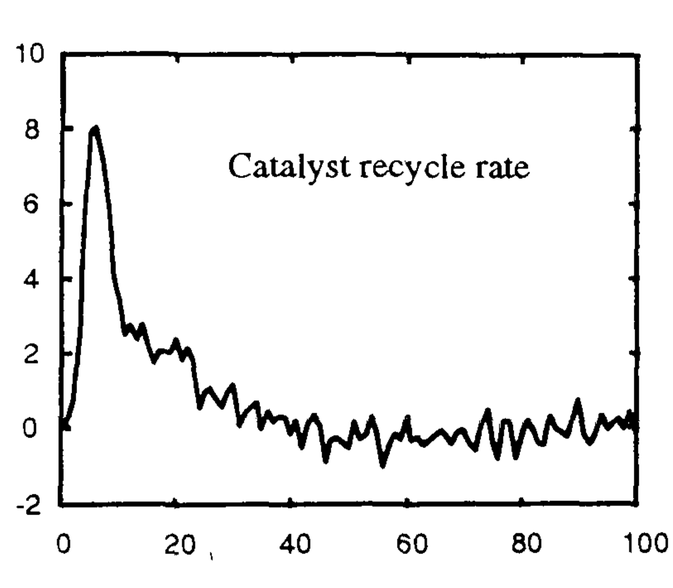
Figure 6 - 13 (1) Trends in operational variables when feed flow rate is increased by 65% and the features extracted by wavelet approach

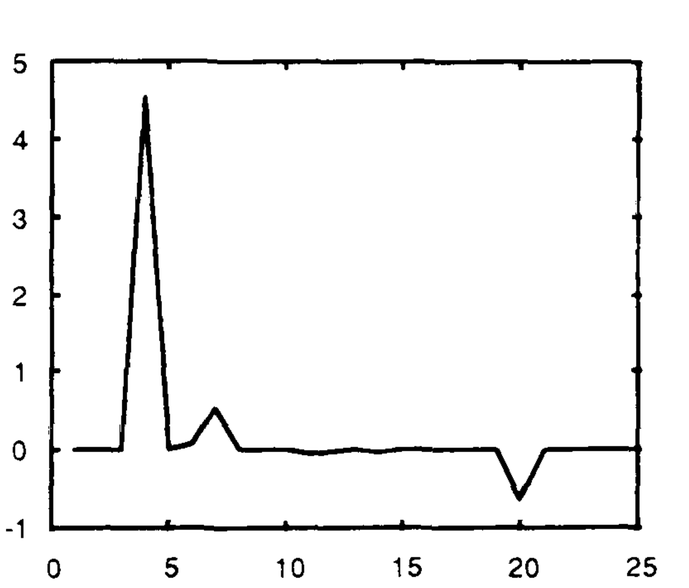


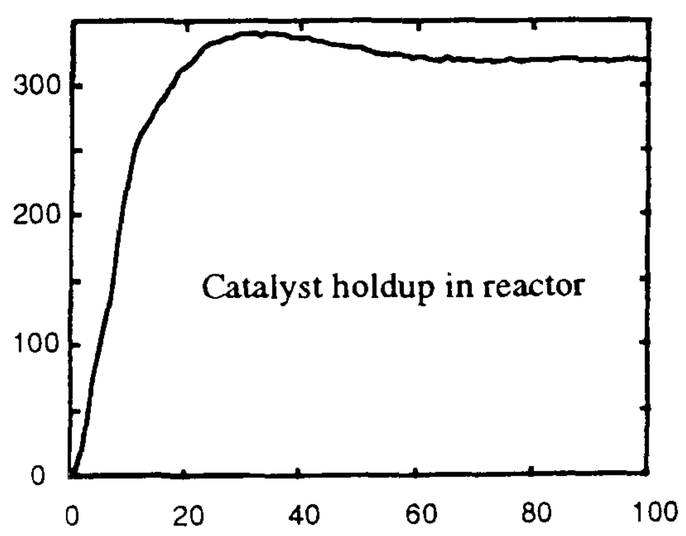












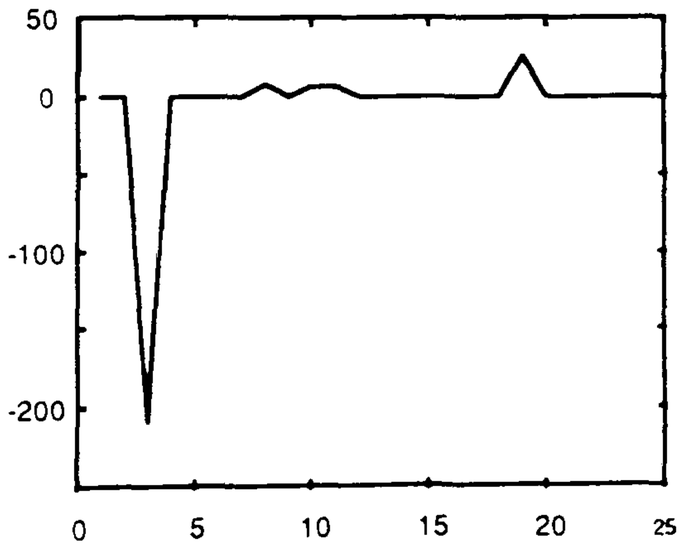


Figure 6 - 13 (2) Trends of operational variables when feed flow rate is increased by 65% and the features extracted by wavelet approach

