Fault Identification and Diagnosis for Telephone Exchange Building Facilities

S.R. De Waard

Dissertation submitted in partial fulfilment of the requirements of the degree Magister Ingeneriae in Electronic Engineering at the North-West University

Supervisor: Professor C.P. Bodenstein

2005

Potchefstroom campus

Abstract

Heating, ventilating and air conditioning (HVAC) systems consume 43 % of the energy used by buildings. This percentage grows when the HVAC system operates with malfunctions. Fault detection and diagnosis (FDD) methods are developed to reduce abnormal events and down-times and to promote energy saving use of equipment.

Most FDD methodologies for HVAC systems found in the literature revolve around first principle models and mathematical models. This dissertation describes a FDD solution based on process history data and artificial neural network (ANN) models.

ANN models, of HVAC components, are built from fault-free operation data. Faulty data are then used with the ANN models to build various residuals and statistical residual transformations. From these residuals, unique residual patterns are assigned to discern between a variety of malfunctions.

This FDD strategy is, firstly, applied to a static pressure control loop and secondly, applied to the overall power consumption of an HVAC system. In both studies, the FDD system successfully detected and classified unwanted anomalies – some deviating as little as 5% from normal operational standards.

Finally, the FDD system is rated according to a common set of criteria reviewed in the literature study. This criterion shows the FDD strategy to be robust and adaptable, with low modelling and computational requirements.

Uittreksel

Verhitting, ventilasie en lugversorgings (HVAC) stelsels gebruik 43 % van die totale elektrisiteitstoevoer tot geboue. Hierdie persentasie vergroot wanneer die HVAC stelsel met abnormaliteite moet funksioneer. Foutsporings en diagnose (FDD) metodes word tans ontwikkel om die duur van foutiewe werksverrigting en hersteltye te verminder. Verder is dit ook handig om lae kragverbruik aan te moedig.

Meeste FDD metodes vir HVAC stelsels is gebasseer op eerste-beginsel modelle en wiskundige modelle. Hierdie verhandeling hou 'n FDD sisteem voor wat ontwikkel is rondom kunsmatige neurale netwerke (ANN) en die gepaardgaande historiese proses data.

ANN modelle van HVAC komponente word opgelei met data vanuit foutlose datastelle. Daarna word foutdata gebruik om, deur middel van die opgeleide netwerke, residue en statistiese verwerkings van residue te bekom. Uit hierdie residue en verskilpatrone kan unieke kenmerke aan verskillende foute toegeken word. So word daar dus tussen foute onderskei.

Hierdie FDD strategie word, eerstens, toegepas op 'n statiese lugdruk beheerlus en tweedens op die oorsigtelike kragverbruik van 'n HVAC sisteem. In beide studies kry die FDD stelsel dit suksesvol reg om foute in die sisteem te ondek en te klassifiseer. Van die foute het 'n afwyking van slegs 5% van die normale lesings.

In albei eksprimente word die FDD stelsel geëvalueer volgens standaarde wat in die literatuurstudie uiteengesit word. Hierdie standaarde beskryf die FDD stelsel as robuust en aanpasbaar met lae modellerings- en berekenings-vereistes.

Acknowledgements

I would like to take this opportunity to thank all the individuals who, by any means were involved the compilation of this dissertation. Without the support, insight and help I received from friends, colleagues and other people this study would not be the work it is today. I would like to acknowledge those who turned this study from an idea into a reality.

The most important contributor, with his knowledge and leadership, is my supervisor Professor Charles Bodenstein. His insight and guidance from the initiation of the study remains an invaluable tool to me.

For their endless support I would like to thank my family, my father, Jan de Waard for everything he had done for me out of love, my mother, Marianne de Waard for her unbounded love and prayers and Liesl de Waard, who relentlessly motivated me.

I would like to thank Anneme Smit who joined my life in the final six months of this study and helped me see it through.

For proofreading the dissertation I thank Ms JA Brönn. Her thorough understanding of spelling and grammar greatly improved the readability of this text.

To Professor Alwyn Hoffman and THRIP I owe thanks for their generous financial support.

Finally I would like to thank TFMC for initiating the project.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Abbreviation	Description
AEM	Abnormal event management
AFMM	Adaptive forgetting through multiple models
AHU	Air-handling unit
ANN	Artificial neural network
ARX	Auto regressive exogenous
CE-graph	Cause-effect graph
CR	Compensatory response
CV	Compensatory variables
EKF	Extended Kalman Filter
ESDG	Extended signed digraphs
FDD	Fault detection and diagnosis
FID	Fault identification and diagnosis
HVAC	Heating ventilating and air-conditioning
IR	Inverse response
IV	Inverse variables
MSCC	Maximal strongly connected component
MSE	Mean square error
MLP	Multi layer perceptron
ODE	Ordinary differential equation
PDE	Partial differential equation
PLS	Partial least squares
PCA	Principle component analysis
QDE	Qualitative differential equations
QPT	Qualitative process theory
QSIM	Qualitative simulation function
ROC	Rate of change
RMS	Root mean square
SDG	Signed digraphs
SCP	Simple causal paths
STD	Standard deviation
SPC	Statistical process control
VAR	Statistical variance
SCC	Strongly connected component
VAV	Variable air volume

CHAPTER 1

1 INTRODUCTION

This chapter is written in order to set the focus of this dissertation. The problem is defined from the background of the world energy situation and a methodology to solve the problem is considered. The introduction concludes by providing an overview of the dissertation.

1.1 Energy consumption problem and availability of building air-conditioning systems

As the worldwide energy crisis grows, energy saving is becoming an increasingly important issue. Building energy consumption occupies one-third of total energy consumption [1]. Because building control systems and heating ventilating and air-conditioning (HVAC) systems do not run under optimal conditions and regularly suffer from faults, the potential for energy saving is considerable. Furthermore, faults in the building HVAC system can also lead to degradation of the indoor climate, raising the complaint level of occupants. Even some critical electronic equipment, like telephone exchanges, can trip or malfunction when temperatures are too high.

Fault detection and diagnosis (FDD) are applied to trace the cause of a decrease in indoor climate quality and energy efficiency of HVAC systems. It can be realized with quantitative and qualitative approaches. The quantitative approach is normally based on the physical laws and requires advanced knowledge about the system. Nowadays, detailed information of the building and its substructures is available from the building management. Therefore, the physical models can be built reasonably accurately. Output from the model is weighed against measured values from the HVAC system to build residuals that are analyzed for FDD purposes. But, building configurations regularly change when partitions are installed or moved. Each alteration requires an impractical redesign of the physics-based model.

As an alternative to first principal models, this dissertation concerns itself with the implementation of a process history based model. Process history based models are built from vast sums of historical datasets. No knowledge of the physical interactions in the plant is necessary to construct process history models, but some form of feature extraction has to be applied to the data in order to generate the models. Feature extraction can be done with either statistical methods or non-statistical methods like fuzzy logic or artificial neural networks (ANN).

With an ANN model at hand, fluctuations in the physical plant can be analyzed for FDD purposes. Quick and effective response to FDD alarms should limit downtime, and reduce resource misuse.

1.2 Problem statement

The above-mentioned factors stress the necessity of being able to predict failure of components or a faulty process status. The goal of this dissertation is to investigate a FDD concept that can detect and recognise faulty behaviour of a HVAC process.

The FDD model under study is the combination of three different sub-systems: a process model, a residual analysis and a fault classifier. Figure 1 presents the interactions in between the FDD sub-systems and how a fault would propagate to the point where it is identified. Mainly, this dissertation focuses on the building of the

process model but the residual analysis and the classification structure are also thoroughly discussed.

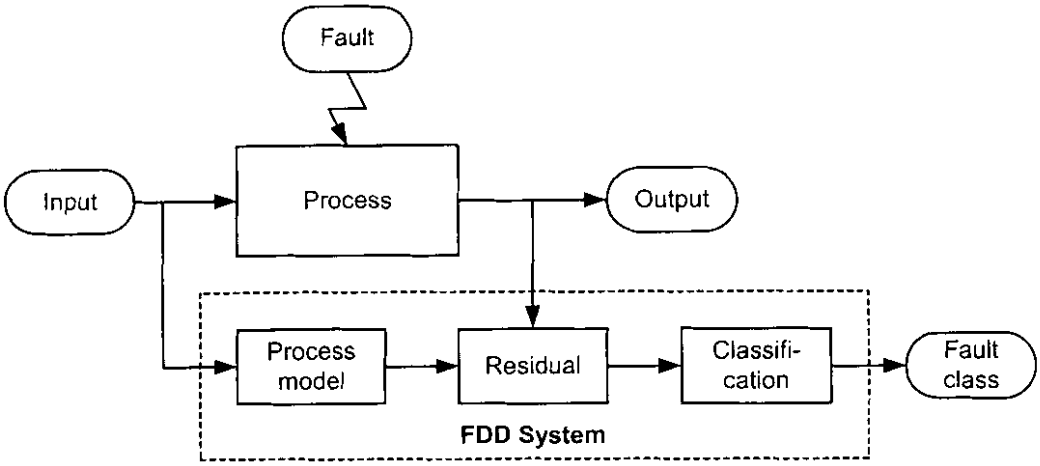


Figure 1. Interaction of a process with an FDD system.

Issues leading from the research goal include most problems associated with FDD systems, such is the ability to rate a FDD system. Desirable characteristics need to be identified for comparison between different systems and their usefulness under different circumstances [2].

System identification concerns the problem of obtaining mathematical models of dynamical systems based on observed data [3]. These models are actually approximations of the real life processes. Neural networks are a subset of system identification [4].

It is proposed that an artificial neural network be used for modelling building systems. System identification, in general, has the ability to model any dynamic system from observed data.

This type of modelling, using ANNs, has the following advantages:

- A short time is needed for model development
- Data from the real physical process are used to obtain models
- Models include disturbance variables
- Models can be implemented in a model based control system.

In the end a generic FDD system should be developed, not only for implementation on HVAC equipment, but for commissioning on any system or process.

1.3 Proposed methodology

The engineering of an advanced FDD system will be developed over a number of stages.

1. Literature studies on system identification, fault detection and diagnostics as well as on heating, ventilating and air-conditioning are required.

2. Then, it is necessary to create and train the neural network within the boundaries of a laboratory, including:
   - modelling a process using neural networks for fault detection;
   - detecting and diagnosing faults by using statistical methods to classify residuals; and
   - integrating the above mentioned solutions;

3. The same methods of point 2 should be implemented on HVAC systems.
   - One approach would be to employ the ANNs on some other type of simulation model of a HVAC system;
   - Ideally, the ANN models should be developed around measured data from some physical HVAC system;

4. Re-evaluation of the proposed FDD system should be done according to the efforts from the literature study in point 1.

1.4 Chapter summary

Chapter 2 summarises the background theory of fault detection and diagnostic systems and weighs different modelling methods up against each other.

Chapter 3 covers the basic theory of heating ventilating and air-conditioning systems. The chapter continues with a literature survey of applied FDD on HVAC systems and

concludes with the motivation of neural networks as this dissertation's method of choice.

Chapter 4 investigates neural networks as fault identifiers. Residuals are obtained by implementing two transfer function models that run in series against two ANN models; one replicating the first system while the second attempting to replicate both transfer functions. From different residual adaptations a fault identification matrix is derived.

Chapter 5 is concerned with applying the methods investigated chapter 4. This is done on a simulation of a static pressure control loop in the attempt to detect and identify errors that are specifically related to such a HVAC component. A complete FDD solution is developed and rated accordingly.

Chapter 6 applies the FDD methods developed in the two previous chapters. The neural networks are trained with data collected from a telephone exchange cooling system. The FDD system is developed to identify measured faults and artificial faults. The system is also capable to detect unknown malfunctions. At the end of the chapter the FDD system is evaluated according to methods found in the literature study.

Chapter 7 summarizes this dissertation in a final conclusion on this research.

Appendix A is a collection of the fine details involved in the development of accurate neural networks. This appendix is added to extend chapters 4, 5 and 6 while not reducing the readability of those chapters.

1.5 References in chapter 1

[1] Yu, B. & van Paassen, A. H. C. *Modeling with simulink and bond graph method for fault detection in an air-conditioned room.* 2001. Lab of Refrigeration Engineering & Indoor Climate Control. Delft University of Technology

[2] Venkatasubramanian, V. et al. *A review of process fault detection and diagnosis.* 2001. Computers and chemical Engineering.

[3] Erasmus, W.O. *Modelling the pebble bed modular reactor using system identification techniques*. 2003. PU vir CHO.

[4] Haykin, S. *Neural Networks, a comprehensive foundation*. 1999. Prentice Hall.

CHAPTER 2

2 GENERAL OVERVIEW OF FAULT DETECTION AND DIAGNOSTIC METHODS

Fault detection and diagnostics (FDD) potentially have great economic impact to increase plant availability. In the case of HVAC systems, energy efficiency is of prime importance; doors and windows left open could greatly increase the cooling load. An overview on FDD is found, amongst others, in an article by Venkatasubramanian [1]. It describes how various FDD methods are implemented to handle abnormal events.

There is an abundance of literature on process fault diagnosis ranging from analytical methods to artificial intelligence and statistical approaches. From a modelling perspective, there are methods that require accurate process models, semi-quantitative models, or qualitative models. At the other end of the spectrum, there are methods that do not assume any form of model information and rely only on historic process data.

In this chapter faults that lead to abnormal events are defined. The characteristics of an FDD system to manage these events are considered before different FDD methods are discussed.

2.1 Introduction to abnormal event management

As our knowledge on process control and computer systems grows, humans in low-level and regulatory control positions are being replaced by machines capable of routinely performing these actions in an automated manner. With progress in distributed control and model predictive control systems, the benefits to various industrial segments have been enormous. However, a very important control task in managing process plants still remains largely a manual activity, performed by human operators. This is the task of responding to abnormal events in a process. Broken down into steps, this task involves the timely detection of an abnormal event, diagnosing its origins and then taking appropriate actions to bring the process back to a normal, safe, operating state. This entire activity has come to be called abnormal event management (AEM).[1].

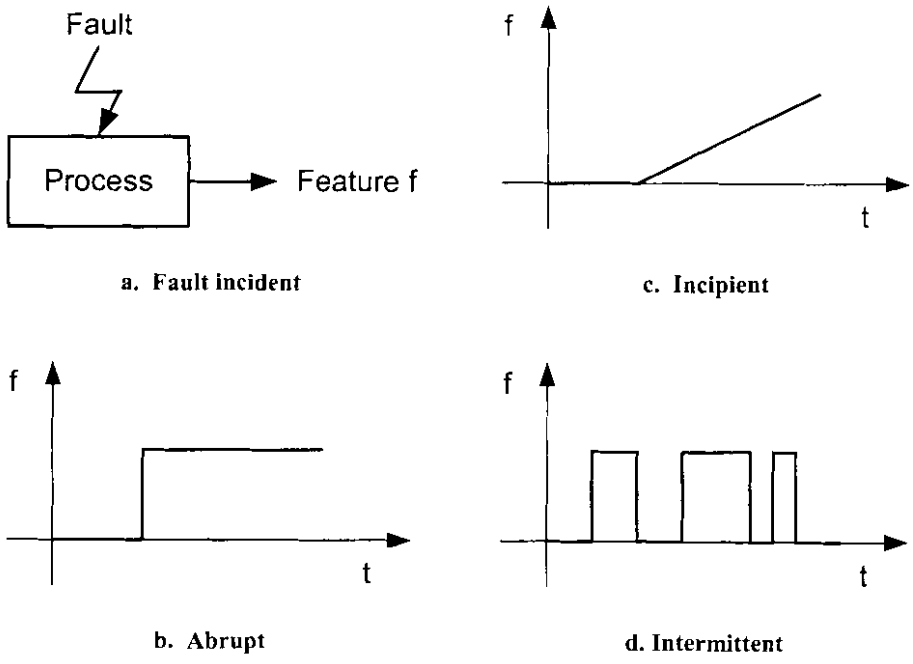
Reliance on humans for AEM is becoming increasingly unsuccessful because of several factors. Firstly, the broad scope of diagnostic activity on multiple failures gets confusing. Secondly, humans have a hard time to encompass the shear size of the modern plant. Some plants have as many as 1500 process variables observed every few seconds [2]. This leads to information overload. A third problem is the failing of measuring equipment and sensors. Incomplete and unreliable data make diagnostics a tedious task. Finally, speed is a rather desirable attribute to any diagnostic system. It would be impossible for a human to reach all the constraints and demands that are required of a modern diagnostic system.[1].

The automation of FDD is the first building block in modern AEM. Various computer-aided methods have been developed to address the difficulties of the broad scope of fault diagnosis and its real time solution. From a modeling perspective, there are methods that require accurate process models, semi-quantitative models, or qualitative models. At the other end of the spectrum, there are methods that do not assume any form of model information and rely only on process history information. In addition, given the process knowledge, there are different search techniques that can be applied to perform diagnosis. [1].

The basic aim of this section is to provide a comparative study of various diagnostic methods from different perspectives. Diagnostic methods are classified into three general categories: quantitative model based methods, qualitative model based methods, and process history based methods. This review includes a perspective showing how these different methods relate to and differ from each other. Included are important assumptions, drawbacks as well as advantages. Due to the broad scope of this review it is not possible to discuss every method, nor to confer fine detail. Hence the intent is to provide the reader with the general concepts, and motivate the choices made for this study on the popular FDD methods.

2.2 Definition of a fault

AEM revolves around process faults. The term fault is generally defined as a departure from an acceptable range of an observed variable or a calculated parameter associated with a process [2]. According to Isermann [3] faults disturb data in mainly three shapes, presented in Figure 2. Abrupt faults assume the form of a step function while incipient faults periodically drift away from the desired value. Faults which appear, disappear and reappear are called intermittent faults.



a. Fault incident

c. Incipient

b. Abrupt

d. Intermittent

Figure 2. Fault appearance in a process [3]

The underlying cause of this abnormality is called the basic event or the root cause. The basic event is also referred to as a malfunction or a failure [2]. Since one can view the task of diagnosis as a classification problem, the diagnostic system is also referred to as a diagnostic classifier. Figure 3 depicts the components of a general fault diagnosis framework. The figure shows a controlled process system and indicates the different sources of failures in it. In general, one has to deal with three classes of malfunctions:

- Gross parameter changes
- Structural changes
- Failures of sensors and actuators

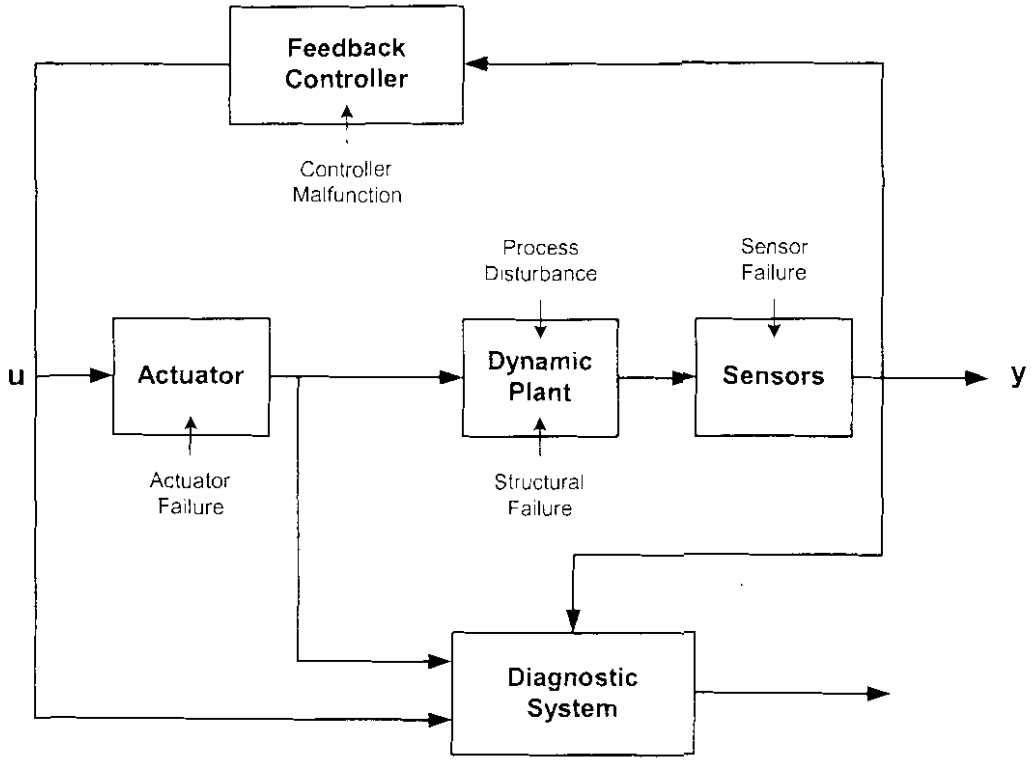


Figure 3. General diagnostic framework

2.2.1 Gross parameter changes in a model

In any model, there are processes occurring below the selected level of detail of the model. These processes, which are not modelled, are typically lumped as parameters and these include interactions across the system boundary. Parameter failures arise when there is a disturbance entering the process from the environment through such independent variables.

As an example of a gross parameter change, consider a pipe in which water flows. If the outside temperature drops below freezing point so that the water inside freezes, flow inside is halted. None of the equipment broke, but a failure occurred because the water, that was considered a liquid, has changed to a solid.

2.2.2 Structural changes

Structural changes refer to changes in the process itself. They occur due to hard failures in equipment. Structural malfunctions result in a change in the information flow between various variables. To identify such a failure, a diagnostic system would require the removal of the appropriate model equations and restructuring the other equations in order to describe the current situation of the process.

Once again consider a pipe as an example. If the pipe starts leaking it goes through a structural change, leading to an error when the input does not match the output. Similarly, a door to the outside left opening a cooled building space would lead to a fault and increase the energy consumption.

2.2.3 Malfunctioning sensors and actuators

Actuators and sensors are needed for, among other uses, detecting system failures. Unfortunately with these sensors another set of possible faults is incorporated. This set is divided into three categories: hard sensor failures, an added constant bias and an out-of range failure. Some of the instruments provide feedback signals, which are essential for the control of the plant. A failure in one of the instruments could cause the plant variables to deviate beyond acceptable limits unless the failure is detected promptly and corrective actions are accomplished in time. It is the purpose of diagnosis to quickly detect any instrument fault, which could seriously degrade the performance of the control system. Outside the scope of fault diagnosis are unstructured uncertainties, process noise and measurement noise. Unstructured uncertainties are mainly faults that are not modelled *a priori*. Process noise refers to the mismatch between the actual process and the predictions of model equations, whereas measurement noise refers to high frequency additive component in the sensor measurements.

2.3 Desirable characteristics of a fault diagnostic system

This section concerns itself with a set of characteristics that any FDD system should possess. This wish list will serve as a requirement set whereby the different diagnostic approaches will be benchmarked. Currently not a single approach fulfills all the requirements. Rather, the benchmark defines the method in terms of the *a priori* (or beforehand) information that needs to be provided, reliability of solution, generality and computational efficiency.

If an abnormality is detected, a general diagnostic classifier would come up with a set of hypotheses that explains the abnormality. Completeness of a diagnostic classifier would require the actual fault to be a subset of the proposed fault set. Resolution of a diagnostic classifier would require the fault set to be as minimal as possible. Thus, there is a trade-off between completeness and resolution.

The following presents a set of desirable characteristics one would like the diagnostic system to possess:

2.3.1 Quick detection and diagnosis

The FDD system should respond quickly in identifying malfunctions. However, quick response to failure diagnosis and tolerable performance during normal operation are two conflicting goals. A system that is designed to detect a failure (particularly abrupt changes) quickly will be sensitive to high frequency influences. This makes the system sensitive to noise and can lead to frequent false alarms during normal operation, which can be disruptive.[1].

2.3.2 Isolability

Isolability is the ability to distinguish between different failures – to isolate one fault from the probable fault set. In a noise-free state, the FDD classifier should be able to generate an output that is uniquely linked to faults that have not been modelled. However, the ability to design isolable classifiers mainly depends on the process characteristics. There is also a trade-off between isolability and the rejection of

modelling uncertainties. Most of the classifiers work with various forms of redundant information and hence there is only a limited degree of freedom for classifier design. Due to this, a classifier with a high degree of isolability would usually do a poor job in rejecting modelling uncertainties and vice versa. [1].

2.3.3 Robustness

One would like the diagnostic system to be robust to various forms of noise. In other words, the performance should degrade gracefully instead of failing abruptly as noise levels increase. Robustness rule out deterministic isolability tests where the thresholds are placed close to zero. In the presence of noise, these thresholds may have to be chosen conservatively. Thus, robustness is weighed against performance. [1].

2.3.4 Novelty identifiability

The first priority of any FDD system is to distinguish between normal and abnormal behaviour of the process. Secondly, in the case of abnormal behaviour, it is used to detect whether the malfunction is known or new (novel). This second criterion is known as novelty identifiability. Generally there are sufficient data available to model the normal behavior of a process. On the other hand, data sets needed for modeling the abnormal regions are usually incomplete. Thus, it is possible that much of the abnormal operations regions may not have been modelled adequately. Achieving complete novelty identifiability remains one of greatest challenges when designing an FDD system. When complete novelty identifiability can not be attained, one would like the diagnostic system to be able to recognize the occurrence of novel faults and not misclassify them as known malfunctions or as normal operation. [1].

2.3.5 Adaptability

Processes in general change due to changes in external inputs or structural changes brought along by retrofitting. These changes are not always failures. Sometimes the operating conditions can change as a result of changing environmental conditions such as changes in production quantities, changes in the quality of raw material etc. An FDD system should be adaptable to changes, with the possibility to gradually

develop the scope of the system as new cases and problems emerge, as the process matures. [1].

2.3.6 Explanation facility

Finding the source of a malfunction is a standard requirement on any FDD system. It would be impressive if the system could explain how the fault originated and propagated to the current situation. This is a very important factor in designing on-line decision support systems. This requires the ability to reason about cause and effect relationships in a process. An FDD system has to justify its recommendations so that operators can accordingly evaluate and act on their experience. As an extension on the capacity to build a certain hypothesis, the FDD system should motivate why another hypothesis is discarded. [1].

2.3.7 Modeling requirements

Another criterion whereby FDD systems are judged is the effort which goes into commissioning and deployment. For fast and easy deployment of real-time diagnostic classifiers, the modelling effort should be minimal. [1].

2.3.8 Dependability

Dependability is mainly a concern of availability of the FDD system, in other words, a system that has the property of always being available when required. It is the degree to which a system is operable and capable of performing its required function at any randomly chosen time during its specified operating time. [3]. Another description is:

$$\text{Dependability} = \frac{\text{Time available}}{\text{Time available} + \text{Time required}}$$

2.3.9 Storage and computational requirements

Some FDD systems require algorithms and an operating code of some sort to function while others are computationally less complex, but might entail high storage

requirements. An adequate diagnostic system is able to achieve a reasonable balance on these two competing requirements. [1].

2.3.10 Multiple fault identifiability

The facility to detect and identify multiple simultaneous faults is an important but difficult requirement. It becomes a complex problem thanks to the interacting nature of most faults. These interactions are usually synergistic and hence the individual fault patterns merge to produce a new pattern. Enumerating and designing separately for all the permutations and combinations between known faults would become combinatorial prohibitive for large processes. [1].

2.3.11 Safety

Safety requirements in an FDD solution does not merely concern the operator or user's safety need. It also requires the protection of the equipment and process hardware involved in the plant. [3].

2.4 Classifications of diagnostic algorithms

The two elements that form a diagnostic classifier are:

- The type of knowledge used and
- The search strategy

The diagnostic search strategy depends on the method or form in which the knowledge was transformed. In turn, the knowledge representation scheme depends on the *a priori* knowledge available. The conclusion is that the *a priori* knowledge available is the most distinguishing feature. The diagnostic systems are classified according to this.

Vekatasubramanian [1] defines *a priori* knowledge as a set of failures, plus a set of observations and the relationship between them. This can be explicitly represented, for example a table lookup scheme, or deducted from a source of domain knowledge. For example, the domain knowledge could be developed from a first principles understanding of the process. This is called model-based or causal knowledge [4].

Domain knowledge with an element of explicitly represented data is referred to as compiled or process history-based knowledge.

The model-based *a priori* knowledge can be broadly classified as qualitative or quantitative. The model is usually developed based on some fundamental understanding of the physics of the process. In quantitative models this understanding is expressed in terms of mathematical functional relationships between the inputs and outputs of the system. In contrast, in qualitative model equations these relationships are expressed in terms of qualitative functions around different units in a process. In contrast to the model-based approaches, in process history based methods only the availability of a large amount of historical process data is assumed. There are different ways in which these data can be transformed and presented as *a priori* knowledge to a diagnostic system. This is known as feature extraction from the process history data, and is done to facilitate later diagnosis. This extraction process can mainly proceed as either quantitative or qualitative feature extraction. In quantitative feature extraction one can perform either a statistical or non-statistical feature extraction. This classification of diagnostic systems is shown in Figure 4.

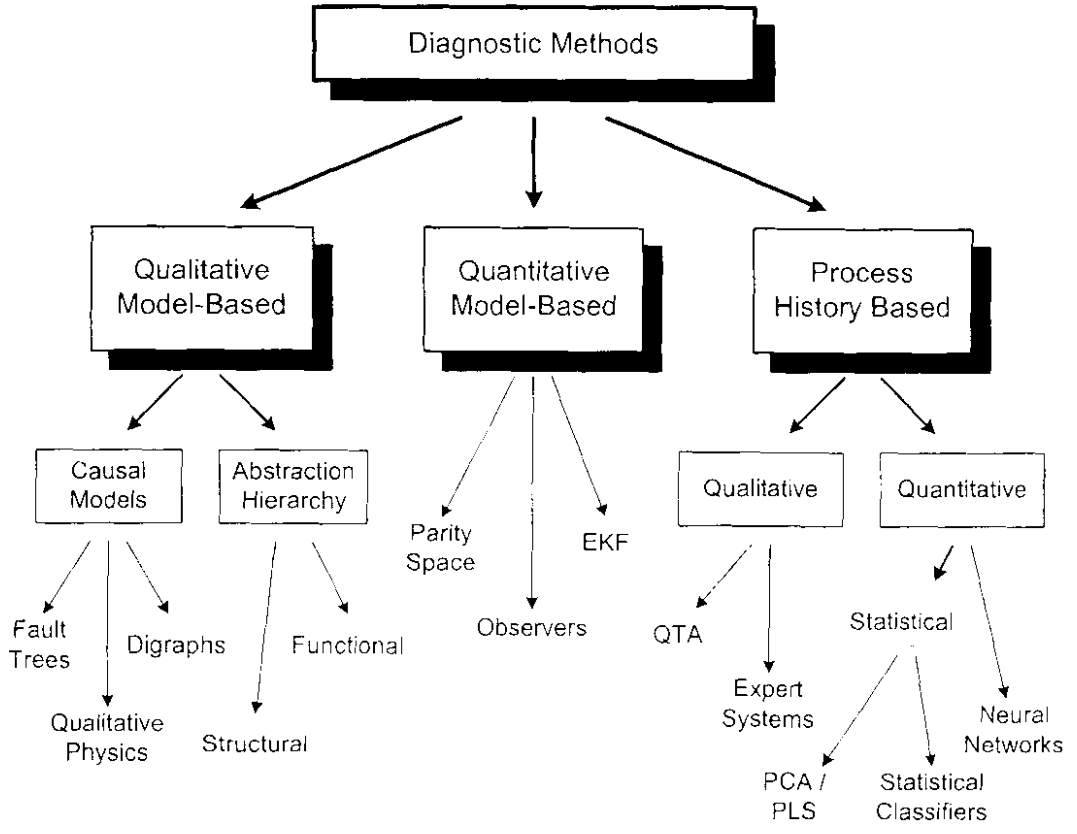


Figure 4. Tree structure of diagnostic methods [1]

Figure 4 serves as a summary for the rest of this chapter. It only includes the more popular and effective methods of process modelling and is by no means complete. Some methods can't be perfectly classified under just one of the three main branches. For example, neural networks approaches are a result of research in pattern recognition and are accordingly found under history-based methods; however, they are directly related to state-space models.

2.5 Quantitative model-based methods

Every modeled-based FDD system is built up from two steps. Firstly, the model is used to generate an expected result set of the physical process. These expected results are then compared with the actual results, measured by sensors, to check for inconsistencies. These inconsistencies are called the residuals and the method to obtain them is known as residual generation. The second step requires a rule set through which the residuals are sifted to isolate and identify faults.

The check for inconsistency needs some form of redundancy. There are two types of redundancies, viz. hardware redundancy and analytical redundancy. The former requires redundant sensors, which can become a costly exercise. Analytical redundancy is achieved from the functional dependence among the process variables and is usually provided by a set of algebraic or temporal relationships among the states, inputs and the outputs of the system.

The analytical redundancy schemes for fault diagnosis are basically signal processing techniques using state estimation, parameter estimation, adaptive filtering and other transformation methods. [1]. Both types of models, state-space or input-output, can be written as:

$$\mathbf{y}(t) = f\left(\mathbf{u}(t), \omega(t), \mathbf{x}(t), \theta(t)\right)$$

where $\mathbf{y}(t)$ and $\mathbf{u}(t)$ denote the measurable outputs and inputs, $\mathbf{x}(t)$ and $\omega(t)$ represent (mostly immeasurable) state variables and disturbance, and θ is the process parameters. Process faults usually cause changes in the state variables and/or changes in the model parameters.

Based on the process model, one can estimate the immeasurable $\mathbf{x}(t)$ or $\omega(t)$ by the observed $\mathbf{y}(t)$ and $\mathbf{u}(t)$, using state estimation and parameter estimation methods. Kalman filters [5] and observers [9] have been widely used for state estimation. Least squares methods provide a powerful tool by monitoring the parameter estimates online [6]. More recently, techniques relying on parity equations for residual generation have also been developed. [7] & [8]. Parity equations are obtained by rearranging or transforming the input-output models, which are relatively easy to generate from on-line process data and are easy to use. All of the popular residual generation methods are discussed in this section.

2.5.1 Diagnostic observers for dynamic systems

The main concern of observer-based FDD is the generation of a set of residuals from which different faults can be detected and uniquely diagnosed. These residuals should be robust in the sense that the decisions are not corrupted by such unknown inputs as unstructured uncertainties like process and measurement noise and modelling uncertainties. The method develops a set of observers, each one of which is sensitive to a subset of faults while insensitive to the remaining faults and the unknown inputs. The extra degrees of freedom resulting from measurement and model redundancy make it possible to build such observers. The basic idea is that in a fault-free case, the observers track the process closely and the residuals from the unknown inputs will be small. If a fault occurs, all observers which are made insensitive to the fault by design continue to develop small residuals that only reflect the unknown inputs. On the other hand, observers which are sensitive to the fault will deviate significantly from the process and result in residuals of large magnitude. The set of observers is so designed that the residuals from these observers result in a distinct residual pattern for each fault, which makes the fault isolation possible. Unique fault signature is guaranteed by design where the observers show complete fault decoupling and invariance to unknown disturbances while being independent of the fault modes and nature of disturbances. For a detailed discussion on general diagnostic observer design for linear systems, the reader is referred to Frank [9]. One important issue to be noted, as pointed out by Frank, is that the observer-based design does not need the application of state estimation theory, instead, only output estimators are needed which are generally realized as filters.

2.5.2 Parity relations

Parity (or consistency) equations are rearranged variants of the input-output or state-space models of the plant [7] & [8]. Primary residuals are formed as the difference between the actual plant outputs and those predicted by the model. These are then subjected to a linear transformation, to obtain the desired fault-detection and isolation properties. The design of parity relations amounts to finding the "residual generator" that satisfies the required response properties. Residuals are designed to enhance fault isolation, so that they exhibit directional or structural properties in response to particular faults. In addition, the residuals need to possess certain dynamic characteristics, for the desired transient behaviour and noise filtering. With parity relation design, these specifications are explicit, compensating only to satisfy causality and stability of the residual generator.

2.5.3 Extended Kalman filters

Kalman filtering is an established technology for dynamic system state estimation that is in common use in many fields including: target tracking, global positioning, dynamic systems control, navigation, and communication. [10]. The Kalman filter comprises a set of recursive equations that are repeatedly evaluated as the system operates. These equations will not be directly derived here; rather, one hopes that the following discussion will aid intuition into the method's workings. The reader is referred to the original reference papers [10] & [11] for further derivation details. Very generally, any causal dynamic system generates its outputs as some function of the past and present inputs. It is often also convenient to think of the system as having a state vector (which may not be directly measurable) where the state summarizes the effect of all past inputs on the system. Present system output may be computed with present input and present state only; past input values need not be stored.

It has been shown that a bank of Kalman filters designed on the basis of all the available possible system models under all possible changes can be used for the isolation purpose [10]. Fathi, Ramirez, and Korbiez [12] included adaptive analytical redundancy models in the diagnostic reasoning loop of knowledge based systems. The modified extended Kalman filter (EKF) is used in designing local detection filters in their work.

2.6 Qualitative model-based methods

An expert system is a computer program that mimics the cognitive behaviour of a human expert solving problems in a particular domain. [13]. It consists of a knowledge base, essentially a large set of if-then-else rules and an inference engine which searches through the knowledge base to derive conclusions from given facts. Also, the tree of these if-then-else clauses grows rapidly with the behavioural complexity of the system. The problem with this kind of knowledge representation is that it does not have any understanding of the underlying physics of the system, and therefore fails in cases where a new condition is encountered that is not defined in the knowledge base. Therefore, this kind of knowledge is referred to as 'shallow' since it does not have a deep, fundamental understanding of the system.

In symbolic reasoning, one often addresses three different kinds of reasoning. They are adductive, inductive and default reasonings. Adduction is the generation of a hypothetical explanation (or cause) for what has been observed. Unlike simple logical deduction, one can get more than one answer in adductive reasoning. Since there is no general way to decide between alternatives, the best one can do is to find a hypothesis that is most probable. Thus adduction can be thought of as reasoning where one weighs the evidence in the presence of uncertainty. Searching for the cause of an abnormality in a process system is thus an adductive reasoning. In addition, adduction also provides explanations of how the cause could have resulted in the abnormality observed. Such a facility is useful in providing decision support to plant operators. The use of knowledge representation matters a great deal in determining the computational effort. Model based reasoning allows for efficient bottom-up adduction by suggesting proper rules to check. The efficiency of such bottom-up search in adduction is considerable [14].

Early work in learning concentrated on systems for pattern classification and game playing. [14]. Inductive learning is the classification of a set of experiences into categories or concepts. Inductive learning is performed when one generalizes or specializes a concept definition learned so that it includes all experiences that belong to that concept and excludes those that do not. A clear definition of a concept or category is rarely simple because of the great variety of experiences and uncertainty

(noisy data or observations). For this reason, one prefers an adaptive learning scheme. An example of an adaptive learning scheme is failure-driven learning. Failure-driven learning is refining a concept from failures of expectations as one has related experiences. The failure of heuristic judgment in detecting a source of malfunction in fault diagnosis can trigger a change in the knowledge or rule that results in the judgment [15]. Experiences with abnormalities in a plant can be used to generate rules that relate a set of observations with specific causes. One can refine this experiential knowledge over time by generalizing to successful cases not covered and specializing when exceptions are noticed.

Frequently default assumptions are made on the values of various quantities being manipulated. This is done with the intention of allowing specific reasons for other values to override the current values, or for rejecting the default if it leads to an inconsistency. A fundamental feature of default reasoning is that it is non-monotonic. In traditional logic, once a fact is deduced, it is considered to remain true for the rest of the reasoning. This is what one means by monotonic. However, as new evidence arises, one often needs to revise the deduced facts to maintain logical consistency. Such a reasoning where retraction of deductions is allowed is non-monotonic. Default reasoning or non-monotonic reasoning is an invaluable tool in dealing with situations where all the information is not available at a time or if one has to reason about many, probably inconsistent, cases simultaneously.

