Tools and techniques

In this chapter, the background on the tools that are utilised in the development and implementation of the Fault Diagnosis System (FDS), to be proposed in the next Chapter 5, are described. First, Artificial Neural Network technology and Fuzzy Logic systems are treated. They are key tools in the proposed FDS. Then, the Hazard and Operability analysis, that is an important source of information for FDS development is described. Finally, some aspects of signal processing using wavelets are commented. This technique is used in the proposed FDS.

4.1. Artificial Neural Networks

Artificial Neural Networks (ANNs) are mathematical models that are designed and adjusted to perform some task like pattern recognition or predictions of variables. Their history can be traced back to the desire to model biological systems by mathematical models. In the ANN literature, it is claimed that ANN is taught and it learns new things. During the 20th century, the research in the area of ANNs has been growing by the development of different ANN architectures, faster training algorithms and a large number of applications. The applications are expanding because ANNs are good at solving problems, not just in engineering, science and mathematics, but in medicine, business, finance and literature as well. Their application to a wide variety of problems in many fields makes them very attractive. Also, faster computers and faster algorithms have made it possible to use ANNs to solve complex industrial problems that formerly required too much computation.

In this section, a brief review of the main concepts behind ANNs are introduced. Detailed theoretical treatment of ANNs can be found in the specialised literature (Freeman and Skapura, 1993; Hagan et al., 1995).

ANNs are networks consisting of simple, usually nonlinear, processing nodes which are "inspired" by the information processing in biological nervous systems. These processing units (see Figure 4.1) compute the product of an input vector *q* and a weight matrix *W* plus a bias *b*, leading to the activation status *n* -Equation (4.1). The subindex *m* corresponds to the node number, being S_0 the number of nodes in the layer 0, the input layer-.

$$n = \sum_{m=1}^{So} w_{1,m} \bullet q_m + b \qquad (4.1)$$

Subsequently, the activation status *n* is mapped to the output via an activation function *f*. It can be linear or nonlinear, for example a sigmoidal function (Equation (4.2)).

$$f(n) = \frac{1}{1+e^{-n}} \qquad (4.2)$$

Within the large group of ANN architectures, the multilayer perceptron is the best known example (Figure 4.2). It is composed of several layers, each of which has a certain number of processing nodes.

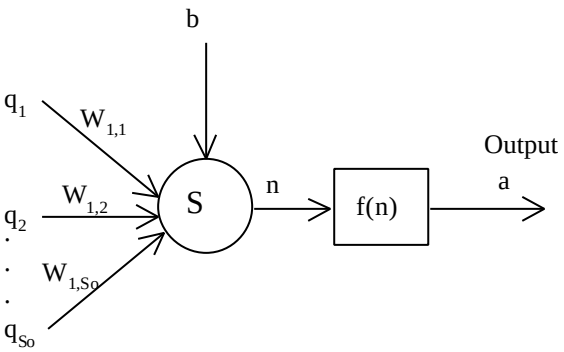

Figure 4.1. Processing node of an Artificial Neural Network

The procedure for modifying the weights and biases of an ANN, in order to train it to perform some task is called learning rule or training algorithm. There are many types of ANN learning rules. They fall into three broad categories: *supervised learning*, *unsupervised learning* and *reinforcement* (or graded) *learning*.

In *supervised learning*, the learning rule is provided with a set of examples (the training set) of proper ANN behaviour:

$\{q_1,g_1\},\{q_2,g_2\}, \ldots, \{q_V,g_V\}$;

where q_V is an input vector to the ANN and g_V is the corresponding correct (target) output. As the inputs are applied to the ANN, the ANN's output are compared to the targets. The learning rule is then used to adjust the weights and biases of the ANN in order to move the ANN outputs closer to the targets. The so called *backpropagation algorithm* falls in this supervised learning category.

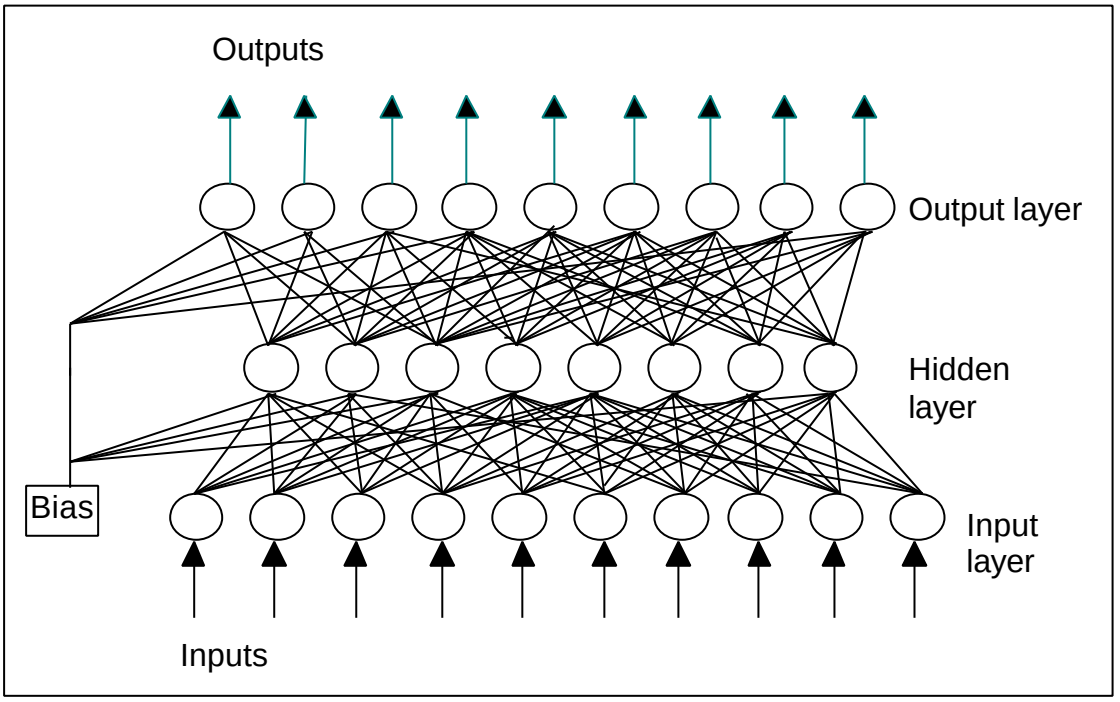

Figure 4.2. An example of a Multilayer Perceptron neural network

Reinforcement learning is similar to the supervised learning, except that, instead of being provided with the correct output for each ANN input, the algorithm is only given a grade. The grade (or score) is a measure of the ANN performance over some

sequence of inputs. This type of learning is much less common than supervised learning. It appears to be most suited to control system applications.

In *unsupervised learning*, the weights and biases are modified in response to ANN inputs only. There are no target outputs available. At first glance this might seem to be impractical. However, most of these algorithms perform some kind of clustering operation. They learn to categorise the input patterns into a finite number of classes.

4.1.1. Types of Artificial Neural Networks

Backpropagation networks

The previously shown multilayer perceptron is known as feedforward backpropagation neural network (BPN). In Figure 4.3, an example in abbreviated notation is shown. The subindices of terms *W*, *b*, *n*, *f* and *a* correspond to the layer number.

