Methodology

A key element to manage abnormal situations in a plant is a robust Fault Diagnosis System (FDS). As it can be seen in Figure 5.1, it receives plant measurements and is able to detect a deviation from normal operating conditions and also can determine its root cause. A fast identification of a fault can be utilised by a scheduling system to update the schedule in the most effective way, by the control system in order to take automated control actions and by the operators, as a support for decision-making. The main objective is to avoid plant shutdowns. The plant should continue working satisfactorily in spite of the faults. By this way the productivity can be increased.

In this chapter, the proposed FDS is described and the steps for the successful development are detailed. The algorithms for its on line implementation are also explained. The design takes into account the usefulness of the fault signal. Hence, in the last section, the basis for the translation of the fault signals to be used by other systems is commented.

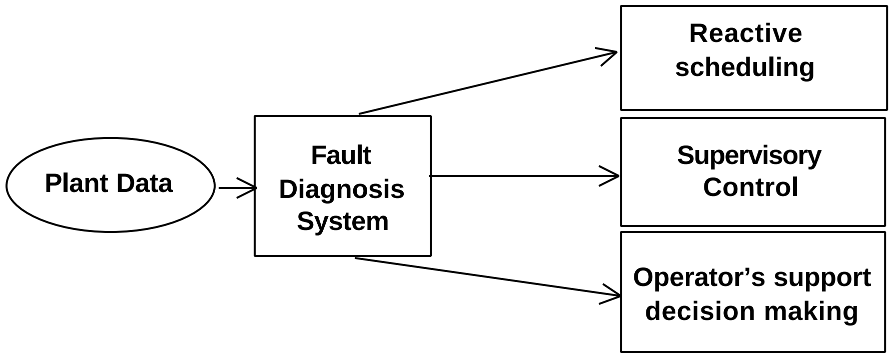


Figure 5.1. Abnormal situation management scheme

5.1. Proposed Fault Diagnosis System

The proposed FDS consists in a combination of a pattern recognition approach and an inference system. With historical data of past faults an ANN is trained to classify them. On the other hand, from a HAZOP analysis, a set of if then rules is defined. This set is complemented with those coming from the experience with the use of the ANN.

An important aspect for the successful implementation of the FDS is the feature extraction to generate the patterns used in the ANN training. The problem of the traditional ANNs related to totally capture the space and time characteristics of process signals is overcome with the use of wavelet functions. Plant signals from the Distributed Control System (DCS) are pre-processed by a multiscale wavelet decomposition, then the extrema of high level of detail are determined and they are the input to the neural classifier. Figure 5.2 shows the information flow in the proposed FDS. The outputs of the ANN are inputs of the Fuzzy Logic inference System (FLS). The fault signal has a value between 0 (no fault) and 1, corresponding to a specified suspected fault.

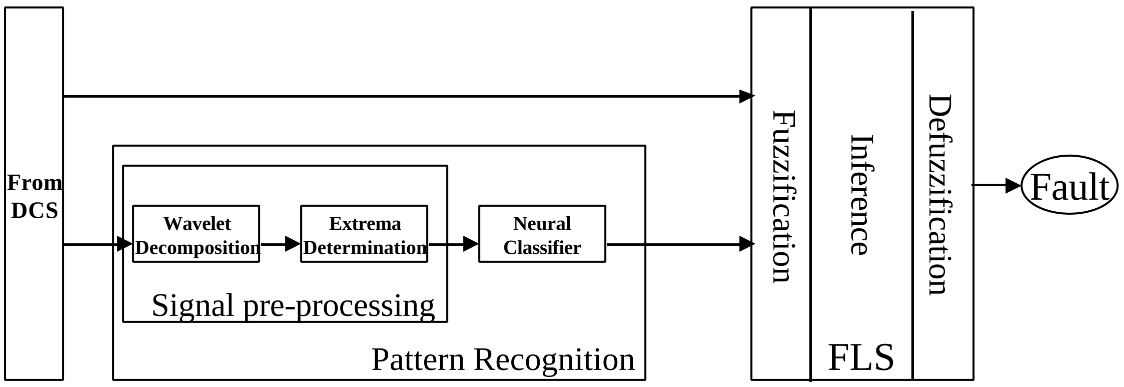


Figure 5.2. Proposed Fault Diagnosis System (Detailed scheme)

5.1.1. The use of a combined approach for the FDS

Generally, automatic fault diagnosis can be viewed as a sequential process involving two steps: the symptom extraction and the actual diagnosis task.

The second step is different from the first in that the output vector is often assumed to be binary. Regarding their form of knowledge acquisition and interpretation, three types of algorithms can be distinguished: classification methods, inference methods and combinations of both.

Classification methods include geometric, statistic, neural and polynomial classifiers and generally use reference patterns for learning. Their structure is not transparent but they can be adapted during use. Inference methods are based on linguistic rules. Most of the time they are given in a fuzzy way. Expert systems fed with fuzzy rules allow a fuzzy decision-making, the so-called "approximate reasoning". The problem with this approach is the long time needed to develop the rules and the difficulties involved in adjusting the rule base.

In order to combine the strengths of both methods, an adaptive neuro-fuzzy system has been developed. The idea is to obtain an adaptive learning diagnostic system with transparent knowledge representation.

5.1.2. General structure of the proposed Neuro-fuzzy FDS

It is considered an ANN - based supplement of a fuzzy logic system (FLS) in a block-oriented configuration (Ruiz et al., 1999b). Figure 5.3 shows a general scheme of this approach.

M1 is the subset of the direct and indirect measurements and/or observations from the plant, and is selected as input of the ANN approach.

N1 is the set of *n1* "pre-faults" diagnosed by the ANN approach. The values *N1*(*i*), *i* from 1 to *n1*, are usually between 0 and 1. They are the input of the FLS.

*M*2 is a set of *m2* direct and indirect measurements and/or observations from the plant, which is selected as input of the FLS.

When using fuzzy logic in a diagnostic environment, the following successive steps are involved: fuzzification of "crisp" values; inference using a rule base in which the logical operations are performed on the membership functions; and defuzzification to obtain "crisp" outputs.

The inference engine has the knowledge base, expressed in a set of if-then rules. These rules are of two types: those containing process deep knowledge and those that are built from experience of the ANN's performance. In Table 5.1, a scheme of the set of rules is presented. The outputs *F* (nf), *j* from 1 to *Nf*, are the *Nf* faults considered.

Detailed aspects of the ANN implementation and the FLS development are described in detail in subchapters 5.4 and 5.5, respectively.

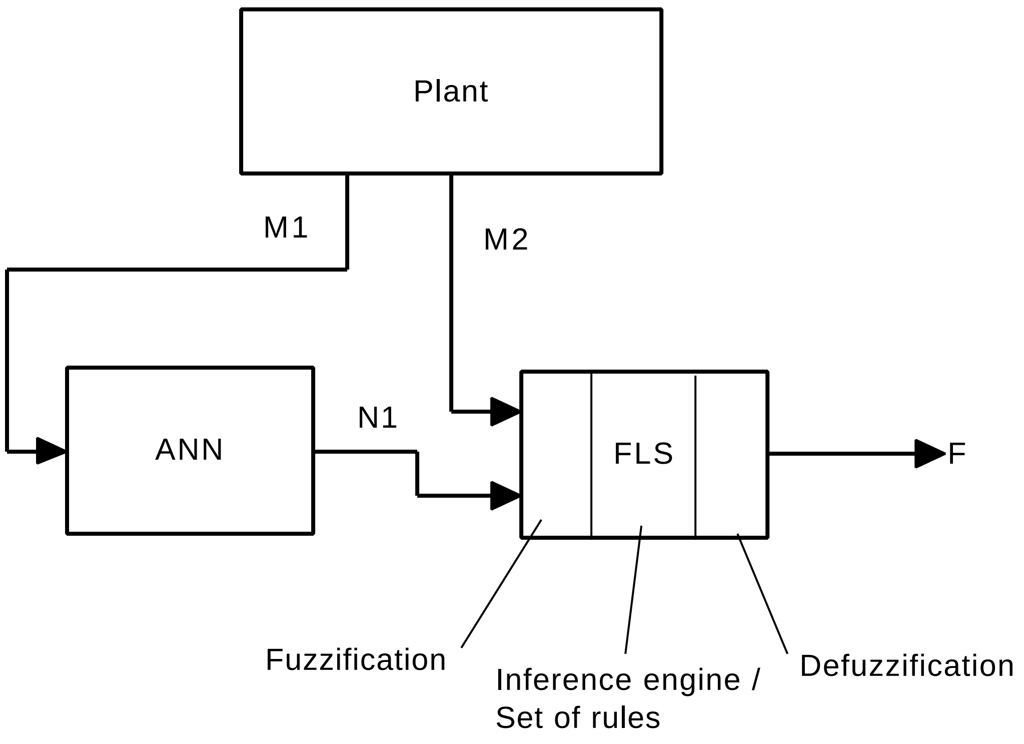


Figure 5.3. Proposed Fault Diagnosis System (General scheme)

Table 5.1. Scheme of the set of rules

Based on experience with ANN performance	IF *N*1(1) is ... AND (*M2*)... THEN *F*(*nf*) is...
	...
	IF *N*1(i) is ... AND (*M2*) ...THEN *F*(*nf*) is...
	...
	IF *N*1(n1) is ... AND (*M2*)... THEN *F*(*nf*) is...

Based on process deep knowledge	IF *M*2(1) is ... AND...THEN *F*(*nf*) is...
	...
	IF *M*2(i) is ... AND ...THEN *F*(*nf*) is...
	...
	IF *M*2(m2) is ... AND... THEN *F*(*nf*) is...

5.2. Step by step methodology

The information needed to implement the FDS includes a historical database, a Hazard and Operability (HAZOP) analysis and a model of the chemical plant. These three sources of information are commented in the following paragraphs. Then, the steps of the proposed methodology are described.

Historical database

Process computers now collect masses of data from a multitude of plant sensors every few minutes or seconds. Present chemical plants do not take advantage of this powerful source of information. A number of applications in process modelling, monitoring and control can be actually performed.

The historical database that includes information related with normal and abnormal operating conditions can be used to train the ANN structure. An analysis of historical data allows the identification of outliers. Additional uses correspond to the FLS tuning and the test of the FDS performance.

HAZOP

The HAZOP analysis has two important utilities. It allows to generate the if-then rules for the Knowledge Base and to determine the information to be sent to other levels in the information system. Implementation of an operator support system by extending the HAZOP method to fault diagnosis characterisations implies a direct and effective solution. The generated rules are kept simple to avoid the general problems of large rule-based knowledge systems, such as contradictory rules, large amounts of irrelevant information and complex tree structures. HAZOP analysis also can help in the definition of the set of faults.

Plant model

At different design levels of a plant a model is used. The mathematical model can be made from experimental data using identification techniques or from energy and mass balances describing the plant dynamics, or a combination of both. The plant model has to be validated against available plant data. Nowadays, the use of a commercial process simulator allows to develop plant models easily. This practice is successful mainly in the petrochemical industry. Process engineers handle different kinds of plant models and the model has not to be completely accurate. It has to be accurate enough in order to have a satisfactory performance in the system that uses it.