The need for a reasoning tool which can qualitatively model a system, capture the causal structure of the system in a more profound manner than the conventional expert systems and yet be not as rigid in nature as numeric simulation led to the development of many methodologies to qualitatively represent knowledge, and to reason from them. In this section we will discuss these various forms of qualitative knowledge.

2.6.1 Digraphs based causal models

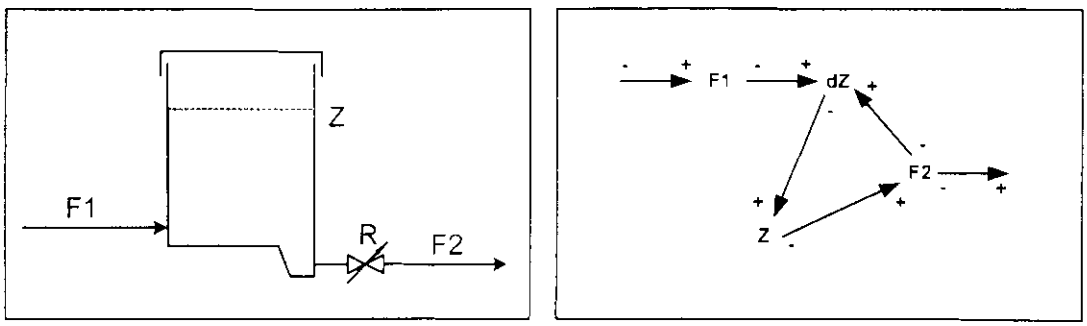
Diagnosis is, in a sense, the inverse of simulation. Simulation is concerned with the derivation of the behaviour of the process given its structural and functional aspects. Diagnosis, on the other hand, is concerned with deducing structure from the

behaviour. This kind of deduction needs reasoning about the cause and effect relationships in the process. In the evidential reasoning approach to diagnosis, heuristic information in the form of resultant rules is used. The underlying cause-effect relationships of the process are implicit in this form of reasoning. [15].

Cause-effect relations or models can be represented in the form of signed digraphs (SDG). A digraph is a graph with directed arcs between the nodes while SDG arcs have a positive or negative sign attached to them. Directed arcs lead from CAUSE nodes to EFFECT nodes. Each node corresponds to the deviation from the steady state of a variable. SDGs have nodes which represent events or variables and edges which represent the relationship between the nodes. They are much more compact than truth tables, decision tables, or finite state models. To understand digraphs, consider the tank of Figure 5.a. where F_1 is the inlet flow, F_2 is the outlet flow, and Z is the height of the liquid in a tank. The equations that represent this system are:

$$F_1 - F_2 = \frac{dZ}{dt}$$

$$F_2 = \frac{Z}{R}$$



a. Basic flow tank **b. Diagraph for a simple tank**

Figure 5. Signed digraph example

A corresponding digraph is given in Figure 5.b. The figure can be read as follows: an external change causes the flow rate F_1 to change; this causes a change in the liquid level in the tank (dZ and Z), this in turn causes the outlet flow rate F_2 to change and this in turn causes the liquid level to change (a feedback loop here). The signs in the arcs represent the direction of change. In a general situation, the arcs may be event dependent, i.e., the relationship between two events or variables may be dependent on other events or variables in the system. SDGs provide a very efficient way of

representing qualitative models graphically. There are mainly three kinds of nodes in a typical SDG representing a process:

- Those with only output arcs from them that represent basic fault variables which change independently;
- Those which have both input and output arcs, called process variables and
- Those with input arcs only, described as output variables. They do not affect any other variable.

SDGs have been the most widely used form of causal knowledge for process fault diagnosis. Hence this review describes the important contributions to the field of SDG representation. But first, definitions of terms used in digraph analysis are in order. A subset of a digraph is called a strongly connected component (SCC), if every node of can be reached from every other node of this subset. Maximal strongly connected component (MSCC) in a digraph is a SCC with no input arcs. Iri, Aoki, O'Shima, and Matsuyama [16] were the first to use SDG for fault diagnosis. SDG can be obtained either from the mathematical model of the underlying process or from the operational data (operator's experience). From SDG, they derive what is called a cause-effect graph (CE-graph). The CE-graph consists of only valid nodes (nodes which are abnormal) and consistent arcs. Consistent arcs are the arcs which potentially explain local propagation of the fault and hence the observed symptom or pattern. Only valid nodes are considered because nodes which are normal do not provide any path from sensor nodes to the fault nodes. Sign of nodes in a SDG constitutes a pattern. When the sign of some of the nodes is not known, then the pattern is called a partial pattern. In a typical process, all the process variables are not usually measured. When some of the nodes show abnormality, a CE graph with partial pattern (known as quasi-CE graph) is considered for diagnosis. The sign of the unmeasured nodes is assumed sequentially and the quasi-CE graph is expanded. All possible MSCCs are identified as potential fault nodes by propagation through the CE graph. When combinatorial search space for the sign of unmeasured nodes is exhausted, the diagnostic reasoning stops. There are some limitations imposed by difficulties in automation (or modularity) and the amount of quantitative information required in generating concealed equations. These types of confluences are sometimes called non-causal confluences because, in general, algebraic equations are unable to represent causality explicitly. [17].

2.6.2 Decision trees

Decision trees are used in analyzing the system reliability and safety. Fault tree analysis was originally developed at Bell Telephone Laboratories in 1961. Fault tree is a logic tree that propagates primary events or faults to the top level event or a hazard. The tree usually has layers of nodes. At each node different logic operations like AND and OR are performed for propagation. [18].

Decision trees are statistical models designed for supervised prediction problems. Supervised prediction is a generic term that encompasses many similar tasks such as predictive modelling, pattern recognition, multiple regression, multivariate function estimation, and supervised machine learning. In supervised prediction, a set of input variables (predictors) is used to predict the value of a target variable. The mapping of the inputs to the target is a predictive model. The data used to estimate a predictive model are a set of cases (observations, examples) consisting of values of the inputs and target. The fitted model is typically applied to new cases where the target is unknown.

A decision tree is so called because the predictive model can be represented in a tree-like structure. A decision tree is read from top-down starting at the root node. Each internal node represents a split based on the values of one of the inputs. The inputs can appear in any number of splits throughout the tree. Cases move down the branch that contains its input value. In a binary tree with interval inputs, each internal node is a simple inequality. A case moves left if the inequality is true and right if it is otherwise. The terminal nodes of the tree are called leaves. The leaves represent the predicted target. All cases reaching a particular leaf are given the same predicted value. When the target is categorical, the model is a called a classification tree. The leaves give the predicted class as well as the probability of class membership.

The leaves of the decision tree partition the input space into rectilinear regions. The predicted target has a different constant value in each partition. Consequently, the fitted regression-model is a multivariate step function. The surface is piecewise constant and not joined continuously at the boundaries. It is capable of modeling

nonlinear trends. A classification tree can be thought of as several multivariate step functions. Each function corresponds to the probability of a target class.

2.6.3 Qualitative physics

One approach in qualitative physics is the derivation of qualitative behaviour from the ordinary differential equations (ODEs). These qualitative behaviours for different failures can be used as a knowledge source. Sacks [19] examines piece-wise linear approximations of nonlinear differential equations through the use of a qualitative mathematical reasoner to deduce the qualitative properties of the system. Kuipers [20] predicts qualitative behaviour by using qualitative differential equations (QDEs) that are an abstraction of the ODEs that represent the state of the system. The goals of these methodologies are to reason from qualitative physical and equational descriptions to qualitative behavioural descriptions and to provide explanations of behaviour based on process observations and system description. The advantage of these qualitative simulators is their ability to yield partial conclusions from incomplete and often uncertain knowledge of the process. Each of the above theories start from a description of the physical mechanism, construct a model, and then use an algorithm so as to determine all of the behaviours of the system without precise knowledge of the parameters and functional relationships. De Kleer and Brown [21] emphasize modelling individual physical components and deriving the behaviour of a system of these components by using their connectivity to constrain the behaviour of the overall system. The qualitative simulation function (QSIM) as proposed by Kuipers [20] involves specifying a constraint model of the physical process in terms of qualitative versions of mathematical relationships such as addition, multiplication, and differentiation. The variables used in modelling the physical system should satisfy these qualitative mathematical constraints. The resulting structure represents a qualitative abstraction of an ODE, or a QDE that models the process. In terms of applications of qualitative models in fault diagnosis, QSIM and qualitative process theory (QPT) have been the popular approaches and these approaches will now be reviewed in some detail.

Conventionally, physical systems in science and engineering are modelled using differential equations, which are solved, either analytically or numerically to yield

functions that represent the system behaviour. Similarly, qualitative models represent an abstraction of the real physical system, and in terms of qualitative constraints, capture the information about the system. These qualitative models are 'solved' to get the qualitative behavioural description of the system. The QSIM representation and simulation algorithm allow one to reason mathematically about the description.

Qualitative simulation of a physical system by QSIM starts with a set of constraints modelling the structure of the process and its initial state and produces the visualization – a graph consisting of all the possible future states of the system. Every path from the node to the root in the graph corresponds to a possible behaviour of the system. The constraint model is a set of symbols representing the process variables, and a set of constraints on how these variables may be related to each other. The constraints allow one to express the simple mathematical relationships between the variables such as addition, multiplication and differentiation.

The fact that the variables in the qualitative simulation are continuously differentiable real-valued functions allows us to apply the mean value theorem, and restricts the possible transitions from a given qualitative description of state. The simulation starts with the initial state, generates all possible transitions that are allowed, and then employs the constraints to check which of the transitions are allowed by them. These transitions are then further filtered using global filters that detect whether a steady state has been reached, or a cyclic behaviour is attained. Thus the successor state is obtained. If the possible successor states are more than one, the simulation branches, and a tree of qualitative behavioural descriptions is obtained.

A powerful feature of the QSIM algorithm is the ability to reason about the dynamic behaviour of a system rather than about just the steady state behaviour. To generate the behavioural description, the QSIM algorithm requires the structural description of the system in terms of the set of qualitative constraints, and the initial state of the system.

2.7 Process history-based methods

All model-based FDD approaches require some form of before-hand information about the process. The previous sections showed that this *a priori* knowledge could be either quantitative or qualitative. In contrast to these model-based methods, process history-based methods are built from large sums of historical process data. Through feature extraction the historical data are transformed into some or other representation of *a priori* knowledge. This extraction process can be either qualitative or quantitative in nature. Two of the major methods that extract qualitative history information are the expert systems and trend modelling methods. Methods that extract quantitative information can be broadly classified as non-statistical or statistical methods. Neural networks are an important class of non-statistical classifiers. Principal component analysis (PCA), partial least squares (PLS) and statistical pattern classifiers form a major component of statistical feature extraction methods. [22]

2.7.1 Expert systems

Rule-based feature extraction has been widely used in expert systems for many applications. An expert system is generally a very specialized system that solves problems in a narrow domain of expertise. The main components in an expert system development include: knowledge acquisition, choice of knowledge representation, the coding of knowledge in a knowledge base, the development of inference procedures for diagnostic reasoning and the development of input-output interfaces. The main advantages in the development of expert systems for diagnostic problem-solving are: ease of development, transparent reasoning, the ability to reason under uncertainty, and the ability to provide explanations for the solutions provided. [13].

As an example of an expert system, consider fuzzy logic since it is conceptually easy to understand. Most of the development and employment with fuzzy logic was done in the early nineties when the need to incorporate linguistic information into control systems arose [23]. Fuzzy logic is based on natural language with sets of IF-THEN rules that can be built on the experience of experts. Further it is described as a flexible system that is tolerant of imprecise data. Fuzzy logic can model nonlinear functions of arbitrary complexity and thus is considered a universal approximator. [24].

The basic configuration of a fuzzy logic system is shown in Figure 6. Although the fuzzifier and defuzzifier are not part of a pure fuzzy logic system, hardly any real-valued application can be performed without it. The fuzzifier and defuzzifier maps crisp points to fuzzy sets and back again. One fuzzy set is an IF-THEN rule of the form:

$$R^{(l)}: \ IF \ x_1 \ is \ F_1^l \ and \cdots and \ x_n \ is \ F_n^l, \ THEN \ y \ is \ G^l$$

where F_i^l and G^l are fuzzy sets, $\bar{x} = (x_1,\ldots,x_n)^T \in U$ and $y \in V$ are input and output linguistic variables, respectively, and $l = 1,2,\ldots,M$. Each fuzzy IF-THEN rule defines fuzzy set $F_1^l \times \cdots \times F_n^l \rightarrow G^l$. These IF-THEN rules provide a convenient frame to incorporate human expert knowledge. Wang [23] describes numerous mathematical methods to import the rule base into the fuzzy inference engine of figure xx, the most popular being the sub-star composition. In short: within each fuzzy set the product of the input vector, $\bar{x}$, fires the output, y^l from G^l, the output of the inference engine is the sum or max of y^l, with $l = 1,2,\ldots,M$.

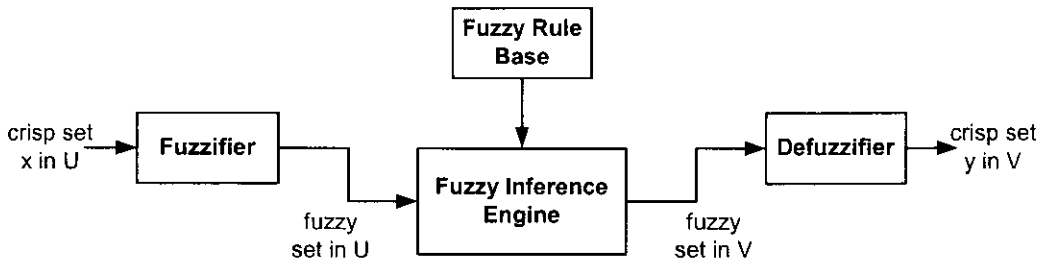


Figure 6. Basic configuration of a fuzzy logic system [23]

IF-THEN rules do not need to be formed from human expert data; various training algorithms have been developed for this purpose. [24]. Thus fuzzy logic can be considered as a process history based FDD method as well.

There are a number of other papers that discuss specific applications of expert systems for fault diagnosis. However, in all the applications, the limitations of an expert system approach are obvious. Knowledge-based systems developed from expert rules are very system specific, their representation power is quite limited, and they are difficult to update. The advantage though is the ease of development and transparent reasoning. [15].

2.7.2 Qualitative trend analysis (QTA)

A second approach to qualitative feature extraction is the abstraction of trend information. Trend analysis and predictions are important components of process monitoring and supervisory control. Trend modeling can be used to explain the various important events happening in the process, to do malfunction diagnostics and to predict future states. From a procedural perspective, in order to obtain a signal trend not too susceptible to momentary variations due to noise, some kind of filtering needs to be employed. For example, time series representations assume, beforehand, certain behaviour as they are identified using a known process behaviour. Alternatively, one may simply use a filter with specifically calculated filter coefficients (optimizing the required degree of smoothing). Filters suffer from the drawback that they cannot distinguish well between a transient and true instability [25]. The essential qualitative characters might be distorted by these filters. Avoiding this problem requires that the trend be viewed from different time scales or different levels of abstraction. Qualitative abstraction allows for a compact representation of the trend by representing only the significant events. For tasks such as diagnosis, qualitative trend representation often provides valuable information that facilitates reasoning about the process behaviour. In the majority of cases, process malfunctions leave a distinct trend in the sensors monitored. These distinct trends can be suitably utilized in identifying the underlying abnormality in the process. Thus, a suitable classification and analysis of process trends can detect the fault earlier and lead to quick control.

Multilevel abstraction of important events in a process trend is possible through scale-space filtering through the use of a bank of filters, each sensitive to certain localized region in the time-frequency domain [26]. There are two underlying ideas in the use of multilevel abstraction of process trends: (a) changes in trends occur at different scales and their optimal detection requires the use of filters or operators of different sizes, and (b) a sudden change will give rise to a peak or trough in the first derivative, or equivalently, to a zero-crossing in the second derivative. The interest in zero-crossings stems from the fact that they are very rich in information on the changes in a trend. This leads one to search for filters with two salient characteristics. First, it should be a differential operator, taking first, second or higher-order derivatives of the

function. Second, it should be capable of being tuned to any desirable scale, so that filters with small scale capture gradual change in the trend. An example of such a filter that has been extensively used in image processing is the Gaussian filter [26]. The main problem with a Gaussian filter is that the representation is highly redundant and the computational time for the filters might become prohibitive.

2.7.3 Statistical feature extraction from process data

In real process operations, one is faced with the problem of dealing with systems subject to random disturbances. In contrast to deterministic systems, the future state of stochastic systems is not completely determined by the past and present states and future control actions. The measurements are considered to be statistical time series – a single realization of an underlying stochastic process. Since the systems are under random influences, it is reasonable or sometimes necessary to formulate the systems in a probabilistic setting. When the process is under control, the observations signal probability distributions corresponding to the normal mode of operation. The underlying distributions change when the process is out of control. In general, probability distributions are characterized by their parameters when a parametric approach is used. For instance, if the underlying distribution of a monitored variable is normal, then the parameters of interest are the values of its mean and the standard deviation. Under faulty conditions, either the mean or the standard deviation may deviate from their nominal values. A composite change can occur as well. Accordingly, fault diagnosis can be stated as the problem of detecting changes in the parameters of a static or dynamic stochastic system. Basseville and Nikiforov [27] present the design of on-line change detection algorithms and an analysis of their performance under a unified framework. Both the Bayesian and the non-Bayesian approaches are discussed therein.

In on-line statistical approach, samples are taken sequentially and decisions are made based on the observations up to the current time. If the decision is made from the values of observations directly, observations $\mathbf{x}(t) = [x_1(t), x_2(t), \ldots, x_n(t)]' \in R_n$ where R_n is the so-called stopping region in statistics, it is concluded that there is a change in

the process. More often a statistic **g**(t), a function of observation x(t), is designed and the decision is made according to the comparison of **g**(t) with some threshold value **c**. This idea can be translated into a 'stopping rule' problem with a standard form:

$$\tau = \inf\{t \geq 1; \mathbf{g}(t) \geq \mathbf{c}\}.$$

τ is the greatest lower bound, i.e., the first time when **g**(t) is greater than **c**. In on-line detection, it is desirable to detect the change as soon as it occurs, i.e., to detect it at the first time when $\mathbf{x}(t) \in R_n$ or $\mathbf{g}(t) \geq \mathbf{c}$. Thus, the fault (change) detection amounts to the design of the trigger rules or proper choices of statistic g(t) and threshold c. Obviously a good detector must be sensitive to change. However, the sensitivity to the process noise usually increases along with the sensitivity to real change. In other words, as the failure and delay of the detection decrease, the number of false alarms tend to increase, which is certainly undesirable. A good design is usually defined as one which will minimize the failure and delay at a fixed false alarm rate.

Quality control represents one of the earliest attempts of using statistics in on-line monitoring and change detection. Control charts approach is based on the assumption that a process subject to its natural variability will remain in a state of statistical control under which certain process variables remain close to their desired values. Therefore, by monitoring the performance of a process over time, abnormal events can be detected as soon as they occur. If the causes of such events can be diagnosed and the problem can be corrected, the process is driven back to its normal operation.

2.7.4 Neural networks

Fault detection and diagnosis using neural networks are achieved by exploiting their non-linear pattern classification properties. There are certain requirements that a neural net-based classifier has to meet for successful fault diagnosis applications. When the measurement patterns are presented, the classifier should be capable of providing one of the following decisions [29]:

- **Normal:** From a definition of normal process operation, through the specification of training patterns corresponding to the normal process behaviour, the network should be able to determine that no fault has occurred.

This is essential to avoid unnecessary alarms due to process and measurement noise.

- **Abnormal and known fault**: If the observation pattern falls in the proximity of the training patterns of a particular fault class, then that fault should be presented.
- **Abnormal and unknown fault**: When the observation pattern falls far from the training patterns, the network should be able to say that abnormal behaviour has occurred but that it cannot determine the class to which the pattern belongs. This is crucial because it is not possible to determine beforehand all the faults that are possible in the process plant. This guarantees that the network would not announce a known fault class, when the actual fault that occurred is an unknown fault class.

Thus, the network should be able to perform reasonable generalizations when the input patterns are in the proximity of known fault classes or the normal region, and avoid faulty generalizations when the measurement patterns fall far outside the known fault classes. [28]

The diagnostic ability of a neural net depends upon the discrimination of decision regions corresponding to various fault classes in the measurement space. A fault space is defined as the region spanned by the measurement data from the various sensors in a process. It turns out that in most fault diagnosis problems, the fault classes form tightly bounded regions in the measurement space. The activation functions in the input layer carve the measurement space into various regions and make them correspond to the different fault classes. The decision regions do not tightly enclose the fault classes, thus leading to erroneous generalizations when the input patterns are far away from the training patterns. Thus, the classification is not robust. [29].

In chapter 4 of this dissertation neural network theory is discussed and used. Back-propagation neural networks are used to model some systems and, in combination with observers described in section 2.5.1, form the backbone of an FDD system.

2.8 References in Chapter 2

[1] Venkatasubramanian, V. et al. *A review of process fault detection and diagnosis Part I: Quantitative model-based methods.* 2003. Computers and Chemical Engineering. 27. 293-311. Elsevier press.

[2] Himmelblau, D. M. *Fault detection and diagnosis in chemical and petrochemical processes.* 1978. Amsterdam: Elsevier press.

[3] Isermann, R. *Supervision, fault-detection and fault-diagnosismethods - an introduction.* 1997. Pergamon.

[4] Milne, R. *Strategies for diagnosis.* 1987. IEEE Transactions on System, Man and Cyber 17 (3), 333-339.

[5] Frank, P. M., & Wünnenberg, J. *Robust fault diagnosis using unknown input observer schemes.* 1989. In R. J. Patton, P. M. Frank & R. N. Clark (Eds.), *Fault diagnosis in dynamic systems: theory and applications.* NY: Prentice Hall.

[6] Isermann, R. *Process fault diagnosis based on dynamic models and parameter estimation methods.* 1989. In R. J. Patton, P. M. Frank & R. N. Clark (Eds.), *Fault diagnosis in dynamic systems: theory and applications.* NY: Prentice Hall.

[7] Gertler, J. *Analytical redundancy methods in fault detection and isolation.*1991. In Proceedings of IFAC/IAMCS symposium on safe process (p. 91), Baden-Baden.

[8] Gertler, J., & Singer, D. *A new structural framework for parity equation-based failure detection and isolation.* 1990. Automatica 26, 381-388.

[9] Frank, P. M. *On-line fault detection in uncertain nonlinear systems using diagnostic observers: a survey.* 1994. International Journal Systems Science 25 (12), 2129-2154.

[10] Basseville, M. *Detecting changes in signals and systems – a survey.* 1988. Automatica 24, 309-326.

[11] Willsky, A. S. *A survey of design methods for failure detection in dynamic systems.*1976. Automatica 12, 601-611.

[12] Fathi, Z., Ramirez, W. F., & Korbiez, J. *Analytical and knowledge-based redundancy for fault diagnosis in process plants.* 1993. AIChE J. 39, 42-56.

[13] Venkatasubramanian, V. et al. *A review of process fault detection and diagnosis Part II: Qualitative models and search strategies.* 2003. Computers and Chemical Engineering. 27. 313-326. Elsevier press.

[14] Charniak, E., & McDermott, D. *Introduction to artificial intelligence.* 1984. Massechusetts: Addison-Wesley Publishing Company.

[15] Rich, S. H., & Venkatasubramanian, V. *Causality-based failure-driven learning in diagnostic expert systems.* 1989. American Institute of Chemical Engineers Journal 35 (6), 943-950.

[16] Iri, M., Aoki, K., O'Shima, E., & Matsuyama, H. *An algorithm for diagnosis of system failures in the chemical process.* 1979. Computers and Chemical Engineering 3 (1-4), 489-493.

[17] Oyeleye, O. O., & Kramer, M. A. *Qualitative simulation of chemical process systems: steady state analysis.* 1988. American Institute of Chemical Engineers Journal 34 (9), 1441-1454.

[18] Potts, J. E. *Decision Tree Modelling Course Notes.* 2001. SAS Institute Inc.

[19] Sacks, E. *Qualitative analysis of piecewise linear approximation.* 1988. Journal of Artificial Intelligence in Engineering 3 (3), 151-155.

[20] Kuipers, B. *Qualitative simulation.* 1986. Artificial Intelligence 29 (3), 289-338.

[21] De Kleer, J., & Brown, S. *A qualitative physics based on confluences.* 1984. Artificial Intelligence 24 (1-3), 7-83.

[22] Venkatasubramanian, V. et al. *A review of process fault detection and diagnosis Part III: Process history based methods.* 2003. Computers and Chemical Engineering. 27. 327-346. Elsevier press.

[23] Wang, L. *Adaptive fuzzy systems and control.* 200x. Prentice Hall.

[24] MATLAB® Help, *Matlab® fuzzy logic toolbox.* 2002. The MathWorks Inc.

[25] Gertler, J. *Intelligent supervisory control.* In A. E. Nisenfeld & J. R. Davis (Eds.), 1989. Artificial intelligence handbook, Volume II. Research Triangle Park, NC: USA.

[26] Marr, D., & Hildreth, E. *Theory of edge detection.* 1980. Proceedings of Royal Society London 207, 187-217.

[27] Basseville, M., & Nikiforov, T. V. *Detection of abrupt Changes – theory and application.* 1993. Information and system sciences series . Prentice Hall.

[28] Jackson, J. E. *A user's guide to principal components*. 1991. New York: Wiley-Interscience.

[29] Cybenko, G. *Approximation by superpositions of a sigmoidal function*. 1989. Mathematics of Control, Signals and Systems 2, 303-314.

CHAPTER 3

3 HEATING, VENTILATING AND AIR-CONDITIONING

The goal of chapter 3 is to provide the reader with an overview of heating, ventilating and air conditioning (HVAC). This includes a list of the common faults encountered in HVAC equipment and different techniques to detect and diagnose these malfunctions.

It is not the intention of this research to investigate all possible faults, but rather to provide principals which may be applied to isolate and identify a wide variety of faults. To achieve this generic transfer functions will be used as well as some simulated HVAC systems.

From first principals a model of a single-loop air control system is built. The purpose of this model is to introduce failures on which FID can be performed. FID strategies on HVAC equipment are studied and the focus for the remainder of the dissertation is set on artificial neural networks (ANN).

3.1 HVAC fundamentals

The main purpose of the commercial HVAC (heating, ventilating, and air conditioning) system is to provide the people working inside buildings with '*conditioned*' air so that they will have a comfortable and safe work environment. *Conditioned* air means that air is clean and odorless and the temperature, humidity, and movement of the air are within certain comfort ranges. [1]

Due to the warm climate in South Africa and the target of this dissertation, these sections focus mainly on the cooling aspect of HVAC. In this particular study telecommunication exchange equipment needs to be operated in a controlled climate inside the exchange building. Starting from the fundamentals, the text leads to the modeling of a basic air conditioning system.

3.1.1 Fundamental cooling cycle

Any cooling system consists of at least three basic blocks. Figure 7 is a schematic representation of an elementary cooling system. [1]. Practically, these functions can be implemented by various mechanical methods.

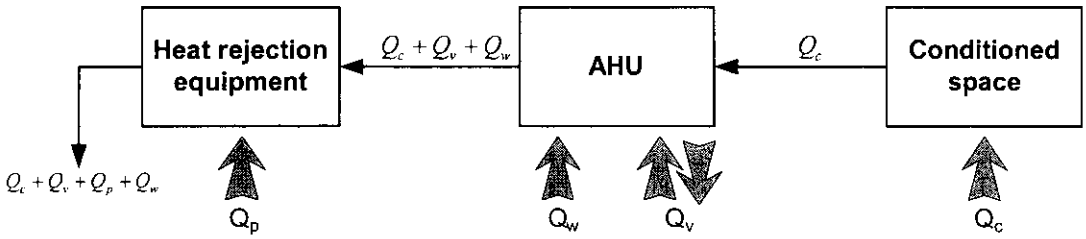


Figure 7. Mechanical cooling cycle

The cooling load Q_c in the conditioned space is a combination of internal and external loads, such as equipment, people, lights and solar heating. This is usually removed by circulating air through the space, where the entering air has a lower temperature and humidity than the desired space condition. The air offsets the cooling load by increasing its temperature and humidity to equal that of the space and then returns it to the air-handling unit (AHU), where it is re-cooled and dehumidified. Though dehumidification is not an essential part of the cooling cycle, it usually occurs. Most spaces require some ventilation (outside) air, which is mixed with the return air at the AHU. This imposes an additional cooling load Q_v. If the outside enthalpy is less than

the space enthalpy, Q_v will be negative and the cycle gains some "free cooling". Work energy Q_w is required to circulate the air. This work, usually an electric motor driven fan, becomes part of the cooling load. The heat rejection equipment, mostly a refrigeration system, introduces an additional load Q_p. Q_p is mainly a build-up from pumping fluids, driving refrigeration machines and condensers or cooling towers. Thus, the target of the heat rejection equipment is to remove the total load, represented by $Q_c + Q_v + Q_w + Q_p$. These work portions are parasitical. They contribute additional heat which must be removed, reducing the overall system efficiency. Ultimately all this heat energy is dumped in a heat sink –sometimes water, most often atmospheric air.

3.1.2 Mechanical two-phase refrigeration cycle

A refrigeration cycle is a means of transferring heat from some place where it is not wanted (heat source) to another place where it can be disposed of (heat sink). The necessary components of this cycle are [1]:

- At least two heat exchangers, one for the source and one for the sink
- A refrigerant
- A conduit for conveying the refrigerant
- Energy to move the heat through the system

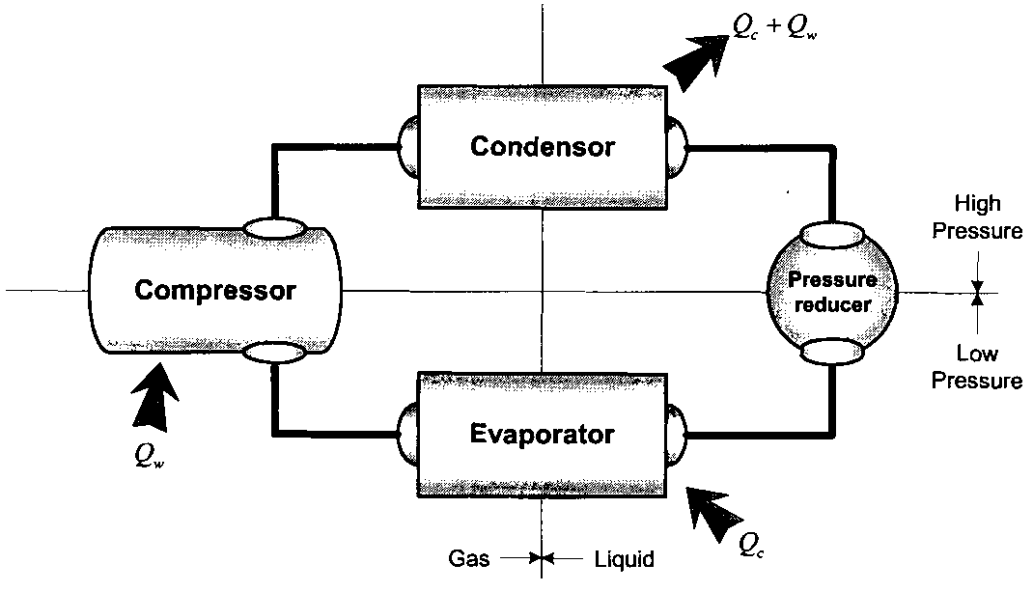


Figure 8. Two-phase mechanical refrigeration cycle [1]

The most common cooling source in HVAC is mechanical two-phase refrigeration. In such a cycle, presented in Figure 8 a compressor is used to raise the pressure of the refrigerant gas. This requires work energy Q_w from an electric motor and increase the temperature of the gas. At the condenser the heat is transferred to a heat sink when the refrigerant is liquefied. The high pressure liquid is passed through a pressure reducing device to the evaporator. At a lower pressure the liquid evaporates, removing the heat of vaporization from its surroundings. This qualifies the evaporator as the heat source Q_c in the cycle. The cold vapour is then returned to the compressor to be recycled. Thus the heat removed in the condenser is the sum of $Q_c + Q_w$. To measure the effectiveness of a two-phase refrigerator we use its coefficient of performance given by Q_c/Q_w.

3.1.3 Chillers

The previous section described the interactions between the different components of the common cooling machine. In practice, the term, chiller, is used to describe the complete package that includes a compressor, condenser, evaporator, internal piping and controls. In the case of liquid chillers, the evaporator is by far the largest component and is often referred to as the chiller. These chillers consist of a shell, which houses the water or brine, and some tubes through which the refrigerant run. The chilled water or brine is pumped through to the cooling coils in the air handling units (AHU) [1]. The AHU is described in the following section.

An elementary chiller plant with multiple cooling coils is shown in Figure 9. According to Heines [1], this is by far the most sufficient design as chillers require a constant flow rate in the brine. Once the work load becomes too large for a single chiller, more chillers are placed in parallel. This calls for additional strainers and pumps in the pipe work to ensure that each chiller is supplied with the needed flow. As more components are added, more gaps for energy loss are created.

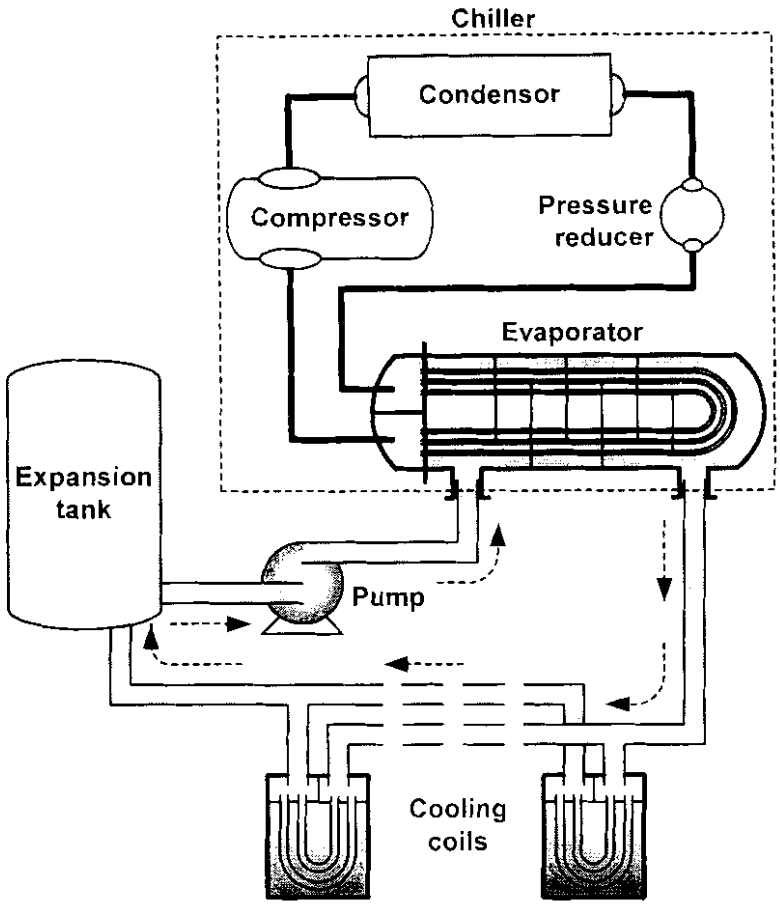


Figure 9. Chiller plant with multiple AHU coils [1]

3.1.4 A simple air conditioning system

With most HVAC systems the goal is to maintain some volume at preferred comfort level. This comfort is influenced by numerous factors, including the temperature, humidity and airflow through the volume. The volume, whether it is a room or a whole building, is considered a pressurized space since fresh air is pumped in to replace the old air. Usually, the air removed from the space is considered to be more comfortable than the outside air, thus 90% is recycled through the air conditioning system. The hardware responsible for this process is an air handling unit (AHU) as shown in Figure 10.

The AHU of Figure 10 receives a mixture of outside air and recycled air. The air is filtered and sucked through either a heating or cooling coil depending on the temperature requirements. The supply fan thereafter blows the conditioned air into the pressurized space. [1].

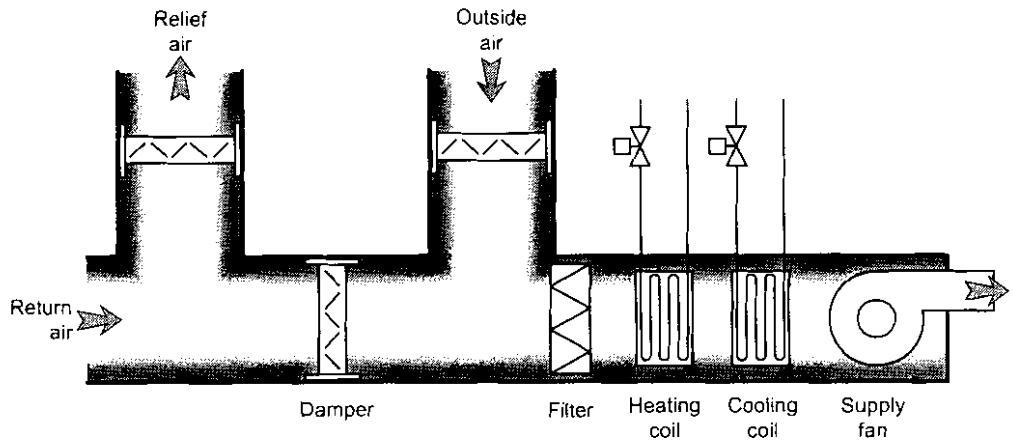


Figure 10. Air handling unit [1]

In the case where the AHU supplies more than one room, as in Figure 11, the conditioned air is transferred via a ducting network. To serve each room equally, the pressure through out the ducting should be the same. Pressure is regulated by controlled dampers within the ducting. In the following section the controlling of these dampers are discussed.

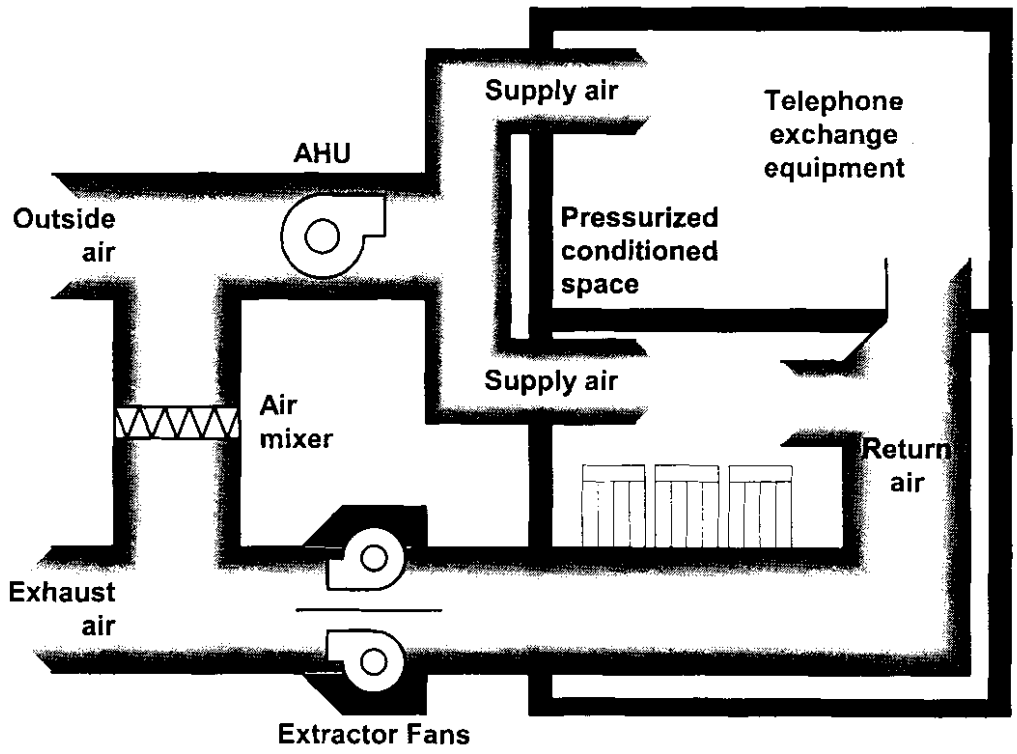


Figure 11. Schematic of an air movement unit [1]

3.2 Qualitative physics models of HVAC equipment

In section 2.6.3 qualitative physical models have been discussed. The fundamental assumption behind model based reasoning, according to Throop [2], is the notion that if the model is correct, all the discrepancies between the prediction and the observed outcome arise from defects inside the equipment. He continues that each fault in the mechanism corresponds to some fault in an individual component.