The training algorithm is the backpropagation one. This algorithm adopts the Generalized Delta Rule (GDR) designed to minimise the Mean Square Error MSE:

$$MSE = \sum_{v=1}^{V} \sum_{m=1}^{Sn} (g_m^v - a_m^v)^2 , \qquad (4.3)$$

where V and S_n denote the number of training patterns presented to the input layer and the number of units in the output layer, respectively, and g_m^v represents the desired value of the mth output element given the vth pattern, while a_m^v is the actual output of the same element.

Given the vth pattern, the updating weight in a supervised learning algorithm follows a general formulation:

$$w_{jm}^v = w_{jm}^{v-1} + \Delta w_{jm}^v , \qquad (4.4)$$

where w^v_{jm} denotes the weight of the connection between the jth element of the upper and the mth element of the lower layer, in the vth learning iteration. In GDR a weight change Dw^v_{jm} in Equation (4.4) is calculated as follows:

$$\Delta w_{jm}^{v} = h \bullet d_j^{v} \bullet O_m^{v} + a \bullet \Delta w_{jm}^{v-1} \quad (4.5)$$

where h and a denote the learning rate and the coefficient of the momentum term, respectively; O_m^{v} is the output value of the mth element in the previous layer. The momentum term prevents divergent oscillations. It also permits the increase of the rate of convergence. It consists of a value, always positive, lower than 1, that multiplies the previous weight change. In this way the changes are maintained in the same direction. The use of this additional term is optional.

The error signal of the *jth* element in the vth learning iteration d_j^{v} in Equation (4.5) is determined as follows:

If *j* belongs to the output layer:

$$d_j^{v} = (g_j^{v} - a_j^{v}) \bullet f_j'(\sum_m w_{jm}^{v} O_m^{v} + w_{j0}^{v}) \quad (4.6)$$

and if *j* belongs to the hidden layer(s):

$$d_j^{v} = f_j'(\sum_m w_{jm}^{v} O_m^{v} + w_{j0}^{v}) \bullet \sum_k d_k^{v} \bullet w_{kj}^{v} \quad (4.7)$$

where *f'* is the derivative of the transfer function given in Equation (4.2). In Equation (4.7) w^{y}_{kj} denotes the weight of the connection between the kth element of the upper and the jth element of the lower layer, in the vth learning iteration.

Therefore, GDR computes an error for each element in the output and hidden layers using Equations (4.6) and (4.7) and recursively updates the weights of all the layers using the Equation (4.5), starting from the output layer and working backwards until the input layer.

This basic algorithm has been improved with a heuristic modification: the adaptive learning rate. The learning rate, h has a significant effect in the network training performance. Normally, h has to be a small number (in the order of 0.05 and 0.25) to

be sure that the network can be settled on a solution. A small value of h implies that the network has to make a large number of iterations, but this is the cost for obtaining a good solution. It is possible to increase the value of h during the learning. Making it higher if the error goes down can accelerate the convergence because the steps to the minimum error are larger, but the network could be crashed going far away from the minimum value if h is too high.

A numerical optimisation technique, the Levenberg-Marquardt algorithm, that is a variation of Newton's method, can be used to speed up the convergence of backpropagation method.

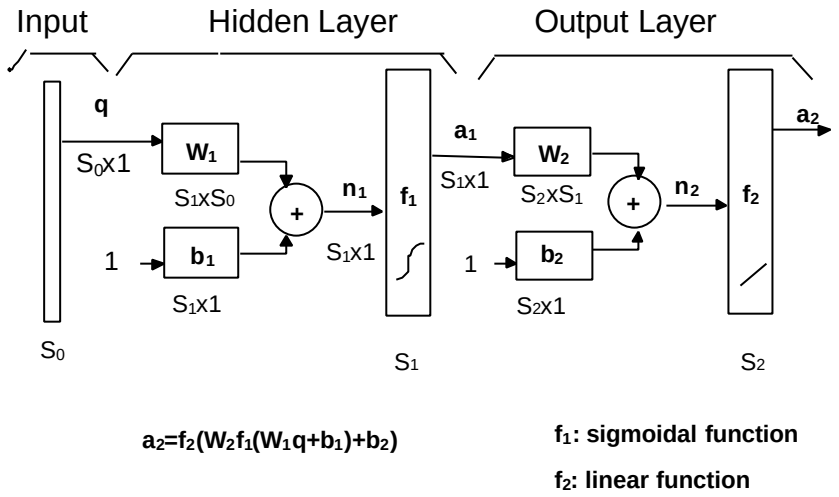

Figure 4.3. Backpropagation Network

Radial basis function networks

The Radial Basis Function Networks (RBFNs) have three layers. The outputs of the hidden layer neurons, each of which represents a basis function, are determined by the distance between the ANN input and the "centre" of the basis function. The output layer is linear and produces a weighted sum of outputs of the hidden layer. The neurons in the RBFN have localized receptive fields because they only respond to inputs that are close to their centres. This is in contrast to the standard BPN where the sigmoid function creates a global response. The RBFN trains faster than BPNs but requires many neurons for high dimensional input spaces.

The training algorithm starts with no neurons in the hidden layer and they are added until the network meets the specified mean squared error goal. The following steps are repeated until it is obtained that goal: 1- the ANN is simulated; 2- the input vector with the greatest error is found; 3- a neuron is added with weights equal to that vector; 4- the linear layer weights are redesigned to minimize the error.

The MSE considered should be zero and the parameter to be optimised is the spread of the radial basis function. The larger that spread the smoother the function approximation will be. Too large a spread means a lot of neurons will be required to fit a fast changing function. Too small a spread means many neurons will be required to fit a smooth function, and the ANN may not generalize well.

Self Organising Maps

In the two previous cases the ANNs need a supervised learning. In those cases, a learning rule is used to adjust the weight and biases of the ANN in order to move the ANN outputs closer to the targets. Otherwise, in unsupervised learning, the weights and biases are modified in response to ANN inputs only. Self Organising Maps (SOMs) learn to categorize the input pattern into a finite number of classes.

Competitive networks are characterized by the fact that the neurons compete with each other to determine which prototype pattern is most representative of the input pattern. This competition can be combined with associative learning rules to produce powerful SOMs. In Figure 4.4, a scheme of a SOM is shown. A parameter to be optimised in this kind of networks is the number of nodes of the feature map.

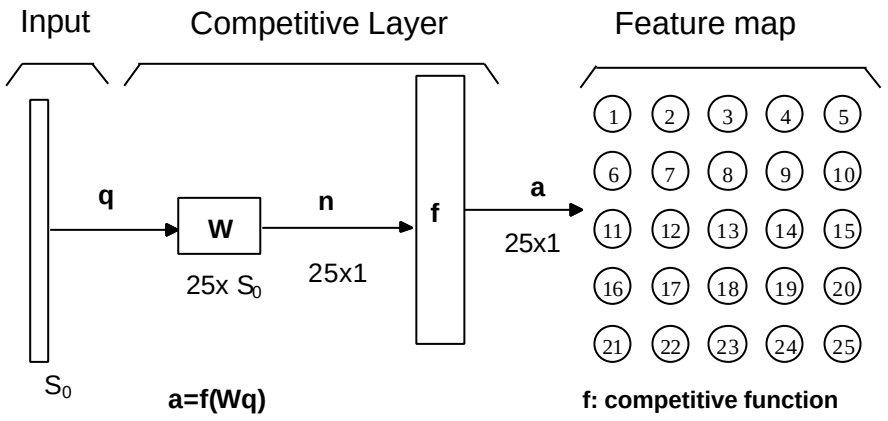

Figure 4.4. An example of a self organizing map with 25 active nodes

Probabilistic Neural Networks

Probabilistic Neural Networks (PNNs) are a kind of RBFN suitable for classification problems. The first layer has RBFN neurons and the second layer has competitive neurons (Wasserman, 1993).