A model of the plant can be used to obtain plant operation experience through simulation. The simulation can provide data on infrequent faults because in the cases of faults that rarely occur it is not possible to test the FDS using only plant data.

In addition, by testing the ANN with the model, the development of the rules based on experience with ANN performance is straightforward. The model is also useful for testing and validating the FDS.

If a plant model is not available it can be developed using the ways already commented (process simulators, identification techniques, etc). If the process complexity or the lack of information makes this step impossible to perform, the FDS can be developed as well but in a limited form. An example of such situation is shown in one of the industrial applications (CAICC sugar cane refinery, Chapter 8).

Steps

The methodology can be summarised in the following steps. A simplified flowsheet is also shown in Figure 5.4 (Ruiz et al., 2001a).

1) Define the faults. Three kinds of faults can be considered: faults in the process (e.g., a failure in a pump), in the controllers and in sensors. Faults whose detection can not be justified in terms of economic impact are not to be considered (e.g., very infrequent faults). With a HAZOP analysis, the determination of possible faults is straightforward. The column Causes in such analysis corresponds to the set of possible faults. A second review should eliminate the faults that are very infrequent, or not important from the economic point of view or that are easily diagnosed by the conventional alarm systems. The use of historical data can help in the definition of the set of faults. Statistical techniques like PCA can be used to determine outliers, but an important point is to have available good information from plant operators in order to relate the found outlier with a reported fault (occurred in the past). Figure 5.5 shows the algorithm for a systematic definition of the set of faults. The main sources of information are the historical data and the HAZOP analysis. After analysing the historical data, the identified abnormal deviations, not included in the HAZOP analysis, are added for their consideration. From HAZOP analysis, each possible cause is analysed by considering the frequency and the economic impact. This systematic treatment of the information allows to define a minimum set of faults. This combined criterion for such definition results advantageous with respect to other alternatives commonly used which only consider one source of information. For example, some important faults can be missed if the historical data are not taken into account.

2) Determine measurements. At the plant design stage, measurements can be selected on the basis of fault diagnosis methods. In most industrial applications the FDS is developed using available measurements and sensors that are usually installed because of their utility for control. It is often necessary to include additional sensors. This topic is an area of research called optimal sensor location. In this thesis, the development of the FDS is presented using available measurements, as is typical in most industrial applications.

3) Obtain the fault patterns: The objective of this step is to have a fault pattern associated to each defined fault. In the cases of defined faults that have occurred in the past, the corresponding profiles of all variables (from historical database) are saved. In the cases of defined faults that have not occurred in the past or no historical data are available, the profiles of all measured variables, obtained by simulation, are saved. The saved information can be used directly in the following step. However, better results can be obtained if the fault patterns are composed by the features of process signals. Feature extraction is performed by pre-processing the signals by a multilevel wavelet using a specific filter. As the noise component are reduced and then disappeared as the scale increases, a detail of high scale is chosen. Then, the extrema of the processed signal are determined (see section 5.4.2). Hence, the fault pattern is composed of a set of features from a set of variables measured from the plant. The generation of the fault patterns can be automated by using the algorithm shown in Figure 5.6. For each defined fault, a matrix is built with the profiles of all measured variables. The corresponding data are obtained from the historical database if the fault occurred in the past and the corresponding data are available. Otherwise, the fault is simulated. A second matrix can be built for each fault with the features of each variable which are obtained by multiscale wavelet decomposition. This automatic generation of fault patterns is useful for the next step.

4) Train an ANN. The first option is to use directly the profiles of the variables saved in the previous step. In the case of continuous processes, using the steady state data an ANN is trained. In the case of batch plants, the measurements from the plant and the time from the start of the batch are the ANN's inputs. The ANN's outputs are the signals of the suspected faults. Among the different kinds of ANNs, multilayer perceptron and radial basis function networks have been used in FDSs in the chemical process industry. The ANN approach is then tested by simulating the faults. This has to be done in “on-line” mode using the simulator or in “off-line” mode in the cases of available experimental data. The outputs of the ANN (its profiles vs. time) are useful for the next step. The second option is to train an ANN with the fault patterns composed by the extracted features of the different signals. ANN implementation in the two described options is treated in detail in section 5.4.

5) Design the FLS: Inputs to the FLS and the set of if-then rules are defined from plant experience, operator’s information, HAZOP analysis, operating experience by simulation and the results of the previous step. Outputs from the ANN are also to be considered as FLS's inputs. The tuning of the membership functions of inputs and outputs requires several simulations of the FDS to adjust them properly. The system is designed with the premise that no diagnosis is better than false diagnosis. FLS development is detailed in section 5.5.

6) Test the new system by simulation. The test takes into account the speed of detection and diagnosis, and exactness. It can be compared with the results obtained with the FLS (without the ANN input) or the ANN working alone. This allows to appreciate the advantages of the combination.

7) Design the adaptive method. Taking advantage of the learning ability of the ANNs, the outputs of the FDS are saved. Then, the ANN is retrained perodically, depending on the cases. Changes to processes are normal in the chemical industry and the system must be able to adapt to these changes easily. This requires good organization of the knowledge base and ability to edit the knowledge base with ease. Having two additional outputs of the FDS, one related to normal operation and the other one related to a new suspected fault can help in the success of the adaptive method implementation. By this way, the FDS can have the attribute of novel identifiability.

8) Test with the model. The performance of the FDS has to be checked by simulating the faults. The use of the system with operators is necessary to train them with the new system.

9) Implementation in the real plant. The final step is the integration of the FDS with the existing software and hardware.

An additional step is the design of the correction of the faults in order that the process continue functioning satisfactorily while the problem is being solved. This aspect is related to the usefulness of the information given by the FDS (Figure 5.1.) In this thesis, the basis of the use of the information provided by the FDS is settled. An explanation with an example is shown in section 5.7.

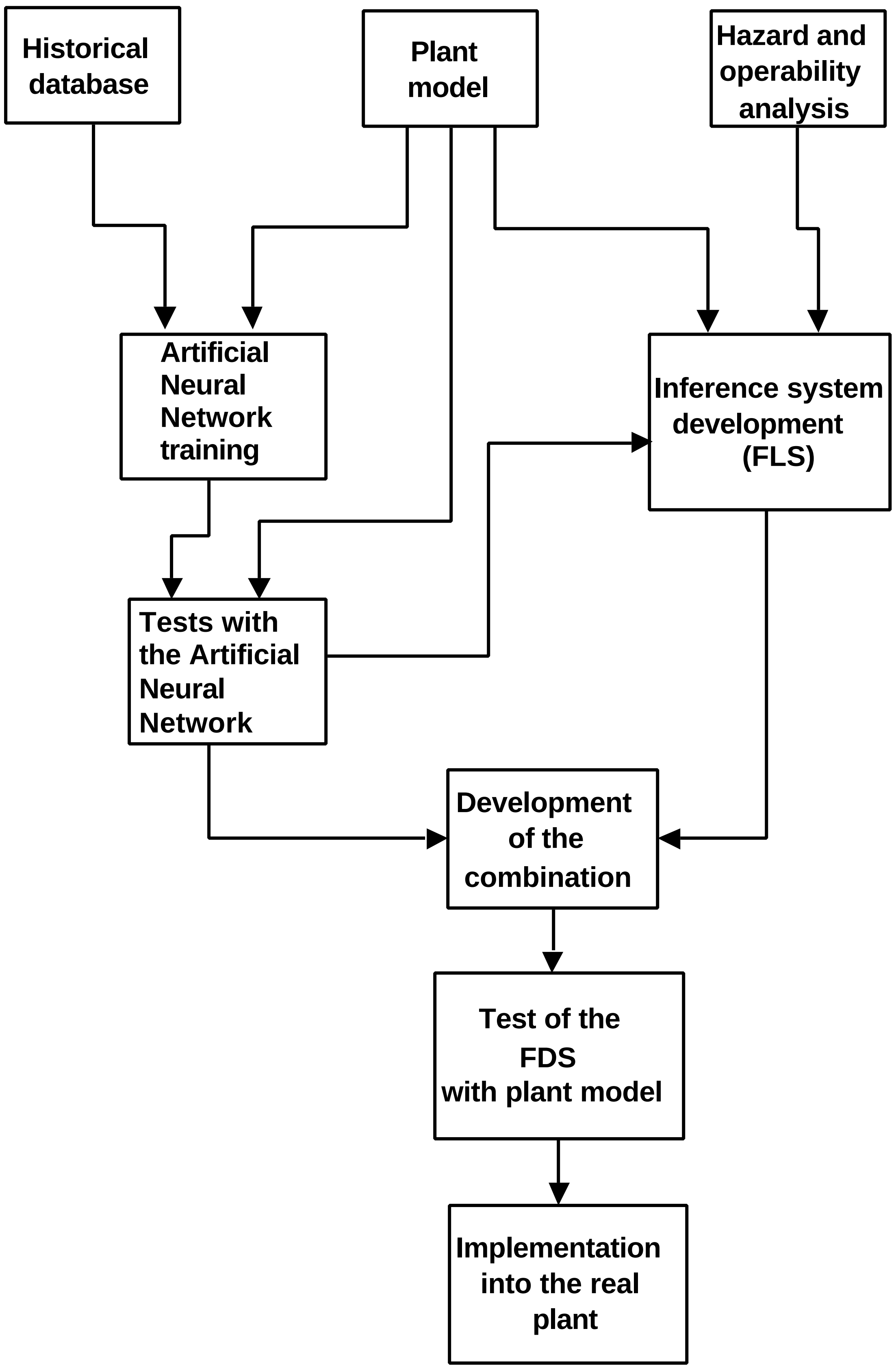

Historical database
Plant model
Hazard and operability analysis
Artificial Neural Network training
Inference system development (FLS)
Tests with the Artificial Neural Network
Development of the combination
Test of the FDS with plant model
Implementation into the real plant

Figure 5.4. Flowsheet of the proposed methodology to design the FDS.