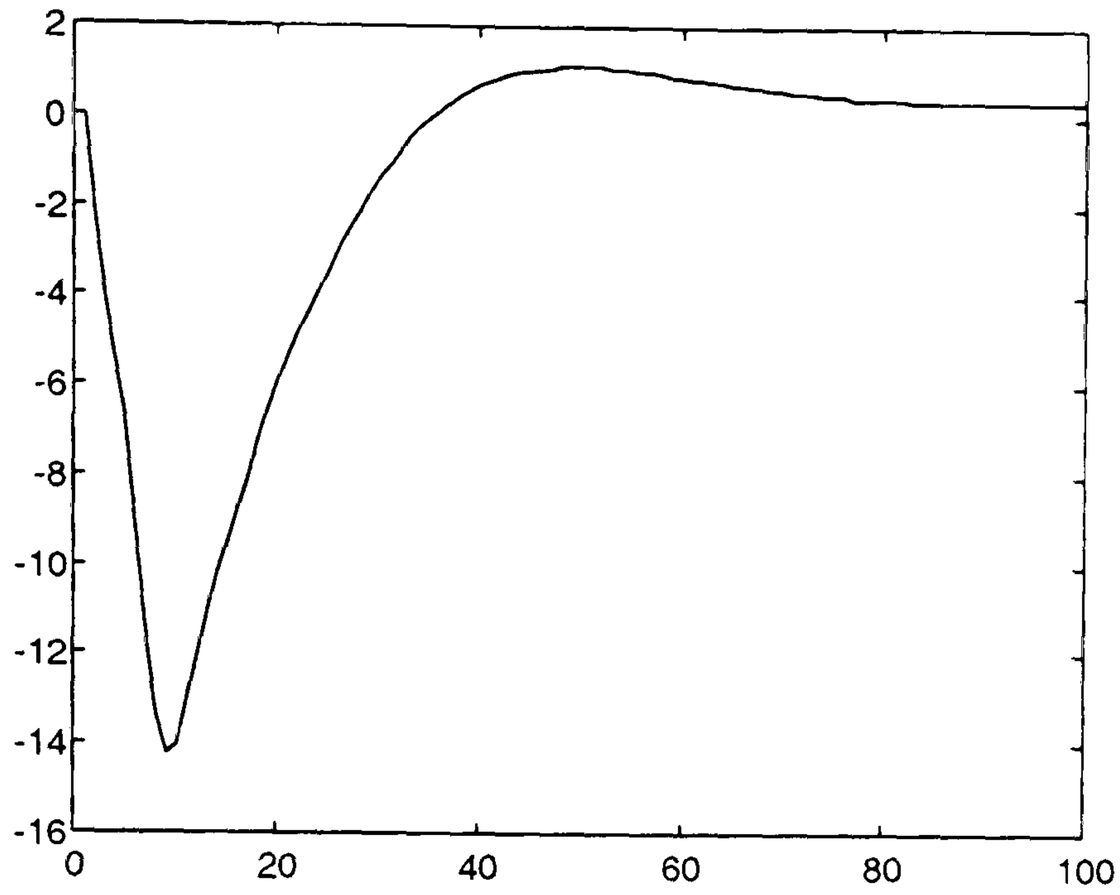


Figure 6 - 14 Approximation of Figure 6-3 generated by wavelet decomposition

In order to retain the time component, compression is based on a fixed length of the piece-wise sub-region. Here, the length of the sub-region is 4 i.e. the element with the biggest absolute value is picked to define the region and the others are discarded. If there is no non-zero item in a region, the value is zero. For example, a vector (0 0 0 3 0 2 0 0 0 0 0 0) is reduced to (3 2 0) using the above compression technique. So the data compression ratio is 4:1. Since there are only three non-zero values in the extrema representation, a higher compression ratio is possible by extending the length of the piece-wise region. However, non-zero components may be overlapped, i.e. non-zero components are in the same sub-region and only one is used in the representation. This results in loss of information.

The advantage of using the simulator is that it is possible to know whether noise components are filtered and the important features of a transient are captured using the proposed approach. Figure 6-12 compares the result of using the noise removal technique with that of the basic signal. The extrema representations for these are the same. The extrema in Figure 6-12(b) correspond to the irregularities of the trend in Figure 6-8. So the wavelet-based approach has proved to be an effective technique for

extracting features and carrying out filtering of noise while achieving data compression of the process signals.

The scale 4 extrema are used as input to ARTNET for pattern clustering because it has the largest values compared with other scales. There are eight variables which can be related to the 65% increase in feed flow rate, and Figure 6-13 (1) and (2) shows these measurements and features. The approach also generates an approximation of the measurements, as shown in Figure 6-14 for the same reaction temperature.

These result demonstrate that wavelet based pre-processing provides a powerful tool for feature extraction, noise filtering and data compression for use in pattern clustering. This is essential in fault diagnosis because it ensures that the system is not affected by signal noise. In a single step, it generates features which can be used in ARTNET pattern classification and approximation and so is fundamental in providing information for knowledge representation.

Based on the features extracted by wavelet based signal pre-processing, the system identifies data set No. 66 as pattern No 3. Using the approximations of the pre-processing approach, it identifies the fault as "Fresh Flow rate very High with degree 1.0." based on the effects of

Reaction temperature is low with degree 0.83

Regenerator temperature is medium low with degree 0.53

Reactor pressure is normal with degree 0.98

Regenerator pressure is normal with degree 0.88

Carbon dioxide concentration is low with degree 1.0

Carbon monoxide is normal with degree 1.0

Catalyst recycle is medium high with degree 0.94

Catalyst hold-up is high with degree 0.55

The reported pattern number is the clustering result using ARTNET. Since the data set No. 66 is for the fresh feed flow increasing by 65%, it is reasonable that it is clustered

into class 3 which consists of data sets No. 3 to 9 corresponding to fresh feed increased by 30% to 90%, as listed in Table 6-2 and 6-8.

When data set No 67 is used for testing, the system identifies it as pattern No. 13 and reports the fault as "Preheat Temperature of Mix feed medium High with degree 0.51" and the effects

Reaction temperature is normal with degree 0.74

Regenerator temperature is medium low with degree 0.55

Reactor pressure is normal with degree 0.51

Regenerator pressure is normal with degree 0.69

Carbon dioxide concentration is medium high with degree 0.77

Carbon monoxide is medium low with degree 0.74

Catalyst recycle is normal with degree 0.54

Catalyst hold-up is normal degree 0.51

This is also correct because data set No. 67 is a disturbance in the preheat temperature of the mixed feed and class No. 13 groups increase in the preheat temperature of the mixed feed.

Clearly, when the current fault or disturbance is within the boundary of the archived knowledge base, the system interprets the pattern correctly based on the dynamic transients of operational variables. The system also provides explicit and transparent cause-effect results in rule form to the process operator, as well as reporting the fault location.

6.4.2 Avoiding extrapolation of reasoning

The results identified by the operational support system reported here may also be obtained using other techniques, for example, a back-propagation feedforward neural network (FFNN). The critical issue is the behaviour of such a system when it is used to identify an operational state which is different from the archived knowledge base. As discussed in section 2.3, an operational support system in a rule-based expert system (RBES) or FFNN shell cannot work properly and report poor results which may lead to

incorrect diagnosis because it requires reasoning outside the knowledge base, also known as extrapolation. Even worse, neither the RBES nor FFNN can flag difficulties automatically when extrapolation is used. This issue is difficult to discuss in the case of RBES because the knowledge base is difficult to define. The behaviour of a FFNN is illustrated below.