4.1.2. Auto-associative neural network architecture and training

The Autoassociative Artificial Neural Network (AANN) is a BPN. It will be described separately due to its importance for performing NLPCA.

The AANN is composed of an input layer, three hidden layers and an output layer equal in dimension to the inputs (Figure 4.5). The first of the hidden layers is called the "mapping layer". The transfer functions of the nodes are sigmoidals. The second hidden layer is called the bottleneck layer. The transfer function in the nodes can be linear. Its dimension is required to be the smallest in the network. The third hidden layer is called the "demapping layer". The model transfer functions are usually sigmoidals. The bottleneck forces an internal encoding and compression of the inputs, with a subsequent decoding or decompression after the bottleneck to the network outputs. The mapping/demapping process represents a nonlinear generalization of PCA. The loss of information involved in this two stage process is measured by the sum of squares difference between inputs and outputs summed over the training set:

$$SPE = \sum_{v=1}^{V} \sum_{m=1}^{So} (Y_{v,m} - Y_{v,m}')^2 \qquad (4.8)$$

where V is the number of training set data, S_0 the number of input nodes (equal to the number of output nodes), Y is the matrix of the training set of inputs and Y' the respective matrix of neural network output.

Supervised training with Equation (4.8) as the objective is equivalent to train a backpropagation network to produce the identity mapping.

To prevent overfitting, the following inequality limiting the number of adjustable network parameters to a small fraction of the number of data points has been observed (Kramer, 1992).

$$Ml + Md << \frac{V \bullet (V - db)}{V + db + 1} \qquad (4.9)$$

where *Ml* is the number of nodes in the first hidden layer, *Md* is the number of nodes in the third hidden layer, and *db* the number of nodes in the bottleneck layer.

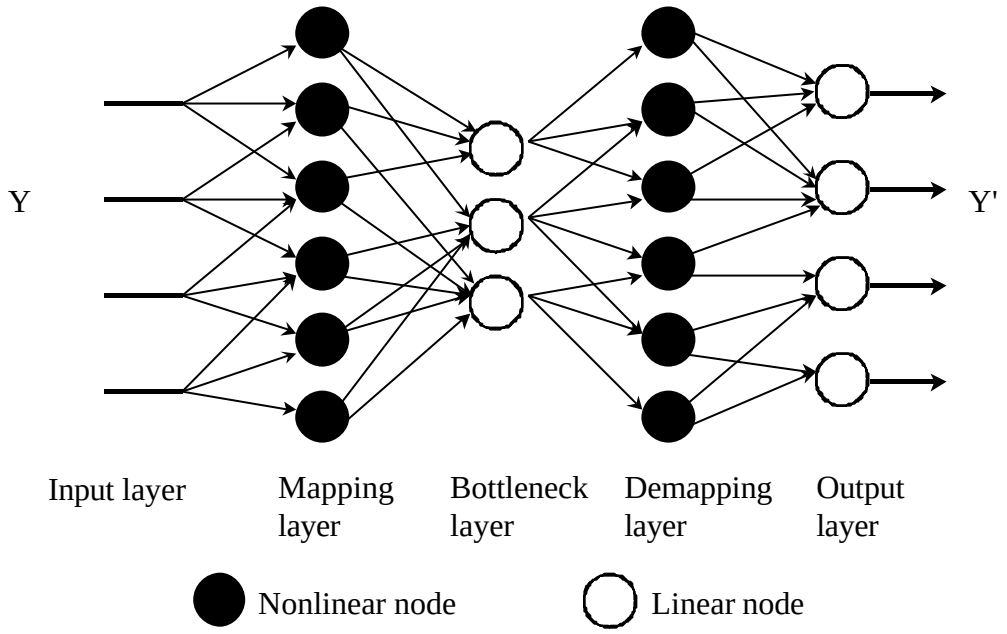

Figure 4.5. Autoassociative Neural Network

4.2. Fuzzy Logic

The concept of Fuzzy Logic (FL) has been introduced by Zadeh in 1965 as a way of characterising non probabilistic uncertainties. A FL system is unique in that it is able to simultaneously handle numerical and linguistic knowledge. It is a nonlinear mapping of an input data vector into a scalar output. Fuzzy set theory and FL establish the specifics of the nonlinear mapping. A large number of applications of FL have performed during the 20th century. They include control applications, scheduling and optimisation and signal analysis for training and interpretation.

In this section a brief summary of the basic aspects of fuzzy logic is shown. Detailed treatment can be found in the specialised literature (Mendel, 1995).

4.2.1. Definitions

If *X* is a collection of objects denoted generically by *x*, then a *fuzzy set A* in *X* is defined as a set of ordered pairs:

$$A = \{(x, m_A(x)) \mid x \in X\} \qquad (4.10)$$

$m_A(x)$ is called the *membership function* (MF for short) of *x* in *A*. The MF maps each element of *X* to a continuous membership value (or membership grade) between 0 and 1. Figure 4.6 shows an example of a fuzzy set with continuous *X*. Let *X* be the set of possible flowrates in a specific pipe, where the normal operating condition is a value of 2 m^3/hr. Then the fuzzy set *A* = "Normal flowrate" may be expressed using Equation (4.10) where, for example:

$$m_A(x) = \frac{1}{1 + \left(\frac{x-2}{0.2}\right)^4} \qquad (4.11)$$

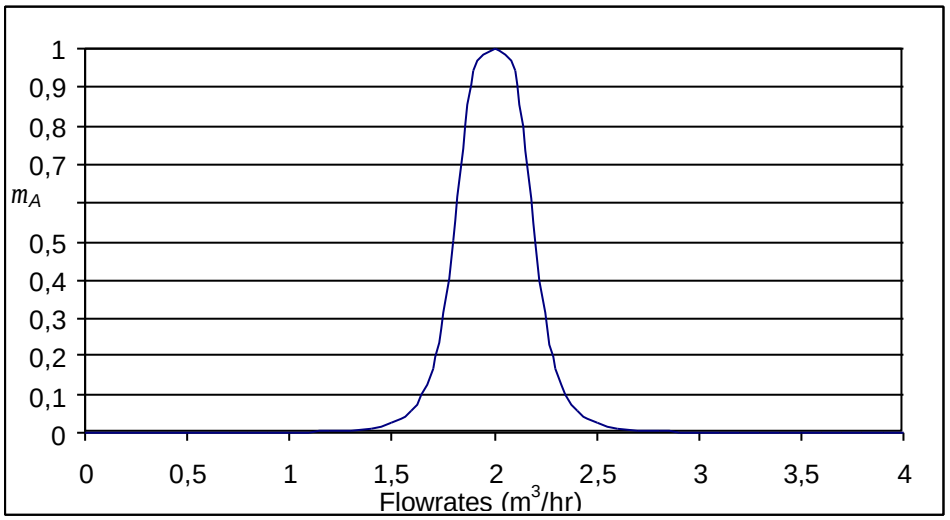

Figure 4.6. Example of a fuzzy set. MF on a continuous X = "Flowrates of a specific pipe", where the fuzzy set is "normal flowrate"

A fuzzy *if-then rule* assumes the form:

If *x* is *A* then *z* is *C*

where *A* and *C* are linguistic values defined by fuzzy sets on universes of discourse *X* and Z, respectively. For example: "if temperature is high then pressure is high". Often "*x* is *A*" is called the antecedent while "*z* is *C*" is called the consequent.