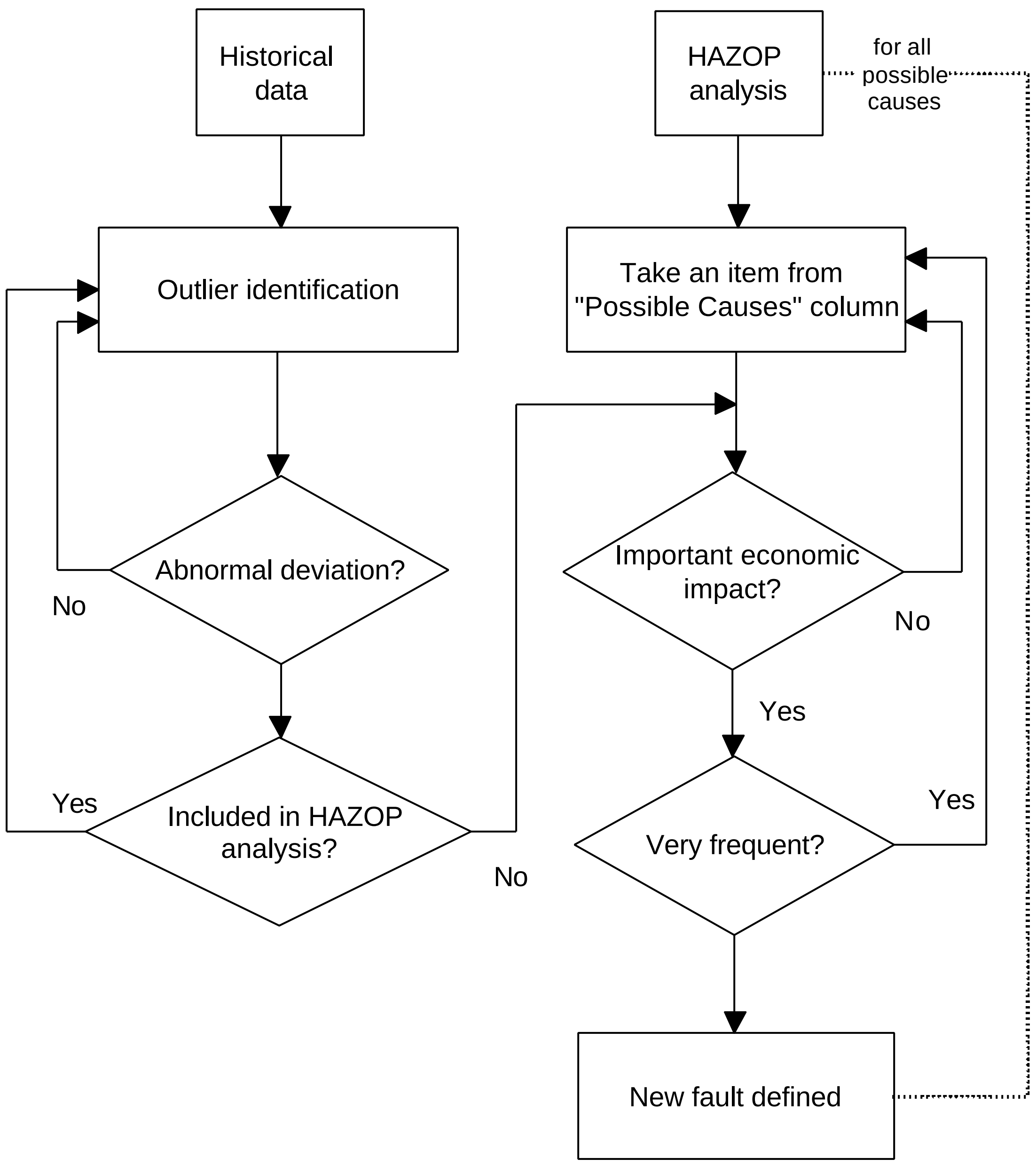


Figure 5.5. Systematic definition of the set of faults

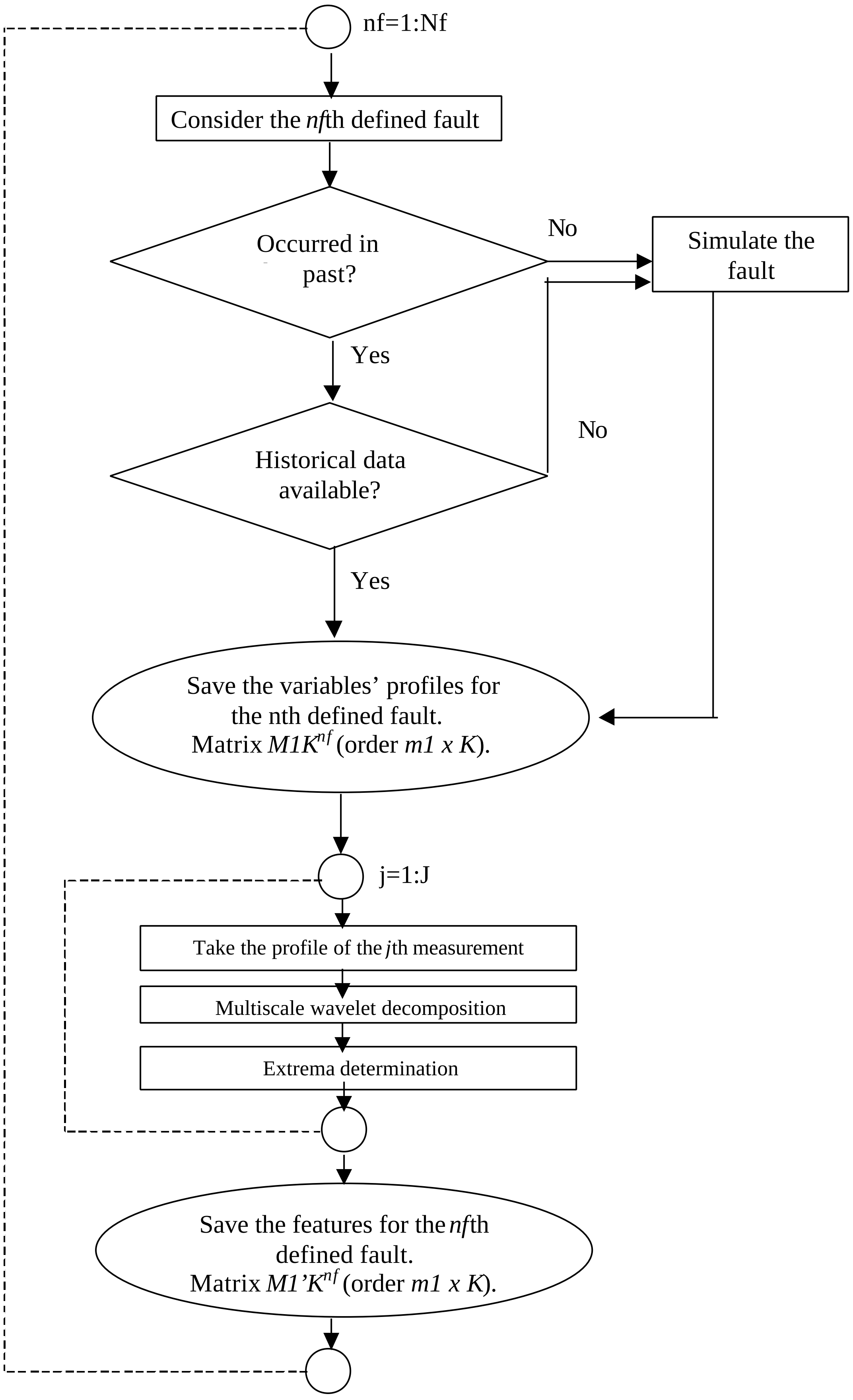

nf=1:Nf
Consider the *nf*th defined fault
Occurred in past?
No
Simulate the fault
Yes
Historical data available?
No
Yes
Save the variables' profiles for the nth defined fault. Matrix $M1K^{nf}$ (order $m1 \times K$).
j=1:J
Take the profile of the *j*th measurement
Multiscale wavelet decomposition
Extrema determination
Save the features for the *nf*th defined fault. Matrix $M1'K^{nf}$ (order $m1 \times K$).

Figure 5.6. Generation of fault patterns

5.3. Performance Index

A FDS has to be robust enough to correctly diagnose a fault. The diagnosis should be independent of the magnitude, time duration and direction of the fault. Two important aspects are the time needed to correctly identify a fault and the no existence of false diagnosis.

A performance index is necessary to compare different FDSs. The first attempts to introduce a performance index correspond to the evaluation of the ANN performance for fault diagnosis in steady state processes. In the following paragraph the criteria for the performance index development is explained.

The ANN is trained with steady-state conditions of a fault. Therefore, when the fault occurs, the ANN will correctly diagnose when the plant arrives to the new steady-state. However, in some processes, for example plant with recycle streams, the time needed to arrive to the new steady state can be long. During that time the ANN can give no diagnosis, right diagnosis or false diagnosis. In order to compare the performance of the different ANN approaches for fault diagnosis the following performance parameter *(%P)* defined by Equation (5.1) has been proposed (Ruiz et al., 1999d):

$$\%P = \left(\frac{t_{ss} - t_d}{t_{ss}} \right) \bullet 100 \qquad (5.1)$$

which gives the percent relation between the time taken from the correct diagnosis to the new steady-state condition (t_{ss} - t_d) and the total time of the dynamic state of the plant from the moment of occurrence of the fault (t_{ss}) (see Figure 5.7).

On the other hand, false diagnosis has to be avoided. For comparison purposes it will be used the percent relation between the number of cases with false diagnosis and the total number of faults simulated. In order to take into account the false diagnosis and the low resolution, *%P** is considered. It is calculated as in Equation (5.1) but in cases of false diagnosis or low resolution, t_d is fixed equal to t_{ss}.

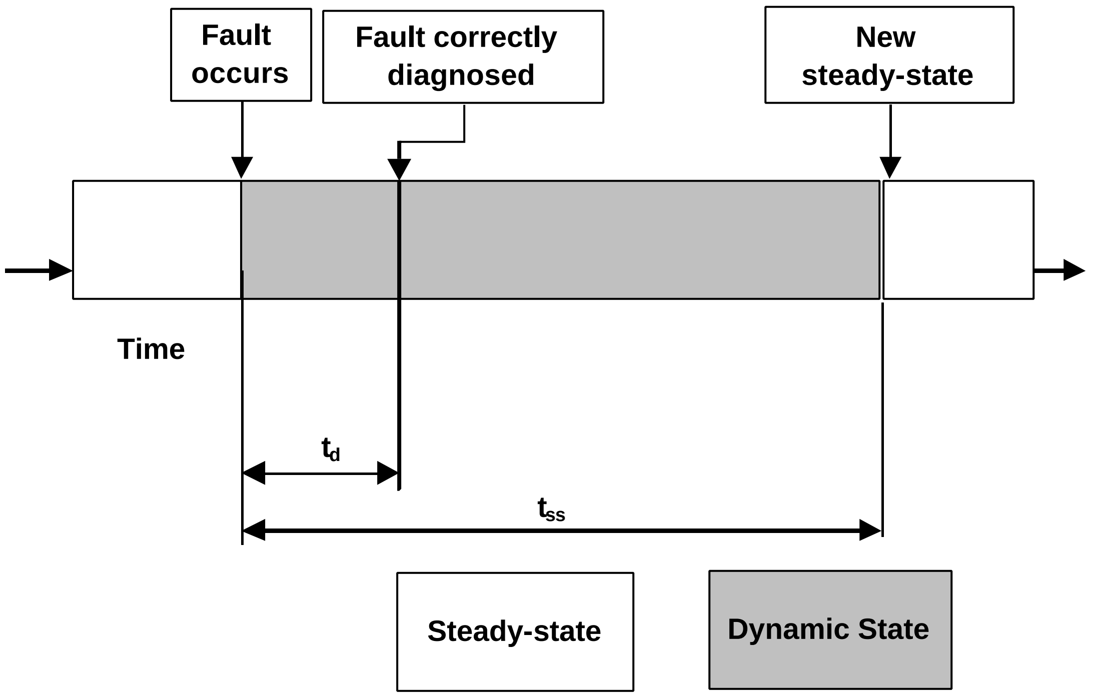


Figure 5.7. Definition of parameters used in the evaluation of the performance of the fault diagnosis system (steady-state processes)

