CASE BASED REASONING AS AN EXTENSION OF FAULT DICTIONARY METHODS FOR LINEAR ELECTRONIC ANALOG CIRCUITS DIAGNOSIS

Carles POUS i SABADÍ

ISBN: 84-688-8620-3
Dipòsit legal: GI-1141-2004
http://hdl.handle.net/10803/7728

Universitat de Girona

UNIVERSITAT DE GIRONA
Departament d'Electrònica, Informàtica i Automàtica

THESIS

Case Based Reasoning as an Extension of Fault Dictionary Methods for Linear Electronic Analog Circuits Diagnosis

Carles Pous i Sabadí

Co-director: Dr. Joan Colomer Llinàs

Co-director: Dr. Josep Lluís de la Rosa i Esteva

Girona, April 2004

Acknowledgements

Yes, at last the thesis is over. It seems impossible, but it is true. One thing that this huge work has demonstrated to me is that one is not alone in this world. There are people that are always ready to listen to you, to put up with your bad moods, and to encourage you when the things get tough.

As a rule of thumb, the first acknowledgements in a thesis are for the thesis directors and this one is not going to be different. So, thanks to them, and especially to Joan Colomer for being, let me say... a Saint. He could be defined as patience made human. Typical suggestions like ”Try this” were most commonly accompanied by my usual answer: ”Joan, I tried that and...it does not work. The results obtained are even worse”, or ”I've got another table of results but they are worse”. It seemed like a battle, with Joan trying to solve the problem on one side, and me on the other, ready to spoil his next suggestion. So, many thanks for guiding my across this universe of failures.

Next to my office, there is another special person whose enthusiastic encouragement I would like to acknowledge. Quim Meléndez, what would I have done without your ideas, proposals, solutions and, above all without your ”Is the thesis finished already?” or ”You are going to finish tomorrow, aren't you?” or ”Dipositar!!!, Dipositar!!!”.

Also, the colleague of the upper flor office has to be mentioned. Why are statistics so complicated, except when Josep Antoni Martín is the person who is explaining the narrow and winding road of statistics to you? Thanks for your most valuable cooperation, in spite of your ”Thanks? For what? Oh, no, it's nothing”. Also, many thanks for taking advantage of my incredible passing shoots (I know they are....) and converting them into goals while playing soccer with the Spartak d'Aiguaviva.

Of course, it is impossible to thank all the people who have helped me in one way or another, but I would like to mention the people of the department that have supported me in many ways. The patience of Josep Tomàs and Marc Rodríguez helping me with the hardware, Toni Verdú and Ingrid Azorín trying to solve my computer hang-ups, installing new programs, and so on. Also, I am very grateful to my students, whose work and questions helped me in many situations steering me in the right direction. Many thanks to all of them.

There is another important group of persons in the department who made the thesis easier. Not because of their direct intervention, but for allowing me to contribute to some research tasks that made me feel very useful and, above all, to participate in a group where the work atmosphere is incredibly good and very difficult to find in the rest of the universe. So, thanks to all of the capsdepooooooooorkkkkkk!!!!! group. (There is no translation for this expression, but they will understand me.)

I would also like to thank my parents and sister for helping me throughout these years, in the primary and secondary school, during my graduate studies, and now with the PhD. There are not enough words to thank them for the support and the encouragement they gave me, especially when the going got too tough. They have always been on my side.

Finally the most important one. I would like to thank Bianca for the support I received from her. A part from being my wife (a very difficult task, indeed), she is my best friend, and the person who stood by me, even in the panic-attack situations. Also, it is important to mention that she has been my best debugger. When the software developed reported errors or incoherent results, it was impossible for them to escape Bianca's intuition and cleverness. She is one of the most intelligent and brightest creatures I've ever met. Her support has been invaluable. Yes, I know, it seems that I cannot be very objective on this point, but I'm sure that I am.

This work has been partially supported by Spanish government and FEDER funds (SECSE, DPI2001-2198) and a contract between UdG and Endesa Distribution SAU, which are hereby gratefully knowledged.

Abstract

Testing circuits is a stage of the production process that is becoming more and more important when a new product is developed. Test and diagnosis techniques for digital circuits have been successfully developed and automated. But, this is not yet the case for analog circuits. Even though there are plenty of methods proposed for diagnosing analog electronic circuits, the most popular are the fault dictionary techniques. In this thesis some of these methods, showing their advantages and drawbacks, are analyzed.

During these last decades automating fault diagnosis using Artificial Intelligence techniques has become an important research field. This thesis develops two of these techniques in order to fill in some gaps in fault dictionaries techniques. The first proposal is to build a fuzzy system as an identification tool. The results obtained are quite good, since the faulty component is located in a high percentage of the given cases. On the other hand, the percentage of successes when determining the component's exact deviation is far from being good.

As fault dictionaries can be seen as a simplified approach to Case-Based Reasoning, the second proposal extends the fault dictionary towards a Case Based Reasoning system. The purpose is not to give a general solution, but to contribute with a new methodology. This second proposal improves a fault dictionary diagnosis by means of adding and adapting new cases to develop a Case Based Reasoning system. The case base memory, retrieval, reuse, revise and retain tasks are described. Special attention to the learning process is taken.

Several circuits are used to show examples of the test methods described throughout the text. But, in particular, the biquadratic filter is used to test the proposed methodology because it is defined as one of the benchmarks in the analog electronic diagnosis domain. The faults considered are parametric, permanent, independent and simple, although the methodology can be extrapolated to catastrophic and multiple fault diagnosis. The method is only focused and tested on passive faulty components, but it can be extended to cover active devices as well.

Resum

El test de circuits és una fase del procés de producció que cada vegada pren més importància quan es desenvolupa un nou producte. Les tècniques de test i diagnosi per a circuits digitals han estat desenvolupades i automatitzades amb èxit, mentre que aquest no és encara el cas dels circuits analògics. D'entre tots els mètodes proposats per diagnosticar circuits analògics els més utilitzats són els diccionaris de falles. En aquesta tesi se'n descriuen alguns, tot analitzant-ne els seus avantatges i inconvenients.

Durant aquests últims anys, les tècniques d'Intel·ligència Artificial han esdevingut un dels camps de recerca més importants per a la diagnosi de falles. Aquesta tesi desenvolupa dues d'aquestes tècniques per tal de cobrir algunes de les mancances que presenten els diccionaris de falles. La primera proposta es basa en construir un sistema fuzzy com a eina per identificar. Els resultats obtinguts son força bons, ja que s'aconsegueix localitzar la falla en un elevat tant percent dels casos. Per altra banda, el percentatge d'encerts no és prou bo quan a més a més s'intenta esbrinar la desviació.

Com que els diccionaris de falles es poden veure com una aproximació simplificada al Raonament Basat en Casos (CBR), la segona proposta fa una extensió dels diccionaris de falles cap a un sistema CBR. El propòsit no és donar una solució general del problema sinó contribuir amb una nova metodologia. Aquesta consisteix en millorar la diagnosis dels diccionaris de falles mitjançant l'addició i l'adaptació dels nous casos per tal d'esdevenir un sistema de Raonament Basat en Casos. Es descriu l'estructura de la base de casos així com les tasques d'extracció, de reutilització, de revisió i de retenció, fent èmfasi al procés d'aprenentatge.

En el transcurs del text s'utilitzen diversos circuits per mostrar exemples dels mètodes de test descrits, però en particular el filtre biquadràtic és l'utilitzat per provar les metodologies plantejades, ja que és un dels benchmarks proposats en el context dels circuits analògics. Les falles considerades son paramètriques, permanents, independents i simples, encara que la metodologia pot ser fàcilment extrapolable per a la diagnosi de falles múltiples i catastròfiques. El mètode es centra en el test dels components passius, encara que també es podria extendre per a falles en els actius.

Index

List of Tables

List of Figures

Chapter 1

INTRODUCTION

1.1 Scope of the Thesis

There have been some Artificial Intelligence applications developed for diagnosing electronic circuits, but much remains to be done in this field, above all in the analog domain. The purpose of this thesis is not to give a general solution but to contribute with a new methodology. Our aim is to develop a methodology for analog circuit diagnosis based on improving the well-known fault dictionary techniques by means of adding or adapting new cases to develop a Case Based Reasoning system. As an example, two fault dictionary methods have been studied in detail. They have been extended for tolerance coverage, and then applied to a biquadratic filter. The faults considered are parametric, permanent, independent and simple.

In this thesis, Fault Detection and Isolation for linear analog circuits are the only tasks addressed, although the methodology can be extrapolated to multiple faults and non linear circuits. The faults are supposed to happen in the passive components. Nevertheless it is not difficult to consider other possible faulty parameters, such as an IC amplifier gain or an IC input impedance, for example.

The proposed diagnosis methodology can be used for other processes different to electronic circuits, taking into account the adequate set of measures necessary for diagnosing the system under test.

On the other hand, even though having the model of the system helps enormously when generating the initial data set, it is not strictly necessary. Data obtained from the real system can be used instead.

1.2 A Brief History of Circuit Testing

The first methods that applied fault detection and location to electronic circuits appeared during the sixties. From that moment on, many new application technologies and methodologies have emerged. Initially, electronic circuits were basically analog, their components were mounted on a board and they were tested by means of a bed of nails tester, allowing access to almost all input and output component voltages. Because of the technological advances, and the possibility of integrating transistors on a large scale, electronic circuits were made up on a chip. So, bigger and more complex global circuits could be built. But the accessibility to the internal IC (Integrated Circuit) nodes for fault detection and location purposes became more difficult. This is the reason for the development of a great deal of methods and technologies for testing these ICs during the 70s and 80s decades (Milor, 1998).

At the end of the 70s, digital circuits were being developed at vertiginous speed, and due to their growing importance and the easiness in testing them compared to the analog ones, new methods and improvements in testing analog circuits decreased in favor of the digital ones. One of the main problems of this period was computing speed. Most of the analog methodologies required a considerable volume of data management, and the computers were too slow to obtain satisfactory results.

But in the 80s and 90s, analog circuit tests gained importance once more. This was due to the explosion of the telecommunications market, multimedia and automotive electronics requiring mixed signal devices to be designed.

That is, integrating digital and analog components on a single chip. Although almost all the circuits are designed using digital technology, a lot of them have analog components (Wey, 1996), (Mir *et al.*, 1996), (Milor, 1998). This is due to the analog nature of the input and output signals they have to deal with (audio signals, signals from sensors, etc.). So, analog interfaces such as filters, AD/DA converters, PLL (Phase Locked Loop), modulators and demodulators were needed.

As circuits become daily more complex and larger, more complexity in test design is needed. The total cost of the circuits is augmented by a significant percentage due to costs in the test stage (Chandramouli and Pateras, 1996), (Milor and Sangiovanni-Vicentelli, 1994), (Murray and Hayes, 1996) and the time the test takes. According to (Pang and Starzyk, 2002), the high quality analogue tests are the most expensive in terms of both test development costs and test implementation. In the commercial market, up to 80% of the test costs are on account of the analogue functions that typically occupy only around 10% of the chip area.

Normally, a mixed-signal circuit test begins with the analog part, goes on to the digital part and finishes with some system tests to check the at-speed interaction between components. To date, the majority of time and cost associated with testing a mixed-signal system has been attributed to testing its analog portion, above all due to the difficulty in accessing it, normally requiring the addition of extra pins (Worsman and Wong, 2000).

Nowadays, the trend is to design circuits that are prepared for the tests. This technique is called Design For Testability (DFT) (Wey, 1996), (Schöber, 1995),(Novak and Biasizzo, 1994). There are several DFT structures, as for example BIST (Built-in Self Test) and the standard IEEE 1149.x. These techniques introduce special devices into the IC, that jointly with the addition of a few extra pins, make the test process more feasible and with access to almost all the internal nodes. These techniques allow the costs of the test phase to be reduced, and avoid the necessity of introducing extra expensive external equipment. At the same time, the test implementation is simplified because there is a certain methodology to follow.

Although the BIST technology has been investigated for 20-25 years, it has only recently been transferred to the industrial world. For example IBM and AT&T have developed their own software to apply these techniques (Chandramouli and Pateras, 1996). In the research field there is a lot of bibliography describing implementation, new structures and methods using the BIST technique and the IEEE 1149.x standard (Chandramouli and Pateras, 1996),(Chatterjee *et al.*, 1996),(Murray and Hayes, 1996),(Gomez *et al.*, 1996), but mostly applied to digital and integrated circuits. In particular the standard IEEE 1149.4 redefines some aspects in order to take into account the mixed-signal nature of the present IC signals, as described in (Kac *et al.*, 2003). But in spite of the advances in this domain, there is no widely accepted paradigm for analog testing or fault diagnosis even with the introduction of the IEEE 1149.4 standard for the mixed-signal test bus.

Test and diagnosis techniques for digital circuits have been successfully developed and automated. But, this is not yet the situation for analog circuits (Mir *et al.*, 1996), (Fanni *et al.*, 1999). Different papers show the complexity of testing of this kind of circuits (Bandler and Salama, 1985),(Dague, 1994). But the interest in testing them is evident. Lucas in (Lucas, 1996), demonstrating the importance of this subject for the industrial world, presented an industrial development for auto-testing circuits, the DVT-100. In (Kaminska *et al.*, 1997) the IEEE Mixed-Signal Technical Activity Committee proposed a common set of benchmarks for analog and mixed-signal circuits for use in research and analog fault modelling, test generation, and evaluation of DFT and BIST methodologies. The company Intusoft has launched the Test Designer Software, a package that allows simulating analog and digital faults, constructing a fault-tree and giving diagnoses for the components. The disadvantage is that this software is not open and only allows implementing separate fault dictionary based techniques.

On the other hand, Artificial Intelligence techniques (AI) did not play a major role during the first years of circuits testing. Some expert systems were developed for detecting board failures, but they were not successful at diagnosing components because of the large number of rules they needed, and the significantly large bottleneck in translating the expert knowledge into rules. Hence, there was no great activity in AI applied to electronic circuits from the 70s to the beginning of the 90s. But in this last decade, several papers addressed the electronic circuit diagnosis task using other AI methods, such as fuzzy, neural networks and Case-Based reasoning systems among others. The results are quite promising and open new ways to explore and research these fields. Hence, AI techniques have an important role to play in diagnosing analog electronic circuits and are fully compatible with the present DFT designs. This is demonstrated by the interest in developing standard integrating diagnosis and AI, such as the Standard IEEE 1232.x set (IEEE1232.2, 2002).

1.3 Types of Faults. Test Capabilities and Selection

The process of testing or diagnosing circuits consists in applying certain types of excitations to a circuit and then analyzing the responses obtained in order to derive a possible failure. A fault could be defined as any variation of a component value from the nominal, which could produce an abnormal behavior of the global circuit (Bandler and Salama, 1985). A typical Automatic Test Equipment (ATE) environment is the one shown in Figure 1.1.

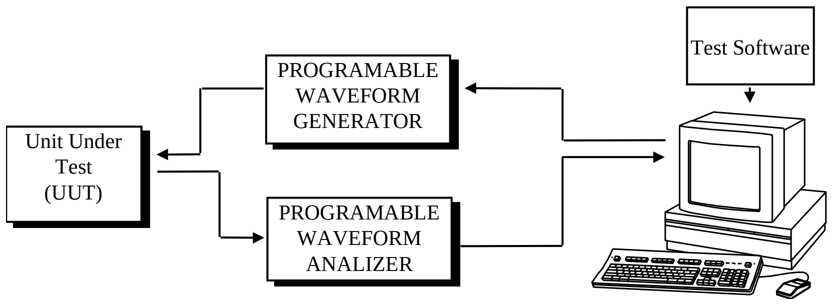


Figure 1.1: Basic structure of an ATE system

According to the figure, an ATE system should carry out the following basic actions:

1. *Signal generation*, using an external device such as an acquisition card, DSP board, waveform generator operated via a GPIB bus, etc. If we are talking about a BIST system, the stimulus signal will probably be generated by an internal IC unit. It is necessary to have a set of stored input signals that produce particular circuit responses that are useful for

fault detection and isolation. These input stimuli could be sinewaves, squarewaves, DC-signals, ramps, etc.

2. *Take measures from the circuit and obtain signatures.* The output response can be interpreted in time or frequency domains (Pan and Cheng, 1995). Once the measures are taken, some features must be extracted, for example a tuple $[overshoot, risetime]$ or a particular sequence corresponding to the signal shape. The set of parameters that characterize a signal is known as a *signature.* Sometimes signatures can not be directly determined from circuit measures. In this case, they are determined by means of adequate processing, as wavelet decomposition or fuzzy rules, for example.

3. *Interpretation of the obtained signatures.* These signatures can be compared with the ones previously stored, or used to derive possible parameter values. These tasks require an appropriate *diagnosis strategy.*

There are many papers in the bibliography concerning the testing or diagnosing of electronic circuits. The objective of these two disciplines is almost the same but, although they have a lot in common, there is an important difference between them. In general, the purpose of the test , as it is known in the industrial domain, is to detect a fault or faults in a circuit, while fault diagnosis is not only to detect but also to locate the fault or faults and identify the incorrect parameter values (Huang, 1998). Both domains are described in the following paragraphs.

Diagnosis Domain

From the fault diagnosis domain, the diagnosis system designed has to have some capability requirements. As described in (Chantler *et al.*, 1996), capability requirements are divided into Task and Fault requirements.

- *Task requirements*: They specify the basic diagnostic tasks to be performed. There are three basic diagnostic functions: Fault detection, fault isolation and fault identification. The same three stages are defined for diagnosing circuits, as described in (Pang and Starzyk, 2002). Using these three fundamental tasks, additional ones can be specified, such as fault explanation, fault remediation, fault prediction and fault simulation. Going into more detail on the diagnosis functions:

1. Fault Detection: This is the process by which abnormal behavior of the circuit is detected and flagged to the user. The final result is only to indicate whether the circuit is faulty or not.
2. Fault Isolation or location: This consists in localizing the faults to physical regions of the circuit, it is possible to arrive at the component level.
3. Fault Identification: This implies the estimation of circuit parameters.
4. Fault prediction: Circuit parameters are non linearly identified and monitored. Then, the fault can be predicted.
5. Fault Explanation: This involves the information generation, which allows the test engineer to understand the link between the current diagnosis and the circuit symptoms.
6. Fault Remediation: This consists in palliating the identified fault by replacing the wrong component or components.
7. Fault Simulation: This is the ability to simulate a hypothetical fault in a circuit. Normally it uses fault model output from the Fault Identification Process. There are several CAD tools that allow circuit behavior simulation. For example Intusoft has developed the Test Designer Software in order to simulate analog and digital circuit faults, or ORCAD, PROTEL or PCAD which all provide a graphical environment in order to simulate circuit responses to different input stimulus.

- *Fault requirements*: These requirements refer to the type of faults, the characteristics of the physical phenomena that produce the circuit deviation from the nominal, and the diagnosis properties the diagnosis system has to deal with.

  1. Type of Faults: This characteristic refers to fault coverage.
  2. The characteristics of the physical phenomena:
     - Relational properties: This refers to the properties of the fault-free relationships between physical phenomena of interest. They can be static or dynamic properties, linear or non-linear, time invariant or time variant, continuous or discrete, etc.
     - Discrepancy definition: There are three categories of discrepancy: a) Fault, a significant and easily detectable deviation

from the nominal. b) Performance degradation, a small drift in the operation over time that is often difficult to detect and separate from noise. c) Perturbation, a fluctuation over an operating point.

 – Symptom propagation: This concerns the direction in which the symptom can be propagated. The symptom is unidirectional if the propagation is only in one direction through the components. In case of feedback loops the symptom can be propagated in the reverse direction as well, and the propagation is said to be multidirectional.

3. Diagnosis system properties:
   – Granularity: This refers to the level at which the fault is specified (at sub-circuit, component, etc.).
   – Reliability: The ability of the diagnosis system to avoid false diagnostics.

Test Circuit Domain

The previous classification matches the one usually given for the circuit domain. There is a direct relationship between the classification of the *discrepancy definition* of the characteristics of the *physical phenomena* item and the fault classification usually given for circuits. The present classification is taken from (Duhamel and Rault, 1979). Although it is not a recent paper, its classification and concept descriptions are still used in the present. The proposed classification is as follows:

– Magnitude of Parameter variation: From this point of view the faults can be: parametric or catastrophic. The first ones take into account a continuous variation of circuit parameter values that can not be accepted because they produce abnormal behavior. Catastrophic faults are produced suddenly, and they produce large deviations of circuit parameters from the nominal values. For example a typical catastrophic fault could be a short circuit or an open circuit. Its equivalent in the diagnosis domain is given in the discrepancy definition on the characteristics of the physical phenomena item. Catastrophic faults correspond to a *Fault* and parametric faults to a *performance degradation*.

- Multiplicity: Depending on the number of faults produced, they can be classified into Simple, when only one component or parameter is affected, and Multiple, when more than one parameter or component have changed.
- Dependency: Two or more faults are dependent when one of them produces the others. Otherwise the fault is independent.
- Temporality: If the fault is constant, while it is not repaired, the fault is called permanent. Otherwise the fault will be intermittent (it appears temporary).

In the particular case of electronic circuits, the faults can be produced by:

- Problems in the components: There are two kinds, infancy faults and delayed faults. Infancy faults affect the component that has a defect from its construction. Delayed faults correspond to components that deteriorate during use.
- Wiring and Mounting: Some mistakes can be made during the mounting process, for example inverting a component, a short-circuit between conducting paths, bad soldering, etc. These errors can cause the circuit to fail.
- Design: As the design of an analog circuit is often anarchic without a specific methodology, errors at the design stage can be produced. Fortunately, the powerful software tools now available reduce this factor. The circuit can be simulated easily before its implementation.

After understanding the types of faults to detect and the basic scheme of a test system, if a circuit needs to be tested, a selection of the appropriate kind of test has to be done. This selection depends on:

1. *The type of signals to measure.* These signals can be used for frequency analysis, DC analysis or transients in the time domain.
2. *The system under test characteristics.* The model of the circuit to be analyzed is divided into two different parts: one of them describes the circuit structure through matrices or differential equations; and the other is the functional description represented by flux

diagram, transfer equations, and experimental data. The exactitude of the system description model will be crucial in obtaining good results.

3. *The type of faults to detect.* A description of a possible classification of fault types was made in the previous paragraphs. The type of faults to detect or locate is very important when selecting the type of test to apply.

According to these selection parameters, a test can be:

– *Concurrent or Non-concurrent.* If the test is performed while the circuit is working in its normal conditions, it is an on-line test- Otherwise, if the process has to be stopped before applying the test it is an off-line test.

– *Exhaustive or partial.* The test is exhaustive when all possible working modes are analyzed. On the other hand, if the analysis is only done on some of them, the test is called partial.

– *Verification type.* If the goal of the test is to analyze the performance of the circuit responses while the parameters are at their nominal values, it is called a functional test or purely, a test. But, if the verification consists in testing the circuit element values, the test is called parametric or diagnosis of the circuit.

It is possible to find failures that produce the same symptoms. This set of faults is called an *ambiguity group*, as defined in the bibliography (Bandler and Salama, 1985), (Stenbakken *et al.*, 1989). In order to clarify this concept, take the voltage divider of Figure 1.2 as an example.

The output voltage V_0 is given by

$$V_0 = \frac{R_2}{R_1 + R_2} V_i \tag{1.1}$$

Consider $V_i = 10V$ DC and $R_1 = R_2 = 10K$. For the nominal values, $V_0 = 5V$. If R_1 increases its value by 20% ($R_1 = 12K$), then $V_0 = 4.545V$. But the same V_0 value will be obtained if R_2 decreases 16.67%. Therefore, the faults R1+20% and R2-16.67% produce the same symptoms at node V_0 making

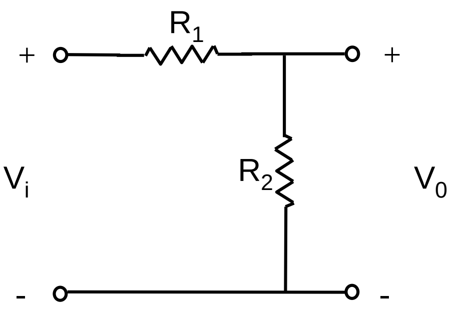


Figure 1.2: Voltage divider

it difficult to distinguish between them. This concept should be considered when deciding on the type of measures to take and where they will be taken from.

1.4 Decision Variables in Circuit Diagnosis

The main goal of a diagnosis system is to provide the correct diagnosis of the highest possible amount of faults in a circuit. It has to be done using the minimum number of stimuli signals, test points and circuit responses. Trying to cover 100% of the faults can generate huge test procedures that can be unaffordable. Hence, the following parameters should be defined:

- Number of nodes to measure.
- type of test (production, functionality, etc.) that best fits our purposes.
- Type of measures to take according to the kind of circuit under test and its topology.
- The sequence in which the stimuli signals will be applied in order to optimize the diagnosis time.

In order to help in the choice of the previous parameters, a measure of how the diagnosis will perform using such parameters can be obtained using the testability definition. Testability of a circuit refers to how well a fault in the circuit can be detected from measuring its voltages and/or currents at certain access points. This information, traditionally referred to as network-element-value-solvability, allows us to determine how many internal system parameters can be uniquely determined or isolated by measuring certain I/O relationships of the system. The method lets us know a priori if a unique solution of the problem exists. If this solution does not exist, the method gives a quantitative measure of how far we are from it (this means roughly how many components cannot be diagnosed with the given test point set). When testability is low, an important concept is that of ambiguity groups. Therefore, testability can provide a basis for selecting test points and test stimulus and choose the optimal combination.

The method proposed by (Sen and Saeks, 1979), is based on the sensitivity matrix calculation. The number of identifiable circuit parameters equals the number of independent columns of the sensitivity matrix. It has also been proven that the testability value is almost independent of the circuit parameter values. Hence, it can be evaluated by assigning randomly chosen integer values to the circuit parameters. For example, (Bandler and Salama, 1985) assign a value of 1 to all circuit parameters when calculating the testability.

1.5 Analog Versus Digital Circuits

There are plenty of methods to test or diagnose analog electronic circuits. Some of them are extrapolated from the digital domain, and others are exclusively for analog testing. Analog circuits are more complex to analyze because they have several differences from digital circuits that make them more difficult to test. Some of these differences are included in the following list:

- Analog circuits have to accomplish certain performances (bandwidth, gain, resonance frequency, etc.). It is not enough to test whether the circuit is working or not as the digital circuits do. There is a continuum of possible failures. This implies a lot of different faulty modes to detect making the diagnosis algorithm more complicated.

- Digital circuit design is more systematic. Design of analog circuits is more anarchic and so the testing methods are as well.

- Both measurements and component parameters are inaccurate. Analog systems are often non-linear, noisy and the components have a tolerance. These factors make deterministic test techniques insufficient. Device tolerance means that a fault is not characterized by a discrete value but by a band fault. Depending on the component value inside its tolerance range, a fault value can take any value in the band.

  These bands can have a partial overlap, a total overlap or non-overlap. It is clear that a significant overlap between bands makes diagnosis more difficult.

  For example, take the voltage divider in Figure 1.2. The output voltage V_0 is given by equation Eq. 1.1. Consider $V_i = 10V$ DC and $R_1 = R_2 = 10K$. For the nominal values, $V_{0nom} = 5V$. Considering a 10% tolerance of the components, the following values could be obtained:

	R_1 +10% R_2 +10%	R_1 +10% R_2 -10%	R_1 -10% R_2 +10%	R_1 -10% R_2 -10%
Voltaje V_0 (Volts)	5	4.2	5.78	5

  Table 1.1: DC values for the circuit divider

  So, the V_{0nom} value can fluctuate between $V_{0nom\ min} = 4.2V$ and $V_{0nom\ max} = 5.78V$. Suppose now that the circuit is faulty and R_2 increases its value to 1.5K (fault R_2+50). Then $V_{0R_2+50} = 6V$. But taking into account the 10% tolerance of R_1 this voltage can fall between $V_{0R_2+50\ min} = 5.77V$ and $V_{0R_2+50\ max} = 6.57V$. Therefore, as it can be seen in Figure 1.3, there is an overlap between the case circuit that is corret and the fault $R_2 + 50\%$. If the output measure is $V_0 = 5.8V$ it cannot be ensured if the circuit is correct or faulty with $R_2 = R_2 + 50\%$.

- The fault statistic distribution is not known with precision. So, the probabilistic methods have some disadvantages when being developed.

- Symptoms do not necessarily give information about the possible faults. For example, this is the case of oscillators, where most of the faults lead

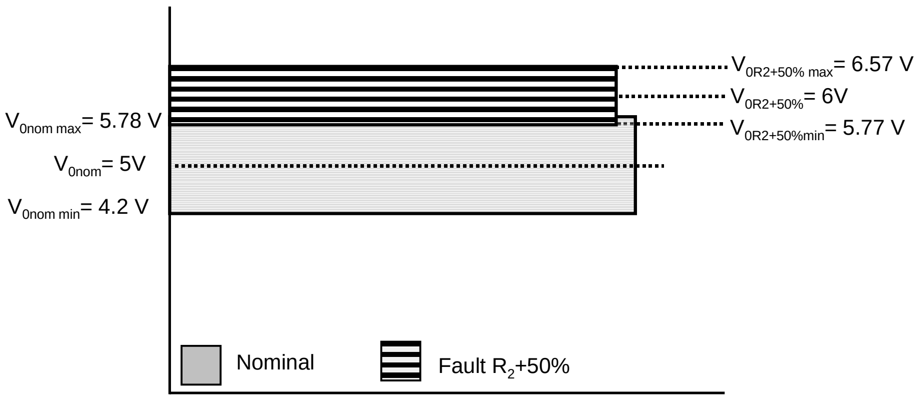


Figure 1.3: Fault band for the voltage divider

to the same type of symptom: The oscillator stops oscillating, and the signals in its nodes remain flat (Dague *et al.*, 1991).

- Multiple faults are very common, since the failure of one component can provoke the failure of another.
- A faulty component can produce other components to shift from their nominal values and still be correct. Thus, trying to model all the possible correct behaviors of the component becomes very complex.

As a conclusion, the brute force methods used in digital testing cannot be extrapolated directly to analog systems because they would be unrealistic. Only partial solutions have been obtained using digital methods in analog circuits, such as the method proposed in (Chantler *et al.*, 1996).

1.6 Related Works

In this section we summarize major works that have paved the way of analog electronic circuit diagnosis. Further details on the related literature are given in Chapters 2, 3 and 4.

In 1978 Schreiber (Schreiber, 1978) made the first Automatic Test Generation Techniques classification. But it was in 1979 when P. Duhamel and J.C.Rault (Duhamel and Rault, 1979) published a more exhaustive categorization of the known analog circuit testing methods. The types of tests and faults to diagnose were given and classified. The methods were grouped into estimation techniques, topological methods, taxonomical methods and methods for linear circuits.

In 1985 J.W. Bandler and A.E. Salama (Bandler and Salama, 1985) reported another excellent classification, including methods that had just appeared and the improvements obtained from them. This is one of the most referenced reviews for analog electronic circuit testing. They classify the methods into two main groups: techniques that need a simulation before the test, and the ones that need the simulation after the test. In this report AI techniques are briefly mentioned, basically because in 1985 they were not very developed and poor results were obtained.

Due to the relative simplicity of diagnosing digital circuits, some ideas for trying to translate the same procedures to analyze analog circuits appeared. For example, (Pan and Cheng, 1995) embed the analog circuit to be tested between a digital to analog converter and an analog to digital converter. Then, they use pseudo-random patterns as is usually done in digital testing. But these brute force techniques used in the digital domain do not perform as well as was expected. Another method that transforms the circuit into a digital model is described in (Zheng *et al.*, 1996).

While digital circuits are normally characterized by a very limited number of fault categories, the nature of analog circuits makes this universe of fault discretization more difficult, since this universe is continuous. The type of measures taken and the circuit topology define the degree of solvability of the circuit, and can be quantified by means of a testability measure. The most useful degree of testability calculation is the one proposed by (Sen and Saeks, 1979). It proposes the use of the column-rank of the system sensitivity matrix as a testability measure for parametric faults in linear analog circuits. The problem still remains far from being solved. It is a field that has a lot of literature, and a good summary of the techniques for deriving the testability is given in (Fedi *et al.*, 1999). It has to be said that the majority of the proposed methods are for linear circuits. As it could seem an important drawback, for mixed-signal circuits almost all the analog modules compounding the circuit are of this nature, while the non linear parts are moved to the digital part (Manetti and Piccirilli, 2003).

The ambiguity groups concept is completely related to the testability parameter. When testability is low, ambiguity groups begin to become important. One of the first algorithms for identifying ambiguity groups is given in (Stenbakken *et al.*, 1989). But a lot of contributions can be found to this domain, such as (Fedi *et al.*, 1999), (Starzik *et al.*, 2000), (Liu and Starzyk, 2002) or (Manetti and Piccirilli, 2003) among others.

As the complexity of the circuits increases, development of new designs for testability is getting more important. DFT techniques try to provide accessibility to internal nodes with the minimum addition of extra components or pins. (Soma, 1990), (Hatzopoulus *et al.*, 1993) and (Chatterjee *et al.*, 1996) are examples that describe some DFT schemes. Later, in (Chatterjee and Nagi, 1997), a classification of these techniques into two groups is made. The first one is based on the circuit reconfiguration in order to improve the testability. The second one relies on the insertion of test points to increase the controllability and observability of the circuit's internal nodes. (Shieh and Wu, 1998) review the main techniques proposed using Controllability and Observability Structures (COS) and describe two new ones. (Wen and Lee, 2001) present an algorithm that can systematically generate all possible COSs after the user defines certain requirements.

(Milor, 1998) presents a tutorial that reinforces the classification of (Bandler and Salama, 1985) and extends it by incorporating how the most recent DFT techniques, and in particular BIST, can be matched . (Hoffmann, 2002) also makes a good review of the evolution of BIST techniques to the present day and a new testability measure is given. In (Renovell *et al.*, 1998) there is a scheme for operational amplifiers configuration. The biquadratic filter is used as an example, and it is possible to use the operational amplifier as a classical one or to configure it as a follower for signal propagation in the circuit without affecting it.

Among all the DFT techniques the standard IEEE1149.x, and in particular the IEEE1149.4 for mixed-signal testing have to be cited. Its main objective is to incorporate in the digital standard IEEE1149.1 characteristics to be used for analog and mixed signal circuits. The standard is described in detail in (IEEE1149.4, 1999). Although there are several studies on this subject trying to analyze the impact of adding the analog blocks and how the standard performs in particular circuits (Loftstrom, 1996), (Calvano *et al.*, 2002), (Su *et al.*, 2003) or (Kac *et al.*, 2003), it is important to mention that, to date, it is not clear if the industry is going to accept and incorporate the standard, as is discussed in the editor's note of (Kac *et al.*, 2003).

About using AI techniques for electronic circuit diagnosis, in (Fenton *et al.*, 2002) the importance of intelligent tools for electronic circuit fault diagnosis is highlighted. A classification of these methods for electronic circuits is given. Basically, they are grouped into rule-based approaches, model-based approaches, learning approaches, other approaches and hybrid combinations of the previous ones. In this domain, there is plenty of work done using expert systems, neural networks, Fuzzy or a combination of them.

Concerning expert systems, (Dague *et al.*, 1991) proposes an expert system based on interval propagation in order to diagnose an astable multivibrator circuit. Also, (Preist *et al.*, 1992) describes the expert system that Hewlett-Packard used for testing a processor board during its production step.

On the other hand, in (Aminian *et al.*, 2002) a Neural network for fault detection in a Sallen-Key filter is described. The acquired data is previously processed using wavelet decomposition and principal component analysis to generate optimal features for training the neural network. (Fanni *et al.*, 1999) treats analog circuit diagnosis as pattern recognition. After a review of AI techniques applied to analog circuit diagnosis, they propose to train a neural network obtaining data from a circuit simulation SPICE environment. Also, several feature extraction techniques and their impact on the final diagnosis are shown. In particular a DC motor driver is tested. The faulty component is detected in (Deng *et al.*, 2000) by means of a Neural Network using the backpropagation architecture. The system under test is a resistive network circuit. In this case, the exact component deviations are not diagnosed. The output only indicates which component is considered to be faulty.

Analyzing the fuzzy techniques applied to analog circuit diagnosis, the work done in (Mohamed *et al.*, 1996) can be cited. The effect of treating the circuit parameter intervals as fuzzy sets for propagation considerations is studied. On the other hand (Catelani and Fort, 2002) proposes a fuzzy system where the inputs are based on a fault dictionary and the outputs are crisp or singleton functions from a Sugeno fuzzy system. (Berenji *et al.*, 2003) provides a similar fuzzy system but applied to the Hybrid Combustion Facility at the NASA Ames Research Center. Therefore, these systems can detect the incorrect component, but not its actual deviation.

Part of the present thesis is based on developing a fuzzy system for analog circuit diagnosis. Hence, there are some publications related to this subject. In (Pous and Colomer, 2001) and (Pous *et al.*, 2002), a fuzzy system is developed that uses information from the fault dictionary to build the input membership functions. There are as many outputs as circuit components. A Mandami fuzzy system with Gaussian shaped membership functions is utilized for each output. (Pous *et al.*, 2003*b*) improves some of these results and shows that the component value estimation, in some cases, is not as good as expected.

Case-Based Reasoning systems are another powerful tool that can be applied to the analog circuit diagnosis. The paper (Cunnigham and Smyth, 1994) presents a CBR system acting as a help-desk that makes the right questions in order to isolate the possible fault in the circuit. In (Cunningham *et al.*, 2003) an incremental case retrieval mechanism is designed for minimizing the number of initial cases necessary to initiates case retrieval with a brief case description, it is not necessary to have all the features available at the first steps. The CBR system is applied to diagnose a switching power supply module by guiding the appropriate questions to the user.

(Aamodt and Plaza, 1994) is one of the most referenced papers that defines the main CBR tasks (the CBR cycle) and describes how these tasks are decomposed into subtasks. Also, a classification of the main types of CBR methods is given. A short description of the previous work and commercial tools in this field is given in (Lopez de Mantaras *et al.*, 1997). At the same time, this paper identifies the main open problems concerning the retrieval, memory organization, matching, adaptation, forgetting and its integration with other methodologies, that still are research topics in the present days.

As the CBR system's core is a base of cases that can be very large, it has a lot in common with the data mining methodologies for data processing. In (Patterson *et al.*, 1998) data mining is used for generating databases for

CBR applications. In particular, and concerning this thesis, (Sheppard and Simpson, 1998) propose CBR systems as an extension of the fault dictionaries. They study the case applied to digital circuits and discuss the appropriateness of using the Nearest Neighbor criterion.

Retrieval is one of the tasks of the CBR on which a lot of researchers focus their attention. Concerning the metric used, three new heterogeneous distance functions are proposed in (Wilson and Martinez, 1997*b*). These new distance functions are designed to handle applications with nominal attributes, continuous attributes, or both. On the other hand in (Gupta and Montezemi, 1997), they present a modified form of the cosine matching function for retrieval and the results are compared with the ones obtained by the nearest-neighbor and Tversky's contrast matching. After clarifying the differences between similarity, similarity measures, and similarity metrics, (Finnie and Sun, 2002) review the relationship between them and propose a unified conceptual framework for the study of fuzzy similarity relations and similarity metrics. The intention is to facilitate the research and development of CBR and fuzzy logic with their applications. (Jarmulak *et al.*, 2000) focuses on reducing the retrieval effort using genetic algorithms to find the optimal parameters and to determine the relevance of case features. Something similar is done in (Pal *et al.*, 2000) using a neuro-fuzzy approach.

Feature weighting methods are also investigated for selecting or giving the appropriate importance to the corresponding attributes used for the distance/similitude calculation. In general the methods can be classified into global or local weighting. There is a good review of these methods in (Aha, 1998) or in (Wettschereck *et al.*, 1997). Also (Atkeson *et al.*, 1997) offers a review of the usual weighting functions for distance and how a local weight can be updated by learning from the provided data.

When the case base grows it is necessary to have a good maintenance policy. This means that we have to delete, add or modify cases or other type of knowledge in order to keep the system performing well. For example the deletion policy proposed by (Smyth and Keane, 1995) is well known, and uses the competence concept, as it does similarly in (Smyth and McKenna, 1999) and in (Brighton and Mellish, 2001). On the other hand, (Zhu and Yang, 1999) criticizes Smyth and Keane's deletion policy, because it can not guarantee a lower bound, where it stops deleting, meaning that in some cases the base coverage is worsened. On the contrary, they suggest using an addition policy, providing a way to find the lower bound that is close to the optimum.

Another deletion policies can be imported from the machine learning domain. Aha proposes several instance based learning algorithms in (Aha *et al.*, 1991), such as the well known IB3. In (Wilson and Martinez, 2000*a*), there is a phenomenal classification of the reduction algorithms grouped into incremental and decremental algorithms. They propose a decremental algorithm known as DROP that has been demonstrated to improve the IB3 in many situations. The book (Witten and Frank, 2000) collects many of these machine learning algorithms, as does (Ferrario and Smyth, 2000) in its review of maintenance methods. (Richter, Sesimbra, Portugal, October 25, 1995) introduce the concept of learning containers and describe where the learning can be produced in a CBR system. (Aha and Wettschereck, 1997), make a good description of these containers as well.

Also, the paper (Pous *et al.*, 2003*a*) has to be cited, which was proposed by the author of the present thesis. The paper uses the DROP algorithm for maintaining a case base for diagnosing a biquadratic filter, after defining the adequate case memory (case structure and hierarchy) and introducing the conflictive cases into another case base, where a different metric is used for retrieval.

To finish with this brief summary of related works, the standard IEEE1232 AI-ESTATE has to be taken into account, it stands for Artificial Intelligence Exchange and Service Tie to All Test Environments (IEEE1232.2, 2002). The purpose of this standard is "to standardize interfaces between functional elements of an intelligent diagnostic reasoner and representations of diagnostic knowledge and data for use by such diagnostic reasoners", as described in its abstract. Hence, the importance of the Artificial Intelligence applied to circuit diagnosis is emphasized.

1.7 Reference Circuit

In order to test the performance of the diagnosis methods developed in the present thesis, a particular circuit has been chosen. The circuit proposed for testing is a biquadratic filter cited in the bibliography, such as the one in (Cota *et al.*, 2000), (Balivada *et al.*, 1996), (Jurisic *et al.*, 1996), (Kaminska *et al.*, 1997), (Mir *et al.*, 1996). This benchmark is a linear system that can be found applied in several electronic schemes. It allows parametric and catastrophic faults due to passive and active components to be tested. The results obtained can be extrapolated to similar systems. The circuit is a low-pass filter, allowing the pass of frequencies from the DC to certain desired frequency. It can be used itself or as part of the leap-frog filter ((Kaminska *et al.*, 1997)). It is useful in audio and multi-media applications. The structure of the biquadratic filter is the one shown in Figure 1.4, with the component values given in Table 1.2.

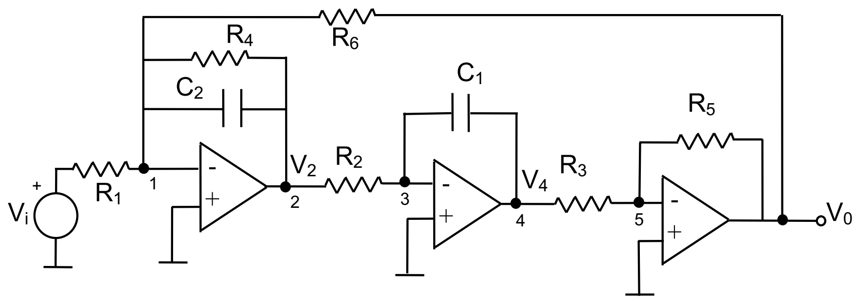


Figure 1.4: Biquadratic filter under test

Component	Value	Component	Value
R1	2.7 K	R5	12 K
R2	1 K	R6	2K7
R3	10 K	C1	10 nF
R4	1K5	C2	10 nF

Table 1.2: Biquadratic filter component values

These component values have been selected in order to make it possible to take measurements with the equipment available in our laboratory. The circuit is linear and only parametric faults of the passive components are considered.

In order to apply the selected methods to the biquadratic filter, first, the main characteristics and expressions that define its behavior are found. From its nodal equations, it is not difficult to derive the transfer functions at nodes

V_2, V_4 and V_0 with respect to the input V_i, taking into account that I_s=V_i/R_1. They are given in the set of equations 1.2.

$$\begin{aligned}
H_1(s) &= \tfrac{V_0}{V_i} = \tfrac{-G_2G_3G_1}{C_1C_2G_5s^2+G_4G_5C_1s+G_2G_3G_6} \\
H_2(s) &= \tfrac{V_2}{V_i} = \tfrac{-C_1G_5G_1s}{C_1C_2G_5s^2+G_4G_5C_1s+G_2G_3G_6} \\
H_3(s) &= \tfrac{V_4}{V_i} = \tfrac{G_2G_5G_1}{C_1C_2G_5s^2+G_4G_5C_1s+G_2G_3G_6}
\end{aligned} \tag{1.2}$$

If it is desired that these functions depend on Impedances, the set of equations 1.3 are obtained.

$$\begin{aligned}
H_1(s) &= \tfrac{V_0}{V_i} = \tfrac{-R_4R_5R_6}{R_1} \cdot \tfrac{1}{C_1C_2R_4R_2R_3R_6s^2+C_1R_2R_3R_6s+R_4R_5} \\
H_2(s) &= \tfrac{V_2}{V_i} = \tfrac{-R_2R_3R_4R_6C_1}{R_1} \cdot \tfrac{s}{C_1C_2R_4R_2R_3R_6s^2+C_1R_2R_3R_6s+R_4R_5} \\
H_3(s) &= \tfrac{V_4}{V_i} = \tfrac{R_3R_4R_6}{R_1} \cdot \tfrac{1}{C_1C_2R_4R_2R_3R_6s^2+C_1R_2R_3R_6s+R_4R_5}
\end{aligned} \tag{1.3}$$

1.8 Thesis Outline

The text is structured as follows: Chapter 2 provides a general classification of test methodologies. Several examples are reproduced from the original papers, and new ones are introduced in order to compare the results obtained by each method. Fault dictionaries are detailed further in Chapter 3. After that, Chapter 4 shows how the fault dictionary methods can be improved using fuzzy logic. Next, a Case Based Reasoning system is developed in Chapter 5. In Chapter 6 the methodology is validated in a real biquadratic filter circuit. Finally some conclusions are derived and future work is described in Chapter 7.

Chapter 2

FAULT DIAGNOSIS IN ANALOG CIRCUITS

2.1 Introduction

As stated in (Bandler and Salama, 1985) and (Mir *et al.*, 1996), fault location techniques are first classified according to the stage in the testing process at which simulation of the tested circuit occurs. Two general groups can be mentioned: Simulation Before Test (SBT) and Simulation After Test (SAT). The former is based on fault dictionaries. Faults have to be simulated (using software or taking measures from the circuit in which faults can be produced) and stored in a table before the test starts. Then, the measures obtained from the faulty circuit are processed and compared with the stored ones. The SAT tests use measurements from the faulty circuits to compute network parameters or locate the faulty components.

According to (Bandler and Salama, 1985) fault location techniques, in general, can be categorized into five groups as depicted in Figure 2.1.