In this section, the focus is on how some HVAC elements are broken down into physical models and are described with equations. Some simulation tools and programs on this topic are available in a variety of literature. Most of these, like Throop [2], model HVAC elements with first order differential equations or linear equations.

H-Tools is a library of Matlab Simulink models, specially constructed for the thermal system analysis in building physics. In the H-tools manual [3] there are two main models, representing the most frequent objects of interest: a construction (wall or window) as a layered structure of different building materials, and a zone (ventilated space), which is enclosed by the building envelope. Construction models provide detailed calculations of the thermal state of each component in the structure, according to the surrounding boundary conditions. The thermal state of the zone is determined by the heat gains through the building envelope and through applied HVAC systems and internal gains, additional components presented by the models of the same name. Direct coupling between the construction element and the surrounding air (outdoors/indoors climate, ventilated space) is accomplished by outdoor and indoor surface conditions – models for heat balance at the wall boundaries.

Using H-tools requires that each different material item be supplied with its own block. This means, for example, that a basic wall is built from 3 blocks, the outer concrete layer, the brick layer and the inner concrete layer. Simulating a whole building could require an extensive amount of block. But, as all of the components in H-tools – referred to in the documentation – are built from first order or linear equations, the underlying mathematics remains feasible.

Athienitis [4] built a standard model structure for control loops. Although this structure (Figure 12) is mainly used for thermal analysis in buildings, it can be easily exported to other HVAC systems as the example below will illustrate.

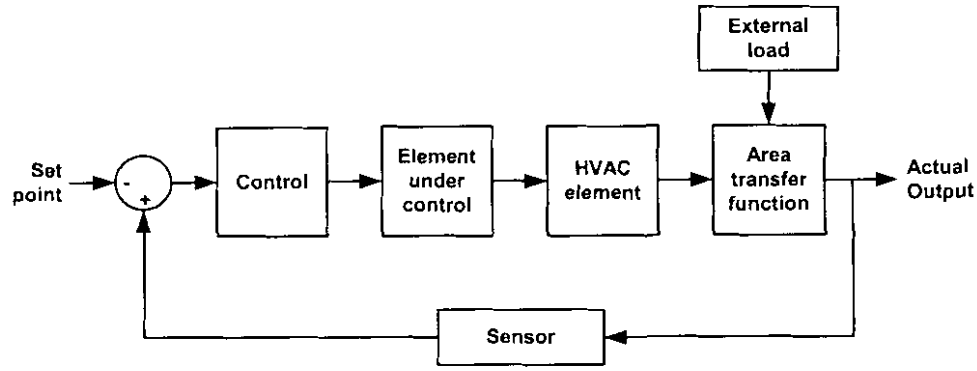


Figure 12. Standerd control loop for HVAC systems [4]

In his paper, Athienitis built equations to model temperature losses through the walls to supply the area transfer block of Figure 12. This was needed to calculate the controllers' parameters. He then states that controller tuning are usually done on-site, for most system parameters such as thermal losses are considered unknown.

3.2.1 Example of a single-loop air control

Figure 14 is a break down of a single-loop air control system of Figure 13, into the sub-systems that interact to maintain the down-stream pressure. The physical components, for instance the damper actuator, are each modelled individually from ground principals.

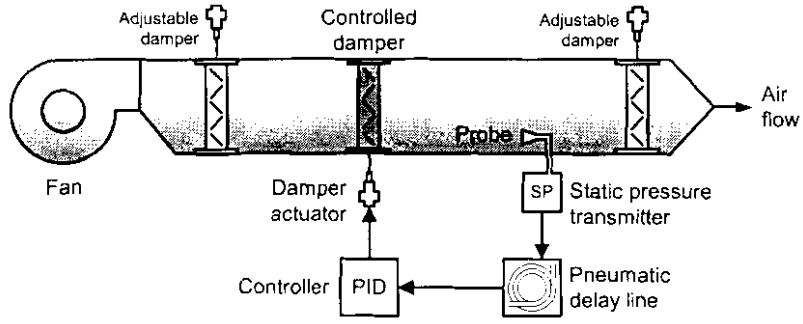


Figure 13. Static pressure control loop

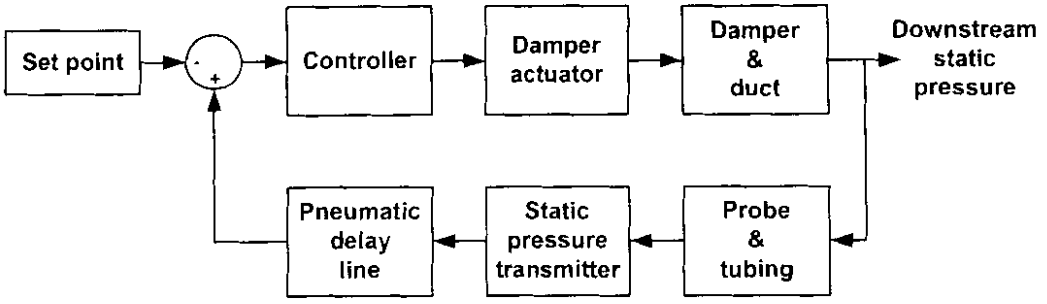


Figure 14. Block diagram of a single-loop air control system

No information on the interactions between components is included in these low-level models. The different components can be linked together to complete the model of the whole system. [4]

Watton, Marcks and Solem [6] have built the model of Figure 14 into matlab simulink. Within the simulink model of Figure 15 the physical relations of the individual components of Figure 14 can be seen. Further into this dissertation, the simulink model is used to capture failure data for fault identification and diagnosis. Notice here as well that all the components are modelled with 1st order transfer functions.

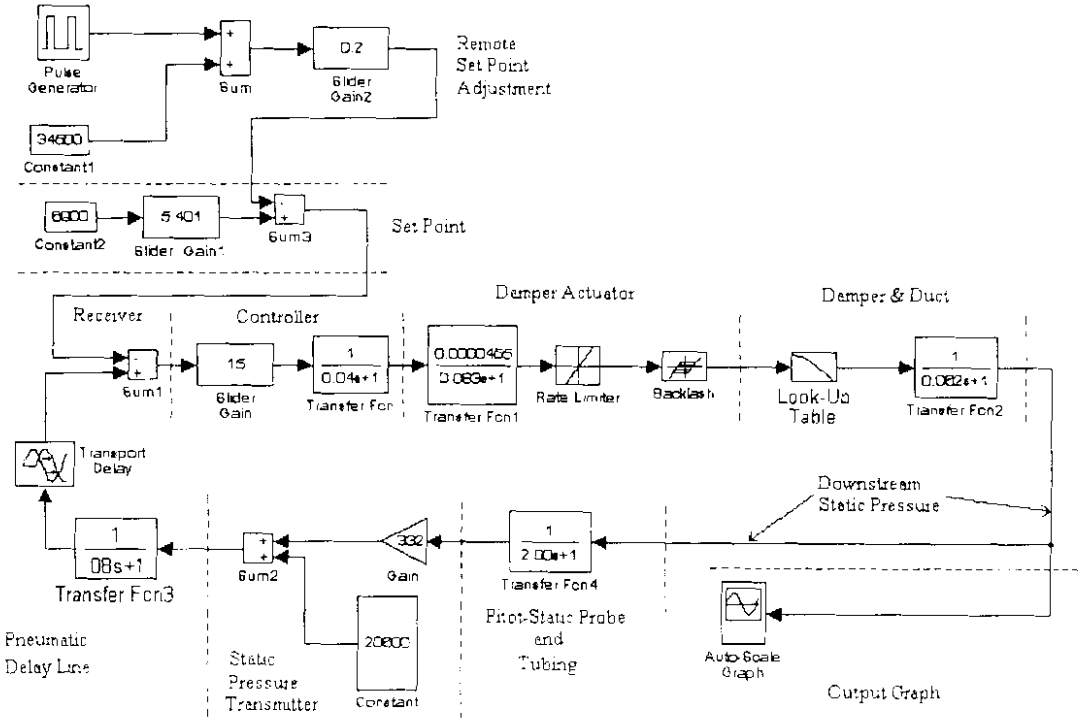


Figure 15. Matlab simulink model of an air controlled system [6]

3.3 Fault detection and diagnosis on HVAC

Isermann [7] presents the application of FDD techniques as a series of four steps termed "process supervision", which is a good reference model for describing many of the FDD methods that have been developed for cooling equipment. Figure 16 shows the four-step process applied to HVAC equipment. The first step is fault detection, in which a fault is indicated when the performance of a monitored system has deviated from expectation. The second step, diagnosis, determines which malfunctioning component is causing the fault. Following diagnosis, fault evaluation assesses the impact of the fault on system performance. Finally, a decision is made on how to react to the fault. This is usually a choice between tolerating the fault, repairing it as soon as possible, adapting the control, or stopping operation until repair is complete.

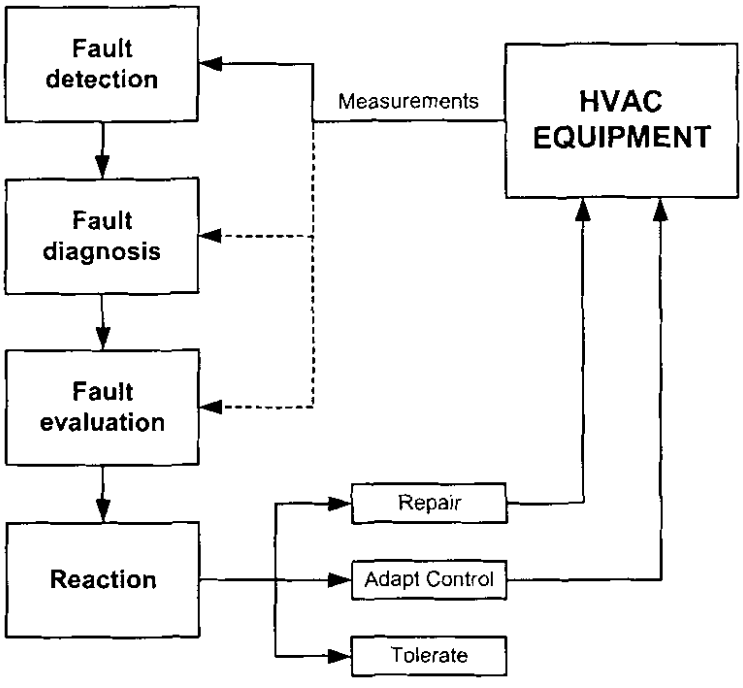


Figure 16. Supervision of HVAC equipment

In each of these steps, it is necessary to define criteria or thresholds for establishing appropriate outputs. The outputs would be 'fault' or 'no fault' for fault detection, the type of fault for diagnosis, and 'repair' or 'do not repair' for the fault evaluation step. In some methods, fault detection and diagnostics are combined into one step. In this case, the diagnostic tool includes "normal operation" as one of the outputs in addition to all the fault possibilities.

3.3.1 Fault Detection

Fault detection is accomplished by comparing performance determined from measurements with some expectation of performance. If the deviation exceeds a threshold, then a fault is indicated. Often this process is divided into two steps: pre-processing and classification. The pre-processor takes measurements from sensors and manipulates them to generate features for classification. Classifiers then operate on the features to determine whether the system contains a fault. [8]

Simple transformations, characteristic quantities, and models are three types of pre-processors that have been employed. Simple transformations involve manipulations of the raw data, such as trend generation (i.e., time derivatives). Characteristic quantities are features that are computed directly from measurements and are indicative of component performance. Examples include overall system efficiencies and heat exchanger effectiveness. Model-based pre-processors utilize mathematical models of the monitored system to generate features. Model parameters could be learned from measurements when the system is operating normally, or determined using physical models. [8]

The features used by the fault detection classifier from a model-based pre-processor could be the differences between measured and modelled performance, physical parameters of the model (e.g., air-side heat transfer conductance), or characteristic quantities that have some dependence on inputs (e.g., compressor efficiency). Both physical and black box models have been used for FDD. A black box model provides a mapping between inputs and outputs of a process or system, but does not take into account any known physics. Black box models can provide excellent mappings to nonlinear behaviour, but require much more data for training than models based upon physical laws and may not extrapolate well. Examples of black box models include artificial neural networks, polynomials, and autoregressive (AR) models. [9]

In a broad sense, the fault detection classifier is an expert system. The knowledge necessary to make a fault decision can be stored in a number of forms, including a set of production rules (i.e. IF-THEN, ELSE rules), a fault tree, and conditional probabilities for statistical pattern recognition classifiers. Typically, it is necessary to

assign the thresholds for deviations between current and normal performance that constitutes faults. In selecting thresholds, there is a trade-off between detection sensitivities and false alarm rates as mentioned in section 2.3.1. Thresholds are often determined based upon heuristics, although better performance (lower ratio of false alarms to correct diagnoses) is achieved when statistical thresholds are employed. [9]

In general, pre-processing simplifies the classification and improves overall performance of the FDD system. In the absence of any pre-processing, the FDD system is a classic expert system. All fault detection is then based upon rules that act directly on the measurements. For instance, a chiller application might use condenser head pressure as an indication of a fault. Without pre-processing, the head pressure would be compared with a fixed maximum value to indicate a fault. However, since the head pressure varies under normal operation with the entering condenser water temperature, the fault detection threshold must be greater than the highest head pressure associated with normal operation. A more complex expert system might contain a set of rules with different head pressure limits for different condenser water temperatures. [8]

Alternatively, a model-based pre-processor could model the relationship between head pressure and condenser water (or air) inlet temperature under normal operation. Then, a fault would be identified if the deviations between measured and modelled head pressures exceed a specified threshold. The FDD system with the model-based pre-processor can be significantly more sensitive to abnormal behaviour than the single rule system and easier to implement than the expert system with many rules. The thresholds for allowable deviations can be established by evaluating the statistical properties of the measurements, and how well the model for normal operation fits the measurements. [8]

3.3.2 Fault Diagnosis

The structure of pre-processing and classification can also be used to describe fault diagnosis. Measurements are processed in order to simplify the classification required to identify the particular component at fault. The overall classification problem is different for fault diagnosis than for fault detection in that the decision is not binary

(i.e., fault/ no fault): the classifier must choose the specific fault from a list of possibilities. However, the diagnostics problem can be reduced to a series of fault detection problems through fault isolation.

With fault isolation, fault detection methods are applied to individual components for which diagnoses are desired. For instance, condenser fouling in an air conditioner could be detected by estimating the heat exchanger effectiveness from measurements on the condenser. The fault is diagnosed as soon as it is detected and no additional classification is necessary. The disadvantage of fault isolation is the large number of measurements required. The diagnosis of heat exchanger fouling would require measurements of all states entering and leaving the heat exchanger. [7]

Another diagnostic approach involves comparing physical parameters determined from measurements with values representative of normal operation. For instance, heat exchanger conductance could be estimated from entering and leaving conditions and used to diagnose fouling [1]. Here again, fault detection and diagnosis are combined and no separate diagnostic classification is necessary.

A more common diagnostic approach that requires fewer measurements involves the use of fault models. For each type of fault to be diagnosed, a fault model predicts the outputs associated with the occurrence of that fault for a current set of inputs. The fault is diagnosed through the use of a classifier that attempts to find the fault model with the best representation for the current behaviour. The advantage of fault modelling for diagnosis is that fewer measurements are required. However, it is necessary to have fault models for each fault and combinations of faults to be diagnosed. Statistical pattern recognition techniques are often employed for finding the best matching fault model. However, when sufficient data are available for training, neural network and other black box approaches can learn patterns for normal and faulty behaviour and provide direct classification of raw measurements. [7]

3.3.3 Fault Evaluation and reaction

Fault evaluation follows fault detection and diagnosis and requires an evaluation of the impact of a fault on system performance. Without this step, the fault must become

obvious enough to justify the expense of servicing the unit. This is the case for many "hard" failures, such as broken fan belts or seized compressors. However, fault evaluation is necessary for many performance degradations, such as heat exchanger fouling, where the fault could be detected and diagnosed well before the need for service. [7]

This dissertation focuses on the detection and identification of faults. The evaluation of faults and the maintenance reaction on it remains circumstantial and is difficult to discuss in broad terms. It largely depends on the company policy and the availability of redundancy in the form of standby units.

3.4 FID with neural networks on HVAC equipment

In this section the aim is to motivate that neural networks, especially the multi layer perceptron, would provide a suitable solution to the detection and identification of failures on chillers and other HVAC equipment. By no means would it be the only solution. It is, however, necessary to base its worthiness on some reasoning.

First this text considers past successes of the applied method, in comparison with that of others. From this one can also learn what to expect of the results in testing of the theory.

3.4.1 Literature on chiller FID with multi layer perceptron

Margaret Bailey [10] wrote a thesis on a neural network FID program. In her research the multi layer perceptron (MLP) network was trained, using chiller operating data collected during both normal and abnormal operating conditions. The faults introduced included refrigerant loss and overcharge, oil loss and overcharge, air-cooled condenser fouling, and the loss of an air-cooled condenser fan. These failures were chosen as a result of surveys conducted with a chiller manufacturer and chiller service contractor.

Neural networks are particularly suited for modelling highly non-linear processes and are also known for their ability to ignore excess data that are of minimal importance and concentrate on more crucial input data.

Inputs to this FID program were mainly temperature measurements through the cooling circuit. These included: chilled water temperature, refrigerant condensing and evaporation temperatures, and superheat and sub-cooling temperatures. Power consumptions as well as discharge pressure were also input measurements.

Two independent variables, fault degree and chiller load, were varied to study the following dependent variables: energy consumption, chilled water supply temperature, superheat and sub-cooling temperatures, suction pressure and discharge pressure. Another approach considered the effects of outdoor air temperature (an uncontrollable independent variable) and chiller load on the same dependent variables. A multiple linear regression model and a non-linear ANN model were used to verify that all the independent variables were important in determining the dependent variables. Moreover, it showed a relation between outdoor temperature and chiller loading.

Table 1. Results of testing networks with training, validation and testing data sets [10]

ANN Configuration	Misclassification Rate		
	Training Data	Validation Data	Test Data
network1	7%	35%	69%
network2	66%	60%	82%
network3	11%	44%	76%
network4	17%	40%	44%
network5	13%	36%	84%
network8	3%	27%	58%
network9	1%	6%	28%
network10	0%	12%	56%
network11	0%	9%	36%
network12	0%	3%	20%
network13	0%	9%	47%
network14	28%	-	-
network15	31%	-	-
network16	2%	14%	40%
network17	0%	10%	28%
network19	5%	21%	42%

A study was performed to check the accuracy of the NN via a parametric study. 19 different training programs were used, and among these the best true performance was

training file 12 with a 0% misclassification rate of the training data, a 3% misclassification rate of the validation data, and a 20% misclassification rate of the test data. However, this particular training set achieved superior results by not using any fan loss data. The results from Bailey's work are shown in Table 1.

3.4.2 Artificial neural networks

Lee [11] programmed ANN into a tool which calculates and normalizes residuals for the following components under steady-state operating conditions:

- Supply air temperature
- Cooling coil valve position
- Cooling coil control signal
- Supply fan speed
- Return fan speed
- Supply duct static pressure
- Volumetric flow difference

Failures were identified successfully in a laboratory setting when residual values exceeded a three-sigma (standard deviation) threshold. Since the set point value of the cooling coil valve is undefined, a different approach for the calculation of the associated residual value is necessary. One option is to use the average and standard deviation of the previous control signals (e.g., the last 20 time steps) as the reference values. This works for detecting sudden, abrupt failures in systems with slowly varying loads. Another option is to use a model of the cooling coil to predict what the control signal should be, given the current operating conditions. This method allows for the detection of degradation faults, such as coil fouling, but requires the development of a model.

Work on this tool was continued by the authors to include fault diagnosis capabilities through the use of artificial neural networks (ANN); however, this work resulted in a tool that was specifically tailored to the laboratory environment under which it was developed. While the results were promising, success in a real building environment was not realized due to the time and cost-prohibitive steps necessary to implement the tool outside of the laboratory.

3.5 HVAC FID with other classifiers

3.5.1 K-Nearest neighbours and mean square error

Two classifier designs were studied by Bailey [10] to evaluate their performance; these classifiers were the non-parametric K-Nearest Neighbours (KNN) and Mean Square Error (MSE) criterion. Depending on how the samples were divided, the average misclassification rates of the KNN and MSE classifiers ranged from 25% to 36% with no significant difference between them. Therefore, the linear and non-parametric classifiers proved to be inadequate for chiller detection and diagnostics. These two classifiers were tested to determine which factors significantly affected classifier performance. It was found that using transient data resulted in poor performance. Moreover, the way in which data were collected in different time segments was also critical.

3.5.2 Quantitative models from first principals

In a comparative study, Yu & Van Paasen [13] successfully found that both bond graphs and matlab simulink models could be used for FDD of a HVAC system. The paper shows feasible application of both techniques individually but suggests using a combination of both. This would give the front-end user access to the advantages of either bond graphs or simulink models.

Typical of first principal modelling, this study required mathematical models of all the role-playing elements to be build. For even the single room air-conditioning system studied, this was a tedious process. The sub-models included all the building elements used in the room ranging from ceiling materials to window measurements. The environment needed to be studied in precise detail to deduct reasonable models. If an environmental element changes, for instance the direction in which the room faces, so must the mathematical model change. This reduces the portability of this particular FDD system.

3.5.3 Fuzzy logic control models

Fuzzy logic merits itself on being easy to understand. This makes it a perfect modelling method for user interfaces where the user doesn't require any knowledge of the underlying mechanisms. Kuntze and Bernard [14] implemented a multi-variable climate control system with fuzzy-based multi-objective optimization.

The system was commissioned on low-energy houses to coordinate the HVAC where the user could specify both the comfort and economy criteria. The writers claim that it would be easy to convert it to other industrial applications.

3.5.4 ARX and AFMM

Yoshida and Kumar [15] present a model based methodology for sudden online fault detection in one of the most widely used Variable Air Volume (VAV) HVAC Systems in Commercial and Institutional Buildings. Two models, Auto Regressive Exogenous (ARX) and Adaptive forgetting through multiple models (AFMM), have been trained and validated on data obtained from a real building. The models are trained using normal real time operational data and validated on data obtained by inducing a fault artificially in the damper control sub-system under normal operating conditions. It may be concluded on the basis of results obtained that the variation of parameters rather than the difference between the predicted and actual output is more prominent and reflective of the sudden fault in the system. The AFMM can detect any change in the system, i.e., when a fault is implemented and when the fault is rectified.

Both the ARX and the AFMM models seem quite robust and promising for use in on-line fault detection of the AHU. The advantage of the technique lies in that it requires only a minimal knowledge of the system, besides other benefits associated with black box modelling. A potential limitation of the ARX model is that it can be used for the operating period only and requires a long period to stabilize its parameters. Therefore, it cannot detect consecutive faults at short intervals. On the other hand, the limitation of the AFMM method seems to lie in the required long window length. This may prove a handicap in detecting faults of lesser magnitude than those analyzed in the paper. In the present analysis, data were obtained by ignoring the controller signal and keeping one of the VAV dampers fully open as if it has stuck suddenly. If the same

damper remains half open due to some fault, the system parameters may not change sharply enough so that the fault can be detected and a warning signal can be activated. It may be concluded that both methods can be used to supplement each other. Though the FDD prototype has been studied and validated by a couple of data bases, it seems to be quite robust and promising for an on-line fault detection technique.

3.6 Motivation to implement ANN models

Due to the extensive number of factors required for building first principal models, and the desire to eliminate the need for a costly HVAC expert, a history based modelling method should be applied. Referring to the study done in chapter 2 this leaves either statistical methods or neural networks.

The crucial difference between an ANN and a first principles model is that the former operates somewhat like regressions: it can readily map between input and output data without detailed knowledge of the physics involved [16]. The advantage of neural networks over linear regressions is that neural networks inherently model non-linear processes without upfront data manipulation. This is desirable when modelling building HVAC processes and overall energy consumption because often the values of interest vary non-linearly with the driving variables.

According to Baily and Curtiss [12], ANN models are particularly well suited for modelling slow processes with long time constants, which adequately describes thermal processes that occur naturally within buildings. In addition, typical building thermal and energy data form a cluster of points from which NN models are better able to extract a pattern than are statistical methods. Finally, NN models learn building operating characteristics and require less user input and operator knowledge than regression approaches.

Furthermore Baily and Curtiss [12] referred to an article which weighed an ANN up against a quadratic fit. It was noted that the neural networks are much easier to use once the initial modelling programs were set up since arbitrary amounts of data could be added to the analysis without significant effort. The ANN predictor network learned the relationship between the uncontrolled and controlled variables and energy

use by studying previously recorded data. The noted advantages of ANN models included their ability to handle over parameterized problems by ignoring excess data that were of minimal significance and concentrating on the more crucial input data. In addition, ANN models were found to be ideal for on-line energy minimization due to their rapid iterative speed.

3.7 References in Chapter 3

[1] Haines, R.W., *HVAC systems design handbook*, 1988, TAB books inc.

[2] Throop, D.R. *Model-based diagnosis of complex continuous mechanisms*. 1991. University of Texas. Austin

[3] Kalagasidis, A.S. *H-Tools – International Building Physics Toolbox. Block documentation*. 2002. Department of Building Physics, Chalmers Institute of Technology, Sweden.

[4] Athienitis, A.K. *Thermal analysis of buildings in a mathematical programming environment and applications*. 1998. Building and environment, 34, (401-415). Pergamon.

[5] Bodenstein, C. P. *Process modelling and pattern identification*. 2003. University of the northwest. Potchefstroom.

[6] Watton, A., Marcks, R. K. & Solem, G. *Improved instructional techniques in HVAC control systems*. 20XX. Sinclair community collage. Dayton, OH.

[7] Isermann, R., *Process fault detection based on modelling and estimation – a survey*, 1984, Automatica,.

[8] Comstock, M.C. & Braun, J.E., *Literature review for application of fault detection and diagnostic methods to vapour compression cooling equipment*. 1999. ASHRAE.

[9] Venkatasubramanian, V., Rengaswamy, R., Yin, K. & Kavuri, S.N., *A review of process fault detection and diagnosis. Part I: Quantitative model-based methods*, 2002, Elsevier. Computers and chemical engineering

[10] Bailey, M.B., *The design and viability of a probabilistic fault detection and diagnosis method for vapour compression cycle equipment*, Ph.D. Thesis, 1998, School of Civil Engineering, University of Colorado.

[11] Lee, W.Y., C. Park, & G. Kelly. *Fault detection in an air handling unit using residual and recursive parameter identifications method.* 1996. ASHRAE Transactions, Vol. 102, part 1, AT-96-3-2, pp. 528-539.

[12] Bailey, M.B. & Curtiss, P.S., *Neural Network Modeling and Control Applications in Building Mechanical Systems.* 2002. School of Civil Engineering, University of Colorado.

[13] Yu, B. & Van Paassen, A.H.C., *Modelling with simulink and bond graph method for fault detection in an air-conditioned room.* 2001. Delft University of Technology.

[14] Kuntze, H. B. & Bermard, T.H., *A new fuzzy-based supervisory control concept for demand responsive optimization of HVAC control systems.* 1998. Proceedings on the 37th IEEE conference on decision and control.

[15] Yoshida, H. & Kumar, S. *ARX and AFMM model-based on-line real-time data base diagnosis of sudden fault in AHU of VAV system.* 1998. Energy Conversion & Management 40 1191-1206. Pergamon.

[16] Jordan, M.I. & Bishop, *C.M. Neural networks.* 1996. Massachusetts institute of technology.

CHAPTER 4

4 FAULT DETECTION AND DIAGNOSTICS WITH NEURAL NETWORKS

In the previous chapter, a literature survey showed that artificial neural network (ANN) models suffice for modelling HVAC systems. In addition, chapter 3 explained that most HVAC systems are modelled with first order transfer functions. In this chapter neural networks are used to model first order transfer functions.

The first elements to an FDD structure are built. The internal workings of ANNs are studied before they are implemented as a modelling mechanism for first and second order systems. Generalized failures are injected into the system and the response between the ANN models and the system is analyzed.

The goal is to find out if different faults produce detectable and identifiable patterns in the residuals. Methods to differentiate between faults are also studied.

4.1 Theory on artificial neural networks

In data analysis, ANNs are a class of flexible nonlinear models used for supervised prediction problems [1]. Yet, because of the ascribed analogy to neurophysiology, they are usually perceived to be more glamorous than other (statistical) prediction models. Due to this popularity, neural network theory has become widely known. Therefore this section will be kept brief and will focus on the specific networks implemented.

The basic building blocks of an ANN are called hidden units. Hidden units are modelled after the neuron. Each hidden unit receives a linear combination of input variables. The coefficients are called the (synaptic) weights. An activation function transforms the inputs and then passes them to another unit that can use them as input.

An ANN is a flexible framework for specifying a variety of models. The most widely used type of neural network in data analysis is the multilayer perceptron (MLP). A MLP is a feed-forward network composed of an input layer, hidden layers built from hidden units, and an output layer.

The input layer is composed of units corresponding to each input variable. For nominal inputs, there may be one input unit for each distinct level. Consequently, the number of input units may be greater than the number of inputs. The hidden layers are composed of hidden units. Each hidden unit puts out a nonlinear function of a linear combination of its inputs—the activation function. The output layer has units corresponding to the target. With multiple target variables there are multiple output units. Figure 17 presents the model of a MLP where the inputs are time delayed samples of a single input vector. It is this type of network that is used in this chapter for signal prediction purposes.

The network diagram is a representation of an underlying statistical model. The unknown parameters (weights) correspond to the connections between the units.

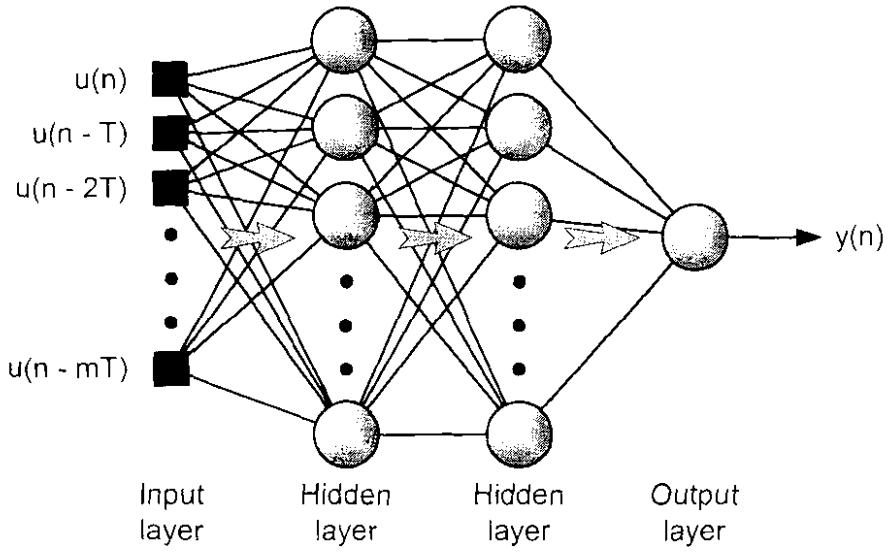


Figure 17. Layout of a MLP

Each hidden unit yields a nonlinear transformation of a linear combination of their inputs. The linear combination is the net input. The nonlinear transformation is the activation function. The activation functions used with MLPs are sigmoidal curves (surfaces) such as Figure 18.

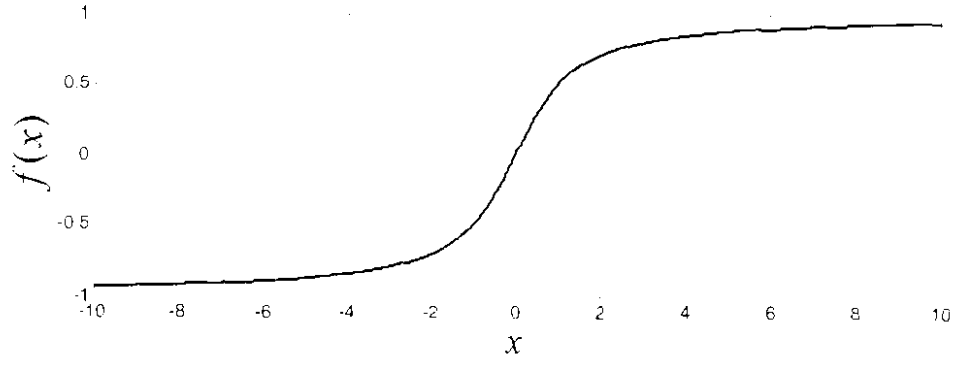


Figure 18. Hyperbolic tangent sigmoid transfer function

An output activation function is used to transform the output into a suitable scale for the expected value of the target. In statistics, this function is called the inverse *link function*. For binary targets, the logistic function is suitable because it constrains the output to be between zero and one. The logistic function is sometimes used as the activation function for the hidden units as well. This sometimes gives the false impression that they are related. The choice of output activation function depends only on the scale of the target.

A regression model, such as a MLP, depends on unknown parameters that must be estimated using the data. Estimating the weights and biases in a neural network is

called training the network. The error function is the criterion by which the parameter estimates are chosen (learned). Every possible combination of parameter estimates corresponds to a prediction of the expected target. Error functions can be thought of as measures of the distance between these predictions and the actual data. The objective is to find the set of parameter estimates that optimizes (minimizes) the error function. [16]

For some simple regression models, explicit formulas for the optimal estimates can be determined. Finding the parameter values for neural networks, however, is more difficult. Iterative numerical optimization methods are used:

- Starting values (initial guess)
- New value = previous value + update
- Iterate until convergence (no further progress).

Optimization can be thought of as searching for a global optimum (minimum) on a multidimensional surface. The contours of the above surface represent level values of the error function. Every pair of values of the two parameters is a location on the surface. There are many algorithms for determining the direction and distance of the update step. As Figure 19 shows, the optimization process can be complicated by multiple minima, saddle points, flat regions, and troughs.

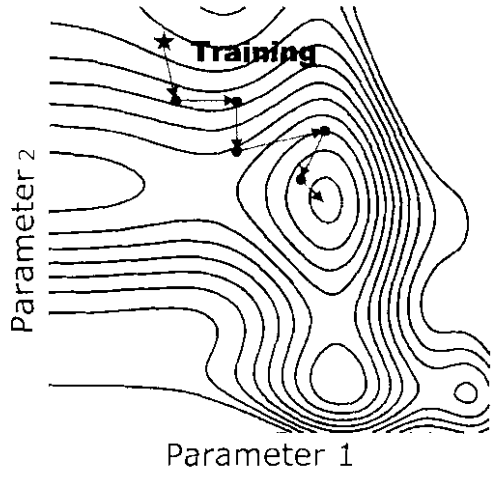


Figure 19. Training converging [1]

Back-propagation most often refers to an optimization algorithm used to fit the network to data (it is actually an analytical step of the algorithm) [3]. The model and optimization algorithm are separate entities. Many algorithms can be used to fit a

particular nonlinear regression model. Back-propagation has historical ties to the development of ANN. Consequently, the term is often used synonymously with neural network. In fact, MLPs are often called Back-propagation networks. The problem with this confusion is that Back-propagation is a variation of an inefficient and obsolete algorithm called gradient descent. It can be painfully slow and may require extensive tuning.

A MLP with one hidden layer is a universal approximator. That is, it can theoretically approximate any continuous surface to any degree of accuracy (for some number of hidden units). In practice, a MLP may not achieve this level of flexibility because the weights and biases must be estimated from the data. Moreover, the number of hidden units that are required for approximating a given function might be enormous.

The price neural networks pay for their flexibility is incomprehensibility. It is difficult to decide which input variables are important or how they interact. Neural networks are usually treated as black boxes. One feeds it the inputs, it does something mysterious with them, and then it generates a sensible prediction. In modelling, comprehensibility is not always a crucial feature of a prediction model. A neural network that produces highly accurate predictions might be preferred over other more understandable but less effective models. Nevertheless, incomprehensibility can be a practical shortcoming. Understanding variable importance and interactions may improve future data collection. In addition, it is easier to convince non-quantitative users or clients of the value of the methodology if they can understand the model.

4.2 Fault detection and diagnosis with ANN models

The purpose of this section is to investigate some properties of residuals between a plant (or simulation) and artificial neural networks. Theoretically, it is shown how to implement a FDD system from these residuals. It was shown in chapter 3, that many components used in HVAC, are modelled by first order differential equations. In the s-plane, first order differential equations translate to first order transfer functions.

Consider a plant which consists of two first order transformations as shown in Figure 20. The sections can experience three types of faults, namely an offset at the input, a

change in steady state gain and a change in time constant. The instruments at the plant outputs can suffer from zero offset changes as well as changes in transduction ratio or gain.

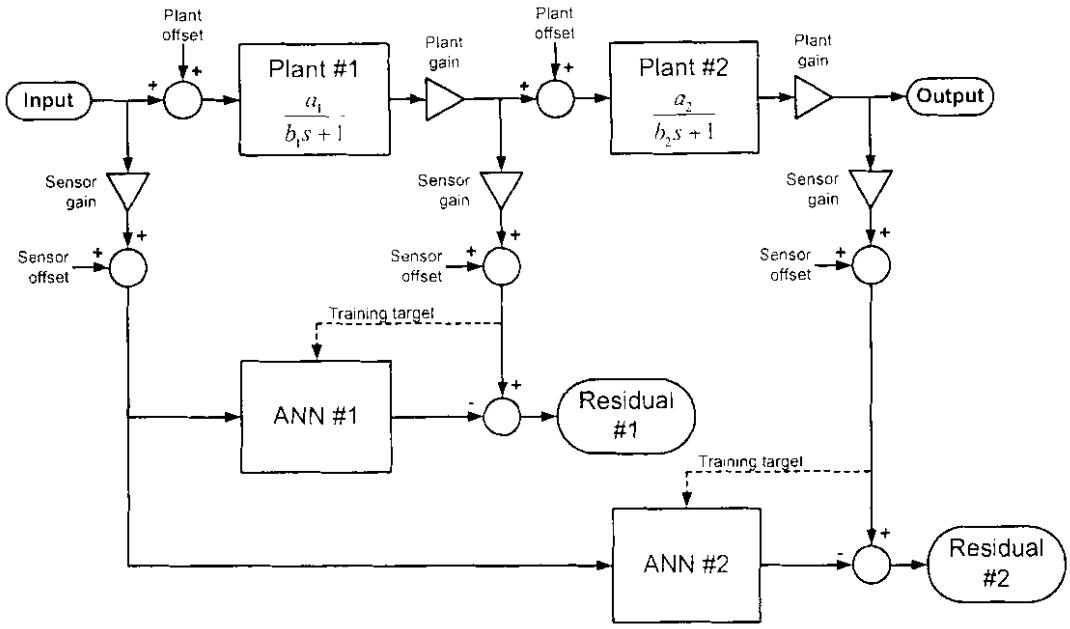


Figure 20. Simulation studied for FDD

For the purposes of the simulation of this section, the input sensor was modelled as fault free (no offset, no change in gain) so that the plant and ANN always a received a fault free input.

Neural net #2 of Figure 20 imitates the second order combination of both plants. As will be shown, this allows the validation of even the sensor outputs. To get to the point of complete process validation, the upcoming sections step through residual generation to fault identification.

4.2.1 Basic simulation and training of a first order system

To clarify how the simulations were done and as an example of training, consider the first order system of Figure 21. In comparison with Figure 20, this reduced system can be viewed as a building block of our target system.