As listed in Table 6-2, the data sets used here are summarised as 13 types of faults or disturbances, which are described by eight operational variables. To construct an FFNN to map the problem, it requires eight input nodes and 13 output nodes for the FFNN. Using the same 64 data set for system synthesis as for training the FFNN, it identifies data set No 65 as normal operation, as be seen in Table 6-9. The actual pattern is derived by fault types 2 and 9 i.e. 65% decrease in fresh feed and a failure in cooling water pump p-02. The diagnostic result from the FFNN is dangerous to use in process diagnosis, since an abnormal is indicated as normal. Because of this, and since the extrapolation cannot be flagged, the reliability of the reasoning process is not guaranteed for operational support systems in either RBES or FFNN which clearly undermines confidence in these methods.

Table 6 - 9 The prediction by FFNN and the actual target for data set No 65 refers to 65% decrease fresh feed and a failure in cooling water pump p-02.

Target	0	0.65	0	0	0	0	0	0	0.99	0	0	0	0
Predict	0	0	0	0	0	0	0	0	0	0	0	0	0

Extrapolation can be avoided in the present case. Based on transients in the operational variables, data set No 65 is detected as a new pattern by the system when it is used as the system input. It is, in fact, a combination of two faults, decrease in fresh feed and increase in the preheat temperature of mix feed. The responses of a combination of these two faults are different from either faults, thus it can be regarded as a new operational state. As discussed in section 4.2, being able to report a non archived operational state as

a new or unknown is important. It alerts process operators that conditions may give rise to a new state and they should pay attention.

Clearly, extrapolation is a critical issue in operational support system based on RBES and FFNN, but such issues may be successfully avoided by the operational support system developed here. Thus, the support system proposed here can provide reliable outcomes to reasoning which are able to assist process operators in process diagnosis.

6.5 Concluding Remarks

Based on this case study, it can be concluded that the system developed here has the following:

1. The system can capture the most important information about dynamic systems and remove the effects due to noise. This ensures that the system is able to interpret transients patterns correctly.

2. The system provides explicit, transparent, and no-conflict cause-effect rules to operators. It also reports fault locations and related process trends which assist operators in fault diagnosis.

3. The system avoids extrapolation of reasoning and so improves the system reliability.

4. The system is easy to maintain and update based on the new technique of knowledge base organisation.

Chapter 7

CONCLUSIONS AND SUGGESTIONS FOR FUTURE WORK

7.1 Conclusions

The work reported in this thesis is concerned with developing effective methodologies and systems for process monitoring and diagnosis. An integrated framework is developed which consists of several components including feature extraction from on-line signals for dimension reduction and noise removal, unsupervised and recursive machine learning for operational state identification as well as automatic generation of knowledge rules from process operation data. The methods and a prototype system have been evaluated using data collected from a dynamic simulator of an industrial refinery fluid catalytic cracking process.

For monitoring and diagnosis of process operational data, dynamic transient signals are probably more important than instant values of measurements for the purpose of identifying operational states and to anticipate evolving behaviour patterns. The transient signals are characterised by the following features:

1. large size of data - a meaningful segment of a transient may consists of several tens of sampling points and a process may have several hundred dynamic trends to be monitored simultaneously;
2. noise and uncertainty – at large noise to signal ratio, the real trend of the signal may not be identified;
3. dynamics such as those due to time delays;

4. correlation between variables.

Wavelet multi-resolution analysis has been investigated for feature extraction from dynamic transients. Features are defined as irregularities and singularities of a signal which are identified as the extrema of wavelet multiresolution analysis. The method developed by Cvetkovic and Vetterli (1995) and Mallat and Huang (1992) have been adapted for this purpose. A method is developed which is able to achieve identification of extrema of a transient signal, noise component removal as well as dimension reduction in a single step. Comparison of the approach with others has also been made, including episode-based qualitative interpretation as well as using window Fourier transforms.

It is important for any process operational state identification method to be based on an unsupervised clustering mechanism because supervised methods require training data and this is difficult to get. It is also important for a method to be recursive so that its performance can be continuously improved through on-line learning. Adaptive resonance theory (ART2) provides the basis for such an approach. While the method has generally been regarded as having a sophisticated component for effectively dealing with noise in data, this study shows that it is still too sensitive to noise contained in process transient signals. A new framework, ARTNET, has been developed which is essentially a modified version of ART2 for process operational identification and uses wavelet feature extraction as a substitute for the data pre-processing component of ART2. Apart from being able to extract features for pattern identification purpose, ARTNET is superior to ART2 in avoiding the adverse influence of noise, in the selection of threshold values and in computational speed, while retaining unsupervised and recursive learning capabilities.

Most methods studied in the literature, both supervised and unsupervised, use similarity or distance measures to discriminate and cluster operational states. As in the case of computer simulation, this results in a specific numerical solution but there is no qualitative explanation. Obviously, it is desirable that operators should be provided with knowledge rules. For this purpose, automatic knowledge generation methods have been examined and the fuzzy neural network approach proposed for automatically generating fuzzy production rules. The approach has a number of important features when assessed

against existing methods. First of all, it uses fuzzy concepts to bridge the gap between quantitative data and qualitative knowledge. Secondly, it uses the advantage of back-propagation neural networks for being able to use a large amount of data effectively and robustly. Thirdly, the feature extraction approach can be incorporated into this method. Methodologies for integrating the several components and making use of them concurrently to compensate for the inadequacies of each are also exploited. The knowledge hierarchy constructed by ARTNET and fuzzy-FFNNs to represent and organise the knowledge base makes the method able to extrapolate beyond the rule base and solve conflicts effectively.

The framework developed here has been tested using data collected from a dynamic simulator of a refinery fluid catalytic cracking process (FCC). This is a complicated process featuring highly non-linear dynamics arising from the strong interactions between the riser tube reactor and fluidised bed regenerator through coupling of the heat, mass and momentum balances. The simulator has been used in industry for several years and shown to have high fidelity and allows the introduction of small and large disturbances and faults. From the prior knowledge about the data generated, methods developed in this study can be tested. Encouraging results have been obtained showing the great potential of the methods developed in this work.

7.2 Suggestions For Future Work

Many more challenges than addressed in this study need to be considered in order to develop an effective and practical on-line decision support system. For example, the validation and confidence bound of backpropagation neural networks (Shao, et al., 1997, Zhang et al, 1997, Chen and Wang, 1998, 1999). Below is a summary of additional work which needs to be carried out:

1. On-line data collection techniques. In this work, on-line signals are assumed to be windowed and available. In practical use, a proper on-line data windowing technique is needed to extend the system to on-line applications.

2. An approach for generating proper residuals of dynamic transients. Although there is a residual generation procedure in this work, a more effective and precise approach is required for residual generation to solve problems such as whether the residual is a true process transient or model error. Otherwise, the result of reasoning provided by the operational decision support system is wrong because of the incorrect residual.