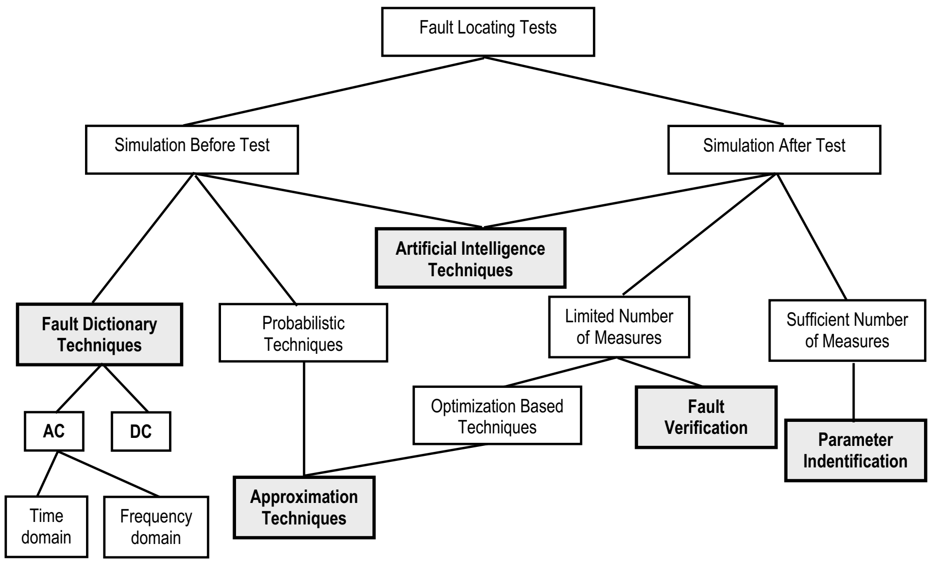

Fault Locating Tests
Simulation Before Test
Simulation After Test
Artificial Intelligence Techniques
Fault Dictionary Techniques
Probabilistic Techniques
Limited Number of Measures
Sufficient Number of Measures
Optimization Based Techniques
Fault Verification
Parameter Indentification
AC
DC
Time domain
Frequency domain
Approximation Techniques

Figure 2.1: Classification of fault locating test

1. Fault dictionary techniques
2. Approximation techniques
3. Fault verification techniques
4. Parameter identification techniques
5. Artificial intelligence techniques

The fifth group, AI techniques, is only mentioned in a short paragraph at the end of the Bandler's paper, and it does not appear in the scheme due to it not being a very relevant technique in the electronic circuit tests in that period. But, these techniques have gained importance in the present day. This is the reason why it has been introduced in Figure 2.1.

The following sections explain these groups in more detail, giving some examples to clarify the methodologies. Special emphasis on fault dictionaries and artificial intelligence techniques is taken in later chapters since they are the basis of the present thesis.

2.2 Fault Dictionary Techniques

Fault dictionaries are techniques completely based on quantitative calculations. Once the universe of faults to be detected is defined (Fault 1, Fault 2, ..., Fault m), selected characteristics of the measured or simulated output are obtained from the system for each considered fault and stored in a table (Table 2.1). This set of output characteristics is known as *Fault signature*. The groups of fault signatures considered constitute the *Fault dictionary*.

Fault set	Measure 1	Measure 2	...	Measure n
Nominal	M10	M20	...	Mn0
Fault 1	M11	M21	...	Mn1
Fault 2	M12	M22	...	Mn2
...	...	...	...	...
Fault m	M1m	M2m	...	Mnm

Table 2.1: Dictionary appearance

The measures obtained from an unknown faulty system are compared with the fault signatures. The comparison is typically performed using the neighborhood criterion, obtaining distances, minimizing certain indexes, and so on. So, the method has two steps: the first one is based on simulation in order to built the dictionary; the second one consists in comparing the measures from the unknown faulty system with the stored fault signatures.

It has to be taken into account that the non-considered faults will not be detected. Mainly the pre-simulated faults will be detected and located. Examples and further details of these techniques can be found in Chapter 3.

2.3 Approximation Techniques

There are two main categories: *Approximation techniques* that obtain a fault probability index related to each component, and techniques that estimate system parameters, including the faulty ones, from a reduced number of measurements. Both approaches have limited accuracy depending on the quality of the estimation that is used. The different methods are based on the following general formulation.

Consider an analogue circuit with n parameters x_i (i=1, 2,...n) in which m measures y_j (j=1,2,...m) are taken,

$$y_j = f_j(x_1, x_2, ..., x_n) \tag{2.1}$$

If the deviations of the parameter circuit from the nominal value are small, a first order Taylor's series expansion of the output magnitude expression (Stenbakken *et al.*, 1989) can be used. Then, for a given pulsation ω, a measured output magnitude could be expressed as:

$$y(\omega) \approx x_1(\omega)b_1 + x_2(\omega)b_2 + x_3(\omega)b_3 + ...x_n(\omega)b_n \tag{2.2}$$

with $i = 1, 2, ...n$ parameters and $y(\omega)$ is the difference from the nominal value, $x_i(\omega)$ is the sensitivity matrix elements and b_i is the deviation from the nominal for each component i.

If several measures $y_j(\omega)$, $j = 1, 2, ..., m$ are taken, then the previous expression can be written in a matricial form as

$$\Delta y \approx A\Delta x \tag{2.3}$$

where Δy is the vector containing the difference from the nominal values, A is the sensitivity matrix and Δx is the vector which contains the parameter deviation from its nominal value. The number of measures m should be less than the number of parameters to estimate (m<n).

Among all the possible techniques that can be classified in this range, the associated criterion, the minimal deviation, the pseudo-inverse and the l1-norm are explained as an example.

- **Associated criterion:** For this criterion, a factor of merit S_i is defined as

$$S_i = \sum_{j=1}^{m} (g_i - y_j(X_i))^2 \quad j = 1, 2, ..., m \quad (measures), \tag{2.4}$$

where parameter g_j is the measurement of the characteristic y_j, and X_i is the vector of components x_1,x_2,...,x_n with their nominal values except for x_i. Each component $i = 1, 2, ..., n$ has a merit factor S_i associated to it. The goal is to minimize the merit factor given in Eq. 2.4. So, the steps of the method are:

1. Measure the value g_i from the faulty circuit (real values).
2. Calculate y_j (simulating the circuit and evaluating the measure i) considering the parameter x_i as a variable (for example several points between the nominal value and $\pm 50\%$ from the nominal). For each x_i considered, evaluate the merit factor S_i. Keep the minimum.
3. Repeat step 2 for the other components of the circuit.
4. Search for the S_i minimum between the minimum merit factors obtained for each component.
5. The faulty component will be the one with the minimum S_i.

This method has good results for simple faults and short deviations from the nominal value. It can be applied to linear and non-linear circuits.

Example 1

Let us take the circuit in Figure 2.2 as an example

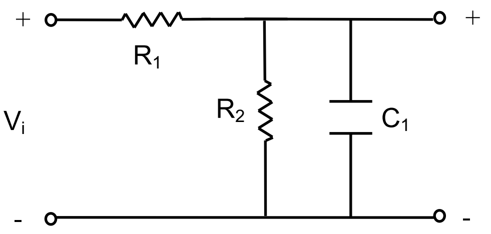


Figure 2.2: R-C network for the associated criterion example

with the values $R_1 = 10K$, $R_2 = 10K$ and $C_1 = 1\mu F$. The circuit has a transfer function

$$H(s) = \frac{V_0}{V_i} = \frac{R_2}{R_2 C_1 R_1 s + (R_1 + R_2)} \tag{2.5}$$

It is a first order system with a pole at $\omega = 200\ rad/sec$. Measures at pulsations $\omega = 10\ rad/sec$ and $\omega = 1000\ rad/sec$ are used. The measures taken will be only amplitudes at node V_0. Now, a deviation up to $\delta = \pm 40\%$ from nominal taking $N = 100$ points is considered for each component x_i while other components stay at their nominal value, and the voltage measured $y_j(x_i)$ for each component x_i deviation point and frequency ω_j is stored. For example, if the component tolerance is $Tol = \pm 5\%$ the step considered will be

$$\Delta R = \frac{\delta \cdot R - Tol \cdot R}{N} = 70 \tag{2.6}$$

for R_1 and R_2, and

$$\Delta C_1 = \frac{\delta \cdot C_1 - Tol \cdot C_1}{N} = 8.4 \cdot 10^{-9} \tag{2.7}$$

R_1, R_2 and C_1 take the values shown in Table 2.2.

R_1	6070	6140	6210	6280	...	13790	13860	13930
R_2	6070	6140	6210	6280	...	13790	13860	13930
C_1(μF)	0.607	0.614	0.621	0.628	...	1.379	1.386	1.393

Table 2.2: Component points explored

Table 2.3 shows the output measures V_0 for the component values of Table 2.2, while other components stay at their nominal value.

R_1	6070	6140	6210	6280	...	13790	13860	13930
$V_0(\omega=10)$	0.6218	0.6191	0.6165	0.6138	...	0.4196	0.4184	0.4172
$V_0(\omega=1000)$	0.1593	0.1575	0.1558	0.1541	...	0.0715	0.0711	0.0708
R_2	6070	6140	6210	6280	...	13790	13860	13930
$V_0(\omega=10)$	0.3775	0.3801	0.3828	0.3855	...	0.5787	0.5799	0.5811
$V_0(\omega=1000)$	0.0967	0.0967	0.0968	0.0968	...	0.0986	0.0986	0.0986
C_1 (μF)	0.607	0.614	0.21	0.628	...	1.379	1.386	1.393
$V_0(\omega=10)$	0.4998	0.4998	0.4998	0.4998	...	0.4998	0.4998	0.4998
$V_0(\omega=1000)$	0.1565	0.1549	0.1533	0.1517	...	0.0718	0.0714	0.0711

Table 2.3: Measures at V_0 for the explored points

They are stored, and everything is ready to start with the faulty circuit. Now, suppose that $C_1 = 1.2\mu F$ ($C_1 + 20\%$). Measuring V_0 at $\omega = 10\ rad/sec$ and $\omega = 1000\ rad/sec$, the results obtained are

$$g_1 = V_0(\omega = 10) = 0.4990\ Volts$$
$$g_2 = V_0(\omega = 1000) = 0.1319\ Volts$$

So, the merit factor associated with each component can be calculated using equation 2.4 and Table 2.3, obtaining Table 2.4

S_1	0.0251	0.0240	...	$3.274\cdot10^{-4}$	...	...	0.0068	0.0071
S_2	0.0149	0.0142	...	$5.192\cdot10^{-4}$	...	...	0.0072	0.0074
S_3	0.0071	0.0068	...	...	$6.819\cdot10^{-8}$	...	$2.082\cdot10^{-4}$	$2.224\cdot10^{-4}$

Table 2.4: Component point associated merit factor S_i

The minimum merit factor S_i associated with each component is then

S_{1min}	S_{2min}	S_{3min}
$3.274\cdot10^{-4}$	$5.192\cdot10^{-4}$	$6.819\cdot10^{-8}$

As a conclusion, the faulty component will be C_1, since the minimum S_i is given by S_3.

Doing the same for $R_1 = 7K$ ($R_1 - 30\%$), the following merit indexes S_i are obtained

S_{1min}	S_{2min}	S_{3min}
0.0042	0.0170	0.0128

Clearly, the minimum factor is S_1, so, the faulty component is R_1. The problem arises when the faulty component has a value outside the deviation margin considered when building the pre-stored values. In the present example, the deviation is $\pm40\%$. If $R_1 = 16K$, a deviation of 60% from the nominal, the minimum S_i indexes obtained are:

S_{1min}	S_{2min}	S_{3min}
0.0021	$9.160\cdot10^{-4}$	0.0132

Hence, the final diagnostic is wrong, since the minimum index is obtained for R_2. Therefore, the deviation factor should be taken into account depending on the fault margin to be detected. Also, the method is only useful for locating the fault but not for diagnosing it.

- **Minimal Deviation method:** This consists in minimizing the index V_j

$$V_j = \frac{1}{m}\sum_{i=1}^{m}(\Delta x_{ij})^2 - \left(\frac{1}{m}\sum_{i=1}^{m}\Delta x_{ij}\right)^2 \qquad (2.8)$$

where m is the number of measurements and the component x_{ij} deviation, Δx_{ij}, is given by

$$\Delta x_{ij} = \frac{\Delta y_i}{A_{ij}} \qquad (2.9)$$

with $\Delta \mathrm{y}_i$ the measure i taken, and A_{ij} the elements of the sensitivity matrix. The elements of the normalized sensitivity matrix A_{ij} can be calculated as (Slamani and Kaminska, 1995), (Boyd, 1999)

$$A_{ij} = \frac{x_i}{y_i}\frac{\Delta y_i}{\Delta x_i} \approx \frac{x_i}{y_i}\frac{\partial y_i}{\partial x_i} \quad \forall\ measure\ i \qquad (2.10)$$

That calculation can be done either using deviations from the nominal values $\Delta \mathrm{y}_i$ obtained in the real circuit or using the derivatives from the y_i mathematical expression. This approximation is valid only for linear circuits or for small component deviations ($\Delta x_i \rightarrow 0$).

Also, if the measure taken is the magnitude given by a transfer function H_i, the sensitivity for linear circuits or small deviations can be rewritten as

$$A_{ij} = \frac{x_i}{y_i}\frac{\partial y_i}{\partial x_i} = \frac{x_{ij}}{|H_i(j\omega)|}\frac{\partial\,|H_i(j\omega)|}{\partial x_{ij}} \qquad (2.11)$$

If $\mid \Delta \mathrm{y}_i| < \epsilon_i$, where ϵ_i is a user considered interval, the circuit under test is said to be correct. Otherwise, the minimum V_j computed gives the faulty component.

Example 2

Let us consider the circuit of Figure 2.2. In order to evaluate the index V_j using equation 2.8, first, the components of the sensitivity matrix A_{ij} must be obtained. This will be done by means of equation 2.11 and the transfer function $H(s)$ given by equation 2.5. The measures used are $\omega_1 = 10\ rad/sec.$ and $\omega_2 = 1000\ rad/sec.$ amplitudes at node V_0. For the nominal circuit, the values obtained are

$$V_{0nom}(\omega_1) = 0.4994V$$
$$V_{0nom}(\omega_2) = 0.0981V$$

Hence, applying equation 2.11,

$$A_{i1} = \frac{R_1}{|H|}\frac{\partial|H|}{\partial R_1} = \frac{-R_1\left(R_1+R_2+R_1R_2^2C_1^2\omega^2\right)}{R_1^2+2R_1R_2+R_2^2+R_2^2C_1^2R_1^2\omega^2}$$

$$A_{i2} = \frac{R_2}{|H|}\frac{\partial|H|}{\partial R_2} = \frac{R_1(R_1+R_2)}{R_1^2+2R_1R_2+R_2^2+R_2^2C_1^2R_1^2\omega^2}$$

$$A_{i3} = \frac{1/C_1}{|H|}\frac{\partial|H|}{\partial C_1} = \frac{R_1^2R_2^2C_1^2\omega^2}{R_1^2+2R_1R_2+R_2^2+R_2^2C_1^2R_1^2\omega^2}$$

In this particular case, $i = 1, 2$, corresponds to the amplitude at the selected $\omega = [\omega_1\ \omega_2]$. Then, the following sensitivity matrix can be written

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \end{pmatrix} = \begin{pmatrix} -5012.5 & 0.4987.5 & -249.4 \\ -9807.7 & 192.3 & -961.5 \end{pmatrix}$$

Consider now that the circuit is faulty with $C_1 = 1.2\ \mu F$. The measures obtained for the faulty circuit are:

$$V_{0f}(\omega_1) = 0.4991V$$
$$V_{0f}(\omega_2) = 0.0822V$$

Hence,

$$\Delta y = \begin{pmatrix} V_{0f}(\omega_1) - V_{0nom}(\omega_1) \\ V_{0f}(\omega_2) - V_{0nom}(\omega_2) \end{pmatrix} = \begin{pmatrix} -2.7375 \cdot 10^{-4} \\ -0.0159 \end{pmatrix}$$

Using equation 2.9, the elements x_{ij} are:

$$|A| = \begin{pmatrix} 5.4610 \cdot 10^{-8} & -5.4880 \cdot 10^{-8} & 1.0981 \cdot 10^{-7} \\ 1.6170 \cdot 10^{-6} & -8.2468 \cdot 10^{-5} & 1.6494 \cdot 10^{-8} \end{pmatrix}$$

Finally, from equation 2.8, the index obtained for each component is

$$\begin{gathered}V_1 = 6.1023 \cdot 10^{-13} \text{ (corresponding to } R_1)\\ V_2 = 1.6980 \cdot 10^{-9} \text{ (corresponding to } R_2)\\ V_3 = 2.1768 \cdot 10^{-15} \text{ (corresponding to } C_1)\end{gathered}$$

The minimum coefficient V_j is V_3. So, the faulty component considered will be C_1.

Repeating the process for the fault $R_2 = 15\ K$ for example, the measures obtained in the faulty circuit are

$$V_{0f}(\omega_1) = 0.5989V$$
$$V_{0f}(\omega_2) = 0.0986V$$

and the final coefficients V_j

$$\begin{gathered}V_1 = 9.8018 \cdot 10^{-11} \text{ (corresponding to } R_1)\\ V_2 = 7.1708 \cdot 10^{-11} \text{ (corresponding to } R_2)\\ V_3 = 3.9860 \cdot 10^{-10} \text{ (corresponding to } C_1)\end{gathered}$$

So, the faulty component will be R_2, since it is the component with the minimum V_j.

The difficulty of the method is when tolerances are taken into account. For example, if in the previous case $R_1 = 10.8\ K$ (a value that falls into 10% of component tolerance margins), the indexes V_j obtained are:

$$\begin{gathered}V_1 = 7.064410^{-11} \text{ (corresponding to } R_1)\\ V_2 = 6.408510^{-10} \text{ (corresponding to } R_2)\\ V_3 = 2.632210^{-10} \text{ (corresponding to } C_1)\end{gathered}$$

Clearly, the faulty component considered will be R_1, when it was R_2. Also, the interval ϵ_i should be adequately selected. This is not a straightforward decision because the results are very sensitive to this parameter.

- **Pseudo inverse method**

  From equation 2.3, the component deviations can be obtained as

$$\Delta x = A^{+}\Delta y \tag{2.12}$$

where A^+ is the pseudo-inverse matrix calculated as $A^+ = A'(AA')^{-1}$. The faulty component will be the component associated with the maximum Δx obtained (largest deviation from the nominal).

The sensitivity matrix A has to be composed from independent rows for system solving. Otherwise, the pseudo-inverse matrix does not exist. In the case that a dependence is found, one possible solution is to eliminate dependent rows before the matrix inversion. It can be done by finding the ambiguity groups and deleting the corresponding rows, leaving just one for each group.

Example 3

Let us consider the circuit in Figure 2.2. The measures to be taken are the same used in example 2, giving the same sensitivity matrix A. Then, calculating the pseudo-inverse A^+,

$$A^+ = A'(AA')^{-1} = \begin{pmatrix} 1.0072 \cdot 10^{-4} & -5.1207 \cdot 10^{-6} \\ 1.1026 \cdot 10^{-4} & -5.6160 \cdot 10^{-6} \\ 1.0770 \cdot 10^{-6} & 0.2340 \cdot 10^{-7} \end{pmatrix}$$

If the circuit is faulty with $R_2 = 12\ K$, the vector Δy for $\omega_1 = 10\ rad/sec$ and $\omega_2 = 1000\ rad/sec$ is

$$\Delta y = \begin{pmatrix} V_{0f}(\omega_1) - V_{0nom}(\omega_1) \\ V_{0f}(\omega_2) - V_{0nom}(\omega_2) \end{pmatrix} = \begin{pmatrix} -2.7375 \cdot 10^{-4} \\ -0.0159 \end{pmatrix}$$

Then applying equation 2.12

$$\Delta x = \begin{pmatrix} \Delta R_1 \\ \Delta R_2 \\ \Delta C_1 \end{pmatrix} = \begin{pmatrix} 4.5579 \cdot 10^{-6} \\ 4.9896 \cdot 10^{-6} \\ -5.0485 \cdot 10^{-8} \end{pmatrix}$$

The maximum absolute value index is obtained in ΔR_2, pointing to the conclusion that R_2 is faulty.

It has to be said that the ΔR_1 value is not far from being the maximum Δx, so, when tolerances are considered, a margin ϵ has to be taken. If $|\Delta x| > \epsilon$, the component is considered faulty. For the previous example, ΔR_1 and ΔR_2 could be seen as faulty due to this factor.

- **L1-norm method:** This is based on using linear programming techniques in order to find the most likely faulty element in an analog circuit. This technique assumes that the catastrophic faults have already been eliminated.

  The problem of fault isolation is approached better by first locating the parts of the network that contain the faulty elements. Then, further diagnosis is carried out on these subnetworks to locate the exact faulty element. It can be used for linear circuits and it can be easily extended to non-linear ones.

  The undetermined system to solve is

  $$Minimize \sum_{i=1}^{n} \left|\Delta I_i^{bf}\right| \tag{2.13}$$

  with the constraints

  $$\Delta V^m = Z_{mb} \Delta I^{bf} \tag{2.14}$$

  with n the number of components. ΔV^m is a vector containing the difference from the nominal voltage value for each measurable node ($\Delta V^{mi} = V_i - V_{i0}$), ΔI^{bf} is the change in a current through a component due to a fault, and Z_{mb} is the impedance matrix that relates current variations in each component to the difference voltage in measurable nodes.

  Z_{mb} can be computed before the test, with the nominal component values admittance matrix Y_n and the incidence matrix Q as follows:

  $$Z_{mb} = Y_n^{-1} Q \tag{2.15}$$

  The solution provided by 2.14 and 2.13 is ΔI^{bf}. With this value, the component deviation from the nominal can be calculated by

  $$\Delta I^{bf} = V_i \cdot \Delta G_i \Rightarrow \Delta G_i = \frac{\Delta I^{bf}}{V_i} \tag{2.16}$$

  and finally, the estimated component value is given by

  $$G_i \approx G_{i0} + \Delta G_i \tag{2.17}$$

The component i with its value G_i outside the tolerance bounding, is considered as faulty.

The method can be summarized in the following steps:

1. Calculate the admittance matrix Y_n and the incidence matrix Q from the nominal circuit.
2. Derive Z_{mb} using eq. 2.15.
3. Obtain the vector ΔV^m from the measures V_i and the nominal voltages V_{i0}.
4. Solve the problem

   $$Minimize \sum_{i=1}^{n} \left|\Delta I_i^{bf}\right|$$

   subjected to

   $$\Delta V^m = Z_{mb} \Delta I^{bf}$$

   The solution of this problem provides us with the deviation current values ΔI^{bf}.
5. Using eq. 2.16 and ΔI^{bf}, calculate the component deviation ΔG.
6. Estimate the component value G by means of equation 2.17.

Example 4

The ladder network provided in Figure 2.3 is used as an example.

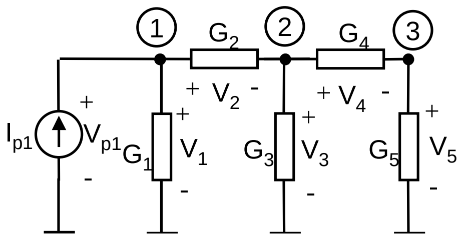


Figure 2.3: Ladder circuit for the L1 norm

Suppose that the voltages at nodes 1, 2 and 3 are the only measures available. Z_{mb} can be obtained by means of the equation 2.15. So, Y_n has to be calculated first:

$$Y_n = \begin{pmatrix} G_1+G_2 & -G_2 & 0 \\ -G_2 & G_2+G_3+G_4 & -G_4 \\ 0 & -G_4 & G_4+G_5 \end{pmatrix} = \begin{pmatrix} 2 & -1 & 0 \\ -1 & 3 & -1 \\ 0 & -1 & 2 \end{pmatrix}$$

$$Y_n^{-1} = \tfrac{1}{8}\begin{pmatrix} 5 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 5 \end{pmatrix}$$

and the incidence matrix Q for the circuit of Figure 2.3

$$Q = \begin{pmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 1 & 1 & 0 \\ 0 & 0 & 0 & -1 & 1 \end{pmatrix}$$

Using equation 2.15

$$Z_{mb} = Y_n^{-1}Q \Rightarrow Z_{mb} = \tfrac{1}{8}\begin{pmatrix} 5 & 3 & 2 & 1 & 1 \\ 2 & -2 & 4 & 2 & 2 \\ 1 & -1 & 2 & -3 & 5 \end{pmatrix}$$

ΔV^m is obtained from circuit measures, and ΔI^{bf} is given as the system solution.

Let $G_1 = G_2 = G_3 = G_4 = G_5 = 1$ for the nominal circuit. Suppose that the faulty component is $G_2 = 1.5$. Then, from the faulty circuit $V_1 = 0.6842$, $V_3 = 0.2105$ and $V_5 = 0.1053$. Hence,

$$V_2 = V_1 - V_3 = 0.4737$$
$$V_4 = V_3 - V_5 = 0.1053$$

Knowing that in the nominal circuit $V_{10} = 0.6250$, $V_{30} = 0.2500$ and $V_{50} = 0.1250$, V_{20} and V_{40} can be calculated as

$$V_{20} = V_{10} - V_{30} = 0.3750$$
$$V_{40} = V_{30} - V_{50} = 0.1250$$

So, if $V^{mi} = V_i - V_{i0}$, then the vector ΔV^m is

$$\Delta V^m = \begin{pmatrix} \Delta V^{m1} \\ \Delta V^{m3} \\ \Delta V^{m5} \end{pmatrix} = \begin{pmatrix} 0.0592 \\ -0.0395 \\ -0.0197 \end{pmatrix}$$

Now the problem to solve is (particularizing 2.13 and 2.14)

$$Minimize \sum_{i=1}^{5} \left|\Delta I_i^{bf}\right| (n = 5\ components)$$

subjected to

$$\begin{pmatrix} \Delta V^{m1} \\ \Delta V^{m2} \\ \Delta V^{m3} \end{pmatrix} = \frac{1}{8} \begin{pmatrix} 5 & 3 & 2 & 1 & 1 \\ 2 & -2 & 4 & 2 & 2 \\ 1 & -1 & 2 & -3 & 5 \end{pmatrix} \begin{pmatrix} \Delta I_1^{bf} \\ \Delta I_2^{bf} \\ \Delta I_3^{bf} \\ \Delta I_4^{bf} \\ \Delta I_5^{bf} \end{pmatrix}$$

The solution, ΔI^{bf}, to the linear programming problem is

$$\Delta I^{bf} = \begin{pmatrix} 1.7774 \cdot 10^{-5} \\ 0.1579 \\ -3.5280 \cdot 10^{-5} \\ -8.2494 \cdot 10^{-5} \\ 1.7506 \cdot 10^{-5} \end{pmatrix}$$

From the measures V_i and the recently obtained ΔI^{bf}, the component deviation can be derived using 2.16

$$\Delta I^{bf} = V_i \cdot \Delta G_i \Rightarrow \Delta G_i = \frac{\Delta I^{bf}}{V_i}$$

$$\Delta G = \begin{pmatrix} \Delta G_1 \\ \Delta G_2 \\ \Delta G_3 \\ \Delta G_4 \\ \Delta G_5 \end{pmatrix} = \begin{pmatrix} 2.5977 \cdot 10^{-5} \\ 0.3333 \\ -1.6758 \cdot 10^{-4} \\ -7.8369 \cdot 10^{-4} \\ 1.6631 \cdot 10^{-4} \end{pmatrix}$$

Finally, applying equation 2.17

$$G_1 = 1,\ G_2 = 1.3333,\ G_3 = 0.9998,\ G_4 = 0.9992 \text{ and } G_5 = 1.0002.$$

The conclusion is that G_2 is faulty, since it is the component with the largest deviation from its nominal value.

Trying with G_4 =0.5 for example, the results of the L_1-norm technique are:

$$V_1 = 0.6154,\ V_3 = 0.2308 \text{ and } V_5 = 0.1538$$

From this we can obtain

$$V_2 = V_1 - V_3 = 0.3846, \text{ and } V_4 = V_3 - V_5 = 0.0769$$

and solving the linear programming problem (2.13 and 2.14),

$$\Delta I^{bf} = \begin{pmatrix} -9.8923 \cdot 10^{-6} \\ 9.8923 \cdot 10^{-6} \\ -1.8837 \cdot 10^{-5} \\ -0.0768 \\ 2.8729 \cdot 10^{-5} \end{pmatrix}$$

Hence

$$\Delta G = \begin{pmatrix} -1.6075 \cdot 10^{-5} \\ 2.5720 \cdot 10^{-5} \\ -8.1626 \cdot 10^{-5} \\ -0.9980 \\ 1.8674 \cdot 10^{-4} \end{pmatrix}$$

and then

$$G_1 = 1,\ G_2 = 1,\ G_3 = 0.9999,\ G_4 = 0.002,\ G_5 = 1.0002$$

The faulty component is clearly G_4, in spite of the error in the final estimation value.

If there is a multiple fault, the L1-norm method can work as well. For example, let us take G_1=2, and G_3=0.5;

The measures obtained in the circuit of Figure 2.3 are V_1= 0.8889, V_3= 0.3333 and V_5= 0.1111. Voltages at the non measurable nodes V_2 and V_4 are calculated as

$$V_2 = V_1 - V_3 = 0.5556$$
$$V_4 = V_3 - V_5 = 0.2222$$

Solving 2.13 and 2.14

$$\Delta I^{bf} = \begin{pmatrix} 0.3863 \\ 0.0582 \\ 2.8327 \cdot 10^{-7} \\ 0.0581 \\ -0.0530 \end{pmatrix}$$

Applying equation 2.16

$$\Delta G = \begin{pmatrix} 0.4346 \\ 0.1048 \\ 8.4980 \cdot 10^{-7} \\ 0.2615 \\ -0.4769 \end{pmatrix}$$

At last, using equation 2.17

$$G_1 = 1.4346,\ G_2 = 1.1048,\ G_3 = 1,\ G_4 = 1.2615,\ G_5 = 0.5231$$

Here, the fault in G_1 and in G_5 could be detected, but G_4 and G_2 could be faulty as well if a component tolerance is taken as 10%, for example. In this case, more equations should be used in order to obtain a sharper solution.

2.4 Parameter Identification Techniques

When the number of measures is sufficient, the parameters of the system values can be calculated using *parameter identification techniques*. These values are obtained from circuit measurements after simulation. The number of parameters that can be identified in a particular circuit is called the testability or diagnosability degree.

One of the first approaches to derive this degree was proposed by (Sen and Saeks, 1979). They demonstrated that the testability degree is given by the column-rank of the sensitivity matrix. They also demonstrated that if a transfer function is rational in its parameters ϕ, then the column-rank of the sensitivity matrix is constant. Hence, the testability is independent of the circuit parameter and can be evaluated by simply randomly assigning values to the circuit parameters.

A similar procedure to find the circuit diagnosability degree μ is proposed in (Bandler and Salama, 1985). It is defined as

$$\mu = n_\phi - \rho \tag{2.18}$$

where n_ϕ is the number of circuit parameters and ρ the rank of the matrix

$$\nabla_\phi H^T(\phi, s) \tag{2.19}$$

with $H(\phi, s)$ the transfer function of the system depending on the parameters ϕ and the complex variable s. The matrix is particularized with a set of parameters $\phi = \phi^*$. This set can be randomly selected, since the parameter values are not going to influence the testability calculations (Sen and Saeks, 1979). If $\mu = 0$, all the circuit parameters will be estimated. Otherwise identification of all the components will not be possible. Let's see an example of diagnosability calculation.

Example 5

Suppose the following circuit is being diagnosed

The circuit parameter, ϕ, are (R_1, R_1, C), and they are unknown. The transfer function of the circuit, when V_2 is taken as the measurable node, is given by

$$H(s) = \frac{V_2}{I_g} = \frac{sCR_1R_2}{1 + sC\left(R_1 + R_2\right)} \tag{2.20}$$

Applying equation 2.19,

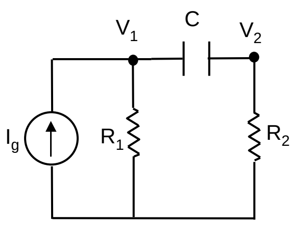


Figure 2.4: Parameter identification

$$\nabla_\phi H^T(\phi,s) = \left[\begin{array}{ccc} \frac{sCR_2(1+sCR_2)}{(1+sC(R_1+R_2))^2} & \frac{sCR_1(1+sCR_1)}{(1+sC(R_1+R_2))^2} & \frac{sR_2R_1}{(1+sC(R_1+R_2))^2} \end{array}\right]$$

Now, taking arbitrarily $\phi = [111]^T$,

$$\nabla_\phi H^T(\phi,s) = \left[\begin{array}{ccc} \frac{s(1+s)}{(1+2s)^2} & \frac{s(1+s)}{(1+2s)^2} & \frac{s}{(1+2s)^2} \end{array}\right]$$

The rank of $\nabla_\phi H^T(\phi,s)$ is clearly $\rho = 2$. Therefore, it will be impossible to determine all the network parameters for this transfer function (only 2 of them).

A similar conclusion is derived if the testability matrix rank is evaluated, as proposed first by (Sen and Saeks, 1979). If the magnitude at V_2 is taken as measure

$$|V_2(j\omega)| = \frac{\omega CR_1R_2}{\sqrt{1+\omega^2C^2\left(R_1^2+R_2^2\right)^2}}$$

the sensitivity for each circuit parameter taking the same arbitrarily selected parameters ($R_1 = R_2 = C = 1$) is

$$S_{V_2}^{R1} = \frac{R_1}{|V_2(j\omega)|}\frac{\partial|V_2(j\omega)|}{\partial R_1} = \frac{1+\omega^2C^2R_2(R_1+R_2)}{1+\omega^2C^2(R_1+R_2)^2} = \frac{1+2\omega^2}{1+4\omega^2}$$

$$S_{V_2}^{R2} = \frac{R_2}{|V_2(j\omega)|}\frac{\partial |V_2(j\omega)|}{\partial R_2} = \frac{1+\omega^2 C^2 R_1(R_1+R_2)}{1+\omega^2 C^2 (R_1+R_2)^2} = \frac{1+2\omega^2}{1+4\omega^2}$$

$$S_{V_2}^{C} = \frac{1/C}{|V_2(j\omega)|}\frac{\partial |V_2(j\omega)|}{\partial C} = \frac{1}{1+\omega^2 C^2 (R_1+R_2)^2 C^2} = \frac{1}{1+4\omega^2}$$

Hence, as the column-rank of the sensitivity matrix

$$rank\begin{pmatrix} S_{V_2}^{R1} & S_{V_2}^{R2} & S_{V_2}^{C} \end{pmatrix} = rank\begin{pmatrix} \frac{1+2\omega^2}{1+4\omega^2} & \frac{1+2\omega^2}{1+4\omega^2} & \frac{1}{1+4\omega^2} \end{pmatrix} = 2$$

there are only 2 identifiable parameters.

Let's now take node V_1 as a measurable node. Applying equation 2.19 again

$$H(s) = \frac{V_1}{I_g} = \frac{R_1(sCR_2+1)}{1+sC(R_1+R_2)}$$

as a transfer function, and then,

$$\nabla_\phi H^T(\phi, s) = \begin{bmatrix} \frac{s(1+s)^2}{(1+2s)^2} & \frac{s^2}{(1+2s)^2} & \frac{-s}{(1+2s)^2} \end{bmatrix}$$

the rank now is $\rho = 3$, and hence $\mu = 0$. According to (Bandler and Salama, 1985) this implies that all the circuit parameters can be estimated. A similar conclusion is obtained using the testability matrix column-rank concept.

$$S_{V_1}^{R1} = \frac{R_1}{|V_1(j\omega)|}\frac{\partial |V_1(j\omega)|}{\partial R_1} = \frac{2+3\omega^2}{2(1+2\omega^2)}$$

$$S_{V_1}^{R2} = \frac{R_2}{|V_1(j\omega)|}\frac{\partial |V_1(j\omega)|}{\partial R_2} = \frac{\omega^2\left(1+3\omega^2\right)}{2(1+2\omega^2)(1+\omega^2)}$$

$$S_{V_1}^{C} = \frac{C}{|V_1(j\omega)|}\frac{\partial|V_1(j\omega)|}{\partial C} = \frac{\omega^2}{(1+2\omega^2)(1+\omega^2)}$$

Observing these results, the rank of the sensitivity matrix is

$$rank\begin{pmatrix} S_{V_1}^{R1} & S_{V_1}^{R2} & S_{V_1}^{C}\end{pmatrix} = rank\begin{pmatrix}\frac{2+3\omega^2}{2(1+2\omega^2)} & \frac{\omega^2\left(1+3\omega^2\right)}{2(1+2\omega^2)(1+\omega^2)} & \frac{\omega^2}{(1+2\omega^2)(1+\omega^2)}\end{pmatrix} = 3$$

meaning that 3 parameters can be estimated using the magnitude at V_1 as a measure point.

Once the diagnosability of a circuit is calculated, there are several methods for trying to solve the n parameter identification problem. The most straightforward one is to obtain as many measures as parameters to be identified. Then a set of n equations with n unknowns has to be solved. An ARMAX identification model can be used for example.

The following example illustrates how the diagnosable values of the Figure 2.4 circuit can be derived, simply by solving a system of equations provided by several measures at node V_1

Example 6

Taking the circuit of Figure 2.4, it has a diagnosability of 3 (all parameters determined) if measures at node V_1 are taken.

First, the transfer function module is calculated

$$|H(j\omega)| = \frac{V_1}{I_g} = \sqrt{\frac{R_1^2+(\omega C R_2 R_1)^2}{1+\omega^2 C^2 (R_1+R_2)^2}}$$

and changing

$$\begin{aligned}\alpha_1 &= R_1^2\\ \alpha_2 &= (C R_1 R_2)^2\\ \alpha_3 &= C^2\left(R_1+R_2\right)^2\end{aligned} \tag{2.21}$$

the module can be rewritten as

$$|H(j\omega)| = \sqrt{\frac{\alpha_1 + \omega^2 \alpha_2}{1 + \omega^2 \alpha_3}}$$

Three measures are necessary to obtain all the parameters ϕ of the circuit. A linear system with three equations with three unknowns. Taking measures at $\omega = 10\ rad/sec$, $\omega = 50\ rad/sec$ and $\omega = 100\ rad/sec$, the following values are derived:

$$|H(j10)| = 6.76 \cdot 10^4 \quad |H(j50)| = 2 \cdot 10^4 \quad |H(j100)| = 1.28 \cdot 10^4$$

Hence,

$$\alpha_1 = 9.61 \cdot 10^9 \ \alpha_2 = 8.95 \cdot 10^5 \ \alpha_3 = 0.011$$

Finally from 2.21,

$$R_1 = 9.8 \cdot 10^4 \ R_2 = 9.83 \cdot 10^3 \ C = 9.87 \cdot 10^{-7}$$

These values can be compared with the nominal ones. The component or components that differ more than the allowed tolerance from the nominal, are considered faulty.

A similar solution is obtained using the ARMAX identification process, for example. The method provides us with the transfer function coefficient values that best approach a given circuit response. Afterwards, these coefficients are equaled to the corresponding components equivalence, as done in the example with the alpha parameters.

The method works quite well, but several drawbacks should be taken into account. First of all, the number of necessary measures to derive the component values is high. Also, the system does not give a solution in all the situations. It is necessary that the inverse of the matrix that constitutes the system equations has a determinant different to 0. This means that the rows are linearly independent, a situation that it is not always true. In this case, not all the components could be found, and a combination of other methods are necessary to solve the problem, if it is possible.

2.5 Fault Verification Techniques

These assume that there are a limited number of measures in the sense that there are not enough to determine all the network parameters. They use network theory, and mathematical theory to check the consistency of an equation set after circuit simulation. An inconsistency implies a fault in one or more of the components related to the equation. Logical relations between inconsistency conclusions give the faulty component. The decomposition approach for large circuits can be classified in this group. Although there are several methods that can use voltage or current measures, the method selected uses voltages as measures to be taken. In general, currents are more difficult to measure.

Let us take the biquadratic filter described in Section 1.7 as an example. In order to detect a possible fault or faults, the consistency of the equations has to be checked. In particular a KCL applied to node $n2$ gives the equation 2.22 in the Laplace domain.

$$V_2 G_2 = -V_4 C_1 s \tag{2.22}$$

The idea is to take measures at V_2 and V_4 and the components G_2 and C_1 at their nominal values. If the equation is satisfied, the components implied (G_2 and C_1) are considered to be not faulty. Otherwise, one of them or both are faulty.

Suppose that there are nominal conditions. Using a signal with a frequency $\omega = 10^4 rad/sec$ and amplitude 1 V as input, the measures obtained at nodes $n2$ and $n4$ are

$$V_2 = -0.0128 - 0.0833j$$
$$V_4 = 0.8327 - 0.1278j$$

Checking equation Eq. 2.22 with G_2 and C_1 at their nominal values

$$V_2G_2 = I_2 = -1.28 \cdot 10^{-5} - 8.327 \cdot 10^{-5}j$$
$$V_4C_1s = I_4 = 1.28 \cdot 10^{-5} + 8.327 \cdot 10^{-5}j$$

Hence, it can be seen that equation Eq. 2.22 is consistent. Consequently, R_2 and C_1 are not faulty. If a fault at another component different to R_2 and C_1 occurs, the consistency still works. Let's suppose that there is a fault $R_6 = 5K$. The following result is obtained

$$V_2G_2 = I_2 = -4.3 \cdot 10^{-5} - 1.485 \cdot 10^{-4}j$$
$$V_4C_1s = I_4 = 4.3 \cdot 10^{-5} + 1.485 \cdot 10^{-4}j$$

Therefore, equation Eq. 2.22 is still consistent and R_2 and C_1 are considered correct, although V_2 and V_4 are not the nominal values.

Let's see what happens in the case that one of the components related to the node $n2$ is faulty. Suppose $C_1 = 1nF$. Now, Eq. 2.22 with the faulty circuit measures, V_2 and V_4, and using G_2 and C_1 nominal values produces

$$V_2G_2 = I_2 = -1.25 \cdot 10^{-7} - 8.3487 \cdot 10^{-6}j$$
$$V_4C_1s = I_4 = 1.25 \cdot 10^{-6} + 8.3487 \cdot 10^{-5}j$$

Therefore, the consistency is broken, since $V_2G_2 \neq -V_4C_1s$. There is a fault in R_2, C_1 or both.

Something similar occurs with R_3 and R_5 (node $n5$). The KCL equation in that node is

$$V_4G_3 = -V_0G_5 \tag{2.23}$$

Taking G_3 and G_5 with their nominal values

$$\begin{aligned} V_4G_3 &= 8.327 \cdot 10^{-5} - 1.28 \cdot 10^{-5}j \\ V_0G_5 &= -8.327 \cdot 10^{-5} + 1.28 \cdot 10^{-5}j \end{aligned}$$

Therefore, the equation checking is correct. So there is no fault neither at component R_3 nor at R_5. But, if R_3 is considered faulty with $R_3 = 20K$, therefore, calculating the equations using G_3 and G_5 nominal values,

$$\begin{aligned} V_4G_3 &= 1.588 \cdot 10^{-4} - 4.989 \cdot 10^{-5}j \\ V_0G_5 &= -7.94 \cdot 10^{-5} + 2.49 \cdot 10^{-5}j \end{aligned}$$

there is no equation consistency, meaning that R_3, R_5 or both are faulty.

The method works quite well for fault detection. The main drawback is that the topology and the equations of the circuit under test have to be known. On the other hand, the method is not adequate for diagnosis, although it can help in the preliminary stages isolating the faulty component or the faulty part of the circuit.

2.6 Artificial Intelligence Techniques

Due to the increase in circuit complexity, system malfunction detection and isolations are becoming more difficult. Artificial Intelligence (AI) techniques have been a major research topic over the last decades. In (Fenton *et al.*, 2001) a good review of AI techniques is shown. The paper proposes the classification of the methods shown in Figure 2.5.

Traditional Approaches: These are the most common techniques used in the industry, and are based on heuristics. The expert experience is collected in a IF-THEN rule base or as a decision tree. These approaches have been well tested and they are very simple and understandable. On the other hand, faults that are not predicted in advance will not be detected. Hence, in this aspect the method has no learning capability.

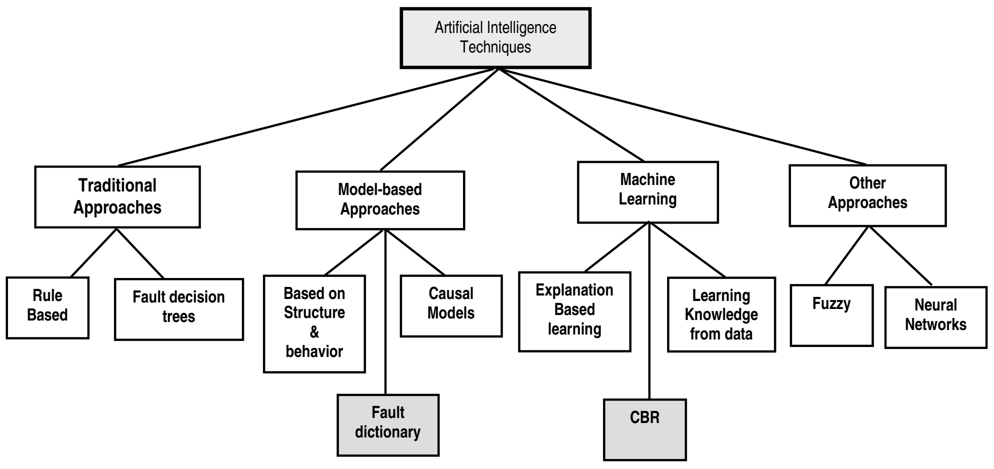


Figure 2.5: AI approaches classification

Model-Based Approaches make use of the model to predict faults in the real circuit. Its main disadvantage is its inability to deal with unsimulated faults and the expert knowledge acquisition when causal models are needed. Since they are not able to learn from new situations, they have a fixed performance level. Fault dictionaries are included in this group.

Machine Learning Approaches take advantage of previous successful or failed diagnoses, and they use this knowledge in order to improve the system's performance. But, in general, big data bases with suitable data are necessary if good results are desired. Case-Based reasoning (CBR) systems can be classified in this group. The main advantage of these techniques is that they can cope with new situations because they can learn from faults that have not been previously predicted.

Other Approaches. Fuzzy and Neural Network techniques can be cited in this group. The former provides a very intuitive way of representing knowledge, and they are normally combined with other techniques. Neural Network techniques have the power to model unknown predicted faults when they are trained.

Hybrid Approaches are one of the research fields that has major interest, in particular the combined use of models and cases in particular situations.

There are several possible combinations:

- Model-Based and Case-Based Reasoning
- Model-Based Reasoning and Fuzzy Logic
- Case-Based Reasoning, Artificial Neural Networks and Fuzzy Logic
- Model-Based Reasoning and Genetic Algorithms

Chapter 4 is dedicated entirely to showing how a fuzzy technique is developed and applied in order to diagnose a biquadratic filter. On the other hand, a CBR system is described in Chapter 5. The following section uses the idea proposed in (Aminian *et al.*, 2002) to classify faults by means of neural networks.