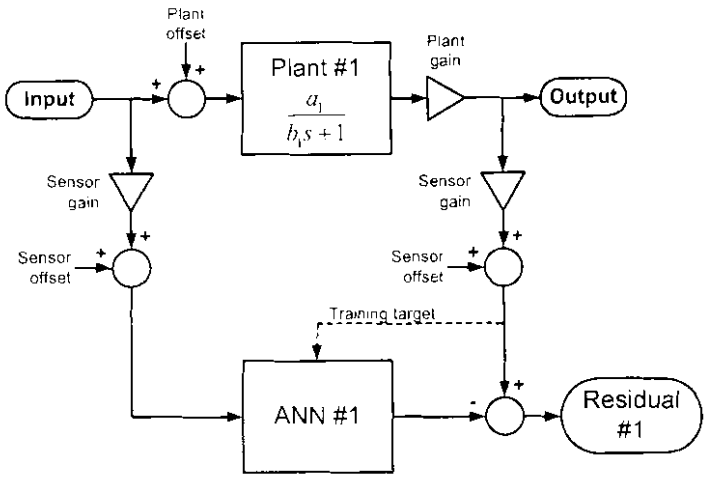


Figure 21. First order system modelled with an ANN

The input signal is a simple random square wave as Figure 22.a shows. This signal is fed into both the plant and the trained neural network. The plant produces a signal in the shape of a first order transformation also displayed in Figure 22.a. It would be in this segment where either a plant offset error or a plant gain error is introduced in the simulation. The plant output is then used to include any of the sensor errors. The output of the neural network in Figure 22.b is an imitation of a fault free plant. The residual, displayed in Figure 22.c, is the difference between the plant output and the neural network output.

The architecture found to be most efficient for this type of network is a 14-4-2-1 structure (14 delayed inputs, 4 neurons in the first hidden layer 2 in the second hidden layer and of 1 output). In appendix A.2 a more detailed discussion about network design for this system can be found.

The neural network in Figure 21 is trained under ideal conditions only. This is the state where there are no faults included in the system. All the gains are set to one while the offsets are zeroed. The trained neural network then mimics the plant under ideal conditions. The target of the neural network was to have a root mean square (MSE) value below 0.001 on normalized validation data. It is however possible to set a target of 0.000 01, which cleans the residuals of a lot of noise. In section A.2.2 the selections and implication of MSE values are discussed.

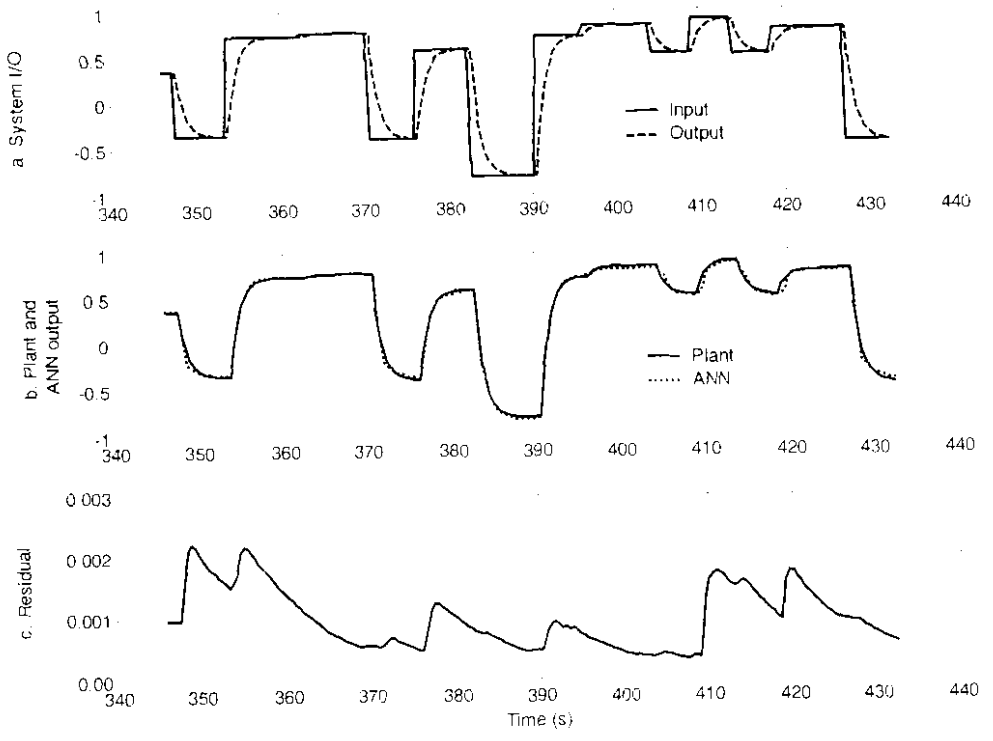


Figure 22. Input and output to a first order system

4.2.2 Fault introduction to first order systems

As stated, the plant of Figure 21 can undergo three types of faults:

- Change in gain
- Offset incrimination and
- Time constant changes

In this section the focus is on how each fault propagates through to the residual. Studying certain aspects of the residual reveals how to detect and identify the fault at hand. The following 3 graph sets, Figure 23 through Figure 25, exhibit the different properties of the residuals.

In each of the simulations, the fault was injected between 870 and 1050 seconds. Notice that the coinciding graphs between the different faults are on the same scale, - displaying the visual differences that can be used for FDD.

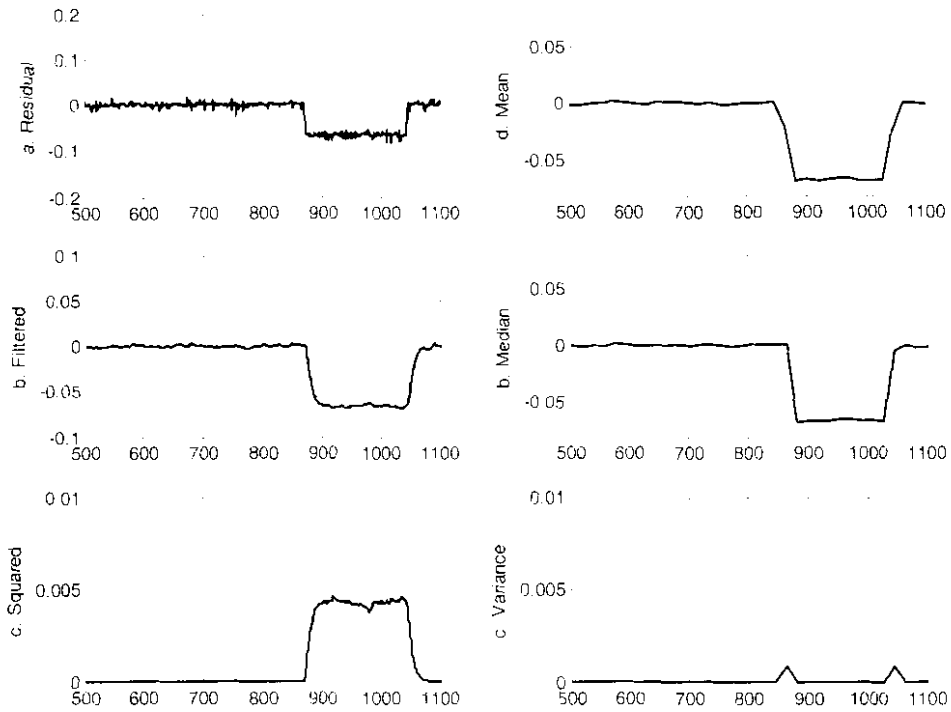


Figure 23. Residual properties of an offset error

Detecting when errors occur is rather easy. The residuals from graphs a. in Figure 23, Figure 24 and Figure 25, fluctuate significantly from zero to pinpoint accurately when the system is undergoing a failure. The same information can be obtained from the filtered residuals in graphs b; however, filtering the signal first, reduces the influence of noise. Squaring the filtered residual provides an even more detectable spread as is displayed in graphs c. of the three figures.

Differentiating between errors tends to be a more tedious task. The offset error of Figure 23 leaves rather unique patterns in graphs a. and b. On the other hand, the spreads of the first three graphs in Figure 24 and Figure 25 are similar in shape. This makes it hard to distinguish gain errors from changes in time constant. Looking at statistical elaborations on the residual resolves this issue.

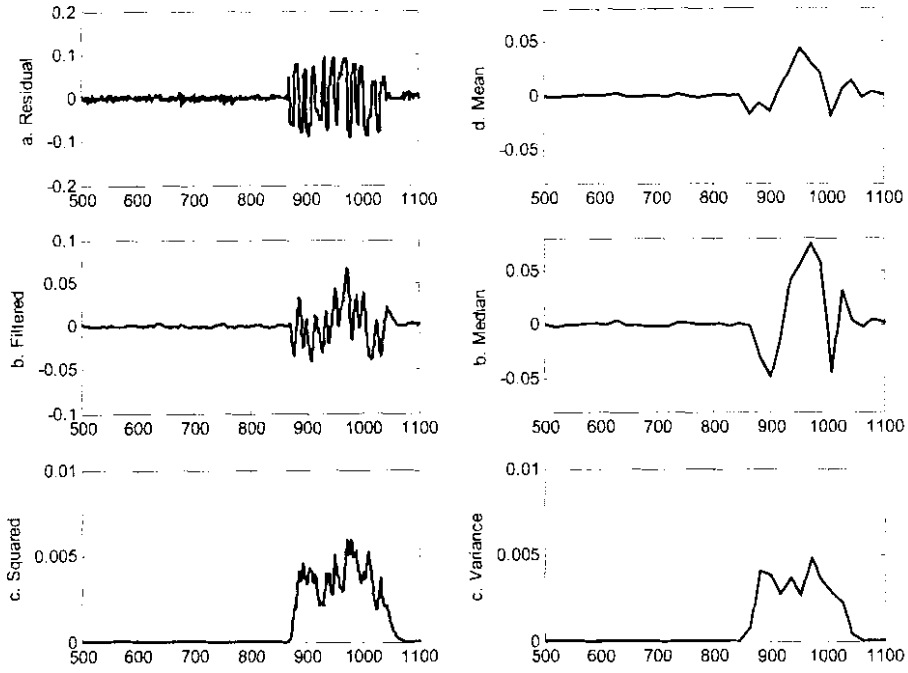


Figure 24. Residual properties of a 10% gain increace

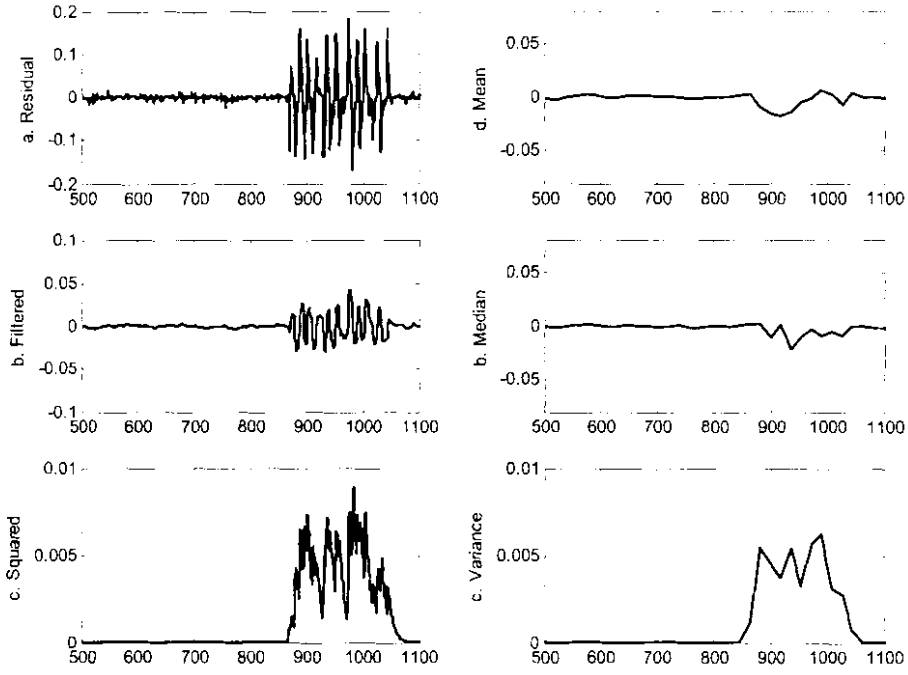


Figure 25. Residual properties for a 10% change in time constant

Of Figure 23, graph d. - the mean, and graph e. - the median, resemble the residual of graph b. revealing nothing new. The mean and, especially, the median graphs of Figure 24 and Figure 25 differ considerably. This allows distinction between changes in time constant and increases in gain. The variance, shown in graphs f. of the figures at hand, offers a simple manner to single out offset errors from the rest.

There are numerous methods to implement these graphs for fault identification. Working with residuals suggests the use of diagnostic observers as discussed in section 2.5.1. Although the residuals are generated differently from the methods described in diagnostic observer theory, implementing a decision matrix seems feasible. Linguistically this could look like:

- If *squared filtered residuals* < *X* then *fault = false*
- If *squared filtered residuals* > *X* and *variance* < *Y* then *fault = Offset error*
- If *squared filtered residuals* > *X* and *variance* > *Y* and *median* > *Z* then *fault = Gain increase*
- If *squared filtered residuals* > *X* and *variance* > *Y* and *median* < *Z* then *fault = Change in time constant,*

where *X*, *Y* and *Z* are thresholds deducted from measurements. These values are influenced by the noise in the system as well as the accuracy of the ANN. For more information on trigger values the reader is referred to appendix A.

In a more appropriate tabular form the identification matrix can be seen in Table 2. This matrix holds unique trigger patterns for each possible failure. An 'X' represents a threshold that has been crossed while an 'O' marks a trigger which does not fire. From this table fault identification is possible.

Table 2. Fault Identification matrix

	Residual Properties		
	Squared filtered	Variance	Median
Normal operation	O	O	O
Offset	X	O	X
Gain	X	X	X
Time constant	X	X	O

One problem still remains. The FDD system can, thus far, differentiate between offset increases, gain changes and changes in time constant. The offset and gain errors,

however, can originate in either the plant or it can be a fault within the sensor measuring the plant. In the following sections this problem is addressed.

4.2.3 Fault propagation through a second order system

Consider Figure 26. Once a fault is generated within the first plant, it is automatically passed on to the second plant. The sensor also picks up this fault and forwards it to the FDD system producing a fluctuation on residual #1. The input to the second plant thus deviates from normal operating conditions. This, in turn, leads to a measurable, unexpected output from the second plant.

If the same stray, instead of originating in the first plant, came from the sensor measuring on the first plant, the failure would not propagate to the second plant and accordingly not to residual #2. This is clear from the directional flow of information in Figure 26. To sum up, if residual #1 shows an error, but the propagation is not visible in residual #2, the fault lays within sensor 1 and not inside the plant.

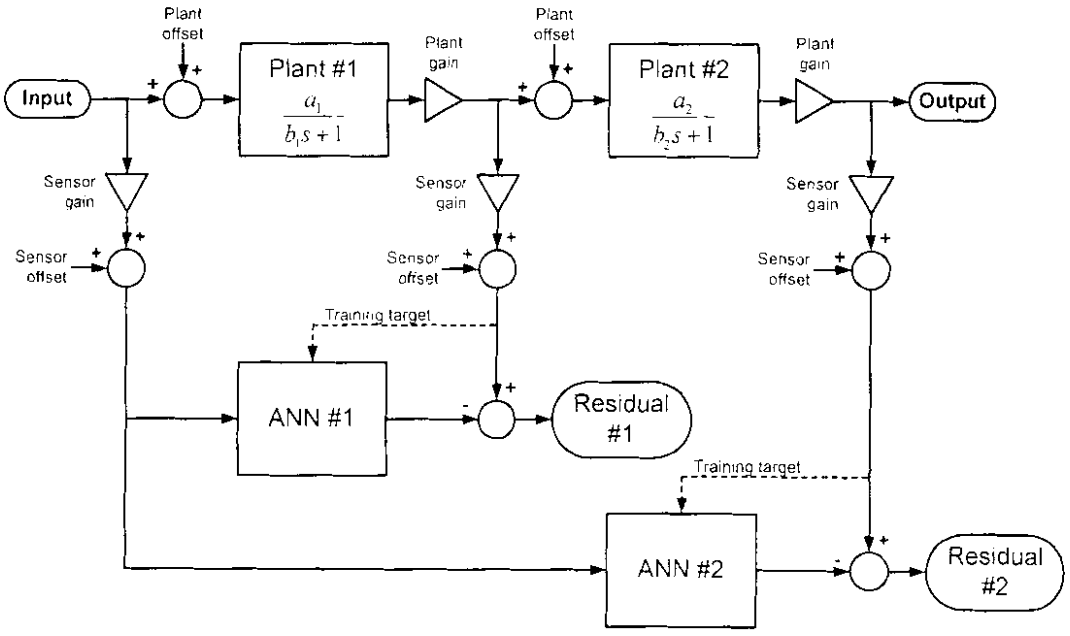


Figure 26. Second order FDD simulation

Notice that the second neural network in Figure 26 receives the same input as the first network but attempts to mimic the output from the second plant. Therefore this neural

network replicates a second order curve comprising the transformations both plants induce. This curve is shown in Figure 27.

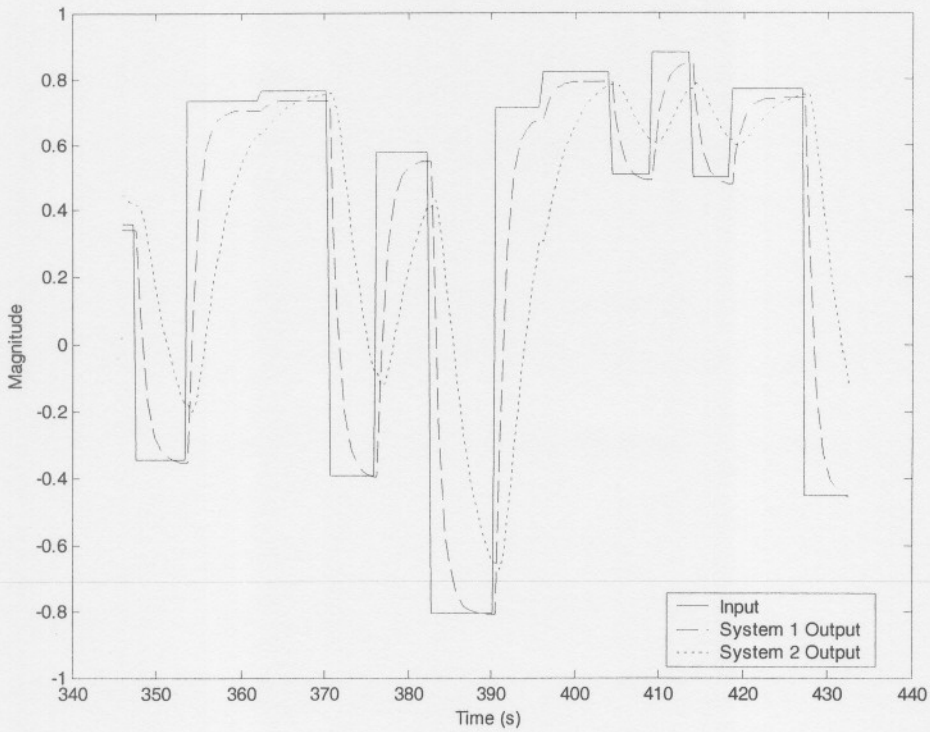


Figure 27. Input and output of a second order system (includes intermediate first order output)

Due to the order of the combined plant, the second ANN needs to imitate a more complex signal than the first ANN. It requires more neurons and therefore trains longer to reach the same level of accuracy as the first network. The most efficient architecture for ANN #2 is a 56-8-4-1 structure (refer to appendix A.2 for more detail). Both networks were set a target MSE value of 10^{-3}; however, a target of 10^{-6} remains obtainable.

Counting faults originating in the sensor and faults originating within the plant individually, leaves a total of five failures to consider:

- Sensor change in gain
- Plant change in gain
- Sensor offset
- Plant offset
- Change in time constant

In the following figures (Figure 28 to Figure 31) the graph sets produced by these faults are discussed. As before, the corresponding graphs are plotted on the same scale to indicate visual differences.

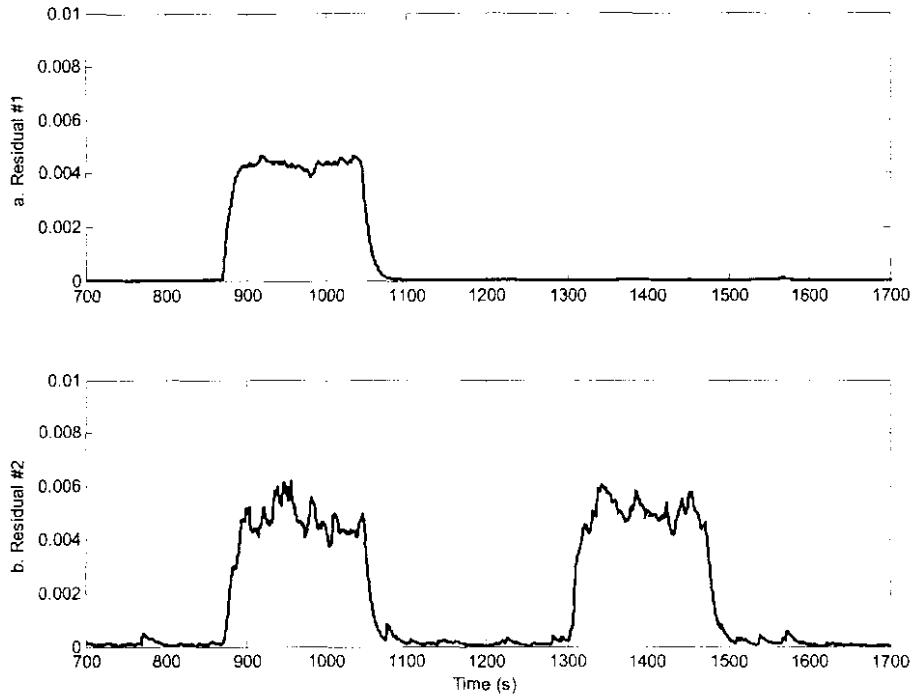


Figure 28. The squared filtered residuals of a 10% Offset incriminations in both plants

The failures are introduced between 870 and 1050 seconds and on average deviate 10% from the normal value. Each figure contains a second fault from 1300 up to 1480 seconds. This second fault is a replication of the first fault, only working in on the second plant or its corresponding sensors. Therefore it is only visible on the second residual.

Figure 28 and Figure 29 depict the squared filtered residual of offset failures in the plant and in the sensors respectively. In Figure 28, the error from the first plant (graph a) is carried over into the second plant (graph b). As expected, the sensor error shown in Figure 29, graph a. does not propagate to a disturbance in Figure 29, graph b.

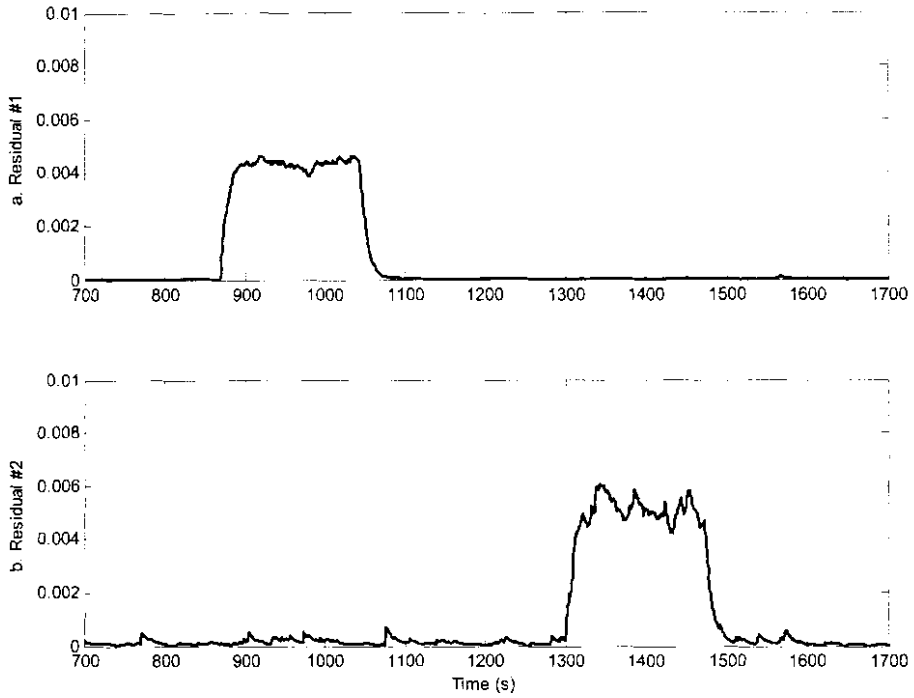


Figure 29. The squared filtered residuals of a 10% Offset incriminations in both sensors

The exact same phenomenon occurs when a sensor gain is compared to a gain increase within the plant. Although Figure 30.a. seems like a duplicate of Figure 31.a; only the gain disturbance in the plant propagates into the second plant, the gain disturbance in the sensor is does not disrupt the residual from the second neural network.

The appearance of failures from the first plant in the second residuals allows a FDD system to distinguish sensor faults from plant faults. The second plant serves as a validation of the sensor on the first plant. This imposes a new requirement on the second neural network: to differentiate between errors within the second plant and errors carried over from the first. A solution presents itself in the residual from the first plant. As stated, a disruption in a residual points to fault. If there is no detectable spread in the residual, the plant is considered operating normal. Concluding, that if the second residual shows a failure while the first residual remains undisrupted, the fault arose in the second plant.

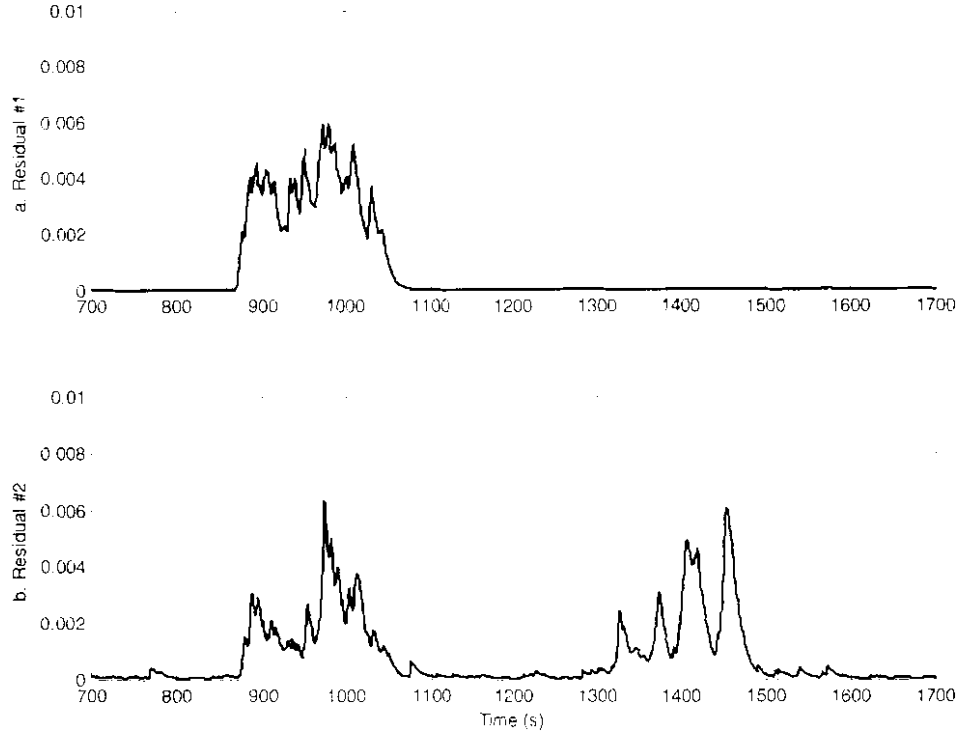


Figure 30. The squared filtered residuals of a 10% gain increase

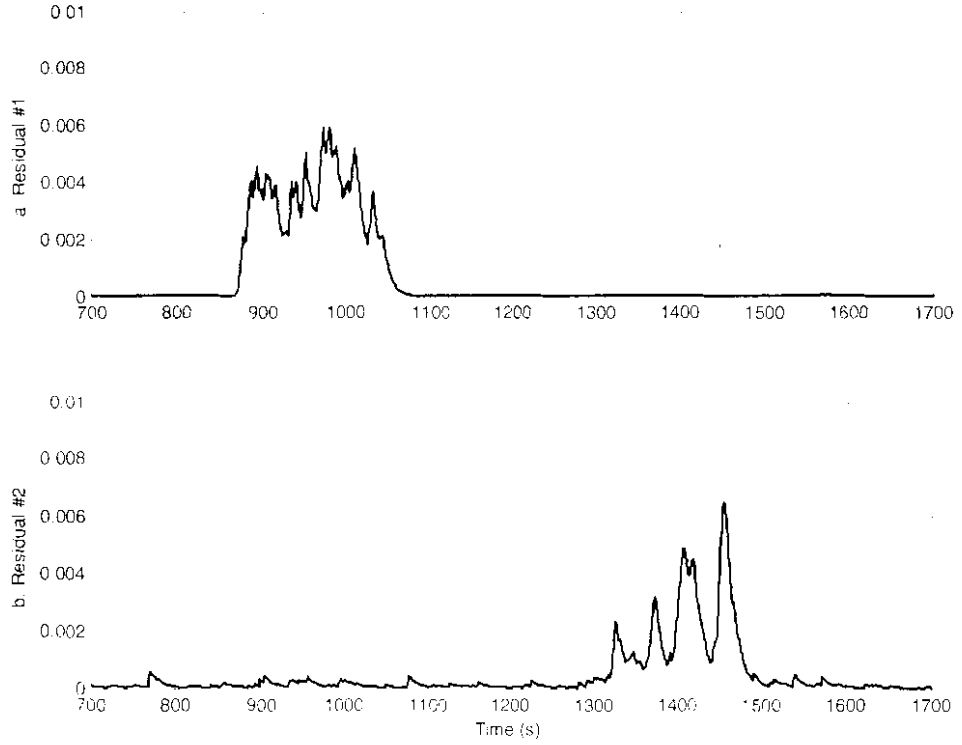


Figure 31. The squared filtered residuals of a 10% increase in gain within both sensors

The fault identification matrix from Table 2 can be broadened for fault detection on a second order system. Table 3 shows a functional matrix. Once again, an 'X' represents a threshold that has been crossed while an 'O' marks a trigger which did not fire. An implementation of such a matrix is included in appendix B – the data disk.

Table 3. Fault identification matrix for a second order system

		Residual #1			Residual #2		
		Squared filtered	VAR	Median	Squared filtered	VAR	Median
Normal operation		O	O	O	O	O	O
Plant #1	Plant offset	X	O	X	X	O	X
	Sensor offset	X	O	X	O	O	O
	Plant gain	X	X	X	X	X	X
	Sensor gain	X	X	X	O	O	O
	Time constant	X	X	O	X	X	O
Plant #2	Plant offset	O	O	O	X	O	X
	Sensor offset	O	O	O	X	O	X
	Plant gain	O	O	O	X	X	X
	Sensor gain	O	O	O	X	X	X
	Time constant	O	O	O	X	X	O

Selecting these thresholds of triggers depends on mainly two factors. First is the noise level of the signal. Although the graphs in this section were generated with noiseless data, in practice, measurement noise could trigger thresholds that have been set too sensitive, leading to false error detection. This is the balance between robustness and performance.

The second factor to consider for selecting thresholds is the accuracy of the neural network. A high performance network allows the detection of much finer faults. This proportionality is discussed in appendix A.

4.3 References in chapter 4

[1] Sarle, W. S. *How to measure the importance of inputs?* (ftp://ftp.sas.com./pub/neural/importance.html), accessed 4/1/98.

[2] Jordan, M.I. & Bishop, *C.M. Neural networks.* 1996. Massachusetts institute of technology.

[3] Haykin, S. *Neural networks – a comprehensive foundation.* 1999 Prentice-Hall, Inc. (156-255).

CHAPTER 5

5 FAULT DETECTION AND DIAGNOSIS ON SOME ASPECTS OF HVAC PRESSURE CONTROL

In chapter 4 a generalized approach was taken to fault finding and fault identification. In this chapter fault detection and diagnosis methods are employed in a more practical environment, with a study of a static pressure control loop.

The structure of an FDD system is extended from the generation of residuals via ANN models through statistical analysis of the residual to the tree structure utilized for fault identification. Processes and methods described in the preceding chapters are combined to form a complete implementation of an FDD system. At the end of this chapter, this FDD system is rated and judged according to the characteristics described in chapter 2.

The aim is to twofold – not only to describe an FDD system, but also to take the reader through the process of the development and the expansion of the FDD system.

5.1 Methodology

5.1.1 Airflow control model

The HVAC component studied in this section is the static pressure control loop presented in Figure 32. Fan driven air is forced through the ducting of a building. To assure that the spread of air remains even all round, the air-pressure is constantly under adjustment by numerous controlled dampers. Figure 32 shows the placement of one such damper in a duct segment.

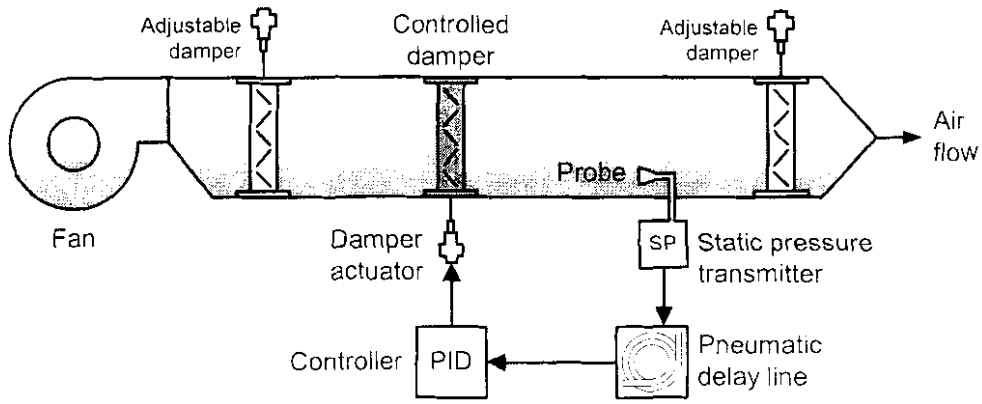


Figure 32. Static pressure control loop

The pressure is measured with a pitot-static probe. The probe's signal is carried from a static pressure transmitter through a pneumatic delay line to the controller. In this study a PID-controller was implemented to control the damper actuator that controls the damper which, in turn, limits the pressure.

Alan Watton, Russell K. Marcks, and Gifford Solem [1] built a first principles model in MATLAB-simulink of the static pressure loop of Figure 32. In Figure 33 an adaptation of their model is shown. This model is used throughout the chapter to optimize a FDD system.

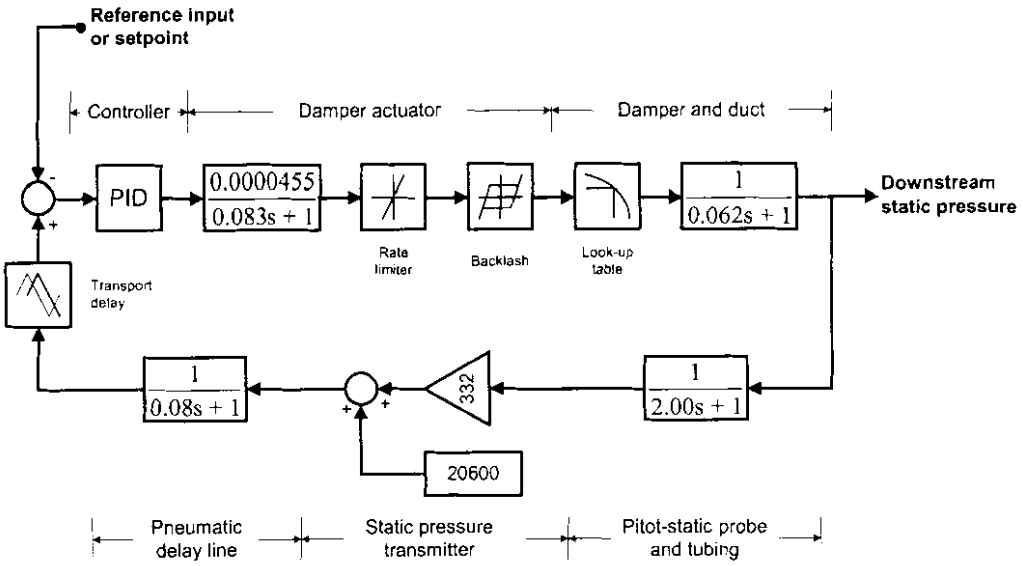


Figure 33. Airflow control diagram [1]

5.1.2 Fault list

The static pressure control loop of Figure 32 can be spoiled by numerous faults. In this section some of these faults are studied by looking at where they originate and how the loop is influenced by such faults, thus serving as an introduction to the next section where a FDD solution is implemented to differentiate between these faults.

Each of the components completing the control loop has its own set of failures. The complete sum of errors is impossible to discuss, never mind thoroughly studied. The plan is to summarize the more common failures over the whole set of components. The faults to be studied include:

- Controller failure
- Increase in the actuator hysteresis
- Leakage in the ductwork
- Damage to the probe
- Puncture in the pneumatic line

Controller failures

Controller failures cover a massive scope of possible failures; however, controller failures are highly uncommon. A complete disconnection between the controller and

the actuator produces a catastrophic failing of the balance in the control loop. Such a problem is simple to detect and identify due to the dramatic repercussions. The controller fault studied here is a locked output. In other words, the controller output is a constant value like the example shown in Figure 34. Like most of the faults which will be studied, the controller lock is simulated from 600 to 800 seconds. Thereafter, as Figure 34.b shows, the PID-controller needs to compensate heavily to rectify the stray in Figure 34.a.

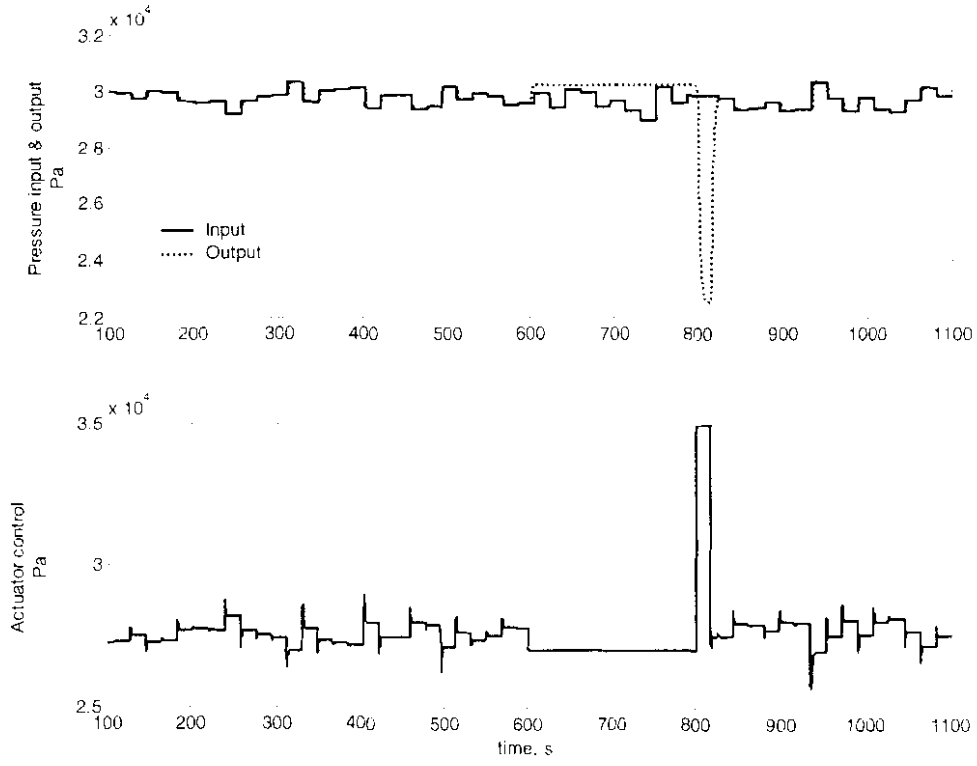


Figure 34. Controller failure.