3. Wavelet transformation is an effective tool for feature extraction. However, the algorithm is afflicted by the so-called 'end effect' as discussed in Chapter 3. Further development of the transform is necessary for dealing with process signals that are not zero-mean based.

4. The distance threshold in the ARTNET algorithm is still a tuning parameter. Finding an algorithm to determine the parameter adaptively so it can be used on-line is important: note, however, there is a distance threshold search algorithm in this work.

5. This study has only been tested on continuous processes, though the approach is not necessarily restricted to these. It is expected that batch processes will provide more challenges because of the distinctive nature of batch operation. Batch operations do not have steady state operations. So it is expected that the approaches used in this study, especially feature extraction using wavelets and knowledge discovery techniques, will have a role to play in processing data. With the shift of focus from large-scale commodity production to smaller scale fine chemical, pharmaceutical and food production, there is a clear need for research on this topic.

6. This study has focused on on-line decision support for operators. In process plant operation, there is a higher level of control: namely supervision and management of long term performance. For this purpose the data collected and averaged over days and weeks are useful. This area has not been widely studied, though there have been a number of publications using multivariate data analysis, inductive learning as well

as data mining and knowledge discovery technologies (e.g., Zhang et al., 1997. Clarke-Pringle and MacGregor, 1998, Saraiva and Stephanopolous, 1996, Wang and McGreavy, 1998, Chen and Wang, 1999).

Appendix A

Extrema Of Wavelet Transform And Singularities Of Signal

In mathematics, the local singularity of a function is often measured with Lipschitz exponents. In practice, the relationship between the Lipschitz exponent and a singularity does not provide simple and direct strategies for detecting and characterising the singularities of a signal. The wavelet transform extrema representation is an efficient approach for studying singular structures in signals. In the following, the relationship between wavelet transform extrema and singularities of a signal are described based on the work of Mallat and Hwang (1992) and Mallat and Zhang (1992).

Notation:

1. Compact support: a function *f(t)* is compact supported, such that $f(t) = 0$ if $t < t_1$ or $t < t_2$, where $-\infty < t_1 < t_2 < \infty$

2. $L^p(R)$ denote the Hilbert space of measurable, functions such that

$$\int_{-\infty}^{+\infty} |f(t)|^p \, dt < +\infty$$

the norm of $f \in L^2(R)$ is given by

$$\|f(t)\|^2 = \int_{-\infty}^{+\infty} |f(t)|^2 \, dt$$

3. Lipschitz space: for $0 < \alpha < 1$, Lipschitz space is the set of all $f \in L^\infty(R)$ such that

$$\sup_{x \in R} |f(t+h) - f(t)| \le C|h|^\alpha$$

where α is called the Lipschitz exponent.

4. The l^{th} moment of a wavelet is defined as:

$$\int_{-\infty}^{\infty} t^l \Psi(t)dt$$

And, a wavelet is said to have M vanishing moments if

$$\int_{-\infty}^{\infty} t^l \Psi(t)dt = 0 \qquad \textbf{(A - 1)}$$

For l =0,1,...,M. Equation (4-21) is equivalent to

$$\left.\frac{d^l \hat{\Psi}(\omega)}{d\omega^l}\right|_{\omega=0} = 0 \qquad l = 0, 1, \ldots..M.$$

Definitions:

1. A function f (t) is said to be Lipschitz α, for $0 \le \alpha \le 1$, at a point t_0 , if and only if there exists a constant A such that for all points t in the neighbourhood of t_0

$$\left|f(t) - f(t_0)\right| \le A\left|t - t_0\right|^{\alpha} \qquad \textbf{(A - 2)}$$

2. Let n be a positive integer. A function f(t) is said to be Lipschitz α, for $n < \alpha \le n+1$, at a point t_0, if and only if there exists a constant A and a polynomial P_n (t) of order n such that for all points t in a neighbourhood of t_0

$$\left|f(t) - P_n(t)\right| \le A\left|t - t_0\right|^{\alpha} \qquad \textbf{(A - 3)}$$

3. Let *f(t)* be a tempered distribution of finite order on an interval (t_1, t_2). The distribution *f(t)* is said to be uniformly Lipschitz α on (t_1, t_2) if and only if its primitive is uniformly Lipschitz α+1 on (t_1, t_2). It is necessary to define properly the notion of negative Lipschitz exponents for tempered distributions because they are encountered in numerical computations.

A classical tool for measuring the Lipschitz regularity of a function *f(t)* is to look at the asymptotic decay of its Fourier transform $\hat{f}(\omega)$. It can be proved that if a bounded function *f(t)* is uniformly Lipschitz α over **R** then it satisfies:

$$\int_{-\infty}^{+\infty} |\hat{f}(\omega)|(1+|\omega|^{\alpha})d\omega < +\infty \qquad \textbf{(A - 4)}$$

Wavelet transform has similar properties as Fourier transform, so the Lipschitz regularity of a function can be related to its wavelet transform. If the wavelet has a compact support, the value of $WT_f(s, t_0)$ depends upon the values of *f(t)* in the neighbourhood of t_0 of size proportional to the scale s, where $|WT_f(s,t)|$ is the wavelet transform over scale and time.

At fine scales, it provides localised information on *f(t)*. The following theorems give the relations between the asymptotic decay of the wavelet transform at small scales and the local Lipschitz regularity of the function. The wavelet $\psi(t)$ is supposed continuously differentiable and that it has a compact support although this last condition is not strictly necessary.

Theorem 1

Let $f(x) \in L^2(R)$. The function *f(t)* is uniformly Lipschitz α over intervals $(t_1 + \varepsilon, t_2 - \varepsilon)$ for any $\varepsilon > 0$, if and only if for any $\varepsilon > 0$, there exists a constant A_ε such that for any $x \in (t_1 + \varepsilon, t_2 - \varepsilon)$ and any scale s,

$$|WT_f(s,t)| \le A_\varepsilon s^{\alpha} \qquad \textbf{(A - 5)}$$

Theorem 2

Let wavelet $\psi(t)$ have n vanishing moments in n times continuously differentiable and have a compact support, and $f(t) \in L^2(R)$. If *f(t)* is Lipschitz α at t_0, $0 \le \alpha \le n$, then there exists a constant A such that for all point t in the neighbourhood of t_0 and any scale s

$$|WT_f(s,t)| \le A\ (s^{\alpha} + |t - t_0|^{\alpha}) \qquad \textbf{(A - 6)}$$

Conversely, *f(t)* is Lipschitz α at t_0, $0 < \alpha \le n$, if the two following conditions holds.