2.6.1 Diagnosis Using Neural Networks

Neural networks are able to adjust their parameters iteratively during the training phase, until all the training data produce relatively close outputs to the desired ones, measured by a predefined error-goal. After the training, the neural network is able to classify previously unseen inputs. The neuron structure is given in Figure 2.6

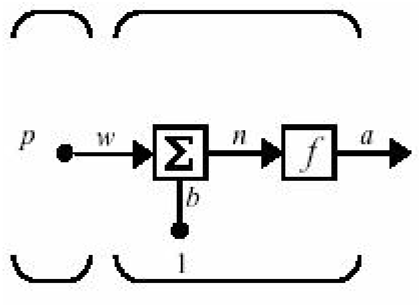


Figure 2.6: Neuron structure

where P is the input and W the weight that multiplies this input. Then, a bias b is added giving an internal value n. This internal value n is passed through a transfer function f, giving the output value a. The weight W and the bias b are the adjustable parameters. Hence, the output is calculated using equation eq. 2.24

$$a = f(n) = f(P \cdot W + b) \tag{2.24}$$

Of course, a simple neuron is not useful for solving complex problems. Figure 2.7, extracted from (Demuth and Beale, 2000), shows a two layer neural network structure. The inputs P_i $(i = 1, 2, 3, ..., R)$ correspond to the features extracted from the output circuit. Hence, there will be as many inputs as measures. Each of these inputs is multiplied by a weight W_{1i}, added to other weighted inputs and to a bias $b1_{si}$. The obtained internal value $n1_i$ serves as the input to the transfer function f_1 providing an output $a1_i$. These outputs are the inputs of the following layer, and the same explanation can be applied to the end of the network.

The structure of the neurons in the same layer is equal for all of them, but can be different between layers. Even the number of neurons contained in a layer can differ. The bigger the network is, the closer its behavior is to the desired behavior, although the training and the posterior test procedures will be slower.

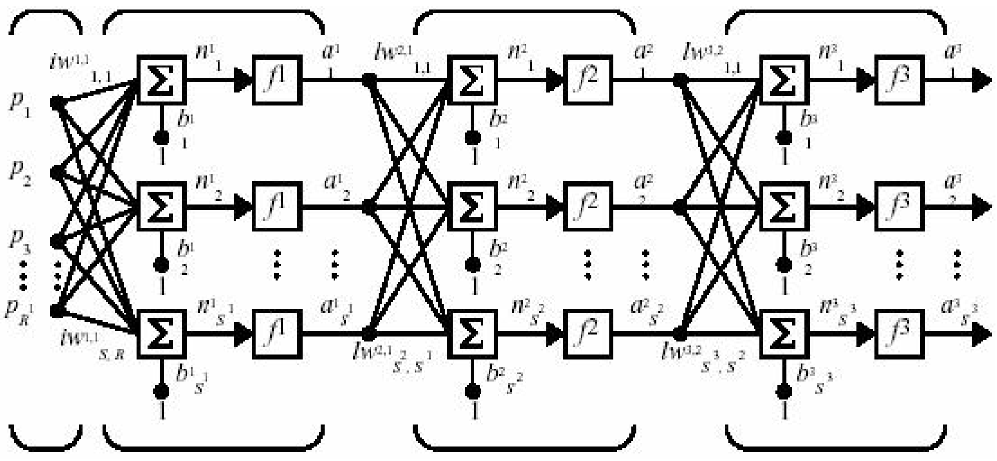


Figure 2.7: Architecture of a multiple layer feedforward neural network

Example 7

Taking the biquadratic filter as an example, a three layer neural network is designed. Suppose that the saturated ramp method described in Chapter 3 is used to obtain the output measures. In this case we have 4 measures: SP, T_d, T_r and V_{est}. Then, we take 4 neural network inputs. The network is built up by 4 layers. The first three of them with a $tan - sigmoid$ transfer function (output from -1 to 1) and the last with $log - sigmoid$ (output from 0 to 1). The hidden layers are constituted by 100 neurons each. The training method used is based on standard numerical optimization that uses the conjugate gradient, and the error considered for stopping the training process is taken as $M.S.E. = 10^{-4}$. For training purposes, a set of 3 cases for each considered fault has been used. Taking a great number of cases produces overtraining of the net, and a bad diagnosis of new previously unseen instances. Figure 2.8 shows an example of how the performance evolves during the training. After 132 epochs, the error reaches its maximum tolerated limit of $M.S.E. = 10^{-4}$.

Once trained, the obtained net is tested with a set of 100 new cases for each considered fault randomly generated (Monte-Carlo) considering a normal probability distribution. The variation is only performed on the tolerance margin for the components that are not faulty. The faulty one stays at its

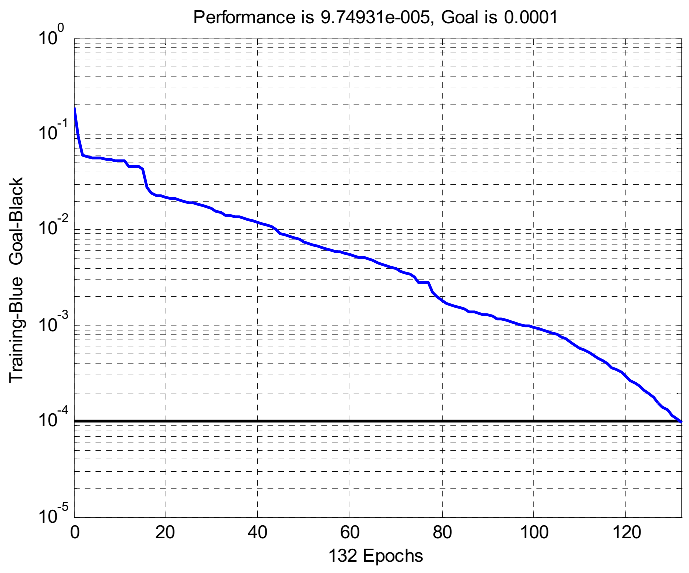


Figure 2.8: Neural network training performance evolution

exact deviation value. Table 2.5 shows the performance of the net when locating faults.

The first column is related to 100 new randomly generated cases taking the components with a tolerance of 5% normally distributed; the second considers a component tolerance of 10%.

The results show that a tolerance of 5% gives much better results. Hence, the neural network designed is useful for low deviations from the predicted faults. Changing the transfer function or the training method type does not produce a significant improvement in these results.

When the new cases correspond to any deviation between 0-70% for each component, the success in locating them drops drastically. When a set of 100 faults for each component considering deviations compressed into the range of 0-70% is taken, the average success in locating the faults decreases to 62%.

As a conclusion, neural networks are quite useful but only for locating small component deviations. The structure of the network quickly gets complicated with the circuit complexity. Also, the number of the hidden layers and the

Fault	5% tolerance	10% tolerance
R1+20	97	78
R1-20	100	88
R1+50	100	97
R1-50	100	100
R2+20,R3+20,C1+20	73	31
R2-20,R3.20,C1.20	62	38
R2+50,R3+50,C1+50	97	78
R2-50,R3-50,C1-50	100	92
R4+20	93	70
R4-20	99	92
R4+50	100	100
R4-50	100	100
R5+20	52	29
R5-20	36	40
R5+50	97	88
R5-50	99	89
R6+20	96	64
R6-20	100	95
R6+50	100	98
R6-50	100	100
C2+20	99	72
C2-20	100	93
C2+50	100	99
C2-50	100	100
NOM	94	62
Average	**91.76**	**79.72**

Table 2.5: Neural network performance for a set of 100 new cases

number of neurons that they should have is not clear, however there is a straightforward relationship with circuit complexity, the number of layers and layers size. The training and diagnosis process get slower when big neural networks are required.

2.7 Conclusions

Approximation techniques are useful only for short deviations from the nominal, since large ones usually produce incorrect solutions. On the other hand, the approximation techniques based on solving a linear programming problem are not easy to implement, above all when frequency based measures are taken.

Parameter identification techniques need more nodes where they can perform measures rather than simply measuring at the output. The circuit model has to be known, and it is not always available. Also, the obtained system of equations has to be solvable.

Something similar occurs with fault verification techniques. Equations describing the circuit behavior have to be available. Hence, the circuit model has to be known. These techniques can be good candidates for locating faulty components, but there is not enough information to diagnose the faults.

Concerning the artificial intelligence methods, neural networks have been demonstrated to be useful for short deviations. Also, their complexity increases dramatically with circuit size, making the system very slow when training and when diagnosing. A part from this, they cannot diagnose faults that have not been previously considered. Other AI techniques, such as Fuzzy logic and Case-Based reasoning, are analyzed in more detail in later chapters.

Fault dictionaries are very simple to apply, but they have certain drawbacks that need to be solved. Among all the previously described methods, they are the ones that can be most easily extended to build a CBR-system. This is the reason why fault dictionaries have been selected as a starting point and studied in more detail in the next chapter.

Chapter 3

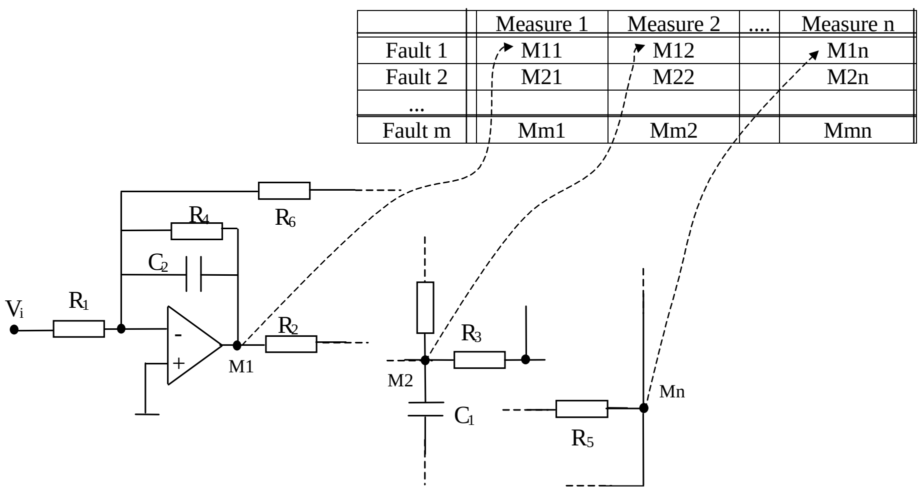
FAULT DICTIONARIES

3.1 Introduction

Although new techniques have been introduced into the industry, fault dictionaries were by far the most widely used technique for testing circuits in the past and continue to be today. They are simple and work quite well for fault detection. There are basically two steps: first, it is necessary to obtain the fault signatures to build the dictionary; and second, the most similar fault signature to the new situation presented is extracted.

The dictionary can be generated by simulating the most likely circuit faults before the test or by obtaining real measures from a prototype circuit. This simulation allows to define the stimuli set and the signatures of the responses to be stored in order to detect and/or isolate the faults. The test for the faulty circuit is done using the same stimuli used when building the dictionary. The simulation provides us with a set of responses related with each fault (Figure 3.1).

The dictionary must be built according to input stimuli, the domain of the analysis (DC, time or frequency domain, etc.) and the signatures. An optimum selection of a limited number of measurements is required for a high testability degree using moderate dimensions. According to (Bandler and Salama, 1985), they can be classified into a DC domain or an AC domain, depending on the type of measures used. The AC group is divided into two subgroups: dictionaries based on time response and dictionaries that use frequency based measures.



	Measure 1	Measure 2	....	Measure n
Fault 1	M11	M12		M1n
Fault 2	M21	M22		M2n
...				
Fault m	Mm1	Mm2		Mmn

Figure 3.1: Dictionary construction

Then, the signature derived from the circuit being tested is compared to the pre-stored ones using tree-decision-based techniques, neighborhood rule, voting techniques, etc. The pre-stored fault in the table most similar to the measured one will be the final conclusion.

A lot of methods corresponding to these techniques can be found in the literature. The following sections detail some of the most commonly used fault dictionaries. First, an example of the DC domain is given. Then, examples based on time and frequency response are given for the AC domain.

3.2 DC Domain

They consist in taking DC signals as input stimuli and measuring DC signals at different nodes. Some DC based methods can be found in (Chatterjee *et al.*, 1996) and (Hochwald and Bastian, 1979). Hochwald proposes checking the effectiveness of the selected stimuli using the Euclidean distance concept given in equation 3.1.

$$d_f = \sqrt{\sum_{j \epsilon M} \left(V_j^0 - V_j^f\right)^2} \tag{3.1}$$

V_j^0 is the nominal DC voltage at node j, while V_j^f is the voltage at node j due to a fault f. M is the set of nodes selected for measuring and F the set of possible failures considered. According to (Hochwald and Bastian, 1979), if the value of *df* is less than 0.5 times the number of nodes selected to be measured (heuristic threshold), it is considered that there are not enough stimuli to provide efficient information about the fault f. Then it will be necessary to include more stimuli to the circuit or try to excite the circuit using other nodes.

Example 8

In the transistor based circuit of Figure 3.2,

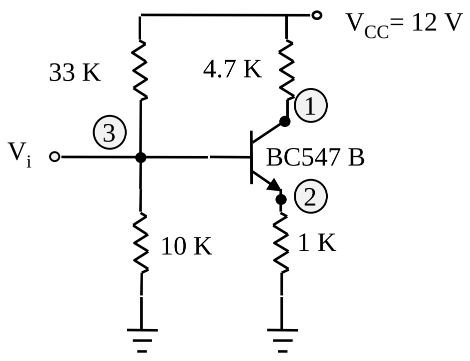


Figure 3.2: DC application

some transistor catastrophic failures can be detected, such as a short-circuit between base-emitter (Q_{BES}), collector-emitter (Q_{CES}) or collector-base (Q_{CBS}) or an open-circuit in the base (Q_{BO}), emitter (Q_{EO}) or collector (Q_{CO}). The transistor is an npn BC547 B.

For testing purposes, the stimuli are DC signals of $\pm 5\ V$ applied to the V_i input. Table 3.1 shows the fault signatures obtained in the circuit nodes 1, 2 and 3 for each fault.

	Stimulus	Non Faulty	Q_{EBS}	Q_{CES}	Q_{CBS}	Q_{BO}	Q_{EO}	Q_{CO}
Node 1	-5	3.1	3.1	2.17	-5	8.27	3.11	-0.96
	5	4.22	12	4.22	5	8.27	12	4.21
Node 2	-5	0	-5	2.17	0	0.85	-1.75	0
	5	4.22	5	4.22	4.29	0.85	6.29	4.21
Node 3	-5	-5	-5	-5	-5	1.44	-5	-5
	5	5	5	5	5	1.44	5	5

Table 3.1: DC measures for the transistor circuit

Applying 3.1 to this case, it can be seen that the *df* is greater than 1.5 (0.5 times the number of measurable nodes). So, the excitations selected to detect and locate the proposed faults in this circuit are sufficient. Observing Table 3.1, it can be seen that only using measures at nodes 1 and 2 is enough to differentiate between all the proposed faults (with 12 measures less than in the previous case). But, if only measures at node 1 are used, there are problems distinguishing between faults Q_{BES} and Q_{EO}. As a conclusion, it seems that the method has good results in catastrophic fault isolation. But, this is not the same for parametric deviations. If there are frequency dependent components in the circuit, a deviation in their values of a certain percentage will not be detectable. For example, a capacitor is an open circuit in DC introducing either a $1V$ or 100 mV DC signal, so the conclusion will be the same if the capacitor has its nominal value or 20% more.

3.3 AC Domain

This domain can be divided into temporal time and frequency analysis. The first one is based on detecting characteristics from transient circuit responses and the second on detecting the characteristics of the response to sinusoidal inputs. Let's see some representative examples of both domains.

3.3.1 Time Methods

The saturated-ramp waveform (Balivada *et al.*, 1996) and the complementary signal proposed by (Capitain, 1982), (Schreiber, 1977) and (Corsi *et al.*, 1993) can be mentioned as examples of time domain analysis. Next, these methods are described and applied to a particular circuit as an example.

The saturated-ramp waveform testing

As proposed in (Balivada *et al.*, 1996), a saturated ramp signal is applied to the circuit input. This method is based on the multifrequency signal contained in a ramp signal. It is used instead of a sinusoidal sweep in the input. The ramp has a rise time t_{rin} and a saturation value V_{SAT} as is shown in Figure 3.3.

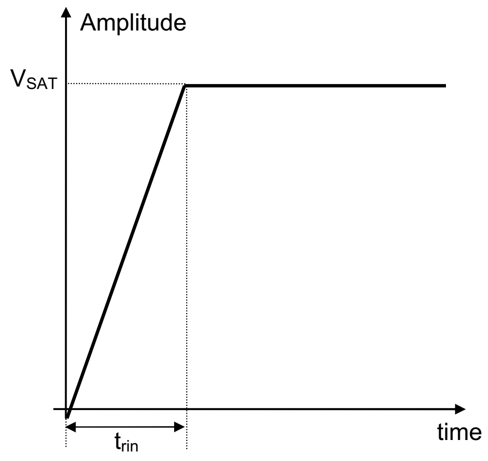


Figure 3.3: Ramp stimulus

A typical circuit response to this input is depicted in Figure 3.4.

The signature that characterizes each fault is compounded by the following parameters:

- *Steady state (V_{est})*: Final value to which the output tends.

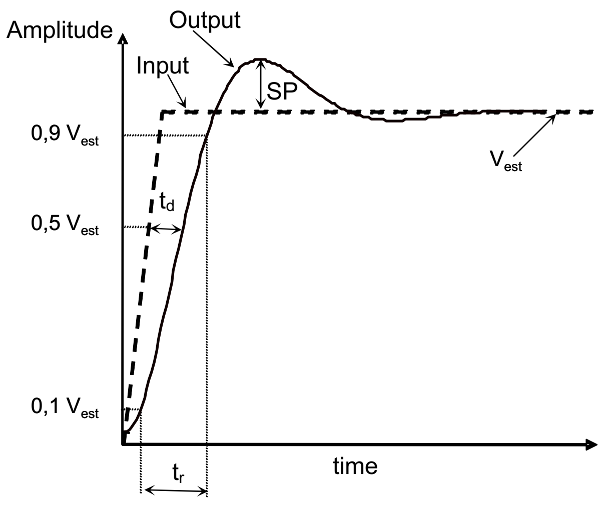


Figure 3.4: Circuit response to a ramp input

- *Overshoot (SP)*: Defined as

$$SP = \frac{V_{\max} - V_{est}}{V_{est}} 100 \tag{3.2}$$

  where V_{max} is the maximum value of the amplitude reached at the output, and V_{est} is the steady state value.

- *Rising time (t_r)*: Time used by the output to rise from the 10% to 90% of the steady state value.

- *Delay time (t_d)*: Interval of time between the moment input and output gets to the 50% steady state value.

The dictionary is built by simulating the circuit, taking the saturated ramp as the input signal and provoking the faults to be detected. Then, the circuit ramp response parameters corresponding to each fault considered are stored in a table. When testing, the faulty circuit is excited with the same ramp input, and the response parameters obtained are compared with the pre-stored ones. The difficulty associated with this method is to choose an appropriate input ramp rise time t_r that produces a satisfactory difference between the values obtained for each fault. Making this choice is not trivial.

As stated in (Balivada *et al.*, 1996) a faster rise time t_{rin} does not necessary imply better results. To illustrate an application of this technique, an example is shown.

Example 9

Let's take the circuit in Figure 3.5 with $R_1 = R_2 = 100\Omega$, $L = \sqrt{2} \cdot 10^2 H$ and

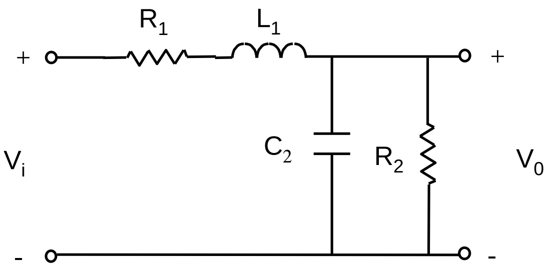


Figure 3.5: Circuit RLC

$C = \sqrt{2} \cdot 10^{-2} F$. The transfer function of this circuit is given by

$$H(s) = \frac{R_2}{L_1 R_2 C_2 s^2 + (L_1 + R_1 \cdot R_2 \cdot C_2)s + (R_1 + R_2)} \tag{3.3}$$

If the saturated ramp is taken with $t_{rin} = 1\ seg$ and $V_{SAT} = 1\ V$, the circuit response of Figure 3.6 is obtained

So, for the nominal case, the following values are obtained

$$V_{max} = 0.517V$$
$$V_{est} = 0.5\ V$$
$$SP = \frac{V_{\max} - V_{est}}{V_{est}} 100 = 4.13\%$$
$$t_r = t_{90\%} - t_{10\%} = 1.45\ sec.$$
$$t_d = 2.24\ sec.$$

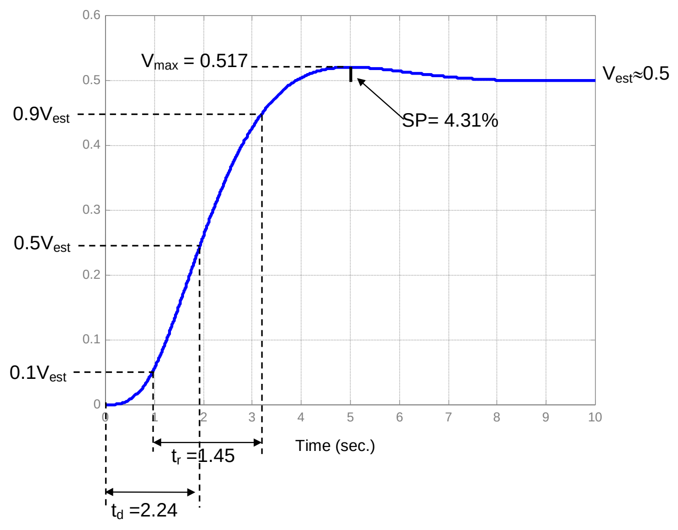


Figure 3.6: Circuit RLC's response to the saturated ramp

If deviations of ±20% and ±50% from the nominal are considered for each component as the possible universe of faults, the dictionary in Table 3.2 is obtained.

As it can be seen there will be several *ambiguity groups*, since there are some measures that are the same. For instance, if for a certain faulty circuit, the parameters measured are

$$
\begin{aligned}
&V_{est} = 0.5\ V\\
&SP = 3.58\%\\
&t_r = 1.78\ sec.\\
&t_d = 2.77\ sec.
\end{aligned}
$$

According to Table 3.6, the closest cases (with a distance of 0) are C_2+50% and $L_1 + 50\%$. Hence they will be taken as the possible failures.

	SP(%)	t_d(sec.)	t_r (sec.)	V_{est}
OK	4.31	1.45	2.24	0.50
R1+20	3.05	1.41	2.25	0.45
R1-20	5.93	1.49	2.23	0.55
R1+50	1.67	1.36	2.28	0.40
R1-50	9.30	1.58	2.23	0.54
R2+20	5.63	1.49	2.23	0.54
R2-20	2.78	1.40	2.26	0.44
R2+50	7.26	1.53	2.23	0.60
R2-50	0.41	1.30	2.35	0.33
L1+20	4.23	1.59	2.44	0.50
L1-20	4.02	1.30	2.04	0.50
L1+50	3.58	1.78	2.77	0.50
L1-50	2.61	1.06	1.77	0.50
C2+20	4.23	1.59	2.44	0.50
C2-20	4.02	1.30	2.04	0.50
C2+50	3.58	1.78	2.77	0.50
C2-50	2.61	1.06	1.77	0.50

Table 3.2: Dictionary for the considered faults and the RLC circuit

Complementary signal

This method consists in building a piecewise constant signal, so that the circuit response is first different from 0 and then decays to it in a finite interval of time t_{dec}. This time interval defines the fault. Complementary signal steps amplitudes are functions of the location of the circuit's poles and, as a consequence, functions of the circuit components. A typical complementary signal could be the one showed in Figure 3.7.

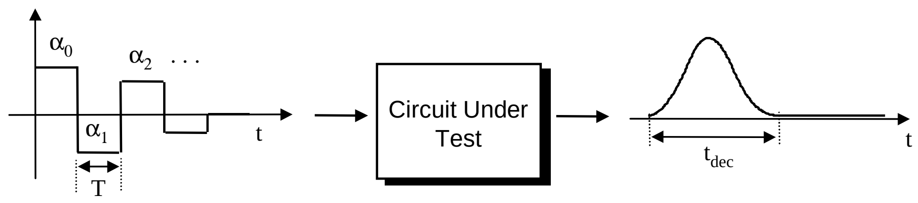


Figure 3.7: Complementary signal concept

The complementary signal associated with a fault is derived using the faulty and nominal circuit response to a pulse $u(t)$ given by equation 3.4.

$$u(t) = \begin{cases} 1 & 0 \leq t \leq T \\ 0 & elsewhere \end{cases} \tag{3.4}$$

According to (Capitain, 1982) and (Schreiber, 1977), the parameters α_i are calculated using the following expressions:

$$\begin{aligned} &\alpha_0 = 1 \\ &\alpha_1 = +\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} e^{(v_i+v_j)T} \\ &\vdots \\ &\alpha_n = (-1)^n \cdot e^{\sum_{i=1}^{n}(v_i T)} \end{aligned} \tag{3.5}$$

where V$_i$ are the poles of the system and T the duration of each α_i.

Then, if the good and faulty circuit complementary signals are α_i and $\hat{\alpha_i}$ respectively, the fault signature is $[\alpha_i - \hat{\alpha_i}]$ for $i = 1, 2, ..., n$, with n the number of faults to detect. The dictionary is constituted by a vector of signatures Q= [α_1 - $\hat{\alpha}_1$, α_2 - $\hat{\alpha}_2$,....]. This technique has the disadvantage that the transfer function of the circuit has to be known or determined empirically.

On the other hand, according to (Bandler and Salama, 1985), the complementary signal can be written as

$$\sum_{i=0}^{n_p} \alpha_i \cdot u(t - iT) \tag{3.6}$$

where n$_p$ is the system order. This response can be transformed as a function of the circuit response v$_u$(t) to the impulse of equation 3.4 as

$$\sum_{i=0}^{n_p} \alpha_i \cdot v_u(t - iT) \tag{3.7}$$

Then, the parameters α_i have to be selected making the response of the system to the complementary signal vanish before $t = (n_p + 1)T$. To calculate the parameters the following system has to be solved

$$\begin{pmatrix} -v_u((n_p+1)T) \\ -v_u((n_p+2)T) \\ \vdots \\ -v_u((n_p+q)T) \end{pmatrix} = \begin{pmatrix} v_u(n_pT) & v_u((n_p-1)T) & \cdots & v_u(T) \\ v_u((n_p+1)T) & v_u(n_pT) & \cdots & v_u(2T) \\ \vdots & \vdots & & \vdots \\ v_u((n_p+q-1)T) & v_u((n_p+q-2)T) & & v_u(qT) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_q \end{pmatrix} \tag{3.8}$$

where q is the number of $v_u(t)$ samples taken after $t = (n_p + 1)T$.

If $q \geq (n_p + 1)$, a system of overdetermined equations in α is built. The advantage of this method for finding the α_i parameters is that the circuit transfer function is not necessary.

The difficulty of the complementary signal method is to find the period T that gives the best results. In (Bandler and Salama, 1985), (Capitain, 1982) and (Schreiber, 1977) this T is chosen arbitrarily. In (Corsi *et al.*, 1993) an approximation value of T is used, which allows the maximum sensitivity to fault conditions to be obtained. This value has to do with the inverse of the circuit bandwidth. To see how the method works, an example is shown.

Example 10

Suppose that the circuit to be analyzed is the RLC circuit shown in Figure 3.5 with the same component values. The circuit has the transfer function given in equation 3.3. For the given values, the poles of the system are situated at -0.5050±0.9192i. The stimulus impulse is chosen arbitrarily to be 1 V amplitude and $T = 0.5$ *seconds* width.

The circuit response to this impulse is depicted in Figure 3.8

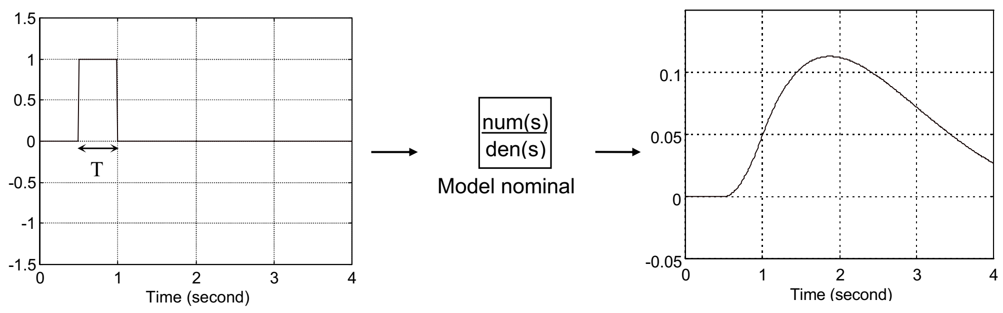


Figure 3.8: RLC impulse response

Choosing the method proposed by (Bandler and Salama, 1985), the following values from the circuit impulse response are taken:

$$V_u(0) = 0, \quad V_u(0.5) = 0.0491, \quad V_u(1) = 0.1034, \quad V_u(1.5) = 0.112,$$

$$V_u(2) = 0.0966, \quad V_u(2.5) = 0.0720, \quad V_u(3) = 0.0473, \; ...$$

and using the set of equations 3.8, with $q = 2$, since it is a second order circuit, the following equation system can be built:

$$\begin{pmatrix} -V_u(1.5) \\ -V_u(2) \end{pmatrix} = \begin{pmatrix} V_u(1) & V_u(0.5) \\ V_u(1.5) & V_u(1) \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix}$$

$$\downarrow$$

$$\begin{pmatrix} -0.112 \\ -0.0966 \end{pmatrix} = \begin{pmatrix} 0.1034 & 0.0491 \\ 0.1120 & 0.1034 \end{pmatrix} \begin{pmatrix} \alpha_1 \\ \alpha_2 \end{pmatrix}$$

Solving the system, the parameters α_i obtained are

$$\alpha_1 = -1.3138, \qquad \alpha_2 = 0.493$$

and α_0 is always set to 1. Something similar will be done with the other faults considered.

Once the dictionary is built, the process for testing circuits consists in applying the pulse of duration T to the faulty circuit. Then, samples of the circuit response at instances $t = T,\ 2T,\ 3T,\ ...,\ (np+1)T$ are taken. These values are used to solve the system of equations Eq. 3.8 obtaining the parameters α_i. Finally, these parameters are checked with the parameters stored in the dictionary. The case of the dictionary with the α_j parameters that are most similar to the recently calculated α_i is taken as the solution of the test.

The method presents good results for simulated circuits with small deviations from the considered faults. But, once tested with real data, the results show that the location of faults is extremely sensitive to noise. Also, the components tolerance makes the diagnosis system go wrong. Hence, the complementary signal method does not seem to be very useful when applied to a real circuit. Therefore, the saturated-ramp is the time method selected for our purposes.

3.3.2 Frequency Methods

The stimuli used are sinusoidal waveforms at different frequencies. Then, measures of the gain, phase or both at these frequencies make up the signatures that will be stored in the dictionary. The same stimuli is applied to the faulty circuit, and the signatures obtained are compared with the stored ones. The most similar one will be taken as the solution.

So, the first point is to choose the adequate test frequencies. It is desirable that the number of frequencies is the minimum and the degree of testability the maximum. There are several procedures that can help in the frequency selection process. Three of them are commented on.

- The Seshu-Waxman method (Bandler and Salama, 1985): Suppose a linear time invariant system with the transfer function

$$H(s) = k\frac{\prod_{i=1}^{n_z}(s - z_i)}{\prod_{j=1}^{n_p}(s - p_j)} \tag{3.9}$$

where s is the complex frequency, and z_i and p_j are the zeros and poles of the transfer function respectively. As the parameter values shift is related to the poles' and zeros' (breakpoints) complex s-plane positions, the test frequencies are selected so that there must be at least one test frequency below the lowest nonzero break frequency, one above the highest finite break frequency and one between successive breakpoints. A change in a particular parameter will produce a change in the module and phase of the circuit response of the breakpoints and consequently in the test frequency set selected. This is because the variation in the circuit parameters produces variation in the pole and zero values and as a consequence a change in the magnitude of the transfer function is produced. The selection of test frequencies between breakpoints allows different measurement values produced by the parameters variation to be obtained.

- Selection by optimization: This method is proposed and developed in (Varghese *et al.*, 1978). It associates a confidence level index with a set of selected test frequencies. This index has to do with the capability of the frequencies to separate and diagnose the proposed set of faults. First, a wide frequency test set is defined. The faults of interest are simulated and the magnitude and phase responses at these frequencies are stored. Then, the algorithm starts calculating the confidence level beginning with a frequency and a measure (module or phase). If the index is not satisfactory (less than a pre-established minimum), more responses and frequencies are introduced. When the confidence level is sufficient, the algorithm stops and provides us with an optimum set of frequencies and measures to perform.

- Sensitivity based analysis: In order to choose test frequencies that have enough measure variations when a fault is produced, an analysis of the circuit's sensitivity to its components' variations is performed. In (Duhamel and Rault, 1979), (Bandler and Salama, 1985) and (Jurisic *et al.*, 1996) some techniques using this frequency selection procedure are shown. The sensitivity concept used in them is the differential sensitivity, defined as

$$S_x^T = \frac{x}{T}\frac{\partial T}{\partial x} = \left.\frac{\Delta T/T}{\Delta x/x}\right|_{\Delta x \to 0} \tag{3.10}$$

where x is the component and T the circuit transfer function. The disadvantage of this definition is that it only allows for soft component deviations from their nominal value. For large deviations, the frequencies at which the sensitivity is at its maximum correspond to other different values. This can be seen in (Slamani and Kaminska, 1995) and an improvement is proposed, defining the concept of incremental sensitivity.

- Selecting frequencies where the maximum error is achieved: As stated in (Balivada *et al.*, 1996), another way of building a set of test frequencies is to obtain the set at which the maximum deviations of the module and phase circuit responses have the maximum difference from the nominal for the considered faults.

Example 11

Let's take the circuit given in (Varghese *et al.*, 1978) and reproduced in Figure 3.9.

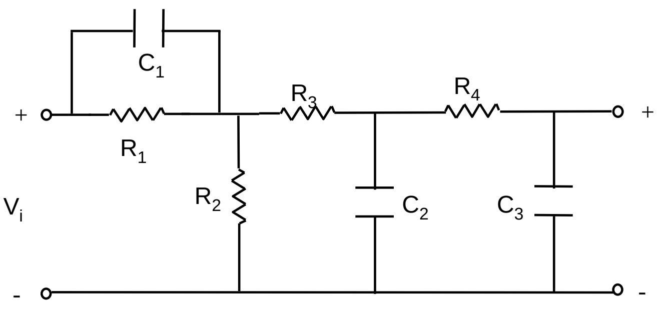


Figure 3.9: R-C network for the selection of frequencies

The circuit transfer function is given by:

$$H(s) = \frac{V_0}{V_i} = \frac{10^5 s + 10^7}{0.2s^3 + 532s^2 + 17.5 \cdot 10^4 + 1.1 \cdot 10^7} \tag{3.11}$$

with $R_1 = 1M\Omega$, $R_2 = 10M\Omega$, $R_3 = 2M\Omega$, $R_4 = 1M\Omega$, $C_1 = 0.01\mu F$, $C_2 = 0.001\mu F$ and $C_3 = 0.001\mu F$. This circuit has a zero in $\omega_z = 100$ rad/sec and poles in $\omega_1 = 83.3$ rad/sec, $\omega_2 = 288.6$ rad/sec and $\omega_3 = 2288.1$ rad/sec. Then, according to the Seshu-Waxman criterion $\omega = 10,\ 95,\ 200,\ 800$ and 5000 rad/sec, could be taken as the set of test pulsations.

If the frequency selection proposed in (Balivada *et al.*, 1996) is applied, the frequencies selected are the ones where the maximum module or phase difference from the nominal are achieved for each considered fault. For example, for the circuit in Figure 3.9, a pulsations sweep from 10^{-1} to 10^5 is performed. For the fault $R_1 + 20\%$, the differences in magnitude and the phase from the nominal, are represented in Figure 3.10 and Figure 3.11.

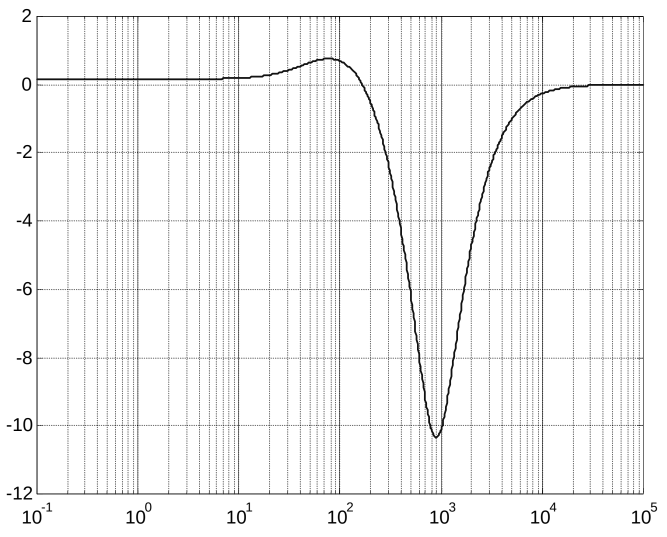


Figure 3.10: Sensitivity magnitude for $R_1 + 20\%$

By doing the same for the set of proposed faults to detect the *nominal* $\pm$ 20% and the *nominal* $\pm$ 50%, Table 3.3 can be built.

In the case that the frequencies are selected depending on the components sensitivity, first the sensitivities according to equation 3.10 must be calculated. For the RC circuit in Figure 3.9, the sensitivity magnitude for each component is obtained and depicted in Figure 3.12.

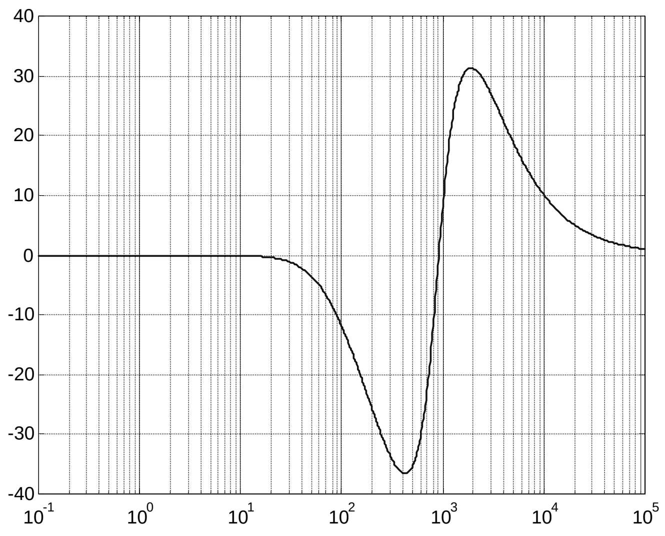


Figure 3.11: Sensitivity phase for $R_1 + 20\%$

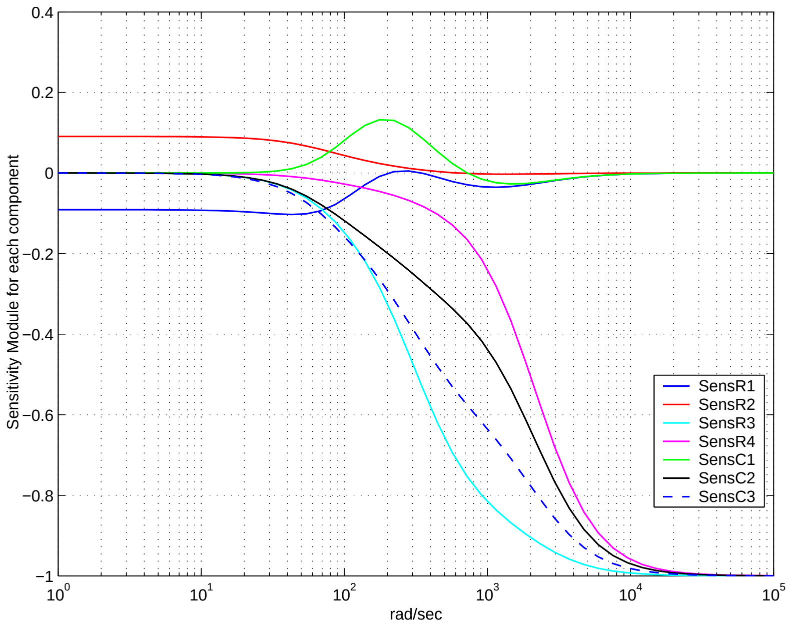


Figure 3.12: Sensitivity magnitude

Fault	Δmag (dB)	ω (rad/sec)	ΔPhase (°)	ω (rad/sec)
R1+20	-10.35	895.26	-36.55	418.43
R1-20	-10.14	946.18	-36.33	460.96
R1+50	-10.42	870.84	-36.52	407.01
R1-50	-9.70	1028.00	-35.30	529.30
R2+20	-10.28	907.73	-36.30	436.15
R2-20	-10.25	920.37	-36.80	436.15
R2+50	-10.29	907.73	-36.10	436.15
R2-50	-10.20	933.19	-37.67	442.22
R3+20	-9.70	841.10	34.80	1763.00
R3-20	-10.90	1000.00	-38.50	461.00
R3+50	-8.90	790.50	39.10	1600.30
R3-50	-12.00	1265.00	-42.20	522.10
R4+20	-9.00	835.50	33.30	1787.50
R4-20	-11.70	1028.00	-41.90	494.00
R4+50	-7.50	737.70	35.80	1622.60
R4-50	-14.70	1318.60	-52.40	624.90
C1+20	-10.84	882.97	-37.22	424.25
C1-20	-9.47	959.36	-35.26	460.96
C1+50	-11.45	847.08	-37.81	407.01
C1-50	-7.50	1101.60	-31.20	529.30
C2+20	-9.80	847.10	35.20	1763.00
C2-20	-10.80	1013.90	-38.50	467.40
C2+50	-9.20	768.90	40.00	1578.30
C2-50	-11.70	1300.50	-42.10	529.30
C3+20	-8.40	847.10	31.40	1837.70
C3-20	-12.50	1000.00	-43.20	494.00
C3+50	-6.10	768.90	31.50	1763.00
C3-50	-17.20	1230.50	-56.80	642.40

Table 3.3: Maximum difference from the nominal

In this case, for components R_3, R_4, C_2 and C_3, the highest sensitivity is obtained for high frequencies, having a value of -1. Hence taking $\omega = 20Krad/sec$ for an example, could be a good choice for diagnosing these components. If the same frequency is used for the rest of the components, the results for them are not distinguishable from their nominal situation, because the sensitivity at high frequencies is 0 (no variation on the measures). R_1 exhibits its highest sensitivity at low frequencies, as does R_2. Therefore a frequency $\omega = 10\ rad/sec$ can be useful for detecting faults related to R_1 and R_2. On the other hand, C_1 presents maximum sensitivity of the transfer function module close to $\omega = 200\ rad/sec$.

If the algorithm proposed in (Varghese *et al.*, 1978) is applied, the following procedure has to be followed:

- The power discrimination of the measure i is determined by

$$D_i = \left[\sum_{f=1}^{n_f - 1} (d_{if} - d_{i,f+1})^2 \right]^{1/2} \quad \forall\ measure\ i \tag{3.12}$$

  where d_{if} are the measures done for the n_f faulty cases considered. Equation 3.12 is useful in order to discriminate between the n_f faults, since a small D_i value means that the fault symptoms are very close, making the distinction between them difficult.

- A separability index between two faults f_1 and f_2 is defined as

$$D_{f_1 f_2} = \left[\sum_{i=1}^{n_m} (d_{if1} - d_{if2})^2 \right]^{1/2} \tag{3.13}$$

  Usually, this expression is given in % as

$$D^*_{f_1 f_2} = \frac{D_{f_1 f_2}}{D^*} \tag{3.14}$$

  where D^* is $D^* = max(D_{f_1 f_2})$

- With these values, the *Confidence level* is calculated as follows

$$CONF = \frac{CONF1 + CONF2}{n_f(n_f - 1)/2} \tag{3.15}$$

where CONF1 is the number of measures that have a separability greater than 50%, and CONF2 is the separability sum of the measures with a separability less than 50%. So, this set of measures is characterized by a confidence level $CONF$. If this index is not satisfactory, the set of measures proposed will be optimized using the algorithm given in (Varghese *et al.*, 1978). As it can be expected, from a certain number of measures taken, the confidence level does not increase significantly.

Applying the algorithm to the R-C network of Figure 3.9, the following table is obtained (extracted from (Varghese *et al.*, 1978)).

Set	N. Meas.	Amplitude	Phase	Conf. Level %
1	1	-	100	50
2	2	10	100	68.25
3	3	10	100 600	85.55
4	4	10	100 600 10000	87.31
5	4	10	100 600 9000	87.55
6	4	10	100 600 8000	87.84
7	4	10	100 600 7000	87.24
8	6	10 80 1000	100 600 6000	87.95
9	7	10 80 500 900	100 600 6000	88.07
10	9	10 80 200 400 700	100 600 5000 9000	89.33
11	9	10 70 200 400 700	100 600 5000 9000	89.27
12	9	10 70 200 400 700	100 600 4000 8000	89.52
13	10	10 70 200 400 700	100 200 600 4000 7000	89.03
14	11	10 70 200 400 600 1000	100 200 600 4000 7000	88.95
15	12	10 60 200 400 600 1000	100 200 600 3000 6000 9000	89.22

Table 3.4: Set of measures versus confidence level

A graphical representation of the Table 3.4 is given in Figure 3.13.

In Figure 3.13, taking more than four measurements does not cause a great improvement. For example, set number 6 needs 4 measurements, and the associated confidence level is 87.84%, while taking the set number 15, 12 measurements are necessary and the associated confidence level is 89.22%. So, in order to improve the confidence level in less than 2%, 8 measures have to be taken.

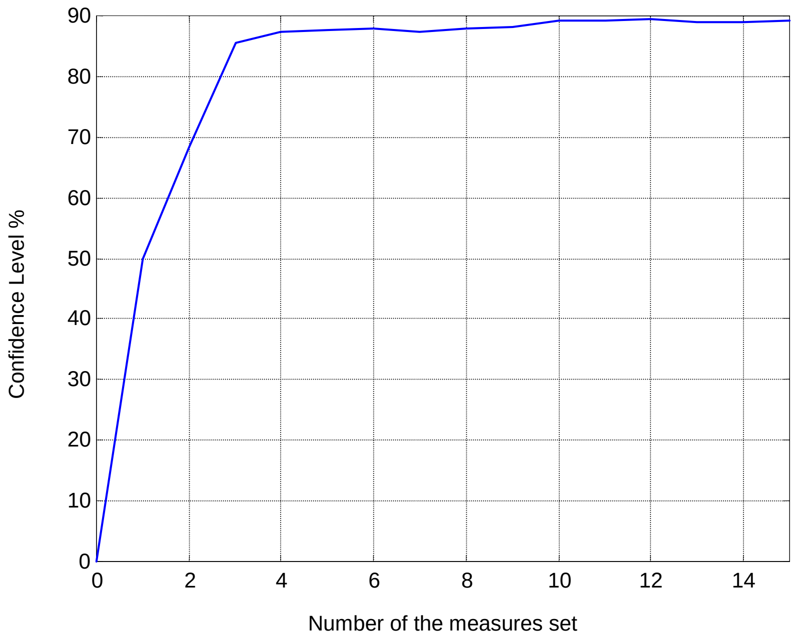


Figure 3.13: Confidence level versus set number

3.4 Numerical Results of the Biquad Filter

Since the biquadratic filter described in Chapter 1 will be used as a benchmark for demonstrating the proposed methodology, some fault dictionary methods will be applied to it. Two AC fault dictionaries are analyzed in detail: one based on the the saturated ramp response, and the other based on the frequency domain. The universe of faults to be detected are deviations of $\pm 20\%$ and $\pm 50\%$ from the nominal for each component.