Other examples of fuzzy if-then rules are more complex because they include the logical operators AND, OR and NOT:

If x is A_1 AND y is B_1 then z is C_1

If x is A_2 OR y is B_2 then z is C_2

If x is NOT A_3 then z is C_3

where the subindex of A, B and C indicates different linguistic values defined by fuzzy sets on universes of discourses X, Y and Z, respectively.

Fuzzy reasoning, also known as approximate reasoning, is an inference procedure used to derive conclusions from a set of if-then rules and one or more conditions.

The logical operations AND, OR and NOT are applied to the membership functions instead of the input values directly. There are at least two different concepts to express them, namely:

The max-min notation

Fuzzy AND: $m_{C1}(z) = \min\left[m_{A1}(x), m_{B1}(y)\right]$ (4.12)

Fuzzy OR: $m_{C2}(z) = \max\left[m_{A2}(x), m_{B2}(y)\right]$ (4.13)

Fuzzy NOT: $m_{C3}(z) = 1 - m_{A3}(x)$ (4.14)

The prod-sum notation

Fuzzy AND: $m_{C1}(z) = m_{A1}(x) \bullet m_{B1}(y)$ (4.15)

Fuzzy OR: $m_{C2}(z) = 1 - m_{A2}(x) \bullet m_{B2}(y)$ (4.16)

Fuzzy NOT: $m_{C3}(z) = 1 - m_{A3}(x)$ (4.17)

The interpretation of multiple rules is usually taken as the union of the fuzzy relations corresponding to the fuzzy rules. For instance, given the following rules:

Rule 1: if x is A_1 AND y is B_1 then z is C_1

Rule 2: if x is A_2 AND y is B_2 then z is C_2 ;

the fuzzy reasoning shown in Figure 4.7 can be employed, using the min-max composition, to derive the resulting output fuzzy set *C'*. In this case, the consequent (conclusion) is: *z* is *C'.*

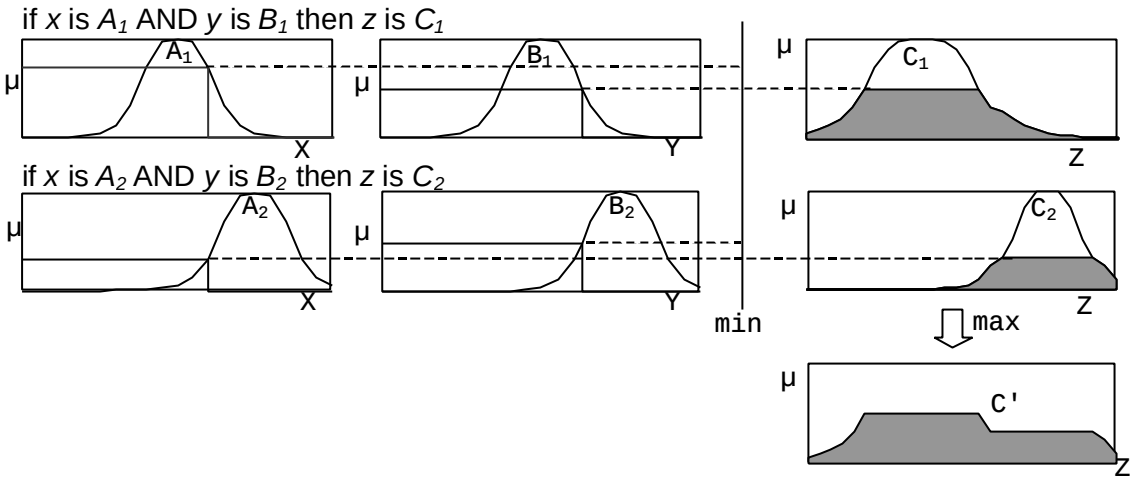

Figure 4.7. Fuzzy reasoning for multiple rules with multiple antecedents

4.2.2. Fuzzy logic systems

A *fuzzy inference system* or *Fuzzy Logic System (FLS)* is a computing framework based on the concepts of fuzzy set theory, fuzzy if-then rules, and fuzzy reasoning. The basic structure of a fuzzy inference system consists of three conceptual components: a rule base, which contains a selection of fuzzy rules, a database, which defines the membership functions used in the fuzzy rules, and a reasoning mechanism, which performs the inference procedure upon the rules and a given condition to derive a reasonable output or conclusion.

Figure 4.8 depicts a FLS that is widely used in fuzzy logic controllers and signal processing applications. The FLS maps crisp inputs to crisp outputs. It contains four components: rules, fuzzifier, inference engine, and defuzzifier. Once the rules have been established, a FLS can be viewed as a mapping from inputs to outputs. Rules can be provided by experts or can be extracted from numerical data and they are expressed as a collection of IF-THEN statements. The fuzzifier maps crisp numbers into fuzzy sets. The inference engine handles the way in which rules are combined and maps fuzzy sets into fuzzy sets. In many applications, crisp numbers must be obtained at the output of a FLS. The defuzzifier maps outputs fuzzy sets into crisp numbers.

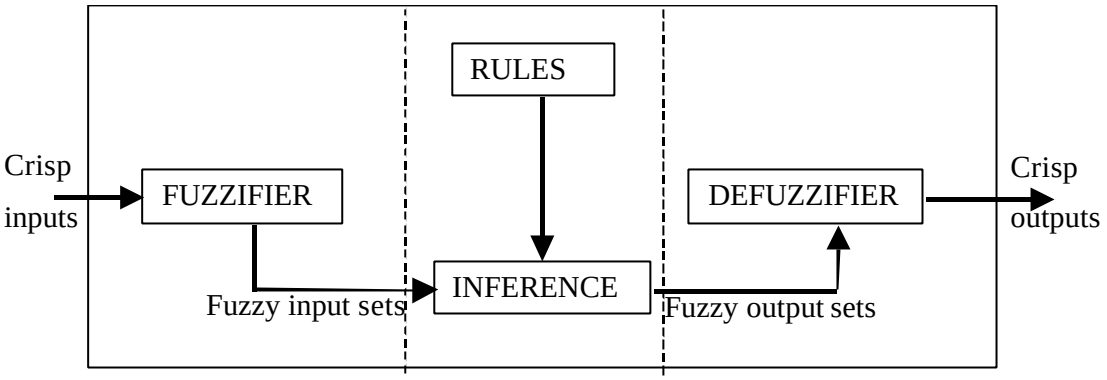

Figure 4.8. Fuzzy Logic System scheme

The defuzzification can be performed with some of the following methods:

- *Maximum*: the fuzzy set *C'* is examined and the value of *z* for which $m_{C'}(z)$ is a maximum is chosen as the output value (z'_{mx}).
- *Mean of maximum*: the fuzzy set C' is examined and the values of z for which $m_{C'}(z)$ is the maximum is first determined. The mean of these values is computed as the output (z'_{mn}).
- *Centroid*: the centre of gravity is determined:

$$\bar{z} = \frac{\int z \bullet m_{C'}(z)dz}{\int m_{C'}(z)dz} \qquad (4.18)$$

  chosen as its output value (z'_{ct}).