Increase in the actuator hysteresis

The build-up of dirt in the actuator increases the slack in the movement of the dampers. This means that the dead-zone around the pivot point of the dampers enlarges. The outcome of these types of faults can be seen as an added random factor inside the loop during the changeover to a new set-point.

Leakage in the ductwork

Any unplanned opening in the ducts of a HVAC system leads to the loss of heat. To compensate for such a loss, higher levels of input power are required, reducing the

overall effectiveness of the system. As HVAC systems account for 40% of building energy usage, such energy loss increases operation costs. [2]

Damage to the probe

The Pitot-static tube is a probe which predicts the flow velocity in a duct based on Bernoulli's equation [3]:

$$\frac{1}{2}\rho v^2 + P_s = P_t$$

This states that the sum of the static pressure, P_s, and the velocity pressure, $½\,\rho v^2$, is the total pressure, P_t, which is constant along a streamline. When a Pitot-static tube is immersed into the flow, as in Figure 35, the velocity at the stagnation point at the tube nose is v = 0 and the local static pressure equals the total pressure, P_t. The flow static pressure, P_s, is measured a short distance downstream from the surface of the tube.

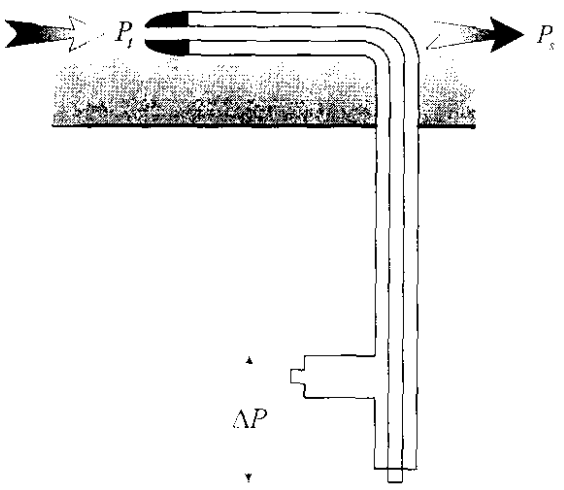


Figure 35. Pitot - static tube in a streamline

The flow velocity is obtained by applying:

$$v = K\sqrt{\frac{2\Delta P}{\rho}}$$

where a constant, K is added to take into account the deviations from the ideal case, $\Delta P = P_t - P_s$ is the pressure difference between the total pressure and the static pressure, and, ρ is the fluid density. The factor K is dependent mainly on the tube construction.

When the axis of the Pitot–static tube is not aligned with the direction of the air flow, an error occurs known as yaw. In this study, yaw is simulated by applying a fractional gain to P_s. Reconsidering the velocity equation, a yaw fault contains elements of non-linear offset:

$$\Rightarrow v = K\sqrt{\frac{2\left(P_t - (1-\varepsilon_p)P_s\right)}{\rho}}$$

$$\Rightarrow \rho\left(\frac{v}{K}\right)^2 = 2\left(P_t - P_s\right) - 2\varepsilon_p$$

From this it is clear that a small alignment mistake ($\varepsilon_p < 0.01$) shows an insignificant change in the balance, but as ε_p goes beyond 10% the fault increases with quadratic proportions to v.

Puncture in the pneumatic line

Any puncture in the pneumatic delay line results in the steady loss of pressure to the pneumatic controller. It should thus be treated as an incipient fault rather than as an abrupt incident. This specific fault is studied to see how the FDD system reacts to incipient faults. It will be modelled as a linear decay until the pressure feedback becomes zero. Refer to Figure 2 in chapter 2.

5.1.3 FDD solution for the static pressure control loop

In chapter 4, FDD work was done on an open loop system. In this chapter a closed loop system is studied. New problems arise because the PID controller tends to compensate for failures in system components. Logically, a change in the loop would be more detectable in the compensation brought along by the controller rather than in the output of the system. Therefore the first neural network mimics the controller output, generating a residual from which significant changes within the controller signal can be studied. See Figure 36.

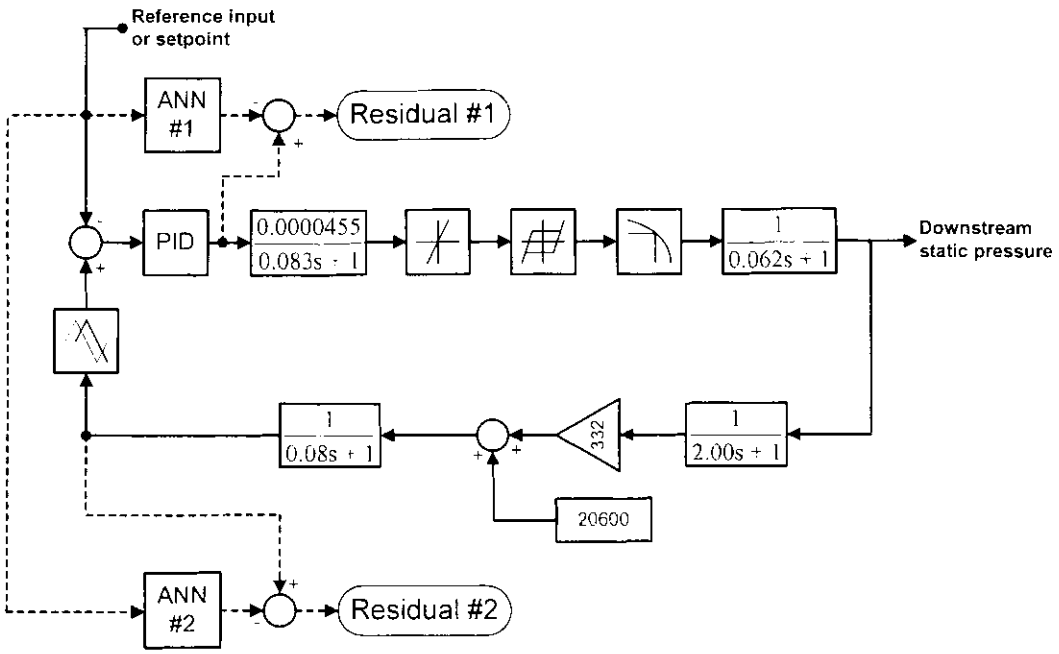


Figure 36. FDD system on the static pressure loop

The second network monitors the pitot-static tube's output. Referring to the work done in chapter 4, it is expected from this second residual to at least differentiate between sensor faults and plant faults on the first network. Moreover, Figure 36 shows that repercussions of all the mentioned faults, eventually, pass through the connection point to the second residual. If the faults are large enough there should be a detectable change in the second residual.

The two neural networks used for FDD purposes are related through similarities. Similar characteristics include the type of network, the training method and the input to the network – as either network receives only the system set-point. Both networks were constructed as feed forward ANNs trained with the quasi-Newton algorithm. Where the networks, that were eventually used, differ from each other are in the architecture. For ANN #1 a 50-15-5 network layout, which requires 5 seconds of delayed input data, was employed. For ANN #2 a much larger network was required to capture the transient. The most sufficient network had a 120-12-4 layout which used 12 seconds of delayed input data. For a broader discussion on neural network training and configurations the reader is referred to appendix A.3

5.2 Results

This section contains residuals from the five failures discussed in section 5.1.2 – the fault list. Over the next five pages the residuals and statistical alterations on the residuals are shown. The attempt would be made, as was done in chapter 4, to deduct individual patterns between the graphs for each fault mentioned. This page contains a summary of what can be expected in Figure 37 to Figure 41.

Each figure contains the residuals and its alterations divided equally between the first and the second residual. The top graphs on each page show how the system responds to a fault within. A white noise component was added to simulate measurement noise. The second row of graphs shows the two neural networks' prediction of what the state should be. In the third row are the residuals – the difference between the plant outputs and the neural network outputs. These are followed by a string of variations on the residual, namely: filtered residuals, squared filtered residuals, median, variance (VAR) and the standard deviation (STD).

The faults graphs are included in the following order:

- Figure 37. Residual signals for a controller failure
- Figure 38. Residual signals for an increase in the damper hysteresis
- Figure 39. Residual signals in case of a leak in the ductwork
- Figure 40. Residual signals when the probe suffered damage
- Figure 41. Residual signals for a puncture in the pneumatic delay line

All these faults commence at 600 seconds and are rectified at 800 seconds, except for the puncture in the pneumatic line that is never rectified, only halted after 800 second. This fault is considered an incipient fault while the other are simulated as abrupt occurrences.

These plots were made with the aid of MATLAB®, making use of Simulink, the neural network toolbox and the statistical toolbox.

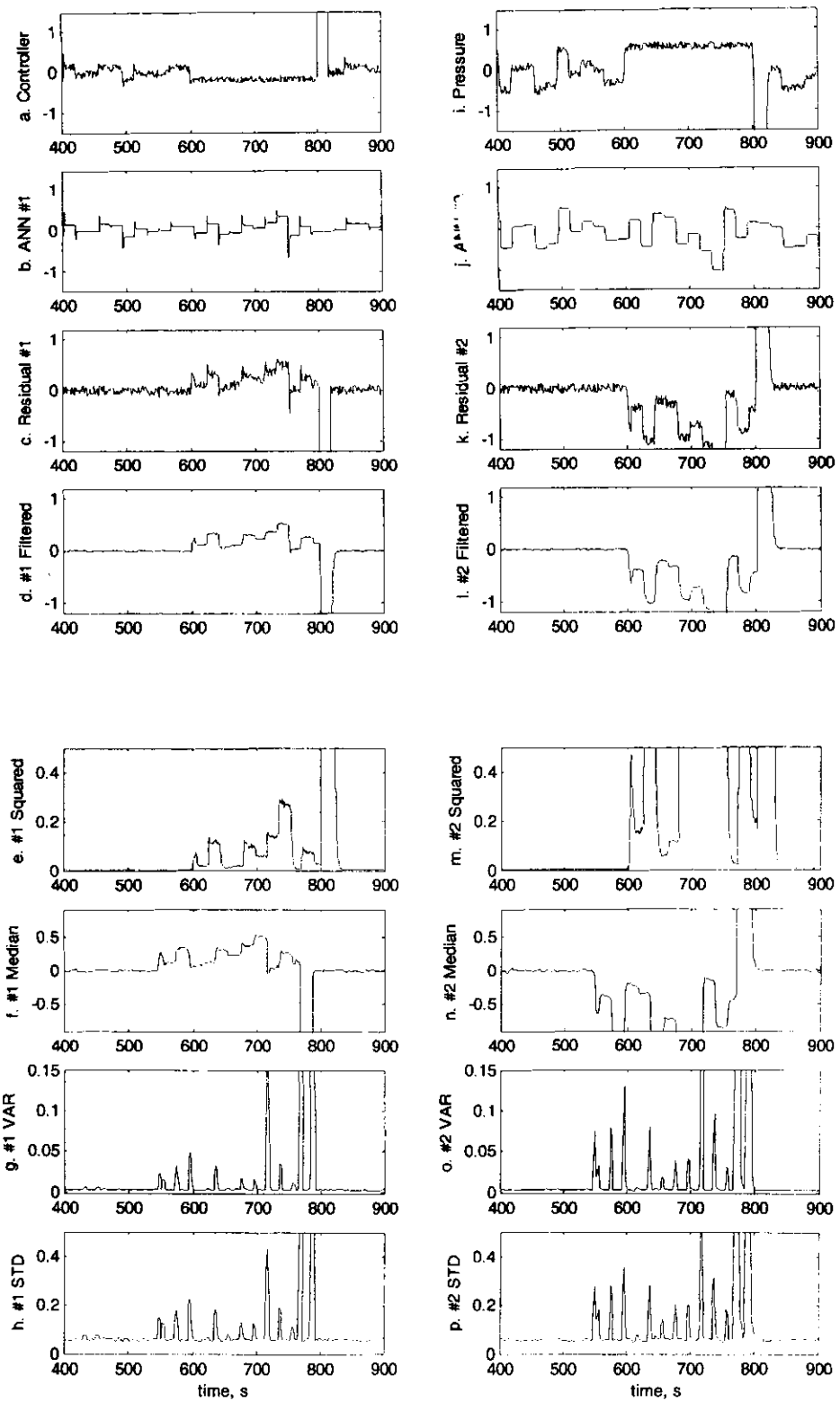


Figure 37. Residual signals for a controller failure

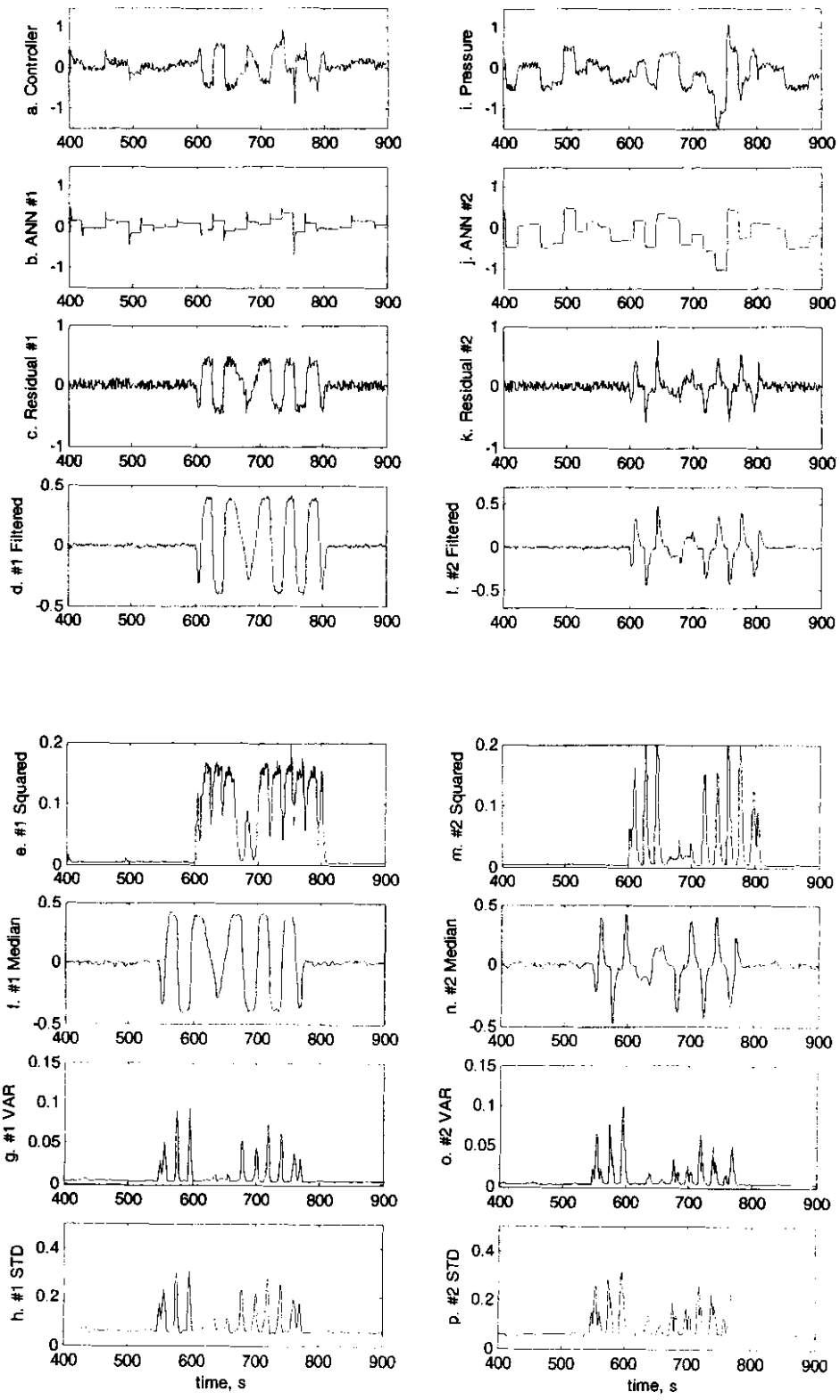


Figure 38. Residual signals for an increase in the damper hysteresis

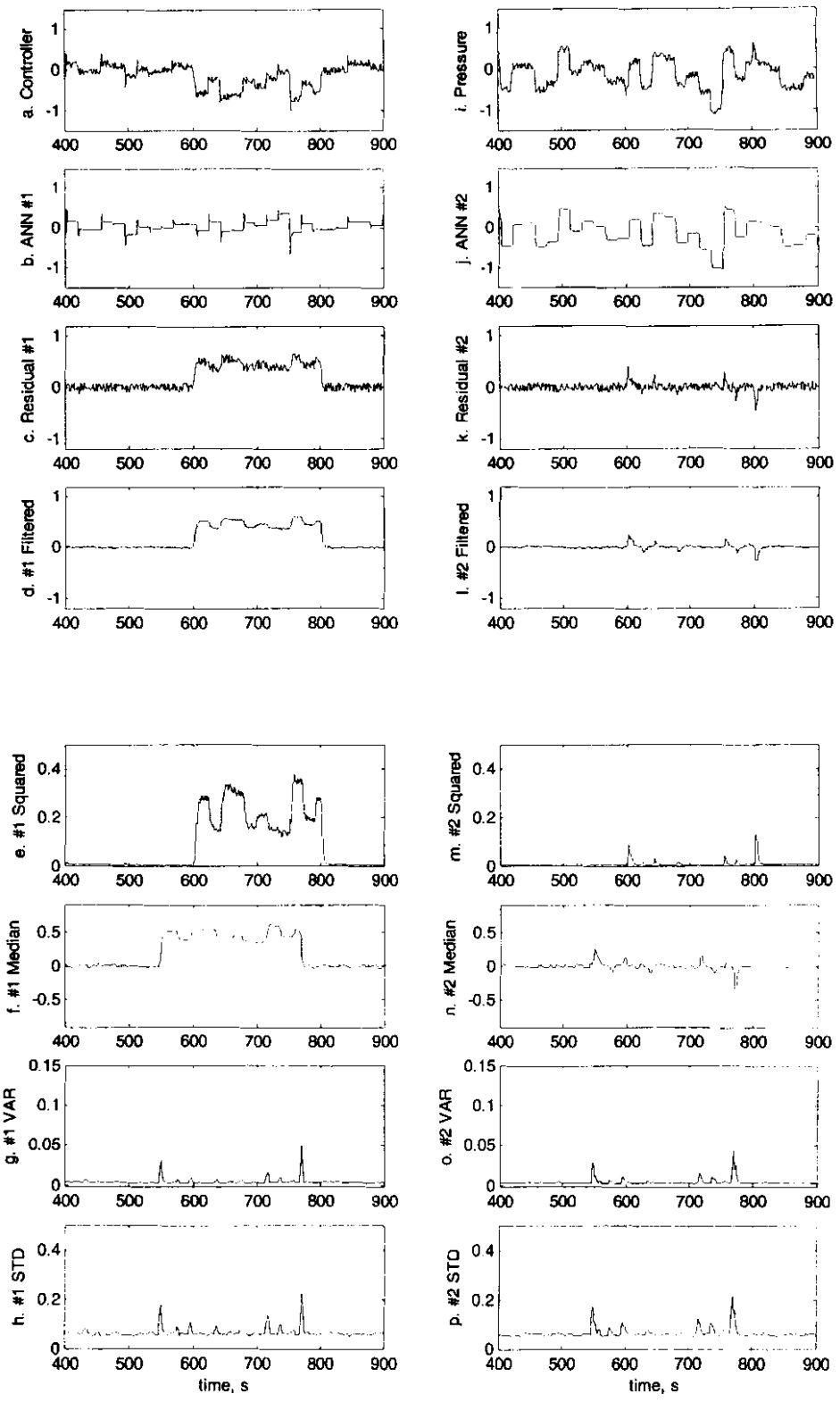


Figure 39. Residual signals in case of a leak in the ductwork

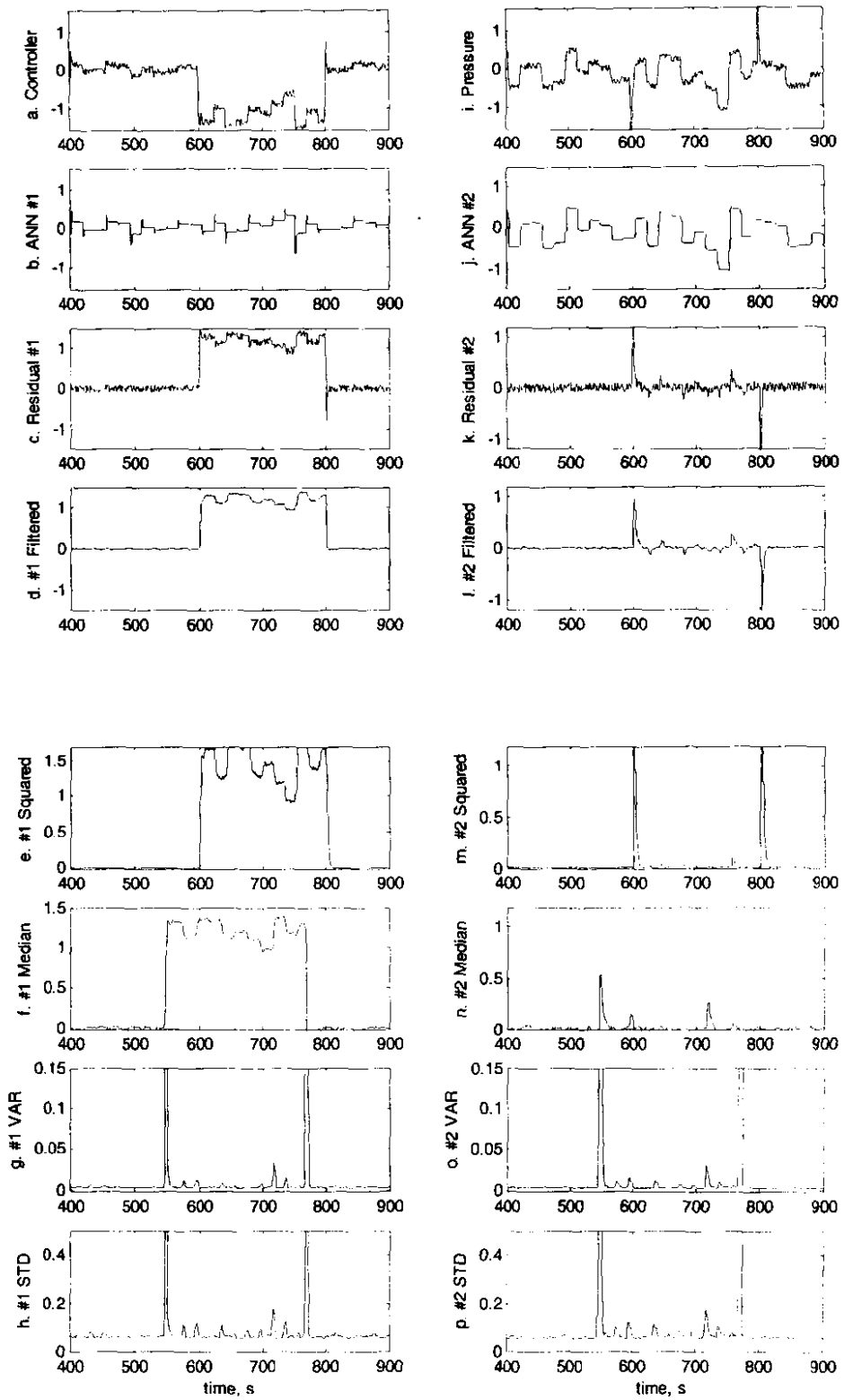


Figure 40. Residual signals when the probe suffered damage

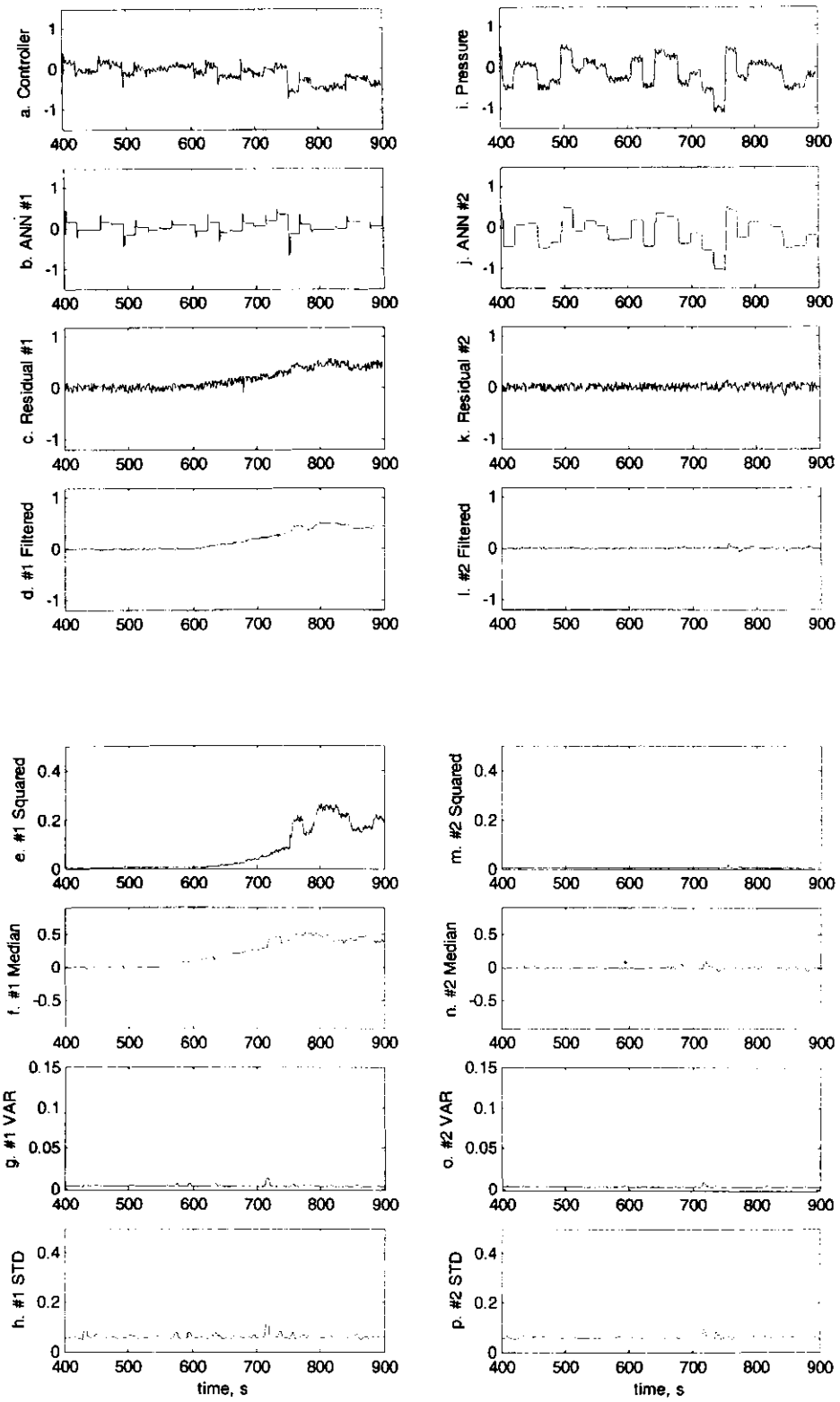


Figure 41. Residual signals for a puncture in the pneumatic delay line

As was done in chapter 4, a fault identification matrix can be deducted from the residuals. Table 4 contains the complete summary from Figure 37 to Figure 41. An 'X' is used to show then a fault would trigger a threshold and an 'O' is used to show that the fault shows no discernable spread on the corresponding graph.

Table 4. Fault identification matrix built from the graph set

	Residual #1						**Residual #2**					
	Residual	Filtered	Squared	Median	VAR	STD	Residual	Filtered	Squared	Median	VAR	STD
Normal operation	O	O	O	O	O	O	O	O	O	O	O	O
Controller failure	X	X	X	X	X	X	X	X	X	X	X	X
Damper clogging	X	X	X	X	X	X	X	X	X	X	X	X
Duct leakage	X	X	X	X	O	O	O	O	O	O	O	O
Probe damage	X	X	X	X	X	X	O	O	X	O	X	X
Pneumatic puncture	X	X	X	X	O	O	O	O	O	O	O	O

For fault detection purposes Table 4 functions sufficient; however, classifying between faults still remains an unaccomplished task. The problem is found in the fact that a controller failure and a clogging in the damper actuator fire the same triggers. Moreover, a leakage in the ductwork and a pneumatic leak also have similar trigger patterns. Further analysis of the residual graphs is required to differentiate between these faults.

The first issue to address is the task of separating control failures from damper actuator clogging. The reader is referred back to Figure 37.i. and m; as well as Figure 38.i. and m. In both cases the deviations are great enough to trigger the thresholds. However, in both the filtered residuals of graphs i. and as well as the squared filtered residuals of graphs m; the signals disagree sufficiently for a trained eye to differentiate between these failures. One method of separating these graphs is to apply a second filter to all four the mentioned residual transformations as was done in Figure 42. In Figure 42, graphs a. and b. show properties of a controller failure while

actuator clogging is shown in graphs c and d. A trigger around 0.1 in Figure 42.b would surely fire while the same threshold in Figure 42.d .would not trigger.

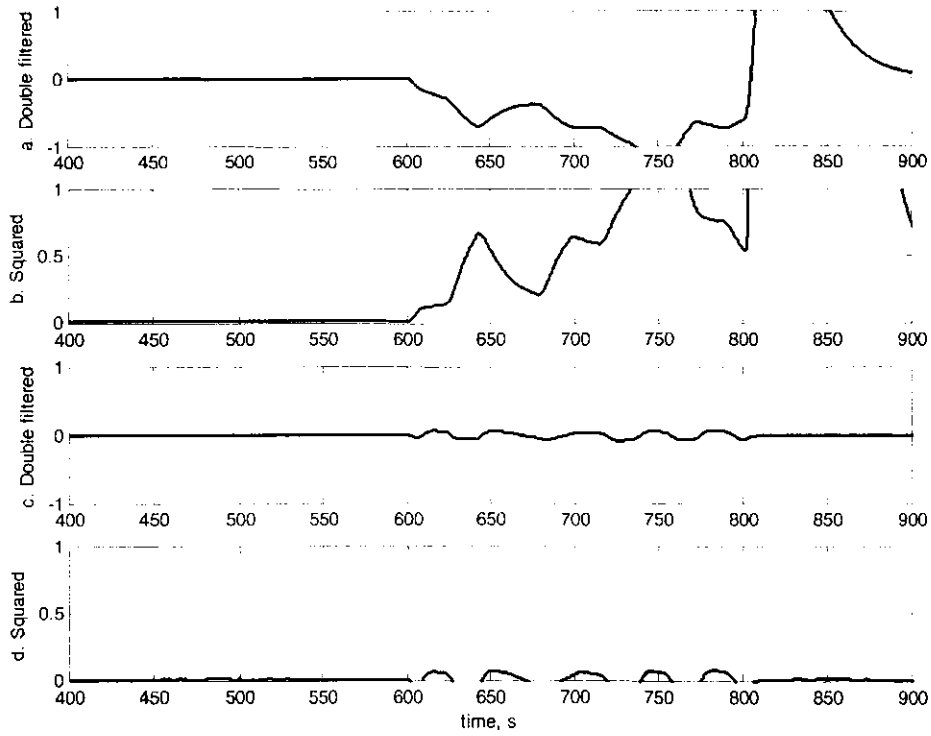


Figure 42. Residuals for separating controller failures from gains in actuator hysteresis

From Figure 42 it is now possible to extend Table 4 with another layer presented in Table 5.

Table 5. Fault identification matrix for discerning controller failures from damper build-up

	Residual #2	
	Double filtered	Squared filtered
Controller failure	X	X
Damper clogging	O	O

The second issue to attend to concerns separating abrupt air pressure leaks from incipient pneumatic leaks. The residuals of Figure 39.c. and d; as well as Figure 41.c. and d; have been redrawn into Figure 43. It is these graphs that differ significantly enough for FDD purposes. In Figure 43, graphs c, d, g and h. shows the rate of change (ROC) of the above-mentioned residuals. Taking into account that the incipient ROC-

graphs are drawn on a much larger scale, it would be easy to employ a threshold that separates these two incidents. Table 6 shows the deducted fault identification matrix.

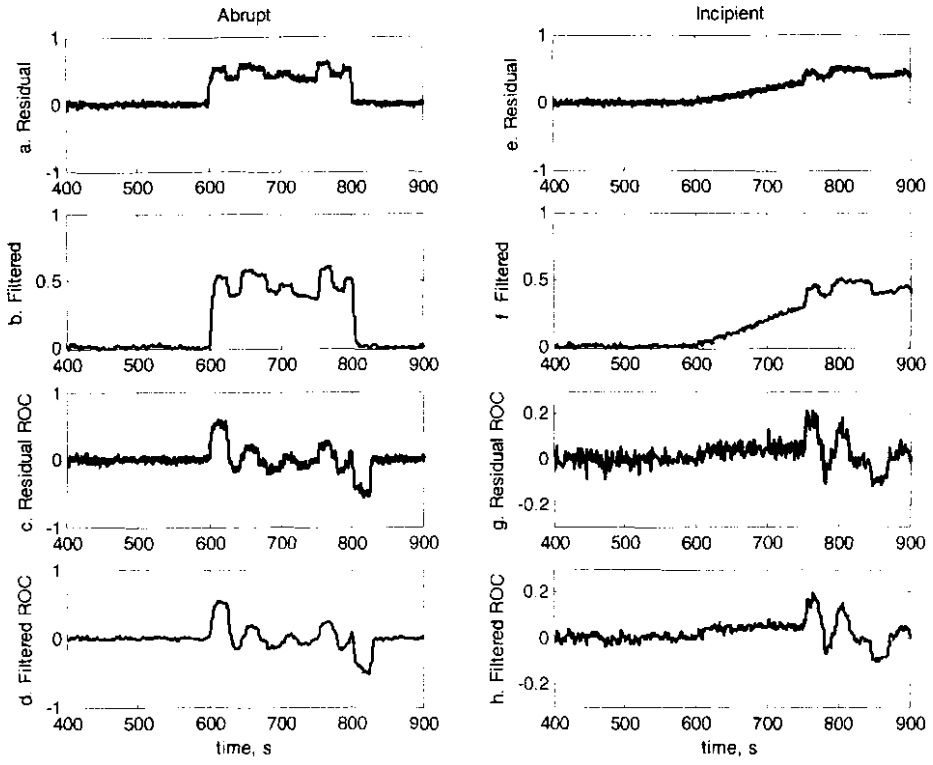


Figure 43. Differentiating to discern between abrupt and incipient faults

It should be mentioned that these ROC-graphs do not represent the instantaneous ROC. An instantaneous ROC would be aggressively dictated by the noise in the system. The ROC-graphs of Figure 43 show the change in the system over 25 seconds.

Table 6. Fault identification matrix for discerning abrupt leaks from incipient leaks

	Residual #1	
	ROC	Filtered ROC
Duct leakage	X	X
Pneumatic puncture	O	O

The result section can be concluded. Unique identification patterns have been developed for all the faults mentioned in this chapter. To summarize the core of the results, a FDD tree is constructed in the following section.

5.3 FDD tree for the static pressure loop

From the results in the previous section it is now possible to construct a complete minimal fault identification matrix. Excluded from this matrix is any residual adaptation that does not present new information to the system. In Table 7 an exclusive pattern for each fault can be found among the residual adaptations.

Table 7. Minimal fault identification matrix

	Residual #1			**Residual #2**	
	Squared	VAR	Filtered ROC	Median	Squared filtered
	$(R_1)^2$	$VAR(R_1)^2$	$f\left(\frac{dR_1}{dt}\right)$	$M(R_2)$	$f(R_2)^2$
Normal operation	O	O		O	
Controller failure	X	X		X	X
Damper clogging	X	X		X	O
Duct leakage	X	O	X	O	
Probe damage	X	X		O	
Pneumatic puncture	X	O	O	O	

As before, an 'X' represents an error that would trigger a threshold while an 'O' accounts for triggers that do not fire. The open spaces relates to untested areas. In these spaces it is insignificant whether a failure fires a trigger since other residual adaptations are used to identify the specific fault.

From the minimal fault identification matrix of Table 7 a binary decision tree (refer to section 2.6.2) can be constructed. See Figure 44. This type of tree structure assigns levels of importance to the different residual adaptations. In this case the most significant signal (or root event) is the squared residual. This signal's threshold deciphers if the system is running fault free or if there is a failure in the system. The least significant signals (or low level branches) are those with a fault type connected to them. The individual fault types are called the leaves of the decision tree.

For readability purposes the formulas given in Table 7 are used in the tree structure of Figure 44 to represent the different residual adaptations.

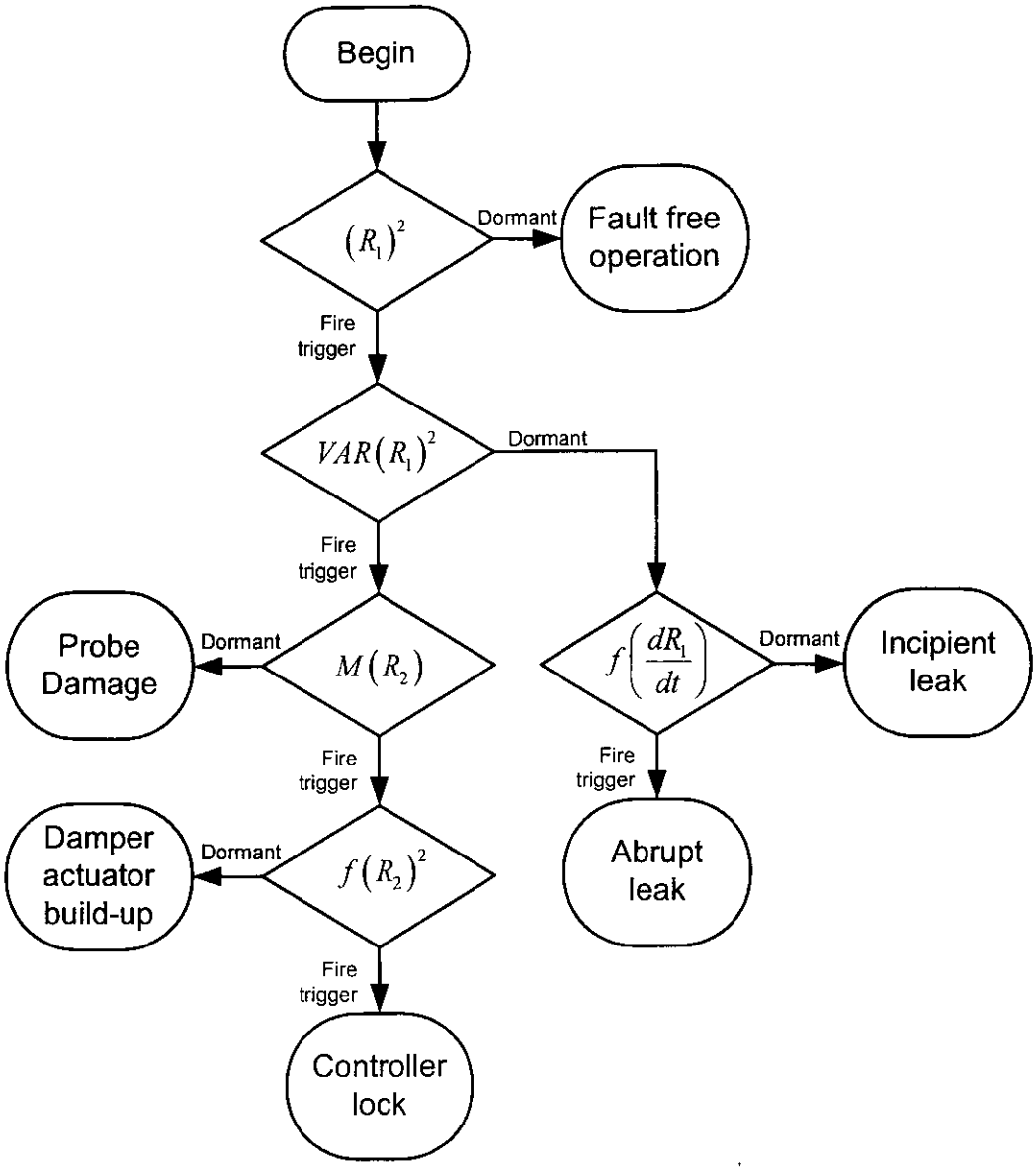


Figure 44. FDD tree for the static pressure loop

If these five faults were the only possible failures the static pressure control loop could experience, this FDD system could be considered perfect because tests have shown a 100 % fault detection and identification ability for faults diverging 10% off the normal operation value. Some faults were investigated, but the method applies to all faults that can reasonably be expected. In the next section, section 5.4, this FDD system is rated according to its abilities.