3.4.1 Saturated Ramp

If a saturated ramp input has values $t_r = 100\ \mu s$ and $V_{SAT} = 1V$, the circuit output V_0 is shown in Figure 3.14.

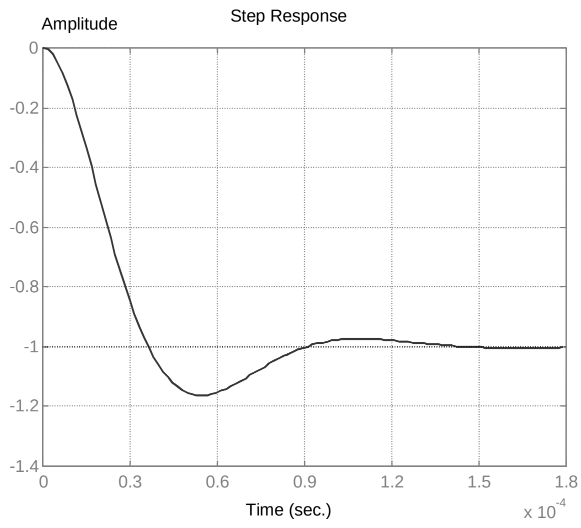


Figure 3.14: Biquad filter saturated ramp response

The response parameters for the nominal case are

$$[SP,\ t_d,\ t_r,\ V_est] = [4.4029\%,\ 15\ \mu s,\ 76\mu s,\ -1\ V]$$

Taking into account the universe of faults considered, the dictionary shown in Table 3.5 is obtained. The faults for R_2, R_3 and C_1 are grouped together because they produce exactly the same measures. They constitute an ambiguity group.

The last separated column of Table 3.5 gives results when a set of 100 new Monte-Carlo randomly generated cases for each considered fault is taken for testing. Observe that in spite of tolerances, the performance is quite good. The table also takes ambiguity groups into account.

Fault	SP (%)	T_d (μs)	T_r (μs)	V_{est} (V)	Success %
R1+20	4.4029	15	76	-0.8332	84
R1-20	4.4029	15	76	-1.2498	87
R1+50	4.4029	15	76	-0.6665	99
R1-50	4.4029	15	76	-1.9996	100
R2,R3,C1+20	4.0473	19	77	-1.0001	41
R2,R3,C1-20	4.6614	11	75	-1.0000	36
R2,R3,C1+50	3.3711	24	80	-1.0003	79
R2,R3,C1-50	5.2359	5	75	-0.9999	94
R4+20	5.7311	12	73	-0.9994	85
R4-20	2.4917	19	80	-1.0000	88
R4+50	6.9145	10	71	-0.9990	98
R4-50	0	29	92	-1.0000	100
R5+20	4.6189	12	75	-0.9999	47
R5-20	3.9447	20	77	-1.0002	38
R5+50	4.8682	9	76	-1.0002	82
R5-50	2.1315	31	86	-0.9996	91
R6+20	4.0473	19	77	-1.2001	79
R6-20	4.6614	11	75	-0.8000	86
R6+50	3.3711	24	80	-1.5004	98
R6-50	5.2359	5	75	-0.4999	100
C2+20	5.7085	16	73	-0.9997	82
C2-20	3.0781	15	77	-1.0000	89
C2+50	7.6834	17	72	-1.0005	99
C2-50	1.0031	15	80	-1.0000	100
Nominal	4.4029	15	76	-1.0000	69
				Average	**82.04**

Table 3.5: V_0 parameters for the saturated ramp input

3.4.2 Frequency Method

For the frequency analysis, first of all, a set of the adequate frequencies at which to take the measures has to be selected. If the Seshu-Waxman criterion is applied, the transfer function of the system is needed. The transfer function of the biquadratic circuit taking node V_0 as the only measurable node is given by

$$H_1(s)=\frac{V_0}{V_i}=\frac{-R_4R_5R_6}{R_1}\cdot\frac{1}{C_1C_2R_4R_2R_3R_6s^2+C_1R_2R_3R_6s+R_4R_5}$$

or normalizing as

$$H_1(s)=\frac{V_0}{V_{in}}=\frac{-R_5}{R_1R_2R_3C_1C_2}\frac{1}{s^2+\frac{1}{R_4C_2}s+\frac{R_5}{R_2R_3R_6C_1C_2}}$$

Giving the corresponding values described in Chapter 1

$$H_1(s)=\frac{-4.44410^9}{s^2+6.667\cdot10^4s+4.444\cdot10^9}$$

The transfer function has no zeros and two complex poles at

$$-3.33\cdot10^4\pm5.77\cdot10^4j$$

Then, there is only a breakpoint located at $\omega_0 = 6.663\cdot10^4$ *rad/sec.* According to Seshu-Waxman, one frequency below ω_0 and another above ω_0 should be taken as possible frequencies where measures can be taken.

If frequencies where the maximum difference from the nominal occurs are taken as the measurement set of frequencies, Table 3.6 is obtained for the biquadratic filter.

	ω x 10^4	ΔVolts	ω x 10^4	Δdegrees
R_1+20	0.7600	0.1993	0.3100	0.0079
R_1-20	3.0100	-0.0320	1.2000	0.1059
R_1+50	0.7600	0.3981	0.2500	0.0067
R_1-50	3.0100	-0.1275	1.2300	0.4047
R_2, R_3 C_1+20	1.0900	0.1930	0.8700	11.0061
R_2, R_3 C_1-20	0.2800	0.0015	0.1047	-0.0001
R_2, R_3 C_1+50	1.0400	0.3933	0.8100	23.0208
R_2, R_3 C_1-50	0.4800	0.0181	0.1047	-0.0003
R_4+20	0.0300	0.0001	1.6200	5.4865
R_4-20	0.9500	0.2281	0.6300	6.6562
R_4+50	0.0300	0.0002	1.5200	12.2086
R_4-50	0.9100	0.5732	0.5500	20.1225
R_5+20	0.2500	0.0010	3.0100	0.1495
R_5-20	1.0700	0.2370	0.8500	13.4128
R_5+50	0.3700	0.0050	3.0100	0.3832
R_5-50	0.9700	0.6040	0.7400	36.2301
R_6+20	1.2500	0.0478	0.8700	11.1163
R_6-20	0.5300	0.2200	0.1047	0.0040
R_6+50	1.2000	0.1002	0.8100	23.2092
R_6-50	0.6600	0.5854	0.1047	0.0023
C_2+20	1.3800	0.0826	0.8700	11.0116
C_2-20	0.7600	0.0831	0.1047	-0.0001
C_2+50	1.2900	0.1994	0.8100	23.0393
C_2-50	0.8100	0.2050	0.1047	-0.0003

Table 3.6: Frequencies at which the difference from the nominal is at its maximum

As an initial set the following frequencies ω could be taken:

$$\omega = [\ 5000,\ 5500,\ 6600,\ 7500,\ 8000,\ 8500,\ 9000,\ 9500,$$
$$10000{,}11000{,}12000{,}12500{,}13000{,}14000{,}15200,\ 30000\ ]\ rad/sec$$

These are the number of frequencies that group the major number of faults, without being too particular for one exclusive fault and too close to the others.

On the other hand, if the set of frequencies are selected using the sensitivity criterion, the circuit sensitivity at nodes V_1, V_2 and V_3 are given in Figures 3.15, 3.16 and 3.17

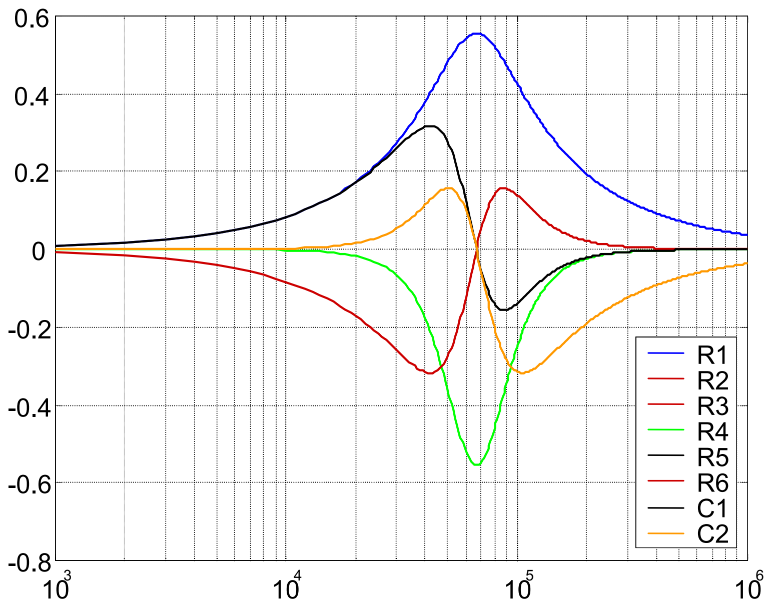


Figure 3.15: Biquad filter module sensitivity at node V_1

The sensitivity phase graphic is the same for the three proposed nodes. This is because the mathematical expression of the phase for these nodes is given by equation 3.16

$$\varphi = \varphi_0 - \arctan\left(\frac{C_1G_4G_5\omega}{G_2G_3G_6 - C_1G_2G_5\omega^2}\right) \tag{3.16}$$

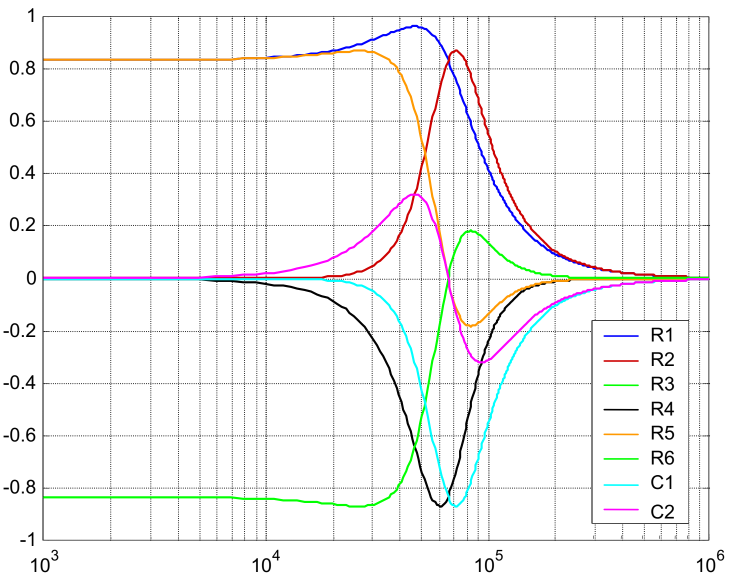


Figure 3.16: Biquad filter module sensitivity at node V_2

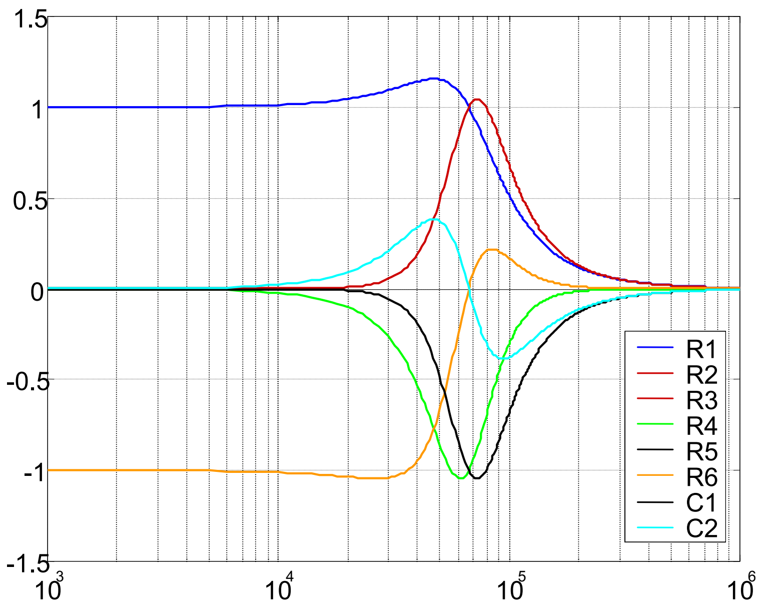


Figure 3.17: Biquad filter module sensitivity at node V_0

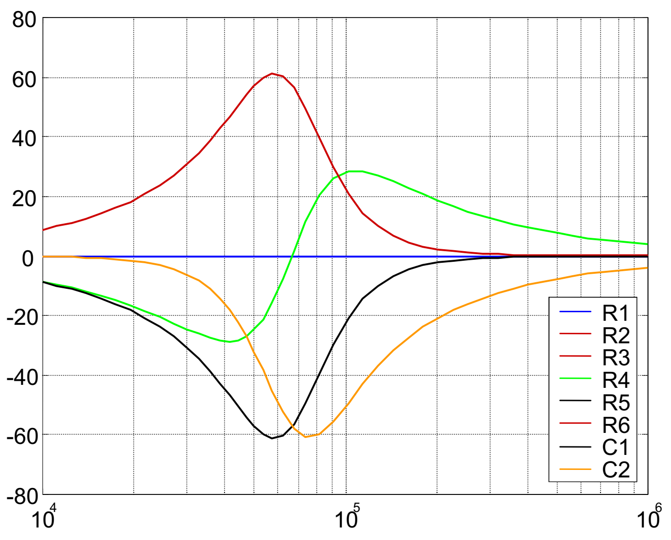


Figure 3.18: Biquad filter phase sensitivity at nodes V_1, V_2 and V_0

The initial angle φ_0 at $\omega = 0\ rad/sec$ is -90^o at node V_1, 0^o at node V_2 and 180^o at node V_3. Hence, when differentiating, the expressions obtained for all three will be the same.

Therefore, if node V_0 is taken as the only measurable node, Figure 3.17 shows that the sensitivity for each component has its absolute maximum value at the frequencies shown in Table 3.7

	R_1	R_2	R_3	R_4	R_5	R_6	C_1	C_2
ω (rad/sec)	$5{\cdot}10^4$	$7{\cdot}10^4$	$7{\cdot}10^4$	$6{\cdot}10^4$	$7{\cdot}10^4$	$3{\cdot}10^4$	$7{\cdot}10^4$	$5{\cdot}10^4$

Table 3.7: Frequencies at which sensitivity is at its maximum

Applying the confidence level method for the optimal frequencies at which to perform the amplitude and/or phase measures, Table 3.8 is obtained, and it is depicted in Figure 3.19.

At the beginning, there is a great increase in the confidence index caused by taking more measures. But, after 6 or 7 measures, the confidence level has a slight improvement while the number of measures to take increases

Meas. Num.	Measures		Conf. Level
	Magnitude (rad/sec.).10^3	Phase. (rad/sec.) .10^3	
1	65		53.84
3	85	9, 65	76.11
4	10, 85	9, 65	80.96
5	10, 85	9, 10, 65	81.13
6	10, 85	9, 10, 65,100	82.28
7	10, 20, 85	9, 10, 65,100	82.95
8	10, 15, 20, 85	9, 10, 65, 95	82.86
9	10, 15, 20, 85	9, 10, 15, 65, 90	82.88
10	10, 15, 20, 25, 85	9, 10, 15, 65, 90	83.40
10	10, 15, 20, 25, 85	9, 10, 15, 65, 85	83
11	10, 15, 20, 25, 85	6, 9, 10, 15, 65, 85	83.09
12	10, 15, 20, 25, 30, 85	6, 9, 10, 15, 65, 85	83.42
13	10, 15, 20, 25, 30, 85	6, 7, 9, 10, 15, 65, 85	83.54
14	10, 15, 20, 25, 30, 85	6, 7, 9, 10, 15, 65, 80,100	83.25
15	10, 15, 20, 25, 30, 85	6, 7, 9, 10, 15, 20, 65, 80,100	83.66
16	10, 15, 20, 25, 30, 85	6, 7, 9, 10, 15, 20, 25, 65, 80,95	83.79
17	10, 15, 20, 25, 30, 35, 85	6, 7, 9, 10, 15, 20, 25, 65, 80,95	84.19
19	10, 15, 20, 25, 30, 35, 50, 85,100	6, 7, 9, 10, 15, 20, 25, 65, 80,95	84.93
20	10, 15, 20, 25, 30, 35, 50, 85,100	6, 7, 8, 9, 10, 15, 20, 25, 65, 80,95	84.92
21	10, 15, 20, 25, 30, 35, 50, 55, 85,100	6, 7, 8, 9, 10, 15, 20, 25, 65, 80,95	84.89
22	10, 15, 20, 25, 30, 35, 50, 55, 60, 85,100	6, 7, 8, 9, 10, 15, 20, 25, 65, 80,95	84.57
23	10, 15, 20, 25, 30, 35, 40, 50, 55, 60, 85,100	6, 7, 8, 9, 10, 15, 20, 25, 65, 80,95	84.31
24	10, 15, 20, 25, 30, 35, 40, 50, 55, 60, 70, 85,100	6, 7, 8, 9, 10, 15, 20, 25, 65, 80,95	84.23

Table 3.8: Confidence indexes for the biquadratic filter

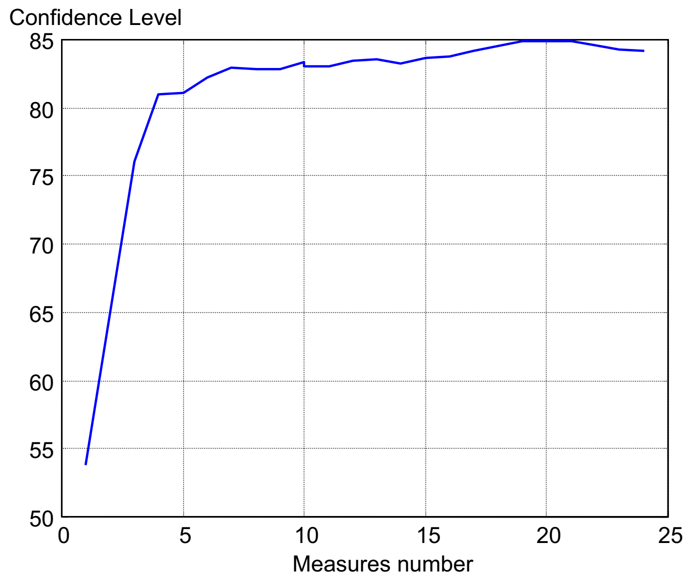


Figure 3.19: Confidence level for the biquadratic filter

considerably. Therefore, selecting the set of 6 measures for the frequency is a good compromise. These measures are reported in Table 3.9

Freq. (rad/sec.)	Amplitude (Volts)	Phase (°)
9000		X
10000	X	X
65000		X
85000	X	
100000		X

Table 3.9: Measures needed to perform the frequency method

Taking these measures for the biquadratic filter, and considering the same universe of faults used for the saturated ramp method, Table 3.10 is built for the frequency dictionary. The last separated column of Table 3.10 shows the test results when a set of 100 new exemplars for each fault generated by the Monte-Carlo algorithm is used. The success average is quite good even in the presence of tolerances.

Fault	Mag $9 \cdot 10^3$	Mag $10 \cdot 10^3$	Phase $10 \cdot 10^3$	Phase $65 \cdot 10^3$	Mag $85 \cdot 10^3$	Phase $100 \cdot 10^3$	Success %
R1+20%	172.17	0.8427	171.28	92.90	0.5868	50.19	84
R1-20%	172.17	1.2640	171.28	92.90	0.8801	50.19	85
R1+50%	172.17	0.6741	171.28	92.90	0.4694	50.19	95
R1-50%	172.17	2.0224	171.28	92.90	1.4082	50.19	100
R2,R3,C1+20%	170.60	1.0106	169.52	83.14	0.5551	46.64	43
R2,R3,C1-20%	173.75	1.0108	173.03	107.07	0.9404	56.31	35
R2,R3,C1+50%	168.24	1.0080	166.89	73.76	0.4179	43.45	80
R2,R3,C1-50%	176.10	1.0085	175.66	137.10	1.5051	80.54	93
R4+20%	173.46	1.0148	172.71	93.48	0.8110	45.00	80
R4-20%	170.25	1.0047	169.14	92.32	0.5841	56.31	85
R4+50%	174.76	1.0177	174.16	94.34	0.9475	38.66	95
R4-50%	164.62	0.9780	162.94	91.45	0.3809	67.38	100
R5+20%	173.48	1.0109	172.74	104.35	0.8927	55.01	42
R5-20%	170.20	1.0103	169.08	81.22	0.5267	45.97	47
R5+50%	174.79	1.0100	174.20	119.40	1.1708	63.43	88
R5-50%	164.35	0.9990	162.56	65.19	0.2940	40.60	95
R6+20%	170.60	1.2127	169.52	83.14	0.6662	46.64	78
R6-20%	173.75	0.8086	173.03	107.07	0.7523	56.31	89
R6+50%	168.24	1.5119	166.89	73.76	0.6268	43.45	96
R6-50%	176.10	0.5042	175.66	137.10	0.7525	80.54	100
C2+20%	172.14	1.0157	171.24	81.79	0.6288	41.42	80
C2-20%	172.20	1.0067	171.32	103.80	0.7634	61.93	91
C2+50%	172.10	1.0227	171.18	66.40	0.5202	32.28	97
C2-50%	172.24	0.9999	171.37	118.29	0.7760	85.24	100
Nominal	172.17	1.0112	171.28	92.90	0.7041	50.19	70
						Average	**81.92**

Table 3.10: Dictionary obtained using the frequency method

3.5 Limitations of Fault Dictionaries

The advantage of these techniques is their simplicity. The problem is that the only faults detected and located will be the ones that have been previously simulated and stored in the dictionary. So, the more faults to locate, the longer the dictionary should be.

These techniques are a compromise between fault coverage and dictionary length. When the fault dictionary is made larger, the scope of faults detected increases as well. On the other hand, if there is only a small set of faults considered when generating the dictionary, the dictionary will be shorter, but the world of detected faults will be reduced as well. So, one of the objectives of the diagnosis designer is to built a dictionary with as few measures as possible that gives a good fault diagnosis coverage.

A lot of systems are seriously affected by tolerances. If the dictionary is obtained by simulating the faults only considering the nominal values of the parameters, the measures obtained from the real system generally will not match the stored ones because of tolerances. In order to find the possible cases produced by the tolerances for a particular measure, several simulation runs have to be carried out. One of the most commonly used methods is the Monte-Carlo. It is obvious that increasing the number of Monte-Carlo runs provokes a dictionary spreading, making it unpractical.

The results obtained by the saturated ramp and the frequency method described above, are quite similar. For the frequency one, the set of 6 measures is selected because it is a good compromise between the number of measures and the confidence level obtained. In Table 3.10 and Table 3.5, there is only a slight difference in the diagnosis results (82.04% and 81.92% of successful) corroborating that the methods are quite equivalent to each other, as stated in (Balivada *et al.*, 1996).

The other important fault dictionary limitation is that they cannot cope with faults that have not been previously considered. For example, if a set of 100 new faults for each component, uniformly distributed between ±70% of the nominal value is taken, the temporal dictionary reduces its success average to 17%, as shown in Table 3.11. A success means that the proposed solution matches the fault with an error less than 10%.

Component	OK diagnosed	Component Correct	Wrong	Non Detect.
R1	19%	57%	16%	8%
R2	20%	25%	42%	13%
R3	5%	44%	37%	14%
R4	15%	65%	17%	3%
R5	15%	24%	50%	11%
R6	20%	62%	12%	6%
C1	12%	42%	35%	11%
C2	30%	52%	7%	11%
Average	**17%**	**46.375%**	**27%**	**9.625%**

Table 3.11: Ramp dictionary successes for deviations of ±70%

So, more cases should be introduced in the dictionary if new situations have to be satisfactorily solved. For example, if a fault such as R3+38% or C2-68% has to be diagnosed, what should we do? The next chapters try to answer to this question.

Chapter 4

IMPROVING FAULT DICTIONARIES USING FUZZY LOGIC

4.1 Introduction

A first solution to the fault dictionary limitations is to refine the diagnosis system by means of fuzzy techniques. These techniques usually perform well in systems with imprecision or uncertainty, such as electronic circuits.

Fuzzy logic has three main decision making steps: fuzzification, inference and defuzzification. The fuzzification unit is the interface between input variables (measures from the circuit in our case) and the inference unit. It has the attributes of acquiring and scaling input variables and their fuzzification. The universe of each considered variable has to be partitioned. This partition is carried out by defining fuzzy sets with a particular membership function shape for each input variable. Fuzzy sets can be built from Monte-Carlo simulations and the dictionary instances can be compacted in fuzzy rules. The system includes tolerance effects and its output is an estimation of each parameter value.

In the inference step, the individually fuzzy sets that result have to be combined. The decision fuzzy set obtained as output is a result of the union of the singular fuzzy sets derived for each particular rule. Once the output fuzzy set is made up, a crisp value belonging to the range of the output variable universe has to be assigned to the obtained fuzzy set obtained.

The goal is to obtain a fuzzy system that is robust to tolerances and to non predicted faults at the same time. The following steps are proposed in order to build the fuzzy logic system:

1. First of all and according to the type of process, **an adequate set of measures** has to be selected (temporal, frequency, continuous, and so on). In our case, only the output process is taken as a measuring point.
2. Secondly, a **dictionary** with a considered universe of faults **without tolerances** is built (the classical dictionary).
3. Then, a **Monte-Carlo simulation** is performed in order to include the tolerance effect. This is done for each considered fault.
4. Afterwards **fuzzy sets corresponding to each input** are built. The number of inputs is equal to the number of measures taken. Each input has as many membership functions as faults considered. The shape of these membership functions will be given by the distributions obtained by the Monte-Carlo simulation.
5. There are as many **fuzzy outputs** as parameters to be identified. Each output is composed by membership functions corresponding to the possible parameter deviations and the nominal case. The shape of the output membership functions depends on the probability distribution function related to the parameter.
6. Finally, **fuzzy rules** are written in the form of IF-THEN sentences.

Let's look at these steps in more detail in the next sections.

4.2 Defining the Inputs

Each measure is considered as a fuzzy system input. For example, for the saturated ramp method, the system will have 4 inputs (SP,t_d, t_r, V_{est}). On the other hand, for a frequency method that uses both, magnitude and phase measures at 10 different frequencies, the fuzzy system will have 20 inputs. The number of membership functions belonging to each input is given by the previously considered universe of faults and the nominal case. That is, if the circuit is composed of N parameters, and deviations of $\pm X\%$ for each parameter are considered as desirable identifiable faults, each fuzzy input will have $2N + 1$ membership functions. Figure 4.1 shows a possible appearance of the measure i input. Each membership function is related to a possible considered fault ($Parameter1 + X\%$, $Parameter1 - X\%$, $Nominal$,...). In general, membership functions are not symmetric.

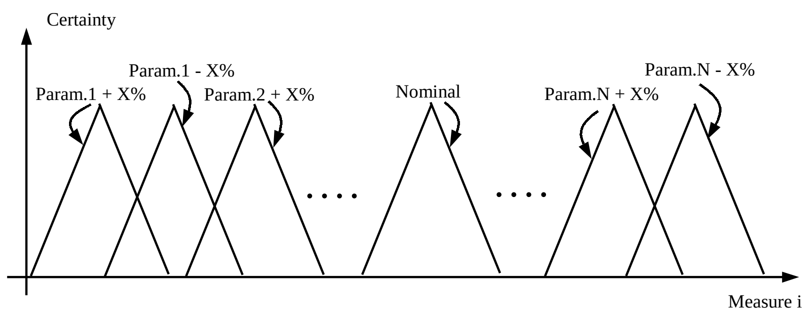


Figure 4.1: Measure i appearance

The membership function shape has been selected taking advantage of the Monte-Carlo simulation results. An analysis of the $Measure\ i$ faults distribution is carried out. For example, $Measure\ i$ for $Fault\ j$ can have the distribution shown in Figure 4.2, after a Monte-Carlo simulation of L runs.

This distribution could be approximated, for example, by a Gaussian or triangular shape as it is depicted in the figure.

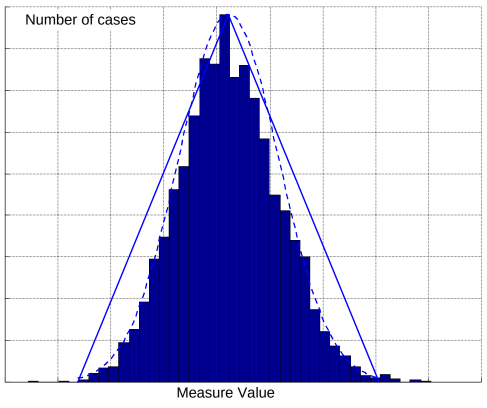


Figure 4.2: Measure i input distribution for fault j

4.3 Defining the Outputs

The outputs of the system are the estimated parameter values. Therefore, there will be as many outputs as parameters. Each output will have a membership function for each considered deviation. For example, if deviations of $\pm X\%$ are considered, the parameter *param i* output will look like Figure 4.3

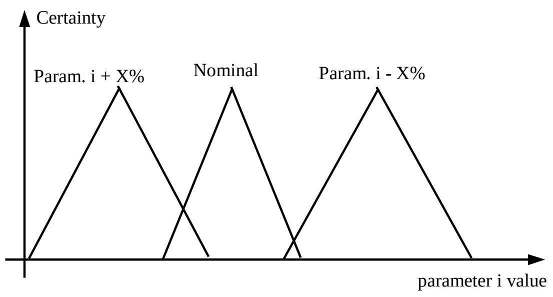


Figure 4.3: Parameter i output appearance

The membership function shape will be taken according to the parameter value distribution. For resistors and capacitors, it is well known that these components have a Gaussian distribution probability function (Loveday, 1995), (Boyd, 1999). Hence a typical output can exhibit a membership distribution as shown in Figure 4.4.

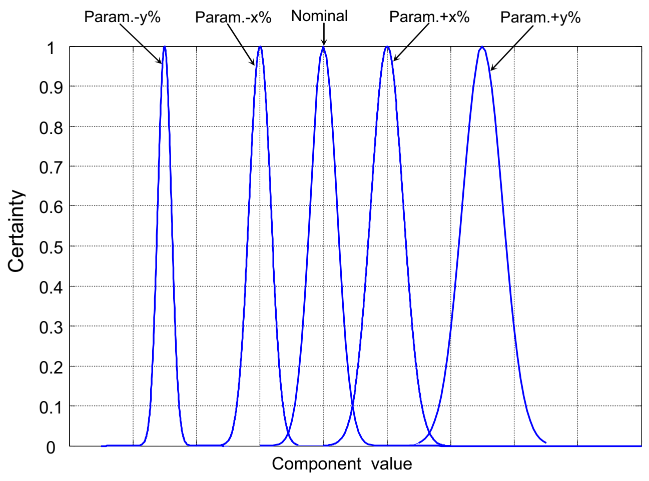


Figure 4.4: Gaussian output membership functions

4.4 Defining the Rules

Once the inputs and outputs are defined, they have to be connected by means of rules. The rule structure for the fault $Param.1 + X\%$ is

if (Meas. 1 is Param. 1+X%)&(Meas. 2 is Param 1+X%)&....(Meas. M is Param. 1+X%) then (Param. 1 is Param. 1+ X%)&(Param. 2 is nominal)&(Param. 3 is nominal)&... (Param. N is nominal).

Hence, there are as many rules as considered faults. The advantage of this method is that it is not necessary to store all the cases, only the rules and membership functions for the inputs and the outputs. The operator selected to combine antecedents is the product. The main reason is to penalize measures falling outside the membership's scope. If one of the M measures falls outside of at least one of the sets defined by the rule antecedents, the final product will be 0, and the rule will not be fired. Otherwise, the rule will be triggered with a value corresponding to the product of the belonging coefficients.

The defuzzification method used is the centroid. Then, after computing a set of measures, each output will provide us with an estimated value of the corresponding parameter with a degree of certainty.

4.5 The Fuzzy Inference Model

Figure 4.5 displays how the inputs, outputs and rules are combined for the ramp method. The example shows that taking the measures SP_1, t_{d1}, t_{r1} and V_{est}, only rules 1 and 3 are activated. The corresponding consequents are derived giving an estimation of the present value for each component.

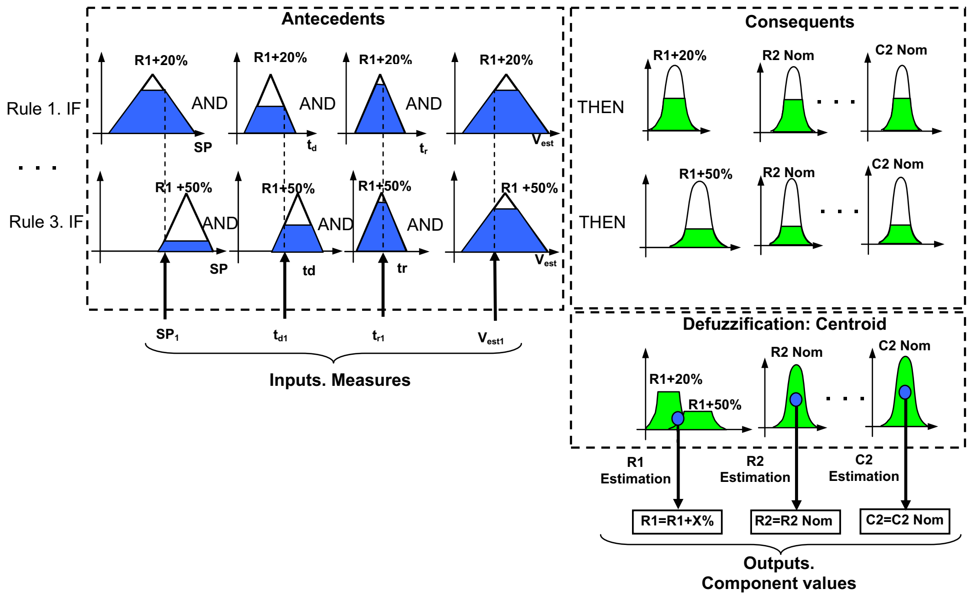


Figure 4.5: Fuzzy inference model

The defuzzification procedure combine the consequents of the activated rules by means of the centroid method. The result is an estimated value for each component.

4.6 Numerical Results

The proposed fuzzy logic system has been implemented in the biquadratic filter from Section 1.7. As the circuit being tested is frequency dependent, the frequency method and the time method described previously can be used. In order to reduce the set of measures, it has been considered that only voltage measures at the output V_0 are possible.

The initial proposed faults to be detected are deviations of $\pm 20\%$ and $\pm 50\%$ from the nominal values of the passive components (a set of 32 faults). So, our fuzzy logic system has a membership function for each fault and the nominal case (33 membership functions per input).

4.6.1 Frequency Method

In the frequency method, measures at each frequency are taken as fuzzy system inputs (6 inputs are defined). Each fuzzy system input corresponds to the amplitude and phase measures displayed in Table 3.9. Figure 4.6 shows the appearance of the input amplitude at 10000 Hz.

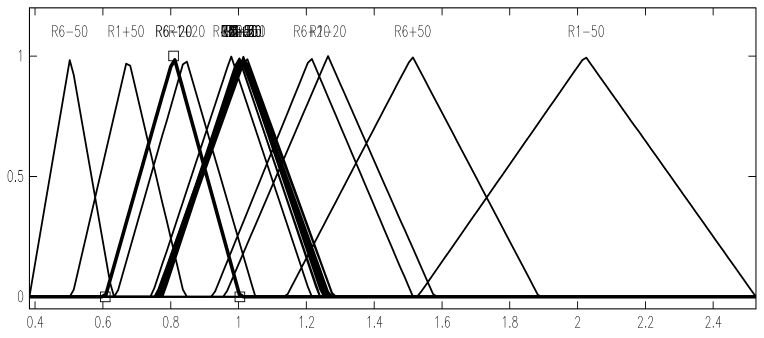


Figure 4.6: The amplitude measure at 10000Hz.

Membership functions are triangular shaped. The maximum value corresponds to the value obtained for this measure and fault, while all the other components stay at their nominal value. Figure 4.7 shows an example corresponding to the measure magnitude at 10000 Hz for the nominal case, where triangular and Gaussian membership functions with the real distribution of the output parameter (histogram) are compared. The abscise magnitude is in Volts. This distribution was obtained after 500 Monte-Carlo runs with the component values being Gaussian distributed in their tolerance margin (nominal case).

The triangle extremes are the maximum deviation of this value produced

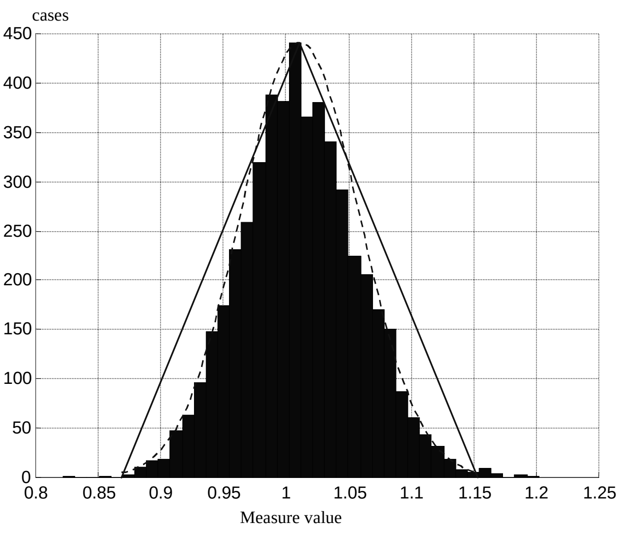


Figure 4.7: Magnitude at 10000. Nominal case

by the component tolerance. For example, for the fault $R_1 - 50\%$, after 500 Monte-Carlo runs, the interval extremes corresponding to the magnitude measurement at 10000 Hz are [1.5278 2.5260], and the value obtained without tolerances is 2.022. Figure 4.7 shows that a triangular shaped approximation for the *membership functions* is not bad. Taking Gaussian distributions does not improve the results significantly, as test results show. In a few cases, the triangular shape slightly improves the diagnostic. The reason is because in some cases the queues of the Gaussian shape introduce overlapping, while triangular membership functions are exactly 0 outside the margin.

There is one fuzzy system output associated with each component (8 for the biquadratic filter). There is a set of *membership functions* corresponding to the component at its nominal value and one for each possible deviation considered, $\pm 20\%$ and $\pm 50\%$ from the nominal value, giving 5 subsets for each output attribute. So, each output attribute provides an estimated value for each component. Figure 4.8 shows the sets considered for the output attribute related to component R_1.

Membership functions belonging to the output attributes have a Gaussian shape, which is the typical distribution for electronic components (Loveday, 1995), (Boyd, 1999). Rules are simple relations, such as the following one:

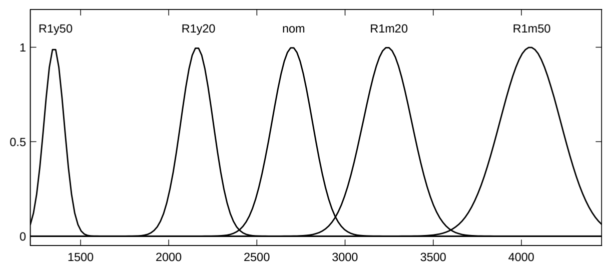


Figure 4.8: Output attribute sets for R_1

if (phase9000 is R1-20)&(mag10000 is R1-20)&(phase10000 is R1-20)&(phase65000 is R1-20)&(phase100000 is R1-20) then (R1 is R1-20)&(R2 is nominal)&(R3 is nominal)&... (C2 is nominal).

The previous rule corresponds to the case $R_1 - 20\%$. The operator '&' is defined by the product function. When the measurements in Table 3.9 are acquired, they belong to a set in a certain degree. Hence, if one of the 6 measures falls outside at least one of the sets defined for this rule, the final product will be 0, and the rule will not be fired. On the other hand, if each of the 6 measures taken fall into the sets defined by the case $R_1 - 20\%$, this rule will be activated with a value corresponding to the belonging coefficients product. Figure 4.9 shows the activated sets in gray for the case $R_1 - 20\%$.

Analyzing Figure 4.9, it can be seen that the rule corresponding to $R_1 - 20\%$ is fired with a belonging coefficient of 1 per set. Rules corresponding to $R_4 - 20\%$, $R_5 - 20\%$ and $R_6 - 20\%$ are fired as well but with a lower index. If $R_1 - 20\%$ was the only fired rule, the diagnostic would be $R_1 - 20\%$ and the other components would have a value belonging to the 10 % tolerance range from the nominal. But, due to the overlapping in the sets, sometimes several rules are fired at the same time for the same set of measures. In this case, the product is the operator selected to combine the fired membership functions. Now, the results of the rules are added providing the output set shown in Figure 4.10 for R_1.

A similar figure is obtained for each component. The centroid method is used for giving a final estimated value for each component. The final outputs for the case $R_1 - 20\%$ are given in Table 4.1.

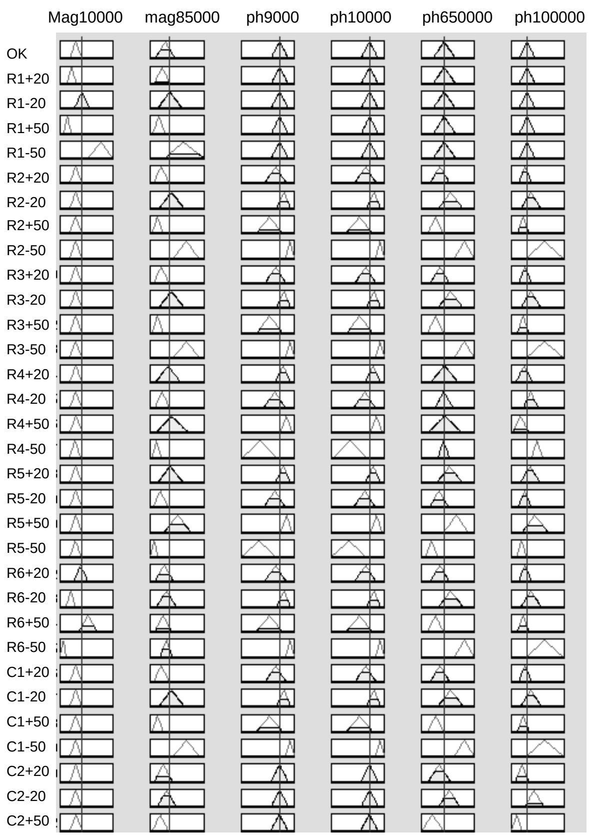


Figure 4.9: Fuzzy input sets activation for $R_1 - 20\%$

Therefore, the conclusion is that R_1 is faulty, since it is the only component that has an estimated value outside the tolerance limits ($R_{1nom} = 2.7K$). Something similar can be done in the other cases. There are 4 components that give a wrong diagnosis: $R_2 - 20\%$, $R_3 - 20\%$, $R_5 - 20\%$ and $C_1 - 20\%$. In the case of R_2, R_3 and $C_1 - 20\%$ the diagnosis is R_5, and the opposite happens for $R_5 - 20\%$. The signature of $R_2 - 20\%$, produces the output set depicted in Figure 4.11 for R_2.

This is because in that particular case the measures overlapping fire rules that produce a wrong centroid when defuzzifying. The centroid method gives an average of $R_2 = 0.911K$, therefore, R_2 is considered non faulty.

On the other hand, Figure 4.12 shows the output for R_5.

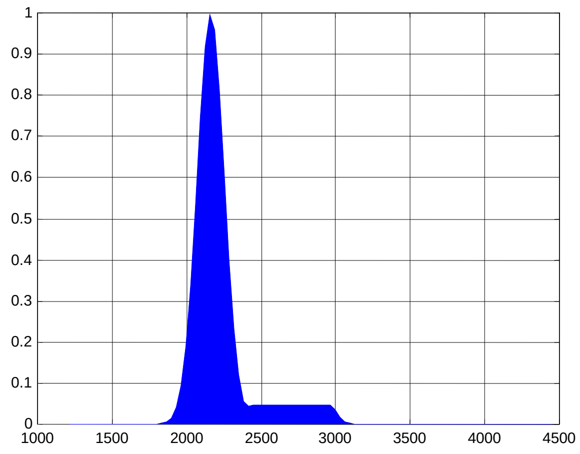


Figure 4.10: R_1 output set for $R_1 - 20\%$

R1	R2	R3	R4
2223	1000	10000	1504

R5	R6	C1	C2
12010	2757	10nF	10nF

Table 4.1: Results with fuzzy

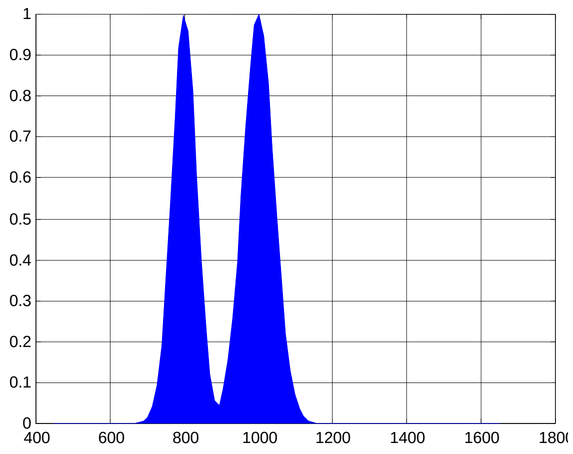


Figure 4.11: R_2 output set for $R_2 - 20\%$

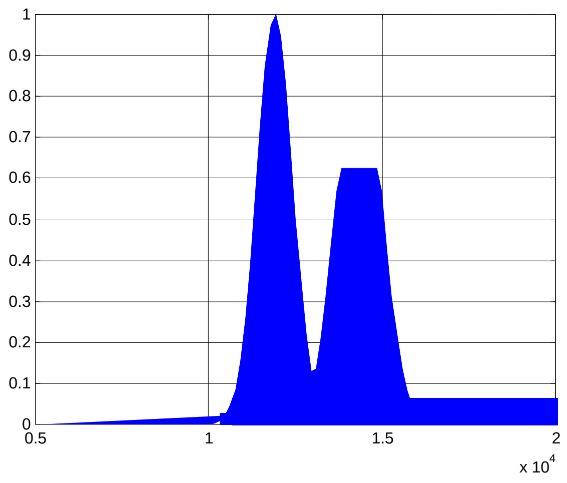


Figure 4.12: R_5 output

The centroid method applied to this set produces an output $R_5 = 13.58K$. So, R_5 is considered to be faulty. The diagnosis being worse is not serious, since without fuzzy, it gave a lot of overlapping that makes the situation difficult to diagnose. Something similar happens with the other cases.