- *Height*: let $\bar{z}^{rl}$ denote the centre of gravity of the fuzzy set C'_r (which is associated with the activated of rule *rl*). This defuzzifier first evaluates $m_{C'r}(z)$ at $\bar{z}^{rl}$ and then computes the output of the FLS as:

$$z'_{ht} = \frac{\sum_{rl=1}^{Rl} \bar{z}^{rl} \bullet m_{C'rl}(\bar{z}^{rl})}{\sum_{rl=1}^{Rl} m_{C'rl}(\bar{z}^{rl})} \qquad (4.19)$$

  where *Rl* is the total number of rules.

The defuzzification methods applied to the example shown in Figure 4.7 are summarised in Figure 4.9.

In summary, a FLS performs the following succesive steps:

- *Fuzzification* of crisp values;
- *Inference* using a rule-base in which the logical operations are performed on the membership functions, using either Equations (4.12), (4.13) and (4.14), or Equations (4.15), (4.16) and (4.17), and aggregation;
- *Defuzzification* to obtain crisp outputs, using for example one of the methods shown, Equations (4.18) or (4.19).

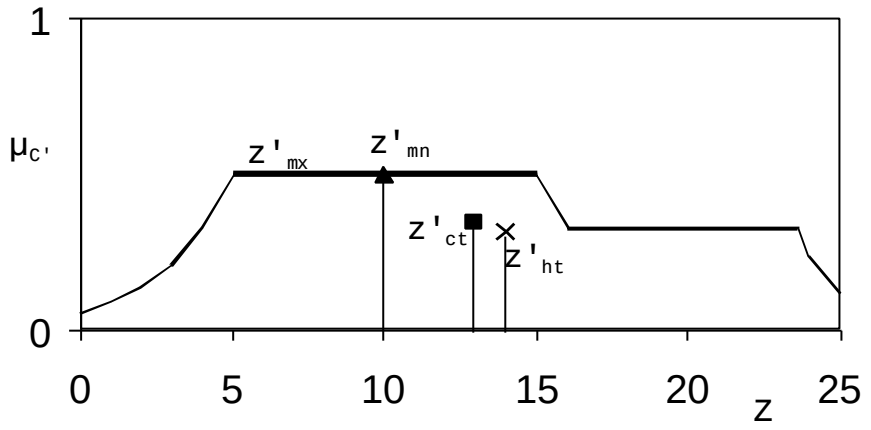

Figure 4.9. Defuzzification methods (z'_{mx}: maximum, z'_{mn}: mean of maximum, z'_{ct}: centroid, z'_{ht}: height) applied to the example shown in Figure 4.7.

4.2.3. Types of Fuzzy logic systems

Among the different FLSs, the most commonly used are the following (Jang and Sun, 1995):

- *Mamdami Fuzzy model*: the crisp inputs are fuzzified according to a set of membership functions. The fuzzy AND and OR operators in the if-then rules are inferred using min-max or product max compositions, or other variations. The fuzzy set obtained is defuzzified to a crisp value using a strategy like the centroid or the mean of maximum.
- *Sugeno Fuzzy model*: it was proposed to develop a systematic approach to generating fuzzy rules from a given input-output data set. A typical fuzzy rule in a Sugeno fuzzy model has the form: "if x is A and y is B then *z=f(x,y)*", where A and B

are fuzzy sets in the antecedent, while *z=f(x,y)* is a crisp function in the consequent. Usually *f(x,y)* is a polynomial in the input variables *x* and *y,* but it can be any function as long as it can appropriately describe the output of the system within the fuzzy region specified by the antecedent of the rule. When *f(x,y)* is a first-order polynomial, the resulting FLS is called a first-order Sugeno fuzzy model. In this case, each rule has a crisp output, the overall output is obtained via weighed average and thus the time consuming procedure of defuzzification is avoided.

- *Tsukamoto Fuzzy model*: the consequent of each fuzzy if-then rule is represented by a fuzzy set with monotonical membership functions. As a result, the inferred output of each rule is defined as a crisp value induced by the rule's firing strength. The overall output is taken as the weighted average at each rule's output. It avoids the time consuming process of defuzzification.

4.3. Hazard and Operability analysis

In this section, Hazard and Operability (HAZOP) analysis is described. Extending HAZOP analysis to fault diagnosis characterisation provides a more "down to the earth" approach for implementing an operator support system (Wennersten et al., 1996). Hence, it is an important source of information for the development of a FDS.

First, an introduction summarising the available techniques to identify potential accidents in the industry is presented. Then, the choice of the HAZOP technique is highlighted. Finally, a step by step description of the HAZOP analysis implementation is done.

4.3.1. Introduction

There are many techniques to identify potential accidents in the industry. Industrialised countries have standards that describe them. For example, in Spain, the references are: "Notas técnicas de prevención" (Technical Notes of Prevention) edited by Instituto Nacional de Seguridad e Higiene en el Trabajo (National Institute of Safety and Hygiene at Work) and "Guías técnicas" (Technical Guidelines) edited by Dirección General de Protección Civil (General Direction of Civil Protection).

There are qualitative and semi-quantitative methods for the identification of potential accidents in the industry (Casal et. al, 1999). Qualitative methods include safety reviews, historical analysis of accidents, Preliminary Hazard Analysis (PHA), checklists,

What if ...?, Hazard and Operability (HAZOP) analysis and Failure Modes and Effect Analysis (FMEA). Semi-quantitative methods include Fire and Explosion Index -DOW- (F&EI), Fault trees, Mond index, Substance Hazard Index (SHI), Material Hazard Index (MHI) and Event trees.

Some of the mentioned methods gives a first approximation of the potential hazards (safety reviews, PHA, checklists, F&EI). Other methods give a more detailed vision (HAZOP, What if ... ?, FMEA). Finally, the fault trees and the event trees give an elevated level of detail on situations of extreme gravity.

There are five limitations inherent to all the identification techniques:

- Exhaustive study: it is impossible to verify all the deviations that can occur.
- Reproducibility of the results: the same study carried out by different specialists can give different results.
- The fact that the conclusions are inextricable: the amount of documentation generated by the study and the lack of important details that can only be materialised by verbal communications in the work sessions, makes the analysis relatively difficult to understand.
- Importance of the experience: all the mentioned techniques are based on the acquired experience and the creativity of the analyst.
- Confidence level generated by the study: the subjectivity introduced in the valoration of the results can generate certain scepticism with respect to the study results.

In spite of the mentioned limitations, the experience shows that an adequate risk management allows to reduce the number of accidents significantly and the magnitude of the consequences are reduced.

4.3.2. The choice of the HAZOP analysis

In most industrialised countries, standards require to perform hazard analysis on a regular basis. HAZOP is the most widely used and recognised as the preferred approach in the chemical process industry. It is typically performed by a team of experts having specialised knowledge and expertise in the design, operation and maintenance of the plant.