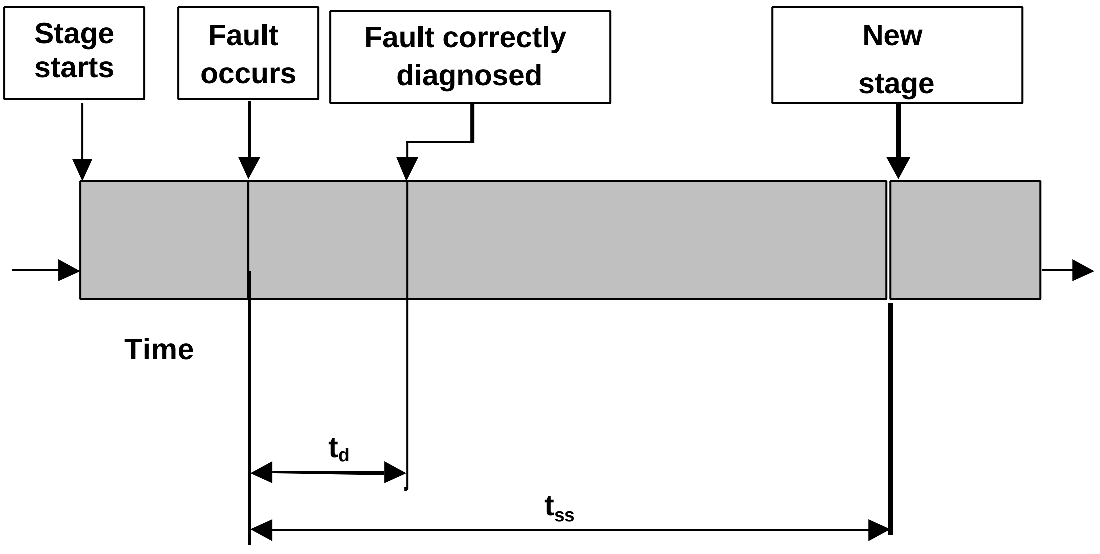


Figure 5.8. Definition of parameters used in the evaluation of the performance of the fault diagnosis system (batch processes)

In order to use the same performance index when dealing with batch processes, the following reasoning has been made. Each stage of the batch process is considered separately. As it can be seen in Figure 5.8, t_d is the time elapsed since the fault occurs until the fault is diagnosed correctly. The parameter t_{ss} is the time needed to finish the current stage since the fault occurs. By this way the same presented equations can be used for *%P* and *%P*.*

The use of the presented performance index *%P* and the *%P** is illustrated in Chapter 7 (Results and Discussion. Plant with recycle and batch reactor cases studies).

5.4. Artificial Neural Network implementation

ANN implementation can be performed in two main different ways: using the saved profiles of the variables of the defined faults directly or by using the corresponding features. This last option allows better results and it is included in the proposed FDS (Figure 5.2).

5.4.1. ANN training using the profiles of variables directly

In the cases of continuous processes, the ANN training inputs are the saved steady state measured variables for the defined faults and the target outputs are the suspected faults with a value of 0 or 1 (No Fault or Fault). In some situations of process faults, as for example "low feed flow-rate", it is necessary to define which percentage of variation is considered unpermitted, That is, which condition will be considered as fault (output = 1). The inputs are scaled down by decreasing in average and dividing by its standard deviation.

In a similar way, the ANN training is performed for the cases of batch processes. The only difference is that the ANN training input includes the time from the start of the step (Ruiz et al., 1999c). Furthermore, each stage has associated an ANN, with the corresponding defined faults.

An improvement to the explained approach consists in the use of a moving window. The inputs corresponding to each variable are the current one plus some previous values. In the case of batch processes the width of the moving window should be the length of the stage and the time from the start of the stage is not considered.

All the described alternatives are summarised in Figure 5.9. Their implementations are illustrated in different case studies (Chapter 7 and 8).

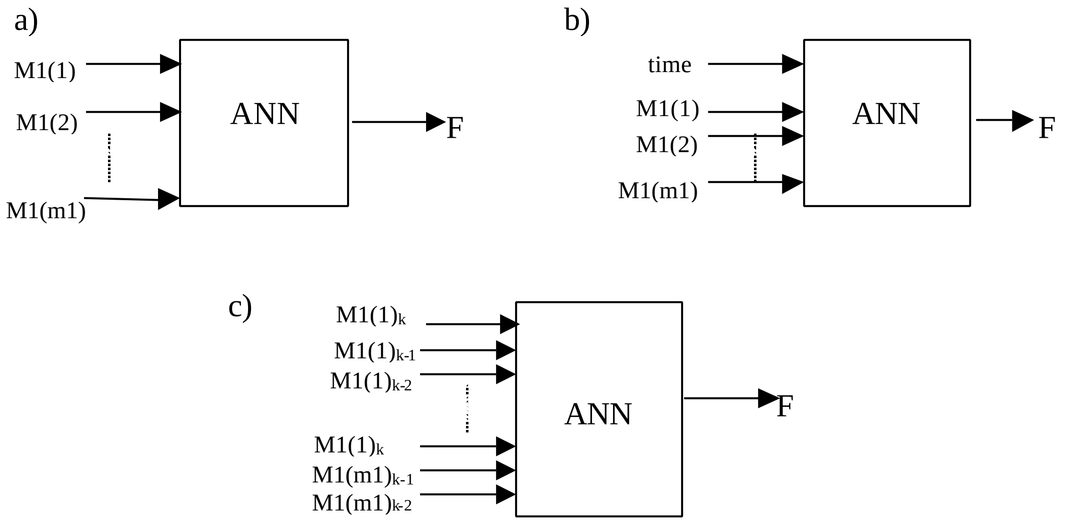


Figure 5.9. ANN implementation using directly the measured variables a)Continuous processes; b) Batch processes; c) With a moving window for complex dynamics

Figure 5.10 shows the algorithm proposed to develop an ANN using directly the measured variables in the case of a continous plant. A loop, considering all the prefaullts is used. For the variables' profiles of each fault, the time interval *k* in which the plant arrives to a new steady state is determined (*knf*). Equation (5.2) is used to determine such condition. *M1K^i* corresponds to the matrix M1K^{nf} (*nf*=*i*, in the loop) determined in generation of fault patterns approach (Figure 5.6), k is the time interval and ε is a small number.

$$\sum_{j=1}^{J} \frac{\partial M1K^{i}(j,k)}{\partial k} \leq e \qquad (5.2)$$

Once the new steady state condition is determined, the column vector *M1K^{nf}(knf)* is the ANN training input (*q*) and the corresponding target (*g*) is set with a value of 1 for the corresponding fault (and zero in the rest of the vector).

Depending on the size of the training data set, a BPN or a RBFN can be recommended. The RBFN trains faster and has better performance than a BPN. However, with many fault patterns, the obtained RBFN can be very big (a large number of centres, nodes in the hidden layer). This aspect implies high computational requirements. Therefore, a BPN is recommended for such cases.

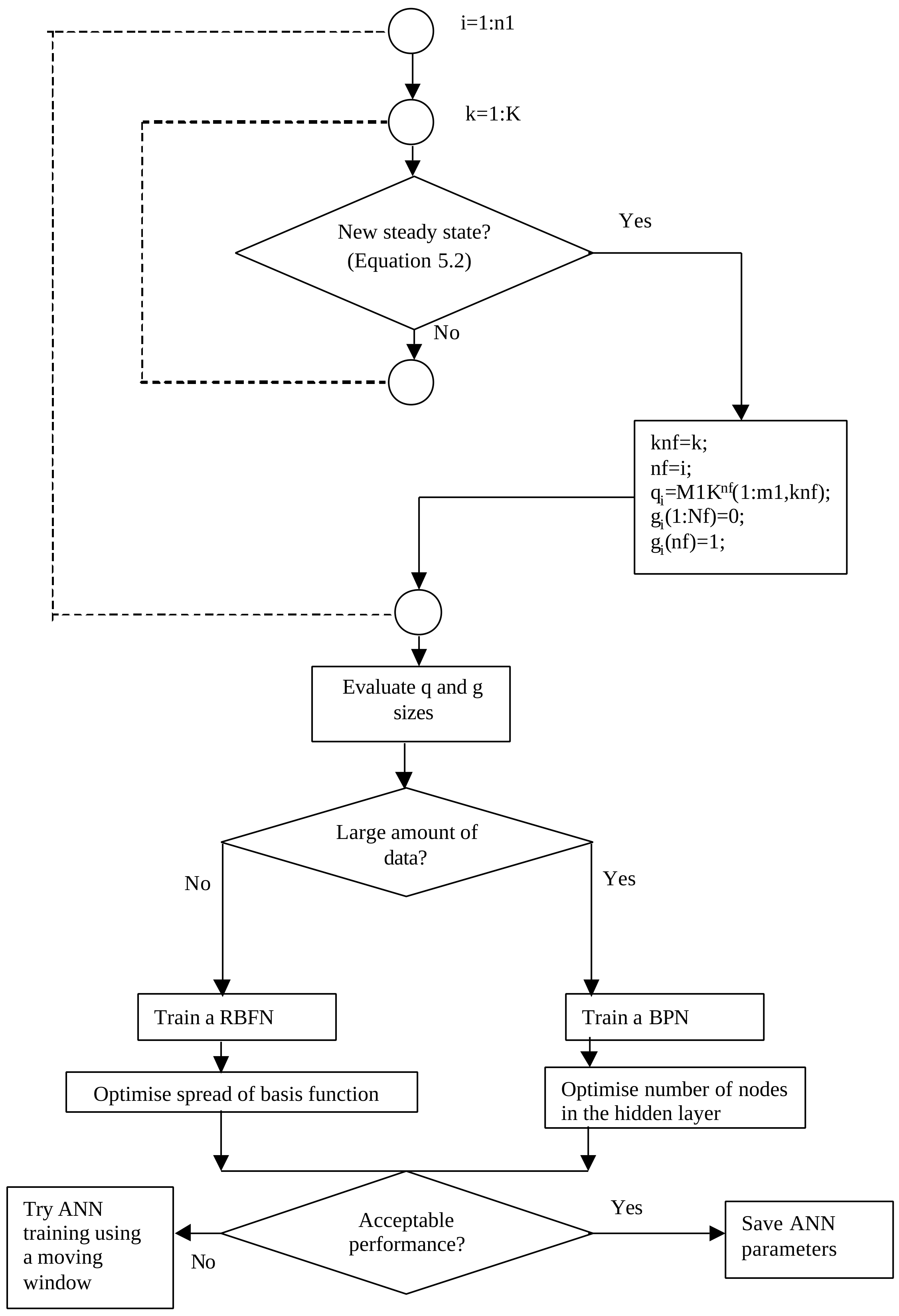

i=1:n1
k=1:K
New steady state? (Equation 5.2)
Yes
No
knf=k;
nf=i;
q_i=M1K^{nf}(1:m1,knf);
g_i(1:Nf)=0;
g_i(nf)=1;
Evaluate q and g sizes
Large amount of data?
No
Yes
Train a RBFN
Train a BPN
Optimise spread of basis function
Optimise number of nodes in the hidden layer
Acceptable performance?
Yes
No
Try ANN training using a moving window
Save ANN parameters

Figure 5.10. ANN training using the measurements from the plant directly (continuous plant)

The optimisation of the ANN is performed by testing the performance with Equation (5.1) in a trial and error approach. In the case of the RBFN the spread of the radial basis function is optimised. On the other hand, the number of nodes in the hidden layer is optimised in the case of the BPN.