As it has been previously explained, the system has to be able to make decisions in other unlearned situations. For example, the system was tested with the new cases $R_1+15\%$, $R_6-30\%$, $C_2-30\%$ and $R_3+70\%$. The diagnosis provided is given in Table 4.2

Fault	Diagnosis
R1+15%	R1
R6-30%	R6
C2-30%	C2
R3+70%	R2, R3, R5, C1

Table 4.2: Diagnosis for unlearned cases

So, the system does what is expected when applying fuzzy. It interpolates to diagnose the new situations, and it gives satisfactory results.

In a real circuit the components have tolerances, and how the diagnosis system copes with this factor should be tested. Trying the system with 100 cases for each considered fault, randomly chosen using Monte-Carlo with a Gaussian distribution, the percentage of successes is shown in Table 4.3.

Fault	Diag. Succes	Fault	Diag. success
R1+20	92%	R5-20	20%
R1-20	84%	R5+50	96%
R1+50	98%	R5-50	93%
R1-50	100%	R6+20	88%
R2,R3,C1 +20	84%	R6-20	83%
R2,R3,C1 -20	18%	R6+50	100%
R2,R3,C1 +50	99%	R6-50	100%
R2,R3,C1 -50	99%	C2+20	95%
R4+20	80%	C2-20	77%
R4-20	86%	C2+50	99%
R4+50	98%	C2-50	99%
R4-50	98%	Nominal	42%
R5+20	82%	Average	**84.40%**

Table 4.3: Diagnosis success for 100 runs. Frequency method.

The percentage of success means that the right component is diagnosed each time, although sometimes it does not appear alone. For example, the case $R_1 - 50\%$ is detected in 100% of the cases. Actually, this percentage corresponds to R_1 in 98.7% of cases and 1.3% to the set $[R_1, R_6]$.

Table 4.3 shows that in the majority of cases, the success percentage is good. In the nominal case, it has a lower value because of overlapping with other sets. For instance, it diagnoses R_5 in 11.25% of the cases, and the set $[R_2, R_3, C_1]$ in 9% of them. It has to be mentioned as well, that the cases corresponding to R_2, R_3 and C_1 constitute an ambiguity group for each deviation considered.

4.6.2 Time Method

Now, the signature parameters *overshoot*, *rise time*, *delay time* and *steady state* will be taken as the fuzzy system inputs. As in the frequency method, each input attribute is divided into 33 triangular shaped *membership functions*. For example, for $R_2 + 20\%$ the final diagnosis is R_2, R_3 and C_1. Doing the same for the other faults, it can be concluded that the fact of applying fuzzy does not worsen any of the results obtained with the fault dictionary using the ramp method, and the huge overlapping is improved. Taking into account that real circuits have tolerances, the system was tested making a randomly Gaussian distributed sweep in the component values. 100 runs were made for each considered fault, and the percentage of successes is given in Table 4.4.

Fault	Diag. Succes	Fault	Diag: success
R1+20	94%	R5-20	15%
R1-20	75%	R5+50	100%
R1+50	100%	R5-50	99%
R1-50	100%	R6+20	91%
R2,R3,C1 +20	88%	R6-20	68%
R2,R3,C1 -20	14%	R6+50	100%
R2,R3,C1 +50	99%	R6-50	100%
R2,R3,C1 -50	99%	C2+20	96%
R4+20	94%	C2-20	77%
R4-20	78%	C2+50	100%
R4+50	100%	C2-50	100%
R4-50	100%	Nominal	40%
R5+20	87%	Average	**84.56%**

Table 4.4: Diagnosis success for 100 runs. Ramp method.

As occurs with the frequency method, some signatures cause the fired rules to point to an incorrect diagnosis. For example, a random simulation of the fault $R_5 - 20\%$ while other components stay at their tolerance margin from a nominal of 10%, produces the following component values: $R_1 = 2.6966K$, $R_2 = 1.0170K$, $R_3 = 9.9316K$, $R_4 = 1.4443K$, $R_5 = 9.6000K$, $R_6 = 2.6995K$, $C_1 = 9.8752\ nF$ and $C_2 = 9.8169\ nF$. Simulating the circuit with these values, the corresponding temporal signature activates the rules as shown in Figure 4.13. It can be appreciated that errors estimating R_2, R_3, R_5 and C_1 are produced.

The conclusions are not made worse by testing the method with non predicted cases, so the fuzzy system is able to interpolate and predict unlearned cases. But, the cases $C_1 + 20\%$, $R_2 + 20\%$ and $R_3 + 20\%$, among others, remain impossible to distinguish, because the responses are almost identical. They form an ambiguity group. A similar conclusion is made for the other deviations of R_2, R_3 and C_1.

Another important fact to highlight, is that the method is quite good locating the fault but it loses efficiency when identifying. Figure 4.14 depicts errors in the components R_1 and R_5 value estimations for the same set of 100 considered cases for each fault in the case of the biquadratic filter with the temporal signature. The x axis corresponds to the case number. Therefore,

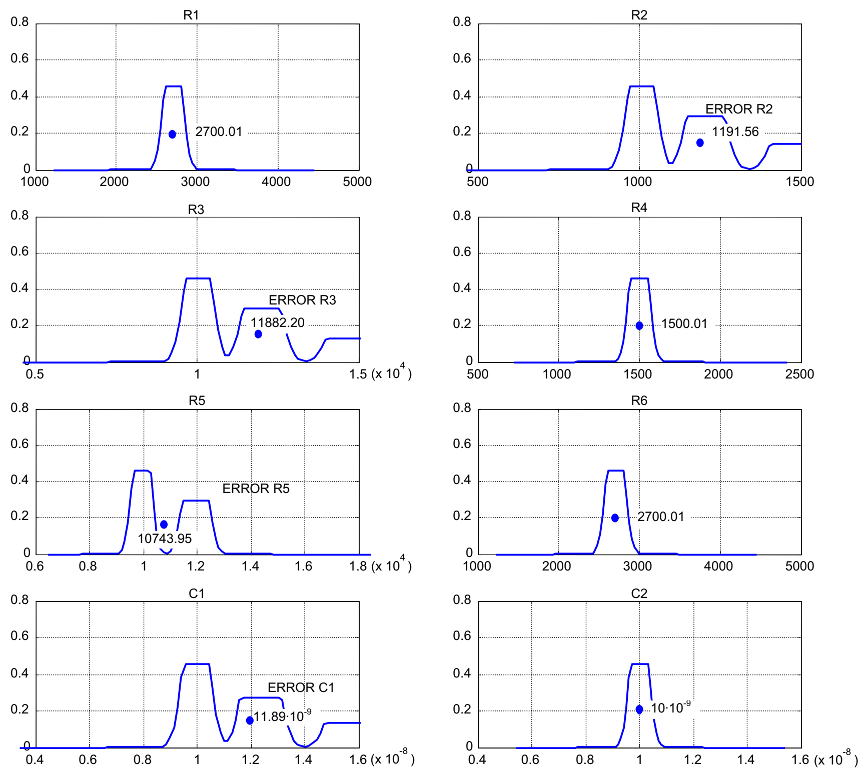


Figure 4.13: Rules activation for the randomly simulated fault $R5 - 20\%$. Ramp method

cases from 1 to 100 correspond to the fault $R_1 + 20\%$; cases from 101 to 200 correspond to simulations of fault $R_1 - 20\%$ while other components stay in the nominal range; cases 3101 to 3200 correspond to the fault $C_2 - 50\%$ and cases from 3201 to 3300 to the non faulty circuit situation (all components with a value contained in the nominal range).

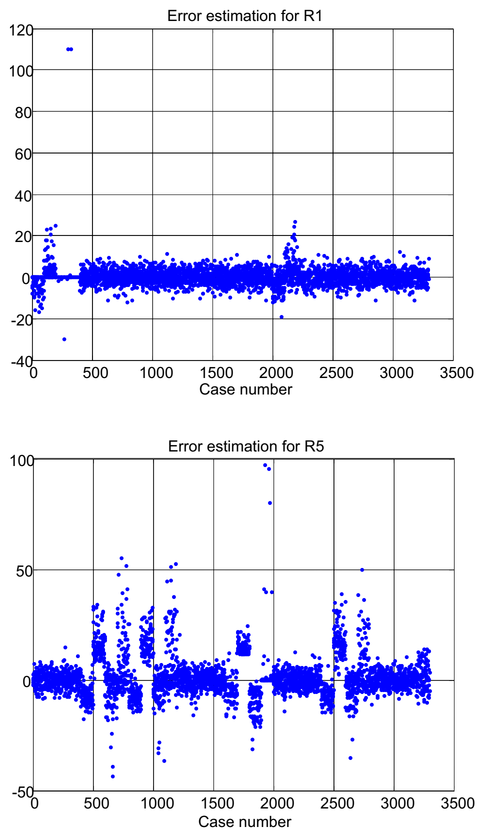


Figure 4.14: Error estimating components R_1 and R_5

As it can be observed, there are some cases where the R_1 value is estimated with an error less that 10%, which is a good approach. But in some cases, the error is superior to the 20% or in a few cases even higher than 100%. The situation is worse for R_5, as there were less cases estimated with an error below 10%. Something similar can be said for the other components. This error is made worse when deviations not previously considered, say 38%, are given.

4.7 Conclusions on the Fuzzy Approach

Parameter identification by means of fuzzy modeling is introduced into the previously described temporal or frequency dictionaries. Triangular membership function shapes were tried first, because of their simplicity. The authors can state that there are no great differences between these and the real measures distribution shape observed at the circuit output.

After applying fuzzy modeling to the time and frequency methods, an improvement in the diagnostic is obtained if compared with the classic dictionary. Furthermore, from the set of faults used to generate the dictionary, the system is able to interpolate and locate the faults that are not in the original set.

The system including fuzzy sets has been tested taking into account that real circuits are affected by tolerances. In particular, when testing the fault isolation capability of the system, 100 cases were generated for each fault considered. The results are summarized in tables 4.4 and 4.3, and show that good diagnoses are made despite of tolerances (84.56% and 84.40% respectively) compared to the classic dictionary methods (82.04% and 81.92%). Again, the equivalence between the saturated-ramp and frequency methods is evident. These tables consider the diagnosis as a success when the component that deviates from its nominal behavior is given. Since only simple faults are allowed, the component that has a great deviation from its nominal value will be the one considered to be faulty. Therefore, the method is quite good for wrong component detection. Also, it has to be taken into account that the method reduces its success rate when the diagnosis of the component is desired. As a conclusion, the fuzzy approach proposed can help in detecting wrong components, but care must be taken when using it for estimating the component deviation.

Chapter 5

DIAGNOSING CIRCUITS USING A CBR-SYSTEM

5.1 Introduction

Diagnosis of parametric faults in analog circuits is the main goal of this work. After a review of the main techniques for analog circuit testing, it is evident that some methods are adequate for detecting faults but not for diagnosing them. Some of the others need to take measures at several nodes in order to identify the component deviation. Also, the majority of the methods do not have learning capability, making diagnosis of new faulty situations difficult.

On the other hand, AI methods are able to learn from new situations. In Chapter 2, a neural network is designed in order to diagnose the biquadratic filter. After adequate training, the system can correctly diagnose the considered faults. But, if more faults have to be considered, a more dense neural net with a great deal of outputs (as many outputs as possible diagnosis) with their corresponding high dimensional hidden networks and a long training process should be carried out.

Fault dictionaries are also a good technique for fault detection, but, as explained previously in Chapter 3, they have two main drawbacks:

- When tolerances are taken into account, their success rate decreases.
- For non previously considered faults, there is a high probability that these faults are diagnosed incorrectly.

One possible solution to cope with these two difficulties is to include more fault signatures in the dictionary. Hence, in this chapter it is firstly proposed to spread the dictionary in order to take the effect of tolerances into account. As the generated dictionary is huge and is built randomly (Monte-Carlo algorithm), it is necessary to reduce its size. Then, algorithms from the data mining environment are proposed to fulfill this objective. In this domain, a *fault* is generally known as a *Class* and the *fault signature* as an *instance* or *exemplar*. Also it is better to talk about *classification* rather than *test* or *diagnosis*. The goal of these reduction methods is to eliminate noisy or redundant cases while maintaining efficiency. In particular, two reduction methods are explained in detail in this chapter.

But, the diagnosis of faults that have not been previously considered has not yet been solved. The idea proposed in the present thesis is to include knowledge about these new situations in the system completing it towards a CBR-system in order to solve these drawbacks. The CBR-system is designed by taking a fault dictionary as a starting point. The main CBR-cycle tasks are described in detail, and special attention is taken with the learning process and case base maintenance.

5.2 Fault Dictionaries as Case Bases

When building a CBR-system, it is very important to select an appropriate initial case base. It can be an empty base that will grow by learning from new situations or a base containing some faults. With regards to this last group, fault dictionaries are good candidates for being a starting point (Sheppard and Simpson, 1998). It is straightforward to use the dictionary table as a case base only with slight modifications.

It is common in fault dictionaries to select $\pm 20\%$ and $\pm 50\%$ from the nominal values of the components as a world of faults, although other deviations could be considered. These faults seem to be distributed well enough in order to cover a possible set of typical faults. Then, each considered fault can be simulated and the measures can be stored in the initial dictionary. This dictionary will have as many rows as there are considered faults, and as many columns as there are signature characteristics.

5.2.1 Spreading the Dictionary

The first approach to take the effect of tolerances into account is to spread the dictionary with more faults. There are several methods for considering tolerances, such as the band fault method (Pahwa and Rohrer, 1982), Monte-Carlo simulations or the high level model proposed in (Ozev and Orailoglu, 2002) among others. In our case Monte-Carlo simulation has been selected because all generated faults can be stored in the first instance, and this is a straightforward step to implement a case base reasoning system.

One of the questions that immediately arises is how many Monte-Carlo runs are necessary to predict a good interval $(L1, L2)$ for each measure. $L1$ and $L2$ are known as the tolerance limits. It is important to find the minimum number of trials that are necessary to obtain a good estimation of these interval limits. Of course, this assertion can only be made in a probabilistic form. Given a number of trials n, it can be asserted that, with a given probability β, at least a proportion γ of the distribution lies between the tolerance limits $L1$ and $L2$.

In our case, as the tested circuits give a Gaussian like measure distribution, the limits of this normal function are chosen. According to (Stuart *et al.*, 1999), given a sample of size n with mean $\bar{x}$, variance σ and the probabilities γ and β, it is possible to find a value λ that approximates the tolerance intervals according to Eq. 5.1

$$
\begin{aligned}
L_1 &= \bar{x} - \lambda\sigma \\
L_2 &= \bar{x} + \lambda\sigma
\end{aligned}
\tag{5.1}
$$

The mathematical expressions proposed by (Stuart *et al.*, 1999) are given in table form in (Kokoska and Nevison, 1992). This table shows the value of λ (they call it k) that defines the tolerance interval $(L1, L2)$ containing a proportion γ of the population with a probability β. Table 5.1 shows some of them.

Number of Trials	β=0.95			β=0.99		
	γ=0.95	γ=0.99	γ=0.999	γ=0.95	γ=0.99	γ=0.999
		...			...	
n=10	3.379	4.433	5.649	4.265	5.594	7.129
n=11	3.259	4.277	5.452	4.045	5.308	6.766
n=12	3.162	4.150	5.291	3.870	5.079	6.477
...		...			...	
n=100	2.233	2.934	3.748	2.355	3.096	3.954
n=110	2.218	2.915	3.723	2.333	3.066	3.917
n=120	2.205	2.898	3.702	2.314	3.041	3.885
...		...			...	
n=500	2.070	2.721	3.475	2.117	2.783	3.555
n=600	2.060	2.707	3.458	2.102	2.763	3.530
n=700	2.052	2.697	3.445	2.091	2.748	3.511
...		...			...	
∞	1.960	2.576	3.291	1.960	2.576	3.291

Table 5.1: Tolerance interval factor

It is necessary to derive interval limits that represent the distribution margin with a certain high confidence. Hence, values of λ corresponding to $\beta = 0.99$ and $\gamma = 0.99$ are selected. That is, there is 99% probability that 99% of the population is contained in the interval $(L1, L2)$. The number of trials n will be according to the error estimation that is tolerated. Of course, this value is related to the standard deviation σ that the measures have. Going back to the biquadratic filter example, if 2000 runs are done for the fault R1+20% with the Monte-Carlo method using Gaussian randomly distributed values, the distributions in Figure 5.1 are obtained. Similar distributions can be depicted for the other considered faults.

As can be observed, measures are distributed in a Gaussian way for the biquadratic filter. Table 5.2 displays the mean and the standard deviation for each measure.

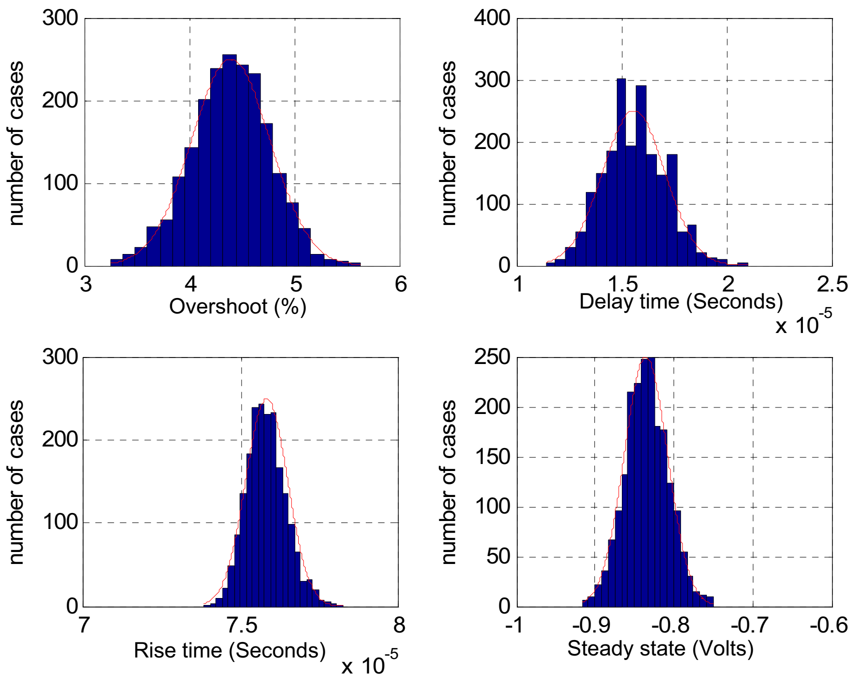


Figure 5.1: Measures distribution for the ramp method

Fault R1+20		
	Mean	Standard deviation
SP	4.393	0.3735
td	15.66 μs	1.479 μs
tr	75.76 μs	0.663 μs
Vest	-0.834	0.0275

Table 5.2: Fault R1+20% statistics

Trials	$\sigma - \sigma_\infty$
n=10	5.594 - 2.576 = 3.018
n=100	3.096 - 2.576 = 0.520
n=500	2.783 - 2.576 = 0.207

Table 5.3: Comparison with the infinite trials case

The case of 2000 runs can be considered as a situation close to infinite trials. But, if a lower number of trials are carried out it has to be evaluated how much closer the situation is to the infinite trials case. If $n = 10$, $n = 100$ and $n = 500$ trials are performed, the difference with the λ parameter compared to the infinite trials case is given in Table 5.3 (using Table 5.1)

Therefore, there is an error of 3.018σ, 0.520σ and 0.207σ respectively on the interval estimation. Applying this to the previous $R_1+20\%$ fault example, it corresponds to the absolute errors given in Table 5.4. The percentage is with respect to the mean of the measure.

	n=10	n=100	n=500
SP	1.127 (19.99%)	0.1942 (3.44%)	0.0773 (1.37%)
td	4.46 μs (21.26%)	0.769 μs (3.66%)	0.306 μs (1.45%)
tr	2 μs (2.56%)	0.345 μs (0.44%)	0.137 μs (0.17%)
Vest	0.083 (9.93%)	0.0143 (1.71%)	0.0057 (0.68%)

Table 5.4: Estimated errors for n=10, 100 and 500 trials

Observing Table 5.4, it is obvious that taking $n = 10$ produces a non permissive error, while taking more than 500 trials is not worthwhile. Therefore, 500 runs for each considered fault, with Monte-Carlo method using Gaussian randomly distributed values of the components were used to generate the spread dictionary. For the biquadratic filter, a total of 16500 cases were obtained. If ambiguity groups are considered, this set is reduced to 12500.

5.2.2 Reduction Algorithms

Previous steps generate a huge dictionary, making it difficult to seek for similar cases, and wasting memory space while keeping noisy and redundant cases. For dictionary size reduction, instance pruning techniques can be applied. These techniques are based on the improvements of the nearest neighbor algorithm. They can be classified according to several factors (Wilson and Martinez, 2000*b*):

- *Search direction.* They can be incremental or decremental. The former starts with an empty set and adds an instance if it fulfills certain criteria. The latter starts with a big set of instances and removes redundant or noisy data. Decremental search direction methods are, in general, computationally more expensive than incremental algorithms. But, they usually provide greater storage reduction and better accuracy. An example of the incremental algorithm is the IB3 (Instance Based Learning Algorithm 3) method, while DROP4 (Decremental Reduction Optimization Procedure 4) is a type of decremental method.

- *Retaining central points, border points or other sets.* The idea behind keeping border points is that central points affect the decision less than border points, so the former can be deleted. Border points are removed if they are considered noisy or they are not in accordance with their neighbor's class.

- *Used distance function.* The Euclidean distance is the most common metric used in nearest neighbor and other algorithms. Several distance functions are defined in order to deal with non numeric and unsorted attributes. Examples of these distances are Euclidean, Clark, Heterogeneous Value Difference Metric (HVDM), Interpolated Value Difference Metric (IVDM) and so on.

The IB3 (Aha *et al.*, 1991) and DROP4 (Wilson and Martinez, 2000*a*) reduction algorithms are applied to the spread obtained dictionaries generated by Monte-Carlo. These techniques are explained here because they are the baseline of the proposed CBR-system learning algorithm.

Decremental Reduction Optimization Procedure (DROP4)

The Decremental Reduction Optimization Procedure (DROP) version 4 classifies an instance by giving the class of the K-nearest neighbors. As described by the authors, the method tends to preserve border instead of center points.

The neighbors of an instance are the k nearest instances according to a certain metric, where k is the number of neighbors to be considered. The method also uses the associate concept in the reduction algorithm. Associates of an instance are the exemplars that have this particular instance as a neighbor. Let's consider the following example to clarify these concepts. Let's take the two dimensional instance space in Figure 5.2

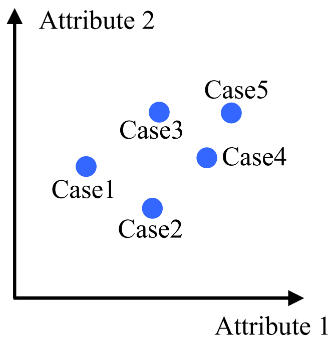


Figure 5.2: Two dimensional case space

If $k = 2$ is selected, the two nearest neighbors of each instance are resumed in Table 5.5

Case Num	Neighbor 1	Neighbor 2
Case 1	2	3
Case 2	1	4
Case 3	5	4
Case 4	3	5
Case 5	3	4

Table 5.5: Neighbors example

Recalling the associate concept, Table 5.6 summarizes the associates of each instance. For example, *case 4* is a neighbor of *case 2*, *case 3* and *case 5*. Hence, the associates of *case 4* are *case 2*, *case 3* and *case 5*.

Case Num	Associates		
Case 1	2		
Case 2	1		
Case 3	1	4	5
Case 4	2	3	5
Case 5	3	4	

Table 5.6: Associated instances

In order to simplify data processing, a matrix with a row corresponding to each exemplar that contains '1' if an instance is one of its neighbors is built. For the previous example, the matrix can be built like the one shown in Table 5.7. The advantage of doing so is that the columns of this matrix have the information about the associates.

Case	Case 1	Case 2	Case 3	Case 4	Case 5
Case 1	0	1	1	0	0
Case 2	1	0	0	1	0
Case 3	0	0	0	1	1
Case 4	0	0	1	0	1
Case 5	0	0	1	1	0

Table 5.7: Neighbor/associates matrix

The associate concept is important because it helps to find the influence produced by the introduction of a new instance in the case base very quickly. When such an influence evaluation has to be done, first, the new instance neighbors are calculated, and then a new row and column in the neighbor/associates matrix are added. Afterwards, associates can be obtained by reading the column corresponding to the new instance.

As a decremental method, DROP4 begins with the entire set of instances T and removes unnecessary instances. S is the reduced set obtained. The basic rules to remove a case are:

1. Remove instance i from S if it is correctly classified by its neighbors, but with the condition that

2. At least as many of its associates in T could be classified correctly without i.

Once k nearest neighbors are extracted, an examination of their class has to be done. The simplest way to do so is to carry out a simple voting procedure, counting the cases corresponding to each class.

Example 12

Recalling the biquadratic filter, let us suppose that a base with several exemplars corresponding to each considered fault is taken, and $k = 3$ neighbors are selected for classification. Now, it has to be evaluated if *case 1* can be dropped or not. Table 5.8 shows the values stored for this particular case.

Case Number	SP	td	tr	Vest	Class
Case 1	3.97	18μs	77μs	-0.8716	R1+20

Table 5.8: Case 1 structure

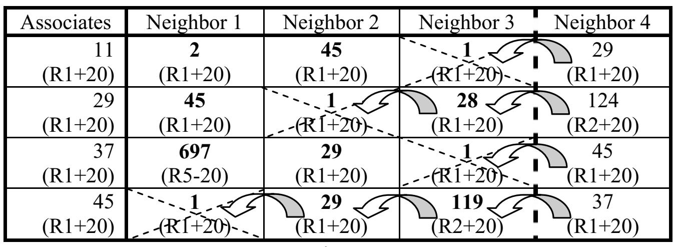
Applying the Euclidean distance, the neighbors depicted in Table 5.9 are obtained. Hence, *case 1* is correctly classified by its neighbors (3 votes for R1+20). So, rule 1 is satisfied for case dropping.

	Neighbor 1	Neighbor 2	Neighbor 3
Number	37	11	19
Class	R1+20	R1+20	R1+20

Table 5.9: Case 1 neighbors

Now, rule 2 has to be evaluated. Therefore, the associates of *case 1* are analyzed. If *case 1* was dropped, the new neighbors of these associates would be the ones displayed in Table 5.10

Associate 11 has 3 votes for R_1+20%, this is correct. Associate 29 has 2 for R_1+20% and 1 for R_2+20% giving a majority for R_1+20%. Something similar can be said for the other associates. The conclusion then is that dropping *case 1* does not make the diagnosis worse.



Associates	Neighbor 1	Neighbor 2	Neighbor 3	Neighbor 4
11 (R1+20)	**2** (R1+20)	**45** (R1+20)	~~**1** (R1+20)~~	29 (R1+20)
29 (R1+20)	**45** (R1+20)	~~**1** (R1+20)~~	**28** (R1+20)	124 (R2+20)
37 (R1+20)	**697** (R5-20)	**29** (R1+20)	~~**1** (R1+20)~~	45 (R1+20)
45 (R1+20)	~~**1** (R1+20)~~	**29** (R1+20)	**119** (R2+20)	37 (R1+20)

Dropping case 1

Associates	Neighbor 1	Neighbor 2	Neighbor 3
11 (R1+20)	**2** (R1+20)	**45** (R1+20)	29 (R1+20)
29 (R1+20)	**45** (R1+20)	**28** (R1+20)	124 (R2+20)
37 (R1+20)	**697** (R5-20)	**29** (R1+20)	45 (R1+20)
45 (R1+20)	**29** (R1+20)	**119** (R2+20)	37 (R1+20)

New associates neighbors after dropping case 1

Table 5.10: Case 1 associates dropping results

Let's now consider *case 5* given in Table 5.11

Case Number	SP	td	tr	Vest	Class
Case 5	4.33	17μs	76μs	-0.8811	R1+20

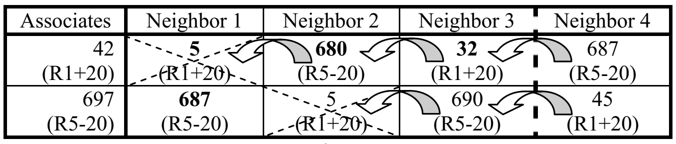
Table 5.11: Case 5 structure

Its neighbors are shown in Table 5.12, and rule 1 is accomplished since *case 5* is correctly classified (2 votes for $R_1 + 20\%$ against 1 vote for $R_5 - 20\%$).

	Neighbor 1	Neighbor 2	Neighbor 3
Number	697	36	45
Class	R5-20	R1+20	R1+20

Table 5.12: Case 5 neighbors

Giving a look at its associates and their corresponding neighbors depicted in Table 5.13, initially they are correctly classified. If *case 5* is dropped, associate 42 of class $R_1 + 20\%$ will be diagnosed as $R_5 - 20\%$. Therefore, rule 2 is not accomplished and *case 5* should not be dropped from the base.



Associates	Neighbor 1	Neighbor 2	Neighbor 3	Neighbor 4
42 (R1+20)	~~**5** (R1+20)~~	**680** (R5-20)	**32** (R1+20)	687 (R5-20)
697 (R5-20)	**687** (R5-20)	~~5 (R1+20)~~	690 (R5-20)	45 (R1+20)

Dropping case 5

Associates	Neighbor 1	Neighbor 2	Neighbor 3
42 (R1+20)	**680** (R5-20)	**32** (R1+20)	**687** (R5-20)
697 (R5-20)	**687** (R5-20)	**690** (R5-20)	**45** (R1+20)

New associates neighbors after dropping case 5

Table 5.13: Case 5 associates dropping results

But this simple method does not take into account any information on the distance. It is logical to think that it is more probable that the new case belongs to the class of the closest retrieved neighbor. Then, as proposed in (Wilson and Martinez, 2000*a*) a weight can be given to the retrieved cases according to their distance from the new case (distance-weighted voting). Wilson proposes 3 types of weights: linear, Gaussian and exponential. They are given by the following formulas:

$$\begin{aligned} w_j &= w_k + \frac{(1-w_k)(D_k-D_j)}{D_k} \quad \textit{Linear} \\ w_j &= w_k^{\frac{D_j^2}{D_k^2}} \quad \textit{Gaussian} \\ w_j &= w_k^{\frac{D_j}{D_k}} \quad \textit{Exponential} \end{aligned} \tag{5.2}$$

where w_k is the weight given to the k neighbor, and D_k and D_j are the distances to the k neighbor and the j^{th} neighbor respectively. These weight-distances are depicted in Figure 5.3.

As it can be seen, the bigger the distance from the new case to the j^{th} neighbor, the lower the weight given to the j^{th} neighbor. After several tests, in our case, the weight-distance method that performs best is the exponential one.

For example, let us suppose that *case 633* that is class $R_5 - 20\%$ has the neighbors given in Table 5.14. Neighbor 650, with class $R_5 - 20\%$, is at a

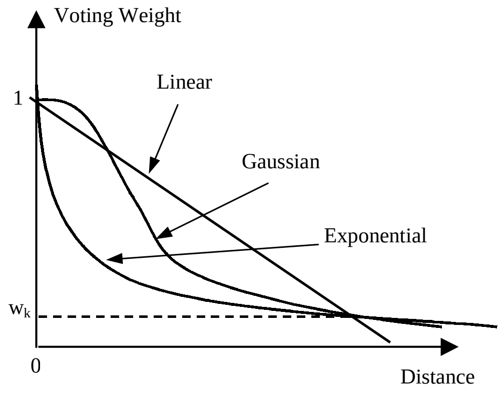


Figure 5.3: Possible weight kernel given to distances

distance of 0.012, while neighbors 26 and 43, with class $R_2 + 20\%$, are at a distance of 0.15 and 0.23 respectively.

	Neighbor 1	Neighbor 2	Neighbor 3
Number	650	26	43
Class	R5-20	R1+20	R1+20
Distance	0.012	0.15	0.23

Table 5.14: Case 633 neighbors

Applying the weighted-voting process with the exponential kernel taking $\omega_k = 0.2$, the obtained weights are $\omega_1 = 0.92$, $\omega_2 = 0.35$ and $\omega_3 = 0.2$ for each neighbor. Hence, the weighted-voting assigns a vote of 0.92 for $R_5 - 20\%$ and 0.55 for $R_1 + 20\%$, giving a final classification of $R_5 - 20\%$.

The method has been applied to the biquadratic filter shown in section 1.7. Taking the spreading dictionary of 12500 cases generated in the previous section (considering ambiguity groups), DROP4 is used to reduce its size trying to keep its efficiency. A reduced dictionary of 1112 cases is obtained, (8.8% of the original number of cases). After that, the reduced dictionary is tested using a randomly generated case set with 100 simulations for each considered fault (a total of 2500 test cases). Again, the Euclidean distance is used and the class of the closest extracted case is taken as a classification of the input case. The results are shown later in this chapter in Table 5.15.

Instance-Based learning algorithm (IB3)

The Instance-Based learning version 3 algorithm is considered as an incremental technique. The idea is to start with an empty base and then begin selecting potential cases to be introduced. The performance of each stored exemplar is monitored and the ones that do not perform well are discarded. This is done by maintaining a classification record for each instance s_i stored. This record indicates how the instance is performing the classification of instances of the same class and it gives an idea of how it will perform in the future. The greater the correct number of classifications, the greater the value contained in the record. Making wrong classifications makes the value of the record decrease.

Two predetermined thresholds are set. If the record of an instance has a figure higher than a certain pre-established value $z = C_{max}$ (confidence index), it is used to classify the subsequent instances. If it is less than a certain value $z = C_{min}$ (confidence index) the instance is believed to be noisy and will be dropped from base S. If it lies between the two, it is not used for prediction but its performance record is updated.

The confidence limits used in IB3 are the ones defined by the success probability of a Bernoulli process. This probability is a random variable which means it corresponds to the true probability of the process and its bounds which, with a certain confidence, can be calculated using Eq. 5.3 (Witten and Frank, 2000).

$$\frac{p + \frac{z^2}{2n} \pm z\sqrt{\frac{p(1-p)}{n} + \frac{z^2}{4n^2}}}{1 + \frac{z^2}{n}} \tag{5.3}$$

In the equation, z is the confidence index, p is the proportion of instances that are of this class, and n is the number of the previously processed instances. The sign $\pm$ gives the lower and upper bounds. A graphical representation is shown in Figure 5.4. When the number of trials increases, the bound interval shrinks and approaches the true probability.

The IB3 idea is to use this concept for accepting or rejecting new instan-

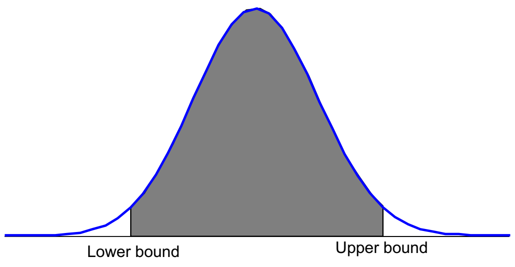


Figure 5.4: Probability success rate distribution

ces. If an instance s_i has c classification successes in n number of times that instance s_i is selected to classify other instances, the probability of success for this instance is close to c/n and contained in the bounds (L_{si}, U_{si}) derived from equation 5.3, with a confidence z. At the same time, if there are N_c instances with the same class C_i of the instance s_i in a set of N trials, similar bounds (L_{ci}, U_{ci}) could be obtained for the probability of an instance belonging to class C_i. The true class probability will be close to N_c/N.

If $L_{si} > U_{ci}$, the instance will be accepted; on the other hand, if $U_{si} < L_{ci}$, the instance will be rejected, as shown in Figure 5.5. This means that if the probability of success for the instance s_i is always above the probability of its class C_i, s_i is performing quite well for classifying instances of its class. On the other hand, if the probability success is always below the probability of its class, the instance is performing incorrectly or has little contribution to the classification of the instances belonging to the same class as it. Hence, it is removed.

IB3 normally takes a confidence index $C_{max} = z = 0.9$ for acceptance and $C_{min} = z = 0.7$ for rejecting. Note that accepting an instance is made more difficult than removing it. This is because exemplars with poor performance only contribute a little to the classification correctness, and they will probably be replaced by a new similar one during future training.

Figure 5.6 displays the evolution of the bounds considering that the instance s_i is participating in classifying subsequent cases. It can be seen that while the number of trainings increases, the instance contributes to the classification, but it performs incorrectly in most cases. The new case $j + 1$ makes the bound U_{si} lower than L_{ci}. At this moment, instance s_i will be rejected.

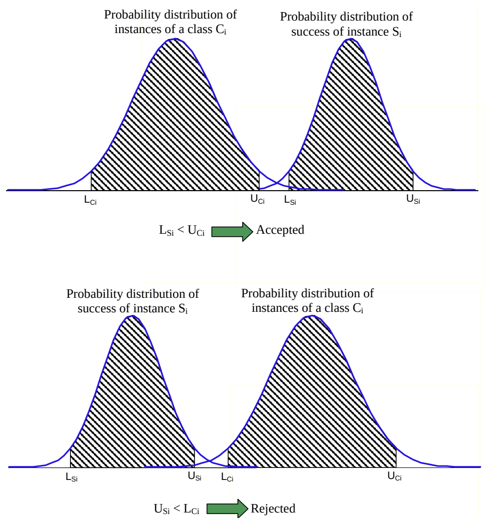

Probability distribution of instances of a class C_i
Probability distribution of success of instance S_i
L_{Ci}
U_{Ci}
L_{Si}
U_{Si}
$L_{Si} < U_{Ci}$ Accepted
Probability distribution of success of instance S_i
Probability distribution of instances of a class C_i
L_{Si}
U_{Si}
L_{Ci}
U_{Ci}
$U_{Si} < L_{Ci}$ Rejected

Figure 5.5: Acceptance-rejection decision

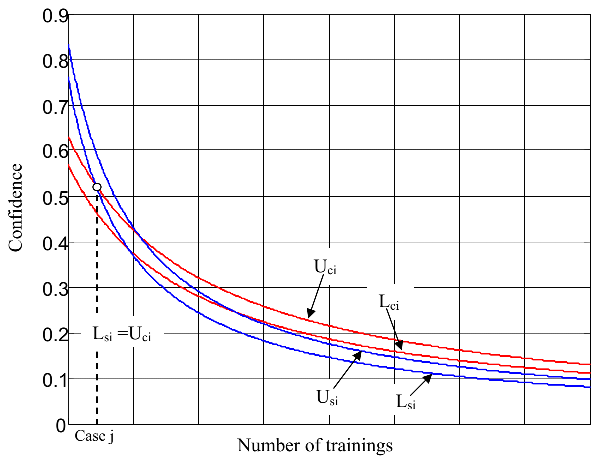


Figure 5.6: IB3 bounds evolution

This technique assumes that the number of successes is binomially distributed. Although in general this is not true, many IBL algorithms based on this process of selecting confidence bounds perform quite well (Aha *et al.*, 1991). Another important drawback is that the derived results are very sensitive to the order in which cases are analyzed in order to be either kept or rejected. Hence, several combinations of cases sorted in different orders should be analyzed and the best one retained.

Let's see a short example to clarify these ideas.

Example 13

Suppose that $N = 200$ trainings have been made and that a particular instance s_i has been used $n = 100$ times to classify other instances, and $c = 80$ of these are correct. Suppose also, that $N_c = 120$ instances of the total number of instances N are of the same class C_i of the particular instance. Now, the bounds of the true success rate of this instance can be estimated, at a particular confidence level, say $z = 0.90$ (90%). Applying Eq. 5.3 the bounds are $L_{si} = 76.17\%$ and $U_{si} = 83.35\%$. Hence, the success probability of the instance s_i is between these bounds with a confidence of 90%. On the other hand the probability that an instance of class C_i is given is between $L_{ci} = 56.85\%$ and $U_{ci} = 63.07\%$. Since $L_{si} > U_{ci}$, the instance s_i is considered acceptable.

Consider now, that after $N = 1000$ trainings we have the same instance with $n = 900$ interventions, $c = 200$ successes and $N_c = 700$ instances of class C_i. The instance has a lot of interventions in the new instances classification, taking into account that there is a great proportion of instances in the same class as it, but its success rate is low. It is a perfect candidate to be rejected. Numerically, the same information is provided by the probability bounds. The new probability success bounds are $L_{si} = 21.27\%$ and $U_{si} = 23.21\%$ and the class probability bounds $L_{ci} = 68.98\%$ and $U_{ci} = 71\%$. These values are obtained from Eq. 5.3 with a confidence index of $z = 0.70$ (70%). Therefore, $U_{si} < L_{ci}$, and the instance s_i will be removed.

Applying this method to the spread dictionary of 12500 cases obtained for the biquadratic filter, the dictionary is reduced to 2457 instances (19.6% of the

Fault	Classic	Spread	DROP4	IB3
R1+20	84	83	85	77
R1-20	87	94	90	90
R1+50	99	98	99	97
R1-50	100	100	100	100
R2+20,R3+20,C1+20	41	30	41	35
R2-20,R3.20,C1.20	36	35	35	31
R2+50,R3+50,C1+50	79	78	83	72
R2-50,R3-50,C1-50	94	99	96	98
R4+20	85	89	87	78
R4-20	88	84	88	80
R4+50	98	98	98	98
R4-50	100	100	100	100
R5+20	47	33	46	36
R5-20	38	40	38	43
R5+50	82	78	83	75
R5-50	91	93	94	93
R6+20	79	78	79	78
R6-20	86	83	82	74
R6+50	98	99	100	100
R6-50	100	100	100	100
C2+20	82	75	74	76
C2-20	89	90	90	83
C2+50	99	99	99	95
C2-50	100	100	100	100
NOM	69	61	72	57
Average	**82.04**	**80.68**	**82.36**	**78.64**

Table 5.15: Comparing classic, spread, and reduced dictionaries efficiency

original size). Table 5.15 shows a comparison of the success diagnosis between a classical dictionary, a spread dictionary and a DROP4-reduced dictionary. The voting procedure is carried out using a distance-weighted exponential kernel with $\omega_k = 0.2$. The file used for testing is the one constituted by 100 instances corresponding to each fault considered and 100 for the nominal situation.

Comparing the IB3 reduced-dictionary results with the ones obtained by the DROP4 reduced-dictionary, the latter produces a smaller base while it performs even better than the others. Hence, although the DROP4 is computationally more expensive than IB3, it results in a greater storage reduction and it will produce computational savings during execution thereafter. On

Component	OK diagnosed	Component Correct	Wrong	Non Detect.
R1	20%	62%	11%	7%
R2	18%	28%	41%	13%
R3	12%	45%	29%	14%
R4	15%	65%	15%	5%
R5	16%	41%	32%	11%
R6	20%	58%	15%	7%
C1	11%	33%	44%	12%
C2	29%	58%	5%	8%
Average	**17.625%**	**48.75%**	**24%**	**9.625%**

Table 5.16: Performance for non previously considered faults. DROP4

Component	OK diagnosed	Component Correct	Wrong	Non Detect.
R1	20%	56%	16%	8%
R2	16%	34%	39%	11%
R3	14%	48%	26%	12%
R4	14%	64%	18%	4%
R5	15%	35%	40%	10%
R6	21%	58%	14%	7%
C1	11%	33%	43%	13%
C2	29%	53%	14%	4%
Average	**17.5%**	**47.625%**	**26.25%**	**8.625%**

Table 5.17: Performance for non previously considered faults. IB3

the other hand, the spread dictionary performs even worse than the classic dictionary. This is because the instances have been generated randomly and there are some that are noisy, in the sense that they disturb the other classifications. But, reduced dictionaries have the same drawbacks when tested with a set of faults corresponding to any deviation in component values between $\pm 70\%$. The diagnosis results are quite poor. In particular, the average of success is 17.625% correctly characterized using a DROP4 reduced dictionary and 17.5% for an IB3 reduced dictionary. Tables 5.16 and 5.17 show the detailed results for both methods when the same 100 case dictionary is used.

In spite of performing slightly better than the classic dictionary, their percentage of success decreases. This situation inspired us to think about designing a complete CBR-system.

5.3 The CBR-System

Case Based Reasoning is an approach to problem solving that is able to use specific knowledge of previous experiences (Lopez de Mantaras *et al.*, 1997). A new problem is solved by matching it with a similar past situation. If the problem is solved, this new situation will be retained in order to solve other new ones. In the case of diagnosis, solving the problem means that the CBR-system proposes a solution that is satisfactory enough to identify the new fault.

It has several advantages with respect to other machine learning schemes: First of all it is easier to obtain rules and there is no bottleneck waiting for expert knowledge to be acquired. On the other hand, it is quite intuitive in certain tasks, such as diagnosis. At the same time it tolerates lazy learning schemes which means that the CBR-system can take advantage of these techniques that are well-known and permanently updated.

One of the main drawbacks is to know when to stop training. If the case base is overtrained, its efficiency falls. This is known as *the utility problem*. Hence, a good policy for the training and maintenance tasks is necessary. Another problem common to machine learning methods is how to train the system. The order in which the new cases are selected is very important, making the method more or less efficient and the case base size bigger or smaller. With this in mind, data mining techniques can be applied in order to help with data treatment and case base maintenance.

The first question that arises is how to represent a case. This is a very critical step since the following steps rely on it completely. A decision has to be made about which features should be stored in a case, to find the appropriate structure for describing the case contents and choose an appropriate structure and hierarchy. This structure is known as *Case Base Memory*.

CBR normally has the four stages depicted in Figure 5.7 for new problems solving (Aamodt and Plaza, 1994), which is repeated for each new experience:

1. Seek for a past situation similar to the new one (**Retrieve**). It is necessary to define the useful features, a metric function and the number of

cases to retrieve from the base.

2. Use that case to propose a possible solution (**Reuse**). According to a similitude index, a voting technique, or adapting the cases, the system provides a possible diagnosis. The adaptation can be *transformational* or *derivational* (Aamodt and Plaza, 1994). The first one uses the past case solution instead of the past method that constructed the solution.

3. Evaluate the suggested solution (**Revise**). The revision can be evaluated in the model or in the real world.

4. Update the CBR knowledge base with the new learned case (**Retain**). After the revision process, according to the proposed solution, it has to be decided if it is useful to retain the knowledge obtained from the new problem.