HAZOP is also one of the most powerful hazard identification methods available and

has been well described in the literature. The imagination of a selected team is used to perturb a model of the system being studied by using a methodical procedure to identify potential accidents.

The system is studied one element at a time, in a "Top Down" fashion. The design intention of each element is defined and then questioned using "Guide-words" to produce deviations from the intention. The causes, consequences and safeguards for each deviation are then discussed and recorded. Some approaches to automate HAZOP analysis have been reported (Vaidhyanathan and Venkatasubramanian, 1996). They have as disadvantage the generation of a large number of irrelevant causes and consequences.

Figure 4.10 shows the logic diagram of HAZOP of a continuous process (Gillett, 1997). In the case of batch processes the HAZOP analysis examines every stage of the batch process sequence (Figure 4.11).

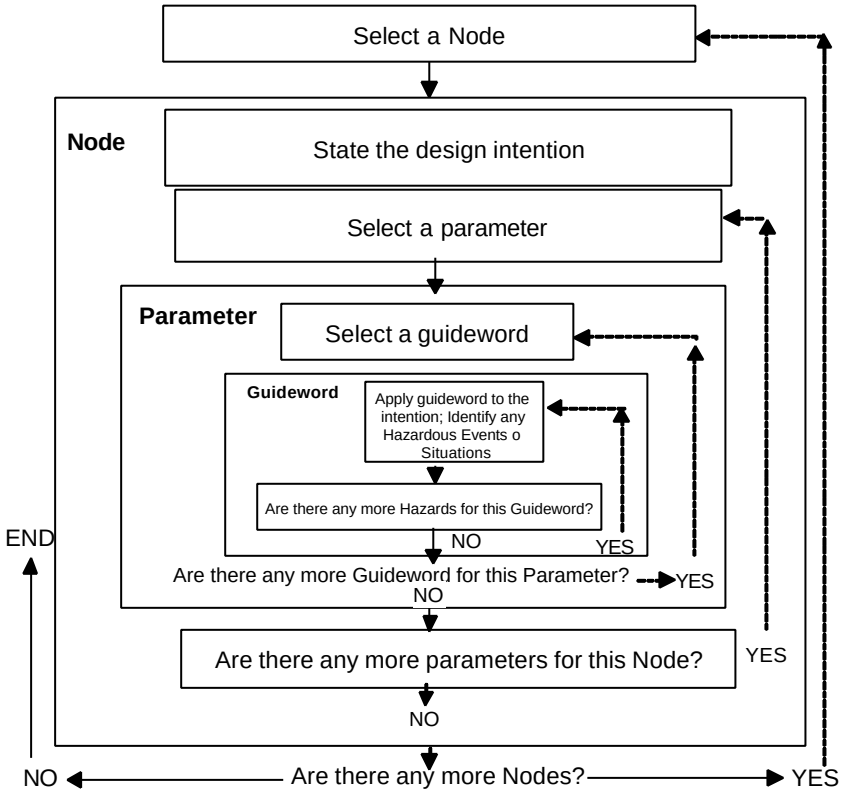

Figure 4.10. Logic diagram of a HAZOP analysis of continuous process

Table 4.1. HAZOP Guidewords

Guideword	**Example of a Typical Deviation**
No (not or none)	No flow in pipe No reactant in vessel
More of	Higher temperature Higher level
Less of	Lower velocity Lower bulk density
More than (or as well as)	Two phase flow Contamination
Less than (or part of)	Reduced concentration Missing component
Reverse (the complete opposite of the intent)	Valve closes instead of opening Heat rather than cool
Other than (a different intent)	Nonroutine operations Maintenance, cleaning, sampling, etc.
Sooner/later than	More/less time Operation out of sequence

4.3.3. HAZOP analysis for continuous processes

Since the procedure for continuous systems is simpler than that for batch systems, it will be described first.

i) Study the system model and subdivide it into its key elements (Nodes).

ii) Select an element (Node) for study and state the design intention of the element (Node). The design intention defines the processes or activities involved in the element and the boundary for examination. The intention will include details of the process parameters that can be changed in the element. Typical parameters stated in the intention are flow, temperature, pressure, level, and time.

iii) Select a parameter for study and apply the guidewords to the intention relating to the parameter selected and identify any deviations from the intent. Table 4.1 lists guidewords and gives brief examples of each.

iv) For each deviation identified, study the causes, effects and safeguards provided.

v) Decide whether the deviation requires a design change or corrective action and

record the decision and allocate the action to a team member for completion by an agreed-on review date.

vi) Once all of the guidewords have been applied to the parameter selected, select the next parameter.

vii) Repeat steps iii-vi until all of the parameters have been studied for the selected system element.

viii) Select the next element (Node) for study and repeat steps iii-vii until all of the system elements (Nodes) have been studied.

4.3.4. HAZOP for batch processes

The detailed hazard study examines every step of the batch process sequence. For each stage, the equipment used is studied element by element for each equipment state that may exist: "Active", "Inactive", or any other state. The parameters for each equipment state are then studied using guidewords. The detailed HAZOP analysis of a batch process must be used wisely as it can be incredibly time-consuming. A simplified logic diagram of the process is shown in Figure 4.11 and it is summarised as follows:

i) Select the first stage on the Activity Diagram or Operating Procedure.

ii) Select a system element in the stage (for example, an item of equipment and its associated connections to the system).

iii) Select the node in the equipment item (for example, part of the equipment item such as a valve or a filter).

iv) Select a state for the equipment node (active, inactive, or other state).

v) Select a parameter for the equipment node state chosen (temperature, pressure, level, etc.).

vi) Apply guidewords to the intention for the parameter at this state/node/system element/step. For each deviation identified, study the causes, effects, and safeguards provided. It is important to identify effects on the total system and the operating sequence. Decide whether the deviation requires a design or operating procedure change. Record the decision and allocate the action to a team member or completion by an agreed-on review date.

vii) Repeat steps v and vi for all parameters involved in the node/state chosen. Mark the node to indicate that it has been studied and move to the next node.

viii) Repeat steps iv-vii for all node states.

ix) Repeat steps iii-viii for all nodes.

x) Repeat steps ii-ix for all system elements.

xi) Repeat steps i-x for all batch process stages.