5.4 Properties of the FDD design

In the second chapter (section 2.3) a set of characteristics of FDD systems was discussed. Accordingly the deducted FDD system of this chapter shall be analyzed under the following attributes:

- Quick detection and diagnosis
- Isolability
- Robustness
- Novelty identifiability
- Adaptability
- Modelling requirements
- Implementation requirements
- Computational requirements

The other characteristics examined in chapter 2 aren't considered applicable to this particular study.

5.4.1 Quick detection and diagnosis

As can be seen from the FDD tree in Figure 44, faults are detected from the squared residual. Except for the incipient fault, all the mentioned errors cross the threshold in the first second of occurrence. The incipient fault first becomes detectable after 39 seconds when it deviates around 6.5% from the normal operation value. This time varies as the rate of the incipient error increases (in this particular case the incipient error rate of change was around 0.16 % each second). Because the thresholds are set at levels so that measurement and system noise do not trigger any alarms only errors straying more than 6% from normal operating values are detectable.

Diagnosis of the error occurs at the same instant as detection does. Thus FDD is instantaneous; only incipient errors remain undetected until the chosen 6% barrier is crossed.

5.4.2 Isolability

Isolability is the ability to distinguish between different failures. All the modelled faults are completely isolatable through the fault tree. However if an un-modelled fault is detected by the tree it would be misclassified as one of the modelled faults.

5.4.3 Robustness

Due to the levels of filtering in the FDD system, neither system noise, nor measurement noise should trigger any false alarms. The threshold levels were selected to handle noise levels around 15% to either side of the true measurement. This system is considered highly robust.

5.4.4 Novelty identifiability

The FDD tree of Figure 44 has null leaves, which points to unidentified or new fault types. Although a deviation from the normal operation conditions would be detected, the decision tree would classify the fault among the known set. This diagnosis system was build only to identify known error and accordingly fails novelty identifiability.

5.4.5 Adaptability

There are two areas under which adaptability should be judged. First is the systems reaction to environmental changes like the replacement of equipment that causes permanent adjustments of transfer equations. Such adjustments should be altered in the ANNs. A package like CSense® provides an on-line trainer; accordingly the neural networks are updated as the process continues to function normally.

Secondly, the fault tree should be extended as new error types are recognized. To extend the fault tree a unique residual pattern needs to be identified. This might require that a new residual adaptation is incorporated into the fault tree. A new residual adaptation would also require a new threshold level to be deducted. All this can be done without the need to alter the current tree in use making this a highly adaptable FDD system.

5.4.6 Modelling requirements

Training of the neural networks requires an efficient set of process history data. It doesn't require any knowledge of the HVAC system at hand. To build the decision tree also required historical fault data. No HVAC specialist is needed. To sum up, the FDD system requires sufficient history data. If enough data are available before hand deploying the FDD system requires minimal resources.

5.4.7 Implementation requirements

To implement the FDD system developed in this chapter, it is critical that the ANN models are trained with data that represent the complete scope of normal operation. If all the product variables (on which the FDD system is implemented) are known, it is simple to generate the required training data with the HVAC model of the static pressure loop.

The second phase of implementation requires malfunctions to be simulated via the trained ANNs. Different residual patterns can be built from the network outputs. From the patterns the fault identification matrix and the binary decision tree can be deducted.

A developed FDD system should be exportable to similar HVAC systems built from the same components without any alterations. However, if components differ greatly between systems, the ANN models require retraining.

5.4.8 Storage and Computational requirements

For storage requirements the two neural networks, collectively, weighed in below 110 kilobytes. All the needed process history information is contained in the trained parameters of the networks. The decision tree information covers a fraction of the ncural network requirements.

Thanks to the parallel configuration of the ANN computational costs are minimal. The *IF-THEN* structure of the decision tree also reduces computational requirements. The largest calculations are required for residual adaptation and even this is

considered tiny for today's processors. Another contributor to computational requirements is the rate used for sampling the data. The sample rate of 10 samples per second is considered low, making computational requirements minimal.

5.5 References in chapter 5

[1] Watton, A., Marcks, R.K., & Solem, G. *Improved Instructional Techniques in HVAC Control Systems.* Sinclair Community College. Dayton, OH.

[2] Yu, B. & van Paassen, A. H. C. *Modeling with simulink and bond graph method for fault detection in an air-conditioned room.* 2001. Lab of Refrigeration Engineering & Indoor Climate Control. Delft University of Technology.

[3] Goodfellow, H. & Tahti, E. *Industrial ventilation design guidebook.* 2001. Academic press.

CHAPTER 6

6 FAULT DETECTION AND DIAGNOSIS ON HVAC ENERGY CONSUMPTION

The fundamental purpose of an HVAC system is to sustain a comfortable climate inside a space. Usually this is different to the climate outside the space. To maintain this difference the HVAC system requires energy. This study looks at the relation between the inside and outside temperatures and the energy consumed maintaining it.

In the previous chapter, an FDD system was developed on a simulation model of an HVAC sub-system. This simulation model was built via first principal methods from physical models. Most of the contributing factors to the HVAC system studied in this chapter are unknown. Thus, a first principal model can not be developed.

The FDD system, discussed in this chapter, is constructed around two neural networks. The networks are trained with process history data. The outputs from the networks are analyzed to detect and diagnose faults in the system. At the end of the chapter the FDD system is evaluated on various aspects such as robustness and diagnosis speed. The ANNs as used in this chapter were implemented using CSense®, developed in South Africa. A short discussion is given in appendix A.

6.1 Background

In essence, an air-conditioning system is used to establish and maintain a temperature difference between the outside and the inside of an enclosure. Naturally, this process requires energy. The energy levels needed relate do the temperature difference required. Accordingly, the power used by the air-conditioning system is some function of the indoor and outside temperatures.

For an air-conditioning system the power level needed to maintain an indoor temperature is proportional to the heat transfer out of the conditioned region [1]. This heat transfer depends on the specific surroundings of the air-conditioned space. Figure 45 shows the first layer of elements surrounding the air-conditioned space.

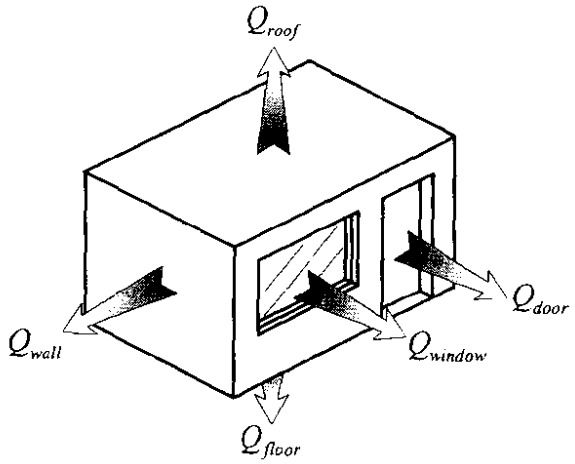


Figure 45. Various heat transfer elements separating the conditioned space from the outside

The total heat loss is the sum of the heat losses through all the surface elements:

$$Q_{total} = \sum_{n=1}^{N} Q_{roof}^{n} + \sum_{m=1}^{M} Q_{wall}^{m} + \sum_{p=1}^{P} Q_{floor}^{p} + \sum_{r=1}^{R} Q_{other}^{r} \quad \ldots (6.1)$$

Here Q^n, Q^m, Q^p and Q^r are the heat transfer of a specific surface segment. For example, a single wall heat transfer segment is modelled with:

$$Q_{wall} = Q_{conv} + Q_{abs} + q_{lw,sky} + q_{lw,ground} \quad \ldots (6.2)$$

Q_{conv} represents the convective heat transfer through the wall, while Q_{abs} represents the absorbed solar radiation. $q_{lw,sky}$ and $q_{lw,ground}$ are models of the long wave radiation between the exterior surfaces and the sky or ground. All the components of (6.2) need

to be calculated from physics equations. Consequently, calculating (6.1) becomes a tedious process relying on fine detail of the environment. This detail includes knowledge of materials used in the construction and building dimensions, to solar patterns and sun angles.

6.2 Methodology

As an alternative to building a model around (6.1), this chapter focuses on an artificial neural network (ANN) that models the air-conditioning power levels from the indoor and outside temperatures. The advantages of an ANN over first principles models in building energy FDD are:

- No expert knowledge of the building or its environment is required. This is because ANN models are built from historical data. These data are discussed in the next section.
- As first principal methods lead to a solution for a highly specific building, ANN based FDD remains a more universal solution that can be deployed in many buildings. However, the ANN needs to be retrained for the particular building.

The drawback is that for a highly accurate ANN model, the data of at least one complete seasonal cycle are required.

6.2.1 System input and output

The data used in this chapter were collected from a telecommunications exchange site. A description of the facilities is provided to give the reader an idea of some of the interactions with the environment. The telecommunication equipment is distributed through some rooms located on the third floor of a building. These rooms are cooled by a dedicated HVAC system. Due to the regular intervention of maintenance personnel the rooms are not kept as isolated as they should be.

System requirements state that the equipment should operate below 22 °C. From the measurements it seems that the operating indoor temperature is set around 18 °C. This might be an overcompensation to avoid high temperature alarms.

The data were measured at 30 minute intervals. Only three measurements were taken into consideration. These are the two temperature readings in Figure 46.a and the instantaneous power usage of the HVAC system in Figure 46.b.

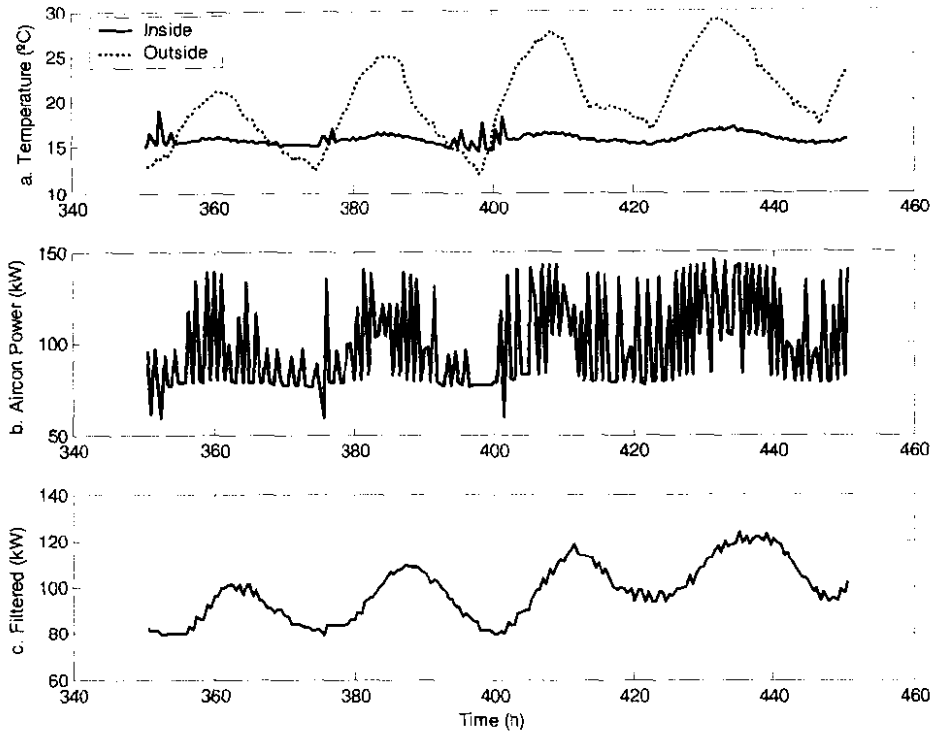


Figure 46. HVAC system input and output

The fluctuations in the instantaneous power (Figure 46.b) render it unpredictable. A neural network trained to predict the data delivers a MSE value around 0.2. This is unacccptable p erformance f or F DD p urposes. A moving a verage filter w as a pplied stretching over six hour's measurements. This filtered power graph is displayed in Figure 46.c. Neural networks trained to predict the filtered data delivers MSE values of 0.003.

Throughout the rest of this chapter the HVAC power refers to the filtered power symbolized with P_{aircon}. The outdoor and indoor temperatures are resembled with $T_{outside}$ and T_{inside} respectively.

6.2.2 FDD solution

Three measurements are studied in this chapter. The relationship between these measurements are considered unknown, although it can be calculated via the method mentioned in section 6.1. In Figure 47 an ANN mimics the air-conditioning power by mapping the two temperatures. The ANN output is compared to the measured power to generate a residual. As in the two preceding chapters, this residual is studied for FDD purposes.

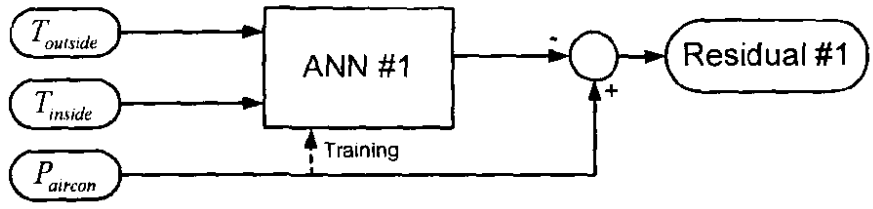


Figure 47. Basic FDD architecture

In the preceding chapters two residuals were used to give the FDD system a higher resolution. The current problem requires at least two of the three measurements to build an effective ANN. The third measurement is needed to build the residual. Thus, all the measurements are used. In order to increase the range of the FDD system proposed in Figure 47 a second ANN is added. This configuration is shown in Figure 48. The additional ANN will cross validate the results from the original ANN.

The indoor temperature is used as a common input to either network. This was chosen because T_{inside} has the lowest variance and usually operates around a set value. $T_{outside}$ serves as an input to ANN #1 and is used to deduct residual #2 by subtracting the output of ANN #2. P_{aircon} is employed in the opposite role. It is used in comparison with ANN #1 to form residual #1 and is used as an input to ANN #2. See Figure 48.

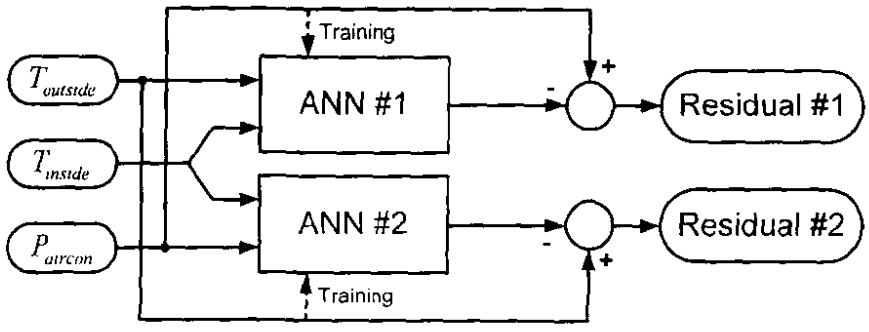


Figure 48. Cross validating FDD architecture

Training the two neural networks of Figure 48 is thoroughly discussed in the appendix of the dissertation. In section A.4.2 it is found that a 48–36–6–1 network trains best for ANN #1, while a 48–40–5–1 network architecture should be used for ANN #2. Both the networks receive a 48 delayed line input of 30 minute samples. This is a 24 hour cycle of data. Refer to Figure 46.

6.2.3 Fault list

Neural networks are considered black box models. It is necessary to study how certain input sets map to an output in order to gain any understanding of the network at hand. Once the two networks of Figure 48 are trained with normal (fault free) operating data, the response to data containing errors can be studied. From these responses FDD criteria can then be deducted. In the results section the two residuals are analyzed to distinguish between the faults. In this section the various faults used during the remainder of this chapter are mentioned.

Ideally, all the faults the FDD system should differentiate between should come from gathered measurements. The HVAC system should have undergone through all the failures and these data are then used to build the FDD system. Unfortunately, the collected data do not include enough failure data. No experimentation was allowed due to the potentially high cost of down-time of telephone exchanges. Faults included in the data are:

- Simultaneous failing of the temperature sensors (data link broken)
- Compressor failing
- 3rd party heat source (lights were left burning)

To increase the resolution of the FDD system more failures need to be studied. Some artificial faults were constructed from the ideas of sections 2.2 and 4.2.2. The value of artificial faults is that they can be sized to test the limits of the FDD design. The artificial faults used to broaden the FDD system are:

- Gain on P_{aircon}
- Incipient offset on P_{aircon}
- Abrupt offset on P_{aircon}

- Incipient offset on T_{inside}
- Abrupt offset on T_{inside}
- Gain on $T_{outside}$
- Incipient offset on $T_{outside}$
- Negative abrupt offset on $T_{outside}$

A gain on T_{inside} was not considered because this input varies little around a significant positive value. A gain delivers the same results as an offset.

6.3 Results

The same approach used in the corresponding section of the previous chapter is used here. The mentioned faults are passed through the two neural networks and subtracted from the third input to generate the residuals – refer to Figure 48. The residuals are then transformed to obtain useful spreads. The transformations include filtering and squaring the residuals, calculating statistical alterations like the mean, median and the variance and sifting to produce constant rate of change. Evidently, numerous graphs are required to display all this information.

6.3.1 Faults from the measured data

Figure 49 shows outputs from the two neural networks in comparison with measured target variables. The bolded lines represent neural network outputs while the thinner lines represent the measured data. The left-hand graphs contain the elements to build residual #1. This is the output from ANN#1 and the HVAC power measurement. On the right-hand side the outdoor temperature measurement and the output from ANN #2 are compared. This is the configuration used during the remainder of this section.

Figure 49.a. and d. show the spreads for a failure with the compressor. As the input variables – mainly P_{aircon} – move past the input limits, the neural networks produce outputs outside the acceptable range. The ANN output is considered low quality and it is thus uncertain if the network would reproduce a similar output for a similar input.

This is a feature built into CSense®. Refer to appendix A for more information. A lack of output prohibits the construction of residuals.

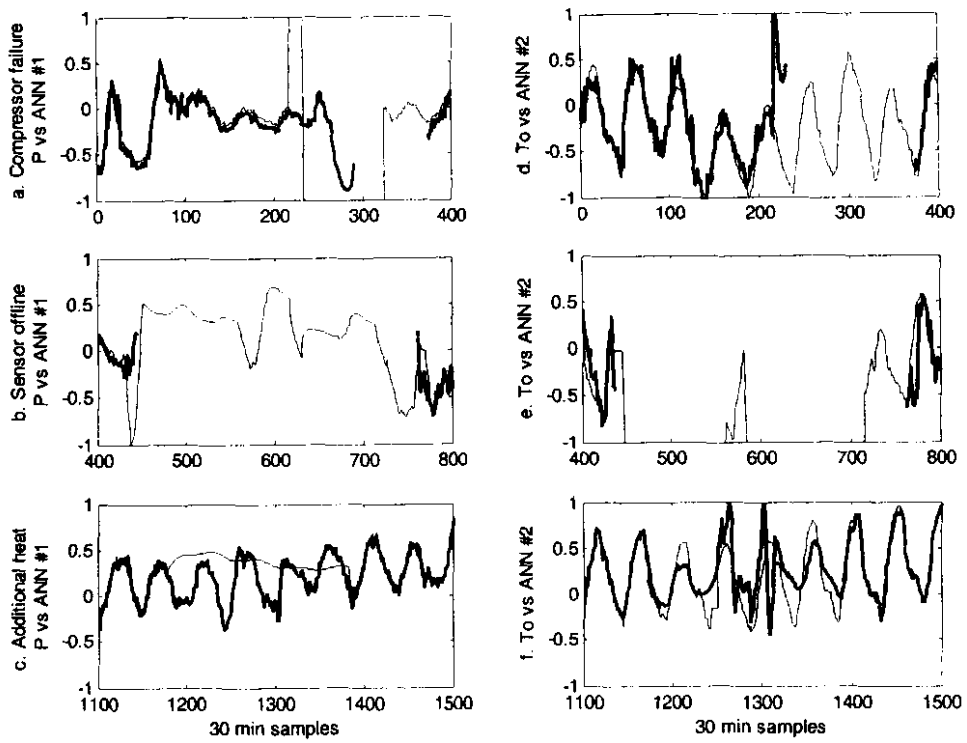


Figure 49. Neural response to failures in the measured data

The same occurrence can be seen in Figure 49.b and e. which contain the graphs for a complete sensor failure. Once again both the ANN # 1 and ANN # 2 fail to deliver good quality data and no residuals can be drawn. For the outside temperature measurement a sensor fault is about the only possible failure as this is an uncontrolled variable. Both the above mentioned failures remain detectable even when no residuals are available. This is because the measurements lie beyond the normal operation scope.

The third fault found in the measured data was the occurrence of a 3rd party heat source. The indoor temperature does not deviate from the set value. The compensation for the added heat is made in the HVAC system, hence ANN #1 differs from the P_{aircon} measurement in Figure 49.c. Figure 49.f. also shows a distorted ANN #2 output as P_{aircon} is an input to the network.

In this case the failure did not move operations beyond the scope of the network and residuals can be obtained for study. See Figure 50. Graphs a. and i. are copies of Figure 49.c. and f. – the components from which the residuals are built. The residuals are shown in Figure 50.b. and j. Here the system noise is still highly visible, making these two graphs less valuable for FDD purposes. A more FDD orientated graph would be a filtered residual from which the system noise had been removed. Refer to graphs c. and k. Figure 50.d. and l. show how the squared filtered residuals that enhance the error spread and restrict calculations to the positive side, simplifying the selection of trigger levels.
The next six graphs, Figure 50.e, f, g, m, n, and o. shows the necessary statistical manipulations of the residuals. It is the mean, the median and the variance (VAR) of the residual calculated over a window of 24 hours.

The last graphs found as part of Figure 50 shows the rate of change (ROC). They have been modified to discard abrupt changes and only lift out steady constant changes in the filtered residual.

Due to the immense scale of this error, almost all the graphs show detectable abruptions. Only the ROC graphs remain dormant due to the varying fluctuations in the residuals. This fault is more identifiable through its shear size than through the repercussions it causes.

In the preceding chapters corresponding graphs were drawn on identical scales. Spreads in this chapter differ too greatly, thus the reader is cautioned to consider the scale of each graph individually. Notice also that some graphs, like Figure 51.g. have a scaling factor on the upper left corner.

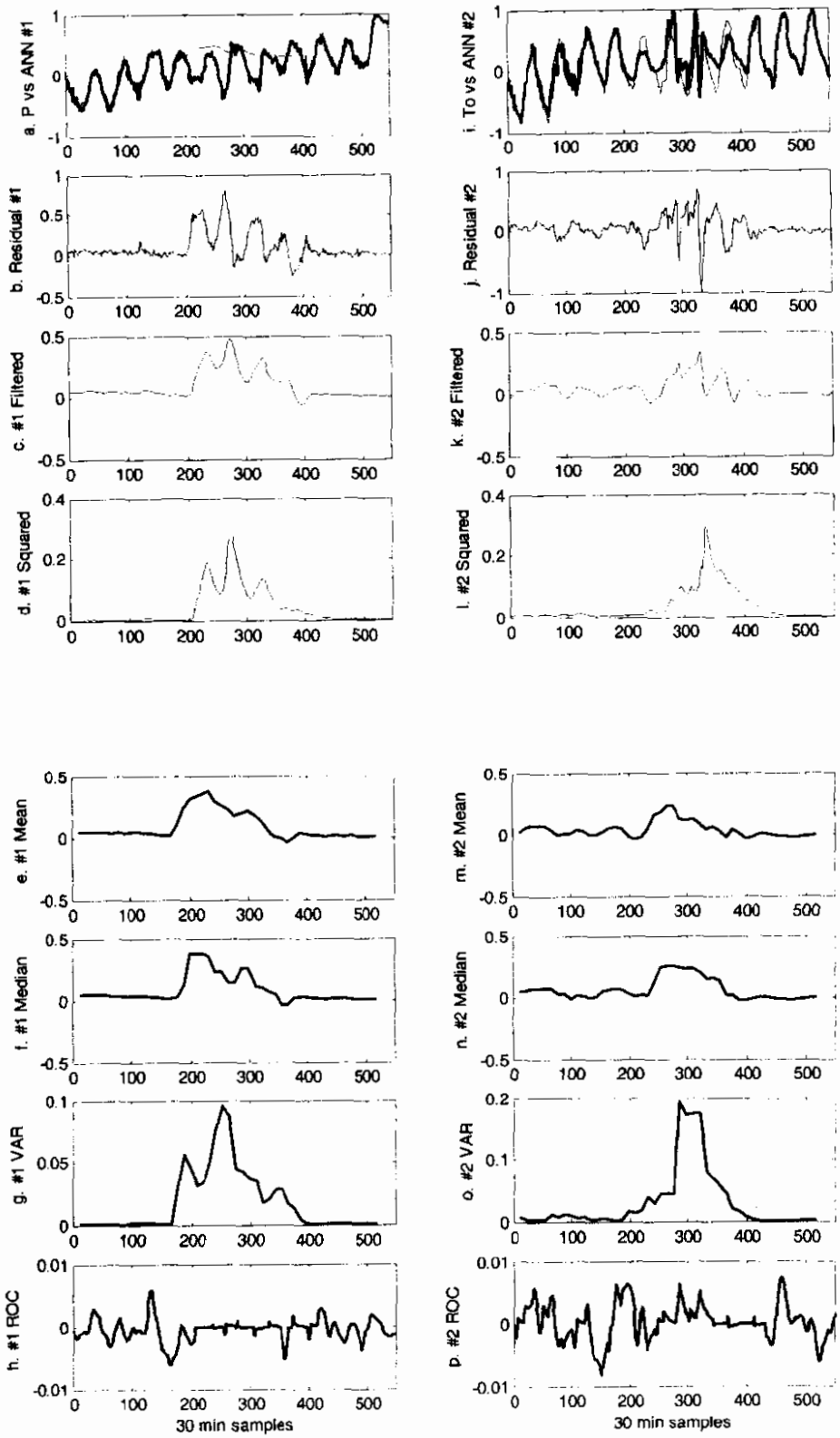


Figure 50. Residual analysis for an unaccounted heat source inside the building

6.3.2 Artificial offset faults

All the artificial faults start after 200 intervals, extend for 300 measurements and end instantly after 500 intervals. The three abrupt offsets, displayed in Figure 51, Figure 52 and Figure 53, were created by increasing one of the signals with 10% of the average signal value.

Figure 51 shows the residuals for an offset error on the power signal. The sudden increase can clearly be seen in the residual graph, Figure 51.b. The fault spread becomes even more detectable in graphs c. and d. The abruptness, however, decreases due to the filtering algorithm used. This causes the slope in the fault spread between 200 and 230 in both graphs c. and d. The filtering algorithm used is:

$$y_1 = x_1 \qquad y_n = K_s x_n + (1 - K_s) y_{n-1} \qquad n = 2,3,4\ldots$$

where y is the filtered signal, x is the unfiltered signal and K_s is the filtering factor. $K_s = 0.05$ was selected for this study.

On the right-hand side of Figure 51, none of the graphs shows an initial disturbance other than system noise. The offset error is thus only detectable in residual #1. It propagates through the mean and median graphs as well (see Figure 51e. and f.). The variance (graph g.) shows the edge conditions of the offset error. The ROC graph (graph h.) does not show any slow constant increase in the residual.

The disturbance found in Figure 51.j. beyond the 480 mark, is the result of bad quality output from ANN #2.

Figure 52 contains the results for an offset error in the inside temperature. Because the inside temperature has a low variation, both neural networks are influenced greatly by the 10% offset. This can be seen in graphs a. and i. All the graphs, except the ROC graphs, show highly detectable divergences. The great variance, caused by the fault, is found in both VAR-graphs, graphs g. and o. The ROC graphs also point to highly variable residuals by not being able to detect any constant changes.

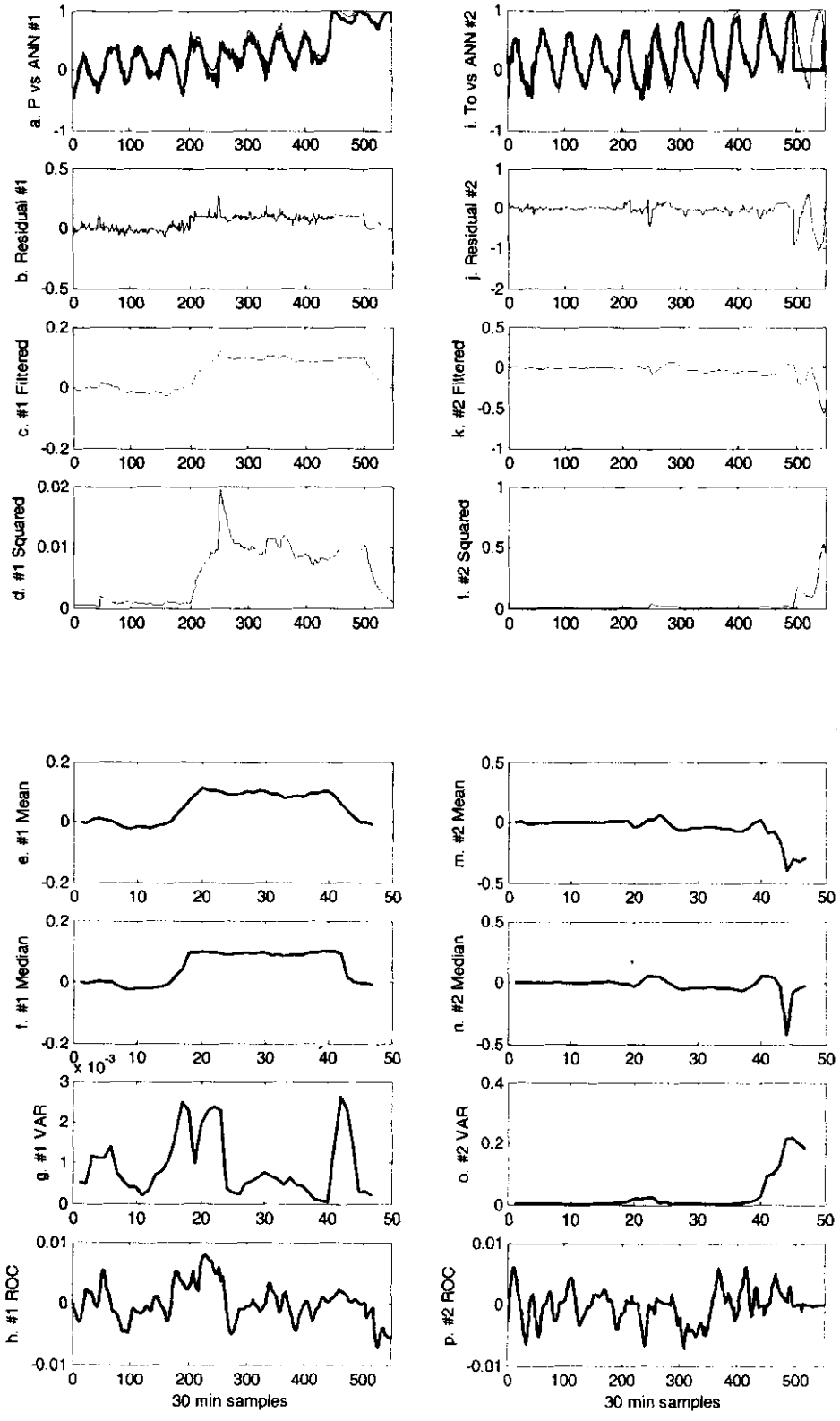


Figure 51. Residual analysis for an *offset* error on the *power* measurement

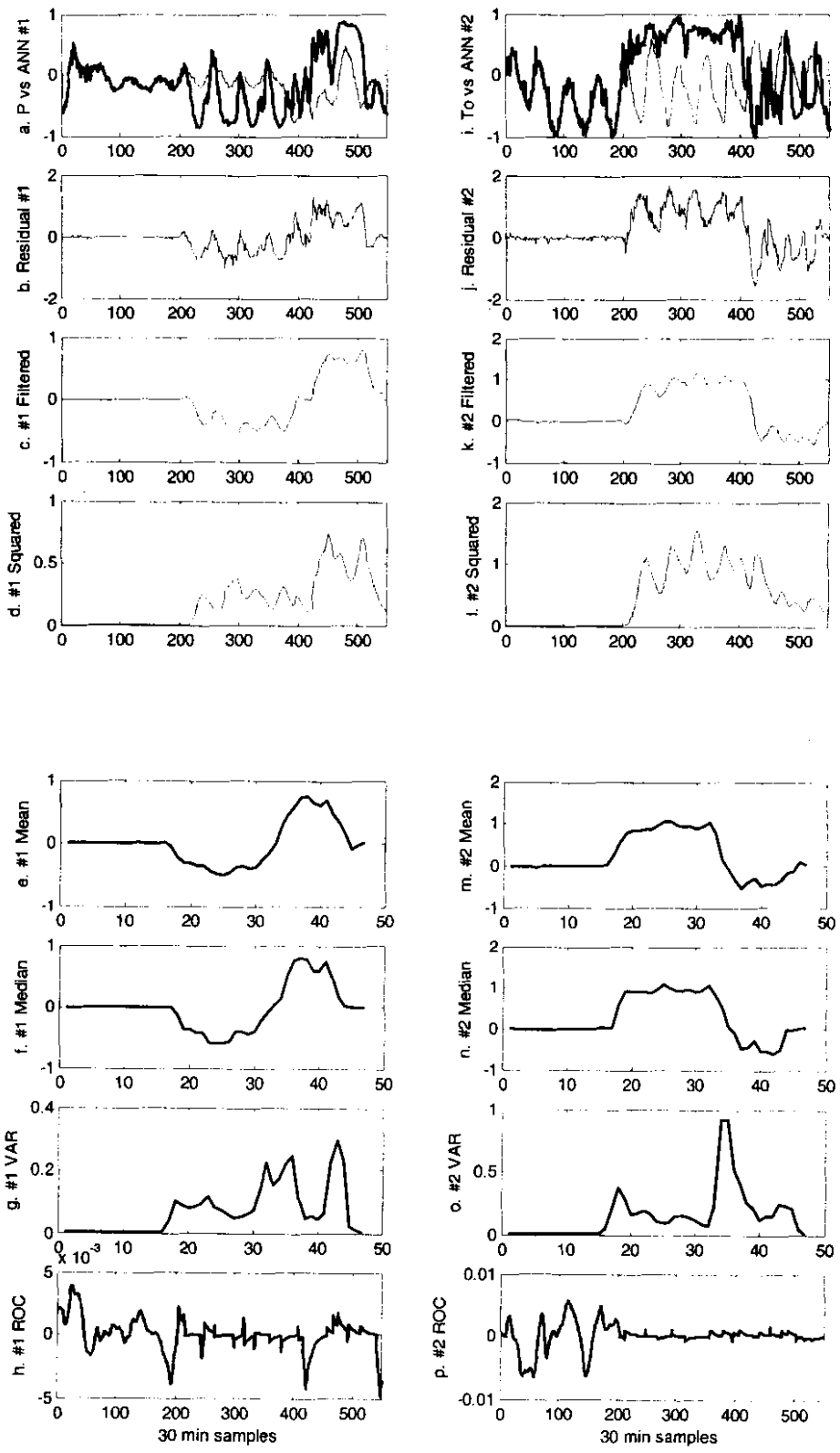


Figure 52. Residual analysis for an *offset* error on *inside temperature*

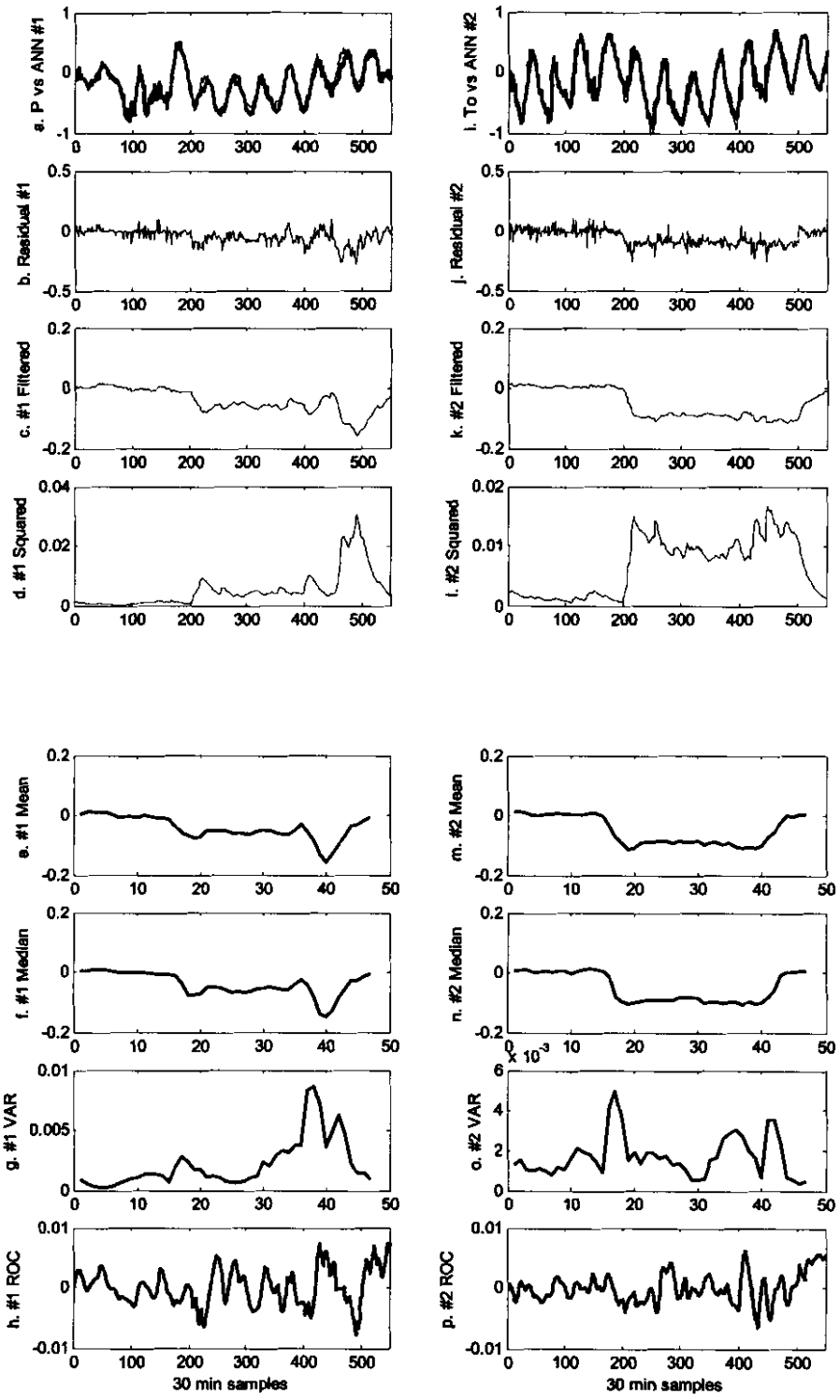


Figure 53. Residual analysis for an *offset* error on the *outdoor temperature*

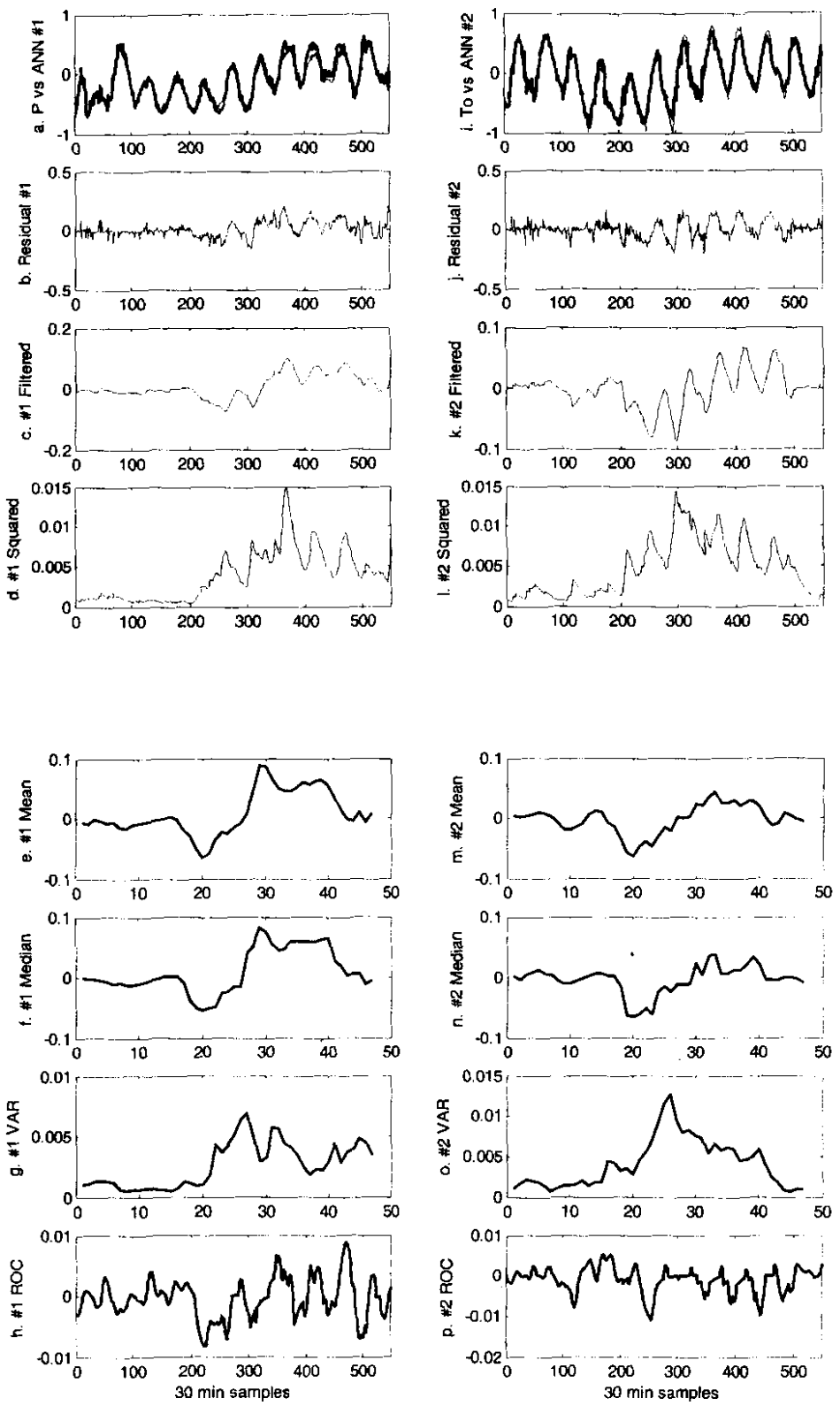


Figure 55. Residual analysis for a *gain* error on the *outdoor temperature*

Figure 53 displays how a negative offset on the outside temperature would shape the residuals. The sign and size of the error can clearly be seen in the residual (graphs b. and j.), the filtered residuals (graphs c. and k.), the mean (graphs e. and m.) and the median (graphs f. and n.). Only around the edges of the fault do the VAR and ROC adaptations reveal any detectable offsets. As expected, the squared residuals (graphs d. and l.) are not able to obtain information about the sign of the error. The squared residuals do, however, elevate the error most. This makes the squared residual ideal for detecting faults, but not for diagnosing faults.