If the performance is not acceptable, ANN training using a moving window has to be done. Otherwise, the ANN training using the fault patterns composed by the signal features has to be done. This alternative is explained in the following section 5.4.2.

5.4.2. ANN training using the fault patterns composed by signals features

First, feature extraction will be explained. The signal preprocessing starts with a multiresolution procedure as has been described in section 4.4 (Figures 4.12 and 4.13). A multilevel wavelet using a specific wavelet filter, in that case the Daubechies-6 filter, has been performed. Observing the detail signals (D1, D2, D3 and D4), the noise components are disappearing as the scale increases. Considering the detail of high scale (for example D5), the extrema are then determined. The algorithm of this extrema determination is shown in Figure 5.11.

```
function M1’(i)=ExtremaDetermination(D5(i),D5’(i),D5’’(i))
if D5'(i)=0 then
    if D5''(i)><0 then
        M1’(i)=D5;
    end
end
```

Figure 5.11. Extrema determination

In such algorithm, *D5'(i)* and *D5''(i)* are the first and second derivatives of *D5(i)* -the detail of the *i*th measurement- respectively, and *M1’(i)* the corresponding determined extrema.

By this way the maxims and minima of the detail are obtained. This extrema are the features extracted from the original process measurement. The following step is to train a ANN classifier (Ruiz et al., 2000b). Figure 5.12 shows an example of Extrema determination. In the case of batch processes, the preprocessing is performed onto the difference between the measurement and a reference profile. The reference profile corresponds to normal operation. By this way, better results have been obtained.

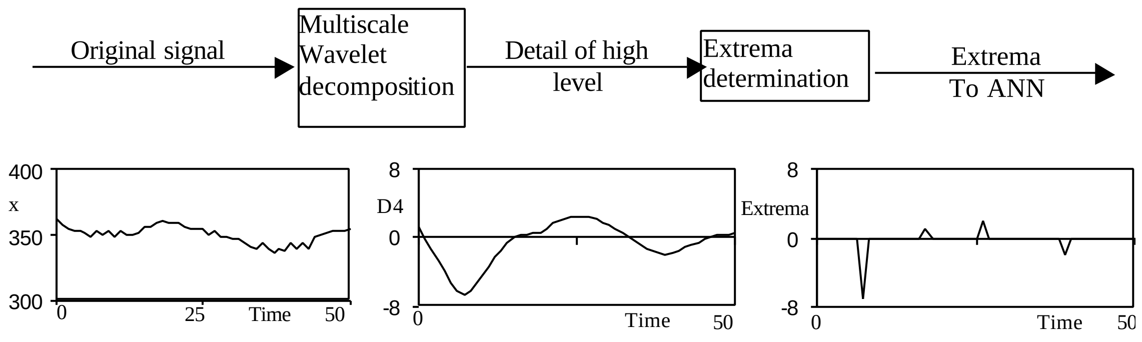


Figure 5.12. Signal preprocessing to generate the fault patterns used in the ANN training

An important aspect is the adequate selection of the variables that are to be considered as the inputs (its extrema) of the ANN. Considering a defined fault, the resulting extrema of all the measured variables are evaluated. Then, the only measured variables that have different features (with respect to normal operation) are selected as training inputs for the corresponding "class" fault.

Some process variables can have no extrema for any defined fault. These variables are not to be considered as inputs.

A PNN is suitable for the classification of these fault patterns composed by signal features.

5.4.3. Fault in sensors diagnosis using Autoassociative Artificial Neural Networks

The special case of the diagnosis of fault in sensors is solved with the use of Autoassociative Artificial Neural Networks (AANNs). This complement of the proposed FDS is considered in this section because of its strong relationship with ANN implementation.

Almost all Distributed Control Systems (DCSs) have standard sensor fault detection features. By this way, sensor failures such as signal loss can be easily detected.

Currently, the main difficult is to capture if the sensor reading is the actual value or if there is a constant bias, or drift.

Regarding process monitoring, principal component analysis (PCA) is a way of reducing the dimension of the space of process variables by linearly mapping the process variables to a smaller dimension space of “scores”. Unusual process behavior can be detected in some ways. One of them is based on the reconstruction of the process vector from the score vector. Therefore, a comparison of the reconstructed vector with the original vector, using the square prediction error (SPE), permits the detection of abnormalities.

If the process is highly nonlinear, as is the case of chemical processes, the nonlinear principal component analysis (NLPCA) is preferred. It uses an AANN consisting in feed-forward nets trained to produce an approximation of the identity mapping between inputs and outputs. The residuals of this mapping can be used to detect sensor failures.

In this work, an AANN (section 4.1.2) is used to diagnose bias errors in sensors.

An AANN is trained using historical data with correct sensor measurements. Input vector is the measurements’ vector and the output target is the same vector.

After the ANN training, *SPE* is determined by Equation (4.8) as well as its standard deviation (σ) of the square differences. Considering 99% control limits, the upper limit SPE_{sup} is calculated by Equation (5.3).

$$SPE_{\text{sup}} = SPE + 3s \tag{5.3}$$

The *SPE* and the σ of each sensor (SPE_j and s_j respectively) are also determined. Considering 99% control limits, the upper limit SPE_{js} is calculated by Equation (5.4).

$$SPE_{js} = SPE_j + 3s_j \tag{5.4}$$

During on-line monitoring the *SPE* is calculated with the new measurements, and it is compared with the SPE_{sup} . If this limit is overcome then a faulty sensor exists. To identify and isolate the fault in one or more sensors, the algorithm shown in Figure 5.13

has been proposed (Ruiz et al., 1999d). *SPE_j* is calculated for every sensor and its percentage respect to the respective *SPE_{js}* is calculated, too, by Equation (5.5). The sensor that has the higher *$SPE_{js}\%$* is considered as the faulty sensor (*j**). In order to determine if the other sensors are also faulty, the range *R* of $\text{SPE}_{\text{js}}\%$ is determined by Equation (5.6), being *$SPE_{js}\%higher$*, the maximum *SPE_{js}* considering all the *J* sensors and *$SPE_{js}\%lower$* the minimum *$SPE_{js}\%$.* All the sensors which have the corresponding *$SPE_{js}\%$* value above the middle of the range are considered faulty sensors, too.

$$SPE_{js}\% = \frac{SPE_j}{SPE_{js}} \bullet 100 \qquad (5.5)$$

$$R = SPE_{js}\%higher - SPE_{js}\%lower \qquad (5.6)$$

The advantages of this algorithm can be viewed in the following example. Figure 5.14a shows the profile of all the *SPE_j* when a bias error in sensor *S1* has occurred at time 100. Almost all of the *SPE_j* overcome their respective *SPE_{js}* . This problem happens due to the extrapolations problems of backpropagation networks. Figure 5.14b shows the *SPE_j* values as a percentage of their respective *SPE_{js}* values, Equation (5.4). The difference between sensor S1 and the others is clear. In this case the calculated value for this sensor is the only one that overcomes the fixed limit. Hence, the faulty sensor has been successfully isolated.

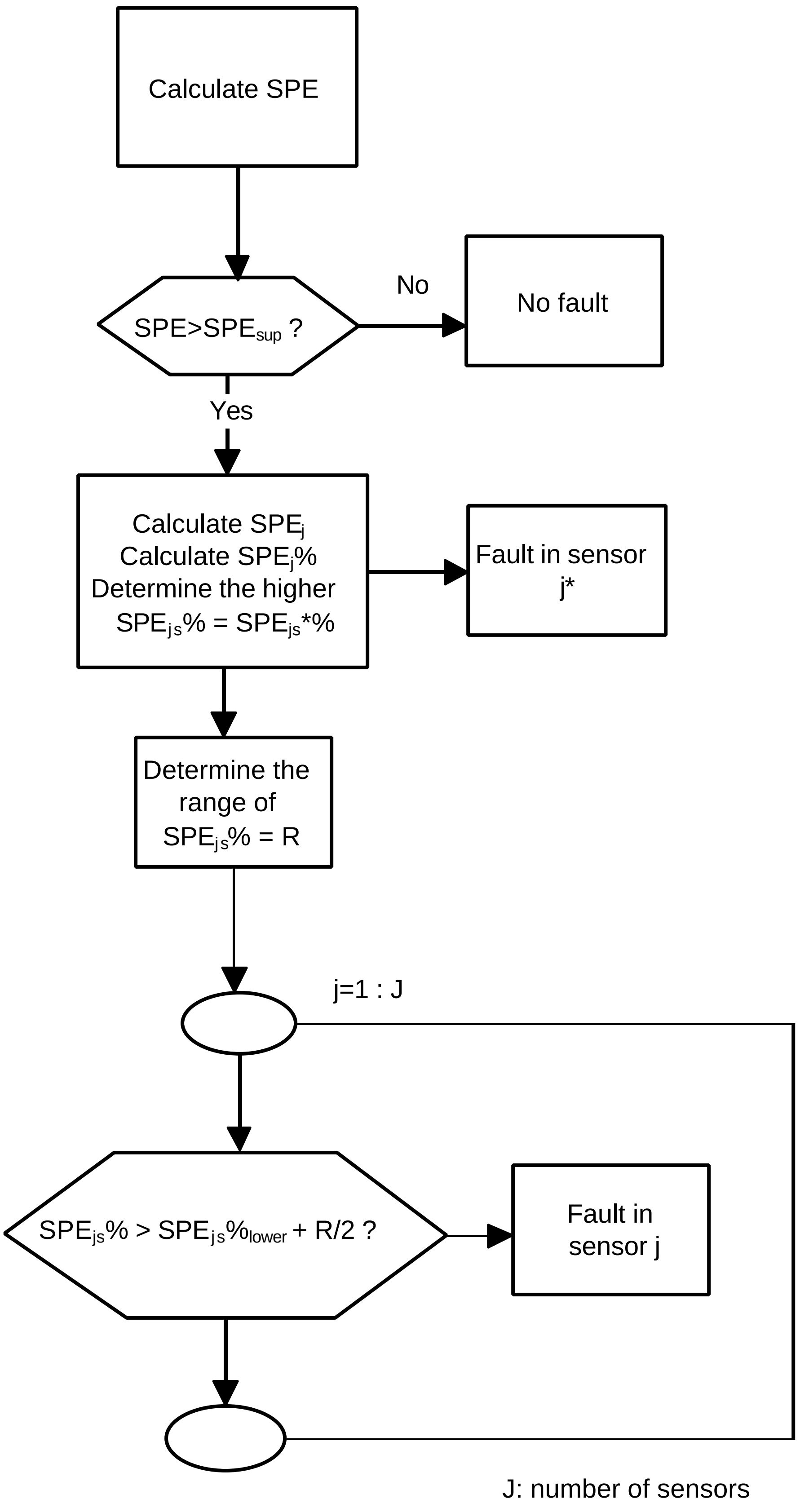
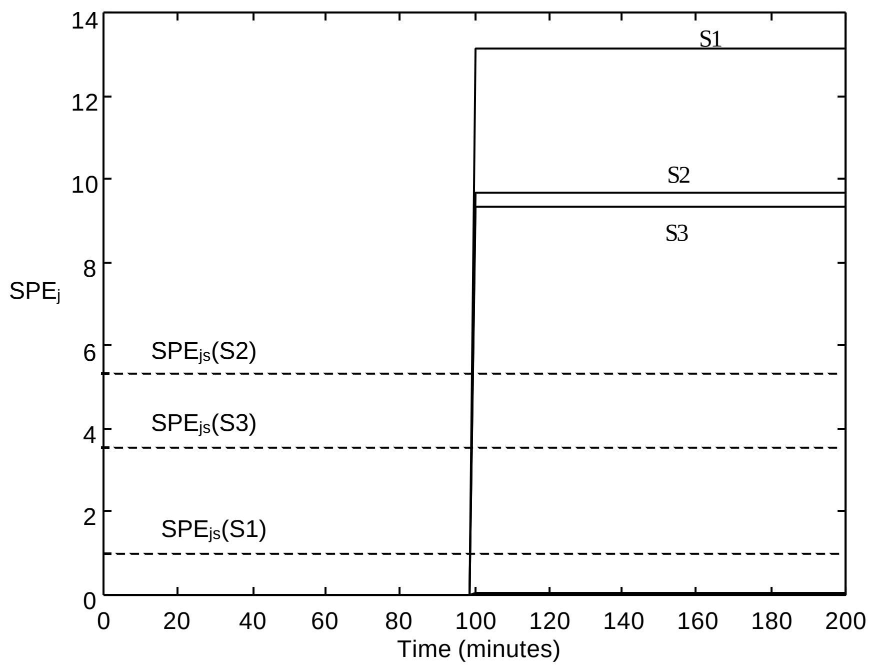