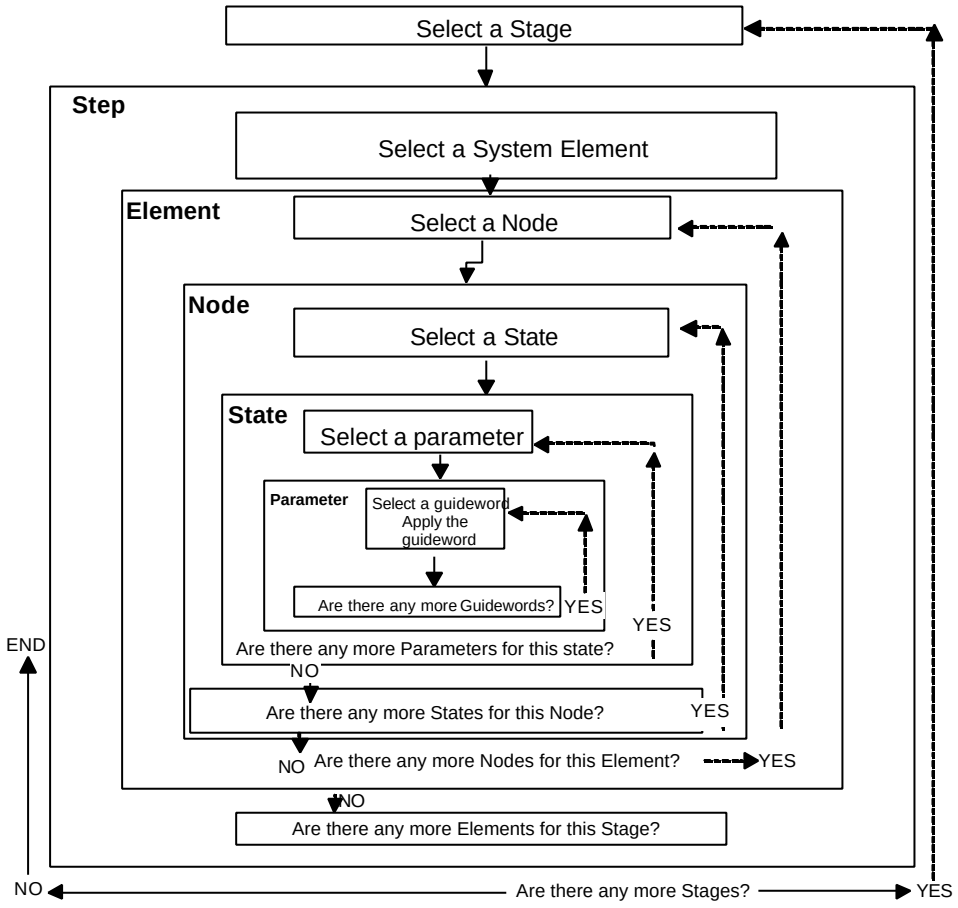

Figure 4.11. Logic diagram of HAZOP for batch processes

4.4. Signal processing using wavelets

Measured data from most processes contain contributions at multiple scales due to the occurrence of events with different localisation in time and frequency. Also, variables are usually measured at different rates or contain segments of missing data.

Most existing methods (Fourier transform, linear filters) represent the data in terms of basis functions at a fixed resolution or scale in time and frequency (as a weighted sum of a set of basis functions). In practice, it is rare for measured data to contain contributions at a single scale. Representation of the measurements in terms of basis

functions at a single scale will not permit efficient feature extraction or noise removal from a typical process signal. Noise removal by eliminating the high frequency contribution by Fourier transform will distort the localised features by excessive smoothing, since their high frequency components will also be removed.

Multiscale methods are ideally suited for the analysis and modelling of such multiscale data. The development of wavelets has resulted in several novel techniques for improved data analysis and empirical modelling. These methods exploit the multiscale representation of wavelet basis functions and their ability to compress deterministic features in a small number of large coefficients, and to approximately decorrelate a wide variety of stochastic processes.

Wavelets are a family of basis functions whose time-frequency localisation or scale is not the same in the entire time-frequency domain. Thus, wavelets possess multiscale character and are able to adjust their scale to the nature of the signal features.

The term wavelet refers to sets of functions of the form given by Equation (4.20):

$$\Psi_{dltl}(x) = |dl|^{-1/2} \bullet \Psi\left(\frac{x - tl}{dl}\right) \tag{4.20}$$

They are formed by the dilations which are controlled by the positive real number *dl*, and translations which are controlled by the real number *tl*, of a single function $y\ (x)$ often named the mother wavelet. Visually, the mother wavelet appears as a local oscillation. The dilation parameter *dl* controls the width and rate of this local oscillation and intuitively can be thought of controlling the frequency of $y_{dltl}(x)$. The translation parameter *tl* simply moves the wavelet throughout the domain.

If the dilation and the translation parameters *dl* and *tl* are chosen such that $dl=2^j$ and $tl=k2^j$, where *j* and *k* are integers, then there exist wavelets $y(x)$ such that the set of functions given by Equation (4.20) constitute an orthonormal basis of the space of functions or signals which have finite energy (Daubechies, 1992), and as above, the two parameters, *j* and *k* can be varied for analysis of local features of a given function. These two degrees of freedom, *j* and *k*, give one the ability to resolve features at a variety of scales by adjusting *j* and at any location by adjusting *k*.

Wavelet transform can be categorised into continuous and discrete. Continuous wavelet transform implies that the scaling and translation parameters *dl* and *tl* change continuously. However, calculating wavelet coefficients for every possible scale can

represent a considerable effort and result in a vast amount of data. Therefore discrete parameter wavelet transform is often used.

For many signals, the low frequency content is the most important part. The high frequency content, on the other hand provides flavour and nuance. In wavelet analysis the low frequency content is called the *approximation* and the high frequency content is called the *detail* parts. It is not necessary to preserve all the outputs from the filters. The decomposition can be iterated, with successive approximations being decomposed in turn, so that one signal is broken into many lower-resolution components. Figure 4.12 shows a scheme of an example of the multiresolution procedure. In this example, four steps and four scales are considered. In the first scale, the original signal is split into approximation *Ap1* and Detail *D1*. The detail *D1* is supposed to be mainly the noise components of the original signal. *Ap1* is further decomposed into approximation *Ap2* and detail *D2*, *Ap2*, to *Ap3* and *D3* and *Ap3* to *Ap4* and *D4*. The signal could be reconstructed with the corresponding approximation plus the details. Figure 4.13 shows the signals resulting from the multiresolution procedure shown in Figure 4.12, using Daubechies-6 wavelet.

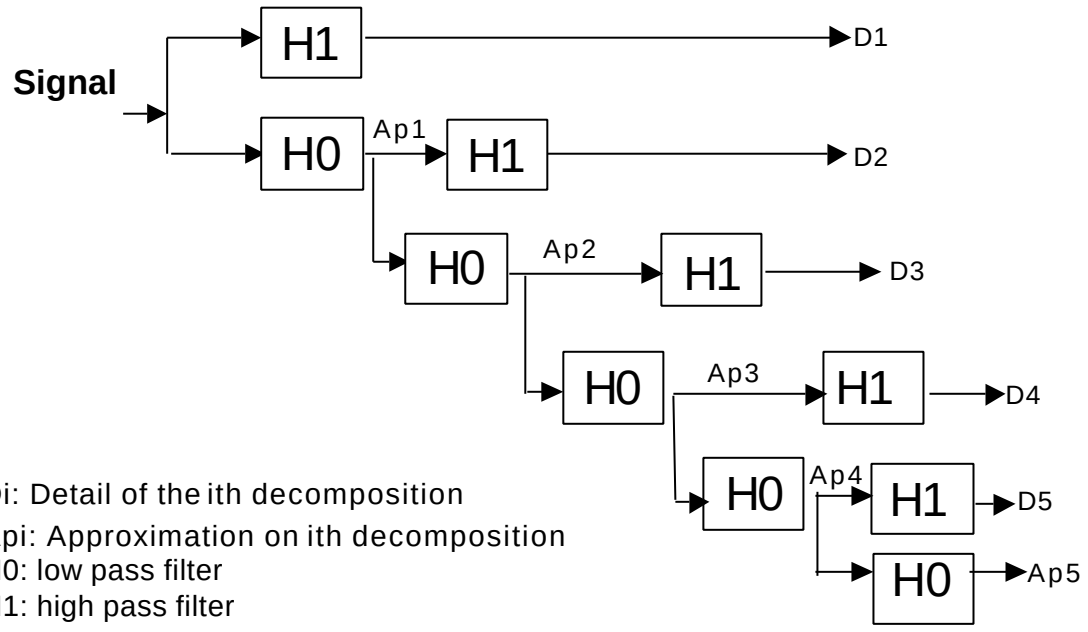

Figure 4.12. Multiresolution analysis

To perform on-line wavelet multiscale filtering a window of dyadic length is used. As new samples are collected the window is translated so that the most recent sample is at dyadic location for at least one translated window.