6.3.3 Artificial gain faults

In Figure 54 a gain of 10% was added to P_{aircon}. This error leads to the fluctuating appearance of residual #1 seen in graph b. Other attributes of this type of error, which correlates with the findings in chapter 4, is a low mean and median and a detectable VAR. The ROC (graph h.) also shows regular changes in the slope of the filtered residual signal. Residual #2 also shows detectable disturbances in the same graphs as residual #1 did.

The residuals displayed in Figure 55 show a 10% error in the outside temperature measurement $T_{outside}$. The output from ANN #1 differs barely from P_{aircon} (graph a.). In graph b. it is also hard to distinguish the faulty area from the faultless area. The filtered residual shows a more discernable spread (graph c.), and as was found with the offset faults, the squared residual shows the most detectable spread (graph d.). Residual #2 shows a slightly more detectable error region than residual #1 did, but in general the graphs of residual #2 are similar to the graphs of residual #1.

This gain error also shows low means (graphs e. and m.) and medians (graphs f. and n.) on either residual. The variance (graphs e. and m.) displays detectable fault areas and the constant ROC shows regular changes in the residual.

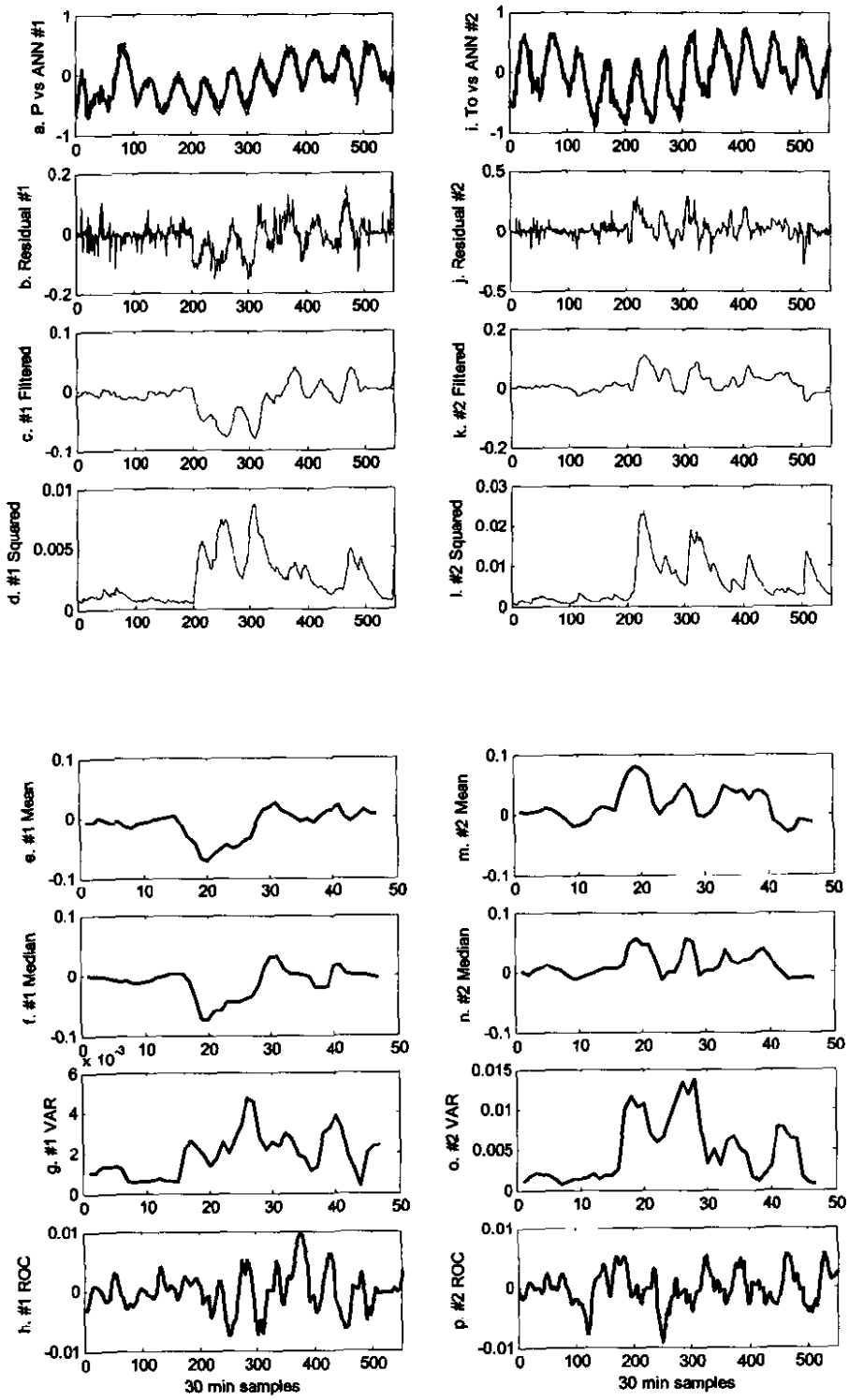


Figure 54. Residual analysis for a *gain* error on the *power* measurement

6.3.4 Artificial incipient offset faults

The incipient offset added to each measurement starts at the 200th interval with a value of zero. As time progresses the error increases linear to 33 % at the 500th interval. The outline of this error is clearly caught in Figure 56.b. and Figure 58.j. It is for these types of errors that the ROC of the residual is studied. In Figure 56.h. and Figure 58.p. the constant ROC shows a constant increase in the residuals.

Figure 56 shows the residuals for an incipient error in P_{aircon}. The shape of the error is clearly visible in graphs b. to f. The VAR (graph h.) displays the step when the error is restored. This information is useless since the error needs to be detected before it can be repaired. Thus graph h. shows no detectable spread. In residual #2 the sloping nature of the error is slightly distorted. But the incipient character remains detectable in graphs j. through n.

As Figure 57 shows, an incipient error on the indoor temperature measurement, T_{inside}, shows a much larger offset in the residuals than the same size error in either P_{aircon} or $T_{outside}$. Around the 400th measurement the error becomes too great for the neural networks to deliver good quality data. Both residuals show significant error spreads in all the graphs except for the ROC signals (graphs h. and p.).

Figure 58 seems to be a mirror image of Figure 56. In Figure 58 the incipient shape can be seen in the right-hand side (graphs j. through n.). The variance (graph o.) is overshadowed by the large spike that represents the end of the offset. The ROC (graph p.) points to a constant steady increase in the residual. Residual #1 does show a detectable error but it does not seem to be of an incipient nature. This residual has a high variance (graph g.) and an inconsistent ROC (graph h.).

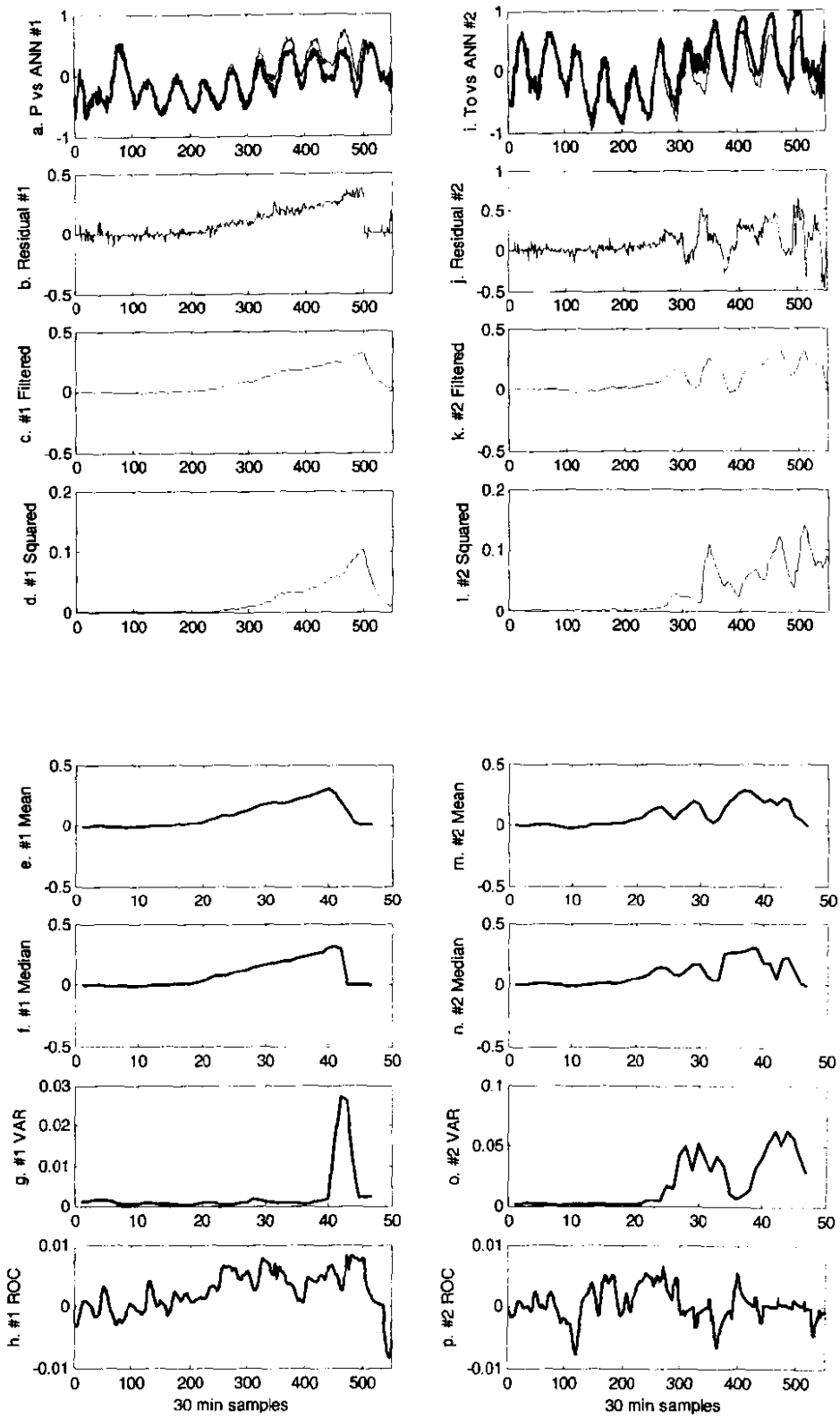


Figure 56. Residual analysis for an *incipient* error on the *power* measurement

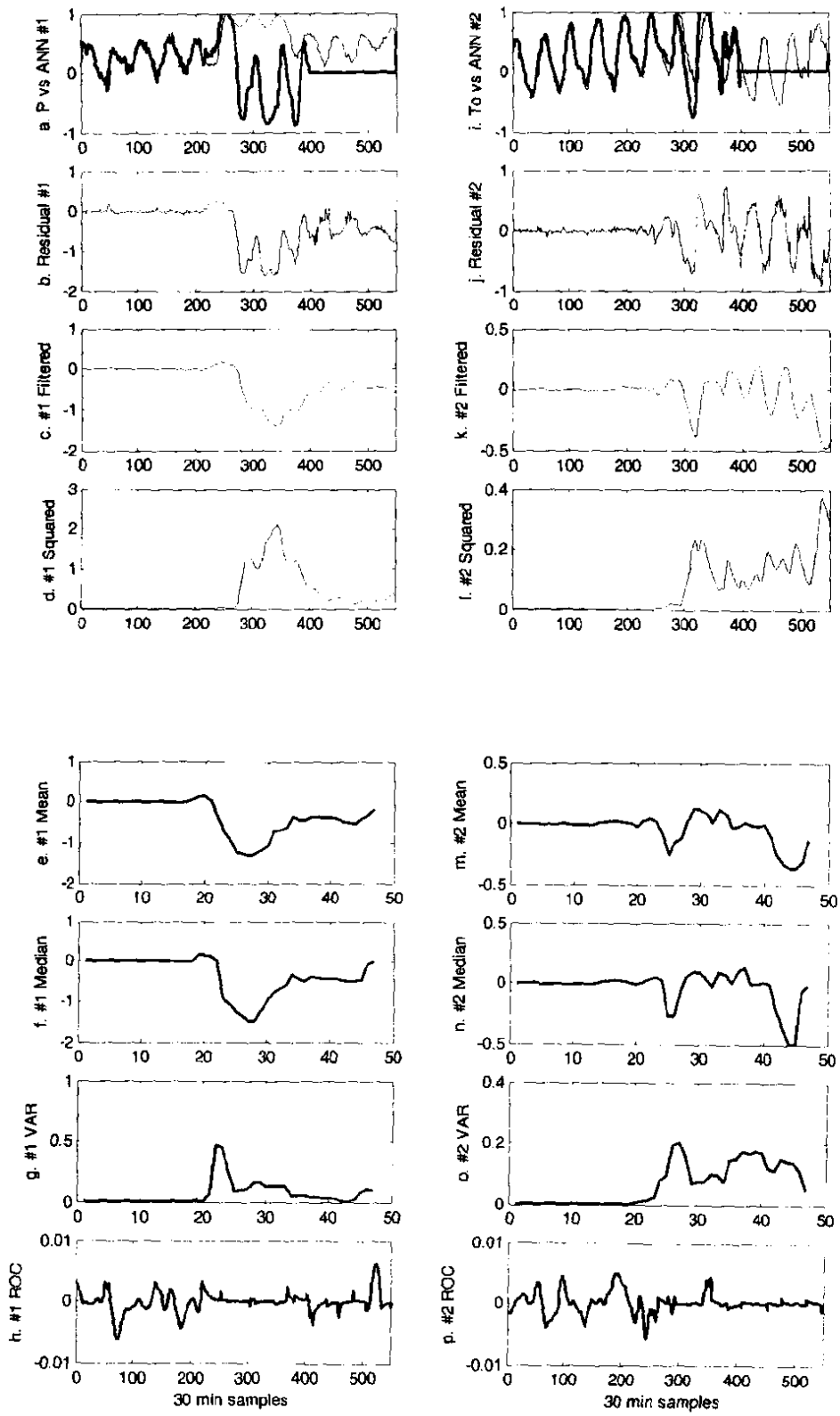


Figure 57. Residual analysis for an *incipient* error on the *inside temperature*

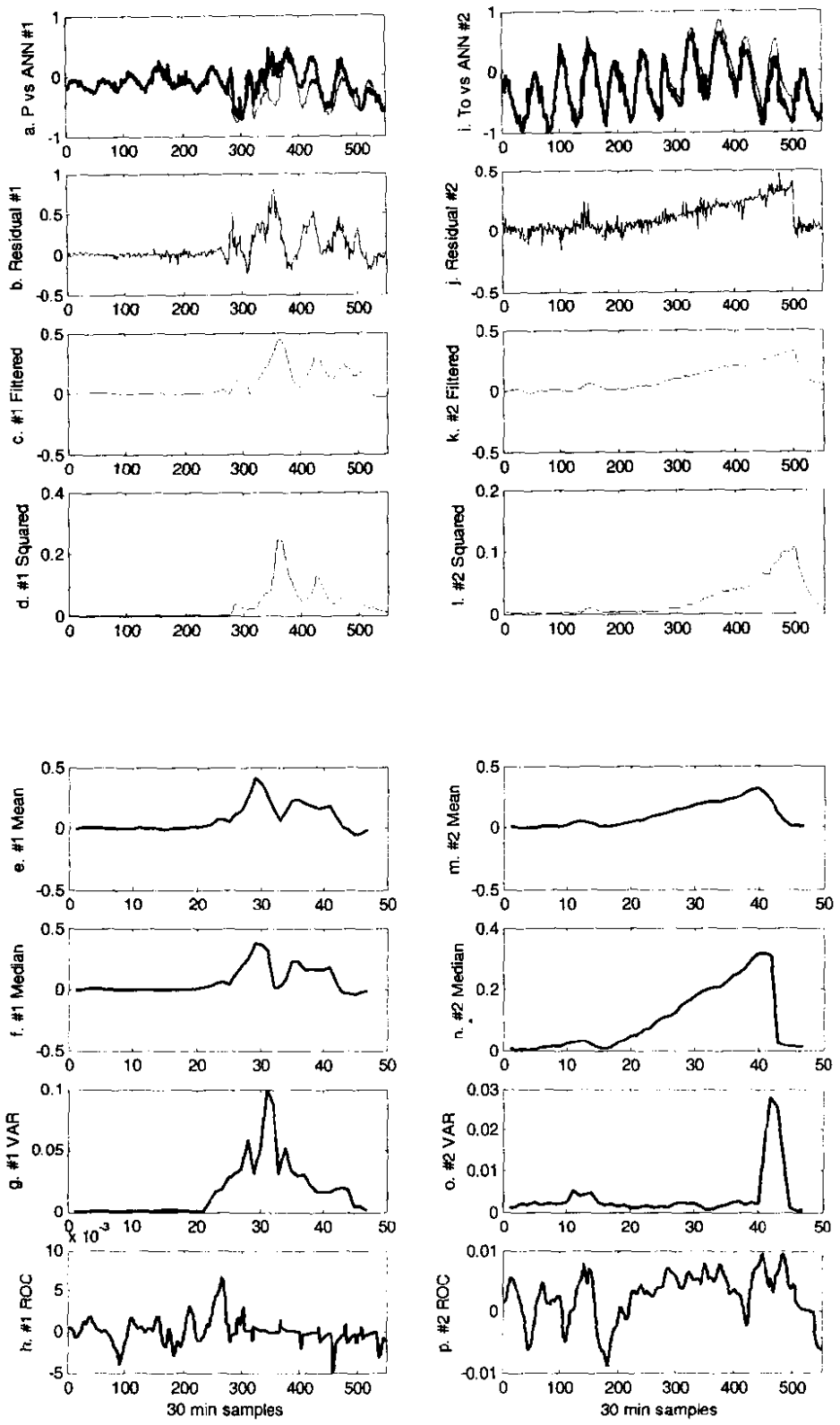


Figure 58. Residual analysis for an *incipient* error on the *outdoor temperature*

6.3.5 Fault identification matrix

After the tedious task of analyzing the residuals, a fault identification matrix can be constructed. Table 8 can be considered a summary of the residuals graphs. Ideally, each fault studied has a unique pattern in this matrix. Unfortunately, both errors on T_{inside} leave the same fault pattern in Table 8. Comparing Figure 52 with Figure 57 confirms this result. Thus this FDD system will be unable to separate abrupt offsets from incipient offsets on T_{inside}.

Table 8. Fault identification matrix.

		Residual #1								Residual #2								R#1 – R#2
		Residual	Filtered	Squared	Mean	Median	VAR	STD	ROC	Residual	Filtered	Squared	Mean	Median	VAR	STD	ROC	
Normal Operation		O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O	O
Offset	P_{aircon}	X	X	X	X	X	O	O	O	X	X	X	X	X	O	O	O	+
	T_{inside}	X	X	X	X	X	X	X	O	X	X	X	X	X	X	X	O	O
	$T_{outside}$	X	X	X	X	X	O	O	O	X	X	X	X	X	O	O	O	-
Gain	P_{aircon}	X	X	X	O	O	X	X	O	X	X	X	O	O	X	X	O	+
	$T_{outside}$	X	X	X	O	O	X	X	O	X	X	X	O	O	X	X	O	-
Incipient	P_{aircon}	X	X	X	X	X	X	X	X	X	X	X	X	X	O	O	O	+
	T_{inside}	X	X	X	X	X	X	X	O	X	X	X	X	X	X	X	O	O
	$T_{outside}$	X	X	X	X	X	X	X	O	X	X	X	X	X	O	O	X	-
Logged	Data disconnect	Bad quality residuals																
	Compressor fail	Bad quality residuals																
	Heat source	X	X	X	X	X	X	X	O	X	X	X	X	X	X	X	X	+

Found in this table is the standard deviation (STD) response of the residuals. These graphs were not included in this text as they yield similar results as the variance graphs that were displayed. The results from chapter 4 and chapter 5 confirm this.

As was mentioned, the relative size of the residuals is useful to differentiate between faults in P_{aircon} and $T_{outside}$. If residual #2 is subtracted from residual #1, as was done in the last column of Table 8, a relative difference is produced. If this difference is positive the fault is more likely in P_{aircon}. A negative difference points to $T_{outside}$, while

a zero difference points to T_{inside}. This works even better if the squared filtered residuals are used.

In order to simplify the FDD tree, the fault identification matrix is simplified. Only the most significant residual patterns are used in Table 9, eliminating residuals comparable to the selected patterns.

Table 9. Reduced fault identification matrix.

		Residual #1				Residual #2				
		Squared	Mean	VAR	ROC	Squared	Mean	VAR	ROC	R#1 – R#2
		$R^2_{\#1}$	$M_{\#1}$	$V_{\#1}$	$\dot{R}_{\#1}$	$R^2_{\#2}$	$M_{\#2}$	$V_{\#2}$	$\dot{R}_{\#2}$	ΔR
Normal Operation		O	O	O	O	O	O	O	O	O
Offset	P_{aircon}	X	X	O	O	X	X	O	O	+
	T_{inside}	X	X	X	O	X	X	X	O	O
	$T_{outside}$	X	X	O	O	X	X	O	O	-
Gain	P_{aircon}	X	O	X	O	X	O	X	O	+
	$T_{outside}$	X	O	X	O	X	O	X	O	-
Incipient	P_{aircon}	X	X	X	X	X	X	O	O	+
	T_{inside}	X	X	X	O	X	X	X	O	O
	$T_{outside}$	X	X	X	O	X	X	O	X	-
Logged	Data disconnect	Bad quality residuals								
	Compressor fail	Bad quality residuals								
	Heat source	X	X	X	O	X	X	X	X	+

6.3.6 FDD tree

A binary decision tree can now be constructed from the fault identification matrix in Table 9. There are numerous other methods to implement the fault identification matrix. A decision tree is, however, one of the easiest and most understandable methods to employ. Figure 59 shows the FDD tree structure built around the reduced fault identification matrix.

In this section, the aim is to describe how Figure 59 functions. The tree describes a chronological order in which events should occur to identify failures in the HVAC system. In brief, the quality of the residuals should be evaluated. If the neural networks deliver usable outputs, the various residual alterations should be constructed. Firstly the type of fault is identified, either a gain, abrupt offset or incipient fault. Then the signal containing the fault is identified.

The data quality check is a functionality build into the CSense® neural network block set. It is based on the extreme values in the training data. If data are received outside the boundaries of the training set an unevaluated output would be generated. Since residuals can not be built with undependable data, all the FDD system can do is to evaluate the input variable boundaries. An "out of bounds" error will be generated.

If residuals can be generated, the first step is to detect a fault in the system. The favourable signal to study is the squared filtered residuals. From Figure 50 to Figure 58 it showed the most detectable spread for any of the faults studied. Then a trigger fires operation is considered faulty. If the squared residuals remain low, the HVAC system would be performing normal.

The second step would be to diagnose the fault. In Figure 59, the two mean signals are evaluated together. If both are low, gain error is identified. If both are high the identification process continues. In Table 9 it can be seen that – for the known faults – the two mean columns are exactly the same. Thus, if one fires high while the other fires low, an unknown fault is discovered. This is shown with a question mark in Figure 59.

Once the type of error is identified as a gain error, the signal in which this error occurs should be detected. This is done by assessing the relative size of the residuals by calculating the difference. If residual #1 is larger than residual #2 the fault lies in P_{aircon}. If the difference is below zero, the fault lies in $T_{outside}$. In the case where the difference is about zero, the fault is unknown. From the last column in Table 9 it is deducted that equal residuals point to an error in T_{inside}, but this remains uncertain. The suggested error is shown in dashed lines in Figure 59.

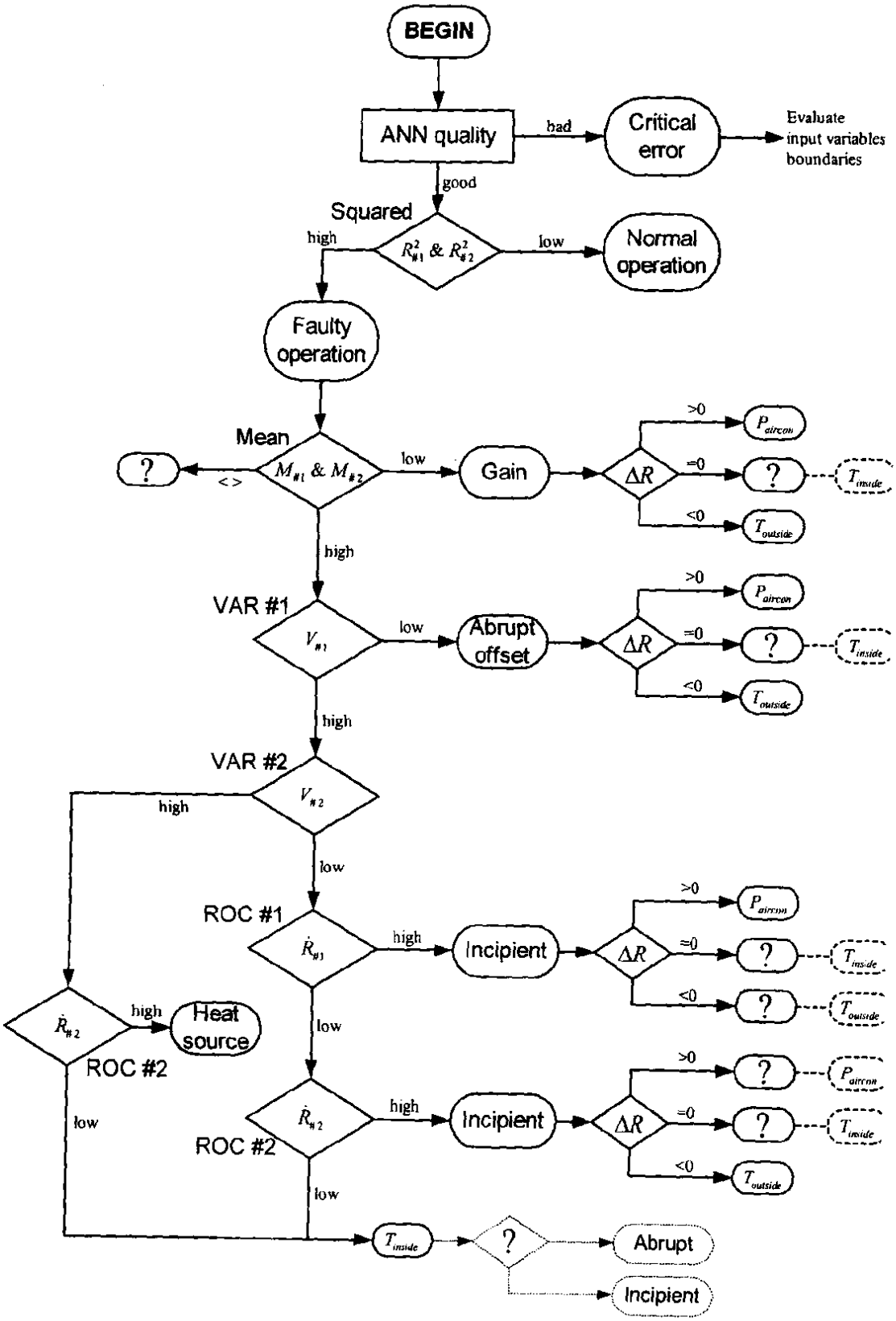


Figure 59. FDD decision tree for HVAC power usage

If the fault is not a gain, the VAR-graph from residual #1 is considered. A low trigger points to an abrupt error. When the trigger is high, the VAR-graph from residual #2 is studied. Here, a low variance leads to the evaluation of the two ROC-graphs from which the different incipient faults are identified. If the second variance is high and the second ROC is high, the fault is an unwanted heat source in the room.

If none of these conditions are met, the fault lies in T_{inside}. There is no method available to separate abrupt errors from incipient error on T_{inside}. At the bottom of Figure 59 a description is added on how to extend the tree once a new method is found to distinguish between abrupt and incipient errors.

6.4 Properties of the FDD tree

In the second chapter (section 2.3) a set of characteristics of FDD systems was discussed. Accordingly the deducted FDD system of this chapter will be analyzed under the following attributes:

- Robustness
- Quick detection and diagnosis
- Isolability
- Novelty identifiability
- Adaptability
- Modelling requirements
- Implementation requirements
- Computational requirements

The other characteristics examined in chapter 2 are not considered applicable in this particular study.

6.4.1 Robustness

As was stated in chapter 2, robustness is weighed against performance. Noise is the most contributing influence on the system's robustness. In an attempt to remove noise, the data are filtered twice before fault detection commences. Firstly the measurement of P_{aircon} is filtered as section 6.2.1 specifies, and secondly the residuals are squared and filtered.

The thresholds are selected at levels that cause minimal fault alarms. In over a month's fault free data three false alarms were activated. To show how the threshold levels influence the performance, consider Figure 60. A threshold of 0.004 is used. A number of different sized fault residuals are shown. Only those higher than 2.3% cross the threshold and those higher than 3.4% remain above the threshold level. The higher the fault, the quicker it is detected.

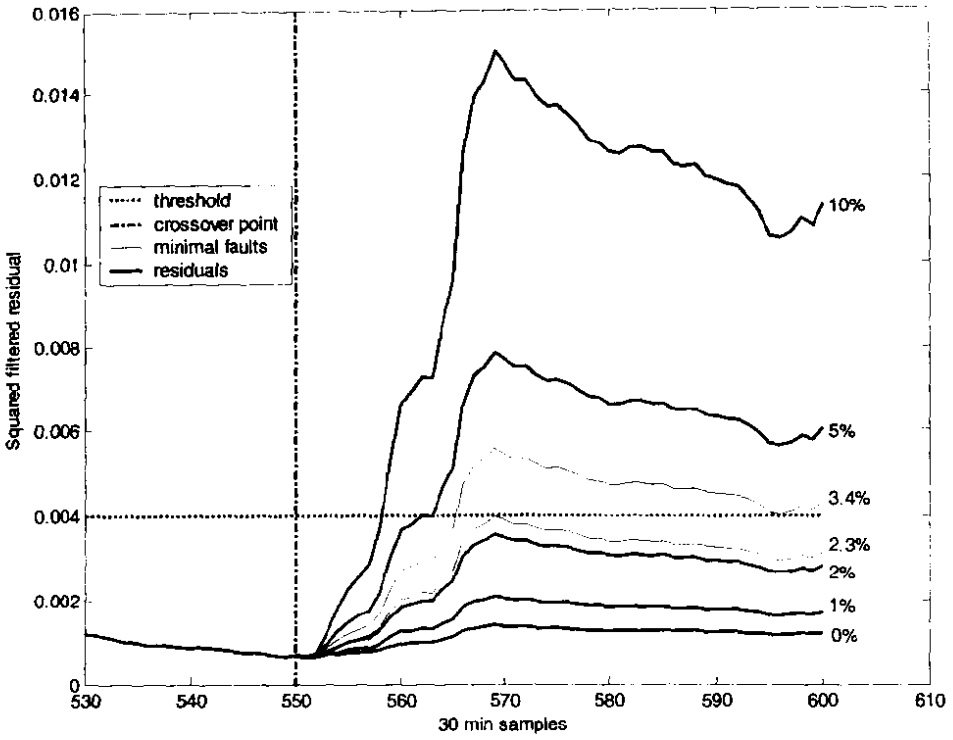


Figure 60. Residual #2 for different size offsets on the outdoor temperature

Figure 60 only considers one fault type on one of the residuals. The threshold for residual #1 can be set lower because ANN#1 out performs ANN #2. Different faults also lead to different detection speeds.

6.4.2 Quick detection and diagnosis

As can be seen from the FDD tree in Figure 59, faults are detected from the squared residual. With the thresholds set at 0.002 and 0.004 respectively, the failures are detected as follows. Gain and abrupt offsets cross the trigger between 2 and 17 measurements, while the incipient offsets studied, are detected between 20 and 64 measurements. In terms of measurements, 64 is much lower than the 390 needed in

the study done in chapter 5. However, the period between measurements is 30 minutes. Thus, 17 measurements represent a total of 8 ½ hours that is not very fast. 64 measurements amount to 1 day and 8 hours for detection on an incipient error. The main factor contributing to the rate of detection is sampling speed.

The speed of diagnosis is determined by the time triggers fire on the diagnosing graph sets. All the residual analysis figures showed that these graphs changed as the squared residual did. Thus, as a fault is detected is can be diagnosed.

Experiments showed that a 10% offset error on P_{aircon} was detected in 5 measurements or 2 ½ hours. A gain on $T_{outside}$ took 17 measurements or 8 ½ hours to detect.

6.4.3 Isolability

Isolability is the ability to distinguish between different failures. All the modelled faults are not completely isolatable through the fault tree. Methods still need to be developed to separate abrupt and incipient offsets on T_{inside}. Since this is the only issue with isolability, the FDD systems separation ability is considered highly effective.

6.4.4 Novelty identifiability

A decision tree will always return one of the outputs defined as leaves in the tree. Contrary to the FDD tree in chapter 5, the tree in Figure 59 contains end nodes that are unknown faults. Unfortunately data to test these nodes are unavailable. Moreover, the system modelled by the neural networks is highly involved. There is probably more unknown faults than known faults although many of these are covered by the artificial faults. It can be stated that the FDD system has some novelty identifiability.

6.4.5 Adaptability

There are two areas by which adaptability should be judged. First is the system's reaction to environmental changes like the replacement of equipment that causes permanent adjustments of transfer equations. Such adjustments should be altered in the ANNs. A package like CSense® provides an on-line trainer; accordingly the neural networks are updated as the process continues to function normally.

Secondly, the fault tree should be extended as new error types are recognized. To extend the fault tree a unique residual pattern needs to be identified. This might require that a new residual adaptation is incorporated into the fault tree. A new residual adaptation would also require a new threshold level to be deducted. All this can be done without the need to alter the current tree in use, making this a highly adaptable FDD system.

6.4.6 Modelling requirements

Training of the neural networks requires an efficient set of process history data. It does not require any knowledge of the HVAC system at hand. To build the decision tree also requirs historical fault data. No HVAC specialist was needed here. To sum up, the FDD system requires sufficient history data. If enough data are available beforehand, deploying the FDD system requires minimal resources.

6.4.7 Implementation requirements

Stand alone FDD systems are usually implemented on equipment that has been in circulation for some time. Thus, operation data should be readily available. This data are required for training the ANNs and should

- be local to the system, i.e. it is best if the data were collected from the site and machinery on which the FDD system will be implemented; and
- span the complete range of normal behaviour of the HVAC system.

After training the networks, fault data should be simulated through the ANN models in order to build a fault identification matrix. From the matrix a decision tree can be constructed.

When the FDD system is implemented over various sites only the training of the ANNs needs to be repeated. This is because similar faults lead to similar residual patterns. Compare Figure 23 with Figure 51 and Figure 24 with Figure 54.

6.4.8 Storage and Computational requirements

The size storage needed for this FDD system is mainly occupied by the database needed for the CSense neural networks. A CSense architect file, where under the FDD system is saved, requires 350 kilobytes.

Computational requirements are extremely low. Data are sampled at 30min intervals. This leaves ample time to process the data. In an experiment it took 7 minutes and 23 seconds to process 37 day's data. That is 0.24 seconds per sample.

6.5 References in chapter 6

[1] Bring, A., Sahlin, P. & Vuolle, M. *Models for Building Indoor Climate and Energy Simulation.* 1999. Dept. of Building Sciences. KTH. Stockholm.

CHAPTER 7

7 CONCLUSIONS AND RECOMMENDATIONS

With this chapter we close the study on FDD systems. The bulk of the preceding chapters are summarized, highlighting the key conclusions made during the course of this investigation. Further recommendations, as well as a number of improvements to increase the accuracy and effectiveness of FDD systems are mentioned.

7.1 Conclusions

In the first chapter the goal of this dissertation was divided into a couple of sub-targets. Mainly, these involve the three sub-systems that interact together for FDD purposes, modelling, residual analysis and classification.