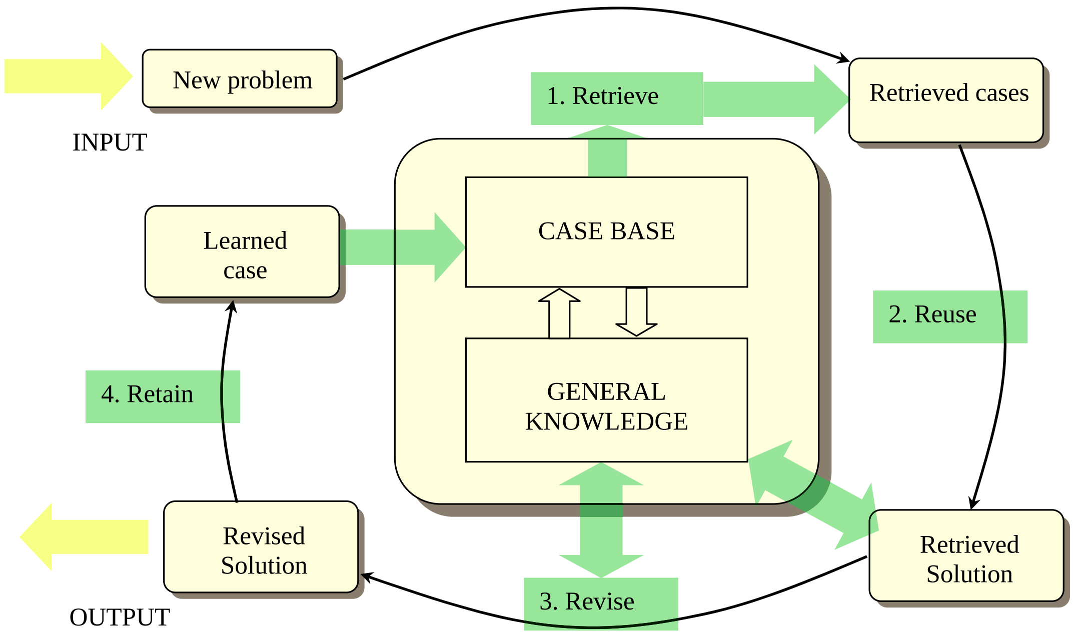


Figure 5.7: The CBR cycle

The diagnosis results obtained by the fault dictionary techniques can be improved if they are completed with the adequate knowledge and learning capabilities. It can be said that the system is learning if its percentage of success diagnosing the circuit increases when the CBR is trained. This happens because the system gains knowledge about the circuit. Hence it is important to know where the knowledge is contained. In a CBR-system there are four ”containers” which can carry knowledge (Richter, Sesimbra, Portugal, October 25, 1995):

1. The vocabulary (attributes, predicates etc.)
2. The similarity measure
3. The case base (Utility problem, competency analysis, Feature recognition, etc.)
4. The solution transformation (adaptation)

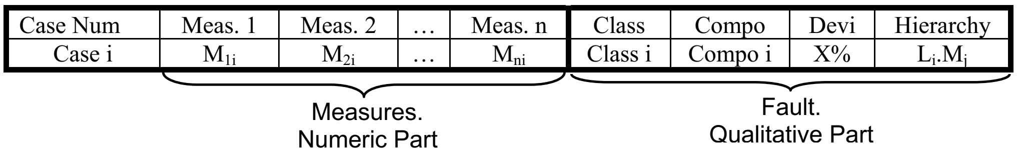
In this case it was decided to focus on the case base knowledge container that performs the learning process when new cases are introduced and gives special attention to maintaining it correctly because of the *utility problem*.

The next sections detail how a fault dictionary technique can be extended towards a Case-Based Reasoning system. First of all, the case base memory is defined, followed by a description of the CBR cycles.

5.3.1 Case Base Memory

The case structure is chosen to be the same used in the fault dictionary techniques, simply introducing a slight difference in the information about the fault. The proposed structure is shown in Figure 5.8. One part of the case is directly related with the measures taken from the circuit at one or several nodes. They could be temporal, frequency or static measures. This numeric part will be used to retrieve the most similar cases. The second part of the case contains information about the fault diagnosis.

Observe that the field *Class* has been maintained. As a reference, the classes associated with the faults considered in the classical dictionary ($\pm 20\%$ and $\pm 50\%$) are taken. When a fault has a deviation that does not correspond



Case Num	Meas. 1	Meas. 2	…	Meas. n	Class	Compo	Devi	Hierarchy
Case i	M_{1i}	M_{2i}	…	M_{ni}	Class i	Compo i	X%	$L_i.M_j$

Measures.
Numeric Part
Fault.
Qualitative Part

Figure 5.8: Case Structure

exactly to one of the original ones, the associated class will be the same given to the closest possible deviations considered as references. For example, if a fault is $R + 40\%$, its associated class will be the same as $R + 50\%$. But if a fault is $R + 35\%$, its corresponding class will be the same as $R + 20\%$. This $Class$ field is not used for classification purposes. It is only used in particular steps to help in the maintenance task explained later on in this chapter.

Concerning the other three qualitative fields, one of them has the faulty component location (*Compo*); the second contains the characterization of the fault (% of deviation from its nominal value *Devi*). When there are deviations of the components smaller than the tolerance, the circuit is considered to be not faulty. This is known as the nominal case (*Compo* $= Nom$).

The third field (*Hierarchy*) has additional information about the component, for example at level L_i and the module M_j to which the component belongs. Case base hierarchy is defined considering several levels depending on circuit complexity (Voorakaranam *et al.*, 1997). Therefore, the diagnosis result could be more or less precise depending on the retrieved qualitative parts, according to Figure 5.9. The last level corresponds to the faulty component deviation. The next upper level is defined as the component level. At this point, the system will only be able to diagnose which component is wrong, but not the fault deviation. Also, it is possible that certain faults can only be located just at a certain module, but not deep inside it. So, going to upper levels, the circuit is divided into modules. The number of module levels depends on the circuit complexity.

It is necessary to have certain knowledge on the circuit topology in order to build the case base hierarchy. For small circuits it can be done simply by inspection. For large circuits the method proposed in (Sangiovanni-Vicentelli *et al.*, 1977) can be used. An example of this method can be found in appendix B.

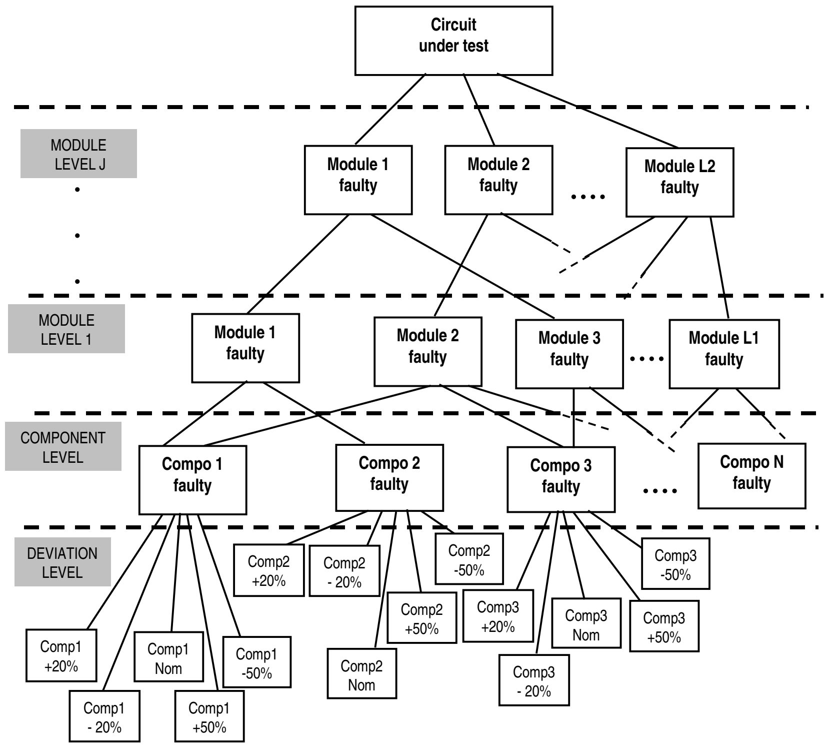


Figure 5.9: General case hierarchy

One of the other important decisions to make is the starting point. It is possible to start with an empty case base, or with a certain case base. As faults ±20% and ±50% are representative for the parametric faults compressed between ±70%, these cases are taken as an initial base. Of course the start can be a DROP-reduced dictionary, for example, but this is not a good idea because there will be a saturation of these cases, and they will be over-represented compared to the other cases.

The biquadratic filter proposed in section 1.7 is a small circuit that can be divided into blocks by inspection, as shown in Figure 5.10, consisting of three blocks (M_1, M_2, M_3) belonging to the same hierarchy level L_1.

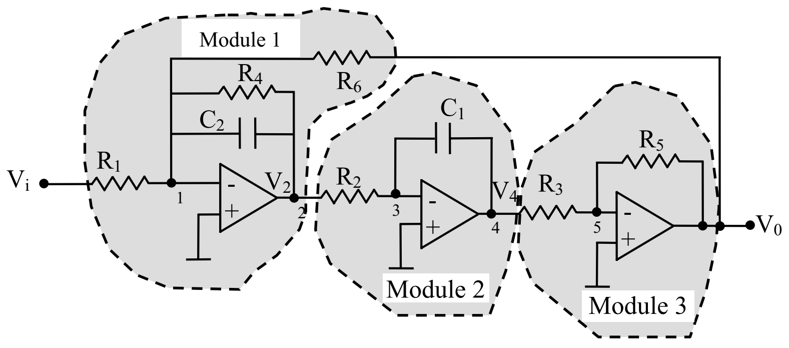
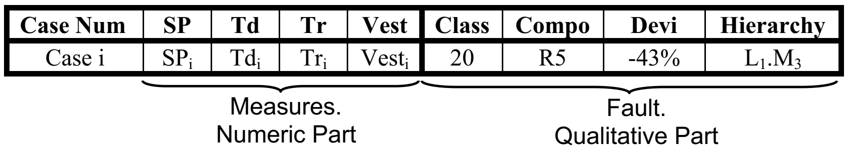


Figure 5.10: Biquad circuit modules decomposition

Component R_1 belongs to level L_1 and module M_1, while component R_5 belongs to the same level but to module M_3. Therefore, if the ramp method is selected, a case corresponding to the fault at R_5 with a deviation of -43% will have the appearance of Figure 5.11



Case Num	SP	Td	Tr	Vest	Class	Compo	Devi	Hierarchy
Case i	SP_i	Td_i	Tr_i	$Vest_i$	20	R5	-43%	$L_1.M_3$

Measures.
Numeric Part

Fault.
Qualitative Part

Figure 5.11: Case structure for the fault R1-43%

The classical dictionary considering faults ±20% and ±50% is taken as a starting point. Then, for the biquadratic filter, this is a set of 33 cases, or 25 cases if the ambiguous groups are considered.

5.3.2 Retrieve

It is necessary to define a metric function and the number of cases to retrieve from the case base. Since the proposed CBR-system uses numerical data corresponding to the measures for retrieval, we deal with continuous linear attributes; that is, attributes that can be any real number. Therefore, from among all possible distance functions (Wilson and Martinez, 1997*a*), the normalized Euclidean distance has been chosen. Attributes Normalization is necessary because of their different order of magnitude. For example the measures can be μsec, and amplitudes that corresponds to a magnitude of a fraction of $Volts$ or even $Volts$.

So, the distance between two instances is calculated as shown in equation 5.4

$$E(\overrightarrow{x}\overrightarrow{y}) = \sqrt{\sum_{i=1}^{m} \left(\frac{x_i - y_i}{range_i} \right)^2} \tag{5.4}$$

Where $\overrightarrow{x}$ and $\overrightarrow{y}$ are the vector instances to be compared, x_i and y_i are the corresponding attribute value i, and m is the number of attributes. $range_i$ is the difference between the maximum and minimum value of the attribute i.

The number of cases k to retrieve from the case base will be related to the value of k that produces the best diagnosis results. Normally it is a small odd number. In general, the more noisy the data is, the greater the optimal value of k. In our experiments, a value of $k = 3$ produces the best results. Taking a bigger value produces confusion in the diagnosis because of the extraction of cases corresponding to other different faults to be diagnosed.

5.3.3 Reuse

Once $k - nearest$ cases are extracted, they are used to propose a possible diagnosis. The proposal is to use the qualitative part of the extracted cases to derive a possible solution. Several situations can be given. If the *Compo* field of all the k extracted cases is the same, then the proposed solution is compounded using the measures of the new case, the *Compo* field of one of them and the average deviation of the extracted cases in the *Devi* field and the same module M_i and level L_j (Figure 5.12).

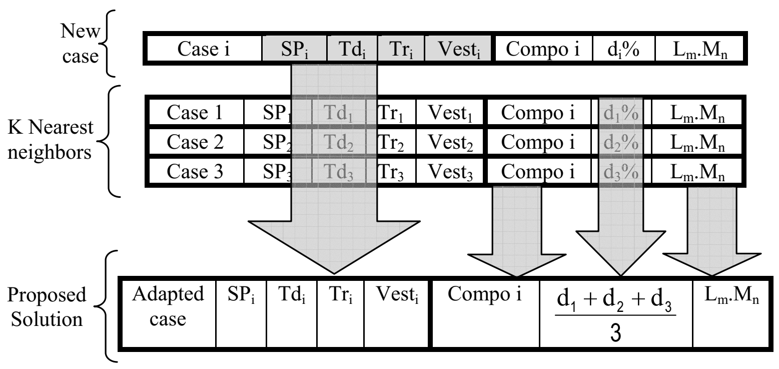


Figure 5.12: With the same *Compo* field adaptation

If the *Compo* is different, the proposed solution will have a *Compo* made up of the different components, and each of them with its corresponding deviation in *Devi*. *Hierarchy* will contain the common module M_n or several if different, and the first common level L_m. This is depicted in Figure 5.13. The case adaptation is carried out completely in the reuse task. It uses the past case solution instead of the past method that constructed the solution (transformational reuse) (Aamodt and Plaza, 1994).

At the same time it has to be taken into account that the nominal case (when there are deviations of the circuit components smaller than 10%) does not have any faulty component. Therefore, a label in the *Compo* field with a value *Nom* indicates that this case belongs to the *Nominal* situation.

5.3.4 Revise and Retain

Once the solution to the new presented case is proposed, it has to be revised. If the solution is considered correct and accurate enough, it is not necessary to retain the new case. On the other hand, if it is considered to be incorrect

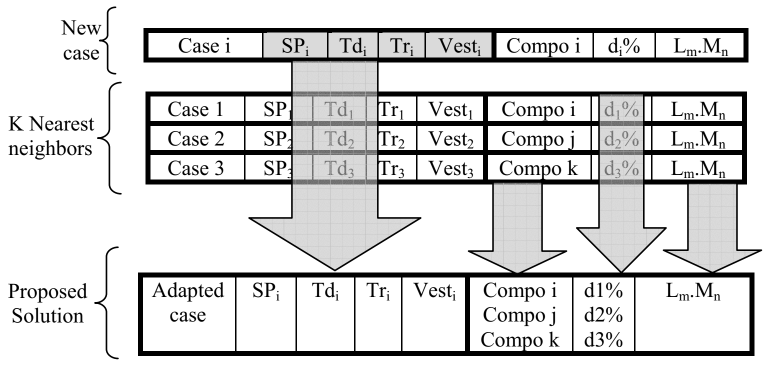


Figure 5.13: With different *Compo* field adaptation

or with poor accuracy, the new case will be retained in the case memory. The revision analyzes how the cases that constitute the adapted solution are performing the diagnosis. Hence it is supposed that the new case diagnosis is known by the user for its revision, which allows a decision to be made about when it should be retained. When the CBR-system is testing circuits with unknown faults, there is no revision task, since the proposed diagnosis can not be contrasted with the correct one. The reasoning follows the flow diagram in Figure 5.14.

There are 8 possible situations considered when revising while training:

1. The *Compo* field of the k extracted cases is equal to the new case, and the average deviation calculated has an error of less than 10%. This threshold is selected because this is the magnitude of the considered tolerances and an error of the same magnitude can be tolerated. The proposed solution is the correct one, and the case memory is enough to diagnose the new case. Hence, it is not necessary to retain the new case.

2. The *Compo* field of the k extracted cases is equal to the new case, but the average deviation has an error bigger than 10%. The present solution is considered to be not performing well and, the new case has to be introduced, if the DROP4 algorithm allows it.

3. The *Compo* field of the k extracted cases are equal between them, but different from the new case. If the *Compo* field of the extracted cases

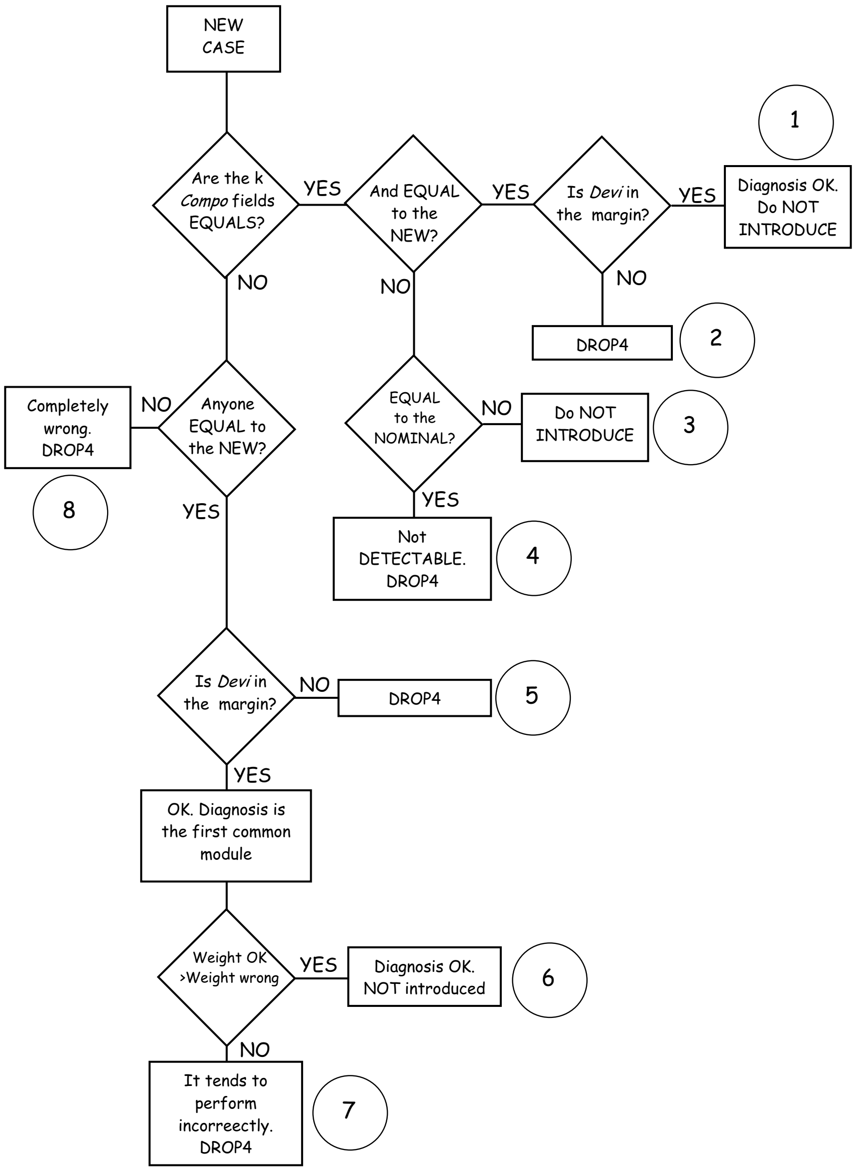


Figure 5.14: Revision task flow diagram

does not belong to the nominal diagnosis, the new case should not be introduced. This case will be an isolated case among all cases belonging to another type of fault. Its introduction worsens the diagnosis of that type of fault.

4. The *Compo* field of the k extracted cases are equal between them, but different from the new case. If the *Compo* field of the extracted cases belongs to the nominal diagnosis, the new case should be introduced, because it is not detectable. Of course its introduction is going to spoil the diagnosis of nominal cases a little, but it is preferable to have false alarms to not detecting faults. Hence, the case will be retained, if DROP4 decides that the new case is not going to disturb others.

5. There is at least one, but not all of the k extracted cases with the *Compo* field equal to the *Compo* field of the new case. If the *Devi* field of these cases is out of the tolerated range, the case should be retained, after DROP4 approval.

6. There is at least one, but not all of them, of the k extracted cases with the *Compo* field equal to the *Compo* field of the new case. The *Devi* field of that cases belongs to the tolerated error range. In this case the diagnosis of the corresponding first common block is correct. But if the sum of the weights associated to these cases is higher than the sum of the weights of the rest of the extracted cases, the case memory tends to be correct. Hence, the new case is not retained.

7. In the same way as the previous situation, if the sum of the weights associated with the extracted cases with the same *Compo* field and equal to the *Compo* field of the new case is lower than the sum of the rest of the extracted cases, the case memory tends to be wrong. In these conditions, the new case is not introduced.

8. There is a *Compo* field of the k extracted cases equal to the *Compo* field of the new case. The diagnosis in these conditions is clearly wrong, and the new case should be retained.

When a new situation is produced, retrieved cases from the case base provide a possible diagnosis. Recalling Figure 5.14, when a new case introduction is necessary, the DROP4 algorithm is applied. There are two possibilities:

1. New case retention does not influence previous cases diagnosis and
2. Its retention will produce misclassification of other previous cases.

In the first situation, the new case can be introduced without any problem. But in the second one, the introduction of the new case in the Case Base could be worse. The influence that a new case retention will have on the diagnosis of the present case base is evaluated using the *associate* concept described in the previous section. The Retaining algorithm is summarized in Figure 5.15

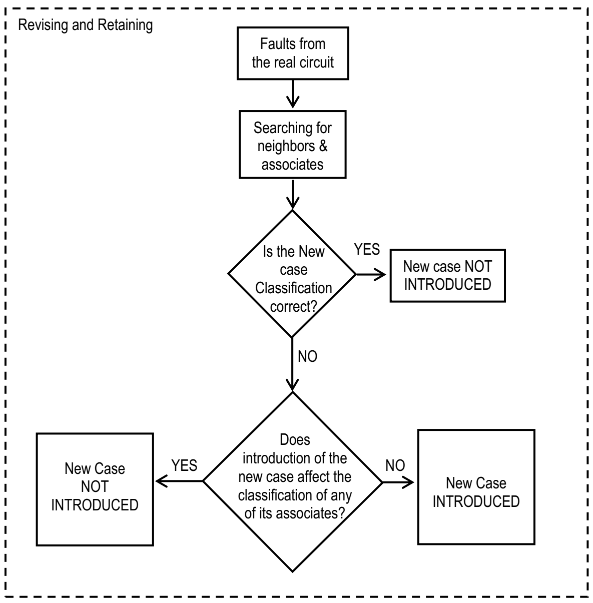

Revising and Retaining
Faults from the real circuit
Searching for neighbors & associates
Is the New case Classification correct?
YES
New case NOT INTRODUCED
NO
Does introduction of the new case affect the classification of any of its associates?
YES
New Case NOT INTRODUCED
NO
New Case INTRODUCED

Figure 5.15: Revising and retaining process detailed

Once neighbors and associates of the new case are calculated, two situations can be given. It could be that a new case does not have associates. This implies that the new case is not one of the k nearest neighbors of the existing cases in the case base. Hence, its introduction will not affect the previous classification of the cases already contained in the case base. On the other hand, if the new case has associates, it is necessary to evaluate how these associates are affected by the new case. If there is no change in the diagnosis of the associates due to the introduction of the new case, it can be added to the case base. Nevertheless, if there are more misclassifications than before its introduction, the case can not be retained directly in the case base, since it will solve its own classification but it will spoil some associate diagnoses.

Of course if $k > 1$ neighbors are taken for the decision, during the very first steps of the training procedure, it is obvious that any case introduction will be considered as disturbing. In this sense, the $Class$ field is preserved. If there are less than k cases belonging to a certain class, the new instance will be introduced, even if it disturbs the classification of the cases already contained in the base.

Forgetting Noisy Exemplars

To avoid the *utility problem* factor, a maintenance of the case base memory is proposed. It is very similar to the IB3 algorithm used when dropping cases. In fact, it uses the same criterion for removing cases, that is, when the performance of a particular case drops below a certain established value with a certain confidence index, the case is considered to be spoiling the diagnosis and it will be deleted. Equation Eq. 5.3 with a particular confidence level z is used to forget cases. If the lower bound of the success probability for a particular case L_{si} is below the upper bound of the probability U_{ci} then the case belongs to this class, the case will be considered to be removed. In this sense, the $Class$ field is used for comparing how the case is performing according to exemplars of its class, as IB3 does. The bigger the confidence z is taken, the sooner the instances are marked for removal. This can be seen in Figure 5.16. This figure is obtained by supposing that a case has interventions in the decisions made in the diagnosis of new instances but it is performing incorrectly each time. Observe that N_{cross} for $z = 0.9$ is lower than for $z = 0.3$.

At the same time, if a case has no intervention in the decisions of new cases of the same class, the index of class U_{ci} is increasing while the case index L_{si} stays the same. This is a situation where the case contained in the base is not significant when diagnosing cases of its class. Therefore, when $L_{si} < U_{ci}$, the case will be dropped.

Another parameter has been introduced in order to control the forgetting process. It is related to the number of interventions a particular case has on the decision process. Sometimes a case that is marked for being removed has

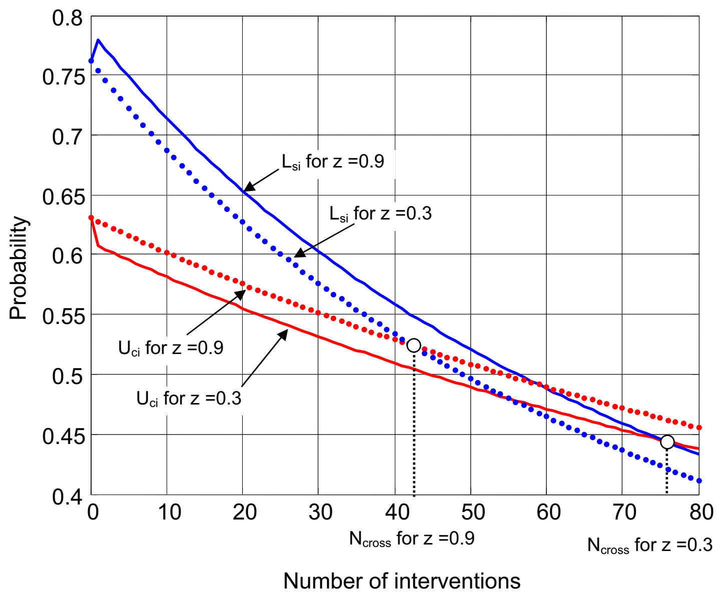


Figure 5.16: Decision of dropping boundaries depending on z

only one or very few interventions. This is not significant enough to know how the case will perform in the future. Hence, a minimum number of interventions n_{min} is needed before removing a case although its confidence index is below the minimum required.

Taking all this into account, a case s_i is removed from the base if it fulfills these conditions:

1. Its lower bound probability L_{si} is below the maximum bound probability of its class U_{ci}.
2. The number of interventions n that the case has done is higher than a certain predefined value n_{min}.

5.3.5 Diagnosis Operation

After this initial training, the system is used to diagnose new cases where the fault is unknown. In these operating conditions, the responsibility of deciding if the case should be retained is given to the user. The diagnosis system shows the user the solution proposed, corresponding to the adapted case obtained from the retrieved ones. In a situation where the extracted neighbors belong to different components (as shown in Figure 5.13) the system shows the weight corresponding to each one, together with the first hierarchy level common to the k retrieved cases. The user makes the decision according to the information provided. For example, recalling again Figure 5.13, but considering that the fault is unknown, Figure 5.17 reproduces the given situation.

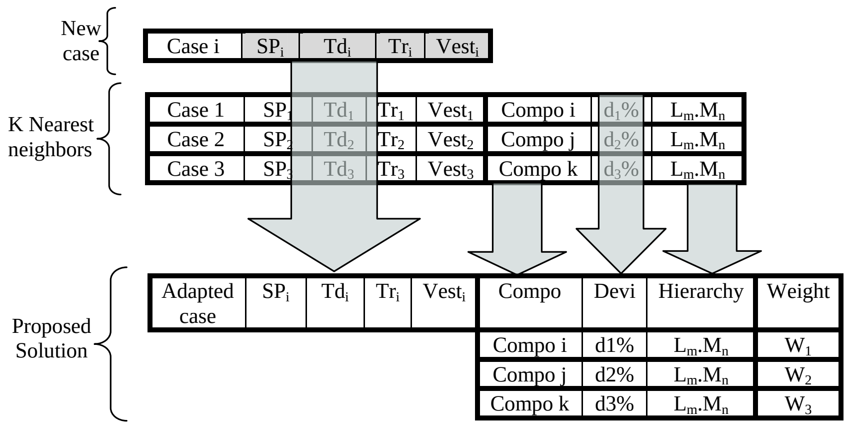


Figure 5.17: Diagnosing unknown faults

Compo i with *Devi* = *d1%* is proposed with a weight W_1. *Compo j* with *Devi* = *d2%* is proposed with a weight W_2 and something similar for the third retrieved case. The weight depends on the distance and it is calculated using equation 5.2. It is up to the user to choose the case with the highest weight as a solution or to select a less precise diagnosis based on the first common hierarchy level Module M_n at level L_m of the extracted cases.

5.4 Results on the Biquadratic Filter

After the description of the CBR-system, it was applied to the biquadratic filter circuit. The case structure and case base hierarchy are defined as previously explained. For the learning process and maintenance, the multi-edit algorithm described in the previous sections is applied. The case revision and retention tasks are performed following the algorithm in Figure 5.14, where the algorithm based on the DROP4 procedure and described in Figure 5.15 plays an important role when making the decision whether to introduce the new case or not. On the other hand, the algorithm similar to the IB3 is used for forgetting noisy exemplars. Let's see some numerical examples demonstrating how the proposed method works.

First of all, let us concentrate on the retaining procedure described in Figure 5.14. Examples corresponding to each type of decision considering $k = 3$ neighbors to extract are displayed. The weights are calculated using the exponential kernel with $w_k = 0.2$.

Consider the results given in Table 5.18, where the measures are normalized.

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.5099	0.5152	2.2353	1.3024	4	1	-54.6410	-
Neighbor 1	0.5411	0.4242	2.2059	1.2593	4	1	-54.5577	0.5167
Neighbor 2	0.5162	0.4545	2.2353	1.1525	4	1	-50.00	0.3765
Neighbor 3	0.5546	0.4545	2.2353	1.0466	4	1	-43.1893	0.2000

Table 5.18: Case of a type 1 decision

The $k = 3$ neighbors all correspond to *Compo* 1, the same *Compo* field as the new case. As proposed by the algorithm in Figure 5.14, the average deviation should be obtained and compared to the new case. The average deviation of the 3 neighbors is $Devi = -49.25\%$, comparing this with the deviation of the new case gives an error estimation of 9.86%. Since the error is less than 10%, the case is supposed to be correctly estimated, and therefore it is not necessary to introduce it into the case base.

On the other hand, consider the results given in Table 5.19.

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.5487	0.4545	2.2353	0.4600	1	1	28.0632	-
Neighbor 1	0.4694	0.5152	2.2647	0.4918	1	1	13.7863	0.8224
Neighbor 2	0.5598	0.4545	2.2059	0.5647	33	1	0	0.8216
Neighbor 3	0.5537	0.3333	2.2059	1.3462	4	1	-57.877	0.2000

Table 5.19: Case of a type 2 decision

The *Compo* field of the retrieved cases is 1 for all of them, and equal to the *Compo* field of the new case. But, the deviation calculated as the mean of the deviation of retrieved cases is -14.69%, that is far from being 28.063%. This is the situation in a type 2 decision from the diagram shown in Figure 5.14. Hence, the case will be introduced if it does not disturbs others.

An example of situation 3 is given in Table 5.20.

	SP	td	Tr	Vest	Class	Compo	Devi	Weight
New case	0.2358	0.9091	2.5000	0.6145	7	3	66.6023	-
Neighbor 1	0.2499	0.9394	2.5294	0.5761	20	5	-50.00	0.3058
Neighbor 2	0.3739	0.8182	2.4118	0.6056	20	5	-41.8863	0.2113
Neighbor 3	0.2369	1.0606	2.6176	0.6033	20	5	-52.3292	0.2000

Table 5.20: Case of a type 3 decision

The 3 neighbors belong to *Compo* 5 while the new case corresponds to *Compo* 3. This is an isolated case surrounded by cases with other *Compo* fields. The case is not introduced.

If the cases that surround a particular new situation correspond to the nominal, this case is not detectable. An example is given in Table 5.21.

	SP	td	Tr	Vest	Class	Compo	Devi	Weight
New case	0.5345	0.4242	2.2353	0.5565	6	2	-15.1905	-
Neighbor 1	0.5309	0.4242	2.2059	0.5767	33	33	0	0.2406
Neighbor 2	0.5554	0.4242	2.2059	0.5493	33	33	0	0.2315
Neighbor 3	0.5162	0.4545	2.2353	0.5763	33	33	0	0.2000

Table 5.21: Case of a type 4 decision

Suppose now, that the situation in Table 5.22 is given.

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.3474	0.8182	2.4118	0.5702	7	2	64.4636	-
Neighbor 1	0.4234	0.7576	2.3529	0.5798	5	2	24.5949	0.6063
Neighbor 2	0.4670	0.6364	2.2941	0.5623	7	2	48.3634	0.3374
Neighbor 3	0.4694	0.5152	2.2647	0.4918	1	1	13.7863	0.2000

Table 5.22: Case of type 5 decision

In this situation there are some retrieved cases, but not all of them have the same *Compo* field as the new case. Neighbor 1 and Neighbor 2 correspond to *Compo* 2, giving an average deviation of $Devi = 36.48\%$. It is clear that the new case is *Compo* 2 but with a deviation far from the proposed one. Hence, the case will be introduced, if it does not disturb the other case base members.

It can happen that some of the neighbors have the same *Compo* field, and the average deviation produces a result with an error less than 10% compared with the new one. We are now in a type 6 situation according to the diagram in Figure 5.14. Table 5.23 gives an example.

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.5871	0.2727	2.2059	0.5912	19	5	53.1641	-
Neighbor 1	0.5935	0.2727	2.2059	0.5877	19	5	54.0804	0.6911
Neighbor 2	0.5843	0.2424	2.2059	0.5924	19	5	46.8429	0.2124
Neighbor 3	0.5820	0.2424	2.2059	0.5988	8	3	-46.6631	0.2000

Table 5.23: Case of a type 6 decision

Two neighbors have the same *Compo* field (*Compo* 5) as the new case. The average deviation is $Devi = 50.46\%$, producing an error in the estimation of 5.08%. As the sum of weights of these to neighbors is 0.9035 and bigger than the weight of the neighbor left (0.2000) the new case will not be introduced into the case base.

If the sum of weights of the cases with different *Compo* fields is bigger than the neighbors with the same *Compo* field, the case should be introduced. Table 5.24 shows an example.

In this situation, there is only one neighbor with the same *Compo* field that approximates the new case with an error of less than 10% in the deviation

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.6463	0.4242	2.1765	0.6110	13	4	15.6715	-
Neighbor 1	0.6474	0.4242	2.1765	0.6042	29	8	20.2282	0.8695
Neighbor 2	0.6093	0.3636	2.2059	0.5991	13	4	14.4257	0.2059
Neighbor 3	0.6693	0.4848	2.1471	0.5762	29	8	20.00	0.2000

Table 5.24: Case of a type 7 decision

(7.94% of error), the sum of weights of the other neighbors is 1.0695 which is bigger than the weight of the neighbor with the same *Compo* field (0.2059). Therefore, the new case will be introduced.

And finally, Table 5.25 is a collection of data corresponding to the situation where none of the neighbors with the same *Compo* field are equal to the new case. The new case should be introduced, if it does not disturb its associates.

	SP	td	tr	Vest	Class	Compo	Devi	Weight
New case	0.5328	0.4848	2.2059	0.5838	5	2	12.2771	-
Neighbor 1	0.5162	0.4545	2.2353	0.5763	33	33	0	0.5195
Neighbor 2	0.5321	0.5152	2.2059	0.5046	1	1	16.1068	0.2989
Neighbor 3	0.5162	0.4545	2.2353	0.4802	1	1	20.00	0.2000

Table 5.25: Case of a type 8 decision

The previous situations correspond to a training step that is not at the beginning. It is normal that taking the classic dictionary as the initial set, the first new previously unseen cases would spoil the classification of the other case base members if they were introduced. But, if there is less than k neighbors of that class, the decision will be to retain the new case, if it is necessary. This is the way to ensure that the case base will grow and learn.

The process of training is extremely sensitive to how the new presented cases are sorted. Hence, a method like the well-known ten-fold cross-validation has to be applied. In our case, we have decided to build several independent sets $\{S_1, S_2, ..., S_n\}$ of randomly generated exemplars corresponding to a N number of cases for each component, with faults uniformly distributed between $\pm 70\%$. The CBR-system is trained with several series $\{T_1, T_2, ..., T_m\}$ of these sets randomly sorted in order to obtain a case base that performs better.

$$T_1 = \{S_1, S_2, ..., S_n\}$$
$$T_2 = \{S_4, S_n, ..., S_1\}$$
$$...$$
$$T_m = \{S_{10}, S_1, .., S_n, .., S_3\}$$

Once the revising and retaining process is clear, the following step is to describe how the noisy cases are dropped. This is performed by the algorithm similar to IB3 previously described. Figure 5.18 demonstrates how the system learns while training. It has been obtained for 20 training sets and taking a confidence index of 0.9.

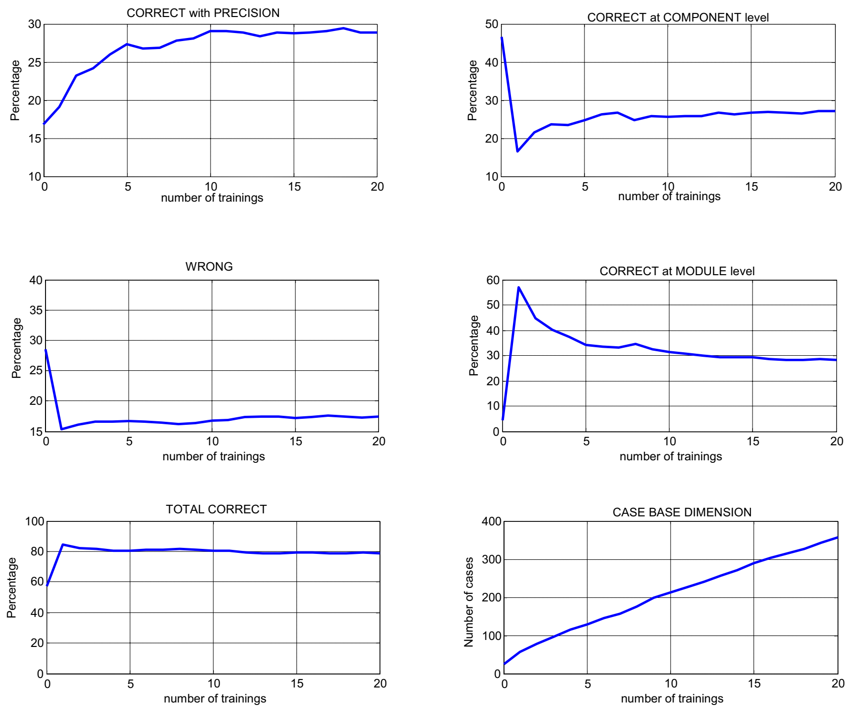


Figure 5.18: Learning process evolution

At the beginning, the total of correct cases increases abruptly. This fact is due to the multiple cases being taken instead of one as a neighbor in the diagnosis. Therefore, it is more probable to extract the correct class between them, but there will be overlapping with the other neighbors. The increase is therefore due to the percentage of the correct module detection rising (second row second column plot in Figure 5.18). As the training advances, the case base substitutes component and module success diagnosis by increasing the average of precision diagnosis. On the other hand, another important factor to observe is that the number of cases continuously increases.

The case base is trained with a bigger number of sets in order to study if there is a saturation of the number of cases introduced or degradation of the system performance is produced. Figure 5.19 displays the results for 250 training sets randomly sorted for training. Observe that it reaches saturation.

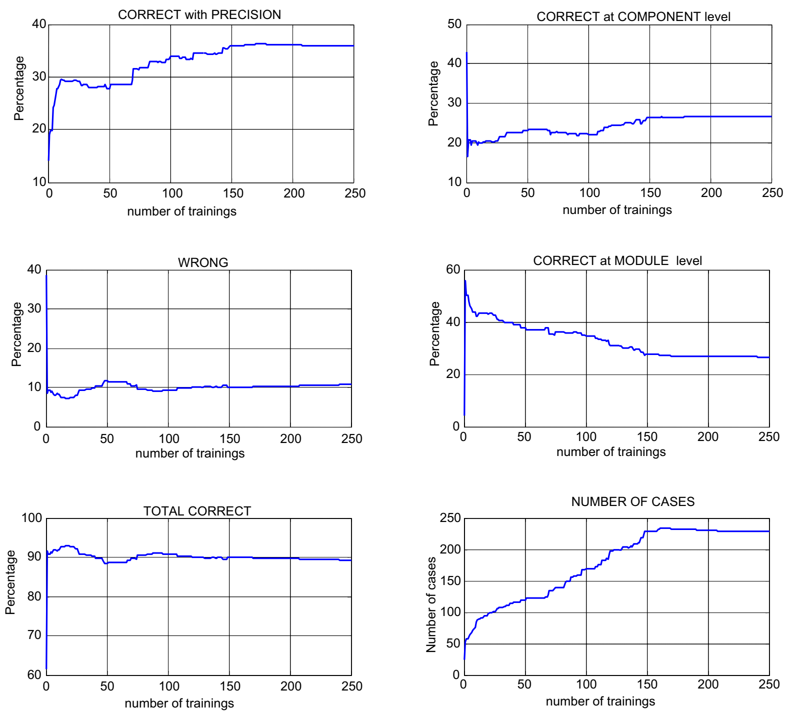


Figure 5.19: Base saturation

Also, it has to be studied which confidence index z and minimum number of interventions n_{min} perform better. For example, for the biquadratic filter, several values of confidence, from 0.3 to 0.9 have been tested, while the minimum number of interventions before removing has been taken from 1 to 10.

Of course the bigger the n_{min} is, the less sensitive the system is to z. The results are shown in Figure 5.20.

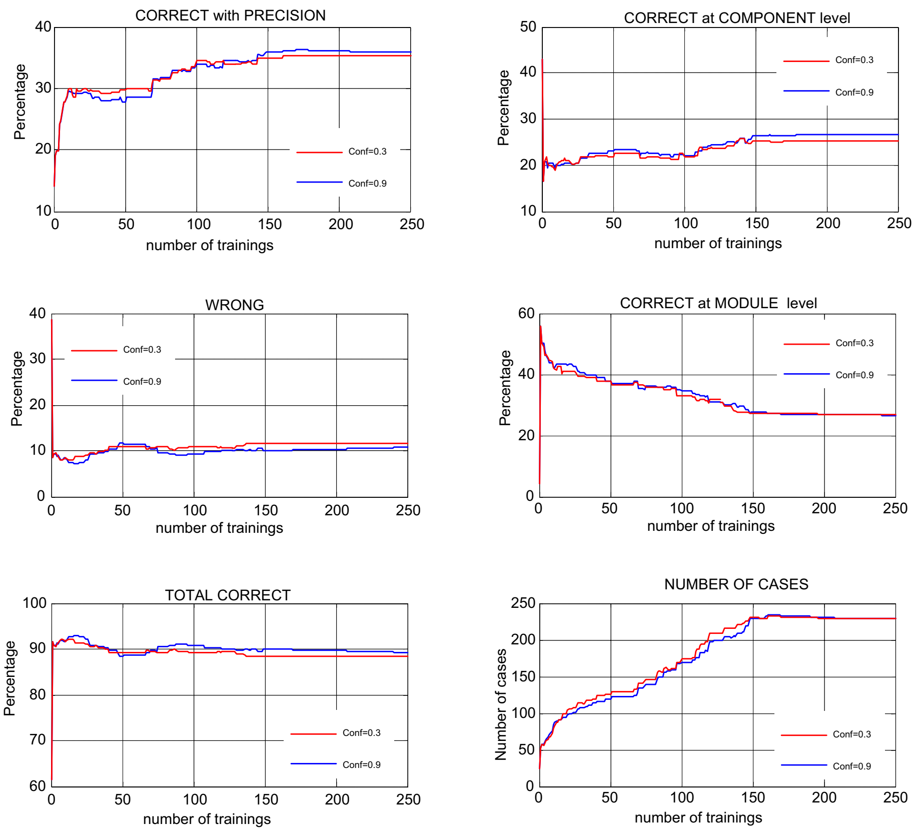


Figure 5.20: Optimal confidence index for forgetting

The maximum performance is obtained for $C \approx 0.9$ as the confidence level to forget cases. Hence it is better for our system to forget irrelevant cases faster. Figure 5.21 shows the evolution of the indexes L_{si} and U_{ci} for some cases during training. These cases were eliminated from the case base, because, in the end their performance fell below the established limit of the selected confidence index.

It can be observed in the previous learning evolution figures that the case base reaches a saturation value for the performance success and for the case base dimension. After the training number 150, there is little variation in the

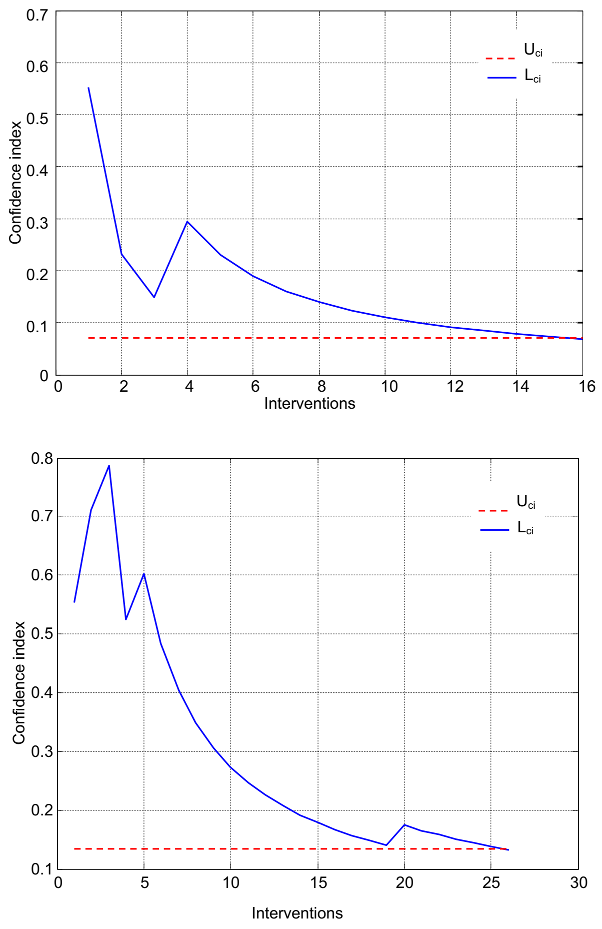


Figure 5.21: Evolution of the confidence index during training

success and the number of cases. Of course, if the training sets are sorted in a different way, the results can change.

Figure 5.22 shows several results obtained from a group of 150 training sets when they were sorted differently.

