The implementation of generalised models of magnetic materials using artificial neural networks

Hamadou Saliah-Hassane

Department of Electrical and Computer Engineering

McGill University

Montreal

September, 1999

A thesis submitted to the Faculty of Graduate Studies and Research

In partial fulfilment of the requirements for the degree of

Doctor of Philosophy

Abstract

This thesis proposes a framework for using an artificial neural network (ANN) as a uniform and universal method for modelling the behaviour of both isotropic and anisotropic materials, exhibiting hysteresis or not. It addresses the complex task of coupling a given material representation with a true analysis system, i.e. one based on finite element analysis. We have demonstrated this approach through numerical examples and experimental results that we have commented and discussed.

In this thesis, we present a broad review of targeted approaches used in the past to represent magnetic materials, pointing out their particular advantages and deficiencies. This comprehensive and detailed review has led us to suggest using mixed type models as a knowledge source in order to acquire training data for the proposed artificial neural network (ANN). We also include information on characteristics pertinent to a better understanding of artificial neural networks (ANNs) to support the proposed architecture.

Sommaire

Nous proposons dans cette thèse, l'utilisation de réseaux de neurones artificiels pour modéliser d'une manière uniforme et universelle des matériaux magnétiques hystérétiques ou non hystérétiques et tenant compte de leurs comportements isotropiques ou non isotropiques. Nous avons également effectué la tâche complexe de créer une interface numérique qui permet de coupler tout model de matériaux magnétiques à un système d'analyse. Il s'agit dans ce cas précis de programmes d'analyse basés sur les éléments finis en électromagnétisme. Nous avons expérimentalement montré la faisabilité de notre approche en traitant des exemples numériques et les résultats obtenus ont été analysés et commentés.

Dans cette thèse, nous avons également fait une analyse critique de certaines méthodes sélectionnés utilisées dans le passé pour modéliser et représenter les matériaux magnétiques en faisant ressortir les avantages particuliers et les inconvénients de ces approches. Cette analyse détaillée nous suggère d'utiliser des modèles de matériaux magnétiques de types mixtes pour l'acquisition de la base de connaissances à exploiter pour l'entraînement des réseaux de neurones artificiel proposés. Nous avons également inclus dans cette thèse, pour soutenir notre approche, des informations sur les caractéristiques pertinents permettant la compréhension des réseaux de neurones artificiels.

Acknowledgements

My PhD supervisor, Professor David Alister Lowther, for his guidance, patience and encouragement, for our countless academic and non-academic discussions, and for his confidence in me and the financial support he provided by means of a research grant whereby I, "a seasoned student", was able to serve as his senior research and teaching assistant during my studies at this reputed university.

Members of my advisory committee, Professors Francisco Galiana and Boon Tec Ooi.

If I were to mention all the many friends and colleagues who encouraged me along the way, I'm afraid that I would be able to boast of very little personal merit myself. Yet I must mention the contributions of Behzad Forghani, Dave Kodjo, Margaret Toussaint and Rachelle Renaud, each offering me assistance in his or her respective field of expertise, sharing enriching ideas, debugging computer programs or putting my thoughts to paper in appropriate English.

Financial assistance from the Canadian International Development Agency (CIDA) and the Natural Sciences and Engineering Research Council of Canada (NSERC) whose scholarship support made this work possible.

Members of Infolytica who generously provided me access to their software source codes to link and test my models.

My parents for their encouragement in my childhood years and for working so hard to provide me with the best possible education.

The entire Linteau family who have been awaiting this moment for many a moon and especially my mother-in-law Gabrielle Blais, who, I am sure, never doubted in my ability to succeed.

nd most importantly, the person whose patience, support, understanding and organisational cills have brought my work to fruition. Yes, I am referring to my dear wife Josée who, along /ith my children Guillaume and Marie-Laure, the latter born during my research period, deserve ıe lion's share of this hard-earned degree.

Table of Contents

List of Figures

Chapter 1

. Introduction

.1 Magnetic Material Modelling and Computer Aided Design Systems

Magnetic materials, i.e. those that exhibit permeabilities greater than air and, possibly, hysteresis, re crucial in the design of electromagnetic devices. They are used as sources of a magnetic field permanent magnets), as magnetic conductors and as the basis of memory in computer systems. n general, they are used to construct the electromagnetic field distributions required to ıccomplish a desired task, whether it be the translation of energy between electric, magnetic and nechanical forms, or the creation of a particular field structure at specified points in space, or yet ıgain the storage of energy.

The design process for such devices must be based on an accurate prediction of how these devices will perform. The concept of using computers to assist in this design process has been around almost as long as computers have been available. However, the accuracy of any computer simulation is only as good as the models used in the simulation. This implies that the behaviour of the materials must be modelled in a precise manner. The magnetic properties of materials are dependent on several parameters, including temperature, stress and frequency. In addition, the materials all exhibit "memory" to a greater or lesser extent in the form of hysteresis. All of these features mean that constructing a computational model of a magnetic material is a complex and difficult task.

.2 Motivation

ı the past, the methodology used for constructing a computer model for a given material has een dependent on the final goal of the analysis package and, since these goals have been both mited and specialised, the material models have been likewise. For example, if the material is to e modelled as non-hysteretic, the initial magnetisation curve can be handled by a polynomial, by iecewise linear segments or by a sequence of cubic splines. If a range of temperatures is a rucial factor, then a different model for each temperature needs to be constructed. For analyses vhere the hysteretic properties become important, polynomials [Hodgdon, 1988a; Cortial et al, .997] as well as phenomenological models based on Stoner-Wohlfarth [Stoner-Wohlfarth, 1948], Preisach [Preisach, 1935] and Mayergoyz [Mayergoyz, 1991] have been used.

Thus, in general, the modelling methodology has been elaborated according to the characteristics of the material pertinent to a specific analysis and has been based on a purely mathematical approach. This has been a satisfactory technique for analysis systems thus far, since most of these have been highly specialised. But current trends have made multiple representations of the same material in analysis systems, within a more generic approach, an absolute must - an unwieldy task if following traditional methods. In addition, because of the interaction of many complex factors in the modelling of magnetic materials, it is difficult to systematise the process by the application of a mathematical approach alone.

The challenge, therefore, is to find a more effective way of representing all the properties and behaviours of materials within a computational environment. This is critical to the success of any simulation system. This thesis proposes the use of artificial neural networks (ANN) to meet this challenge, as has been illustrated in our recent publications [Saliah-Lowther, 1995, 1997a, 1997b; Saliah et al, 1997, 1998, 1999] and as has been referred to in papers by respected researchers in

the field [Takahashi et al, 1998; Adly-Abd-El-Hafiz, 1998; Benbouzid, 1998]. This approach is obviously gaining in popularity. Artificial neural networks provide a means of representing complex multi-dimensional surfaces in a uniform manner. Such a system offers the possibility of creating a uniform and universal model of all the properties of a magnetic material, including the effects of hysteresis, temperature, frequency and stress. It also offers reduced data and computational requirements compared to other approaches, resulting in a more efficient system in terms of memory usage and execution speed.

1.3 Claim of Original Contribution

In this thesis, the capability of various neural network architectures to handle different types of hypersurfaces is analysed in order to identify networks that are suitable for encompassing the multiple facets of magnetic material modelling. To that end, a framework for using Artificial Neural Networks as a uniform method for modelling the behaviour of both isotropic and anisotropic magnetic materials, with and without hysteresis, has been established.

We followed a series of steps in our research by modelling successively, first anhysteretic and isotropic material, then hysteretic material and anisotropic material. But there is still some modifications of the finite element solver to be done before including the full hysteresis model. The contribution here consists of extracting and using the knowledge acquired from the neural network within a Computer Aided Analysis and Design System (CAD) or a Computer Aided Learning (CAL) environment, following an inductive method.

This work also contributes to a unified and consistent format for a hierarchical classification of magnetic hysteresis models.

As well, this thesis outlines a methodology for using artificial neural networks to identify electromagnetic hysteresis parameters and for determining the properties that this ANN should

xhibit, namely fast convergence and the ability to adapt its memory requirements to cope with ıe complexity of a given problem.

inally, this work takes a step towards automating the knowledge acquisition process required for ıe design of electromagnetic devices. This is a useful approach because much of this knowledge ; deduced from concrete examples.

I.4 Thesis Outline

I.4.1 Chapter 1

This chapter provides an introduction to the complexities surrounding the process of adequately nodelling magnetic materials. It also establishes the need for an accurate model of such naterials. Previous approaches used to achieve this end are briefly examined and some of their imitations are highlighted, thus demonstrating the need for a more comprehensive way of modelling these magnetic materials. The use of artificial neural networks is proposed as one such comprehensive way. Finally, the original contributions of this thesis are outlined.

1.4.2 Chapter 2

This chapter presents a literature survey of state-of-the-art magnetic material modelling techniques. A brief classification of the various material models is outlined with a focus on how the material properties impact on the choice of computation systems. Certain properties of the materials are discussed in regard to their suitability for various applications. The material model interface requirements for FE solvers are presented.

.4.3 Chapter 3

'his chapter provides some essential information on learning systems as well as a basic ntroduction to neural networks. It focuses only on characteristics pertinent to the understanding)f the neural networks used in the present work.

1.4.4 Chapter 4

Γhis chapter introduces the concept of using an artificial neural network as a tool for modelling nagnetic materials. The requirements of such a model for magnetic materials are analysed and methods for mapping these requirements onto the properties of neural networks are presented.

In this chapter it is shown how knowledge provided by the neural network, related to the model for magnetic materials, can be extracted and used within a Computer Aided Analysis and Design System (CAD), namely by coupling the model with a finite element analysis system. The requirements for such an interface are identified and discussed.

1.4.5 Chapter 5

This chapter provides conclusions on the proposed technique for modelling magnetic materials and provides pointers for further research in this area.

1.5 Conventions

All plots of B or M against H are in International System (SI) or Gaussian units (cgs), as indicated in the Conversion Table below. Since the latter are often used for certain quantities in practical magnetics work, we have also retained them for experimental comparison purposes.

Quantity	cgs (Gaussian)	Conversion Factor	International System (SI)
Magnetic field, **H**	oersted (Oe)	$(1/4\pi) \times 10^3$	ampere/meter (A/m)
Magnetisation, **M**	emu/cm^3 or emu/cc	10^3	ampere/meter (A/m)
Flux density, **B**	Gauss (G)	10^{-4}	tesla (T), weber/m^2, (Wb/m^2)

Chapter 2

2 Review of Magnetic Material Modelling Techniques

2.1 Definitions for Magnetic Materials

Every material can be classified in terms of its magnetic property, according to its capability of responding to an applied magnetic field, H. The resulting induced *flux density*, B, is determined by the constitutive relation

$$B = \mu H \tag{2.1}$$

where μ is the *magnetic permeability* and is equal to $\mu_0 = 4\pi \times 10^{-7}$ *Henry/meter* for free space. The expression (2.1) can be replaced, using the *relative permeability* μ_r, by the relation

$$B = \mu_r \mu_0 H \tag{2.2}$$

Most of the time, H and B are not oriented in the same direction. It is then better to consider the vector form of the scalar relation 2.2 together with an additional vector term M, the *magnetisation* of a given substance. It follows that the flux density is expressed by the general relation

$$B = \mu_0 (H + M) \tag{2.3}$$

M can be formulated as

$$M = \chi H \tag{2.4}$$

where $\chi = \mu_r - 1$, the ratio M/H, a dimensionless parameter, called the *susceptibility* of the material. Susceptibility is one of the most commonly used parameters to classify a given substance into a *diamagnetic* category, if $\chi < 0$ or into a *paramagnetic* category, if $\chi > 0$.

The following tensor representation relating the reluctivity ν (i.e inverse permeability) and the flux density B_p measured along the principal axis is sometimes used to model anisotropic hard materials [Lowther-Silvester, 1986].

$$\nu(B_p) = \frac{1}{\mu_0}\begin{bmatrix} \nu_r(B_p) & 0 \\ 0 & c \end{bmatrix} \tag{2.5}$$

where the constant c, a characteristic of the material, is expressed in terms of the anisotropic field H_A and the saturation magnetisation M_s by:

$$c = \frac{\mu_0 H_A}{\mu_0 H_A + M_s} \tag{2.6}$$

Figure 2.1 illustrates various responses, also called the *first* or *initial magnetisation curves*, together with magnetisation states that occur when various classes of materials are submitted to an increasing field. In fact, some materials cannot be classified in either the paramagnetic or diamagnetic category. The $M(H)$ relationship is multi-valued and M may be non-zero at $H = 0$ and, moreover, its value depends on the past history of H. These types of materials are qualified as *ferromagnetic* materials exhibiting *hysteretic* behaviour.

Figure 2.2 shows an example of the outermost loop formed by the $M(H)$ relationship known as the *limiting hysteresis curve* or the *major loop*. The initial slope is called the *initial permeability* (or *initial susceptibility* for the inverse). It is not mandatory to define the permeability of a given ferromagnetic material. Instead, it is common to use the term $\mu_{eff} = \Delta B_h / \Delta H_h$ to describe the effective, the differential or the incremental permeability at a particular value of the applied field.

The maximum value of the magnetisation is the *saturation magnetisation*. The value of M on the major loop where the applied field $H = 0$ is called *remanent magnetisation*.

Ferromagnetic materials showing hysteretic behaviours are defined in two groups, namely soft materials and hard materials. Soft materials are characterised by a low coercive force and high initial permeability, whereas hard materials exhibit a high coercive force and low initial permeability. Most magnetic materials, in particular permanent magnets, are fabricated so as to enhance their properties along a preferred direction. These preferred directions of magnetisation for a given sample are called *easy axes*.

This raises the topic of anisotropy, a property that expresses why and how a given material exhibits different $M(H)$ behaviours, depending on the direction of the applied field intensity. Anisotropy is a key concept to be considered when dealing with magnetic hysteresis effects. The need to deal with hysteresis and anisotropy effects for a given magnetic material reinforces our choice for the use of a vector hysteresis model as a starting point for the model proposed in this thesis.

A key goal of this thesis is to present a more general approach based on the behaviour of linear, non-linear, isotropic, anisotropic, hysteretic and anhysteretic materials within the same computational framework representation. A finite element solver using the provided representation frame can then quickly compute the hysteresis and eddy current losses within a magnetic circuit or the read-write fields of magnetic recording heads.

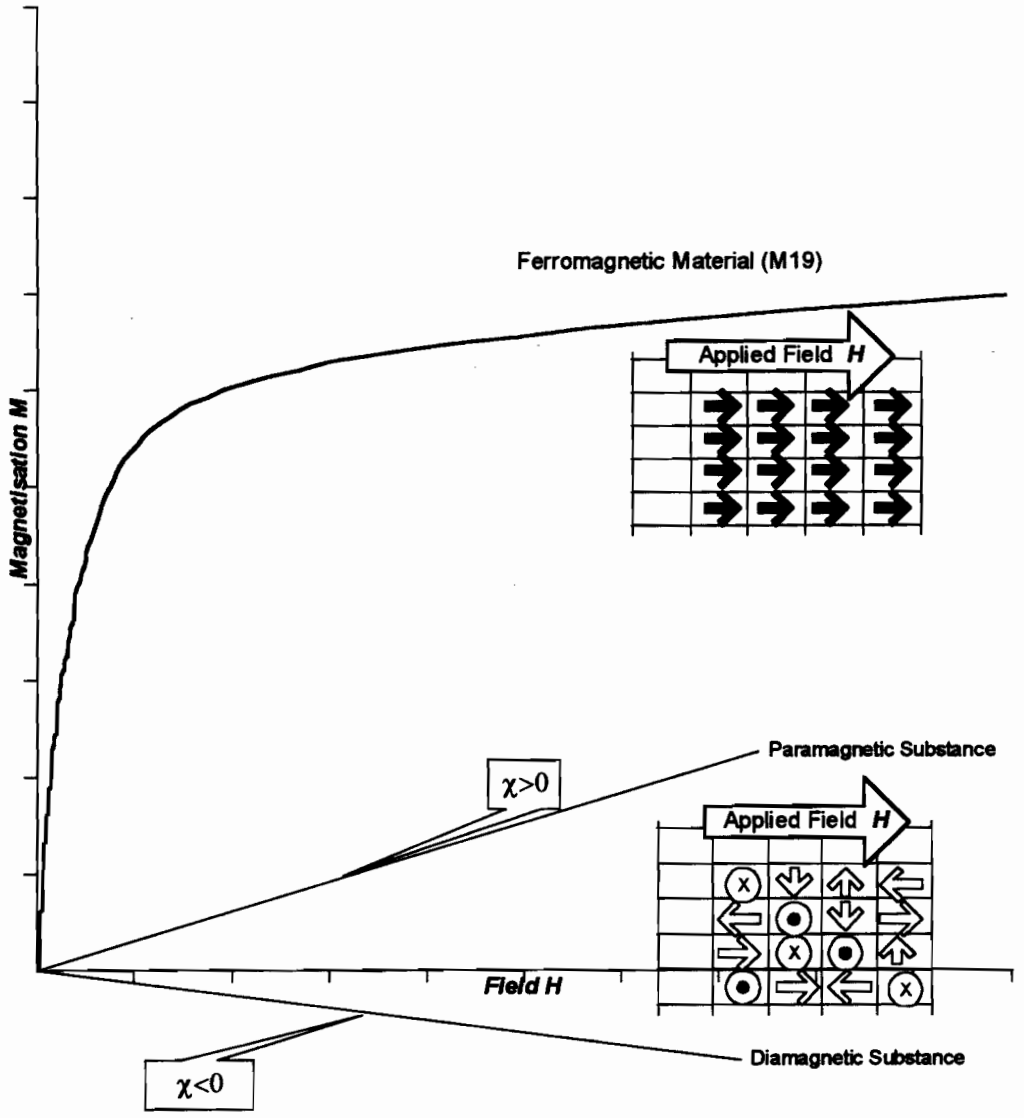


Figure 2.1 The Magnetisation Process

In the case of paramagnetic substances, beneath the Curie temperature, random orientation occurs above the field H axis, with an increasing external applied field. But in the case of diamagnetic substances, below the field H axis, as shown in Fig. 2.1, the material microstructure can be simulated by two loop currents of the same values, but with an opposite-directed magnetic moment.