(1) There exists $\varepsilon > 0$ and a constant A such that for all points t in the neighbourhood of t_0 and any scale s

$$\left|WT_f(s,t)\right| \le As^{\varepsilon} \qquad \textbf{(A - 7)}$$

(2) There exists a constant B such that for all points t in the neighbourhood of t_0 and any scale s

$$\left|WT_f(s,t)\right| \le A(s^{\alpha} + \frac{\left|t - {t_0}^{\alpha}\right|}{\left|\log\left|t - t_0\right|\right|}) \qquad \textbf{(A - 8)}$$

Theorem 1 and 2 prove that the wavelet transform is particularly well suited to estimate the local regularity of functions. However, a direct application of theorems 1 and 2 is quite ineffective to detect singularities and to characterise their Lipschitz exponents. These theorems measure the decay of $\left|WT_f(s,t)\right|$ in the two-dimensional neighbourhood of t_0 in scale-space (s, t), which requires a lot of computations. In fact, the detection of singularity points is naturally related to the behaviour of the wavelet transform local maxima.

Theorem 3

Let ψ(t) be a wavelet with compact support and n vanishing moments in n times continuously differentiable. Let $f(t) \in L^1([t_1, t_2])$. If there exists a scale $s_0 > 0$ such that for all scales $s < s_0$ and $t \in (t_1, t_2)$, $\left|WT_f(s,t)\right|$ has no modulus maxima, then for any ε > 0, *f(t)* is uniformly Lipschitz n on (a+ε, b+ε).

Local extrema of the wavelet transform of *f(t)* is: any point (s_0, t_0) such that

$$\frac{\partial WT_f(s_0,t_0)}{\partial t} = 0$$

And the modulus maxima of the wavelet transform of *f(t)* was at any point (s_0, t_0) such that when t belongs to either the right or the left neighbourhood of t_0, $\left|WT_f(s_0,t)\right| < \left|WT_f(s_0,t_0)\right|$ and when t belongs to the other side of the neighbourhood of t_0, $\left|WT_f(s_0,t)\right| \le \left|WT_f(s_0,t_0)\right|$. A maxima line is any connected curve in the scale space along which all points are modulus maxima of the wavelet transform.

Based on theorem 3 and the definition of local extrema and modulus maxima, a corollary can be made: the closure of the set of points where *f(t)* is not Lipschitz n is included in the closure of the wavelet transform maxima of *f(t)*. The corollary proves that all the singularities of *f(t)* can be located by following the maxima lines when the scale goes to zero. Theorem 4 will characterises a particular class of isolated singularities from the behaviour of the wavelet modulus maxima.

Theorem 4

Let *f(t)* be a tempered distribution whose wavelet transform is defined over (t_1, t_2). and let $t_0 \in (t_1, t_2)$. There exists a scale $s_0 > 0$ and a constant C such that for $t_0 \in (t_1, t_2)$ and $s < s_0$, all the maxima of $WT_f(s, t)$ belong to a cone defined by

$$\left|t - t_0\right| \leq Cs \qquad \textbf{(A - 9)}$$

Then, at all points, $t_{01} \in (t_1, t_2)$, $t_{01} \neq t_0$, f(t) is uniformly Lipschitz n in the neighbourhood of t_{01}. The function f(t) is Lipschitz α at t_0, for $\alpha \leq n$, if and only if there exists a constant A such that along each maxima line in the cone defined by Equation (A-9),

$$\left|WT_f(s,t)\right| \leq As^{\varepsilon} \qquad \textbf{(A - 10)}$$

Equation (A-10) is equivalent to

$$\log\left|WT_f(s,t)\right| \leq \log(A) + \alpha \log(s) \qquad \textbf{(A - 11)}$$

By Equation (A-11), the Lipschitz regularity at t_0 is the maximum slope of maxima line of wavelet transform that remains above $\log\left|WT_f(s,t)\right|$ on a logarithmic scale. Therefore, the singularities and irregularity of a function can be detected from the wavelet transform modulus maxima. Therefore, a signal can be represented by its local extrema of wavelet transform.

Appendix B

The Adaptive Resonance Theory And Its Algorithm

The basic ART architecture as shown in Figure B-1 consists of three groups of neurones in two subsystems - attentional and orienting subsystems: an input pre-processing field F1 layer, the cluster units F2 layer, and a reset mechanism to control the degree of similarity of patterns placed on the same cluster. Procedures of bottom-up and top-down between F1 and F2 carry out adaptive filtering and short term memory, modulation mapping and encoding. The orienting subsystem generates a reset wave to F2 when mismatches between bottom-up and top-down patterns occur at F1. This reset wave selectively and enduringly inhibits active F2 cells when the input is shut off. Each time a pattern is presented, an appropriate cluster unit is chosen and that cluster's weights are adjusted to let the cluster unit learn the pattern.