Modelling non-linear processes with the aid of neural networks in chapters 4, 5 and 6 was highly effective. Even with the random factor of a backlash hysteresis, found in damper actuators in chapter 5, MSE values below 10^{-4} were obtained between the plant and the ANN model. Thus, the neural network predicts plant output with acceptable accuracy.

In chapter 6 a physical system was modelled with measured data. In this HVAC system there are many unknown elements and variables. Without these values the HVAC system could still be modelled with an MSE of 0.003.

The major advantage gained from neural network modelling lies in the training of the model from process history data. This means that implementation of the FDD system does not require an HVAC expert or even specific HVAC system information. The drawback is that training requires historical data from the targeted plant.

Residual analyses showed that statistical methods can be used extensively to mine for information in the residuals. Some faults have similar spreads in some residuals, while acquiring completely unrelated patterns after a transformation on the residual. This is excellent for separating one fault from another.

Classification of errors was done using a decision tree. This is a very simple logical pattern to follow through the residual sets to obtain the error status of the plant. In chapter 5 the tree was a binary tree. All the branches split in two and all the leaves pointed to some error. In chapter 6 the tree considered fault residual patterns that are unknown or new to the FDD system. This allows for novelty identifiability.

Chapter 4 implemented a multiple network FDD system. It was shown that such a configuration has the ability to separate sensor errors from plant errors. In chapter 5 such a system was utilized to differentiate between controller failures, plant failures and sensor failures.

Finally the FDD system was measured and judged according to the desirable attributes mentioned in the literature study. This system is classified as highly robust and adaptable with low modeling and computing costs. This makes the FDD solution quick in the identification of faults.

In chapter 5 the FDD system fails to identify novel faults. This reduces the system's isolability, as known faults are diagnosed perfectly while unknown faults are categorized under known fault sets. In chapter 6 this issue was addressed and facilities to incorporate novel faults were developed.

Measuring FDD performance through characteristics shows a designer where there is space for improvement of the system, and in the next section this is discussed, but the depth of this investigation must be kept realistic. This investigation most certainly *points* to a feasible concept.

7.2 Recommendations

The FDD system has room for improvement when unknown faults are encountered. In the current state the decision tree limits the FDD system to misclassify a new fault as a known fault. This is because of the minimal amount of unknown termination points in the tree. To resolve this issue the following methods are recommended:

- One method would be to extend the tree to carry more leaves reserved for unknown faults. This will be a daunting task, as unknown faults could share properties with known malfunctions. It would require the implementation of more residual adaptations and consequently expand the tree.
- Another method would be to replace the decision tree with a less discreet system like a fuzzy logic classifier.
- A third option would be to implement a committee of tree structures, assigning to each a priority level for certain fault sets.

The resolution of the FDD system is somewhat small because the attempt was to develop a concept rather than to commission a full FDD system. The available data also reduced the scope of the system as fault data was sparse. For implementation of a similar system it is recommended that the scope of errors analyzed be extended dramatically.

A study can be inducted to the effect of faulty input sensor readings to ANN as this was neglected during this study. Faults deducted from the symptoms should always include the possibility of an input sensor fault.

A APPENDIX: NEURAL NETWORK DESIGN

A.1 Software packages

In chapters 4 and 5 artificial neural networks (ANN) are implemented for residual generation. These residuals are then further studied for FDD purposes that form the core of this dissertation. To build these simulations and validate the theory, two software packages were used: Matlab® and CSense®.

A.1.1 Matlab® neural network toolbox

Specific network architectures tend to be more efficient with certain training algorithms. These algorithms were implemented with the Matlab® neural network toolbox on feed forward networks. These networks implement neurons in layers (See Figure 61). Each layer has an arbitrary number of neurons. For complex problems this number is increased to retain the knowledge that is needed. Increasing the number of layers may enable the network to follow more complex behaviour in its training data.

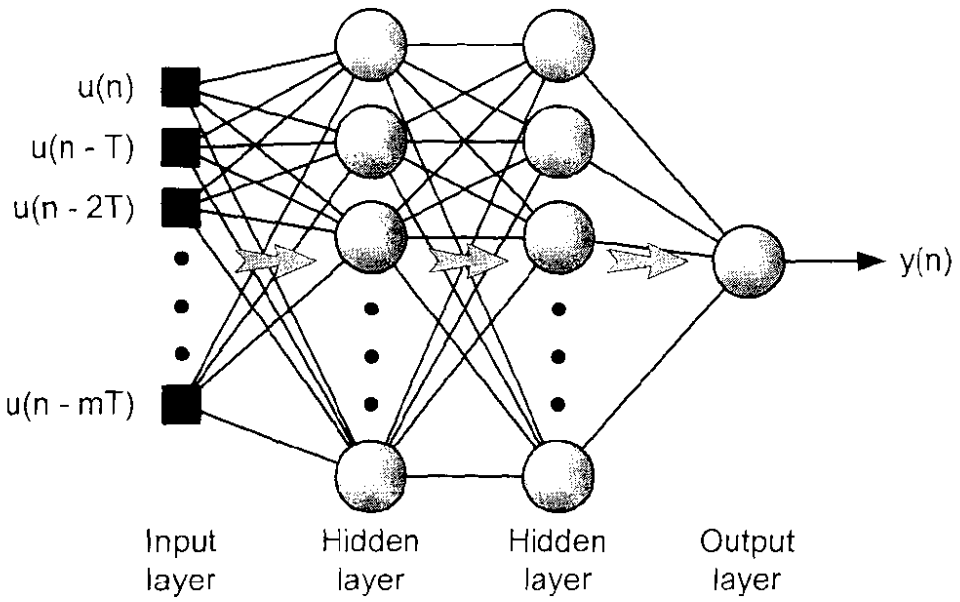


Figure 61. Layout of a feed forward neural network

The number of times the training data are fed to the network, called the number of epochs, is also important to ensure that the network captures the behaviour of the data. There is a trade-off between the number of neurons, the number of layers and the time required to train the network. Furthermore, problems are sometimes better solved with a less complex network.

With the Matlab® neural network toolbox the developer has complete control over the activation functions, dimensions and training algorithm of the ANN. One can specify multiple outputs and other layout aspects. The multilayer perceptron is but a choice amongst many network topologies to choose from. Popular training algorithms are:

- Gradient descent back-propagation
- Gradient descent with momentum and adaptive learning rate back-propagation
- Levenberg-Marquardt back-propagation
- Resilient back-propagation
- BFGS quasi-Newton back-propagation

According to the Matlab® help, the multilayer neural network is quite powerful since it states that a two-layer network can approximate any function to an acceptable degree of accuracy [1]. Also refer to Haykin [2].

A.1.2 CSense® Architect

CSense® is a comprehensive development software platform that enables the development of real time applications for process diagnosis, decision-support, supervisory process control, and condition management of equipment and processes. As for Matlab® being an interpretive mathematical environment. These are two very different packages made to satisfy different needs.

CSense® limits the designer to a single output multi-layer perseptron. The option is for either one or two hidden layers and the amount of neurons each layer contains. There is no choice for activation functions other than the sigmoid function. In essence none of these factors impose a significant loss in neural network accuracy. What does influence the performance of CSense® in comparison with Matlab® is the training

algorithm. CSense® simply uses the Gradient descent back-propagation algorithm for ANN training. To CSense's advantage is the implementation of databases, especially for neural network training. This allows for training with enormous sets of data.

To sum up, a tweaked MATLAB® network can produce a residual with a MSE value below 10^{-6} while a CSense® network performs with an MSE value below 10^{-4}. This difference can easily be overshadowed by noise in the system. (These MSE values are from the simulation of Figure 20 in chapter 4.)

A.2 Networks from chapter 4

A.2.1 Network design

Both the networks in Figure 62 receive the square wave input of Figure 63. From the second network 'n more complex transformation is required, accordingly the network is built with more neurons in each layer.

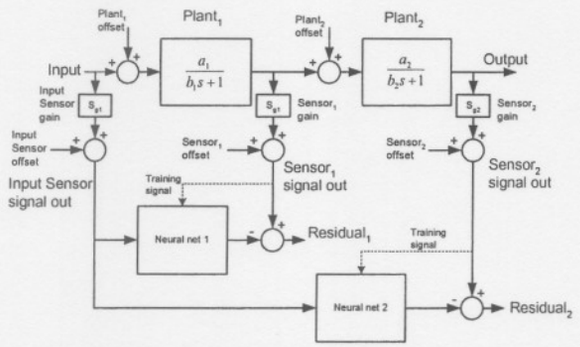


Figure 62. Simulation of chapter 4

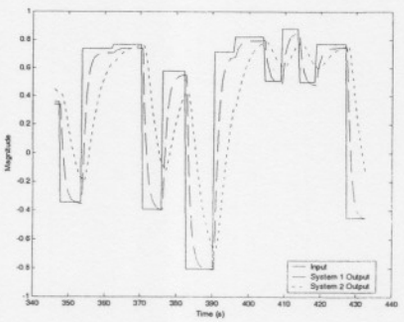


Figure 63. Input and output of Figure 62

For the first network experimentation found that a network with 14 delayed inputs, 4 neurons in the first hidden layer and 2 in the second delivers optimum results. Additional neurons lead to overtraining whilst reducing the number delivers poorer results. Through the same method the second network dimensions are: 56-8-4-1 (56 delayed inputs, 8 neurons in the first hidden layer 4 in the second hidden layer and of course 1 output). In both networks the first layer is build with the hyperbolic tangent sigmoid transfer function. The second and output layers contain a hard limit transfer function.

A.2.2 Network performance

Usually ANN performance is measured in the MSE value of the residual between the network output and the target. This criterion was used to select the highest performing topology. But these networks form part of an FDD system and it seems appropriate to judge the network on the success of the FDD system.

Mainly, the concern is successful fault detection. Table 10 shows this as a percentage value when the FDD network, correctly, predicted the state of the system. Further performance criteria, also in Table 10, include:

- Training time, measured in epochs
- Network deviation from the training target, measured in MSE values and
- Network deviation from validation dataset targets, measured in MSE values

The percentage error column in table 1 refers to the size of the error, deviating a percentage from the normal operating value. The threshold levels were selected at the lowest possible level which would never trigger false alarms for the data at hand.

Table 10 clearly shows a direct relation between the threshold level and the training MSE value. A superior trained network can, therefore, detect smaller errors but takes longer to learn.

Table 10. Summary of ANN training

		Networks trained		
		#1	**#2**	**#3**
	Training MSE	1×10^{-5}	1×10^{-4}	1×10^{-3}
	Validation MSE	1.19×10^{-5}	1.22×10^{-4}	1.2×10^{-3}
	Threshold	1.1×10^{-4}	5.7×10^{-4}	6.2×10^{-3}
	Epoch	4799	565	116
Error deviation size from normal operation value.	1 %	0.25 %		
	1.5 %	25.04 %		
	1.75 %	64.63 %		
	2 %	83.37 %	0.74 %	
	2.5 %	96.03 %	1.69 %	
	3 %	97.27 %	24.57 %	
	4 %		64.27 %	
	5 %	98.51 %	92.56 %	
	7.5 %		97.02 %	0.00 %
	10 %	100.00 %	98.26 %	27.79 %
	12.5 %			44.91 %
	15 %		100.00 %	77.67 %
	17.5 %			87.34 %
	20 %			96.03 %

To summarize, a network trainer with an MSE value below 1×10^{-5} would detect 80% of the errors 2 % or greater than faultless behaviour. In comparison with the same network trained to an MSE value around 1×10^{-3} incapable to identify 80 % failures smaller than 15 % off the normal operation value.

A.3 Networks from chapter 5

A.3.1 Network design

Both the networks in Figure 64 receive the square wave input of Figure 65.a. The first network produces a signal equal to that of the implemented PID controller (Figure 65.b). From the second network a more complex transformation is required; accordingly the network is built with more neurons than the first network.

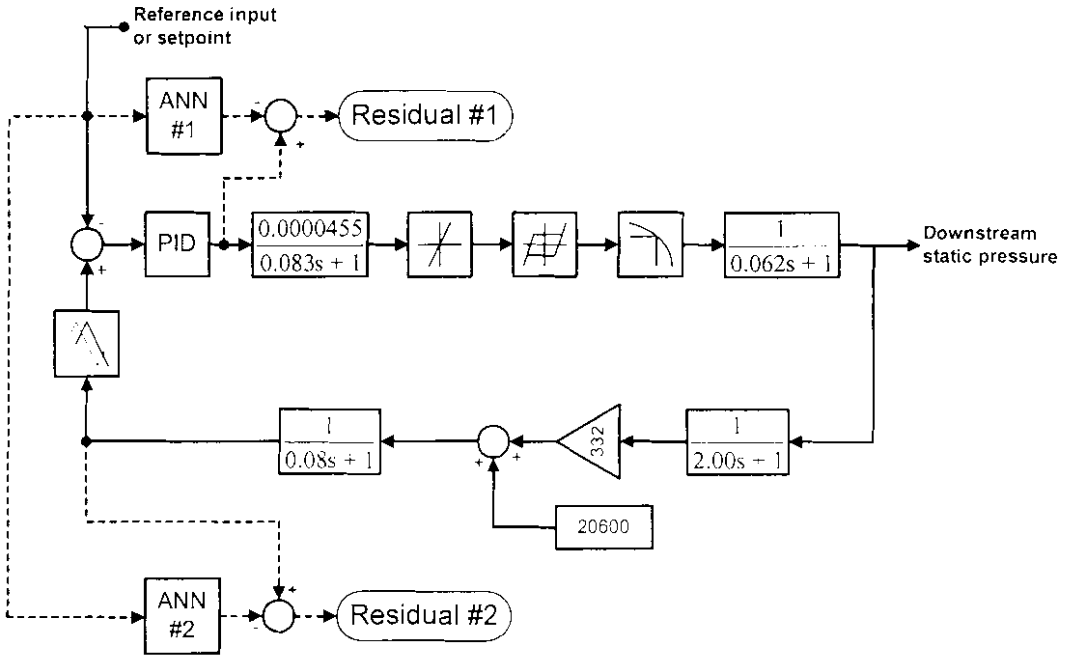


Figure 64. Simulation of chapter 5

In an attempt to capture the transient of the PID controller (refer to Figure 65.b), the neural network needs delayed inputs spanning the duration of the transient which is 5 seconds. The waves are sampled each tenth of a second, meaning at least 50 delayed inputs are required for the first network.

By the same principle, the second network requires at least 120 delayed inputs as the stretch of the transient is much larger in Figure 65.a. In total, a 120–12–4–1 network architecture has (120 × 12) + (12 × 4) + (4 × 1) = 1492 weights. Table 14 shows that these sized networks take around 3 days to train. (It should be mentioned that ANNs coded with compilers – like C, rather than with interpreters – including MATLAB® , train much faster, making training times an insignificant factor.)

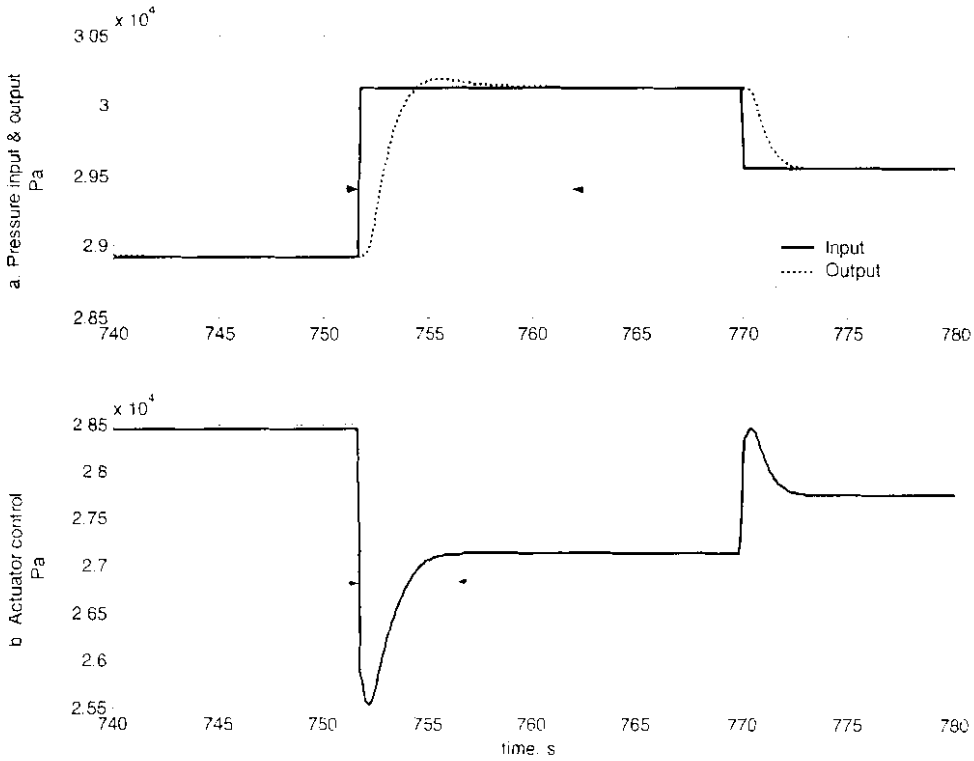


Figure 65. Input and output of Figure 64

To speed up training time, the sampling length was increased from 0.1s to 0.4s, thus the delayed inputs are reduced from 120 to 30, cutting network size to 412 neurons. Unfortunately, the training waves lost some smoothness limiting the accuracy of the ANN. An intermediate sampling level, employing every second data point of the high resolution data, required reasonable training times doubling the accuracy of the under-sampled networks. The intermediate data were used on a network with 60 input delays.

A.3.2 Network performance

In this section results are listed, weighing different network architectures up against each other. As was the case in chapter 4, training the first ANN is relatively effortless compaired to training the second. This is because, considering Figure 64, the data transformation required from the first network is less intricate than the transformation of the second network. Accordingly, more effort was devoted to optimizing the second network.

The first network training results are shown in Table 11. From Figure 66 it is visible that network no. 10, a 50–15–5–1 network, performs most sufficiently with a training MSE value of 2.19×10^{-5}. However, for FDD purposes any of the topologies listed in Table 11 would suffice since the FDD performance is mainly bounded by the second network's abilities.

Table 11. ANN training for the first network of chapter 5

#	Architecture	MSE		Epoch	Training time
		Training	Evaluation		
1	50 - 6 - 2 - 1	2.93×10^{-5}	3.288×10^{-5}	1221	156 min
2	50 - 6 - 4 - 1	2.26×10^{-5}	3.128×10^{-5}	6624	625 min
3	50 - 8 - 2 - 1	2.491×10^{-5}	2.665×10^{-5}	2637	312 min
4	50 - 8 - 4 - 1	2.297×10^{-5}	2.647×10^{-5}	2700	311 min
5	50 - 8 - 6 - 1	2.653×10^{-5}	2.7×10^{-5}	3769	106 min*
6	50 - 10 - 2 - 1	2.27×10^{-5}	2.412×10^{-5}	2677	435 min
7	50 - 10 - 6 - 1	2.24×10^{-5}	2.303×10^{-5}	3921	126 min*
8	50 - 12 - 3 - 1	2.292×10^{-5}	2.412×10^{-5}	5275	173 min*
9	50 - 12 - 4 - 1	2.504×10^{-5}	2.593×10^{-5}	4477	153 min*
10	50 - 15 - 5 - 1	2.19×10^{-5}	2.254×10^{-5}	3443	170 min*
11	50 - 15 - 9 - 1	2.7×10^{-5}	3.182×10^{-5}	2235	131 min*

* *trained on a high performance computer*

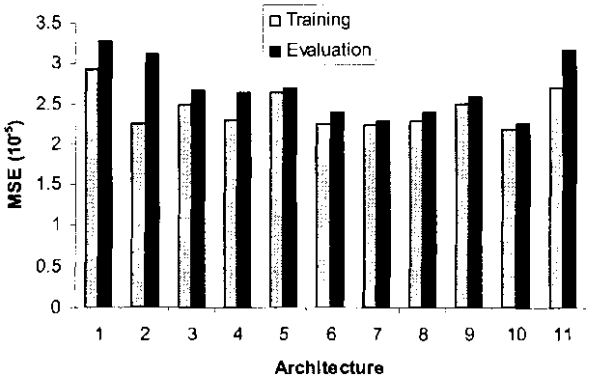


Figure 66. Performance comparison of networks architectures for the first ANN of chapter 5

The next couple of pages contain the training results of the second network from Figure 64. The results are grouped according to the sampling resolution.

Table 12. Neural network training with heavily sub-sampled data

#	Architecture	MSE		Epoch	Training time
		Training	Evaluation		
1	30 – 6 – 2 – 1	4.51×10^{-4}	4.92×10^{-4}	16201	240 min
2	30 – 6 – 4 – 1	4.5×10^{-4}	4.79×10^{-4}	10000	168 min
3	30 – 8 – 2 – 1	4.47×10^{-4}	4.66×10^{-4}	9836	190 min
4	30 – 8 – 4 – 1	4.72×10^{-4}	4.8×10^{-4}	11812	249 min
5	30 – 8 – 6 – 1	4.47×10^{-4}	4.62×10^{-4}	8851	201 min
6	30 – 10 – 2 – 1	4.53×10^{-4}	4.6×10^{-4}	6689	166 min
7	30 – 10 – 4 – 1	4.4×10^{-4}	4.48×10^{-4}	10000	258 min
8	30 – 12 – 2 – 1	4.45×10^{-4}	4.46×10^{-4}	4897	145 min
9	30 – 12 – 3 – 1	4.43×10^{-4}	4.44×10^{-4}	8451	251 min
10	30 – 12 – 4 – 1	4.57×10^{-4}	4.59×10^{-4}	7451	233 min
11	30 – 12 – 6 – 1	4.49×10^{-4}	4.62×10^{-4}	9701	420 min
12	30 – 15 – 2 – 1	4.41×10^{-4}	4.67×10^{-4}	12104	451 min
13	30 – 15 – 5 – 1	4.49×10^{-4}	4.94×10^{-4}	8776	380 min
14	30 – 15 – 9 – 1	3.69×10^{-4}	6.28×10^{-4}	34365	1873 min

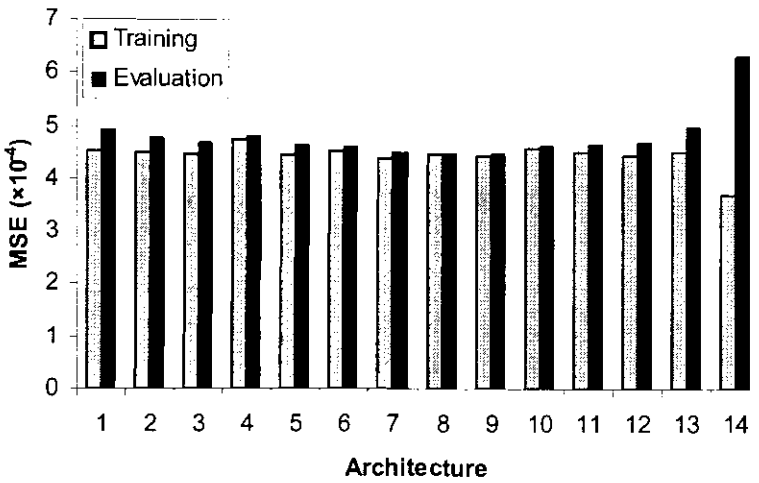


Figure 67. Performance comparison of networks with 30 input delays

Table 13. Neural network training with ½ sub-sampled data

#	Architecture	MSE		Epoch	Training time
		Training	Evaluation		
1	60 - 6 - 2 - 1	2.09×10^{-4}	2.12×10^{-4}	14526	154 min*
2	60 - 6 - 4 - 1	2.09×10^{-4}	2.13×10^{-4}	15351	177 min*
3	60 - 8 - 2 - 1	2.11×10^{-4}	2.14×10^{-4}	7967	216 min
4	60 - 8 - 3 - 1	2.13×10^{-4}	2.15×10^{-4}	6458	93 min*
5	60 - 8 - 4 - 1	2.07×10^{-4}	2.09×10^{-4}	13930	1000 min
6	60 - 8 - 6 - 1	2.08×10^{-4}	2.1×10^{-4}	13776	848 min
7	60 - 10 - 2 - 1	2.09×10^{-4}	2.09×10^{-4}	9301	623 min
8	60 - 10 - 4 - 1	2.08×10^{-4}	2.085×10^{-4}	32616	2335 min
9	60 - 12 - 2 - 1	2.03×10^{-4}	2.05×10^{-4}	10500	710 min
10	60 - 12 - 3 - 1	2.07×10^{-4}	2.071×10^{-4}	6109	559 min
11	60 - 12 - 4 - 1	2.05×10^{-4}	2.06×10^{-4}	6905	612 min
12	60 - 12 - 6 - 1	2.02×10^{-4}	2.09×10^{-4}	36751	3638 min
13	60 - 15 - 2 - 1	2.13×10^{-4}	2.18×10^{-4}	5776	168 min*
14	60 - 15 - 5 - 1	2.27×10^{-4}	2.32×10^{-4}	6826	936 min
15	60 - 15 - 9 - 1	2.29×10^{-4}	2.4×10^{-4}	6026	950 min

** trained on a high performance computer*

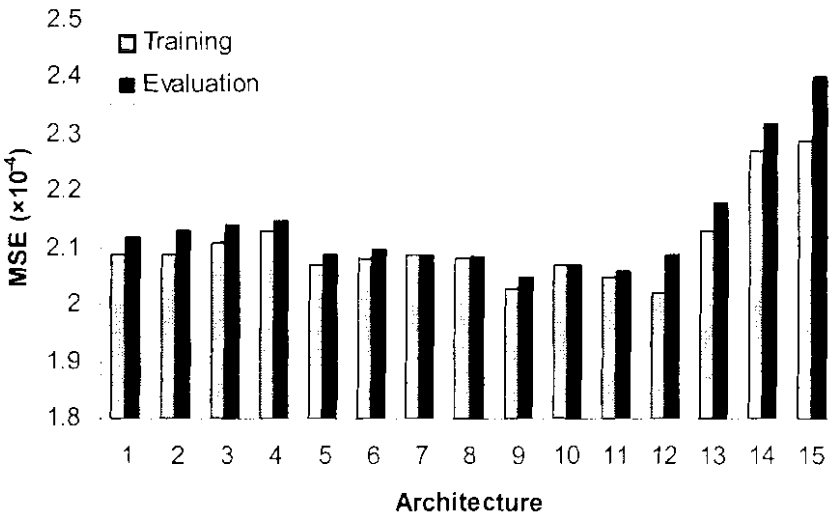


Figure 68. Performance comparison of networks with 60 input delays

Table 14. Neural network training with high resolution data

#	Architecture	MSE		epoch	Training time
		Training	Evaluation		
1	120 - 8 - 2 - 1	1.51×10^{-4}	1.71×10^{-4}	15797	824 min*
2	120 -10 - 4 - 1	0.999×10^{-4}	1.23×10^{-4}	68222	3807 min*
3	120 -12 - 4 - 1	0.998×10^{-4}	1.17×10^{-4}	71576	4489 min*
4	120 -15 - 5 - 1	0.998×10^{-4}	1.37×10^{-4}	59552	4302 min*

** trained on a high performance computer*

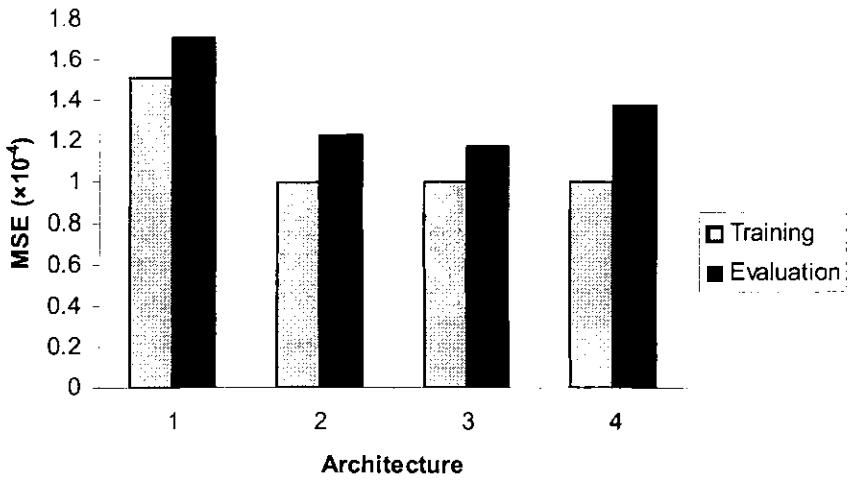


Figure 69. Performance comparison of networks with 120 input delays

Figures 51, 52 and 53 as well as Tables 10, 11 and 12 show how under-sized networks performances are limited while large networks over-trains. The last mentioned phenomena can be seen in the difference between the training MSE values and that of the evaluation set.

From this section's tables, implementing a 50–15–5–1 network for the first ANN seems the best option. For the second network a 120–12–4–1 is used for the FDD purposes in chapter 5. According to Table 10, faults deviating approximately 5 % from normal operation should be isolated and identified with the FDD system of Figure 64. Both these networks use data sampled 0.1 seconds apart.

A.4 Networks from chapter 6

A.4.1 Network design

A great deal is known about the systems modelled in chapter 4 and chapter 5. The physical interactions are understood up to a point where mathematical models could be constructed. In chapter 4 the models were first order and second order transfer functions and in chapter 5, a sixth order transfer function was implemented. All that was understood in the models gave away some clues as to how the ANN models should look. The transfer functions of the HVAC system in chapter 6 are unknown.

What is known is how the input and output to the system looks. Figure 70 shows how the system cycles periodically. Each cycle stretches 24 hours as the temperature changes from day time to night time. See Figure 70.a. The HVAC power also have a 24 hour cycle as Figure 70.b. shows.

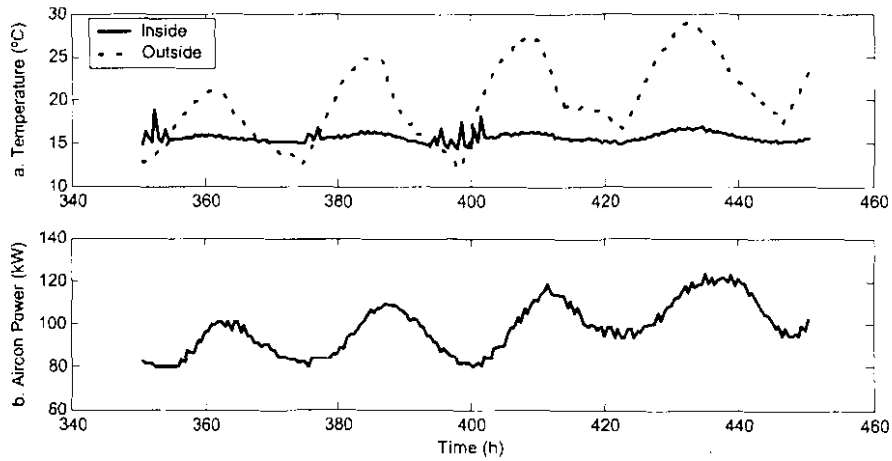


Figure 70. Measured variables of the system from chapter 6

The inside temperature, T_{inside}, is used as a common input to both ANNs. The outside temperature, $T_{outside}$, is used as an input to the first ANN and serves as the target output of the second ANN. The HVAC power, P_{aircon}, have an opposite function to $T_{outside}$. P_{aircon} is an input to ANN #2, and the comparative output of ANN #1. This cross configuration can be seen in Figure 71.

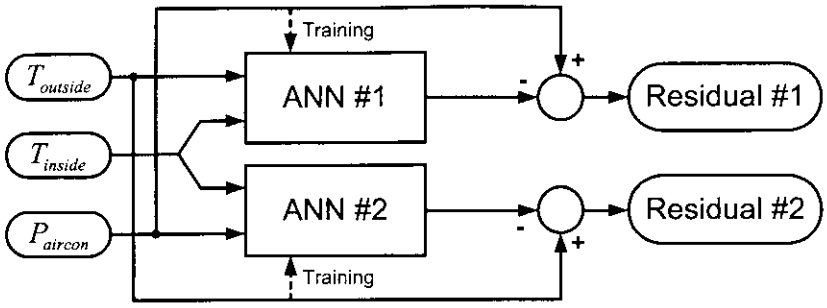


Figure 71. Cross validating FDD architecture

$T_{outside}$ and P_{aircon} appears to be similar, but there is a time shift in the waves of Figure 70. P_{aircon} is running between 4 to 6 hours behind $T_{outside}$. Accordingly, one ANN should perform a forward shift while the other ANN should make a backward shift. To accomplish this, both networks need a complete cycle as input. As a cycle is 24 hours and data are sampled every 30 minutes both networks has a 48 input delay pipeline.

A.4.2 Network performance

Training in chapter 4 and 5 was done with Matlab® which allows the designer to manipulate every aspect of the neural network. In chapter 6 CSense® is used. This limited the design to variations in only the number of neurons in 2 hidden layers. In this section results are listed, weighing different network architectures up against each other.

CSense® only uses the gradient descent back-propagation algorithm for ANN training. Together with the CSense® database access facility training times for all the networks tabulated hereafter, was less than 2 hours. In fact some of the smaller networks trained within 10 minutes.

Table 15 along with Figure 72 shows comparative architectures and their training capabilities for ANN #1. CSense® randomly selects 20% of the data for evaluation purposes. Therefore the evaluation data set differ for each architecture. From Figure

72 it is clear that a 48–36–6 1 network delivers a high RMS valve without over-training.

Table 15. Training results for ANN #1

#	Architecture	MSE		epoch
		Training	Evaluation	
1	48 - 10 - 2 - 1	3.27 $\times 10^{-3}$	3.31 $\times 10^{-3}$	412
2	48 - 15 - 3 - 1	3.19 $\times 10^{-3}$	3.25 $\times 10^{-3}$	633
3	48 - 20 - 2 - 1	3.13 $\times 10^{-3}$	3.19 $\times 10^{-3}$	593
4	48 - 20 - 6 - 1	3.05 $\times 10^{-3}$	3.1 $\times 10^{-3}$	656
5	48 - 28 - 4 - 1	3.04 $\times 10^{-3}$	3.04 $\times 10^{-3}$	1203
6	48 - 28 - 8 - 1	2.96 $\times 10^{-3}$	3.01 $\times 10^{-3}$	1523
7	48 - 36 - 3 - 1	2.97 $\times 10^{-3}$	3.04 $\times 10^{-3}$	877
8	48 - 36 - 6 - 1	2.9 $\times 10^{-3}$	2.93 $\times 10^{-3}$	1741
9	48 - 36 - 10 - 1	2.9 $\times 10^{-3}$	2.95 $\times 10^{-3}$	1826
10	48 - 40 - 2 - 1	3.11 $\times 10^{-3}$	3.14 $\times 10^{-3}$	914
11	48 - 40 - 5 - 1	2.89 $\times 10^{-3}$	2.96 $\times 10^{-3}$	1367
12	48 - 40 - 10 - 1	2.88 $\times 10^{-3}$	3.07 $\times 10^{-3}$	2012
13	48 - 42 - 8 - 1	2.87 $\times 10^{-3}$	3.07 $\times 10^{-3}$	2175
14	48 - 48 - 8 - 1	2.84 $\times 10^{-3}$	3.08 $\times 10^{-3}$	2285

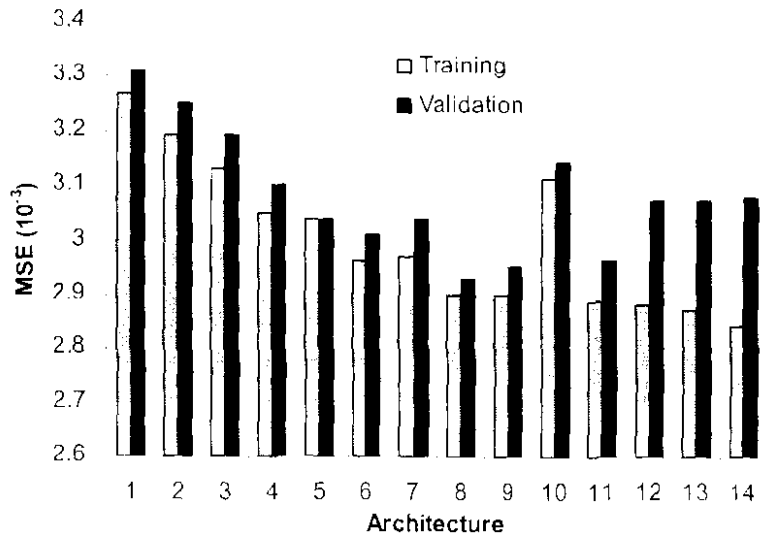


Figure 72. Performance comparison of architectures for ANN #1

Table 16. Training results for ANN #2

#	Architecture	MSE		epoch
		Training	Evaluation	
1	48 - 10 - 2 - 1	5.78 $\times 10^{-3}$	5.78 $\times 10^{-3}$	317
2	48 - 15 - 3 - 1	4.57 $\times 10^{-3}$	4.59 $\times 10^{-3}$	476
3	48 - 20 - 2 - 1	4.32 $\times 10^{-3}$	4.32 $\times 10^{-3}$	540
4	48 - 20 - 6 - 1	3.92 $\times 10^{-3}$	4.27 $\times 10^{-3}$	722
5	48 - 28 - 4 - 1	3.82 $\times 10^{-3}$	3.89 $\times 10^{-3}$	705
6	48 - 28 - 8 - 1	3.58 $\times 10^{-3}$	3.87 $\times 10^{-3}$	1038
7	48 - 36 - 3 - 1	3.4 $\times 10^{-3}$	3.42 $\times 10^{-3}$	896
8	48 - 36 - 6 - 1	3.12 $\times 10^{-3}$	3.37 $\times 10^{-3}$	1143
9	48 - 36 - 10 - 1	3 $\times 10^{-3}$	3.16 $\times 10^{-3}$	1847
10	48 - 40 - 2 - 1	3.12 $\times 10^{-3}$	3.15 $\times 10^{-3}$	1252
11	48 - 40 - 5 - 1	2.97 $\times 10^{-3}$	3.02 $\times 10^{-3}$	1664
12	48 - 40 - 10 - 1	2.74 $\times 10^{-3}$	3.28 $\times 10^{-3}$	1739
13	48 - 42 - 8 - 1	2.77 $\times 10^{-3}$	3.22 $\times 10^{-3}$	1578
14	48 - 48 - 8 - 1	2.69 $\times 10^{-3}$	3.35 $\times 10^{-3}$	1670

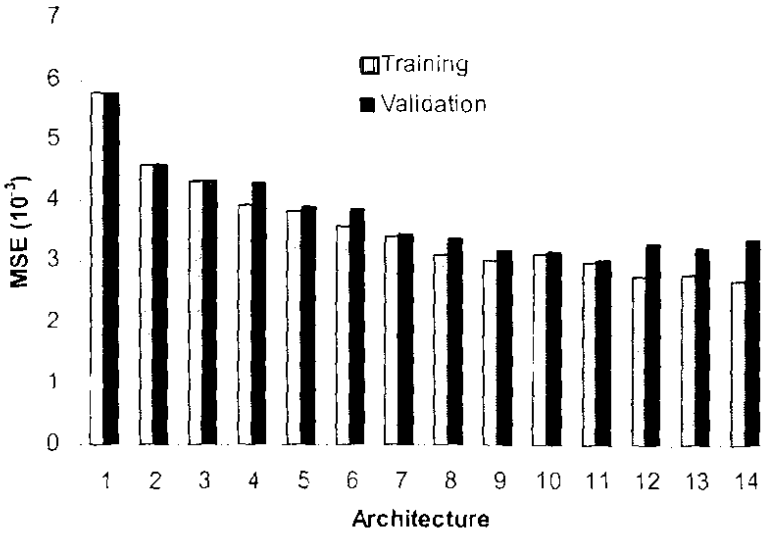


Figure 73. Performance comparison of architectures for ANN #2

Table 16 and Figure 73 show that an architecture of 48–40 -5–1 working best for ANN #2. This network's neuron count is not far from the architecture used for ANN #1. This is expected because the networks need to transform data in similar manners.