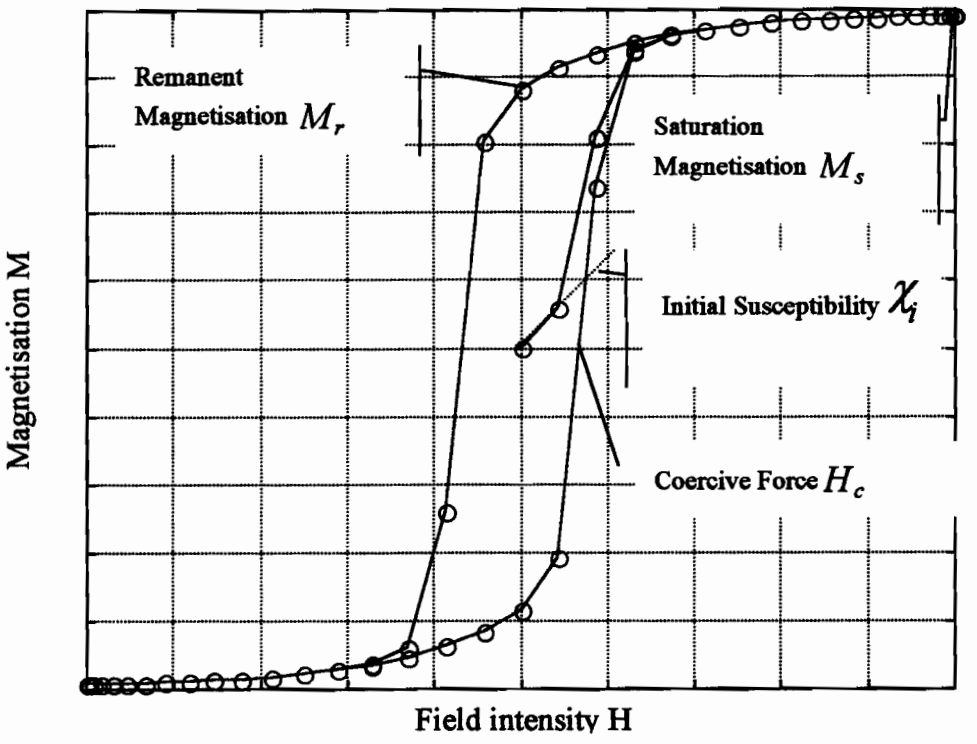


Figure 2.2 The First Magnetisation and the Limiting Hysteresis Curve

2.2 Classification of Material Models

2.2.1 Single-valued $M-H$ and $B-H$ Magnetic Material Models

A magnetic material model should be described only in terms of a few parameters based on underlying physical principles and should exhibit adequate agreement with the given set of measurements. To cope with gradient-based iterative solution techniques such as Newton-Raphson, the gradient information should be smooth. There are many theories regarding methods of modelling magnetic material properties and the most popular models using CAD systems

Lowther-Silvester, 1985] are based on single-valued $B-H$ curves represented by one of the following relations:

1. The reluctivity as a function of flux density squared: $\nu = \nu(B^2)$.
2. The field as a function of flux density: $H = H(B)$.
3. The permeability as a function of field squared: $\mu = \mu(H^2)$.

The above formulas are often stored in tabular or list form and models used to represent single-valued $B-H$ curves are created using one of the following mathematical curve fitting or interpolation processes:

2.2.1.1 Polynomial Interpolation Methods

Spline approximation techniques and *polynomial interpolation* methods using *Hermite* or *Lagrange* polynomials, need the interpolant to pass through the data set used to calculate the interpolation. A number of lower degree piecewise-continuous polynomials for interpolation of a data set are then used to avoid problems associated with higher degree single polynomial interpolation, where Lagrange or Hermite polynomials tend to fluctuate and exhibit wiggles. However, the method fails to represent the same material over the full range of $B-H$ values, as shown in Figures 2.3 and 2.4. On the other hand, cubic and natural cubic splines, for example, use cubic polynomials and differ only in the necessary end-point conditions of the interpolation.

A *cubic spline interpolant* of a given data set requires information regarding the first derivative of the interpolant at specified end points. In the case of a *natural cubic spline* interpolant of a given data set, the second derivative of the interpolant at the end points must be zero. When using the Lagrange interpolation method, the values in the data set have to be distinct. This is not always the case when the data are acquired from experimental measurements. On the other hand, the Hermite interpolation technique tolerates repeated values, that is, more than one point with

the same $B-H$ co-ordinate can be found in a given data set. To model a non-linear $B-H$ curve in MagNet 5, 40 cubic Hermite polynomials are used to guarantee the continuity of both the interpolant function and its derivative [Lowther-Silvester, 1985; Forghani et al, 1982].

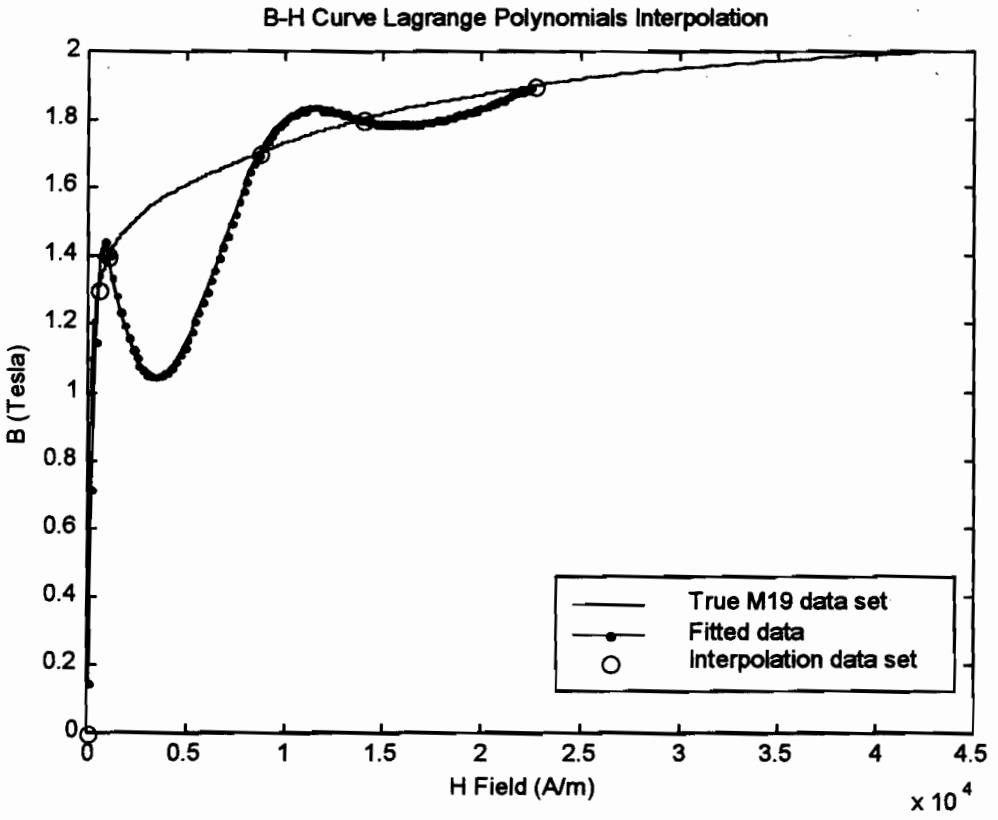


Figure 2.3 Wiggle Problems Exhibited by Lagrange Polynomials

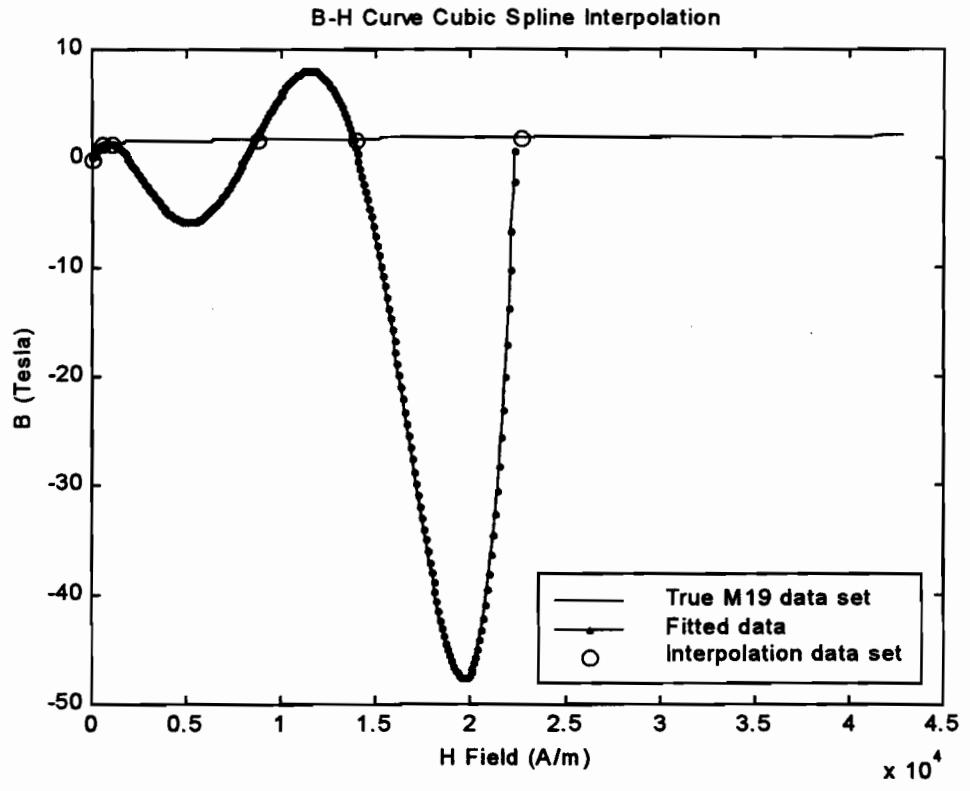


Figure 2.4 Cubic Spline Interpolation of Six Points Lying within the Range [0, 1.9]

2.2.1.2 Curve Fitting by Exponential Functions

An experimental B-H curve can be roughly approximated by a typical exponential function of the form $B(H) = B_{max}(1-e^{-aH})$. This representation leads to a large error near the origin and the knee of the magnetisation curve. El-Sherbiny [El-Sherbiny, 1973] and Hwang-Lord [Hwang-Lord, 1976] proposed a representation of a magnetisation characteristic by a sum of exponentials resulting from the equation:

$$B(H) = a_o + a_1 e^{-\alpha_1 H} + a_2 e^{-\alpha_2 H} + a_3 e^{-\alpha_3 H} + a_4 e^{-\alpha_4 H} \qquad (2.7)$$

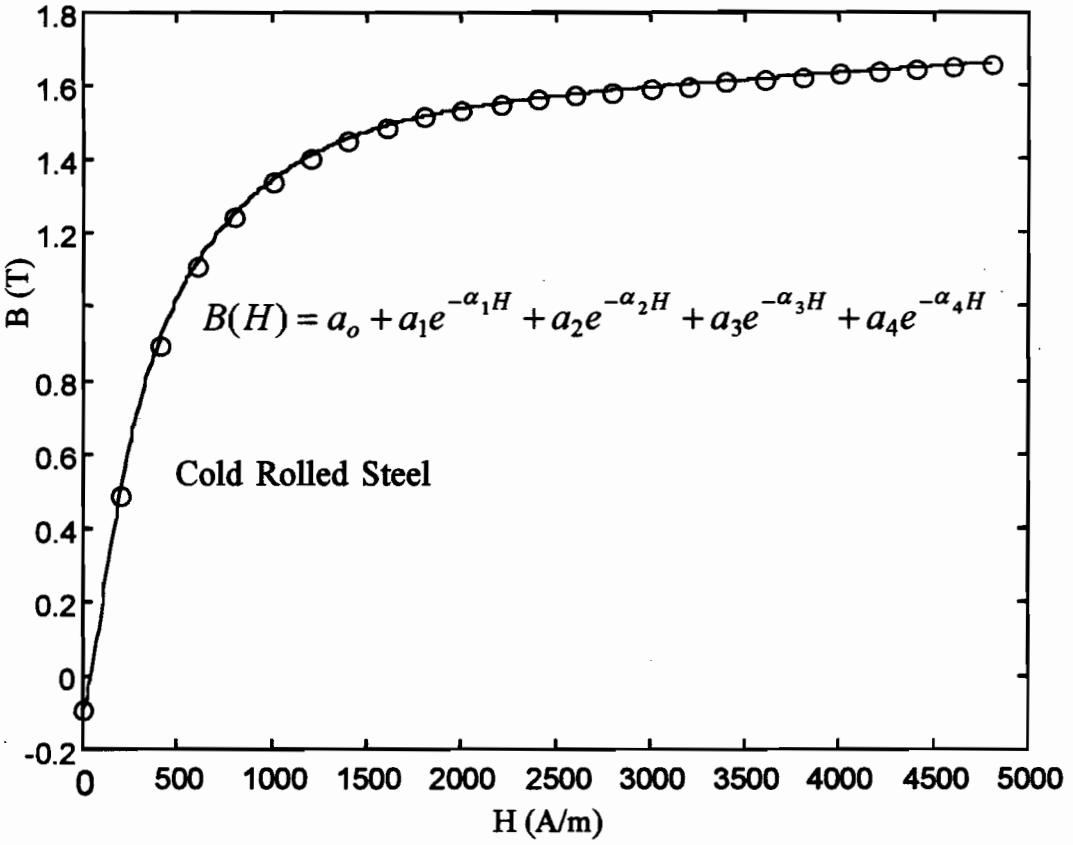


Figure 2.5 B-H Curves Representation by a Sum of Exponential Functions

where the values of the constants are determined beforehand from experimental data for each material. Figure 2.5 shows an example of the model for a cold rolled steel. One can imagine the computational cost of evaluating each exponential for its implementation within a FEM code.

2.2.1.3 Rational Fraction Approximations

Widger [Widger, 1969] and Rivas et al [Rivas et al, 1981] proposed simple approximations of magnetisation curves by means of second-order rational functions resulting from the following formula:

$$M = \frac{a_0 + a_1 H + a_2 H^2}{1 + b_1 H + b_2 H^2} \tag{2.8}$$

where the coefficients have to be related to the material constants such as the initial magnetic susceptibility and the Raleigh material constant. This approach attempts to prevent the power series approximation scheme from saturating at high fields and these authors claim that the model is efficient both for single $B-H$ curves and hysteresis loops.

2.3 Hysteresis Models

2.3.1 Hysteresis and Electric Circuits

There are two classes in common use:

- Electric circuits containing hysteretic components
- Simulation of hysteresis behaviour using circuit components (analogy).

Simulating hysteresis behaviour by means of circuit components takes into account conditions in an electric circuit containing a set of linear and non-linear electric components such as resistances, inductances, capacitances and, in some cases, diodes are added to the model. For example, an analogous R, L circuit can be used to model the relationship between the field intensity H and the flux density B using the following equations [Hannalla, 1980]:

$$H = \nu B + \xi \frac{dB}{dt} \tag{2.9}$$

$$\text{with } \nu = {}^{1}\!/_{\mu_r \mu_0} \tag{2.10}$$

$$\text{and } \xi = \tfrac{2}{\pi}\left(\frac{H_0}{B_m}\right)(T_1 - t_1) \tag{2.11}$$

where ν (the reluctivity) and ξ are considered to be analogous to resistance and inductance respectively and T_1 represents the time from the peak of B in one direction to the maximum

'alue in the other direction and t_1 represents the time since the last peak value of B. The term B_m stands for the last peak value of the flux density with the corresponding coercive force H_0.

An example of a basic circuit model consists of a ladder structure containing linear capacitors and ıon-linear resistors. It can be perceived as an element of a hysteresis model. The static and lynamic hysteretic behaviours generated by this circuit can be simulated in such a way that they 'esemble those observed in most experimental situations. Moreover, the ladder structure of the :ircuit can be the starting point towards identifying the formation process of local and non-local nemory.

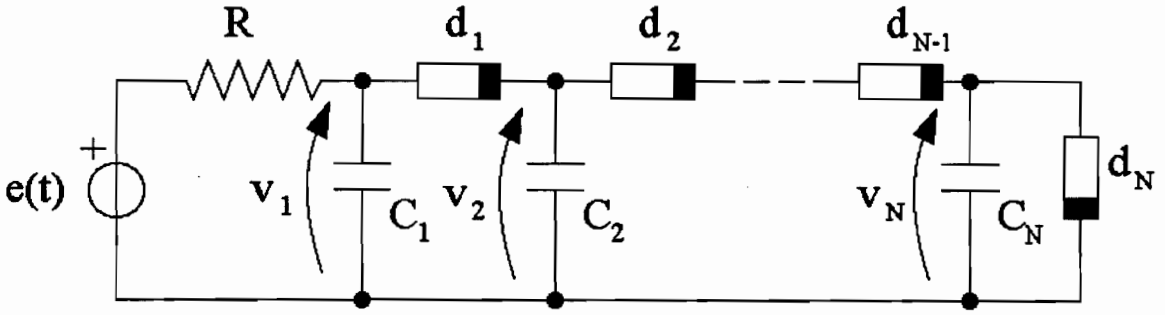


Figure 2.6 A Ladder Network Simulating an Hysteresis Behaviour

Zhu [Zhu, 1993, 1995] suggests a discrete modelling technique and using the numerical aspect of the implementation of the Preisach theory of hysteresis effect in magnetic cores, [Preisach, 1935] via the transmission line modelling method to solve problems exhibited by electric circuits with magnetic cores. The goal is to track the flux density versus the magnetic field strength path. The Preisach plane, as we will see later, allows the representation of the states of magnetisation and the corresponding Preisach diagram. The method claims to provide a simple, fast and yet accurate hysteresis model.

.3.2 Differential Equations and Functional Models

'or years, hysteresis problems have been treated with due mathematical rigour by a number of esearchers. Differential equations and "functionals" [Verdi-Visintin, 1985] play an important ole towards the best representation of mechanical, magnetic and at times thermodynamic ıysteresis phenomena. Bouc [Bouc, 1971], in his pioneering research, used a kind of nathematical functional F to represent the hysteretic behaviour of a variable $x(t)$ in a system vhen the characteristic diagram is time-independent. The F functional puts some functions end to :nd at each value of dx/dt. The method, given a suitable choice of key functions, can be :ffectively applied to the experimental behaviour of hysteretic electrical or mechanical systems. For example Bouc [Bouc, 1969] developed the following operator Λ :

$$[\Lambda(u)](t) = k\,u(t) + \int_0^t F(\int_s^t |u'(\tau)|\, d(\tau))\varphi(u(s))u'(s)ds \qquad \forall t \in [0,T] \tag{2.12}$$

where u is an absolutely continuous input, k is a positive constant, F and φ are continuous real functions with F positive and non-increasing that could be represented by $F(v) = Ae^{-\alpha v}$ with A and α positive. Similar methods are described by Kranoselskii et al [Kranoselskii et al, 1983] and Visintin [Visintin, 1994]. In a similar vein, two practical differential models developed by Hodgdon [Hodgdon, 1988b] and Jiles–Atherton [Jiles–Atherton, 1984] are noteworthy. We will briefly describe these models in the following sections.

2.3.3 Hodgdon Differential Equation Model

In the Hodgdon model, H is related to B by the following constitutive relations:

$$\frac{dB}{dH} = \begin{cases} k\,[f(H) - B] + g(H), for\ \dot{H} > 0; \\ -k\,[f(H) - B] + g(H), for\ \dot{H} < 0; \end{cases} \tag{2.13}$$

/ith the following inequality constraints:

$$f'(H) \geq g(H) \geq ke^{kH}\int_{H}^{\infty}\left[f'(\tau)-g(\tau)\right]e^{-k\tau}d\tau \qquad (2.14)$$

n the above equation, k is a real constant, g and f the material functions and f must be a iecewise continuous, monotonically increasing, odd function of H and its derivative has a finite imit for large H. On the other hand, g must also be piecewise continuous and an even function of H with a finite limit satisfying $g(\infty) = f'(\infty)$ [Hodgdon, 1988b]. The required experimental lata for Hodgdon's model measured on the major loop are: the coercive field H_c, the slope μ_c of he major loop at the coercive point, the flux density B_{cl} and the magnetic field H_{cl} at the positive closure point in the first quadrant as well as the slopes μ_s and μ_{cl} at and beyond this point respectively and, finally, the flux density B_r at full magnetic remanence and the corresponding slope μ_r.

2.3.4 Jiles and Atherton Differential Equation Models

The Jiles-Atherton theory of ferromagnetism hysteresis [Jiles-Atherton, 1984], is a phenomenological model. Five parameters are used to define the model:

- a, the form factor for the anhysteretic curve,
- c, the ratio of the initial susceptibility on the first magnetisation curve to the initial anhysteretic differential susceptibility. This parameter represents the reversible wall motion,
- α is a factor that takes into account the coupling between domains,
- k determines the hysteresis loss,
- M_s is the saturation magnetisation, the most easy parameter to obtain experimentally.