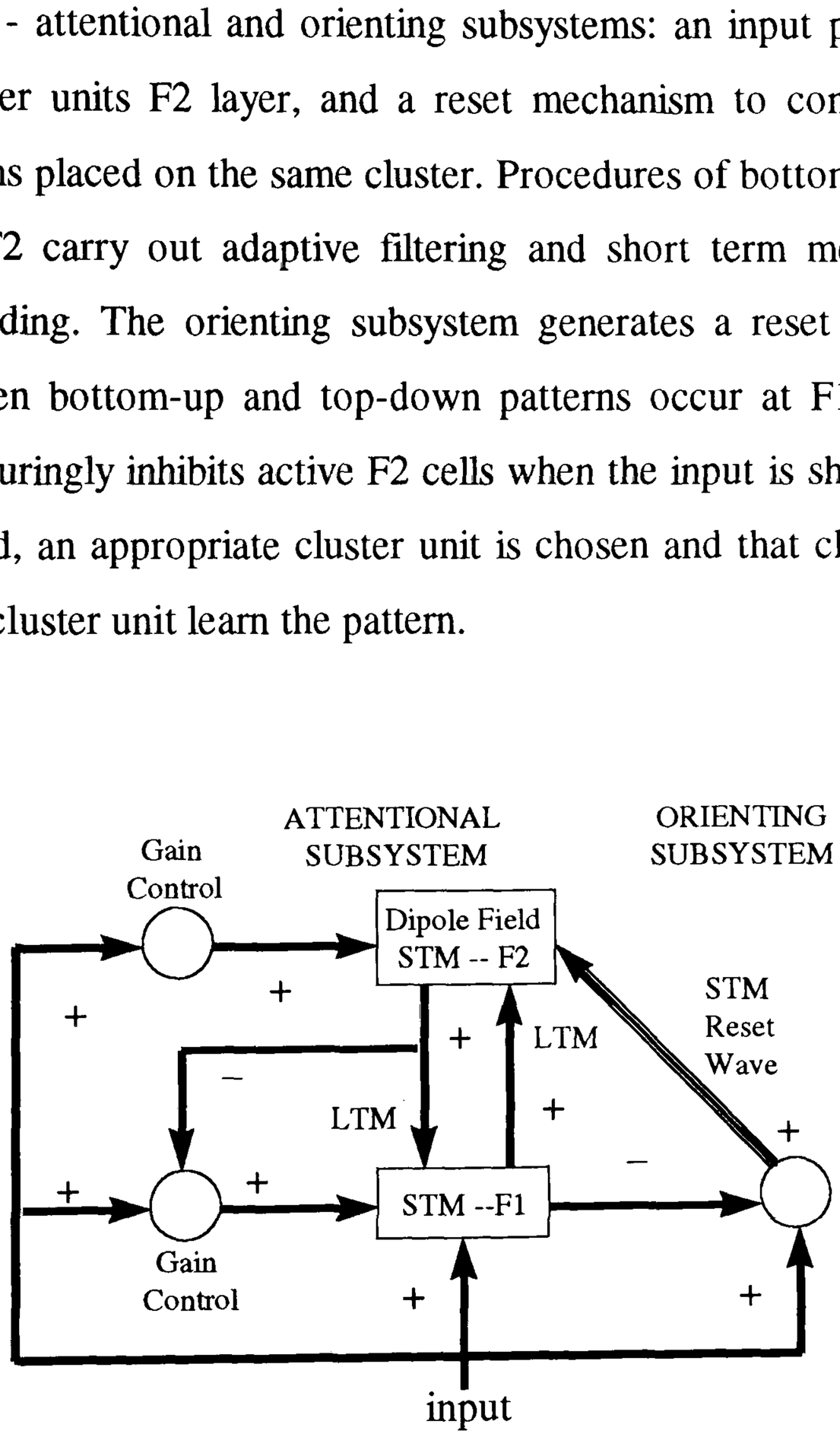


Figure B - 1 The ART network architecture

There are several architectures which can be used to implement ART2. The typical architecture of the ART2 network is illustrated in Figure B-2. The F1 layer is compound for six types of units, *W*, *X*, *U*, *V*, *P*, and *Q*. There are n units of each of these types,

PAGE NUMBERING AS ORIGINAL

Nomenclature

A – constant

a – wavelet dilation parameter

A(•) – activation function in the FFNN

a_0, b_0 -- discrete wavelet transform parameters

b – wavelet translation parameter

C – class vector

C_ψ – defined as Equation 3-6

CF – confidence factor

c_i – random uncorrelated coefficient defined as Equation 3-13

C_{noise} – constant to adjust signal to noise ratio

$d_k(x)$ – discriminant function of piece-wise

d_s – density of local extrema of wavelet transform

E – over all error in the fuzz-FFNN

E[•] – expected value

Eu – Euclidean distance

$F(\omega) = \hat{f}(\omega)$ -- Fourier transform

G – wavelet synthesis filter

g(k) – kth wavelet synthesis filter

g(t) – window function of transform

H – wavelet analysis filter

h(k) – kth wavelet analysis filter

m, n – discrete wavelet transform parameters

n(t) – white noise

N_{attr} – number of attributes in an input vector of clustering

NF – number of fuzzy regions

NI – input variables of fuzzy-FFNN

N_{in} – number of input patterns in clustering

NP – number of patterns in a class

o_j – output of the jth neuron in the fuzzy-FFNN

p – pattern matrix

QS(x, t) – qualitative state

s – scale

$\tilde{T}$ -- target value in fuzzy-FFNN training

t – time

w_{ij} – weights in the fuzzy-FFNN

Wn(t) – wavelet transform of white noise

$WT_f(s,t)$ – wavelet transform of a function

X – feature vector

$\|x\|_p$ -- p order norm

x – input of fuzzy-FFNN

$\tilde{Y}$ – output of fuzzy-FFNN

Z – integrates

z_i – exemplar vector in clustering

Greek Symbols

α -- Liptchitz exponent

η -- learning rate in the FFNN

μ -- fuzzy membership function

ρ -- distance threshold of clustering

σ -- variance

ψ -- wavelet function

$\phi(t)$ -- wavelet scale function or orthogonal function

ω -- frequency (domain)

References

Abe, S. and M.S. Lan, 1995a, A Method for Fuzzy Rules Extraction Directly from Numerical Data and its Application to Pattern Classification, *IEEE Trans. on Fuzzy Systems,* **3**(91), pp18-28.

Abe, S and M. S. Lan, 1995b, Fuzzy Rule Extraction Directly from Numerical Data for Function Approximation, *IEEE Trans. On System, Man and Cybernetics* 25(1) pp119-129

Anderberg, M. R., 1973, Cluster Analysis for Applications, Academic Press, New York.

Bader, F. P. and T. W. Tucker, 1987, Data Compression Applied to A Chemical Plant Using A Distributed Historian Station, *ISA Trans.* **26**(4) pp9-14

Bakshi, B. R. and G. Stephanopoulos, 1994, Representation of Process Trends -- III. Multiscale Extraction of Trends from Process Data, *Computers Chem Eng* **18**(4) 267-302

Bakshi, B. R. and G. Stephanopoulos, 1996, Compression of Chemical Process Data by Function Approximation and Feature Extraction, *AIChE J* **42**(2) pp477-492.

Berman, Z. and J. S. Baras, 1993, Properties of the Multi-Scale Maxima and Zero-Crossing Representations, *IEEE Trans. Signal Processing* **41** pp3216-3231

Bourlard, H. and Y. Kamp, 1988, Auto-association by Multi-layer Percetrons and Singular Value Decomposition, *Biol. Cybern.* **59** pp291-294

Bristol, E. H., 1990, Swinging Door Trending: Adaptive Trend Recording, *ISA National Conf. Proc* pp749-753

Candy, J. V., 1988, Singnal Processing the Modern Approach, McGra-Hill Book Company

Carpenter, G. A. and S. Grossberg, 1987a, A Massively Parallel Architecture for A Self-Organizing Neural Pattern Recognition Machine, *Computer Vision, Graphics, and Image Processing*, **37** pp54-115.

Carpenter, G. A. and S. Grossberg, 1987b, ART2: Self-Organization of Stable Category Recognition Codes for Analogue Input Patterns, *Applied Optics*, **26** pp4919-4930.