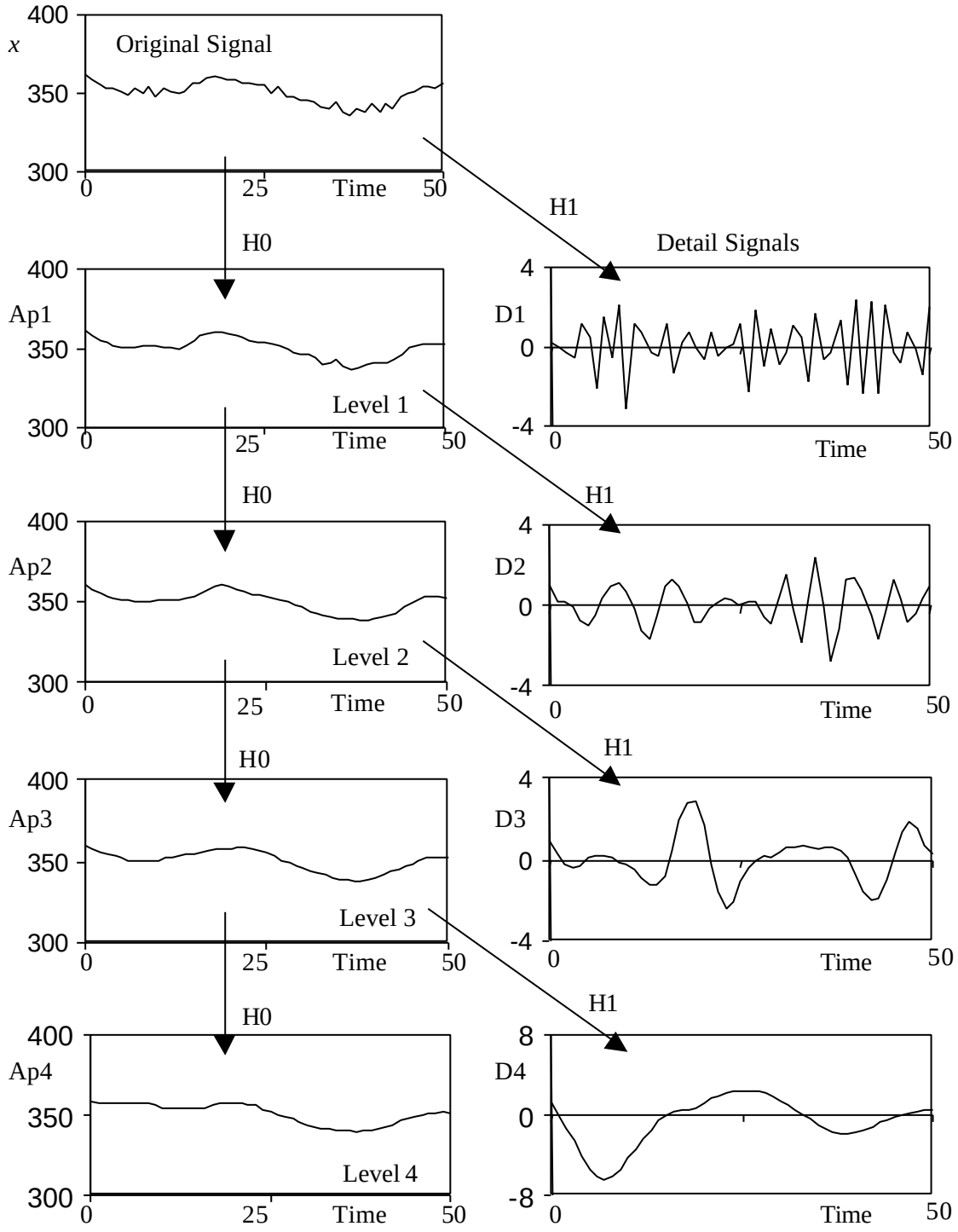

Figure 4.13. Wavelet decomposition using Daubechies-6 wavelet

4.5. Conclusions

Basic concepts of the tools and techniques utilised in the development of the FDS, that will be proposed and explained in the next Chapter 5, have been summarised.

A brief revision of the ANN and FLS technologies has been presented. Detailed treatment can be found in the specialised literature.

HAZOP analysis, a powerful technique to identify potential accidents in the process industry, has been described because it is an important source of information for the development of the FDS.

Finally, wavelet multiscale filtering, an useful signal pre-processing technique for extracting features from process measurements has been briefly explained.

Acronyms

AANN	Autoassociative Artificial Neural Network
ANN	Artificial Neural Network
BPN	Backpropagation Artificial Neural Network
F&EI	Fire and Explosion Index
FDS	Fault Diagnosis System
FL	Fuzzy Logic
FLS	Fuzzy Logic System
FMEA	Failures Modes and Effect Analysis
GDR	Generalized Delta Rule
HAZOP	Hazard and Operability study
MF	Membership function
MHI	Material Hazard Index
MSE	Mean Square Error
NLPCA	Nonlinear Principal Component Analysis
PCA	Principal Component Analysis
PHA	Preliminary Hazard Analysis
PNN	Probabilistic Artificial Neural Network
RB	Rule Based
RBFN	Radial Basis Function Neural Network
SOM	Self Organizing Map
SPE	Squared Prediction Error

Notation

a	ANN output vector
Api	Approximation on the ith decomposition
b	ANN bias vector
db	Number of nodes in the bottleneck layer of an AANN
Di	Detail of the ith wavelet decomposition
dl	Dilation parameter of a wavelet
f	Function
g	Target vector for ANN training
$H0$	Low pass filter
$H1$	High pass filter
l	Index for layers in an ANN
m	Index for nodes in a layer in an ANN
Md	Number of nodes in the demapping layer of an AANN
Ml	Number of nodes in the mapping layer of an AANN
n	Activation status of a node in an ANN
N	Total number of nodes in a layer of an ANN
q	ANN input vector
rl	Index for rules
Rl	Total number of rules in a FLS
S_i	Total number of nodes of the ith layer of an ANN
S_n	Total number of nodes of the output layer of an ANN
tl	Translation parameter of a wavelet
v	Index for training patterns
V	Total number of training patterns for an ANN
W	ANN weight matrix
z'_{ct}	Output of the centroid defuzzification method
z'_{ht}	Output of the height defuzzification method

z'_{mn}	Output of the mean of maximum defuzzification method
z'_{mx}	Output of the maximum defuzzification method

Greek symbols

a	Momentum term in the GDR
d	Error signal in the GDR
h	Learning rate for GDR
m	Membership function
y	Mother wavelet function