All these parameters can be extracted from the experimental measurements of the coercive field (H_c), the remanent magnetisation (H_r), the saturation magnetisation (M_s), the initial normal susceptibility (χ_{in}), the initial anhysteretic susceptibility (χ_{an}) and the maximum differential susceptibility (χ_m). Jiles [Jiles, 1992] describes in detail the experimental procedures to follow in order to acquire the above-mentioned parameters. The equations of hysteresis are expressed in terms of their reversible and irreversible susceptibilities as follows:

$$\frac{dM_{irr}}{dH} = \frac{M_{an} - M_{irr}}{k\delta - \alpha(M_{an} - M_{irr})} \tag{2.15}$$

and

$$\frac{dM_{rev}}{dH} = c\left(\frac{dM_{an}}{dH} - \frac{dM_{irr}}{dH}\right) \tag{2.16}$$

The summation of the aforementioned two terms gives the total magnetic susceptibility as expressed by $\frac{dH(M)}{dM}$ in the following equation:

$$\frac{dM(H)}{dH} = (1-c)\delta \times \frac{M_{an}(H_e) - M(H)}{k(1-c)sign(\dot{H}) - \alpha[M_{an}(H_e) - M(H)]} + c\frac{dM_{an}(H_e)}{dH_e} \tag{2.17}$$

where $H_e = H + \alpha M$ is the effective field and M_{an} is the anhysteretic magnetisation, a modified Langevin function resulting from the equation:

$$M_{an} = M_s\left[\coth\left(\frac{H_e}{a}\right) - \frac{a}{H_e}\right] \tag{2.18}$$

and δ, the directional parameter, has the following values:

$$\delta = \begin{cases} 0, & if\ \dot{H} < 0\ and\ M_{an}(H_e) - M(H) \geq 0 \\ 0, & if\ \dot{H} > 0\ and\ M_{an}(H_e) - M(H) \leq 0 \\ 1, & otherwise \end{cases} \tag{2.19}$$

The Jiles model [Jiles, 1992b] is a generalisation of the above-mentioned Jiles-Atherton [Jiles-Atherton, 1986] model that takes into account asymmetric hysteresis loops that occur in many practical magnetic applications such as those using permanent magnets. Jiles-Atherton's model describes a rate-independent hysteresis process demonstrating a good correlation with experimental data. Hence, this model is useful for fields that exhibit slow fluctuations in which there is almost no lag between the field and flux density. However, for rate-dependent problems such as eddy current damping in metals, this model's parameters have to be adjusted according to each type of material.

In fact, to achieve more comprehensive results in simulation of ferromagnetic systems, one should distinguish between rate-dependent and rate-independent memory effects. Rate-dependent memory is typically fading, hence scale-dependent. Rate-independent memory is persistent and scale-invariant [Bertotti, 1998]. Hodgdon [Hodgdon, 1988b] considers two types of rate-dependent responses: 1) the true pulse response in which the applied field changes instantaneously to a new value and then maintains that value while magnetic materials seek equilibrium; and 2) the second category of rate effect occurs when both the fluctuation rates of the applied field and flux density are zero.

Kvasnica and Kundracik [Kvasnica- Kundracik, 1996] have proposed a modification to the Jiles-Atherton anhysteretic model which allows temperature and stress to be taken into account by making some of the model parameters, functions of temperature and stress. The approach will give parametrized anhysteretic curves at different temperatures, as seen in the example illustrated in Figure 2.7, obtained by a Newton-Raphson technique from the following equation

$$M = M_s \, L\left(\frac{H + \alpha M(1 + \beta M^2)}{a}\right) \qquad (2.20)$$

where M_s is the saturation magnetisation and α and β are the constant mean field parameters expressing the coupling between domains. The value a is the initial magnetisation slope parameter and L is the Langevin function given by $L(x) = \left(\coth(x) - \frac{1}{x}\right)$.

The following table gives the temperature-dependent parameters used to solve the modified Jiles-Atherthon model.

Temperature	25 °C	50°C	75 °C	100°C
α	0.00636	0.00695	0.00754	0.0081
β	0.435×10^{-12}	0.435×10^{-12}	0.435×10^{-12}	0.435×10^{-12}
M_s (MA/m)	1.462	1.462	1.462	1.462
a (A/m)	3375	3657	3939	4221

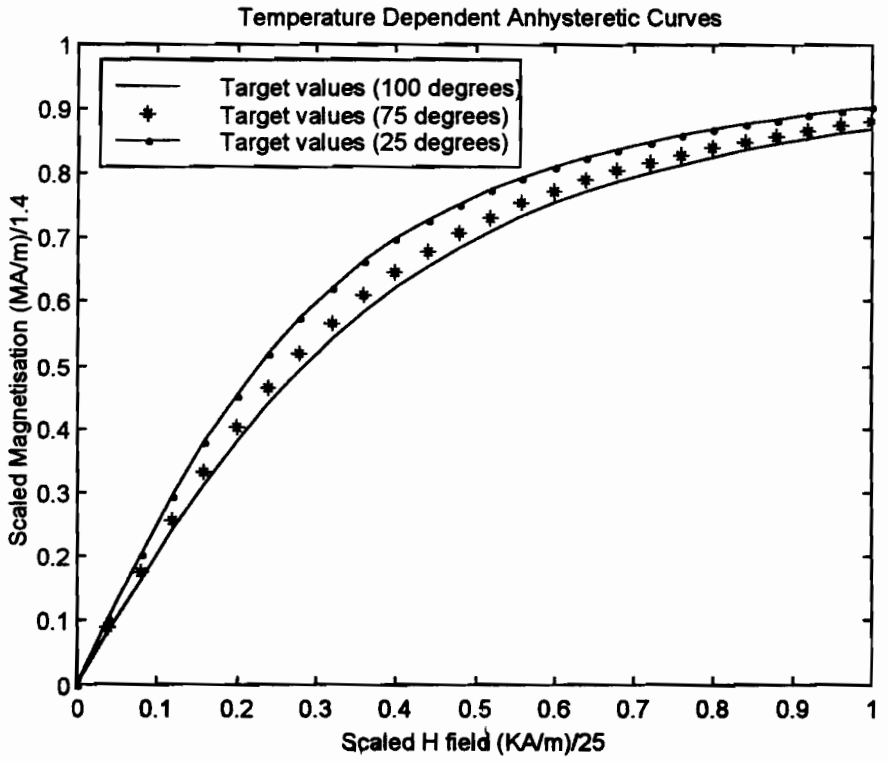


Figure 2.7 The Anhysteretic Curves from the Modified Jiles-Atherton Model

2.3.5 The Hyperbolic Tangent Model

The hyperbolic tangent model of hysteresis can be classified among the analytical or polynomial models [Ossart, 1990]. They are simple models used by Potter [Potter, 1970] to model the writing process in magnetic recording. The magnetisation closure point on the major loop, the remanent magnetisation field and the coercive field are required to model hysteretic behaviour. Minor loops are deduced by homothetic transformations. These models are simple enough to implement but cannot appropriately represent a given magnetic material. Figure 2.8 shows a hyperbolic tangent model exhibiting symmetrical loops.

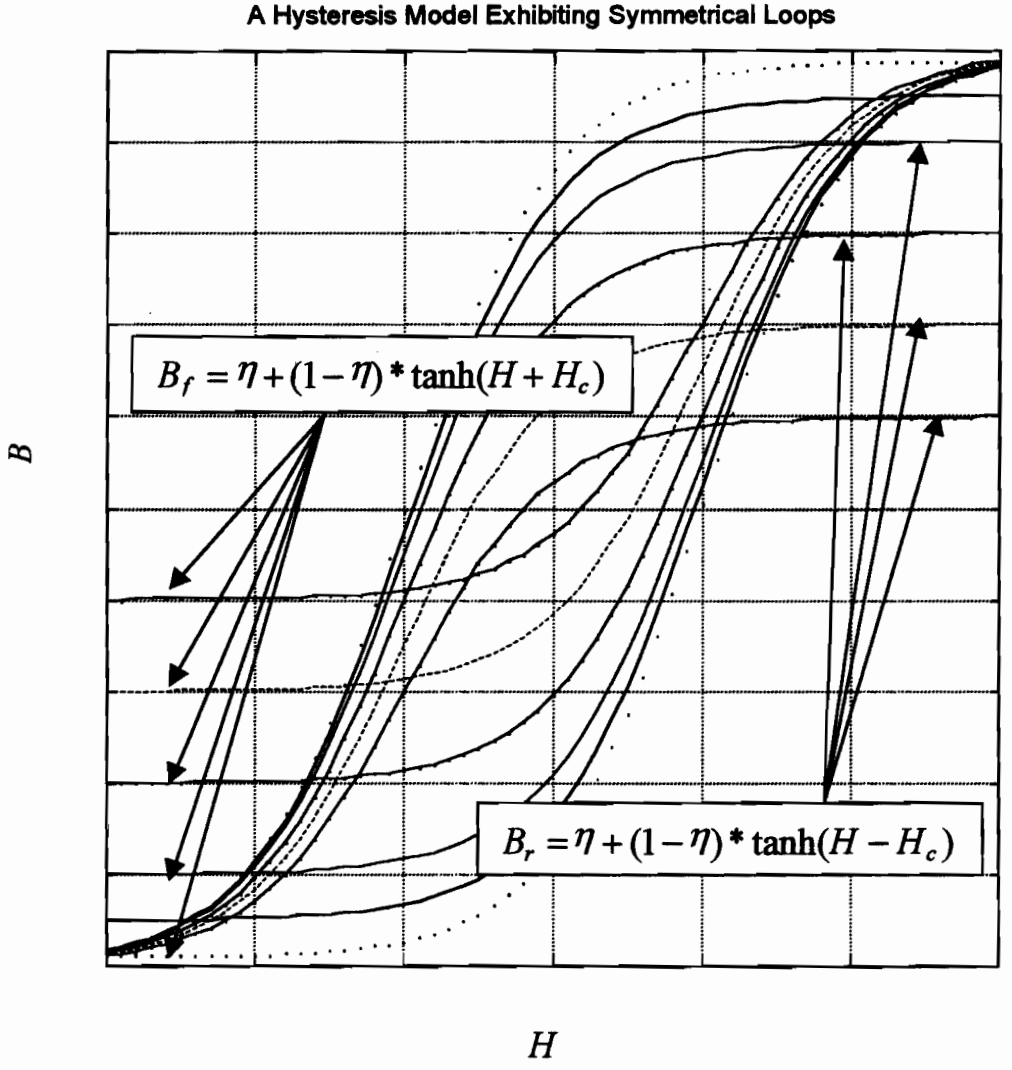


Figure 2.8 Hyperbolic Tangent Model of Hysteresis

In order to deal with the frequency for a broad range of materials, manufacturers provide a set of parameterised B-H curves as a function of frequency, as shown in Figure 2.9 obtained using a modified Zaher-Shobeir approach based on a hyperbolic tangent model [Zaher-Shobeir, 1983]. The following set of equations are used to compute the curves of Figure 2.9:

$$B = \varepsilon\left[a \arctan(b\,\varepsilon\,H + c) + \alpha\,\varepsilon\,H + e\right] \quad if\ -H_s < H < H_s \qquad (2.21)$$

$$B = B_s \quad if\ H \geq H_s \tag{2.16}$$

$$B = -B_s \quad if\ H \leq -H_s \tag{2.17}$$

$$\begin{cases} \varepsilon = -1 & if\ dH/dt > 0 \quad (ascending\ curve) \\ \varepsilon = \ 1 & if\ dH/dt < 0 \quad (descending\ curve) \end{cases} \tag{2.18}$$

with

$$a = 2\left(B_s - \alpha\, H_s\right)/\left(\arctan X_1 - \arctan X_2\right) \tag{2.19}$$

$$\begin{cases} e = -a\left(\arctan X_1 + \arctan X_2\right)/2 \\ X_1 = b\, H_s \quad and \quad X_2 = -b\, H_s + c \end{cases} \tag{2.20}$$

From experimental studies, observations are made that neither remanence induction B_r nor the slope around the coercive force vary with increased frequency. However, the coercive force varies with frequency. Frequency has a direct effect on parameter c in equation 2.21, therefore it is modified to take into account the frequency influence. The parameters α, a, b, and c are computed by a Newton-Raphson method from the system of non-linear equations derived from equation 2.21 to 2.26 given the prior experimental knowledge that b doesn't depend on α and c however, it is related to the slope around the coercive field H_c. On the other hand, parameter c, varies with the H_c obtained from the ascending experimental curve, Figure 2.9.

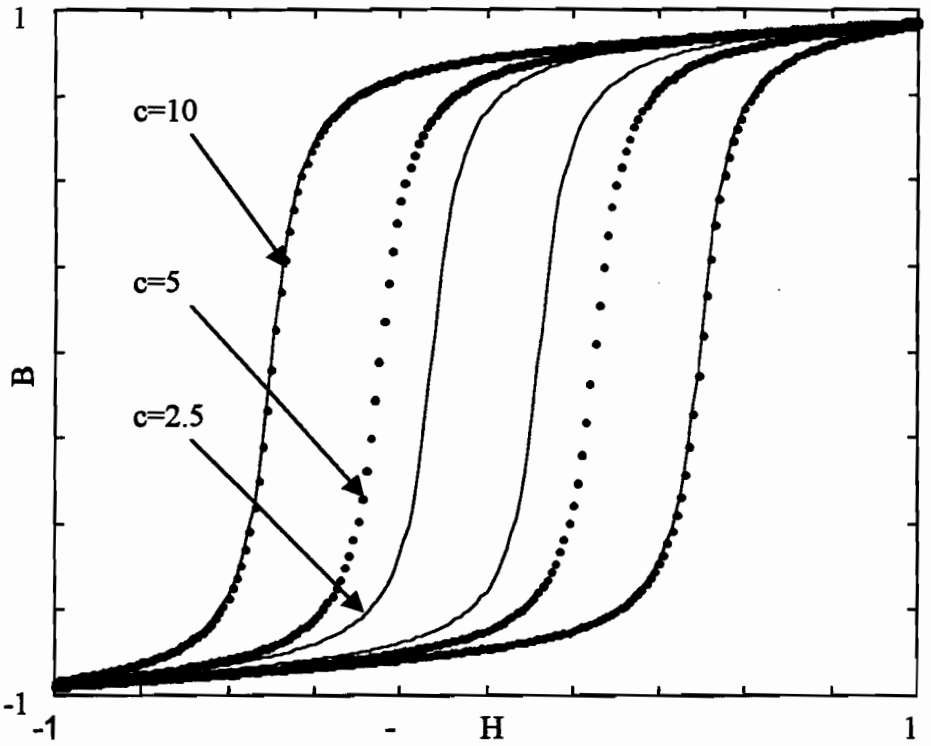


Figure 2.9 Parametrized B-H Curves with respect to Frequency (Scaled)

2.3.6 The Classical Preisach Model

The Preisach theory, first published in 1935, seems to be one of the most practical for hysteresis modelling and the advantages of using this approach have been pointed out by several researchers [Mayergoyz, 1991; Della Torre, 1990]. The Preisach model describes the hysteresis of a magnetic material using an infinite set of dipoles, which have non-symmetric rectangular hysteresis loops, as shown in Figure 2.10. A set of simple hysteresis operators $\gamma_{\alpha\beta}$ which is associated with positive and negative switching values α and β and a statistical distribution function $\mu(\alpha,\beta)$, also called the Preisach or weighting function, are used as a mapping artefact to model the $M-H$ behaviour for a given material.

The principle is better understood using the geometric interpretation of the Preisach model. Figure 2.11 represents the Preisach diagram where the principle is based on the fact that there is a one-to-one correspondence between the switching operators γ and points (α,β) on the half-plane $\{\alpha \geq \beta\}$. The weighting function is symmetrical with respect to the line $\alpha = -\beta$ and tends towards zero in all directions. Outside the limiting triangle $\mu(\alpha,\beta) = 0$, the evaluation of the resulting magnetisation $M(t)$ due to an applied field $H(t)$ is then restricted to the surface S of the limiting triangle T by applying the following equation:

$$M(t) = \iint_S \mu(\alpha,\beta)\gamma_{\alpha\beta}H(t)d\alpha d\beta \qquad (2.27)$$

where

$$S = \{\alpha \geq \beta,\ \beta \geq -H_{\max},\ \alpha \leq H_{\max}\} \qquad (2.28)$$

Figures 2.11 to 2.14 illustrate the numerical implementation of the classical scalar Preisach model and demonstrate an example of the evolution of a magnetisation process from a virgin state. At each instant, S splits into two sets of positive and negative magnetisation states, as shown in figure 2.12:

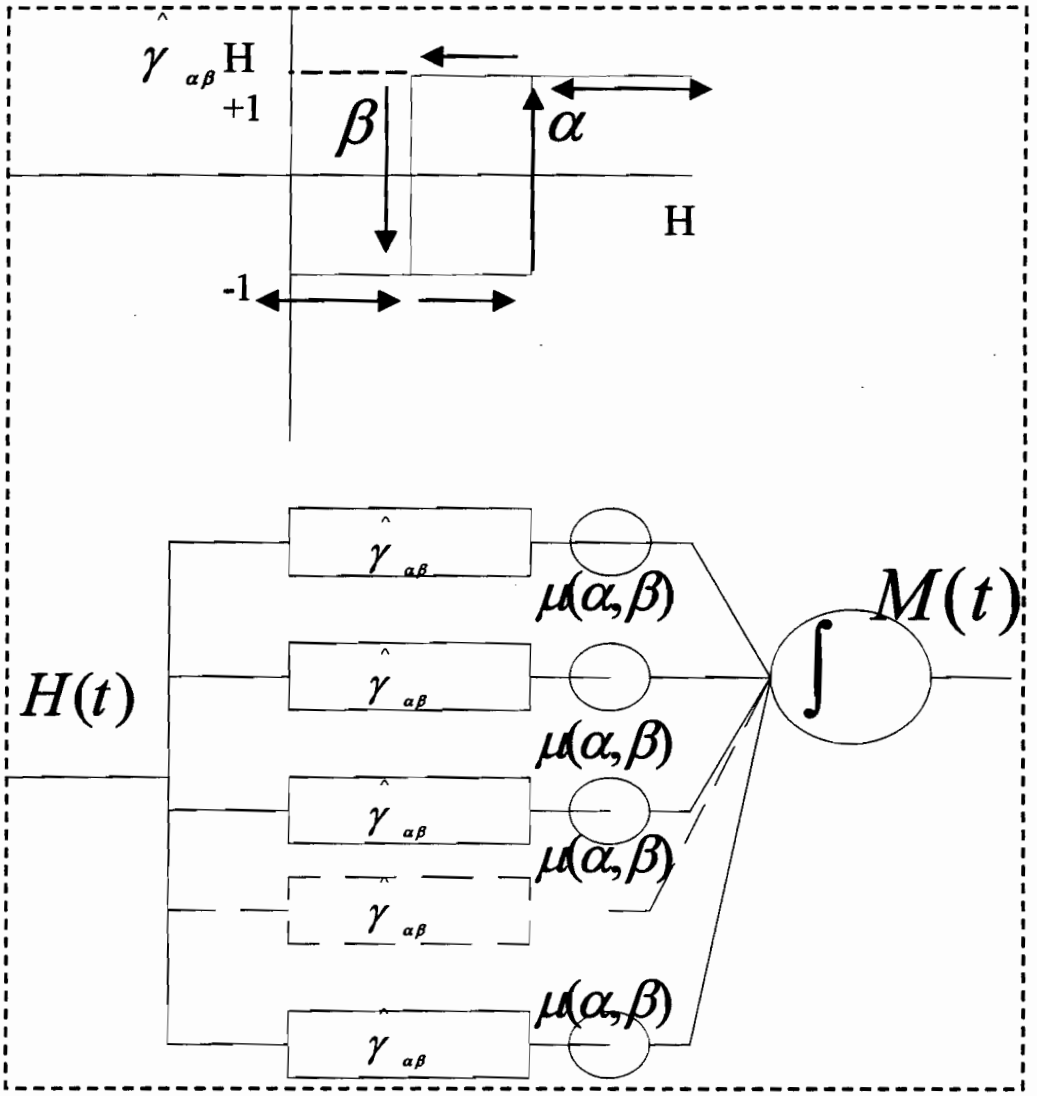


Figure 2.10 Preisach Hysteresis Operator Block Diagram adapted from Mayergoyz [Mayergoyz, 1991]

$S^+(t) \Rightarrow \gamma_{\alpha\beta}H(t) = +1$ and $S^-(t) \Rightarrow \gamma_{\alpha\beta}H(t) = -1$. Before undertaking the numerical implementation of the model, one must determine the Preisach distribution function using the transition curves [Rouve et al, 1995; Mayergoyz, 1991]. The major steps in the numerical implementation include the discretization of the Preisach plane and the evaluation of the integral.