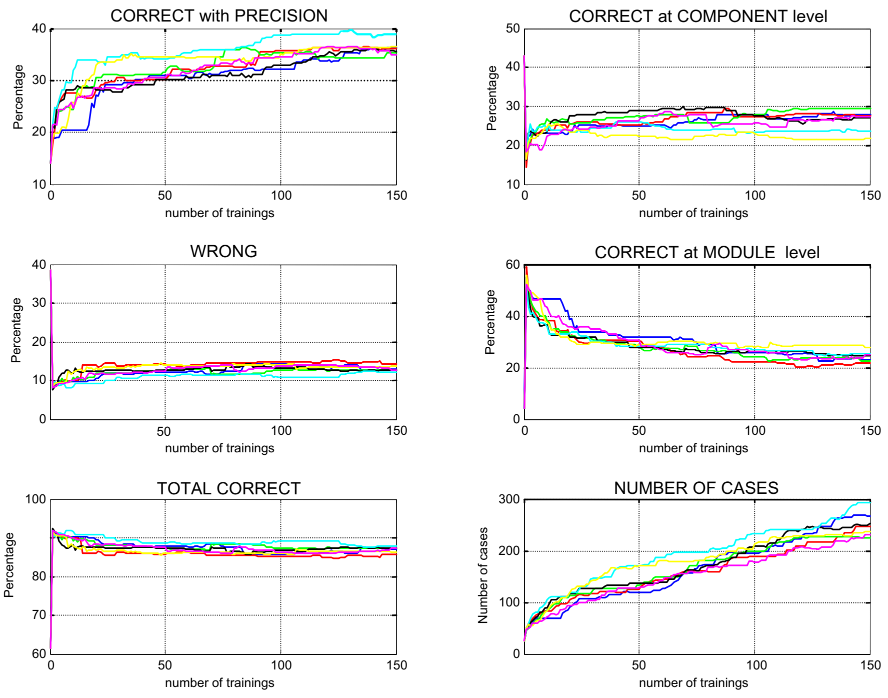


Figure 5.22: Analysis to sort the training sets

Figure 5.22 shows that the training set order probably has the best performance corresponding to the cyan curve, but the results do not differ substantially from the others. Some training orders make the learning process faster at the beginning stage, but the final results for the precision success are all contained between 35% and 40%, and the total success between 85% and 90%, while the case base size has a number of cases between 200 and 300 cases.

For example, taking the color cyan curve, it can be seen that at the training number of approximately 132 the best result when diagnosing correct cases with precision is produced. If the set of cases obtained at this moment is used

Component	OK diagnosed	Component OK	Module OK	Wrong
R1	50%	24%	10%	16%
R2	41%	20%	35%	4%
R3	40%	20%	36%	4%
R4	38%	29%	17%	16%
R5	29%	6%	35%	30%
R6	40%	37%	11%	12%
C1	40%	26%	32%	2%
C2	37%	24%	29%	10%
Average	**39.375%**	**23.25%**	**25.625%**	**11.75%**

Table 5.26: Performance for faults not previously considered. CBR

to diagnose the same test set composed of 100 cases corresponding to each component with uniformly distributed deviations compressed between ±70%, Table 5.26 is obtained.

If the table is compared with the one obtained using the classic dictionary, the diagnosis rate of 39.375% represents an increment of more than 100% (it is 17% for the classic dictionary). Hence, the CBR-system has clearly learnt and it is able to improve the diagnosis of the circuit. Table 5.27 recalls the results obtained by the dictionary using the ramp method.

Component	OK diagnosed	Component Correct	Wrong	Non Detect.
R1	19%	57%	16%	8%
R2	20%	25%	42%	13%
R3	5%	44%	37%	14%
R4	15%	65%	17%	3%
R5	15%	24%	50%	11%
R6	20%	62%	12%	6%
C1	12%	42%	35%	11%
C2	30%	52%	7%	11%
Average	**17%**	**46.375%**	**27%**	**9.625%**

Table 5.27: Ramp dictionary successes for deviations of ±70%

But DROP4 tends to retain border points in order to reduce the number of instances to keep while it preserves the success ratio. Of course, if center points are deleted, diagnosis with less than a certain percentage error will be difficult. Hence, a new refinement in the method is introduced. In spite of the case base size growing, when the algorithm in Figure 5.14 is applied, if a decision to keep the new instances is made, the case will be introduced even if its retention is going to spoil the classification of other cases. Figure 5.23 shows the results obtained after training the case base with the same sets used to generate Figure 5.22. Observe that the number of cases diagnosed with precision has increased up to 50%. Also, the total number of successes (considering correctness at component and module levels) has reached a success rate of almost 90%.

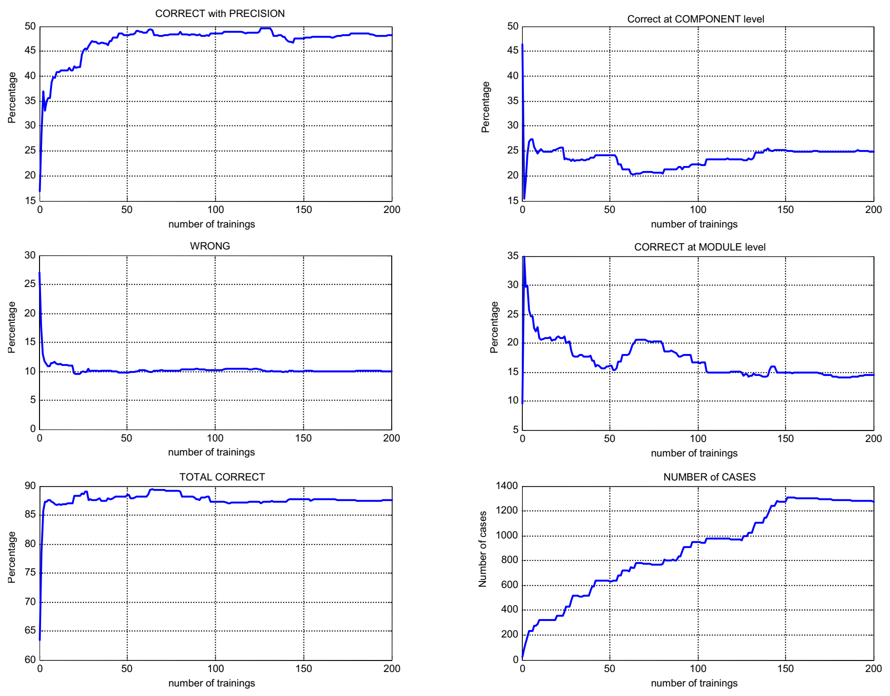


Figure 5.23: Analysis keeping cases that disturb classification of other cases

From these graphics, taking the situation after 63 trainings as an example, the system is able to provide the following diagnosis using a case base size of 742 cases:

49.40%	*successes with precision*
20.25%	*successes at component level*
19.80%	*successes at module level*
10.37%	*diagnosed incorrectly*

The total of successes is 89.45%. The addition of the successes and wrong diagnosis is 99.82%. The rest of the cases to complete the 100% correspond to a 0.18% of non detectable cases. That is, cases that cannot be diagnosed because they are not distinguishable from the nominal situation.

5.5 Conclusions

When applying the CBR system two different usages have to be taken into account: training and testing. For the training stage, several actions have to be considered. First of all, an initial case base has to be selected. It seems that the classic dictionary is a good starting point, although other bases can be selected. Then, the system has to be trained with new exemplars. This can be done by simulation, since it is easier and faster than taking real data. When a new case is given during training it is supposed that the correct diagnosis is known in advance. After the retrieval of the k-nearest exemplars, the algorithm described in Figure 5.14 is applied in order to decide if the new case should be retained or not. If the case has to be kept, an analysis of the convenience of such an action has to be done using the associated concept, as described in Figure 5.15. The metric used is the Euclidean, because for the numerical attributes it is the simplest that produces good results. In case of taking nominal response characteristics as attributes other metrics should be considered.

Also, the confidence indexes of the retrieved cases are updated according to the IB3 algorithm for maintenance purposes. If the confidence index L_{si} of one or some of the recent k retrieved cases decreases below to the confidence index of its class U_{ci}, the case is considered to be performing incorrectly and it is deleted from the case base.

Repeating the cycle for each new exemplar, the case base reaches saturation. There is an equilibrium between the retained and deleted cases; some cases are introduced but approximately the same amount deleted; that is, there is no increasing ore decreasing of the case base size and success percentages. At this point, the CBR-system can be considered to be doing its best. From this point, the case base can be trained and tested with real data coming from the real circuit.

Concerning the test stage, it has a lot in common with the training stage, but with an important difference. In the training phase, the revision task is done automatically, since it is supposed that the correct diagnosis of the new case is known. During a real test stage, this is not true. Only the circuit symptoms are known. The CBR-system will provide a possible diagnosis, retrieving cases and adapting a solution, but the revise and retain steps are not done automatically, since there is no user feedback. Of course, the test process can be stopped and some information on the bad diagnosis done and new data can be used to refine the CBR-system, when necessary. But, again the user has to close the CBR-loop supervising the revise and retain stages.

The proposed CBR-system is based on fault dictionaries, taking advantage of its case structure and provided signatures, and completed with learning and maintenance tasks. Taking the classic dictionary with a faults universe of ±20% and ±50% as an initial case base, produces better results than using a DROP-reduced dictionary. This is because the latter saturates the case base in these faults.

The knowledge is given using a multi-edit technique. That is, an algorithm similar to DROP4 is used to introduce new cases and an algorithm based on the IB3 to delete noisy instances. The associated concept shows good results when used to decide if it is useful for the system to retain a particular case. The algorithm used to forget noisy exemplars has also demonstrated to be adequate. But examples, demonstrate that better diagnosis results are obtained when the cases are introduced when necessary, even if their retention spoils the classification of the cases already contained in the case base.

For the biquadratic filter circuit, the results have shown that the method improves the diagnosis provided by the fault dictionary. For a set of 100 faults for each component corresponding to deviations compressed in the range of ±70%, the classic dictionary has a success rate of 17% while using the proposed multi-edit technique increases it to 39.375%. Also, the incorrect diagnostics decrease from 27% to 11.75%. A transition of success rates from the component success rate to the precise success rate can be observed, meaning that the system tends to be more precise when diagnosing. The increase in the case base size is not dramatic. The classic dictionary has 25 cases while the multi-edit technique produces a case base of 263. The confidence index selected for forgetting is 0.9. This index shows the best results after several simulations. A case that is performing badly can quickly be deleted because it will be substituted by another training case that can perform better. An index of 0.3 means that the system removes cases from the case base more slowly and the results are worse. If the cases are introduced even though they have a bad influence on the classification of other cases already contained in the base, the system performs even better. In this situation, success with precision is increased up to 49.40% using the same test set. The case base reaches saturation as well, but as shown in the previous paragraph, the training can be truncated at training 63, while the system performance is maintained with a case base of 742.

The next chapter shows the application of the method to a real biquadratic filter in order to test its performance with real measures and it explains how to deal with the possible differences from the simulated data.

Chapter 6

RESULTS FROM THE REAL CIRCUIT

6.1 Introduction

In order to show the performance of the proposed method, a real biquadratic filter is built. This circuit is designed to easily provoke the desired faults to be diagnosed. Its measures will include noise and inaccuracies due to the instruments. The idea is to obtain measures from this real circuit and test the performance of the designed CBR-system with data from the previous simulations. It is expected that the difference between real and simulated data will be small, since the circuit is linear and very close to the ideal situation.

The next sections give a short description of the general environment used. After that, the possible sources of error are analyzed and their effect on the measures described. In the end, measures from the real circuit are used to test the proposed CBR diagnosis system.

6.2 The Environment Used

The environment is composed of a computer with a data acquisition card and the biquadratic fault simulation board. The simulation board is connected to the data acquisition card mounted in a computer, where the software developed configures the board and stores the acquired data in a file.

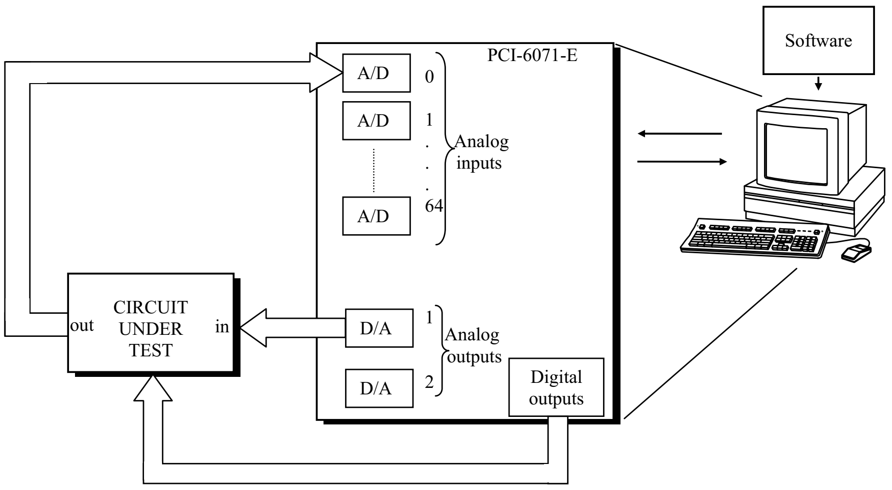


Figure 6.1: Environment

The circuit is specifically designed for test purposes, allowing the universe of faults to be generated in an easy manner. Appendix A describes how the circuit has been set up and the main characteristics are given for the data acquisition board, the PCI-6071-E from National Instruments, used for generating test signals and obtaining the circuit measures.

6.3 Sources of Error

According to (Pallas Areny, 1995), the errors can be produced by controllable or uncontrollable factors. Since the circuit is well isolated (physically) and insulated (electrically), there is no electromagnetic interference and the circuit is not submitted to any air flow that can change its working temperature. Also, the power supply offers great protection from the electrical net fluctuations.

Hence, the uncontrollable errors are negligible and the only errors considered are the controllable ones. In our case, the main controllable factors that can influence the measurements are due to:

- *Sampling time*: The simulations are done using continuous signals. But when the data is acquired from a real circuit, values are only taken at a particular sample time. Hence there could be a difference between a value calculated at a certain instant and the data acquired close to this instant. On the other hand, if the value of interest is based on a measurement of time, there will be a difference due to the discretization of the time when acquiring signals. A part from Shannon's criterion, the sampling time T_s should be small enough to provoke a negligible error in the time measures.

- *Relative accuracy*: This parameter is related to the ADC resolution used by the acquisition board. The ADC number of bits also produces a quantization of the amplitude space. Due to this fact, there will be a quantization noise. If the board has a n bits AD converter, it divides the amplitude space range into 2^n zones. Therefore, there is a maximum error in the amplitude measures of

$$n_Q = \frac{Range}{2^n} \tag{6.1}$$

- *Noise*: The noise considered is thermic noise, produced by the resistors and the semiconductors. Its mean is zero, which indicates that averaging several measures can improve the acquired data. Electromagnetic interferences are avoided if the circuit is built with this in mind (shielding the circuit, building a good ground plane, and so on).

Let us particularize the previous effects on the measures proposed to develop the diagnosis system considering the saturated ramp response method. Concerning the sample time effect, the error affects the parameters t_d and t_r. For example, if the parameter t_d is calculated, the instant of time at which the output signal reaches 50% of the steady state of the output voltage is approximated in a margin given by the sampling time T_s. It is well known that to ensure a good measurement, the sampling time should be taken at least 10 times less than the rise time of the signal to be acquired, as given in equation 6.2.

$$T_s \leq \frac{t_r}{10} \tag{6.2}$$

Since the saturated ramp input has a slope of $1V/100\mu s$, that is, it takes 90 μs to rise from 0.1 V (10% of the steady sate) to 0.9 V (90% of the steady state), the sampling time should be selected at least as

$$T_s \leq \frac{90\ \mu s}{10} = 9\ \mu s$$

But, in the present situation, as we are dealing with time parameters of tens of μs, the sampling time T_s is selected to be $t_s = 1\mu s$, close to the maximum allowed by the acquisition board data used.

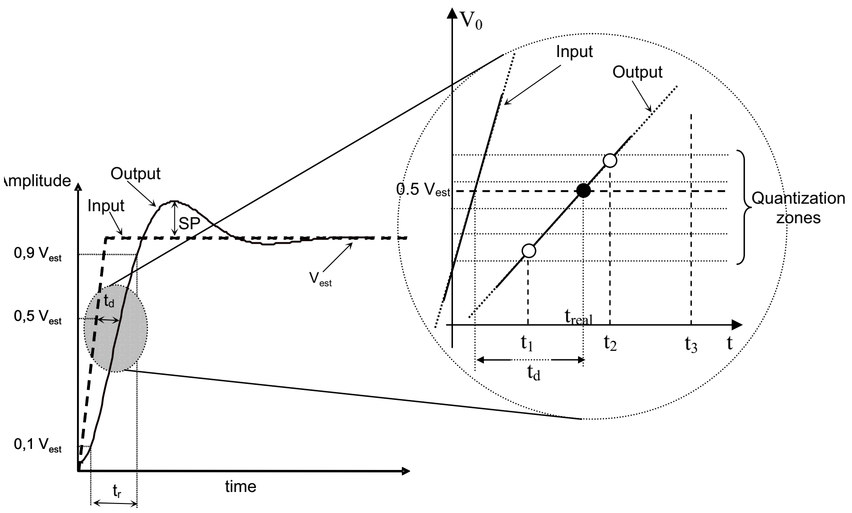


Figure 6.2: The effect of discretizing the time space

As shown in Figure 6.2, although the signal reaches 50% of the steady state at t_{real}, the measurement system only allows t_{real} to be estimated with the sample at t_1 or t_2. Hence, in our case there is a maximum error of $\pm 1\mu s$ on the time measure t_d due to the sampling time. Something similar can be concluded for the measure of the rise time t_r. This fact can affect some results, making the acquired measure differ $\pm 1\mu s$ from the ideal one. Of course, a more in-depth study has to be carried out if more precise knowledge of the influence of this time measurement error on these parameters is to be determined.

On the other hand, simulations indicate that the considered faults produce responses with amplitudes between $\pm 5\ V$. Hence, the acquisition board inputs are configured as bipolar and with a voltage range from $\pm 5\ V$ (10 V of range). According to Table A.1 given in Appendix A, the 12 bits AD converter produces 4096 discretization levels, meaning that the quantization noise is

$$n_Q = \frac{Range}{2^n} = \frac{10}{4096} = 2.4\ mV$$

If dither is used to reduce the effect of quantization noise, the expected maximum output deviation due to this factor is reduced to

$$n_Q = 2.4\ mV \cdot 0.5 = 1.2\ mV$$

The question now is how many measures are necessary to reduce the effect of noise. For less than $n = 200$ samples and with an unknown standard deviation of the measures, s, the confidence interval of the measure can be calculated as (Pallas Areny, 1995)

$$\hat{x} \pm t_{1-\alpha} \cdot s/\sqrt{n} \tag{6.3}$$

where $\hat{x}$ is the mean of the measure, n the number of measures, s the standard deviation and $t_{1-\alpha}$ the value used for calculating the confidence interval at the α confidence level. This value is given in tables in (Pallas Areny, 1995) and in (Kokoska and Nevison, 1992). For example, if $n = 10$ measures are taken for a particular magnitude, the confidence interval required to have a $\alpha = 99\%$ probability of this interval including the correct value of the measured magnitude is given by

$$\hat{x} \pm 3.250 \cdot s/\sqrt{10} = \hat{x} \pm 1.028 \cdot s$$

where $t_{1-99\%} = 3.250$ is taken from the mentioned tables. We can look at an example using the biquadratic filter in a nominal situation in order to estimate the number of measures necessary to reduce the error when averaging them. Figure 6.3 depicts 5 consecutive measurements when all the components stay at their nominal value. The acquired signals are zoomed to show the noise effect.

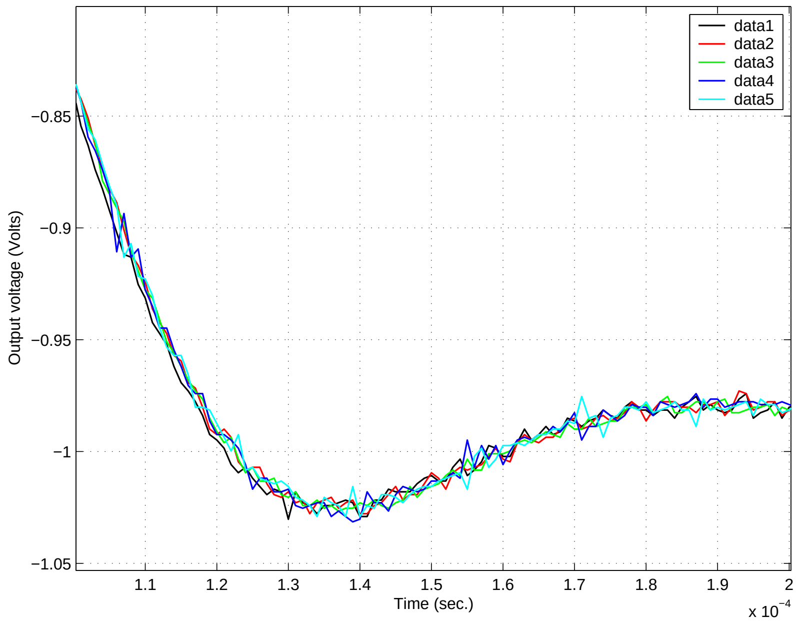


Figure 6.3: Acquired signal with several ramp responses

Table 6.1 shows the interval confidence obtained for the saturated ramp parameters when increasing the number of measures taken. Observe that the highest dispersion is produced for the parameter SP. The interval calculated in the last column corresponds to this parameter, and is much narrower for the rest of the parameters.

The same case simulated has $SP = 4.08\%$, obtained from $V_{max} = -1.0254$ and $V_{est} = -0.9852$. If the quantization noise $n_Q = 1.2mV$ is taken into account, the worst expected overshoot SP would be

Measure Number	SP (%)	t_d (μs)	t_r (μs)	V_{est} (Volts)	Average	Standard deviation	$t_{1-99\%}$	Confidence Interval for SP
1	4.2079	16	76	-0.9863	-	-	-	-
2	5.1058	16	76	-0.9802	4.656	0.4030	63.66	4.656 ± 18.14
3	5.2105	17	76	-0.9851	4.508	0.2672	9.92	4.508 ± 1.53
4	4.2131	17	75	-0.9863	4.588	0.2034	5.84	4.588 ± 0.59
5	4.8268	17	76	-0.9839	4.688	0.2022	4.60	4.688 ± 0.41
...	...				...			
15	4.2131	17	75	-0.9839	4.454	0.1313	2.97	4.454 ± 0.10
...	...				...			
60	4.3317	17	75	-0.9851	4.330	0.1079	2.664	4.330 ± 0.03

Table 6.1: Confidence interval for the overshoot SP

$$\text{SP}= \frac{\left(1.0254+1.210^{-3}\right)-\left(0.9852-1.210^{-3}\right)}{\left(0.9852-1.210^{-3}\right)} = 4.3293\%$$

Therefore, the estimated SP value after averaging several acquired signals would have to be closer to the simulated value than this value. This ensures that the error introduced in the measures by the noise is filtered. Looking at Table 6.1, this happens after averaging 60 acquired ramp responses, approximately. At this point, the maximum error for this particular measure corresponds to a $0.0371/4.330 \cdot 100 = 0.86\%$. Figure 6.4 shows the raw data of five acquisitions compared to the averaged and the simulated nominal situations. Observe that the averaged waveform is quite a good approximation of the simulated one.

Something similar can be done for faulty situations. For example, Figure 6.5 depicts the waveform obtained by simulation and the one derived after averaging 60 waveforms acquired from the real circuit when there is a fault $R_3 - 33.8\%$. On the other hand Figure 6.6 shows the same comparison for a fault $R_5 + 25\%$

In conclusion, the 60 averaged measures are very close to the simulated responses. Hence, when considering real data, great differences from the simulated results are not expected. The 1.2 mV of possible error on the voltage measure due to the quantization error and the error of $1\mu s$ in the time measures due to the sampling time are expected to have little influence on the results. But a more in-depth analysis must be done for each particular circuit

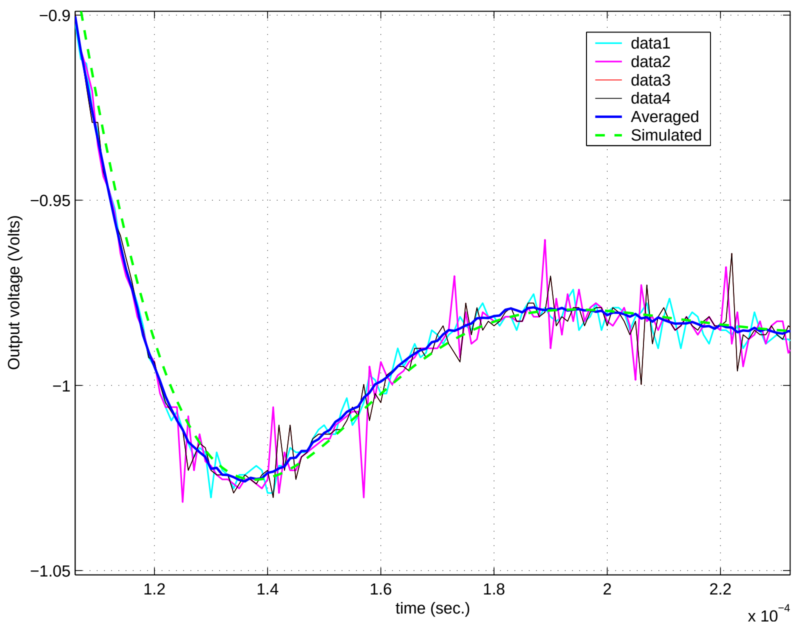


Figure 6.4: Comparing raw data with simulated and averaged waveforms

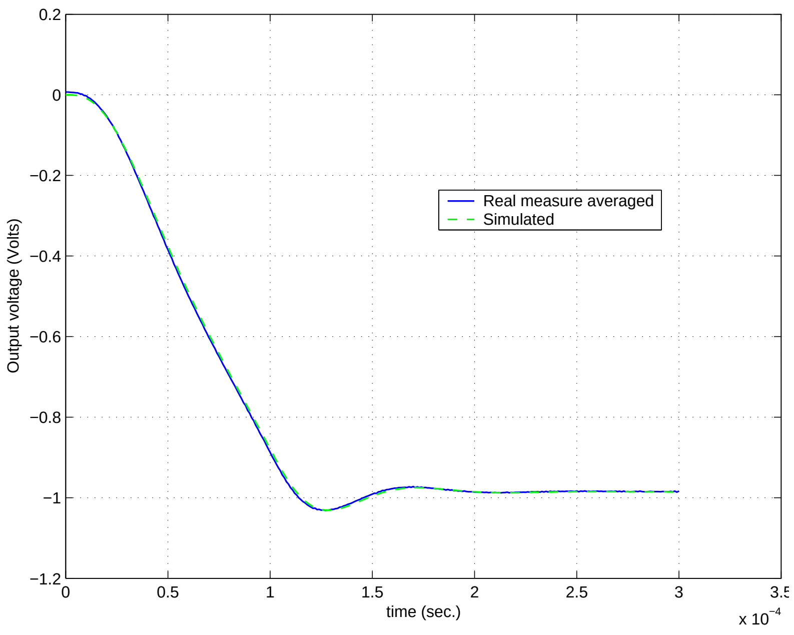


Figure 6.5: Simulated and averaged waveforms for a fault $R_3 - 33.8\%$

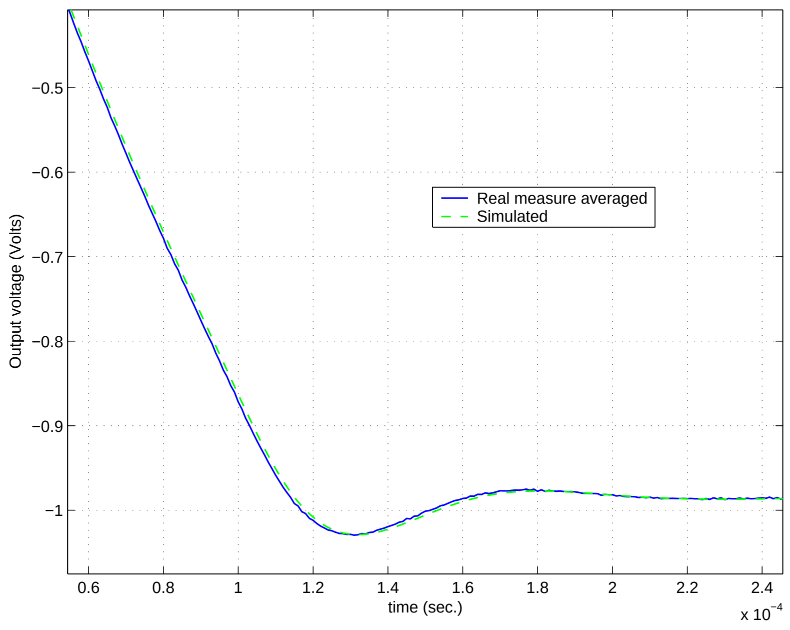


Figure 6.6: Simulated and averaged waveforms for a fault $R_5 + 25\%$

and the considered universe of faults. A fault that is correctly diagnosed in the simulation process can be classified incorrectly because it is very close to the border of the domain of influence belonging to another fault.

6.4 Numerical Results

In order to show the difference between the performance of the system using real data instead of simulated responses, some examples are given. Consider, for example, the case where R_1 is faulty with a deviation of $R_1 + 44.44\%$. The closest class is $R_1 + 50\%$ ($Class = 3$). The simulated measures give the following parameters for the output response:

$$[\mathrm{SP}, t_d, t_r, V_{est}] = [4.08,\ 16\mu s,\ 76\mu s,\ -0.6850]$$

Using as a case base the set of cases obtained in Chapter 5 after 63 trainings, the $k = 3$ closest extracted neighbors are shown in table 6.2.

	Class	Compo	Devi	Weight
New case	3	R1	44.44	-
Neighbor 1	3	R1	42.41	0.2751
Neighbor 2	3	R1	42.32	0.2437
Neighbor 3	3	R1	48.81	0.2000

Table 6.2: Retrieved neighbors for the simulated case $R_1 + 44.44\%$

Applying the decision algorithm of Figure 5.14, the conclusion will be decision number 1, since all retrieved cases correspond to component R_1 and the mean average

$$Devi = \frac{(42.41\% + 42.32\% + 48.81\%)}{3} = 44.51\%$$

Hence, the proposed solution is $R_1 + 44.51\%$, that is, a diagnosis with an error of 1.6% in the deviation.

Provoking the same fault in the real circuit, and averaging the output response using 60 measures, the saturated ramp response parameters obtained are:

$$[\text{SP}, t_d, t_r, V_{est}] = [4.14,\ 16\mu s,\ 75\mu s,\ -0.6818]$$

and the extracted $k = 3$ closest neighbors are given in Table 6.3.

	Class	Compo	Devi	Weight
New case	3	R1	44.44	-
Neighbor 1	3	R1	42.32	0.2279
Neighbor 2	3	R1	42.41	0.2120
Neighbor 3	3	R1	48.81	0.2000

Table 6.3: Retrieved neighbors for the real case $R_1 + 44.44\%$

Observe that the retrieved cases are exactly the same as the ones extracted for the simulated fault, although there are certain differences in the distances and the neighbors are sorted slightly differently. But the conclusion is the same, giving the same diagnosis as when using the data from the simulation.

Consider now the case where R_5 is faulty with a deviation of $R_5 + 25\%$. The simulated measures give the following parameters of the output response:

$$[\text{SP},t_d,t_r,V_{est}]=[4.424,\ 12\mu s,\ 75\mu s,\ -0.9852]$$

Retrieving the three most similar cases from the case base, the neighbors of Table 6.4 are obtained.

	Class	Compo	Devi	Weight
New case	17	R5	23.9	-
Neighbor 1	6	R2	-14.7979	0.5378
Neighbor 2	6	R2	-14.3477	0.2783
Neighbor 3	6	R2	-16.7959	0.2000

Table 6.4: Retrieved neighbors for the simulated case $R_5 + 23.9\%$

Observe that this situation belongs to a decision of type 3, according to the decision diagram shown in Figure 5.14. The new case is completely surrounded by cases with the same *Compo* field between them, but different from the *Compo* field of the new case. Hence, the system is going to give an incorrect diagnosis.

Doing the same for the real circuit, and averaging the output response using 60 measures, the obtained parameters are:

$$[\text{SP},t_d,t_r,V_{est}]=[4.619,\ 12\mu s,\ 74\mu s,\ -0.9840]$$

The closest retrieved neighbors are depicted in Table 6.5.

	Class	Compo	Devi	Weight
New case	17	R5	23.9	-
Neighbor 1	6	R2	-14.7979	0.3881
Neighbor 2	6	R2	-14.3477	0.2631
Neighbor 3	6	R2	-16.7959	0.2000

Table 6.5: Retrieved neighbors for the real case $R_5 + 23.9\%$

The situation does not differ from the one derived with the simulated values.

If a deviation of -33.8% from the nominal case is taken as fault for R_3, the simulation produces the neighbors of Table 6.6.

	Class	Compo	Devi	Weight
New case	6	R3	-33.80	-
Neighbor 1	6	R3	-30.05	0.4992
Neighbor 2	19	R5	41.21	0.4489
Neighbor 3	17	R5	25.57	0.2000

Table 6.6: Retrieved neighbors for the simulated case $R_3 - 33.8\%$

Since the weight associated with the first neighbor is the highest one (the other two cannot be added since they belong to different classes), the proposed diagnosis is $R_3 - 30.05\%$. The component is correct, but there is an error of 11% estimating the deviation.

For the real circuit the closest neighbors are given in Table 6.7

	Class	Compo	Devi	Weight
New case	6	R3	-33.80	-
Neighbor 1	6	R3	-30.05	0.2274
Neighbor 2	17	R5	33.77	0.2112
Neighbor 3	19	R5	46.44	0.2000

Table 6.7: Retrieved neighbors for the real case $R_3 - 33.8\%$

The conclusion drawn is the same as that drawn from the simulated faults. Hence, comparing the averaged measures with the simulated ones, the results obtained are very similar. Of course, there will be circuits and situations where this is not going to happen. Then, it is possible to train the CBR-system with real data in order to learn from real situations.

A set of 100 faults for each component has been generated, and the responses that each one produces are captured. After choosing the data base that seems to perform better, the diagnosis results obtained are shown in Table 6.8.

Recalling that the simulation results for the same test set are

Component	OK diagnosed	Component OK	Module OK	Wrong
R1	56%	20%	10%	12%
R2	43%	28%	16%	12%
R3	46%	21%	21%	7%
R4	62%	19%	8%	7%
R5	24%	2%	21%	19%
R6	55%	24%	7%	12%
C1	43%	26%	20%	9%
C2	61%	21%	10%	4%
Average	**48,75%**	**20,13%**	**14,13%**	**10,25%**

Table 6.8: Performance of the CBR system with real data

49.40%	*successes with precision*
20.25%	*successes at component level*
19.80%	*successes at module level*
10.37%	*diagnosed incorrectly*

As can be observed, there is not a great difference between the diagnosis using simulated and real data. If several different randomly generated test sets, with 100 faults per each component, are used to compare the performance of the system using simulated and real data, the results in Table 6.9 are obtained.

Test number	Simulation				Real measures			
	OK	**Compo**	**Module**	**Wrong**	**OK**	**Compo**	**Module**	**Wrong**
1	49,40	20,25	19,80	10,37	48,75	20,13	14,13	10,25
2	46,87	25,37	10,75	10,12	47,37	24,50	10,25	10,75
3	46,75	26,00	9,25	11,25	48,00	24,12	10,12	11,12
4	48,12	23,00	10,50	11,25	49,25	22,37	10,25	11,37
5	45,75	26,12	11,50	10,75	46,00	25,00	11,50	11,12
6	48,00	21,50	13,50	10,75	46,00	21,87	15,50	11,25
7	47,87	23,62	11,62	10,12	45,62	24,62	13,25	10,37
8	51,25	20,87	11,37	10,25	50,50	22,25	11,12	10,37
9	46,50	25,12	10,50	11,62	45,12	24,62	11,37	12,62
10	47,62	24,50	11,25	10,75	47,12	23,75	11,37	12,50
Average	**47,81**	**23,64**	**12**	**10,72**	**47,37**	**23,32**	**11,89**	**11,17**

Table 6.9: Comparing the simulation with real data performance

Observe that there is only a slight difference between the averages corresponding to the simulated and real data, as was expected.

In conclusion, if the range of the inputs and the sample time of the acquisition board are adequately selected, and the number of measures to average is properly chosen, the measures do not differ excessively from the simulated data. Also, a specific study should be carried out for each particular circuit in order to determine if the sampling time and the quantization noise magnitude could produce any misclassification.

Chapter 7

DISCUSSION AND SUMMARY

7.1 Discussion and Summary

This thesis deals with the diagnosis of analog electronic circuits. A review of the literature, show that in recent years diagnosis of analog electronic circuits has taken on importance once again, after having been pushed aside by digital circuits and the importance they had taken on. Even though circuits are becoming more and more digitalized every day, there is still a small part of analog circuits that makes the process of testing difficult and expensive. The integration of circuits dramatically increases the expectations of the electronic world, but it also makes the test stage extremely complicated. DFT techniques, including the standard IEEE 1149.4, attempt to simplify the problem but, despite the effort, it is far from solved. Also, the literature shows that in recent years, the interest in AI techniques for diagnosing analog electronic circuits has increased. The appearance of a standard, the IEEE 1232, reinforces that idea. Therefore, the design of a new methodology for analog circuit diagnosis makes sense. This thesis makes two specific proposals: The first is based on fuzzy logic and the second develops a methodology to build a CBR-system for analog electronic circuit diagnosis.

It is common to use particular circuits as benchmarks to compare results. The biquadratic filter is one of the circuits used extensively to test the performance of diagnosis systems on analog electronic circuits and is used in this thesis for the same purpose. Its values have been adapted to allow our laboratory test equipment to perform the diagnosis proposed.

There are plenty of methods proposed for analog electronic circuit diagnosis. Chapter 2 presents a general classification of these methods: fault dictionaries, fault verification, parameter identification, approximation techniques and AI techniques. Included are some examples showing the philosophy and the bases of these methods.

As fault dictionaries are the more extensively used technique for detecting and locating faults, they are the base line of the present thesis. Chapter 3 describes some of them in detail, giving examples on the DC and the AC domain. Above all, special attention is paid to two of them: the saturated ramp method, based on a temporal response, and the frequency method, based on taking certain gains and phases at particular calculated optimal frequencies. The results show that in spite of only taking measures at the output node, the percentage of faults located at the output is high, even when tolerances are considered. The results using the saturated ramp or the frequency method are not very different. Hence, the saturated ramp can be used instead of the frequency method, because it needs to keep less measures, in general, while maintains the performance. Other types of circuits will probably need other types of measures to provide sufficient information to make the cases different enough. The problem with fault dictionaries is when non-previously simulated faults are presented and the rate of success decreases.

A first solution to this problem is given in Chapter 4 using an AI technique based on fuzzy logic. The measures used for generating the dictionary are taken as fuzzy inputs. Each membership function is related to a fault considered in the universe of faults taken to generate the dictionary. The membership function shape is related to the obtained distribution measure when a Monte-Carlo simulation is performed for each fault and taking tolerances into account. After approaching the shape with a triangle or a Gaussian function, the results obtained are not very different. The AND function and the centroid method have proven to be the best functions for combining the membership functions and defuzzificate the output. The syntax of the rules is quite simple to derive. There are as many outputs as components to diagnose, each of them with Gaussian-shaped membership functions corresponding to the same faults considered in the fault dictionary (±20%, ±50% and the nominal situation, for example). The fuzzy system performs quite well when locating faulty components, but not when estimating the real value of the components. Also, the problem of diagnosing non previously considered faults still remains to be solved.

Admitting more cases in a fault dictionary can be seen as a natural development towards a CBR system. Hence, Chapter 5 develops a Case-Based Reasoning system from the previous fault dictionaries. Taking advantage of the Monte-Carlo simulations, the dictionary is expanded to cope with the tolerance effect. This chapter also studies the minimum number of representative cases necessary for a particular fault. Thanks to the Gaussian distribution of the measures obtained from the biquadratic filter, the statistical properties of the normal distribution can be applied. The conclusion is that, for this circuit, a set of 500 cases for each class approximates quite well 99% of the population with 99% confidence. Something similar can be done for other types of circuits. It should be taken into account that if the probability distribution of the given instances for each measure and fault differs from the Gaussian shape, this estimation contains an error. Hence, a dictionary of 16500 cases is obtained for the filter under test.

As the cases are generated randomly, noisy cases are kept as well, although they increase the dictionary size considerably. Therefore, reduction techniques such as DROP4 and IB3 are applied to reduce the dictionary size while keeping or even improving its efficiency. These methods are explained in detail because they will be used later on in the learning procedure. The results comparing the dictionary of 16500 cases with the reduced one with the DROP4 algorithm shows that DROP4 improves the success average with less cases, even when using IB3. But as dictionaries, their performance decreases for non-previously considered faults, and they are not able to learn from new situations.

At this point, at the core of Chapter 5, the CBR-system is designed. The same structure used by the dictionary is used for cases, but with a slight modification. The case has two parts. One of them contains information on measures, exactly the same as the dictionary, and the other contains qualitative information: The label corresponding to the closest class, the component, the deviation that produces that particular fault and the hierarchy information. A method that divides the circuit into blocks is applied, creating a hierarchy. Therefore each component belongs to a superior level block, and depending on the circuit size, this block can belong to other superior blocks in the hierarchy. The influence of the circuit hierarchy information on the results has to be studied more carefully in future versions.

The retrieval part is done extracting the k closest neighbors ($k = 3$ for the biquadratic filter). The metric used is the Euclidean, where the part of the case corresponding to the measures is used. The adaptation process is carried out using the qualitative part of the k extracted cases, following the algorithm proposed in Chapter 5.

The learning task is performed by a mixture of the DROP4 and IB3 algorithms. A DROP4-like algorithm is used to decide if a new case should be introduced to the case base. It presents eight possible situations and shows how they are solved, deciding when a new case should be introduced. But DROP4 is too restrictive for diagnosis, since it tends to preserve border points instead of the central ones. This fact means that the deviation of several new cases is diagnosed with an error which is higher than the desired one. Hence, it is decided that a case will be introduced in the case base when the algorithm described in Chapter 5 proposes it, even when the new instance spoils the classification of the cases already contained in the case base.

On the other hand, an IB3-like algorithm decides when to forget a case that is performing wrong. For the biquadratic filter, a confidence index of 0.9 for forgetting cases seems to provide the best final results. But, it is not clear if this value can be extrapolated to other circuits. It should be studied for each particular situation, although it is expected that the same conclusion will be obtained.

The results obtained by the proposed CBR method provide improvement on the results with a case base of relatively moderate size, and with the advantage that the system can learn from new situations. The learning graphics show that, after 250 trainings the system is clearly saturated. Furthermore, stopping at 150 trainings can save a lot of time, with little affect on the final results. This is again the case of the biquadratic filter. For other circuits this can happen earlier or later. It is supposed that larger circuits will need more training and the retention of more cases, while for small circuits less trainings will be enough to describe the circuit behavior and the saturation would be reached earlier.

As is usual with the training processes, the best order in which to present the new situations to the case base for learning is not clear. Results from the system demonstrate that, depending on the order of training, the case base can learn faster, although the final results are not extremely different. Therefore, a set of training sets was generated and then sorted randomly, and the best combination of several trainings were selected to obtain the final case base. Now this case base is ready to learn from a real circuit, with real data.

The method can also be extrapolated to non-linear circuits, or even to different systems that are not electronic circuits. In case measures at more nodes are necessary for ambiguity reduction, they can easily be included by simply enlarging the part of the case corresponding to the measures. The main advantage of the method is that the case base can be generated by simulation, but without necessarily having the transfer function. The simulation allows hundreds of combinations to be tested in a relatively short time, and then the system can be fine tuned using real data.

Finally, some general remarks to be considered when applying this methodology to any analog circuit are summarized:

- First of all, the type of fault dictionary has to be selected according to the kind of circuit to be diagnosed. If the circuit is frequency dependent, the saturated ramp and the frequency methods described in Chapter 3 can be used. Otherwise, other types of measurements should be made to properly represent the faults.
- The classic fault dictionary with deviations of $\pm 20\%$ and $\pm 50\%$ from the nominal value is used as the initial case base, since the initial fault coverage is wide enough for dictionaries. Also, a study of ambiguous cases has to be done in order to group them when diagnosing. For large circuits, it will be necessary to take measures at other nodes, besides the output, if ambiguities are to be reduced.

- The structure of the cases that constitute the case base of the CBR-system is made up of two parts that are independent from the size and complexity of the circuit to be diagnosed: one of them corresponds to the measurements and the other to the fault characteristics. The former will be longer for large and complex circuits, since more measures will be needed to characterize its behavior. The latter is composed of the closest class, the faulty component, the deviation that the component has from the nominal value, and the hierarchy that contains information about the module and level to which the component belongs. For a simple circuit, hierarchy decomposition does not make much sense, but for large ones, their decomposition can help to localize the fault, at least, at the module level. The hierarchical decomposition can be done using the algorithm proposed in (Sangiovanni-Vicentelli *et al.*, 1977) and detailed in Appendix B.

- The system has to be trained in order to cope with non-previously considered faults. The training is done by randomly sorting new situations and studying their diagnosis. The order in which these new cases are presented to the CBR-system tremendously affects the performance of the system. Therefore, it is necessary to carry out several series of trainings in order to see which one performs better. The number of trainings per serie depends on the circuit complexity. A more complex circuit is expected to show a percentage of success saturation with a bigger case base than a simpler circuit, since more cases will be necessary to model the circuit behavior.

- The retrieval process is done using the Euclidean distance. For numerical attributes it shows good results. The number of neighbors, k, to be extracted has to be studied in other circuits since the information obtained from the experiments made is not conclusive enough to affirm that $k = 3$ neighbors is the best choice. Something similar can be said for the weights associated to each retrieved case; the exponential kernel with a weight of $\omega_k = 0.2$ associated to the farthest extracted case, seems to be a good choice to help in the circuit diagnosis, although further studies would be desirable in order to corroborate this conclusion in other analog circuits.

- The proposed case adaptation is transformational and the method proposed in Chapter 5 seems to be adequate for other types of analog circuits as well.

- New case retention while training is decided by the algorithm proposed in Figure 5.14. Although not demonstrated, it is expected that for other analog circuits this retaining process would be adequate. In any case, a more in-depth study of other machine learning methods should be done.

- Forgetting cases that perform incorrectly is done by the IB3-like algorithm detailed in Chapter 5. The confidence index taken is $C = 0.9$, which means that the system easily forgets cases with poor performance. This index has demonstrated quite good results for the biquadratic filter. But these results can be extrapolated to any other analog circuit since easily forgotten cases that do not help in the diagnosis of new situations are going to be quickly replaced by newly accepted cases.

- It has also been proved that simulation can be used to build the proposed CBR-system case base before using the real circuit. It saves a lot of time and allows the user to test the system performance in advance, and no great differences are expected when the diagnosis system is used with a real circuit. But, a good measurement system has to be ensured. For the biquadratic filter it has been demonstrated that by averaging 60 acquired signals, the results are quite similar to those obtained by simulation. For other analog circuits, a similar analysis has to be done in order to derive the minimum number of measures necessary that have to be averaged to eliminate the effect of noise. The majority of analog circuits have the same noise probability distribution and, therefore, the average of 60 acquired signal will be sufficient to compensate for these real circuit effects if the same acquisition board is used to obtain the measures.

- Once the system is trained, when new unknown faults have to be diagnosed, the user decides whether to keep the new case or not.