Figure 5.13. Proposed algorithm to isolate the faulty sensor from the Auto-associative Artificial Neural Network's response

a)



b)

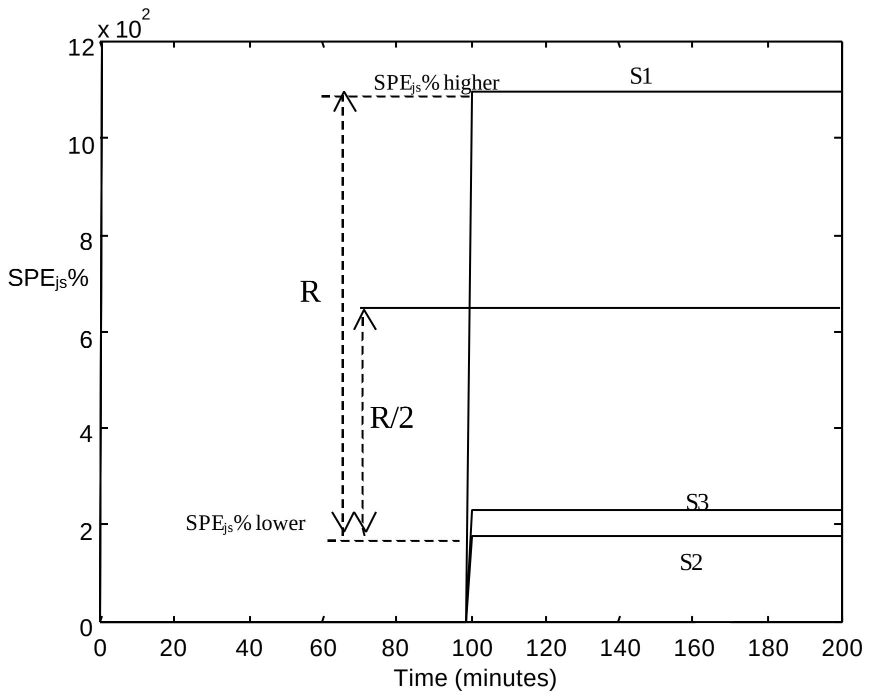


Figure 5.14. Fault in sensor 1 at time 100. a) SPE_j of each sensor; b) $SPE_{js}\%$ of each sensor

5.4.4. Recommendations for successful ANN implementation

Beginners in the use of ANN usually have problems in relation to a successful implementation of this technology. The following recommendations (some of them based on Freeman, 1999) put in practice in the present thesis, should be taken into account.

First of all, ANN never has to be trained without a previous conscientious analysis of data. It is necessary to be sure that are enough data for training. Extensive data analysis can be performed using data representation techniques (statistics, for example). In some very simple cases, a least square data fit performed as well as an ANN. In such cases (it is not the case of FD for complex processes) an ANN is not needed.

A typical problem during ANN training is the overfitting or overtraining. If an ANN is overtrained, it will work well with training data but the results from test data will likely be unacceptable.

Undoubtedly, an adequate selection of the data is the most critical point. Sometimes, several similar patterns for the same defined fault are available. In other cases, similar steady state data for the same defined fault are available. The data has to be split in three data sets: for training, for testing and for cross validation. This last group consists of a data set for final validation. Sometimes, the developer may split the data into different random sets (to retrain and test again). The cross validation data set is unused data that the ANN never "sees" during the development. The objective is to provide an independent test for ANN performance.

The input scaling has to be taken into account. In order to have acceptable results, in complex problems a multiple ANN, one for each fault (multiple input, one output) has to be chosen. Generally, one hidden layer is enough. It is important to experiment with different types of ANN.

Finally, it is necessary to ensure that someone from the end user's staff is available to assist with integrating the ANN into their system.

5.5. Fuzzy Logic System development

The FLS has two kind of inputs: the ANN's outputs and the measurements from the plant. The ANN's outputs are already determined by the step of ANN implementation. The selection of the measurements from the plant that are also inputs to the FLS can be determined according to the generated if-then rules. The outputs of the FLS are the set of the defined faults.

The inference engine has the knowledge base, expressed in a set of if-then rules. These rules are of two types: those containing process deep knowledge and those that are built from experience of the ANN's performance. In Table 5.1, a scheme of the set of rules has already been presented

5.5.1 Generation of if-then rules from HAZOP analysis

HAZOP analysis allows to generate the if-then rules based on process deep knowledge. The column Causes in the HAZOP analysis are considered as faults. Other important point is the consideration of available measurements (direct or indirect). They are going to be the antecedents in the if-then rules and they are the parameters in the HAZOP analysis. By this way, the number of if-then rules generated from the HAZOP analysis is reduced. The final step is the adjustment of the membership functions.

The following example shows the easy generation of if-then rules from HAZOP analysis. Table 5.2 shows a partial HAZOP analysis of the plant shown in Figure 5.15. The fault is “pump 1 malfunction” (column "Causes"). Consequently, the conversion to an if-then rule is as follows:

IF Flow IS Low THEN “Pump malfunction” IS HIGH.

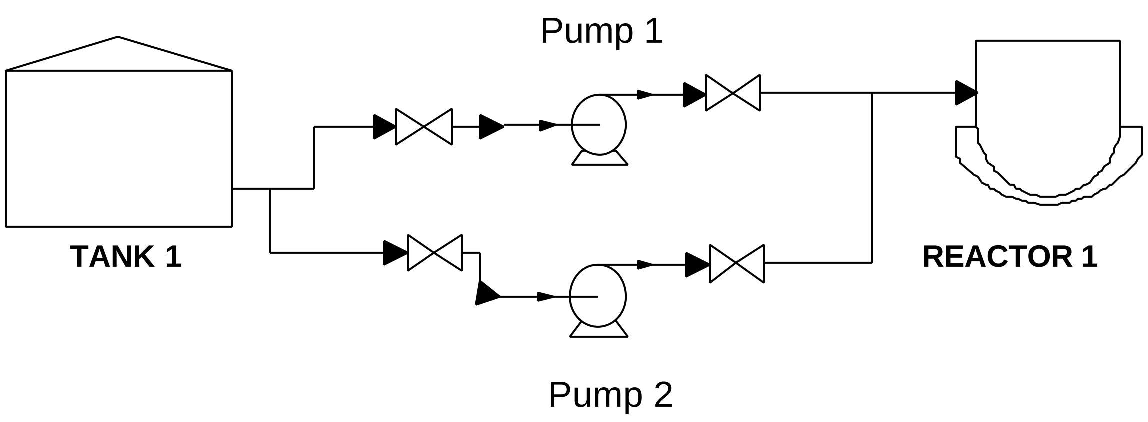


Figure 5.15. Example: Some stream lines in a chemical plant

Table 5.2. A HAZOP analysis of a stream line in a batch process.

Stage: Reactor charge; Element: Tank nº1; Node: Pipe from tank 1 to pump 1; State: to provide reactant to the reactors; Parameter: Flow

Guideword	Causes	Consequences	Corrective actions	Safeguards
Low	Pump 1 malfunction	Time needed for reactor charge increased	Switch to Pump 2	Maintenance tests

The generation of if-then rules from HAZOP analysis can be automated by using the algorithm shown in Figure 5.16.

Each parameter in the HAZOP analysis, that is measured or observed from the plant, is considered. The deviations HIGH and LOW in such analysis drives to the if-then rules, only if the considered deviations are included in the set of faults.

For each fault, the if-then rules are defined. One of them has as a consequent the fuzzy set HIGH, corresponding to the considered deviation. The other one has a fuzzy set LOW corresponding to the normal operating condition of the considered parameter.