Chan, C.C., 1991, Incremental Learning of Production Rules from Examples under Uncertainty: Rough Set Approach, *Int. J. of Software Engng and Knowledge Engng.* **1**(4), 439-461.

Chang, C. C. and C. C. Yu, 1990, On-line Fault Diagnosis Using the Sighed Directed Graph, *Ind Eng Chem Res* 29 pp1290-1299.

Chen, F.Z. and X.Z. Wang, 1998, Software Sensor Design Using Bayesian Automatic Classification and Backpropagation Neural Networks. *Ind. Eng. Chem. Res.*, **37**, 3985-3991.

Chen, F.Z. and X.Z. Wang, 1999, An Integrated Data Mining Systems and its Application to Process Operational Data Analysis, *Comput. Chem Engng.* **24,** Suppl. (ESCPE-9).

Cheung, J. T. Y. and G. Stephanopoulos, 1990, Representation pf Process Trends -- I. A Formal Representation Framework, *Comp Chem Eng* **14**(4/5) pp495-510

Chmielewski, M.R., J.W Grzymala-Busse, N.W. Peterson and S. Than, 1993, The Rule Induction System LERS - A Version for Personal Computers, *Foundations of Computing and Decision Sciences,* **18**(3/4) pp181-212.

Chryssolouris, G., M. Lee and A. Ramsey, 1996, Confidence Interval Prediction for Neural Network Models, *IEEE trans. on Neural Networks* **7**(1) pp229-232

Chui, C. K., 1992, An Introduction to Wavelets, Academic Press, New York

Clarke-Pringle, T. and MacGregor, J.F., 1998, Product Quality Control in Reduced Dimensional Spaces, *Ind. Eng. Chem. Res.,* 37, 3992-4002.

Cox, E., 1994, The Fuzzy Systems Handbook: A Practitioner's Guide to Building, Using, and Maintaining Fuzzy Systems, Academic Press

Cvelkovic, Z and M. Vetterli, 1995, Discrete-Time Wavelet Extrema Representation: Design and Consistent Reconstruction, *IEEE Trans. On SP* **43**(3) pp681-692

Daubechies, I., 1992, Ten Lectures on Wavelets, Society of Industrial and Applied mathematics, Philadelphia, Pennsylvania.

Dai, X., B. Joseph and R. L. Motard, 1995, Process signal features analysis, *Wavelet Application in Chemical Engineering* edited by R. L. Mortard and B. Joseph, Kluwer Academic Publishers.

Dong, D. and T. J. McAvoy, 1996, Non-linear Principal Component Analysis – Based on Principal Curves and Neural Networks, *Computers Chem Engng* **20**(1) pp65-78.

Etezadi-Amoli, J. and R. P. McDonald, 1993. A Second Generation Non-linear Factor Analysis. *Psychometrika* **48**(3) pp315-327

Fathi, Z, J. Korbicz, W. F. Ramirez and G. Gilliland, 1992. Integrate Diagnostic System Using Analytical Redundancy and Artificial Intelligence for Coal-Fired Power Plant, IFAC Symposia Series No 9 p193-198

Fu, LiMin, 1994, Rule Generation from Neural Networks, *IEEE Trans. on Systems, Man, and Cybernetics,* **24**(8) pp1114-1124.

Gabor, D. 1946, Theory of communication, *J. Inst. Elec. Eng.* **93** pp429-441

Gallant, S. I., 1988, Connectionist Expert Systems, C*ommunications of the ACM* **31**(2) pp152-169.

Gallant, S.I., 1993, Neural Network Learning and Expert Systems. Cambridge, Mass. MIT Press.

Gnanadesikian, R., 1989, Methods for Statistical Data Analysis of Multivariate Observations, Wiley, New York

Grossberg, S. 1976, Adaptive Pattern Classification and Universal Re-Coding, I: Parallel Development and Coding of Neural Feature Detectors, *Biological Cybernetics*, **23** pp.121-134.

Grossmann, A., 1986, Wavelet Transform and Edge Detection, *in Stochastic Processes in Physics and Engineering*, M. Hazewinkel, Ed. Didrecht: Reidel

Hale, J. C. and H. L. Sellars, 1981, Historical Data Recording for Process Computers, *Chem. Engng Prog.* Pp38-43

Hastie, T. and W. Stuetzle, 1989, Principal Curves, *J. Am. Stat. Ass.* **84**(406) pp502-516

Hayashi, Y., 1990, A Neural Expert System with Automated Extraction of Fuzzy If-Then Rules and its Application to Medical Diagnosis, *in Advances in neural information processing systems* Vol 3 San Mateo, CA: Morgan Kaufman

Hernandez, E. and G. Weiss, 1996, A First Course on Wavelets, CRC Press

Janusz, M. E. and V. Venkatasubramanian, 1991, Automatic Generation of Qualitative Descriptions of Process Trends for Fault Detection and Diagnosis, *Engng Applic Artif Intell* **4**(5) pp329-339

Joshi, A., A. Kumar and R. L. Motard 1995. Trend analysis using the Frazier-Jawerth Transform, *Wavelet Application in Chemical Engineering* edited by R. L. Mortard and B. Joseph, pp115-137 Kluwer Academic Publishers

Kohonen, T., 1982, Self-Organised Formation of Topologically Correct Feature Maps. *Biological Cybernetics* **43** pp59-69

Kohonen, T., 1995, Self-organised Maps. Berlin: Springer.

Koivo, H. N., 1994, The Artificial Neural Networks in Fault Diagnosis and Control. *Control Engineering Practice*, **2**(1) pp.89-101.

Kramer, M. A., 1992, Auto-associative Neural Networks, *Computers Chem Engng* **16**(4) pp313-328

Leonard, A., M. A. Kramer and L. H. Ungar, 1992, A Neural Network Architecture That Computes its Own Reliability, *Computers Chem Engng* **16**(9) pp819-835

Lindheim, C and K. Lien, 1997, Operator Support Systems for New Kinds of Process Operation Work, *Computers & Chem Engng*, **21** suppl., pp s113-118

Looney, C. G. 1997, Pattern Recognition Using Neural Networks: Theory and Algorithms for Engineers and Scientist, Oxford University Press

MacQueen, J. B., 1967, Some Methods for Classification and Analysis of Multivariate Observations, *Proc. Symp. Math. Statist. and Probability 5th*, Berkeley, AD669871, Univeristy of California Press Berkeley, **1** pp281-297

Mah, R. S. H., A. C. Tamhane, S. H. Tung and A. N. Patel, 1995, Process Trending with Piece Wise Linear Smoothing, *Computers & Chem Engng*, **19**(2) pp129-137

Mallat S. and S. Zhong, 1992, Characterisation of Signals from Multi-scale Edges. *IEEE Trans. Pattern Analysis and Machine Intelligence* **14**(7), 710-732 .