The latter is simply the sum of the triangles formed by the vertices and the corners at each time step.

For a detailed explanation of this section, see references from Mayergoyz [Mayergoyz, 1991], Naidu [Naidu, 1990] and Henrotte [Henrotte, 1991].

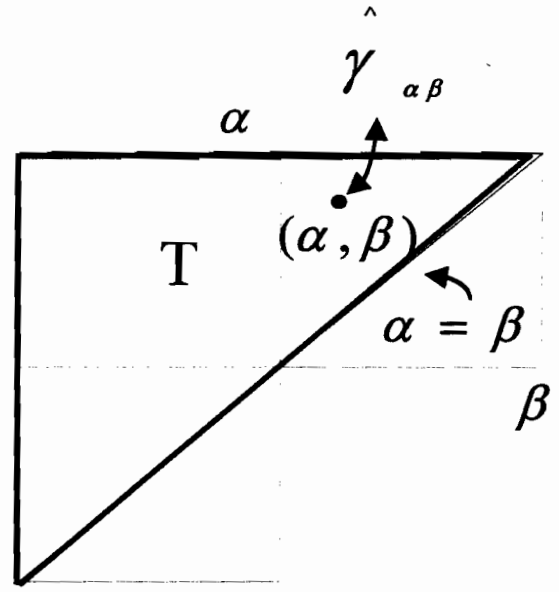


Figure 2.11 Preisach Triangle

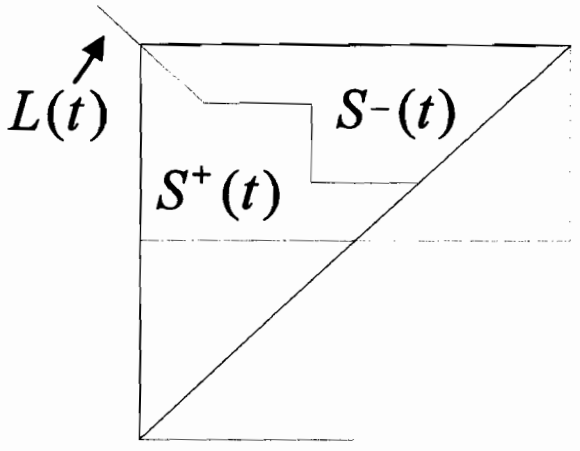


Figure 2.12 A Typical Dipole Distribution

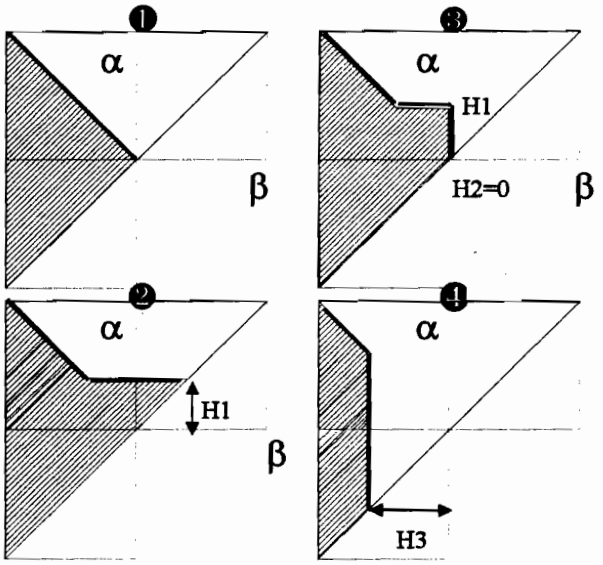


Figure 2.13 Preisach plane from Virgin Material to Negative Saturation

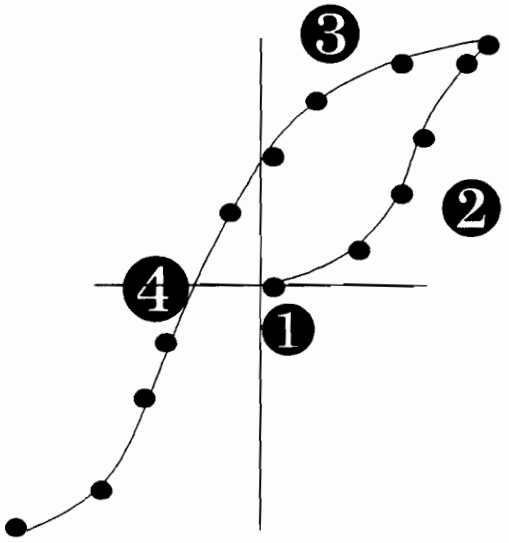


Figure 2.14 $M(H)$ Evolution from Virgin State to Negative Saturation

2.3.7 The Combined Moving Vector Stoner-Wohlfarth-Preisach-based Model

The theoretical foundation of a vector model includes the Stoner-Wohlfarth approach [Stoner-Wohlfarth, 1948], the classical scalar Preisach phenomenological model [Preisach, 1935] and Mayergoyz's treatment [Mayergoyz, 1991]. In fact, a vector model can be thought of as the superposition of scalar models. Research work on the combined moving vector model can be found in Oti [Oti, 1990], Ossart-Davidson [Ossart-Davidson, 1995] and Davidson [Davidson, 1996]. Figure 2.15 [Della Torre-Vajda, 1995] illustrates our interpretation of Della Torre and Vajda's classification of Preisach-based hysteresis models.

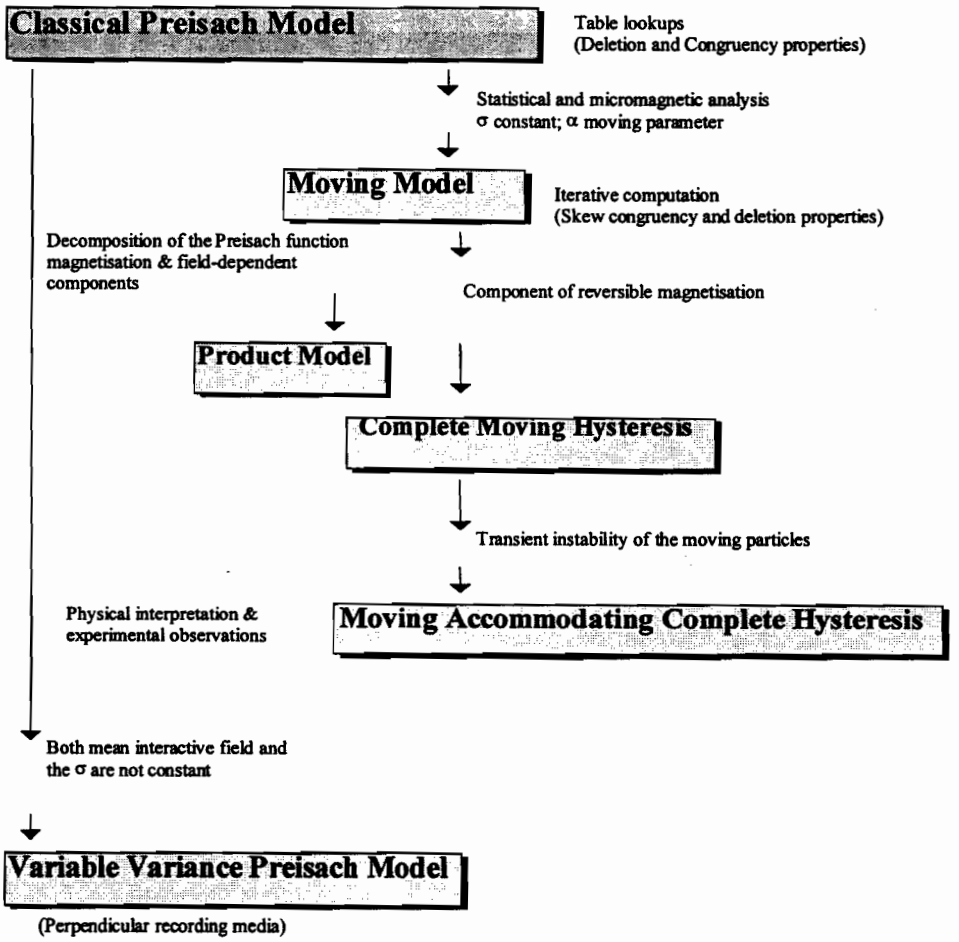


Figure 2.15 Hierarchy of Scalar Preisach Models for Recording Media

This classification could be summarised as follows. The *Classical Preisach Model (CPM)* is numerically efficient. It requires only a table lookup to compute the magnetisation. Necessary and sufficient conditions for its applicability are achieved by the deletion property and the congruency property of minor loops. However the CPM takes into account the statistics of the interaction field that are considered to be independent of the magnetisation process. The *Moving Model (MM)* uses a fast-converging iterative computational process. It replaces the congruency property with its skew-congruency counterpart and preserves the deletion property. The *Product Models (PM)* are magnetisation-dependent Preisach models. They suggest a decomposition of the Preisach functions into a magnetisation-dependent and a field-dependent component. Via these models, non-linear congruent minor loops can be computed and the deletion property maintained. To compensate for the inability of the classical Preisach and Moving models to describe both the irreversible and reversible components of magnetisation, the *Complete Moving Hysteresis (CMH)* annexes a magnetisation state-dependent, locally reversible component into the moving model. The *Moving Accommodating Complete Hysteresis model (MACH)* is based on a physical interpretation of the Preisach-based modelling and computes accommodation features compatible with experimentally observed accommodation phenomena. In fact, this model takes into account the transient instability of moving particles in the Preisach plane, that is, during this time, the particles are no longer composed of an up and down switched region separated by a staircase line. When both the standard deviation and the mean interaction field are magnetisation-dependent, the physical assumptions that led to the moving model are no longer valid. The variable variance Preisach model, considered to be the most general Preisach Model, takes care of this phenomenon that appears when characterising perpendicular recording media.

Our knowledge source for the hysteresis model used in this thesis, instead of being a true measurement laboratory, is a 3D moving vector hysteresis computational model by Oti [Oti,

.991] and Davidson [Davidson, 1996]. This model transforms a sequence of vector magnetic ield inputs, H, into a sequence of vector magnetisation, $M(H)$. But it is important to note that he specific predicted sequence of magnetisation depends on the values of the input parameters of the model. These parameters have been identified beforehand by using an appropriate method in such a way that the modelled sequence $M(H)$ fits the experimental magnetisation sequence. Validation of the model requires a good measurement database and the implementation of other material modelling techniques for comparison purposes.

We have used the results from a three-dimensional vector magnetisation model, along with applied fields and resulting magnetisation components in any direction in space, to acquire comprehensive magnetic hysteresis knowledge. We have also trained various neural network models. The usefulness of this type of model has been demonstrated through simulations related to a standard C-core problem and a ferrite dual-component magnetisation process. Certain issues related to the memory usage and computational speed of artificial neural networks and their capabilities to approximate non-linear curves, evoked in earlier works, have been successfully addressed in the present work.

2.3.8 Basic requirements for Finite Element solvers

Magnetic materials play a major role in the operation of many electromagnetic devices from transformers to recording systems. Structures which have been proposed for representing them have largely been based on either attempts to match the measured curves [Hodgdon, 1988] or phenomenological models which try to simulate the actual mechanisms of hysteresis [Vajda-Della Torre, 1996], [Ossard et al, 1995]. The latter models are necessary if the detailed operation of magnetic recording systems are to be simulated. However, in the more general situation, an

effective material model is required for use in general purpose finite element magnetic analysis systems. In these systems, the material model has to be applied to each element and, in a large three dimensional system, the number of elements needing to be able to track the material behaviour can be in the thousands or tens of thousands. Additionally, during the non-linear iteration process and the overall time evolution, it is necessary to evaluate the appropriate $M-H$ relationship many times for each element. Thus there is a requirement for a material modelling approach which is both memory efficient, to minimise the required memory when a large number of elements are involved, and fast enough to speed up computation. In an attempt to meet these two criteria, the use of a neural network has been examined.

The general procedure to implement finite element analysis program involves mainly four steps [Silvester-Ferrari, 1996]:

1. Discretisation or meshing where the domain to be analysed translated into a number of nodes and triangles in two dimensional problems or tetrahedra in three dimensional cases.
2. Evaluation of potentials in each element
3. Finding the energy in each element
4. Assembling the elements to get the energy of the whole system
5. Minimisation of an energy functional

Iterative solution techniques of a large set of simultaneous equations, which are highly dependent on the material properties, are required to achieve the minimisation of the energy functional. Newton-Raphson method are widely used to achieve this task. Special attention has been taken to insure that the material model proposed in this thesis copes, in terms of accuracy and convergence, with the iterative search processes taking place.

Chapter 3

3 Principles of Inductive Learning and Fundamentals of Neural Networks

3.1 Inductive Learning

Learning occurs when a system constructs or modifies its knowledge representations McCarthy [McCarthy, 1968]. Based on this definition, the acquisition of any kind of information may be seen as a form of learning. Learning is also achieved through inferences. The three basic types of inferences distinguished in the literature are *deductive*, *analogical* and *inductive* [Michalski, 1993; Kodratoff-Bares, 1991].

Inductive learning systems are based on the process of reasoning from a given set of facts according to certain general principles or rules [Durkin, 1991]. Induction processes derive knowledge from a set of examples and, thus, a record of past events must be available. Knowledge acquisition based on the system's learning from its experience is also called example-based learning or induction [Dym, 1991]. Michalski and Kodratoff [Michalski-Kodratoff, 1990] outline a good survey of the classification of learning processes (Figure 3.1). In this thesis, we have selected an *inductive based inferencing* mechanism that acquires knowledge from a set of examples by using a multi-layered feed-forward artificial neural network paradigm. To be efficient, a knowledge acquisition task should be automated. This automating process increases learning competence as well as logical capabilities and also reduces human resource costs needed for the development of Knowledge Based Systems [Buchanan et al, 1983]. As shown later through this thesis, it is desirable that a trained multi-

ayered artificial network generalise and abstract information from the training set of :xamples and reuse it in different, but analogous situations.

Generalisation is related to interpolation [Wasserman, 1993]. It is a statement about a class of objects that also applies to members of that class. For a computational learning system, an economical storage of information must be available [Coyne et al, 1989]. When generalisation takes place, the learning system recognises examples that have never been presented to the network before. Such a characteristic is essential for innovative and creative design processes requiring the ability to build general principles from knowledge about specifics and apply them as needed.

On the other hand, deduction requires that a set of information, generally expressed in an explicit way, be defined first, before any derivation of new knowledge occurs or before any consistency checking of the information takes place. This presents some limitations because previous information has to be judiciously chosen and expressed qualitatively and quantitatively in order to represent the whole situation.

The use of a *deductive inferencing mechanism* to acquire knowledge initially requires a good formulation of the background knowledge that may require sustained interaction with a human expert. This method of logic inference, nonetheless, presents some drawbacks because of the unavoidably incomplete character of the required adequacy of the inputs (i.e. all the information should be explicitly formulated by the system designer). Another drawback is that laborious search strategies, executed mostly across "decision trees" (the representation paradigm in this case), are always needed for deductive systems. Finally, the deduction inference does not allow the handling of some types of knowledge, such as modelling the non-linear mathematical relationship between magnetisation and the magnetic field. Deductive learning systems have shown their limitations in the earlier rule-based expert

;ystems. The *analogical inferencing,* approach, a case-based reasoning that transitions)etween case replay, case adaptation, and general problem solving has drawbacks similar to :he deductive systems described earlier. Both of them are not used in this thesis.

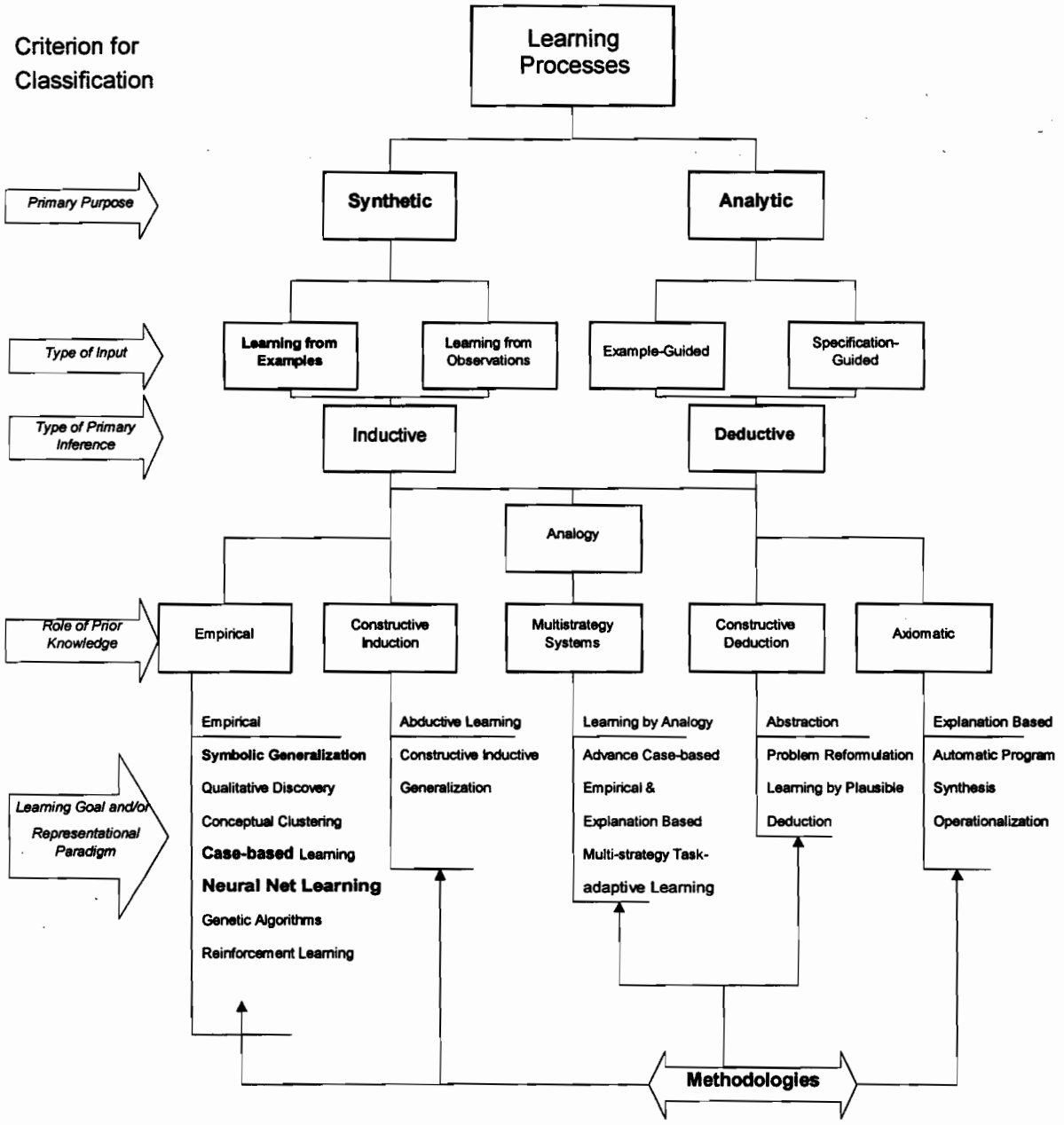


Figure 3.1 Classification of Learning Processes [Kodratoff, 1990]

To summarise, our approach of using an artificial neural network paradigm to acquire magnetic material behaviour knowledge, i.e. a mathematical knowledge, copes with a

complex task where all the underlying physical principles are not explicitly known. Also, because of the interaction of many complex factors in the identification and modelling of a magnetic material, it is difficult to systematise the process by the application of a mathematical approach alone. Therefore, we have proposed applying a neural network-based approach, coupled with the mathematical Preisach-based technique to counter current costly modelling and implementations of hysteresis phenomena to solve electromagnetic problems by finite element methods.