7.2 Future Work

Several unresolved issues are in need of further research. Some improvements are possible, all of which require further feasibility and performance analysis. To begin with, the method has been successfully tested on a typical benchmark for analog circuits, the biquadratic filter, which is linear and simple; it should be tested on larger and on non-linear circuits. In addition, the components diagnosed are only the passive ones; in future versions diagnosing faults in active components and integrated circuits should be included as a mandatory task.

Finally, taking more measures at different nodes to reduce ambiguity can be implemented as well. It is expected that making the case structure larger in its measures part will improve the results.

Concerning the CBR system, a way of finding the optimal number of neighbors k to extract for the diagnosis should be further investigated. In the present thesis it has been done empirically, testing values from 29 to 3, and choosing the one that provides the best performance. Of course this procedure is slow and takes a lot of time for big circuits.

In order to improve the retrieval stage results, weights can be added to the attributes according to their importance on a particular measure. The methods for obtaining local or global weights proposed in (Wettschereck *et al.*, 1997) or in (Aha, 1998) can be cited among others.

The new cases order when training is also of major importance. In this thesis we have used brute force (as is typically done in the data mining field), training with multiple combinations and selecting the one that produces the best results. Therefore, a method to provide the optimal order of cases, of which to give first, should be found, although it will not be easy.

The classic dictionary has been taken as starting point, because it can easily be extended to a CBR-system. Other initial case bases can be explored in order to see if some improvements are obtained in the final results.

The function used for forgetting non-useful cases is based on the IB3 algorithm. The shape of this function (a Bernoulli function) can be changed and an analysis of how other functions perform can be made. Also, the forgetting index value should be studied according to the type of circuit to be diagnosed.

Concerning the number of representative cases corresponding to each fault, the approximation of the measure distribution by a Gaussian distribution function has been used. It is important to study this approximation when the measure distribution is different from the Gaussian, in order to know how this factor can affect the number of representative cases selected, although the circuit complexity should be related to this number of representative cases.

In the end, as hybrid combinations of AI techniques, a study of the combination of the fuzzy system designed in Chapter 4 and the proposed CBR system of Chapter 5 can be considered. As the diagnosis provided by the fuzzy system is not very accurate, it can be used to locate the fault component and the CBR system can provide more information on the probable deviation associated with the component.

Bibliography

Aamodt, A. and E. Plaza (1994). Case-based reasoning: Foundationalu issues, methodological variations and system approaches. *AI Communications* pp. 39–59.

Aha, D. W., D. Kibler and M. Albert (1991). Instance based learning algorithms. *Machine Learning.* **6**, 37–66.

Aha, D.W. (1998). *Feature weighting for lazy learning algorithms.* Norwell MA: Kluwer.

Aha, D.W. and D. Wettschereck (1997). Case-based learning: Beyond classification of feature vectors. *European Conference on Machine Learning* pp. 329–336.

Aminian, F., M. Aminian and H. W. Collins, Jr. (2002). Analog fault diagnosis of actual circuits using neural networks. *IEEE Transactions on Instrumentation and Measurement* **51**(3), 544–550.

Atkeson, C.G., A.M. Moore and S. Schaal (1997). Locally weighted learning. *Artificial Intelligence Review* **11**(1-5), 11–73.

Balivada, A., J. Chen and J.A. Abraham (1996). Analog testing with time response parameters. *IEEE Design and Test of computers* pp. 18–25.

Bandler, J.W. and A.E. Salama (1985). Fault diagnosis of analog circuits. *Proceedings of the IEEE* **73**(8), 1279–1325.

Berenji, H.R., J. Ametha and D. Vengerov (2003). Inductive learning for fault diagnosis. *IEEE International Conference on Fuzzy Systems.* To be submitted.

Boyd, R.R. (1999). *Tolerance Analysis of Electronic Circuits Using Matlab.* Electronics Engineering. CRC Press. ISBN: 0-8493-2276-6.

Brighton, H. and C. Mellish (2001). Chapter1: Identifying competence critical instances for instance-based learners. *Instance Selection and Construction for Data Mining* pp. 77–94.

Calvano, J.V., V. Castro, M. Lubaszewski and A.C. Mesquita (2002). Filters designed for testability wrapped on the mixed-signal test bus. *Proceedings of the 20 th IEEE VLSI Test Symposium (VTS.02)* pp. 201–206.

Capitain, P.H. (1982). Complementary signal sensitivity to component tolerance. *Proceedings of IEEE International Automatic Testing Conference AUTOTESTCON'82* pp. 223–227.

Catelani, M. and A. Fort (2002). Soft fault detection and isolation in analog circuits: Some results and a comparison between a fuzzy approach and radial basis function networks. *IEEE Transactions on Instrumentation and Measurement* **51**(2), 196–202.

Chandramouli, R. and S. Pateras (1996). Testing system on a chip. *IEEE Spectrum* pp. 42–47.

Chantler, M., S Cermignani, K.W. Mathisen and O. Saarela (1996). Selecting model-based diagnostic solutions. In: *DX'96*.

Chatterjee, A. and N. Nagi (1997). Design for testability and built-in self-test of mixed-signal circuits: A tutorial. *Proceedings of the 10th International Conference on VLSI Design* pp. 388–392.

Chatterjee, A., B.C. Kim and N. Nagi (1996). DC built-in self-test for linear analog circuits. *IEEE Design and Test* pp. 26–33.

Corsi, F., M. Chiarantoni, R. Lorusso and C. Marzocca (1993). A fault signature approach to analog devices testing. *IEEE Transactions on Circuits and Systems* pp. 116–121.

Cota, E. F., M. Negreiros, L. Carro and M. Lubaszewski (2000). A new adaptive analog test and diagnosis system. *IEEE Transactions on Instrumentation and Measurement* **49**(2), 223–227.

Cunnigham, P. and B. Smyth (1994). A comparison of model-based and incremental case-based approaches to electronic fault diagnosis. *Proceedings of the 12th National Conference on Artificial Intelligence in Case-Based Reasoning.*

Cunningham, P., B. Smyth and A. Bonzano (2003). An incremental retrieval mechanism for case-based electronic fault diagnosis. *Knowledge-Based Systems* **11**(3-4), 239–248.

Dague, P. (1994). Model based diagnosis of analog electronic circuits. *Annals of Mathematics and Artificial Intelligence* **11**(1-4), 439–492.

Dague, Ph., O. Jehl, Ph. Deves, P. Luciani and P. Taillibert (1991). When oscillators stop oscillating. *International Joint Conference on Artificial Intelligence. Sydney. Australia* pp. 1109–1115.

Demuth, H. and M. Beale (2000). *Neural Network Toolbox for Use with Matlab. User's Guide. Version 4.* Mathworks Inc.

Deng, Y., Y. He and Y. Sun (2000). Fault diagnosis of analog circuits with tolerances using artificial neural networks. *The 2000 IEEE Asia-Pacific Conference on Circuits and Systems. IEEE APCCAS 2000.* pp. 292–295.

Duhamel, P. and J.C. Rault (1979). Automatic test generation techniques for analog circuits and systems: A review. *IEEE Transactions on Circuits and Systems* **Cas-26**(7), 411–440.

Fanni, A., A. Giua, M. Marchesi and A. Montisci (1999). A neural network diagnosis approach for analog circuits. *Applied Intelligence 2* pp. 169–186.

Fedi, G., S. Manetti, M.C. Piccirilli and J. Starzyk (1999). Determination of an optimum set of testable components in the fault diagnosis of analog linear circuits. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications.* **46**(7), 779–787.

Fenton, B., M. McGinnity and L. Maguire (2002). Fault diagnosis of electronic systems using artificial intelligence. *IEEE Instrumentation and Measurement* pp. 16–20.

Fenton, W. G., T. M. McGinnity and L. P. Maguire (2001). Fault diagnosis of electronic systems using intelligent techniques: A review. *IEEE Transactions on Systems, Man and Cybernetics. Part C: Applications and Reviews* **31**(3), 269–281.

Ferrario, M.A. and B. Smyth (2000). Collaborative maintenance-a distributed, interactive case-base maintenance strategy. *Advances in Case-Based Reasoning: 5th European Workshop, EWCBR 2000* **LNCS 1898/2000**(ISSN: 0302-9743), 393–405.

Finnie, G. and Z. Sun (2002). Similarity and metrics in case-based reasoning. *International Journal of Intelligent Systems* **17**, 273287.

Gomez, D., F. Lucas, A. Quiros and A. Vizcaíno (1996). Aplicacion de la norma de rastreo periferico. *Automática e Instrumentación* pp. 72–76.

Gupta, K.M. and A.R Montezemi (1997). Empirical evaluation of retrieval in case-based reasoning systems using modified cosine matching function. *IEEE Transactions on Systems, Man and Cybernetics. Part A* **27**(5), 601 –612.

Hatzopoulus, A.A., S. Siskos and J.M. Kontoleon (1993). A complete scheme of buit-in-self-test (BIST) structure for fault diagnosis in analog circuits and systems. *IEEE Trans. Instrumentation and Measurement* pp. 689–694.

Hochwald, W. and J.D. Bastian (1979). A DC approach for analog fault dictionary determination. *IEEE Trans on Circuits and Systems* pp. 523–529.

Hoffmann, C. (2002). A new design flow and testability measure for the generation of a structural test and BIST for analogue and mixed-signal circuits. *Proceedings of the 2002 Design, Automation and Test in Europe Conference and Exhibition (DATE 02)* pp. 197 –204.

Huang, W.H.;Wey, C.L. (1998). Test points selection process and diagnosability analysis of analog integrated circuits. *Proceedings of the International Conference on Computer Design: VLSI in Computers and Processors. ICCD'98.* pp. 582–587.

IEEE1149.4, Standard (1999). IEEE standard for a mixed-signal test bus. *IEEE-SA Standards Board* pp. 1–78.

IEEE1232.2, Standard (2002). IEEE standard for artificial intelligence exchange and service tie to all test environments (AI-ESTATE). *IEEE Standards Board* pp. 1–113.

Jarmulak, J., S. Craw and R. Rowe (2000). Self-optimizing cbr retrieval. *Proceedings of the 12th IEEE International Conference on Tools with Artificial Intelligence.ICTAI 2000* pp. 376–383.

Jurisic, D., N. Mijat and V. Cosic (1996). Frequency domain approach of fault diagnosis of analog filters. *ICECS '96* pp. 1134–1138.

Kac, U., F. Novak, F. Azaïs, P. Novet and M. Renovell (2003). Extending IEEE std 1149.4 analog boundary modules to enhance mixed signal test.. *IEEE Design and Test of Computers* **20**(2), 32–39.

Kaminska, B., K. Arabi, P. Goteti, J.L. Huertas, B. Kim, A. Rueda and M. Soma (1997). Analog and mixed signal benchmark circuits. first release. *IEEE Mixed Signal Testing Technical Activity Committee ITC97* pp. 183–190.

Kokoska, S. and C. Nevison (1992). *Statistical Tables and Formulae.* Springer Text in Statistics. Springer Verlag. ISBN 0 387 96873 -3.

Liu, D. and J. A. Starzyk (2002). A generalized fault diagnosis method in dynamic analogue circuits. *International Journal of Circuits Theory and Applications* **30**, 487–510.

Loftstrom, K. (1996). A demonstration IC for the p1149.4 mixed signal test standard. *Proceedings of the International Test Conference* pp. 92–98.

Lopez de Mantaras, R., and E. Plaza (1997). Case-based reasoning: An overview. *AI Communications* pp. 21–29.

Loveday, G.C. (1995). *Electronic Testing and Fault Diagnosis.* third ed.. Longman Group. ISBN 0-582-25242-3.

Lucas, B. (1996). Automated fault injection speeds resolution of design quality issues. *IEEE Robotics and Autom. Magazine* pp. 58–60.

Manetti, S. and M.C. Piccirilli (2003). A singular value decomposition approach for ambiguity group determination in analog circuits. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications* **50**(4), 477–487.

Milor, L.S. (1998). A tutorial introduction to research on analog and mixed-signal circuits testing. *IEEE Transactions on Circuits and Systems* pp. 1389–1407.

Milor, L.S. and A.L. Sangiovanni-Vicentelli (1994). Minimizing production test time to detect faults in analog circuits. *IEEE Trans. On Computer-Aided Design of IC* **13**(6), 796–807.

Mir, S., M. Lubaszewski, V. Kolarik and B. Courtois (1996). Automatic test generation for maximal diagnosis of linear analog circuits. *Proceedings of IEEE European Design and Test Conference.*

Mohamed, F., M. Marzouki and M.H. Touati (1996). FLAMES: A fuzzy logic ATMS and model-based expert system for analog diagnosis. *Proceedings of the European Design and Test Conference. ED and TC 96.* pp. 259–263.

Murray, B.T. and J.P. Hayes (1996). Testing ICs: Getting to the core of the problem. *IEEE Design and Test* pp. 32–38.

Novak, F. and A. Biasizzo (1994). Enhancing design-for-test for active analog filters by using CLP. *Journal of Electronic Testing* pp. 315–329.

Ozev, S. and A. Orailoglu (2002). Boosting the accuracy of analog test coverage computation through statistical tolerance analysis. *20th IEEE VLSI Test Symposium* pp. 213–219.

Pahwa, A. and R. Rohrer (1982). Band faults: Efficient approximations to fault bands for simulation before fault diagnosis of linear circuits.. *IEEE Transactions on Circuits and Systems* **CAS-29**(2), 81–88.

Pal, S.K., R.K. De and J. Basak (2000). Unsupervised feature evaluation: A neuro-fuzzy approach. *IEEE Transactions on Neural Networks* **11**(2), 366–376.

Pallas Areny, R. (1995). *Teoria Básica d'Errors.* Temes d'instrumentacio electronica. Servei de Publicacions de la UPC. ISBN 84-7653-555-4.

Pan, C.Y. and K.T. Cheng (1995). Pseudo-random testing and signature analysis for mixed-signal circuits.

Pang, J. and J. Starzyk (2002). Fault diagnosis in mixed-signal low testability system. *International Journal of Circuit Theory and Applications* pp. 487–510.

Patterson, D., W. Dubitzky, S. Anand and J. Hughes (1998). On the automation of case base development from large databases. *Proceedings of the AAAI 1998 Workshop: Case-Based Reasoning Integrations* pp. 126–130.

Pous, C. and J. Colomer (2001). Combinacion de metodos para el diagnostico de circuitos electronicos analogicos. *Diagnosis, Razonamiento Cualitativo y Sistemas Socieconomicas. Valladolid* pp. 137–145.

Pous, C., J. Colomer, J. Melendez and J. LL. de la Rosa (2002). Introducing qualitative reasoning in fault dictionary techniques for analog circuits analysis. *QR2002. Sixteenth International Workshop on Qualitative Reasoning.*

Pous, C., J. Colomer, J. Melendez and J.L. de la Rosa (2003*a*). Case base management for analog circuits diagnosis improvement. *Proceedings of the 5th International Conference on Case-Based Reasoning ICCBR03* **Case-Based Reasoning Research and Development**(LNAI 2689), 437–451.

Pous, C., J. Colomer, J. Melendez and J.L. de la Rosa (2003*b*). Fuzzy identification for fault isolation. application to analog circuits diagnosis. *Artificial Intelligence Research and Development. ISBN 1 58063378 378 6. IOS Press* pp. 409–420.

Preist, C., D. Allred and A. Gupta (1992). An expert system to perform functional diagnosis of a bus subsystem. *Proceedings of the Eighth Conference on Artificial Intelligence for Applications* pp. 88–95.

Renovell, M., F. Azais and Y. Bertrand (1998). Optimized implementations of the multi-configuration DFT technique for analog circuits. *Proceedings of Design, Automation and Test in Europe.* pp. 815–821.

Richter, M. (Sesimbra, Portugal, October 25, 1995). The knowledge contained in similarity measures. *remarks on the invited talk given at ICCBR'95.* http://www.cbr-web.org/documents/Richtericcbr95remarks.html.

Salama, A.E., J.A. Starzik and J.W. Bandler (1984). A unified decomposition approach for fault location in large analog circuits. *IEEE Transactions on Circuits and Systems* **CAS-31**(7), 609–622.

Sangiovanni-Vicentelli, A., L. Chen and L.O. Chua (1977). An efficient heuristic cluster algorithm for tearing large-scale networks. *IEEE Transactions on Circuits and Systems* **cas-24**(12), 709–717.

Schöber, V. (1995). A DFT expert system for integrated circuits. In: *2nd Workshop on Hierarchical Test Generation*.

Schreiber, H.H. (1977). A state space approach to analog fault signature generation. *Proceedings of the 20th Midwest Symposium on Circuits and Systems* pp. 200–205.

Schreiber, H.H. (1978). A review of analog automatic test generation. *IEEE Int. Automatic Testing Conference AUTOTESCON* pp. 1–8.

Sen, N. and R. Saeks (1979). Fault diagnosis for linear systems via multifrequency measurement. *IEEE Transactions on Circuits and Systems* **CAS 26**(7), 457–465.

Sheppard, J. W. and W. R. Simpson (1998). *Research Perspectives and Case Studies in Systems Test and Diagnosis*. Vol. 13 of *Frontiers in Electronic Testing*. Kluwer. Chapter 5. Inducing Inference Models from Case Data.ISBN 0-7923-8263-3.

Shieh, Y.R. and C. W. Wu (1998). Control and observation structures for analog circuits. *IEEE Design and Test of Computers* pp. 56–64.

Slamani, M. and B. Kaminska (1995). Multifrequency analysis of faults in analog circuits. *IEEE Design and Test of Computers* pp. 70–80.

Smyth, B. and E. McKenna (1999). Building compact competent case-bases.. *Proceedings of the 3rd International Conference on Case-based Reasoning. ICCBR99* pp. 329–342.

Smyth, B. and M.T. Keane (1995). Remembering to forget: A competence-preserving case deletion policy for cbr systems. *Proceedings of the 14th International Joint Conference on Artificial Intelligence* pp. 377–382.

Soma, M. (1990). A desing for test methodology for active analog filters. *Proceedings of the IEEE International Test Conference* pp. 183–192.

Starzik, J., J. Pang, S. Manetti, M. Piccirilli and G. Fedi (2000). Finding ambiguity groups in low testability analog circuits. *IEEE Transactions on Circuits and Systems. Fundamental Theory and Applications* **47**(8), 1125–1137.

Stenbakken, G.N., T.M. Souders and G.W. Stewart (1989). Ambiguity groups and testability. *IEEE Transactions on Instrumentation and Measurement* **38**(5), 941–947.

Stuart, A., J.K. Keith Ord and S. Arnold (1999). *Kendall's Advanced Theory of Statistics*. Vol. 2A of *Classical Inference and Linear Model*. 6th ed.. Arnold. ISBN 0 340 66230 1.

Su, C., C. H. Wang, W.J. Wang and IS Tseng (2003). 1149.4 based on-line quiescent state monitoring technique. *Proceedings of the 21st IEEE VLSI Test Symposium (VTS.03)* pp. 197–202.

Varghese, K.C., J.H. Williams and D.R. Towill (1978). Computer aided feature selection for enhanced analogue systems fault location. *IEEE Patern Recognition* pp. 265–280.

Voorakaranam, R., S. Chakrabarti, J. Hou, A. Gomes, S. Cherubal and A. Chatterjee (1997). Hierarchical specification-driven analog fault modeling for efficient fault simulation and diagnosis. *International Test Conference* pp. 903–912.

Wen, Y.C. and K.J. Lee (2001). Analysis and generation of control and observation structures for analog circuits. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems* **20**(1), 165–171.

Wettschereck, D., D. W. Aha and T. Mohri (1997). A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. *Artificial Intelligence Review* **11**, 273–314.

Wey, C.L. (1996). Mixed-signal circuit testing. a review. *IEEE International Conference on Electronics, Circuits, and Systems, ICECS '96* pp. 1064–1067.

Wilson, D. and T. Martinez (2000*a*). Reduction techniques for instance-based learning algorithms. *Machine Learning* **38**(3), 257–286.

Wilson, D. R. and T. Martinez (2000*b*). An integrated instance-based learning algorithm. *Computational Intelligence* **16**(1), 1–28.

Wilson, D. R. and T. R. Martinez (1997*a*). Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research* pp. 1–34.

Wilson, R. and R. Martinez (1997*b*). Improved heterogeneous distance functions. *Journal of Artificial Intelligence Research* **6**, 1–34.

Witten, I. H. and E. Frank (2000). *Data Mining. Practical Machine Learning Tools and Techniques with Java Implementations.* Data Management Systems. Morgan Kaufmann Publishers. ISBN 1-55860-552-5.

Worsman, M. and M. W. T. Wong (2000). Non linear analog circuit fault diagnosis with large change sensitivity. *International Journal of Circuit Theory and Applications* (28), 281–303. Ed. John Wiley and Sons, Ltd.

Zheng, H. H., A. Balivada and Abraham J. A. (1996). A novel test generation approach for parametric faults in linear analog circuits.. *Proceeedings of the 14th IEEE VLSI Test Symposium* pp. 470–475.

Zhu, J. and Q. Yang (1999). Remembering to add: Competence-preserving case-addition policies for case base maintenance. *International Joint Conference on Artificial Intelligence, IJCAI99* pp. 234–241.

Appendix A

THE DEVELOPED ENVIRONMENT

A.1 The Software for Simulation

In order to test the different diagnosis methods on different linear circuits, an environment based on Matlab has been developed. The main menu is given in Figure A.1

The menu is clearly divided into several parts. *Circuit Selection* allows the user to define the circuit by means of a transfer function or by drawing the circuit using a PSPICE environment. This last option is quite useful when dealing with integrated circuits or with non-linear circuits.

When selecting the *Circuit Selection* option on the main menu, there is possible to choose between the PSPICE environment or the transfer function. If the latter is selected, Figure A.2 appears.

First of all the coefficients of the transfer function numerator and denominator have to be entered in a symbolic way. For example, a transfer function denominator

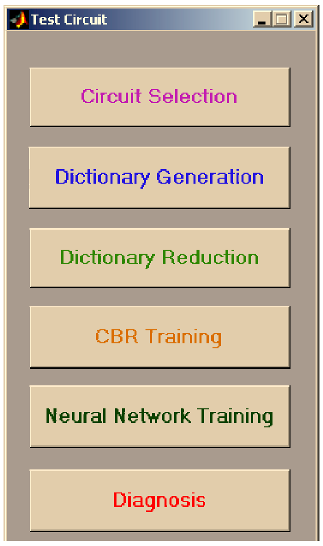
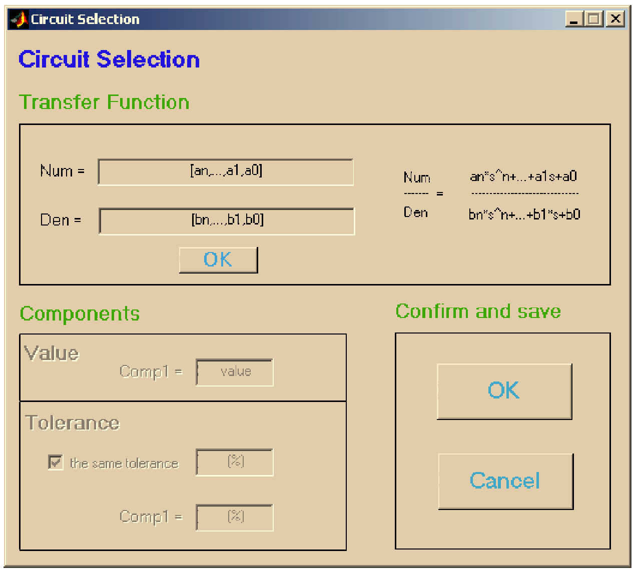


Figure A.1: Main menu



Num = [an,...,a1,a0]

Den = [bn,....,b1,b0]

Num / Den = an*s^n+...+a1s+a0 / bn*s^n+...+b1*s+b0

Comp1 = value

☑ the same tolerance (%)

Comp1 = (%)

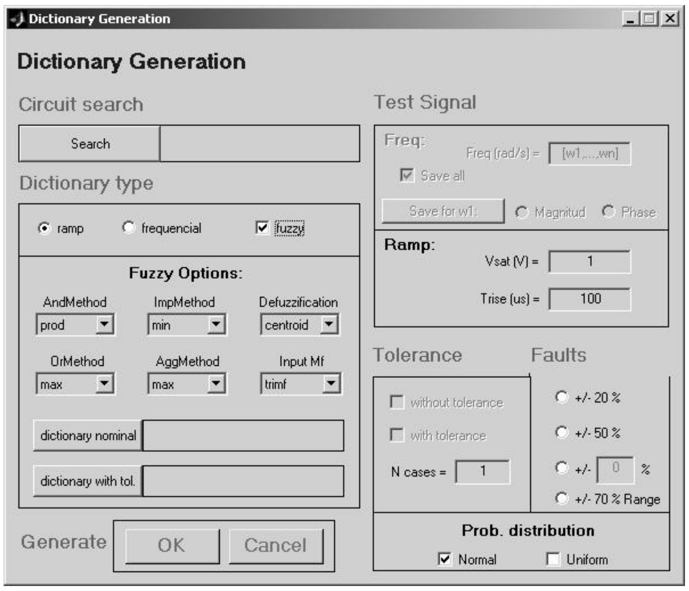
Figure A.2: Circuit selection menu

$$RCs + 1$$

will be introduced as [R*C, 1].

The next step allows the nominal value of the components to be introduced in a sequential manner. The tolerance is also selected, making it possible to assign the same tolerance to all the variables or to enter a particular tolerance value for each component. At this point, we have a transfer function describing the circuit stored.

The following part of the main menu is related to dictionary case generation. It can also serve to generate a file of new cases for the CBR-system training or testing purposes. If *Dictionary Generation* is selected from the main menu, the submenu of Figure A.3 is displayed.



Dictionary type: [✓ ramp] frequencial [✓ fuzzy]

Fuzzy Options:

AndMethod	ImpMethod	Defuzzification
prod	min	centroid
OrMethod	**AggMethod**	**Input Mf**
max	max	trimf

Freq (rad/s) = [w1,....,wn]
[✓ Save all]

Vsat (V) = 1
Trise (us) = 100

N cases = 1

+/- 0 %

Prob. distribution: [✓ Normal] Uniform

Figure A.3: Dictionary generation menu

After selecting the transfer function circuit, we have to choose the type of dictionary to generate, based on the saturated ramp or sinusoidal (frequency method) input. At the same time, the option of generating the fuzzy sets in parallel as explained in Chapter 4 can be chosen. If this last action is taken, the fuzzy options are activated. After introducing the adequate input test signal parameters, the generated set can be the classic dictionary, with one case per fault, or several cases per considered fault can be generated. Finally the universe of faults to simulate has to be defined. It is possible to produce deviations of ±20%, ±50% or ±X% separately or simultaneously. Also, random faults distributed in the margin between 0% and ±70% can be used as alternatives. This last option is very useful when testing the methods for non-previously considered faults. The probability distribution of the faults can be normal, uniform or a set can be selected for each of them. The result is stored in a variable that is a structure containing the values of the measures for each simulated situation, but also includes the universe of faults considered, the characteristics of the test signal used and the components involved.

The *Dictionary Reduction* option from the main menu is used to eliminate noisy or redundant data by means of the DROP4 or IB3 algorithms. The reduction method, the metrics, the normalization procedure (by range or using the variance) and several other options can be selected previously.

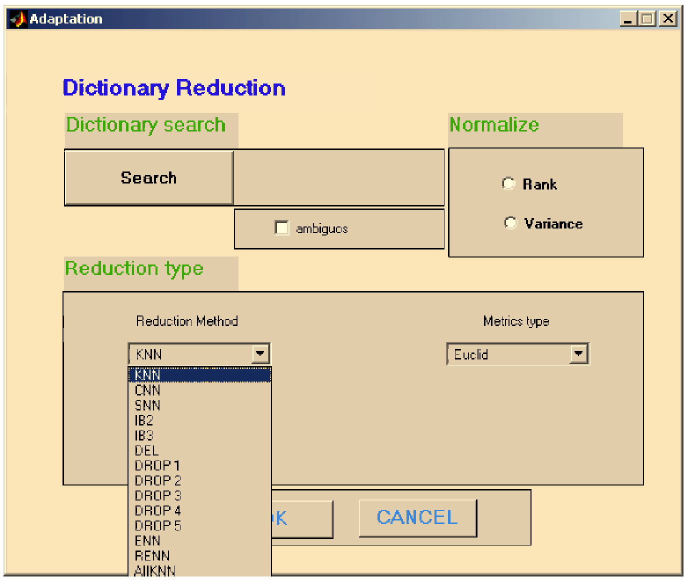
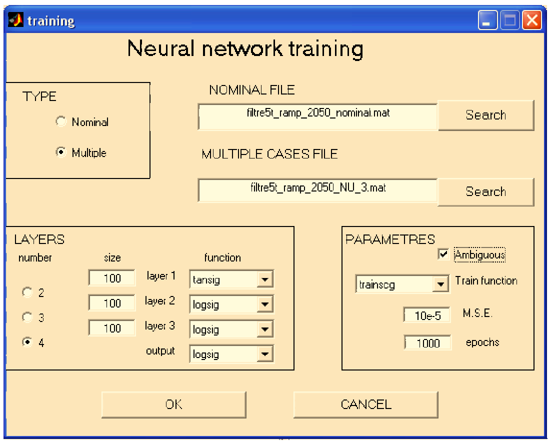


Figure A.4: Dictionary reduction menu

The *Neural Network Training* window offers the possibility of designing and training a neural network. The data used for training can be selected from the dictionaries directory. Only the nominal cases or a dictionary can contain several cases of each fault (multiple option).


training

Neural network training

TYPE: Nominal [✓ Multiple]

NOMINAL FILE: filtre5t_ramp_2050_nominal.mat

MULTIPLE CASES FILE: filtre5t_ramp_2050_NU_3.mat

LAYERS

number	size		function
2	100	layer 1	tansig
3	100	layer 2	logsig
[✓ 4]	100	layer 3	logsig
		output	logsig

PARAMETRES

[✓] Ambiguous

Train function: trainscg

M.S.E.: 10e-5

epochs: 1000

Figure A.5: Neural network training menu

The neural network input and output cells are known. The user has to design the internal layers, choosing how many layers and the number of neurons per layer. Also, the transfer function shape can be chosen. Afterwards, the training function, the minimum error considered to stop (MSE) and the maximum number of epochs allowable may be specified.

Last but not least important of the methods implemented is the *CBR Train*.

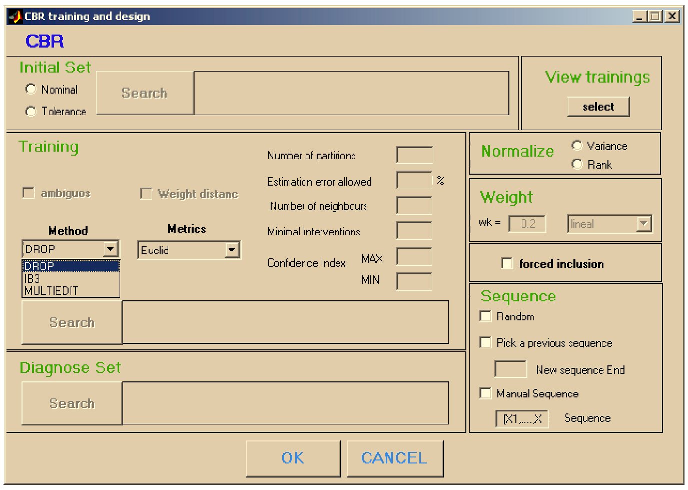


Figure A.6: CBR menu

Figure A.6 displays the parameters necessary to configure and train the CBR system. First of all, the initial set used as the initial case base is selected. This set can be a nominal dictionary (one case per considered fault) or a dictionary with several cases taken per fault considered. The training method can be DROP4, IB3 or Multedit. The metric has to be selected as well. The error allowed when estimating the parameters value is introduced in the *Estimation Error Allowed* cell. Also, the *Minimum number of interventions* necessary to consider a case for removal and the maximum and minimum values of the Confidence index for the IB3 method has to be introduced. The confidence index is not activated when the DROP4 method is selected for training (it makes no sense) and only one of them is activated when Multiedit is chosen (the method only needs one index). The *Number of partitions* refers to the divisions that have to be made in a file to generate several subfiles for training.

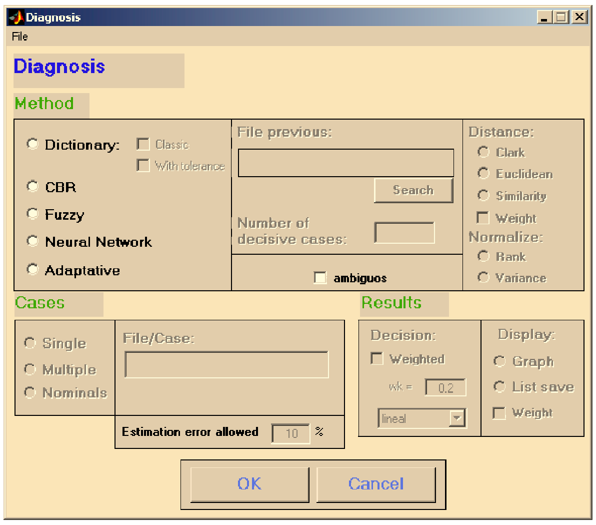


Figure A.7: Diagnosis menu

The last button of the main menu is for diagnosis purposes, and the window that appears after clicking this option is shown in Figure A.7. First of all, the method used to diagnose has to be selected. Afterward, the parameters to introduce for the particular selected method are highlighted to introduce their values. Then, a single case, a file containing the data to test or a file containing only one case per considered fault is picked up. The results can be stored in a file or they can be shown graphically.

A.2 The Hardware Environment

A.2.1 The Data Acquisition Board

The card is a PCI-6071-E from National Instruments and the main characteristics of its analog inputs and outputs are shown in Table A.1 and Table A.2.

Analog Input Characteristics		
General		
	Number of channels	64 single-ended 32 differential
	ADC resolution	12 bits, 1 in 4096
	Max simple rate	1.25 Msamples/sec.
	Input single range for bipolar configuration	From ±50 mV to ±10V (depending on the gain)
Transfer characteristics		
	Relative accuracy	±0.5 LSB typ dithered ±1.5 LSB max undithered
	DNL	±0.5 LSB typ ±1 LSB max
	Offset error (after calibration)	Pregain error: ±12 µV max
		Postgain error: ±0.5 mV max
Amplifier characteristics		
	Input impedance	100 GΩ in parallel with 100 pF
	Input bias current	±200 pA
	CMRR (for a gain of 1)	100 dB
	Bandwidth	1.6 MHz
	Crosstalk	-80 dB (DC to 100KHz)

Table A.1: Main analog inputs characteristics of the PCI 6071-E board

The board has digital inputs and outputs as well. Some of the digital outputs are used to configure the faults in the biquadratic simulation board. One of the analog outputs is used as a generator and is synchronized to one of the analog inputs, that is used to capture the signal.

The input signal range can be unipolar or bipolar. The input range goes from $\pm 10\ V$ to $\pm 50\ mV$ for the bipolar mode. The selection of one margin or another is made by the gain parameter. This gain is software programmable. A gain of 0.5 limits the bipolar input range to $\pm 10\ V$. For a gain of 1 the input

Analog Output characteristics		
General		
	Number of channels	2
	Resolution	12 bits, 1 in 4096
	Max update rate per channel	1 Msamples/s
Transfer characteristics		
	Relative accuracy	±0.3 LSB typ
	DNL	±0.3 LSB typ
	Offset error	±1 mV max
Voltage output		
	Ranges	±10 V, 0 to 10V
	Input impedance	10 KΩ
	Bandwidth (-3dB)	1 MHz
	Slew Rate	20 V/μs

Table A.2: Main analog outputs characteristics of the PCI 6071-E board

range is decreased to $\pm 5\ V$ and so on. Of course, when reducing the input range, the resolution of the measure increases since the 4096 discretization levels (2^{12}) are confined in a narrower range. Hence, the input range should be selected as close as possible to the expected maximum input level, avoiding any saturation of the board amplifiers.

Also, the PCI 6071-E board allows the dither to be enabled, which means an addition of approximately 0.5 LSB of Gaussian noise to the signal to be converted by the ADC. This addition increases the resolution of the PCI board and is useful for applications involving averaging, such as the present one.

A.2.2 The Fault Simulation Board

To facilitate the faults generation, digital potentiometers are used for resistors fault simulation. The AD8402 from Analog Devices containing two potentiometers has been selected for this purpose. Their values are fixed by a digital code sent to the potentiometer. Of course, the universe of possible wiper positions is finite and it depends on the number of steps allowed by the potentiometer. It is possible to find from 32- up to 1024- step potentiometers. In our case, 256-position potentiometers are used because they offer

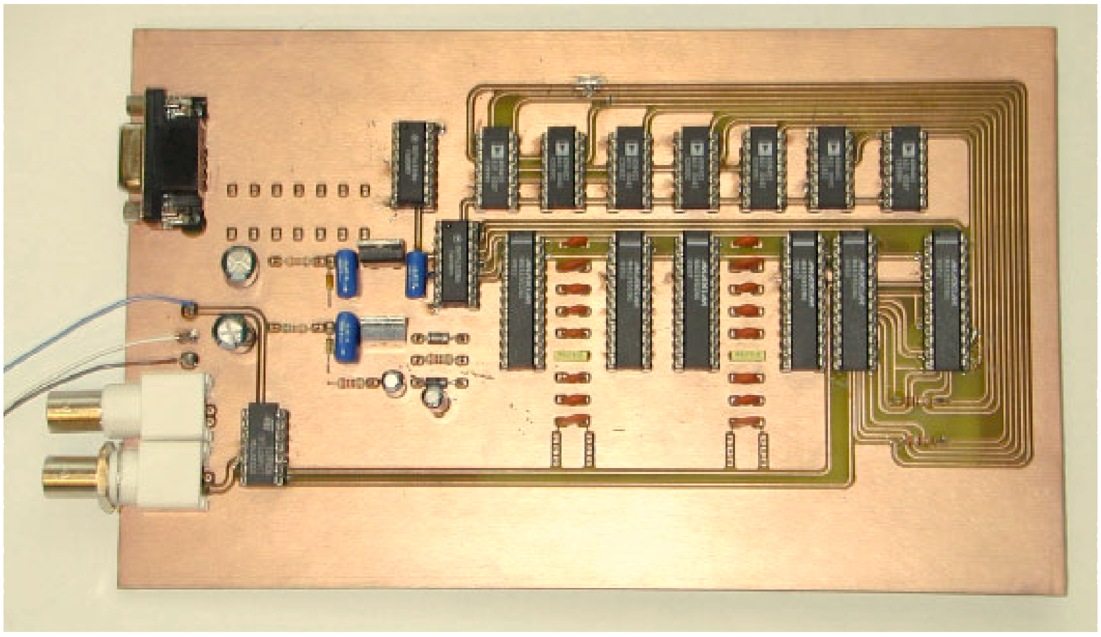


Figure A.8: The real circuit

enough resolution for testing the methods in the real circuit. Also, they can be selected according to the desired value: 1K, 5K, 10K, 50K or 100K. When the user selects a possible resistor value, the potentiometer is adjusted to the closest possible value, taking into account the smallest step the potentiometer has. Also, the wiper terminal introduces a small resistance that is typically $R_{wiper} = 50\ \Omega$ for the selected potentiometers. This value is also taken into account in order to calculate the real value the resistor is going to take.

Capacitors are simulated using an electronic switch and a bank of discrete capacitors. Figure A.8 shows the biquadratic fault simulating board.

The integrated circuit TL074 is used for the operational amplifiers. It has a maximum output short circuit current of $I_{os} = 40$ mA, a Slew Rate of $SR = 13V/\mu s$ and a gain bandwidth product of $GBP = 3\ MHz$. These limitations will not influence the response to a saturated ramp input of a $1V/100\mu s$ slope with a saturation value of 1 V for the considered faults. Concerning the dynamical part of the circuit response, a slope of $1V/100\mu s$ is equivalent to $0.01V/\mu s$, which is far from surpassing the Slew Rate limitation. On the other hand, the currents at the output of the operational amplifiers are much less than its limit of 40 mA, even in the overshoot that can reach close to 3 V at V_0 depending on the fault provoked.

Labwindows is used to program the faults, to access the board and to capture the real data. There are several possible options. One of the Labwindows screens allows the user to program single faults. The desired value of the component is introduced and the output for a ramp input is captured after a click on the *OK* button. The other Labwindows screen is useful for randomly generating multiple faults. In both situations, the input is a 60 periods ramp signal. The advantage of doing this, is that the 60 periods of the response can be averaged in order to reduce the noise present at the output. After that, the program calculates the signature parameters (SP, t_d, t_r and V_{est} for the saturated ramp input stimuli) and saves the fault signatures in a file, with their corresponding fault label, the faulty component and its deviation. The file can be easily converted to a format readable by the Matlab software in order to test the CBR-system proposed with real data. In spite of the limitation on the sampling time due to the Windows operating system, the buffer of the data acquisition board allows samples to be obtained in real time and sampled at a shorter sample time, i.e. $1\mu s$.

Appendix B

HIERARCHICAL DECOMPOSITION OF CIRCUITS

The idea is to carry out a nodal decomposition in order to divide the circuit into blocks. The heuristic cluster algorithm described in (Sangiovanni-Vicentelli *et al.*, 1977) is used together with the hierarchical decomposition approach proposed in (Salama *et al.*, 1984).

It uses the concept of *contour tableau*, which consists of a three-column table, like Table B.1. The first column is called the *iterating set* (IS) and it contains iterating nodes. The second one refers to the *adjacent set* (AS), and the last one has the *contour number* (CN).

IS	AS	CN
IS(1)	AS(1)	CN(1)
IS(2)	AS(2)	CN(2)
IS(3)	AS(3)	CN(3)
⋮	⋮	⋮

Table B.1: Contour Tableau

The algorithm to build the table is described in (Sangiovanni-Vicentelli *et al.*, 1977). Let's see an example of how a hierarchical decomposition is performed. Taking the active filter of Figure B.1,

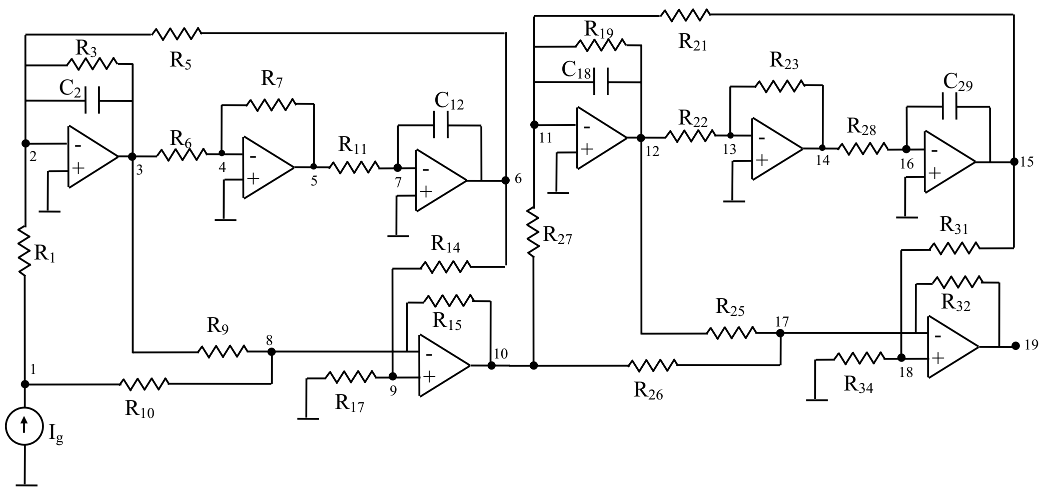


Figure B.1: Active filter as an example of a large circuit

with the values shown in Table B.2 and with an input current of $I_g(t) = 0.01cos(2000t)\ A$.

R_1	182	R_{11}	2.64 K	R_{23}	10 K	R_{34}	10 K
R_3	1.57 K	R_{14}	5.41 K	R_{25}	500 K	C_2	10 nF
R_5	2.64 K	R_{15}	1 K	R_{26}	111.1 K	C_{12}	10 nF
R_6	10 K	R_{17}	1 K	R_{27}	1.14 K	C_{18}	10 nF
R_7	10 K	R_{19}	4.84 K	R_{28}	2.32 K	C_{29}	10 nF
R_9	100 K	R_{21}	2.32 K	R_{31}	72.4 K		
R_{10}	11.1 K	R_{22}	10 K	R_{32}	10 K		

Table B.2: Component values for the large active filter

and considering nodes $n1$, $n3$, $n5$, $n6$, $n8$, $n10$, $n12$, $n14$, $n15$, $n17$ and $n19$ as the measurement nodes, the equivalent nodal graph interpretation of the circuit is depicted in Figure B.2

Applying the Sangiovanni-Vicentelli algorithm, contour table B.3 is obtained.

Hence the cutting point can be taken at node $n10$ since it is the iteration

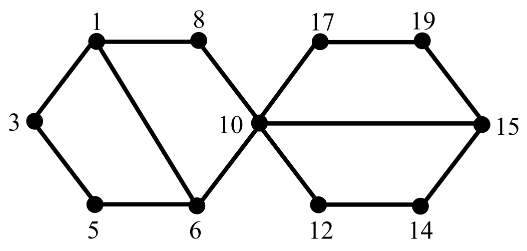


Figure B.2: Equivalent nodal graph of the active filter

IS	AS	CN
n1	n3 n6 n8	3
n3	n5 n6 n8	3
n5	n6 n8	2
n6	n8 n10	2
n8	n10	1
n10	n12 n15 n17	3
n12	n14 n15 n17	3
n14	n15 n17	2
n15	n17 n19	2
n17	n19	1
n19	-	0

Table B.3: Contour Tableau for the large active filter

with de minimum CN. This indicates that the circuit can be split into two subcircuits, one part compressed between nodes $n1$, $n10$ and ground and the other between nodes $n10$, $n19$ and ground. A similar procedure can be applied to each of these subcircuits in order to continue with the block division. Finally the hierarchical diagram of Figure B.3 is obtained.

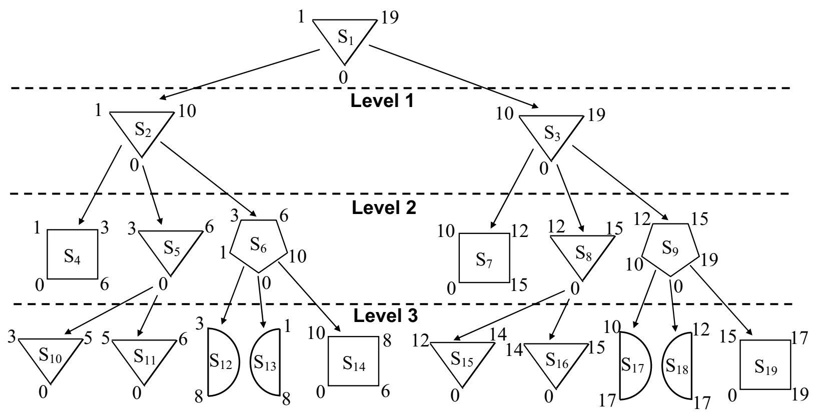


Figure B.3: Hierarchical decomposition of the active filter

Once the circuit is hierarchically divided in blocks, a fault verification is performed in order to isolate any faults, at least the block level.