Mallat, S. and W. L Hwang, 1992, Singularity Detection and Processing with Wavelets, *IEEE Trans. Inform. Theory*, **38**(2) pp617-643

Mallat, S., 1991, Zero-Crossing of A Wavelet Transform, *IEEE Trans. On Information Theory* **37**(4) pp1019-1033

Michalski, R. S., 1983, A Theory and Methodology of Inductive Learning. In: *Maching Learning: An Artificial Intelligence Approach*, ed R. S. Michalskim J. G. Carbonell and T. M. Mitchell. Tioga Palo Alto, CA

Michalski, R. S., I. Bratko and M. Kubat, 1998. Machine Learning and Data Mining: Methods and Applications, London:John Wiley & sons

Morris, A.J., G.A. Montague and Willis, M.J.. 1994, Artificial Neural Networks: Studies in Process Modelling and Control. *Trans. IChemE.* **72A,** 3-19.

Palus, M. and I. Dvrak, 1992, Singular-Value Decomposition in Attractor Reconstruction: Pitfalls and Precautions, *Physica* D **55**, pp221-234

Pao, Y. H., 1989, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley Publishing Company, Inc.

Pawlak, Z., 1982, Rough Sets, *Int.J. Computer and Information*, **11**, pp341-356.

Pawlak, Z., 1984, Rough Classification, *Int. J. Man-machine Studies,* **20** pp469-483.

Pawlak, Z., 1985, Rough Sets and Fuzzy Sets, *Fuzzy Sets and Systems,* **17** pp99-102.

Quafafou, M. and C.C. Chan, 1995, An Incremental Approach for Learning Fuzzy Rules from Examples, *EUFIT' 95*, Aachen, Germany, pp520-523.

Quinlan, J. R., 1986, Induction of Decision Trees. *Maching Learning Journal* **1** pp81-106

Ripley, B. D. 1996, Pattern Recognition And Neural Networks, Cambridge University Press

Rumeelhard, D., G. Hinton and R. Williams, 1986, Learning Internal Representations by Error Propagation, *In Parallel Distributed Processing,* MIT Press Cambridge MA

Saelid, S., A. Mjaavatten and K. Fjalestad, 1991, An Object-oriented Operator Support System Based on Process Models and Expert System Shell, *Computers Chem. Engng.*, **15** pp s97-s108.

Saito, K. and R. Nakano, 1988, Medical Diagnostic Expert System Based on PDP Model, in *Proceedings of ICNN*, pp255-262

Shao, R., E. B. Martin, J. Chen and A. J. Morris, 1997, Confidence Bounds for Neural Network Representations, *Computers Chem Engng* **21** suppl pps1173-1178

Shih, R. F. and L. S. Lee, 1995, Use of Fuzzy Cause-Effect Digraph for Resolution Fault Diagnosis for Process Plants. 2. Diagnostic Algorithm and Applications , *Ind Eng Chem Res* **34** pp1703-1717

Srinivasan, A. C. Batur and C. C. Chan, 1993, Using Inductive Learning to Determine Fuzzy Rules for Dynamic System, *Engng Applic. of Artif. Intell.,* **6**(3) pp257-264.

Sudkamp, T. and R.J. Hammell II, 1994, Interpolation. Completion. and Learning Fuzzy Rules, *IEEE Trans. on Systems, Man, and Cybernetics*. **24**. (2). 332-342.

Sylvester, J. J., 1889, On the Reduction of A Bi-line Quantic of The Nth Order to Form of A Sum of N Products by A Double Orthogonal Substitution. *Mess Math*. **19**. pp42-46

Vedam, V. and V. Venkatasubramanian, 1998, A B-spline Based Method for Data Compression, Process Monitoring And Diagnosis, *Computers Chem Engng* **22** suppl pps827-830

Vinson J. M. and L. H. Ungar, 1995, Dynamic Process Monitoring and Fault Diagnosis with Qualitative Models, *IEEE Trans. On System, Man and Cybernetics*, **25**(1) pp.181-189.

Wang, D. L., 1993, Pattern Recognition: Neural Networks in Perspective, *IEEE expert*, pp52-60

Wang, L.X. and J.M. Mendel, 1992, Generating Fuzzy Rules by Learning from Examples, *IEEE Trans. on Systems, Man, and Cybernetics,* **22**(6) pp1414-1427.

Wang, L.X., 1992, Fuzzy Systems Are Universal Approximators, *Proc. IEEE Int. Conf. on Fuzzy Systems,* San Diego, pp1163-1170.

Wang, X. Z. and B. H. Chen, 1998, Clustering of Infrared Spectra of Lubricating Base Oils Using Adaptive Resonance Theory, *J. Of Chemical Information And Computer Science* **38**(3) pp457-462

Wang, X. Z., B. H. Chen, S. H. Yang and C. McGreavy, 1996, Neural Net, Fuzzy Sets and Digraphs in Safety and Operability Studies of Refinery Reaction Processes, *Chem Engng Sci* **51**(10) pp2169-2178.

Wang, X. Z., B. H. CHEN, S. H. Yang, C. McGreavy and M. L. Lu, 1997, Fuzzy Rules Generation from Data for Operational support, *Computers Chem Engng* **21** suppl. pp s661-666

Wang, X.Z. and C. McGreavy, 1998, Automatic Classification for Mining Process Operational Data, *Ind. Eng. Chem. Res.*, **37**, 2215-2222.

Whiteley, J. R. and J. F. Davis, 1994, A Similarity-Based Approach to Interpretation of Sensor Data Using Adaptive Resonance Theory, *Computers Chem Engng* **18**(7) pp673-661.

William, B. C., 1986, Doing Time: Putting Qualitative Reasoning on Firmer Ground. *National Conf. on Artificial Intelli.*, Philadelphia.

Wishart, D., 1969, FORTRAN II Programs for 8 Methods of Cluster Analysis (CLUSTN I) Comput. Contrib., 38 State Geological Survey. Univ. of Kansas. Lawrence.

Xu, L., E. Oja and C. Y. Suen, 1992, Modified Hebbian Learning for Curve and Surface Fitting, *Neural Networks* **5** pp441-457.

Zhang, J., Martin, E.B., Morris, A.J., 1997, Process Monitoring Using Non-linear Statistical Techniques, *Chem. Eng. J.* **67**, 181-189.