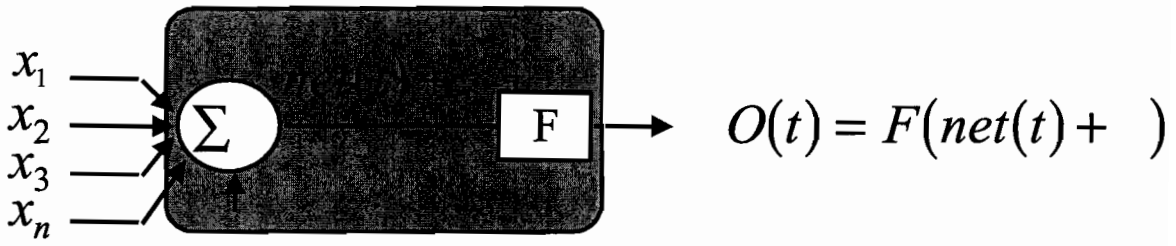
The resurgence in neural network research has facilitated the growth of a whole range of methodologies to acquire and store data inductively for given material models and to use these models within a finite element solver. Now that neural network techniques are reliable, efficient and are providing a fast computational framework, it is possible to include them in magnetic problem-solving tasks.

3.2 Fundamentals of Neural Networks

3.2.1 Background

Artificial neural networks have been inspired by the neurological architecture and operation of the human brain formed by billions of interconnected biological neurons simultaneously activated or inhibited. Hence, a computation and knowledge representation paradigm consisting of a model of an individual neuron was constructed by Warren McCulloch and Walter Pitts in 1943 [McCulloch-Pitts, 1943] In the *McCulloch-Pitts* model, each neuron's activity x_i is determined by the sum of weighted inputs. The outgoing signal from neuron i is restricted to one or zero, a binary value.

An artificial neuron is represented in Figure 3.2. A neuron receives *n* inputs from the environment or from other neurons and produces a single output unit to the environment or to other neurons. Each neuron consists of a summing unit followed by an "activation function". The other mathematical operation performed within an artificial neuron consists of adding biases to tune its overall activity to a desirable behaviour. The networking of artificial neurons forms the so-called artificial neural network. Figures 3.3 represent a typical feedforward neural network topology.



$$net(t) = \sum_{i=1}^{n} x_i w_i$$

Figure 3.2 The Artificial Neuron; The Processing Unit

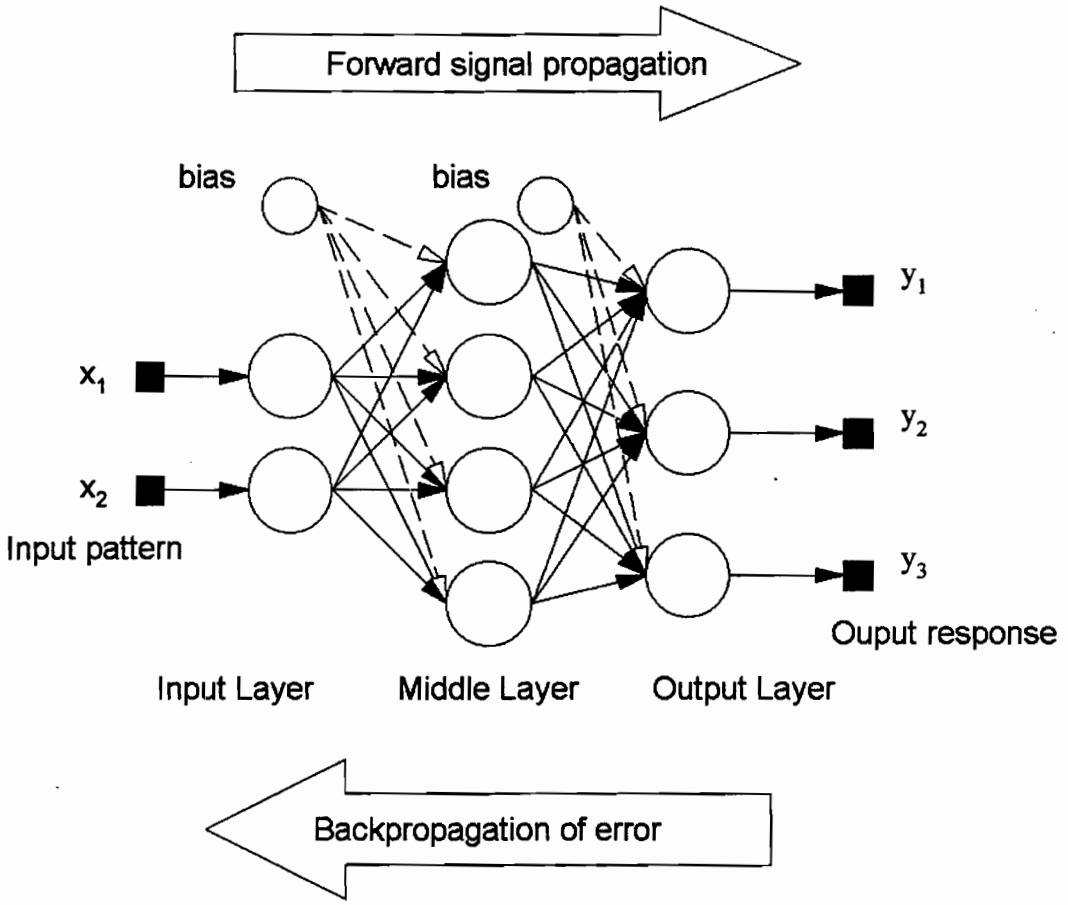


Figure 3.3 Topology of a Feedforward Neural Network. This figure shows a [2-4-3] three-layered network with 2, 4 and 3 neurons in the input, hidden and output layer respectively.

The Warren McCulloch and Walter Pitts [McCulloch-Pitts, 1943] landmark work was able to model some basic logic functions such as the AND, OR and NOT functions. By introducing so-called self-organisation capabilities to the network, Rosenblatt extended the McCulloch-Pitts model and developed many variations of the *perceptron*, the first artificial neural network [Rosenblatt, 1958]. Minsky and Papert showed that a single-layer net can learn only linearly separable problems, i.e. functions such as the exclusive OR (XOR) function are beyond the capability of a single layer net. [Minsky-Papert, 1969].

To overcome the shortcomings of the single layer net pointed out by Minsky and Papert, the research efforts by the Anderson-Rosenfeld and Rumelhart-McClelland teams renewed the passion for the use of artificial neural networks [Anderson-Rosenfeld, 1988; Rumelhart-McClelland, 1986]. These include the multi-layer feedforward neural network trained with the backpropagation of error learning rule, the Hopfield net [Hopfield, 1982], the Kohonen net [Kohonen, 1982], the probabilistic neural network (PNN) [Specht, 1990] and a number of variants that are all of great interest for various design tasks and are able to provide reasonable solutions to real-life problems.

In this thesis we will take advantage of the properties of artificial neural networks which provide a means of representing complex multi-dimensional surfaces in a uniform manner. A trained network can be considered as providing a form of least squares fit to the hypersurface defined by the input and output vectors. It has been shown that this architecture can approximate any function over a compact set [Pao et al, 1989; Funahashi, 1989; Hornick et al, 1989]. However, this statement should be considered with care. Proof that three layer neural networks are universal approximators usually includes the following conditions [Cybenko, 1989]:

1. The function is to be approximated over a finite domain.
2. The function is defined and continuous over that domain.
3. The number of hidden units is unlimited (but finite).
4. The size of weights is unbounded (but finite).
5. The function is approximated to some arbitrary small, but non-zero degree.

Artificial neural networks could be classified according to the following criteria:

- Architecture or topology (feed-forward, laterally connected, single-layer, multi-layer, recurrent).
- Learning or training strategies (supervised, unsupervised or reinforcement manner).
- Learning rules
- Activation functions that are used
- Metrics used for error computations
- Validation technique.

Some of these criteria will be explored a bit later in the present work.

3.2.1.1 Learning Strategies

Learning laws or rules and training algorithms are what drive an ANN towards a desired behaviour, i.e. their generalisation ability. Two types of learning systems – *supervised* and *unsupervised* - are used to train ANNs. In a *supervised* learning system, an output evaluation is provided and the desired response is encoded in a cost function. The network is given the training data consisting of both the input pattern and the desired output it is supposed to induce. After many trials, the network tunes itself to achieve the mapping from the input to the output.

In an *unsupervised* learning system, input patterns are clustered according to pertinent properties that the network learns or discovers. The learning method is not based on a comparison with some target output. This interesting property is also known as self-organisation or, in other words, the network is considered autonomous. No desired response is given and no evaluation of output is provided. In this thesis, we have considered ANNs using

supervised learning systems and the backpropagation learning strategy and its variants, that represent the most well known approach.

3.2.1.2 Learning Rules and Error Metrics

The training processes can be implemented in three ways – gradient-based, deterministic and stochastic. Those chosen for this work are of the gradient-base type, a generalisation of the least mean square method. This technique can be summarised in three steps. The first step involves the computation of an error function based on the difference between the desired and actual outputs, the second stage is the determination of the contribution of each weight to the error and, ultimately, weights are updated to minimise the error. This last process of adjusting the weights as shown in Figure 3.4 is implemented by weakening weights that contribute to wrong outputs and reinforcing those that contribute to correct ones.

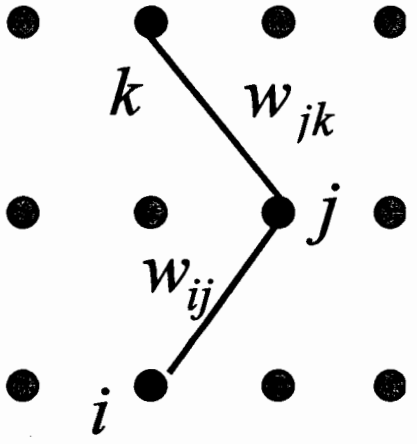


Figure 3.4 Training a Three-Layered Neural Network. The training involves adjusting the weights (,...,) using the appropriate learning algorithm.

The classical learning rule for standard feedforward neural networks is called the *backpropagation of error* [Rumelhart et al, 1986] and is the most popular supervised learning

echnique for multi-layered networks. The learning process is implemented by using the gradient descent-based algorithm. Backpropagation of error is a generic concept and many variants currently exist.

Levenberg-Marquardt algorithms [Hagan-Menhaj, 1994] and *Quasi-Newton,* also known as *Brayden-Fletcher-Goldfarb-Shanno* (*BFGS*) algorithms [Fletcher, 1987], are the other learning mechanisms applied in this thesis. These learning algorithms are treated in greater detail in Appendix 2. Major handicaps mentioned in the literature are the local minima problem and the sluggish speed of the standard backpropagation of error training process. The former refers to the optimisation problem over a complex error surface that can be resolved by using computational methods, such as genetic algorithms, simulated annealing or certain stochastic gradient descent techniques.

3.2.1.3 The Activation Functions

Each neuron consists of a summing unit followed by an "activation" function, this is used to limit the outputs of the network to a specific range and is designed to be differentiable in order to speed up the network training. The activation function can take many forms including sinusoidal, sigmoidal, gaussian and similar activation functions. Its purpose is to limit or cluster each neuron's activity in a specific region. Yet the only necessary condition to be satisfied by these functions is differentiability. However the choice of a particular activation function and its parameters is crucial for a given neural network to acquire knowledge. In this thesis, we observed that the hyperbolic tangent function in the hidden layers as well as linear functions in the output layer outperformed all others. Figure 3.5 illustrates some typical activation functions.

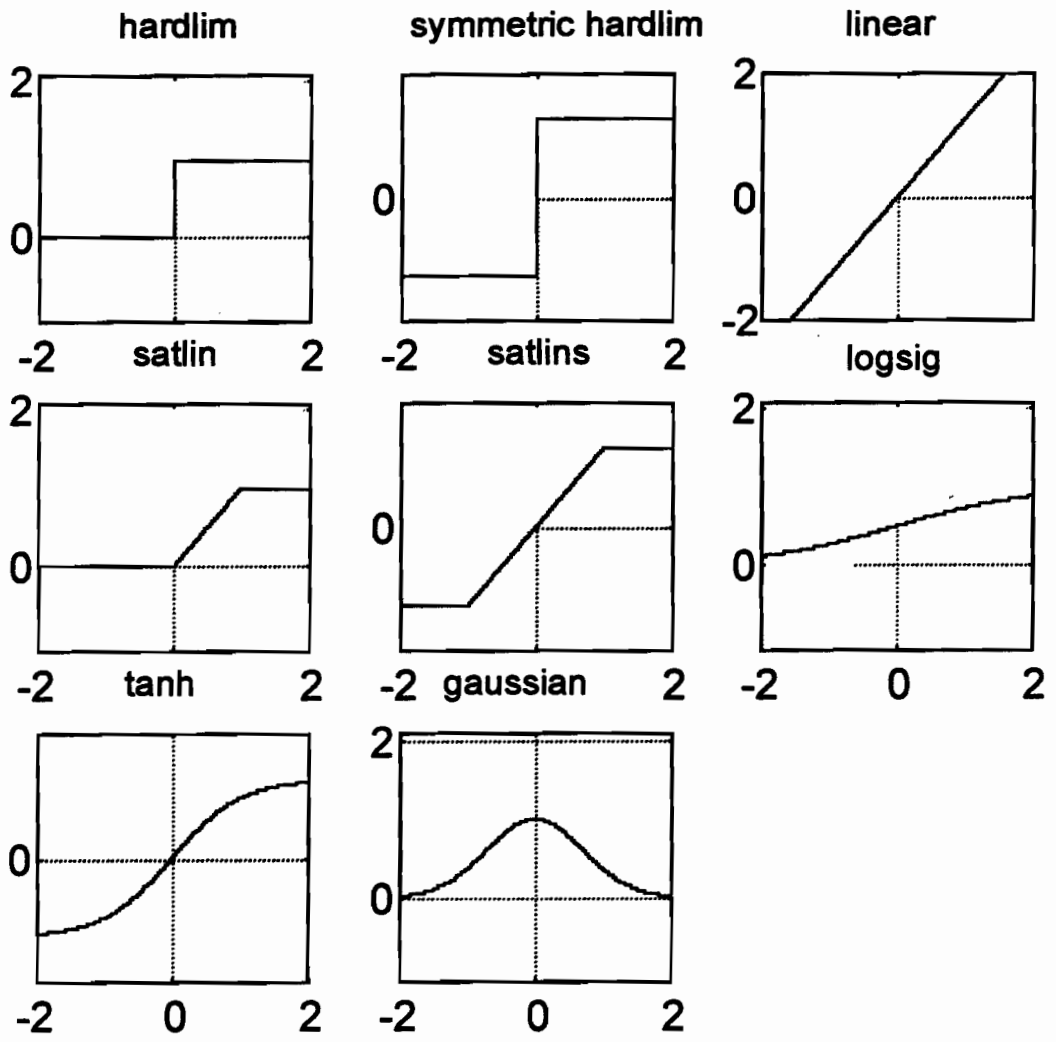


Figure 3.5 Typical Activation Functions

3.2.1.4 Artificial Neural Network Validation Techniques

Cross-validation is the most frequently used algorithm for combining training with validation techniques [Masters, 1995]. Numerous methods of cross-validation exist. A typical method is provided in Table 3.1. The main purpose is to reduce the bias caused by estimating the

eneralisation error due to the training set, rather than to test the robustness of the model. The ɛrm "generalisation" refers to the ability of neural networks to produce reasonable outputs for ıputs not encountered during the training process.

ʌ simple approach to cross-validation consists of dividing the training set into three parts: *raining, testing* and *validating*. The Artificial Neural Network (ANN) is taught using the raining set, but is periodically tested with the test set. Performance measures on the test set ıre used to determine when to terminate the training process. Finally, the use of the validation ɛet establishes how well the ANN really performs. Many implementations of ANN training nclude automatic training and testing methods, but usually the validation process parameters ıre virtually non-existent and the user must develop them.

Table 3.1 Typical Validation Steps

Validation Technique
1. Randomly partition the data into f statistically equivalent subsets.
2. Repeat k times:
a. Train on k-1 subsets.
b. Test on the "holdout" subset.
c. Accumulate the moment and/or rank error statistics.
3. Use the results to estimate the summary statistics of the generalisation error.

3.2.2 RBF Networks

Radial basis function (RBF) networks are attracting a great deal of attention due to their rapid training period, their potentially generic applications and their inherent simplicity. They have

nique features that are most noteworthy in fitting multivariate functions and are part of a roader class of interpolation techniques. The RBF interpolation process does not require the ivision of the domain into a mesh of sub-domains. This class also includes a host of similar aradigms directed towards complex mapping tasks where the mapping is continuous. A ypical radial basis function neural network consists of three layers of neurons: input, hidden nd output. It is a fully connected feedforward-like perceptron architecture in which the utput units have a linear activation function. The network paradigm is based on the simple ntuitive idea that an arbitrary function $\hat{y}(x)$ can be approximated as the linear superposition f a set of localised basis functions, $\varphi_i(x)$, expressed by the following equation:

$$\hat{y}_i = \sum_j w_{ij}\varphi_j(x) \tag{3.1}$$

where $\varphi_i(x)$ is a radially symmetric function, called a "kernel function", centred on the *i*th lata point and x is the corresponding input. A common basis function is usually the bell-haped gaussian function represented by

$$\varphi_j(x) = e^{\mathbf{D}_j^2/2\sigma_j^2} \tag{3.2}$$

Other kernel functions that have good theoretical backing are thin plate splines, $\varphi(x) = x^2 \log x$, and multiquadric and inverse multiquadric that are expressed respectively by $\varphi(x) = (x^2 + c^2)^{1/2}$ and $\varphi(x) = (x^2 + c^2)^{-1/2}$. The Euclidean distance $\mathbf{D}_j(\mathbf{x}) = \|\mathbf{x} - \mathbf{c}_j\|$ between the input vector $\mathbf{x}$ and $\mathbf{c}_j$ is determined by

$$\mathbf{D}_j^2 = (\mathbf{x} - \mathbf{c}_j)^T(\mathbf{x} - \mathbf{c}_j) \tag{3.3}$$

where the vector $\mathbf{c}_j$ represents the centres; σ_j, the standard deviation, describes the width or the spreading factor of the gaussian basis function at node j; and w_{ij} are the second-layer weights. The network is entirely defined when the parameter set $\{\mathbf{c}_j, \sigma_j, \{w_{ij}\}\}$ is determined. Each hidden unit has a localised response, that is, valid responses range within a limited zone, generally a circular shape, named the receptive field, σ being the size of the receptive field. The implementation should be adaptive (i.e. weights, centre location, widths are all tuned to the data).