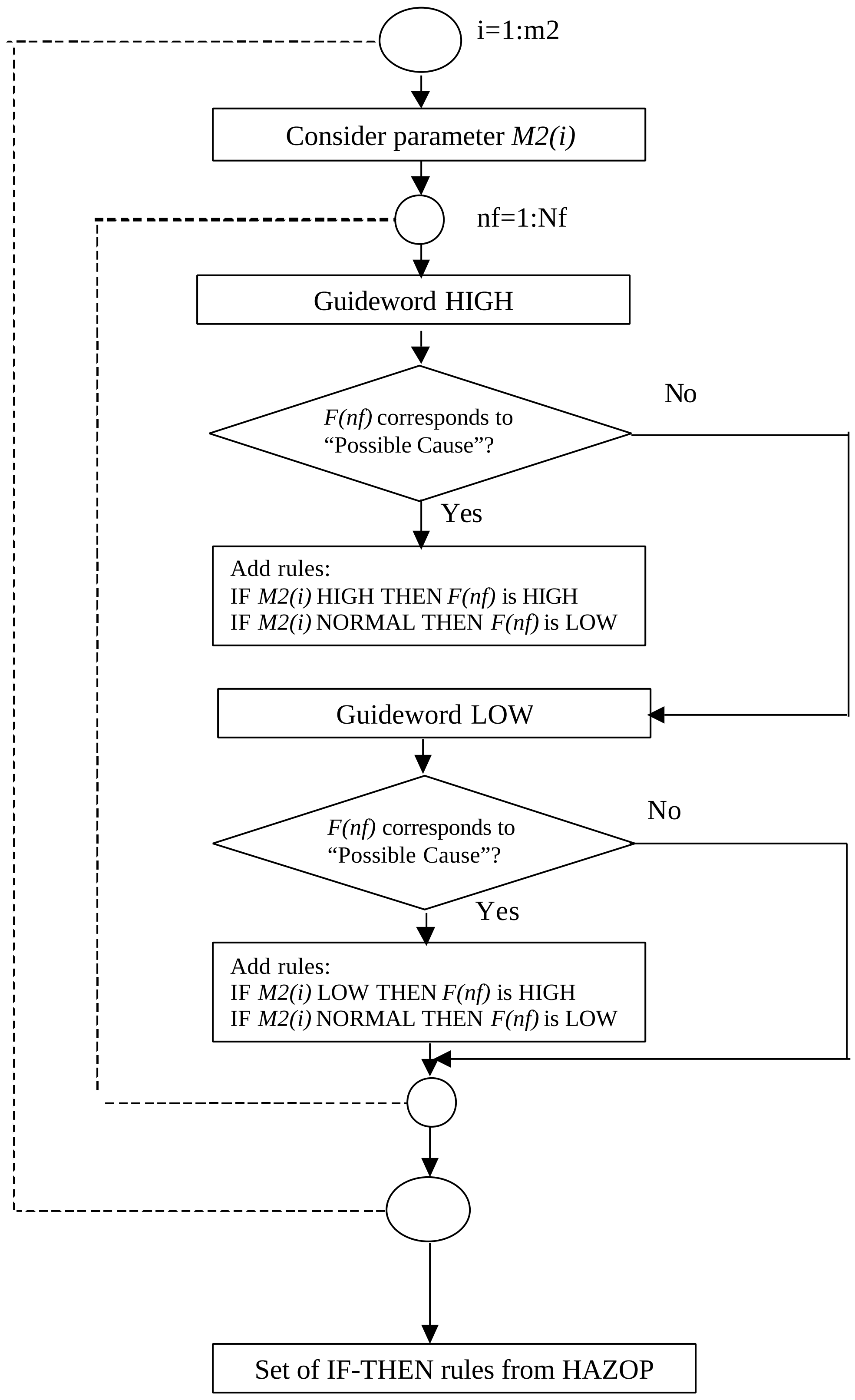

i=1:m2
Consider parameter *M2(i)*
nf=1:Nf
Guideword HIGH
F(nf) corresponds to "Possible Cause"?
No
Yes
Add rules:
IF *M2(i)* HIGH THEN *F(nf)* is HIGH
IF *M2(i)* NORMAL THEN *F(nf)* is LOW
Guideword LOW
F(nf) corresponds to "Possible Cause"?
No
Yes
Add rules:
IF *M2(i)* LOW THEN *F(nf)* is HIGH
IF *M2(i)* NORMAL THEN *F(nf)* is LOW
Set of IF-THEN rules from HAZOP

Figure 5.16. Generation of if then rules from HAZOP analysis

5.5.2. Generation of if-then rules from ANN performance experience

The if-then rules obtained on the basis of the experience with ANN performance can have the following form:

If N1(nf) Is HIGH then F(nf) is HIGH

If N1(nf) is LOW then F(nf) is LOW

More complete if-then rules takes into account the other measurements. For example:

If N1(nf) is HIGH and M2(i) is LOW then F(nf) is HIGH

These last if-then rules can be obtained by simulating the different faults and observing the ANN behaviour.

5.5.3. Adjustment of the membership functions

Trapezoidal MFs are proposed for *M2*. Three main MFs are necessary: µlow, µnormal and µhigh. The following steps are followed in order to adjust the mentioned membership functions:

Assign µnormal=1 to the range of normal operating conditions.

Assign µnormal =0 from the value considered as abnormal situation

Assign µhigh=0 / µlow=0 in the bounds of the range of µnormal =1

Assign µhigh=1 / µlow=1 in the considered abnormal conditions (low or high, respectively).

Figure 5.17 shows the general scheme of the membership function according to the described assignment. In this example, the range of normal operating conditions is 70-80 (e.g. ºC). A value less than 60 has been considered low (an abnormal operating condition). On the other hand, a value that is more than 90 has been considered high (an abnormal operating condition).

With respect to the outputs, the crisp functions are only two: µfault (=1) and µnofault (=0).

By this way, it is easy for plant engineers to adjust the MFs.

There are other variations of MF adjustment. They are illustrated in some comparisons of FLSs in Chapter 7, section 7.2.4. However, the shown choice is recommended (Ruiz et al., 2001f).

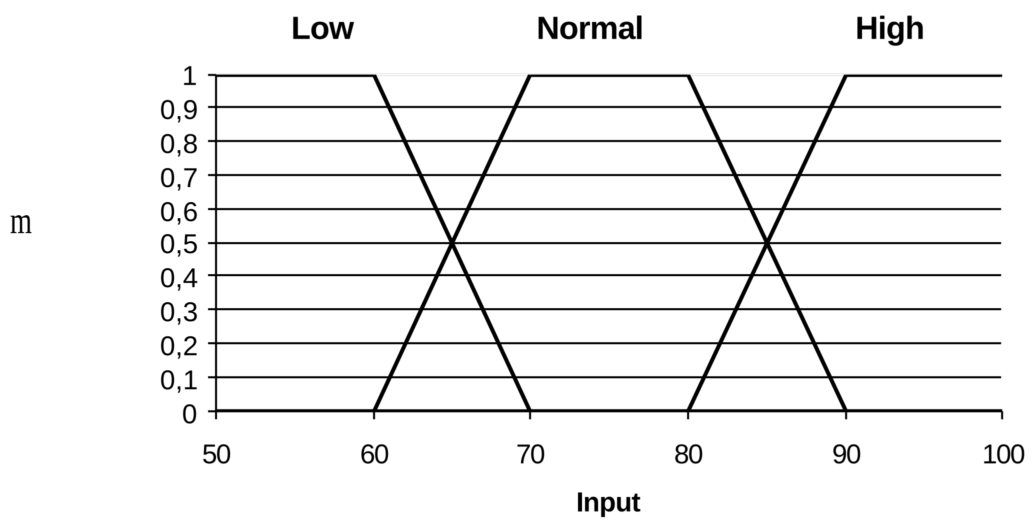


Figure 5.17. Input membership functions adjustment

5.6. On line implementation of the FDS

Figure 5.18 shows the algorithm for on line implementation of the FDS. It is in accordance to MATLAB programming language that has been the one utilised. The inputs are the *M1* and *M2* vectors and the time. The output is the vector *F*. Some values from the previous function FDS call are necessary if the signal pre-processing is utilised. Such variables are indicated as global variables.

Then, a loop is used to perform wavelet decomposition and extrema determination for all the *m1* variables that are inputs of the ANN block. Details at level 5 and the mother wavelet Daubechies-8 have been chosen. Then, the first and second derivatives of the signal Detail 5 (D5) are determined. Extrema determination is performed using the algorithm shown in section 5.4.2 (Figure 5.11).

After the explained loop, the variables time, detail and its first and second derivatives are updated.

Then, the output of the ANN block is determined. In case of using direct measurements (without signal pre-processing), the input is M1 instead of the extrema (M1').

Finally, the vector F is obtained by calling the function FLS. Its inputs are the M2 and M1 vectors.

```
function F=FDS (M1,M2,time)
global timeprevious D5previous D5'previous D5''previous
for i=1:n1
        D5(i)=WaveletDecomposition(M1(i),time,5,'db8');
        D5'(i)=(D5(i)-D5previous(i))/(time-timeprevious);
        D5''(i)= (D5'(i)-D5'previous(i))/(time-timeprevious);
        M1'(i)=ExtremaDetermination(D5(i),D5'(i),D5''(i));
end
timeprevious=time;
D5previous=D5;
D5'previous=D5';
D5''previous=D5'';
N1=ANN(M1');
F=FLS(M2,N1);
```

Figure 5.18. Main program for on line implementation of the FDS

Figure 5.19 shows the determination of the ANN output for the case of using a BPN and the vector *M1* as input. The weights and biases (*W* and *b*, the subindex indicates the layer number) have already been determined during the FDS development procedure. The number of nodes S_1 in the hidden layer has been the result of the performance optimisation approach. In the first loop, the hidden layer output is calculated. A sigmoidal function is used. In the second loop, the ANN output is determined by a linear funcion.

```
function N1=ANN(M1)
for i=1:$S_1$
    $n_1(i) = \sum_{m=1}^{m1} (W_1(i,m) \bullet M1(m) + b_1(i))$ ;
    $a_1(i) = \frac{1}{1+e^{-n_1(i)}}$ ;
end
for i=1:n1
    $n_2(i) = \sum_{m=1}^{S_1} W_2(i,m) \bullet a_1(m) + b_2(i)$ ;
    $N1(i) = n_2(i)$ ;
end
```

Figure 5.19. ANN output determination, using a BPN

Figure 5.20 shows the FLS function algorithm. The first loop considers the so called "firing" of the if-then rule *rl* by applying the MF to the antecedents. They can be MF values of the *M2* or *N1* variables. In the case of *M2*, the MFs can be low, normal or high, as has been shown in section 5.5.3. In the case of *N1*, the MF can be low or high.

After this fuzzification of the input, the fuzzy set for the consequent *F(nf)* is determined (C_{nf}^{rl}). The approach shown in section 4.2.1, Figure 4.7, is used for multiple antecedents. After this inference procedure, the loop continues with the following rule until finishing with all the defined *Rl* if-then rules. The following loop considers the determination of the fuzzy sets for each of the defined faults *F(nf)*. This procedure usually called aggregation is done by determining the union of the fuzzy sets calculated in the previous loop. By this way, the fuzzy output set for each fault is obtained. Finally, the defuzzification step using the centroid method is done (section 4.2.2).

The resulting crisp output vector is the FLS output, and also corresponds to the FDS output.