3.2.2.1 Training RBF Networks

The goal of the training process is to find the appropriate parameter set $\{\mathbf{c}_j, \sigma_j, \{w_{ij}\}\}$ to map a given input vector onto a desired output vector efficiently and with an adequate degree of accuracy and generalization. Accuracy, of course, means minimisation of a given cost function and the generalisation was defined earlier. RBF networks are often trained via hybrid techniques, in which the hidden weights (centres) are first obtained by unsupervised learning, after which the output weights are obtained by supervised learning. Unsupervised methods for choosing the centres include:

- The distribution of the centres in a regular grid over the input space;
- Choosing a random subset of the training cases to serve as centres;
- Clustering the training cases based on the input variables, and using the mean of each cluster as a centre.

Various heuristic routines are also available for choosing the RBF widths. Once the centres and widths are determined, the output weights can be ascertained very efficiently, since the computation reduces complex data to a linear or generalised linear model. The hybrid training

pproach can thus be much faster than with some classical non-linear optimisation methods, uch as the conjugate gradient descent algorithm, that would be required for supervised aining of all the weights in the network.

.2.2.2 Radial Basis Function Networks (RBF) compared to feedforward neural networks trained with Backpropagation or its variant algorithms (FFNN-BP)

Vhen executing complex approximation or prediction tasks, RBF and FFNN-BP exhibit imilar performances. However, the following considerations have to be taken into account:

- In terms of accuracy, RBF networks typically require 10 times or more data to achieve a precision equivalent to that of FFNN-BP.
- RBF networks require less training time than backpropagation nets.
- When used as function approximators, RBF net predictions are poor when dealing with extrapolation tasks. In fact, zones that are far beyond the input space are mapped onto low values by the RBF localised receptive fields.
- RBFs, without an enhanced tuning scheme, are unable to model the saturation region of a ferromagnetic material properly. In fact, in this zone, there could be many to one mapping obstacles where matrix singularity problems appear.
- RBF architectures with large numbers of inputs could lead to what is known as a curse of dimensionality as well as the above-mentioned singularity drawback [Haykins, 1994].
- The need for large training data for RBF networks, as revealed earlier, also implies an enormous amount of computer memory for training at the first stage. This also entails a number of time-consuming pre-processing tasks such as clustering methods. These tasks range from classical statistical methods to neural networks, fuzzy sets, wavelet theories and genetic algorithms [Masters ,1993].

.2.3 CMAC Networks

'he *CMAC,* first developed by Albus [Albus, 1971], stands for *Cerebellar Model Articulation :ontroller* or *Cerebellar Model Arithmetic Computer*. This CMAC model belongs to a class ›f contents-addressable-memory-based neural networks that have an associative memory that levelops the ability to realise complex non-linear functions. A CMAC can provide a model 'or a physical system that has an unknown transfer function, just as other artificial neural ietworks. Among the numerous benefits of the properties a CMAC offers, we can mention he following:

- Local generalisation capability (The input vectors in "local learning networks," such as radial basis functions (RBF) and CMAC, act on a delimited region as shown by the coarse coding scheme for the data representation in Figure 3.6.);
- Fast learning without fixation problems (In fact, a CMAC network is considered as an alternative to backpropagation multi-layer networks.);
- Incremental training and output superposition capability (A CMAC network is an adaptive system that uses local learning and also permits incremental training. This enables the net to be retrained on-line to produce the correct signals for locally changed conditions.).

3.2.3.1 Data Representation

The operation of the conventional CMAC can be described in terms of a large set of overlapping, multi-dimensional *receptive fields* with finite boundaries [Albus, 1975a, 1975b, 1981; Edgar et al, 1991; Harris-Brown, 1994]. By using the CMAC technique, the problem space is discretized into several regions called *blocks* as shown in Figure 3.6, a two-

imensional problem space example. A quantization scheme is applied to each variable. ıreas formed by quantized regions are called *hypercubes*. The quantization for each variable s shifted by one interval. The conventional Albus CMAC can be seen as a technique with a ıasis function that has a constant value of "1" within a restricted area and "0" elsewhere. In his case the basis function is called binary. The area is a square, a cube, or a hypercube lepending on the dimension of the input space. For example, blocks into M, B, H, and R lescribing the variable v_1 and n, c, i and r for variable v_2, define the activated state where ıypercubes Mn, Bc, Hi and Rr are involved. Any input vector falls within the range of ;pecific local receptive fields (the excited hypercubes), and so falls outside of the range of nost of these hypercubes. The response of the CMAC neural network to a given input is the ıverage of the responses of the receptive fields excited by that input, and it is not affected by he other receptive fields that are not covering the state. The *generalisation width* is often :alled the *generalisation parameter c*, or referred to as the number of simultaneously excited :eceptive fields for each input. It could also be thought of as a quantization resolution. CMAC nakes use of a number of overlapping sets of receptive fields of this width. Hence, the CMAC has an *input resolution* of the generalisation width divided by the number of sets of :eceptive fields.

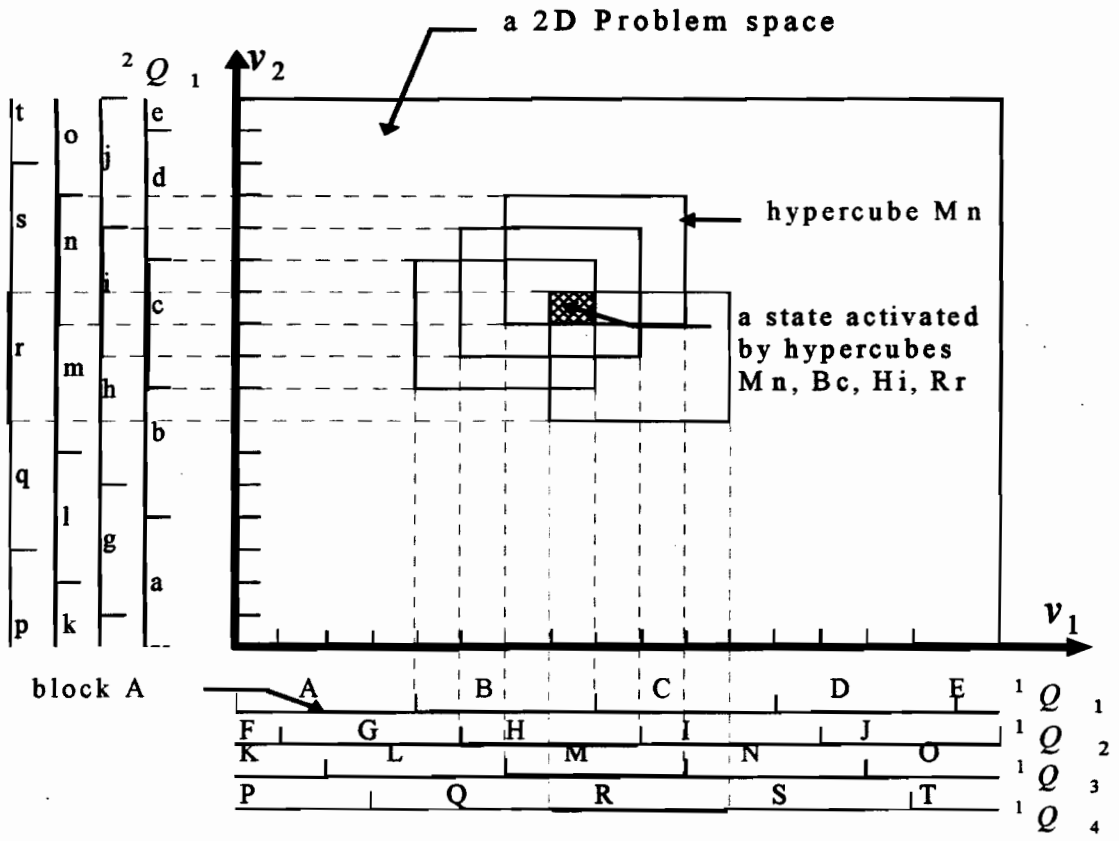


Figure 3.6 A Four Layer Block Division of CMAC for a Two Variables Example

3.2.3.2 A CMAC Computation

A CMAC network is an adaptive system that works by means of functional relationships. This is basically carried out by a series of sequential mappings as shown in Figure 3.7:

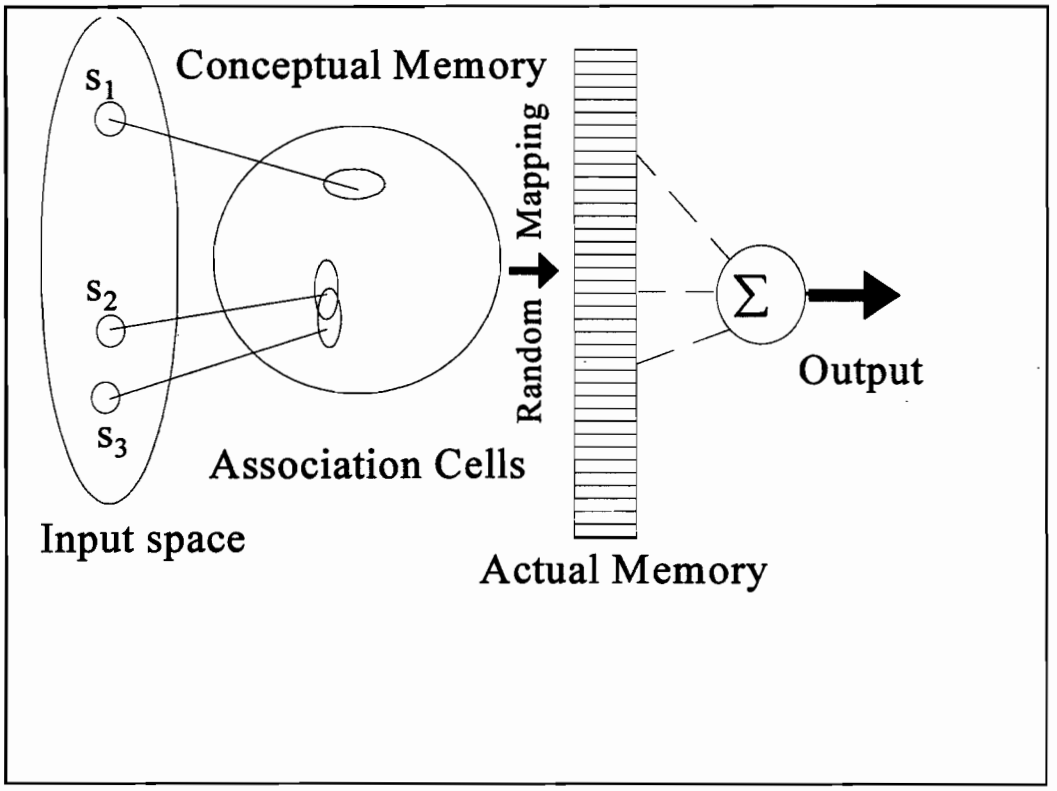


Figure 3.7 CMAC Mapping Operations

Consider a set of N-dimensional input spaces $S = \{\text{set of all input vectors}\} = \{s_1, s_2, \ldots, s_n\}$ with a generalisation parameter c, i.e. the identified number of simultaneously excited receptive fields. A CMAC operation involves the following steps:

Step 1: **Pre-processing the input vector (Normalisation)**

Each component of an input vector is first treated to get a set of normalised integer values obtained by dividing the original input set by an appropriate factor, called a *quantization parameter* ω_i, according to the following equation:

$$S' = \{s_1', s_2', \ldots, s_n'\} = \left\{\text{int}\left(\frac{s_1}{\omega_1}\right), \text{int}\left(\frac{s_2}{\omega_2}\right) \ldots, \text{int}\left(\frac{s_n}{\omega_n}\right)\right\} \qquad (3.4)$$

Step 2: **Determination of the vectors' virtual addresses**

The c receptive fields which consist of the set $S^{'}$ vector virtual addresses are determined using the modulus operator "%" by the expression A_i, that is an address in N dimensional space:

$$A_i = \left\{ s_1^{'} - \left(\left(s_1^{'} - i \right) \% \, c \right), s_2^{'} - \left(\left(s_2^{'} - i \right) \% \, c \right), \ldots, s_n^{'} - \left(\left(s_n^{'} - i \right) \% \, c \right) \right\} \qquad (3.5)$$

where the index i spans from a value of one to the generalisation width c which corresponds to the number of parallel layers. As mentioned earlier, Figure 3.6 shows 4 parallel layers for the 2D problem example.

Step 3: **Converting the virtual addresses to physical addresses**

This operation is also called hashing the virtual memory and its ultimate goal is to form the scalar physical addresses so as to reduce the required amount of computational resources (memory). In fact, the number of receptive fields in a space of dimension N can be very large and so cannot be implemented by allocating a physical address for each state space. The hashing operation consists mainly of a many-to-one mapping and uses some combinations of AND or/and OR gates to perform this task. For example, in the circuit shown in Figure 3.8, a simplification of the CMAC network mapping process, the internal layers consist of connected matrix such that each input vector (eg. X, Y, Z) activate exactly "*width*" number of AND gates and the output is summation of weights corresponding to activated AND gates. To summarise, the operation of obtaining the physical addresses $A_1^{'}$ can be written using the equation (3.6) as shown below:

$$A_1^{'} = \Xi\left(A_1\right) \qquad (3.6)$$

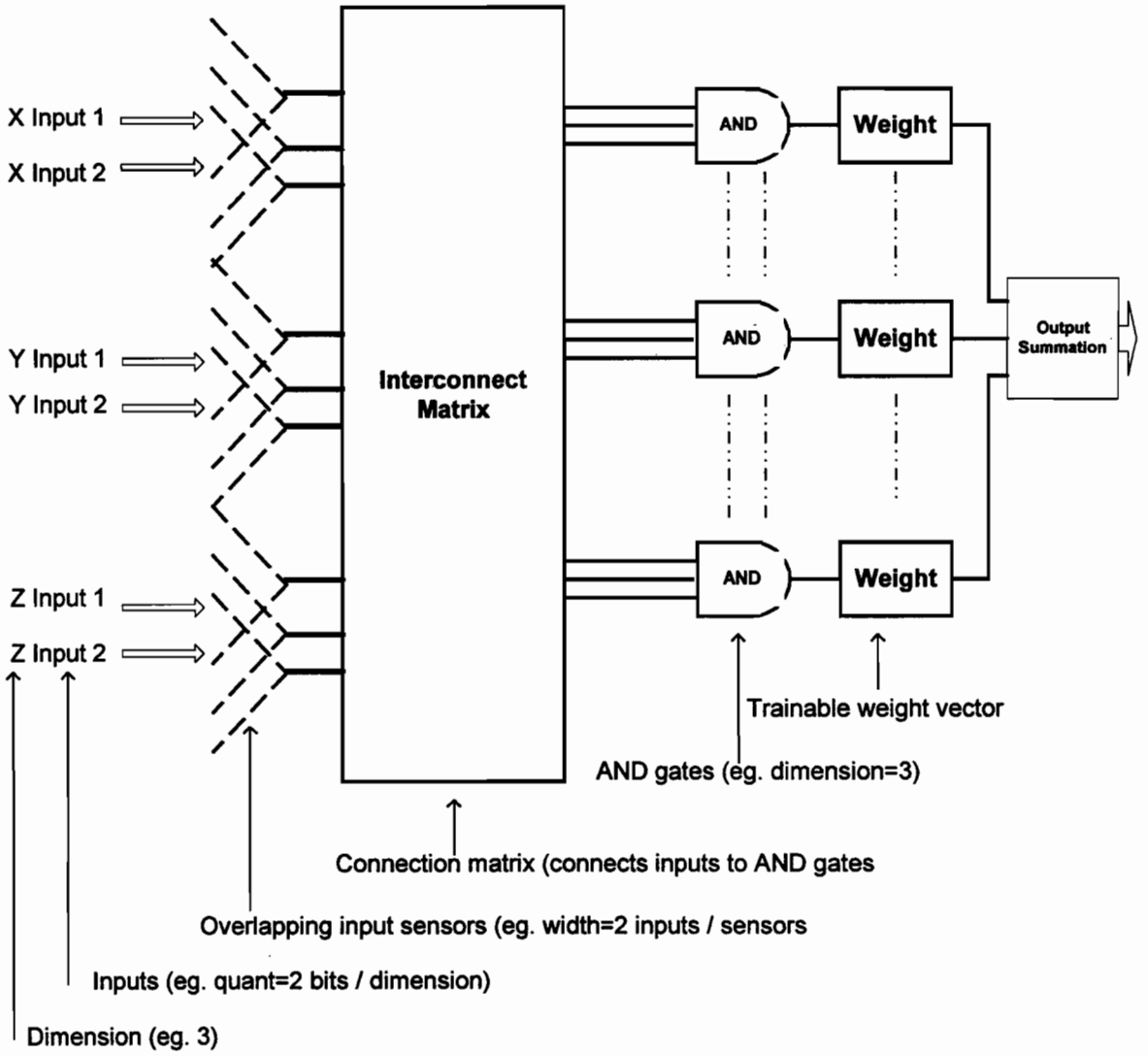


Figure 3.8 CMAC Layout as described in Ref. [Burgin, 1992]

Step 4 **Training CMAC Networks**

Training a CMAC network is a supervised process; i.e. the CMAC's output for various inputs is compared to the desired output. The output values obtained are then adjusted to reduce the mismatch by modifying the weights. The number of memory locations required depends on the degree of non-linearity of the system being modelled. Finally, the average addressed weights, also called the CMAC network scalar output $Y(s)$, is computed.

$$Y(s) = \frac{1}{c} \sum_{i=1}^{c} W_i(A_i) \tag{3.7}$$

The learning algorithm employed to update the weights is basically a variant of the LMS method and can be expressed by the formula:

$$\Delta W = \beta \cdot (Y_d(S) - Y(S)) \tag{3.8}$$

where β is the *training gain.*

This technique involves an adjustable mixture of fixed basis functions as well as a linear update rule. A Gauss-Seidel iteration algorithm of a linear system can also be efficiently used. The techniques of Radial Basis Functions (RBF) and CMAC can be combined in order to benefit from the merits of both. Figure 3.10 illustrates the concept of using gaussians instead of binary functions. In fact, as we have previously seen, while the conventional CMAC uses binary basis functions, the RBF uses mostly gaussian ones. The advantages of RBF networks include the continuity and differentiability of the approximate function. Similarly, a recent paper [Lin-Chiang, 1998] suggests an integration of CMAC techniques and weighted regression to achieve enhanced accuracy as well as output differentiability. However, non-differentiability of the output is the major drawback when using a conventional CMAC magnetic material model within a finite element solver. In fact, the output provided by a standard CMAC is not smooth, as is shown in Figure 3.9. Two other major problems related to CMAC networks are the collisions that may occur when the hashing operation is not appropriately carried out and of course, the huge memory requirements.

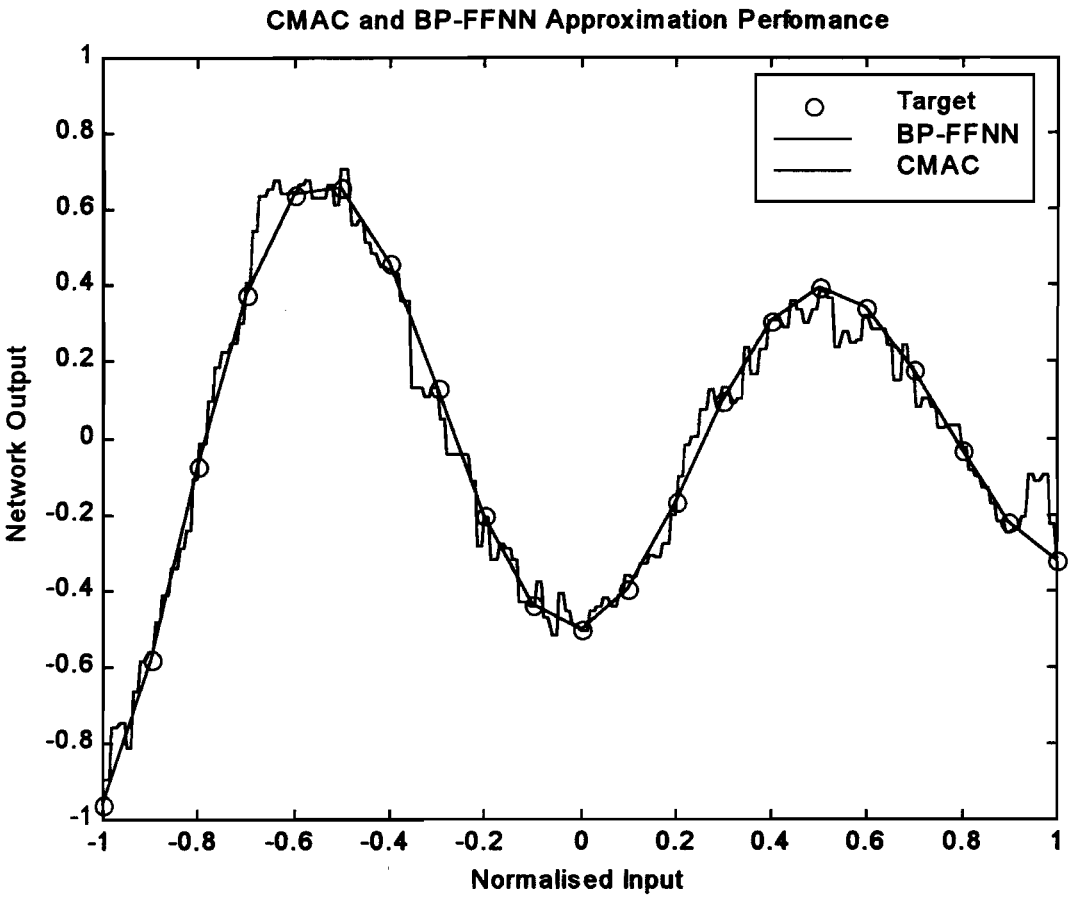


Figure 3.9 CMAC and BP-Feedforward Neural Network Approximation of a Damping Sinusoidal Waveform

Network Type	Sum Squared Error	Epoch	Number of Weights
FFNN.	0.0198747	536	16 (1-5-1)
CMAC	0.015	8	260

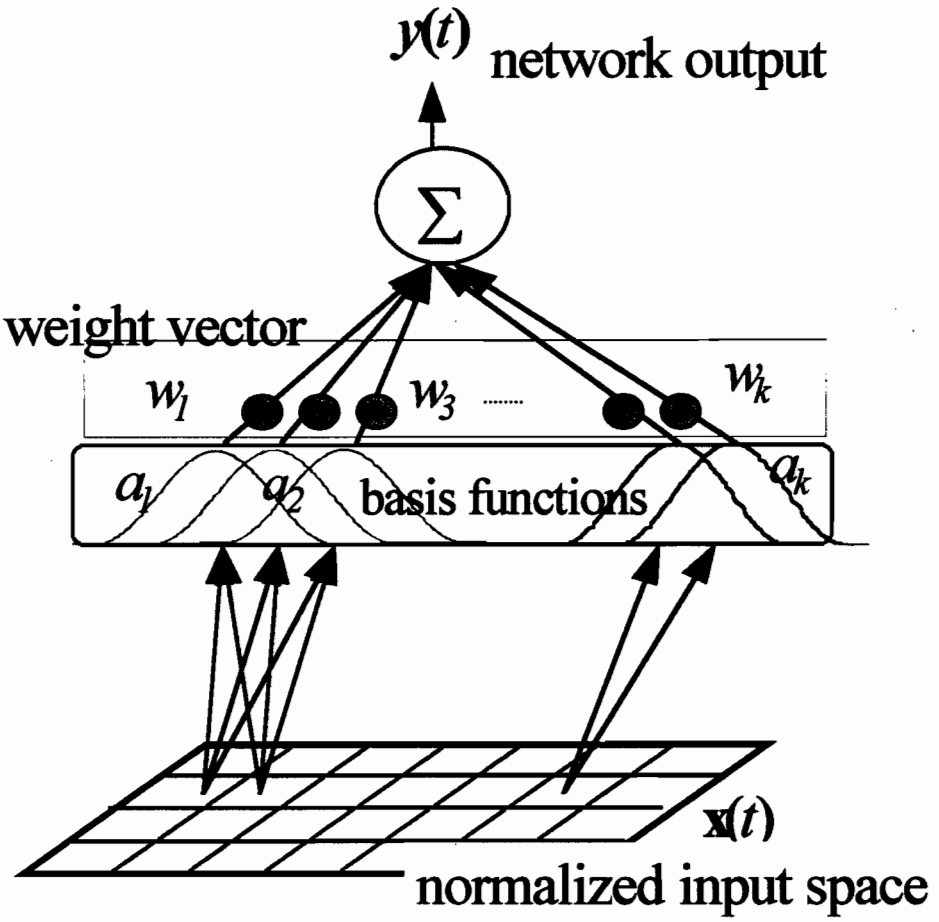


Figure 3.10 CMAC and Gaussian Basis Function

However, CMAC nets have advantages over the classical feedforward neural network trained with the backpropagation neural networks in terms of speed of learning, and the ability to cope with changing goals, so there is no need to retrain the CMAC network from scratch with new data.

3.2.4 Adaptive Logic Networks

Artificial Neural Networks tend to be "opaque" black boxes which doesn't affect simple material modelling, but designers are rather clueless about how they arrived at their

conclusions. The Adaptive Logic Network (ALN) approach is proposed by Armstrong [Armstrong et al, 1991; Atree 3.0, 1996] to avoid the opaqueness of conventional neural net approximating functions. The network estimates the parameters of linear functions which are combined by maximum and minimum operators to form a model. ALN-derived models can be used in conjunction with standard multiple regression programs to significantly improve the process of building statistical models. Furthermore, the fact that this approach can be looked at as a regression model identification device brings us to consider that a work similar to [Vajda-Della-Torre, 1994] can be accomplished much more easier using ALN. In this thesis, only a bit of work has been carried out to test the ALN ability for modelling non-linear behaviour of magnetic material. The net is trained to reproduce an $B-H-\Delta B$ relationship. The output of the program leads to a piecewise linear function. The domain of the learned function is divided by thresholds on the variables successively (i.e. by a decision tree), until the functions for the regions involve only a few linear pieces. The major drawback is that the number of function segments forming the generated decision tree can be very large. This means that our goal of representing a magnetisation function with only a few parameters does not hold in terms of the required memory storage. In order to use the ALN to solve a practical problem, a parser is also needed to retrieve the encapsulated knowledge. When the generated decision tree is large, this task is usually time-consuming and creates problems, particularly when a fast, easy to use model is needed. Some experimental results are outlined in Appendix 3. The results show a large number of piecewise decision trees generated when capturing an $M-H$ behaviour within an ALN framework.

3.2.5 The Use of Neural Networks in Magnetic Hysteresis Identification

3.2.5.1 Introduction

This section describes an approach to the problem of magnetic hysteresis identification based on the use of neural networks. The problem for many of the models currently being proposed for representing hysteresis in analytical systems is the identification of the model, i.e. the reliable determination of the appropriate parameters for a particular material. Some work has been published on the use of neural networks in this area and the present work seeks to develop this approach even further. In addition, the networks being considered are structured in such a manner that the information acquired by the network may be extracted at the end of the learning process, thus providing explicit values for the parameters, if needed. The basic methodologies being considered are based on the Preisach models and neural networks. Two techniques structured around the use of radial basis functions and a CMAC architecture are examined. The CMAC paradigm can allow incremental training and thus retraining the network for a new hysteresis curve can take place relatively quickly. Once the parameters have been established, the model can be used within a conventional finite element analysis system.

As we have said earlier, some mathematical hysteresis models use, as prior knowledge, the underlying physical principles of the devices, expressed in terms of partial integro-differential equations. Unfortunately, the physical principles governing most hysteretic elements are generally not well understood and there are no generic mathematical methods for solving the resulting strong non-linear equations. Among the representations those based on Preisach theory as well as the Jiles-Atherton counterpart, seem to be the most practical for hysteresis modelling. Although Vajda and Della Torre [Vajda-Della Torre, 1995] present the

relationship between the various Preisach-based scalar hysteresis models, there is still a need, from a practical point of view, for a concise classification of these models; i.e. a designer should know which model is the most suitable for a given design task. An embedded material selection process can then be ultimately automated within a design framework to match the requirements with the device configuration and the fabrication process as shown in Figure 3.11.

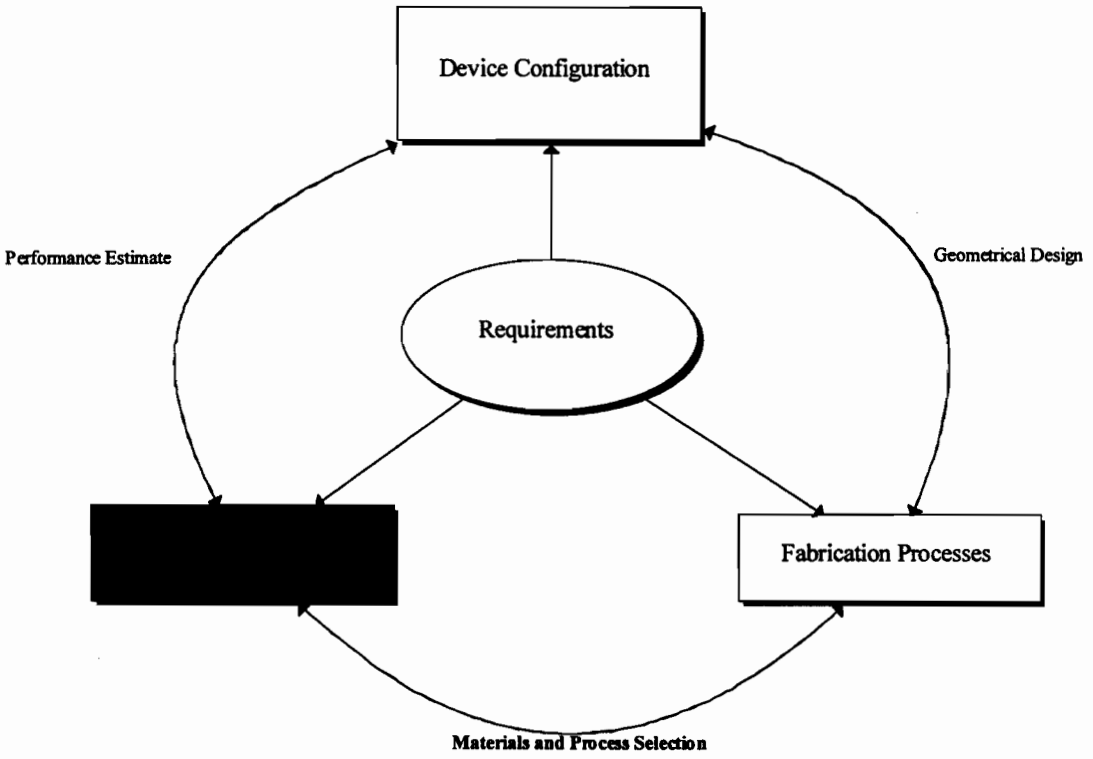


Figure 3.11 Material Model in The Design Framework

Another critical problem is the identification of the model's parameters. While some parameters are hard to identify, the use of empirical strategies makes the process a specific

problem-dependent one. Finally, a practical, suitable model to be implemented in a finite element process should be fast enough and not greedy in terms of the required memory for storage and computation.

In this thesis, we propose using the neural network paradigm to circumvent the drawbacks intrinsic to some classical approaches in regard to the identification, modelling or estimation of hysteretic systems and their corresponding parameters, as shown in Figure 3.12.

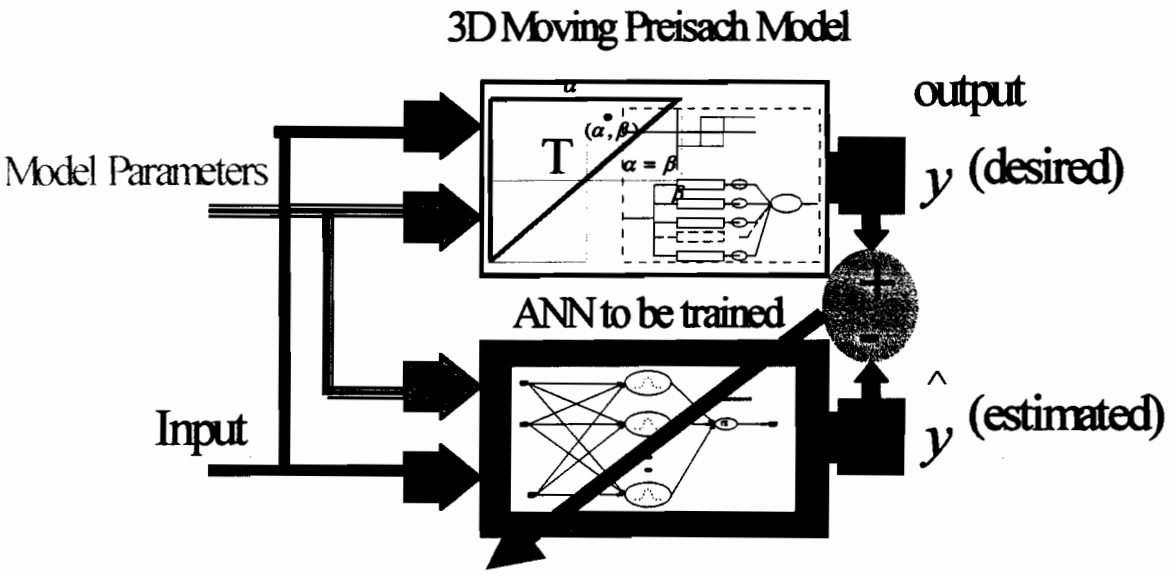


Figure 3.12 Learning a Preisach-Based Hysteresis Model using Neural Networks

The work published so far on the use of neural networks in this area is mentioned in publications by Ghaboussi, Alam et al, Madayam et al and Xu-Refsum [Ghaboussi et al, 1991; Alam et al, 1993; Madayam et al, 1994; Xu-Refsum, 1993], Saliah, Lowther and Forghani [Saliah-Lowther, 1995, 1997a, 1997b; Saliah et al, 1997, 1998, 1999]. The neural network approach uses a combination of experimental data and data provided by some previously identified mathematical models to ease the modelling of the global hysteretic behaviour of the system at hand. Four different types of artificial neural network structures are

considered for this application, namely the standard feedforward multilayer network (FFMLN), the Radial Basis Functions (RBF) [Chen et al, 1991], the Cerebellum Manipulator Controller (CMAC) [Albus, 1975; Miller, 1990] and the Adaptive Logic Networks (ALN) [Armstrong et al, 1991]. While RBF and FFMLN are appropriate for function and functional representations, CMACs are suitable for implementing compact and quickly retrievable lookup tables and the ALN can be used to fit functions to given data points or compute decision trees. The strength and attractiveness of the ANN approach to the solution of the hysteresis identification problem resides in its inherent capability to generalise from a relatively limited number of examples.

3.2.5.2 System and Parameter Identification

In terms of the Preisach approach mentioned earlier, "identification" relates to the determination of the model's parameters so that the model matches experimental measurements. The models proposed by Vajda and Della Torre [Vajda-Della Torre, 1994] require the determination of the moving parameter, saturation magnetisation, squareness, zero field reversible susceptibility, remanent coercivity and the switching field distribution expressed by the standard deviation in the irreversible and critical field. These are found by a combination of control systems and statistical physics theory. In the case of Ossart et al [Ossart et al, 1995] a heuristic trial and error strategy is used to direct the search of their 3D Moving Vector Preisach Hysteresis Model. This model is characterised by a Preisach density function portrayed by the three parameters of the Preisach density function and five other parameters. The experimental information used in the identification process is the limiting hysteresis and delta M curves.

In this thesis, the term "identification" means an adaptive process using the neural network paradigm as well as control systems and signal processing theory. In general, the problem

involved in material modelling is one of finding a mathematical model which both represents the material behaviour accurately and is matched to the required numerical processes. In this thesis, this search process is replaced with the more direct approach of storing information relating to the magnetic performance of a material in an associative network.

3.2.5.3 Example 1: Learning M-H excursion using Radial Basis Functions

The goal of this learning task is to capture the variation of a given magnetisation M subject to a driving H field using major loop behaviour. The raw data is made up of vectors consisting of five columns acquired from a simulation of a 3D Moving Vector Preisach Hysteresis Model [Ossart et al, 1995], namely (H_n, H_{n+1}, M_n, M_{n+1}, $\tilde{M}$) as shown in Figures 3.13 and 3.14. The RBF network was able to predict the variation of the magnetisation with good accuracy without overfitting, as shown in Figure 3.15.