F= FLS(M2,N1)

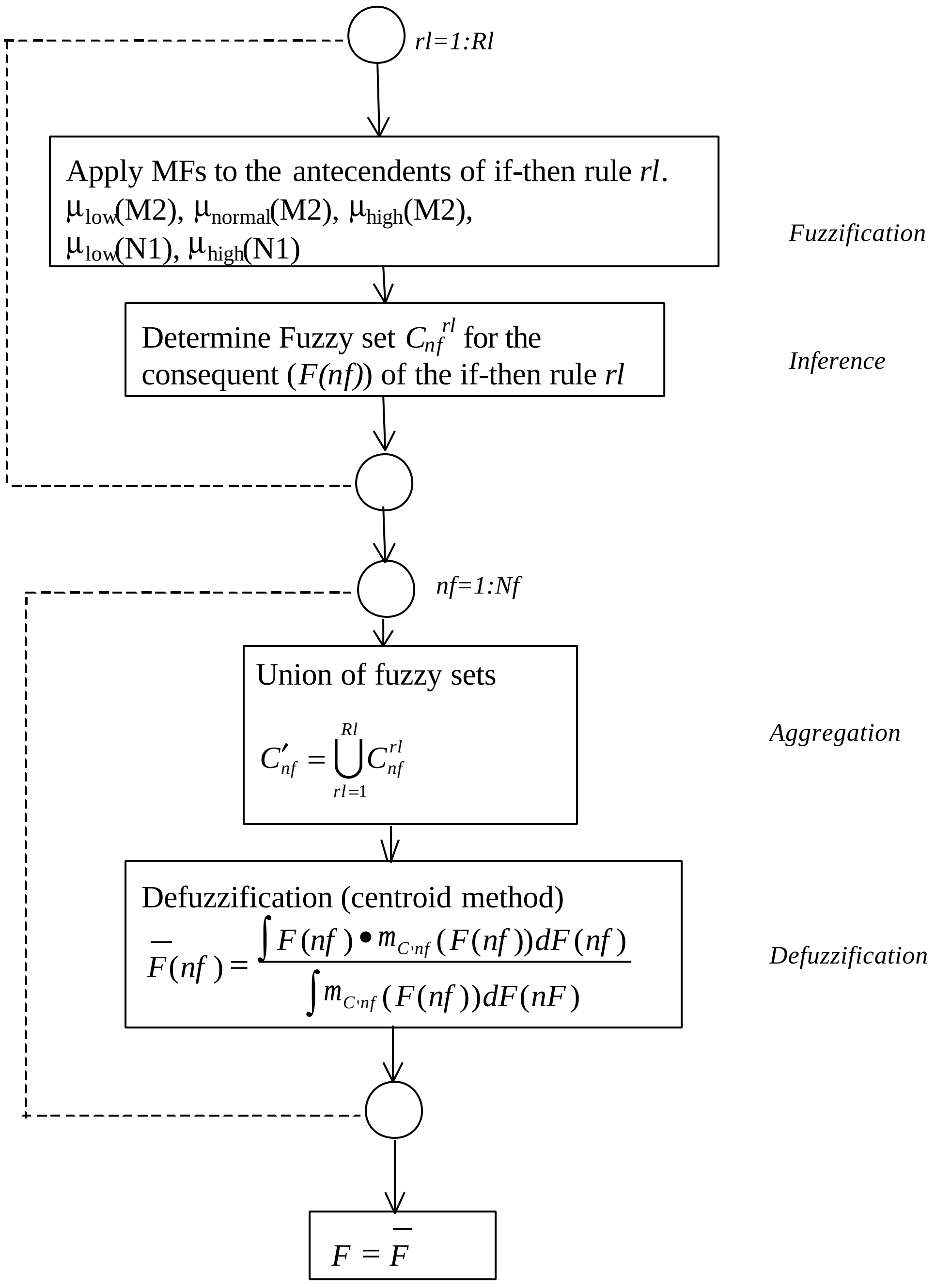

rl=1:Rl
Apply MFs to the antecendents of if-then rule rl.
μ_{low}(M2), μ_{normal}(M2), μ_{high}(M2),
μ_{low}(N1), μ_{high}(N1)
Fuzzification
Determine Fuzzy set C_{nf}^{rl} for the consequent (F(nf)) of the if-then rule rl
Inference
nf=1:Nf
Union of fuzzy sets
$C'_{nf} = \bigcup_{rl=1}^{Rl} C_{nf}^{rl}$
Aggregation
Defuzzification (centroid method)
$\overline{F}(nf) = \frac{\int F(nf) \bullet m_{C'nf}(F(nf))dF(nf)}{\int m_{C'nf}(F(nf))dF(nF)}$
Defuzzification
$F = \overline{F}$

Figure 5.20. FLS algorithm for on line implementation of the proposed FDS

A simple example is shown in Figure 5.21. Two if then rules are considered. One of them has a *M2* value as antecedent and the other one a *N1* value. A fuzzy set for the consequent fault *F(nf)* is determined for each rule. The inference in each rule is simple because there is no AND/OR operators. Then, the aggregation is performed by the union of the obtained fuzzy sets. Finally, using the centroid method, the crisp output is obtained.

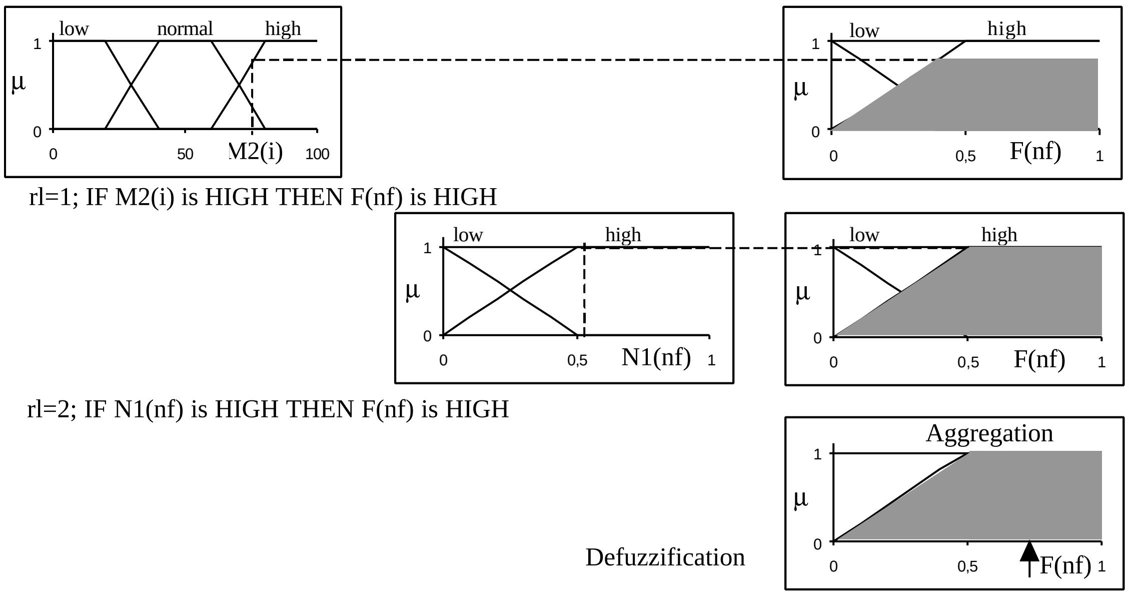

low
normal
high
μ
M2(i)
rl=1; IF M2(i) is HIGH THEN F(nf) is HIGH
low
high
N1(nf)
rl=2; IF N1(nf) is HIGH THEN F(nf) is HIGH
F(nf)
Aggregation
Defuzzification

Figure 5.21. A simple example of the application of the FLS algorithm

5.7. Determination of the information to be sent to other levels

The FDS system receives sensor data from the plant and the control signals. They can be continuous signals (temperatures, flowrates, pressures) or discrete signals (valves open or close, pumps on or off). The outputs are a set of suspected faults. The signal corresponding to each suspected fault is considered binary (0 or 1). This output can be used by the advanced control module in order to take control actions, or by the operators who have to take decisions or by other levels in the computer system as the scheduling system. The output of the FDS system has different forms according to the level of information.

HAZOP analysis can help in this "translation" of the information provided by the FDS, as it will be shown now (Ruiz et al., 2000a). Following the example shown in Figure 5.15 and according to the explained HAZOP analysis (Table 5.2), the information at different levels from the FDS system output when the fault “Pump 1 malfunction” is diagnosed, is summarised in Table 5.3. Note that the construction of that table is straightforward from the HAZOP analysis.

Table 5.3. Information to be sent to other levels

Module	Translation from FDS output
Control System	Switch to Pump 2
Scheduling System	Time needed for tank charge increased
Operator's console	Check the Pump 1

5.8. Conclusions

The proposed FDS has been introduced. It consists in a combination of a pattern recognition approach based on an ANN and inference system based on fuzzy logic.

After a general description of the approach, the successive steps for its development has been described. Then, the development of the ANN and FLS blocks has been specially considered due to their importance in the FDS framework.

The ANN is trained by supervised learning. It can be done directly with the saved variables' profiles or by using the signal features. If the first alternative does not show an acceptable performance, the last one is recommended. Such option requires signal pre-processing using wavelets at a high level and extrema determination for feature extraction. The selection of the type of ANN is not a critical point. Nevertheless, a RBFN trains faster and usually has better performance. However, for high dimensional training data sets, it is not the recommended option because of the high computational requirements involved (RBFN of high dimensions). On the other hand, a PNN is suitable for the classification of fault patterns composed by signal features.

With respect to the FLS, the use of a Mamdami system has been commented. However, Sugeno FLS type can be also utilised. Results of its application will be shown later on (Chapter 7, section 7.2.4).

The program for the on line implementation of the FDS has been shown and explained in detail.

Finally, the basis for the translation of the FDS signal, in order to be utilised at other levels in the information system as the scheduling level has been presented

Acronyms

AANN	Autoassociative Artificial Neural Network
ANN	Artificial Neural Network
DCS	Distributed Control System
FDS	Fault Diagnosis System
FLS	Fuzzy Logic System
HAZOP	Hazard and Operability study
MF	Membership function
NLPCA	Nonlinear Principal Component Analysis
PCA	Principal Component Analysis
PNN	Probabilistic Artificial Neural Network
SOM	Self Organising Map
SPE	Squared Prediction Error

Notation

a_i	Output vector of the ith layer in a BPN
b_i	Bias vextor of the ith layer in a BPN
Di	Detail of the ith wavelet decomposition
F	Fault vector
j	Index for measurements (sensors)
J	Total number of measurements (sensors)
k	Index for time intervals
K	Total number of time intervals
m	Index for nodes in an ANN
$M1$	Vector of measurements from the plant that are the ANN's input
$M1K^{nf}$	Matrix with the measurements profiles for the fault nf
$M1'K^{nf}$	Matrix of extrema of the measurements for the fault nf
$m1$	Length of vector M1
$M2$	Vector of measurements from the plant that are part of the FLS's input
$m2$	Total number of measurements that are part of the FLS's input
n_i	Activation status vector of the ith layer in a BPN
$N1$	Output vector of the ANN in the proposed FDS
$n1$	Total number of "pre-faults" diagnosed by the ANN
Nf	Total number of defined faults
nf	Index for faults
$\%P$	Performance parameter for FDS
$\%P^*$	Modified %P which takes into account cases of false diagnosis
R	Range of the $SPE_{js}\%$ of the set of sensors
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
SPE_j	SPE calculated for the measurement j and the corresponding output of the AANN
SPE_{js}	99% upper control limit of SPE_j for a set of correct measurements
$SPE_{js}\%$	Percentage relation between SPE_j of a new measurement and SPE_{js}
SPE_{sup}	99% upper control limit of SPE for a set of correct measurements
td	Time spent by a FDS for correct diagnosis
tss	Time with the plant in non-steady state after a fault occurs
W_i	Weight matrix of the ith layer in a BPN

Greek symbols

e	Small number
m	Membership function
σ	Standard deviation