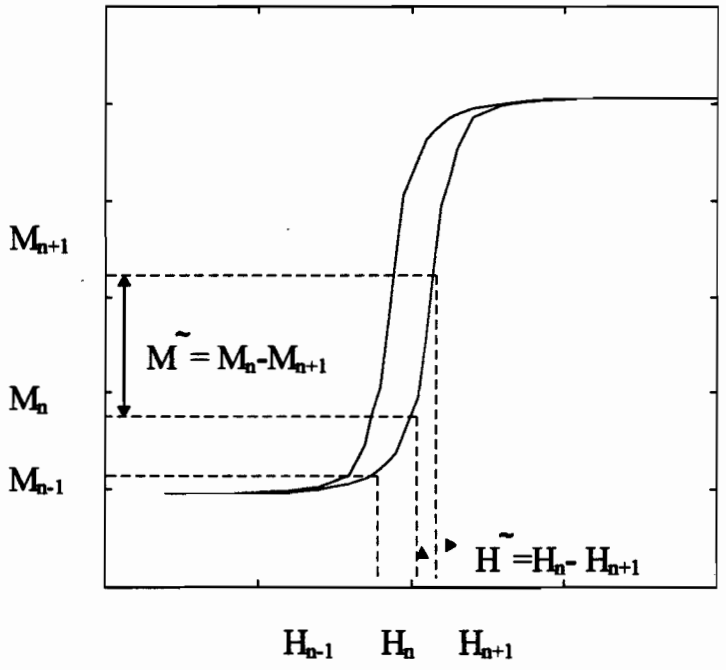


Figure 3.13 Training Data for a Neural Network

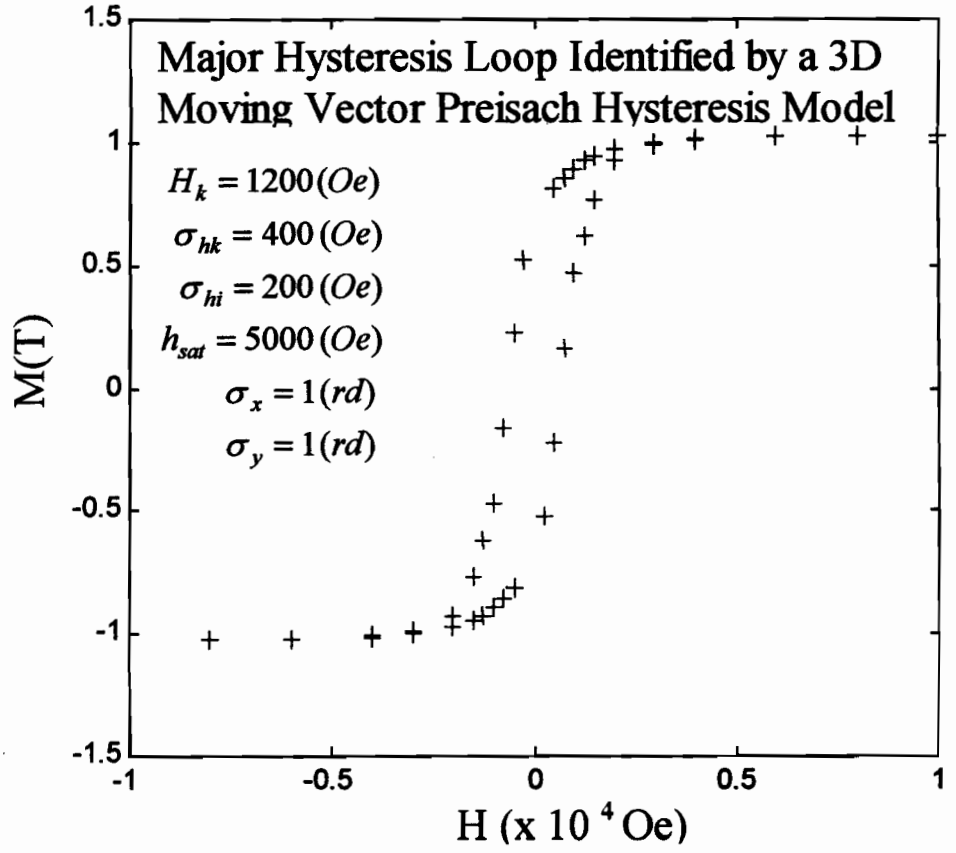


Figure 3.14 A Major Hysteresis Loop

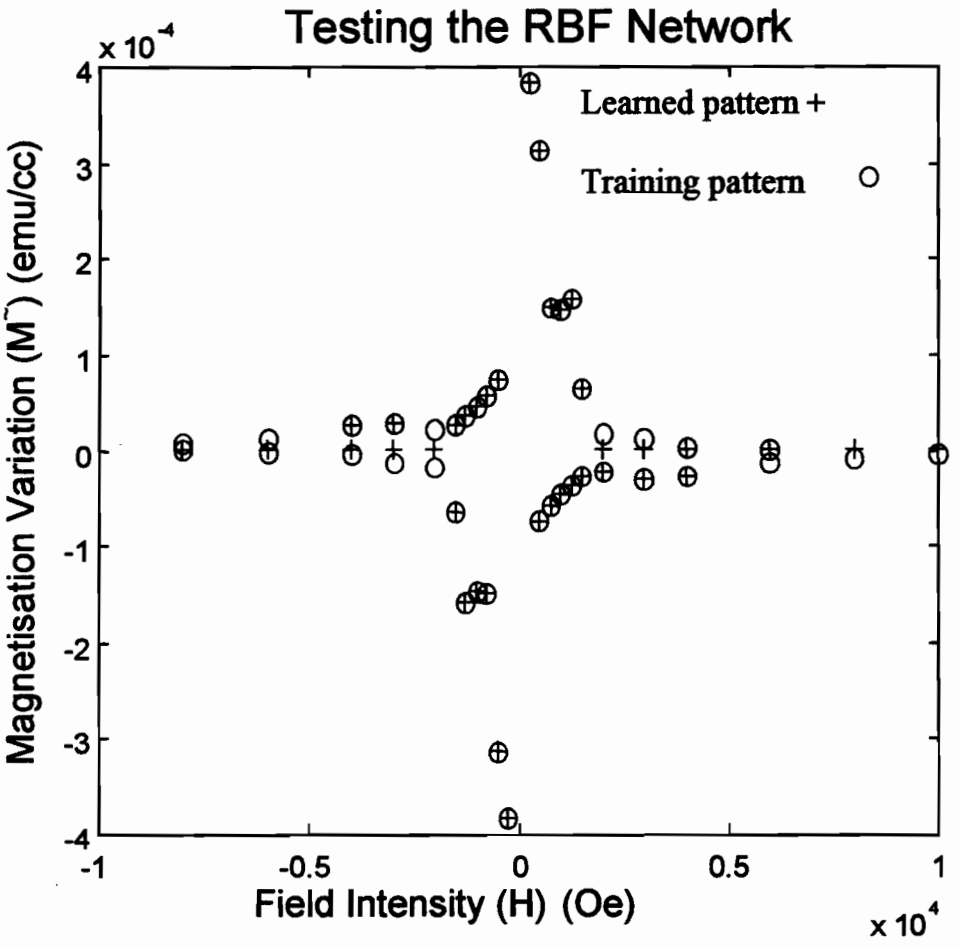


Figure 3.15 Learning Magnetisation Variation using a RBF

3.2.5.4 Example 2: Learning M-H excursion using CMACs

In this example, two 1D CMACs are used to construct a 2D CMAC. The training vectors are formed by a set of input pairs {M, H} and a corresponding differential variation of M ($\tilde{M}$). Here again, data are provided by the use of a Preisach Model-based simulator. The training data used for the CMAC system are shown in Figures 3.13 and 3.14 and learned responses are indicated in Figure 3.16. It can be seen that the CMAC system appears to be accurate and it exhibits good generalisation characteristics.

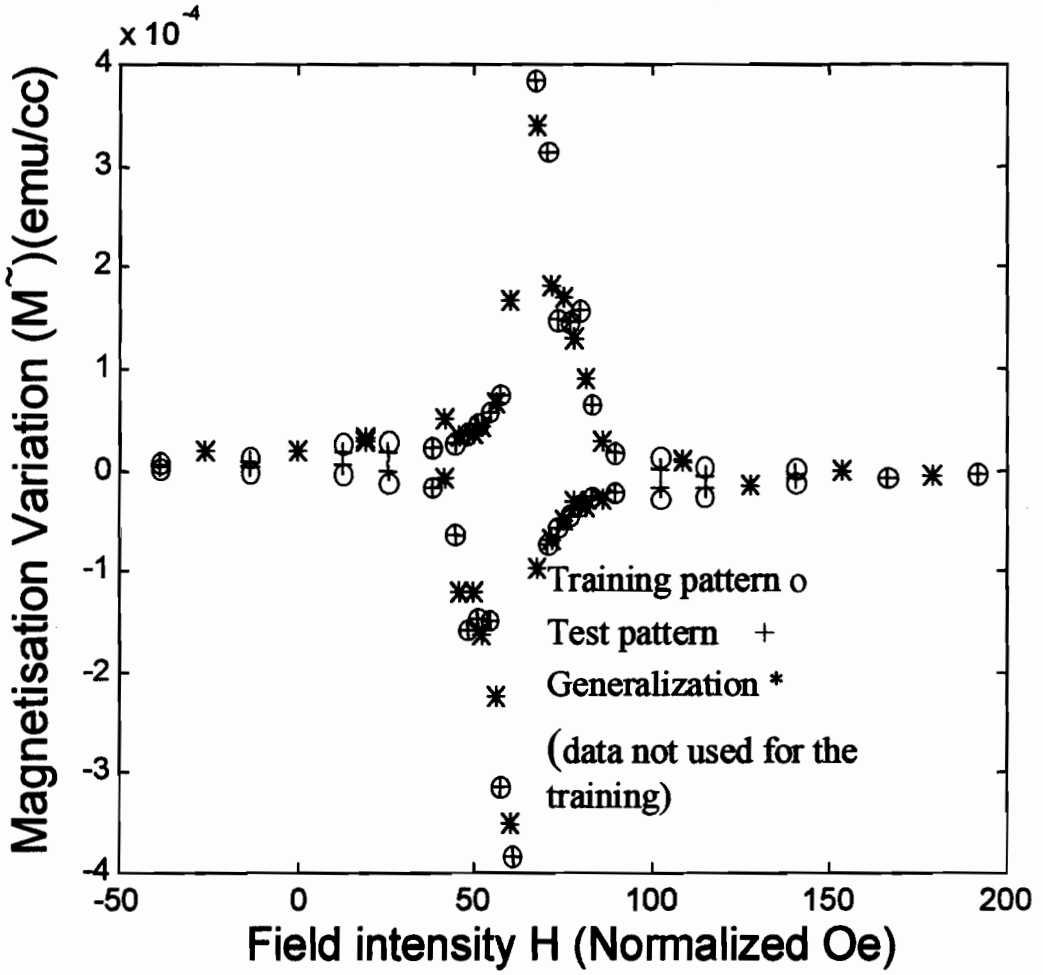


Figure 3.16 Learning Magnetisation Variation using a CMAC

3.2.5.5 Conclusion

The principles of the RBF and CMAC neural networks have been presented and their capabilities for magnetic hysteresis behaviour identification have been established. The utility of CMAC and RBF networks in modelling a non-linear system with unknown dynamics have been verified through numerical examples. However, it has generally been observed that experiments with CMAC networks show the need for finding techniques in adaptively selecting the quantization level, in order to cope with the function to be learned so as to reduce the required memory space and the speed of the learning process [Brown-Harris, 1994]. It is intended that future work will demonstrate that it is also possible to structure the networks in order to identify the necessary parameters for the Preisach model itself. This is also the subject of ongoing research and this task could be done after more comprehensive and relevant experimental data for a broad range of magnetic materials become available.

Chapter 4

4 Contribution to A Universal Representation of Magnetic Materials using Artificial Neural Networks

4.1 Introduction

This chapter discusses magnetic material modelling as applied to artificial neural networks, demonstrating the various requirements needed to make models compatible with a finite element solver interface.

First we will discuss models for materials that do not exhibit hysteretic behaviour. Then, we will treat models, both isotropic and anisotropic, as well as with and without hysteresis.

To establish our methodology and the adequacy of various neural network models, we will use standard feedforward networks and radial basis functions and test their ability to represent the desired behaviours.

4.2 The Need for Universal ANN material models

This section discusses the use of artificial neural networks as a uniform method for modelling the behaviour of magnetic materials, both isotropic and anisotropic, as well as with and without hysteresis.

In the past, the methodology used for constructing a computer model of a material has been dependent on the ultimate and highly specific goal of the analysis package and, since these have been both limited and specialised, the material models have been likewise. For example, if the material is to be modelled as non-hysteretic, the initial magnetisation curve can be handled by a polynomial, by piecewise linear segments, by a sequence of cubic splines, etc. If a range of temperatures is to be considered, then a different model for each temperature needs to be

constructed. For analyses where hysteretic properties become important, polynomials [Hodgdon, 1988; Cortial et al, 1997] have been used, as well as phenomenological models based around Stoner-Wohlfarth and Preisach [Mayergoyz, 1991].

Thus, in general, the modelling methodology has been dependent on the characteristics of the material needed for the analysis. While this has been a satisfactory approach, given that many of the analysis systems have been highly specialised, it is becoming unwieldy to handle multiple representations of the same material in analysis systems that are becoming much more generic in character.

Therefore, what is needed is a system offering the possibility of creating a uniform and universal model for all the properties of a given magnetic material, including hysteretic, thermal, frequency and stress effects. In addition, the data and computational requirements of such a system should be extremely low, resulting in a highly efficient approach from the point of view of memory and time.

4.2.1 Neural Network Design

Previous work involving the use of neural networks for modelling magnetic materials [Saliah-Lowther, 1997a; 1997b] has concentrated on the hysteretic properties and has shown that a conventional feedforward neural network based around perceptron-like neurons [Rosenblatt, 1962] is capable of modelling such aspects of a material. It would seem reasonable to expect that such a model will also be effective for non-hysteretic behaviour. However, little attention was paid in the previous work in regard to the construction of the input vector. To construct a feasible model, one should first establish if one or more hidden layers are required. In order to handle the non-linear nature of the data, the network requires at least one hidden layer. The size of the hidden layer can be determined in two ways. The first is a simple trial and error approach, i.e.

neurons are added to the layer until the network can provide a close fit to the training data. The second is to use a pruning algorithm [Hassibi-Stork, 1993] to remove neurons without affecting the accuracy of the representation. The goal of the network design, then, is to create a structure that can handle the full range of material characteristics including anisotropy and hysteresis. The training system has to be constructed so that an effective model is created for the desired application.

Within a finite element analysis system, the material model is subjected to an arbitrary driving field, H. Thus the training scheme for the neural network necessarily includes teaching it an appropriate response to each arbitrary H value.

4.2.2 The Input Vector to the Network

The design of the input vector for a neural network is absolutely critical. If the inputs are not independent, then the training time can increase, with little benefit to the modelling itself. If the input set is incomplete, then the result will not provide an accurate representation of the system. The proposed method, if it is to eventually handle hysteretic as well as non-hysteretic materials, has to retain a history of the behaviour under an arbitrary driving field, H. For a non-hysteretic material, only one input is needed and only one output. If the system is anisotropic, then three inputs are required - the three orthogonal components of the magnetic field - and three outputs need to be considered. If there is no coupling between the three principal directions, i.e. if the permeability tensor has no off-diagonal terms, the system is in actual fact the equivalent of three independent networks.

However, when hysteresis is considered, the problem becomes somewhat more complex. At any point in the M-H plane, the next state depends not only on the current state but also on the previous state. The current state alone is not sufficient because, for any given (M, H) pair, there

are a large number of possible M-H trajectories that pass through the point. Thus, enough information must be provided to determine which trajectory is being followed. The minimum is to determine two states in order to define the curve. While this is overkill for a non-hysteretic material, the same input vector can handle both hysteretic and non-hysteretic materials, providing a uniform method for handling all forms of a given material. The complete input vector for a single component of H then consists of 5 variables: the previous (M, H) pair, the current (M, H) pair and the next H. The output is the next value of M.

4.2.3 The Network Architecture

The architecture is a feedforward neural network trained with a variant of supervised learning to determine the parameters (the weights and biases). Five (5) inputs and one (1) output are enough to capture and represent the material behaviour within a finite element solver. However, the number of units in the hidden layer depends on the complexity of the problem and the required degree of accuracy. The number of units in the hidden layer doesn't affect at all the generality of the proposed neural network interface for a finite element solver since this information is provided only in the external initialisation file described later in section 4.4. The goal, from the point of view of the analysis system, is to determine the appropriate value for M, i.e. the magnetisation level, given the current value of H, the magnetic field. While, from a number of our experiments , six hidden units are enough to represent an anhysteretic material with an infinitesimal margin of error as shown in Figure 4.6, the five twelve and one ([5-12-1]) model is sufficient to represent magnetic materials of all kinds.

The final architecture for each magnetic axis of the material consists of five inputs and twelve hidden units with a hyperbolic tangent activation function and one output to be predicted. This

[5-12-1] model provides a universal framework, since the same interface can be used for all types of materials, isotropic or not.

4.2.4 Training Sets

Some level of pre-processing is always needed to generate an appropriate input set for a neural network used in a training set. In the particular case of magnetic material modelling, the training is critical and must take into account the way in which the neural network will be expected to function within the analysis system. A finite element solver will tend, at least in the early stages of the process, to present random magnitudes of H and thus the network should be able to work effectively in such a context. In the light of work on time series prediction methods using multi-layered feedforward networks, we have developed our own way of better generating and sampling our input to the neural networks. Selecting the appropriate training sets has been a very lengthy process and is still a task that requires special consideration. A number of methods are explained or referred to by Shimodaira [Shimodaira, 1996], including the "moving window data learning" method (MWDL) [Peng et al, 1990] and a "similar data selective learning" method (SDSL) [Peng et al, 1992] . The pre-processing scheme used in this case is described below:

1. The inputs and the output are scaled in such a way that the maximum values of H and M are equal to one.
2. A lookup table is created and more data are generated using a cubic spline based model.
3. The inputs are processed so that a large spectrum of data with diversified amplitudes and sequences can be used to capture the random nature of a finite element solver's requests.
4. The data is resampled at a lower rate, after low pass filtering, using a Chebyshev filter. This procedure contributes to appropriately reducing the amount of data to be used for the

training and aims at achieving a good generalisation. (Validation tests will be done later using the whole set of data).

Figures 4.1 to 4.5 illustrate the above process. And Figures 4.6 to 4.13 present some successful experimental modelling results achieved with a number of training algorithms used for the task. Those based on a variant of the Levenberg-Marquardt approach [Murray-Wright, 1981; Hagan-Menhaj, 1994] and on the BFGS described in Appendix 2 outperformed the standard gradient descent backpropagation. Special care is taken to verify the generalisation ability of the models when a Newton-Raphson, a non-physical search, process is carried out on a limiting ascending or descending hysteresis branch, as shown in Figures 4.11 to 4.13.

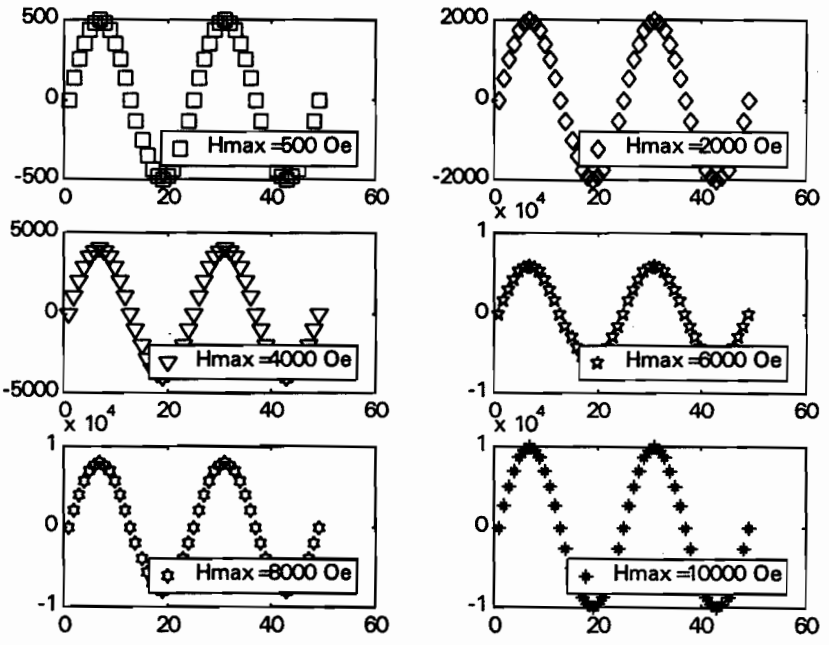


Figure 4.1 Diversified Driving Field Amplitudes

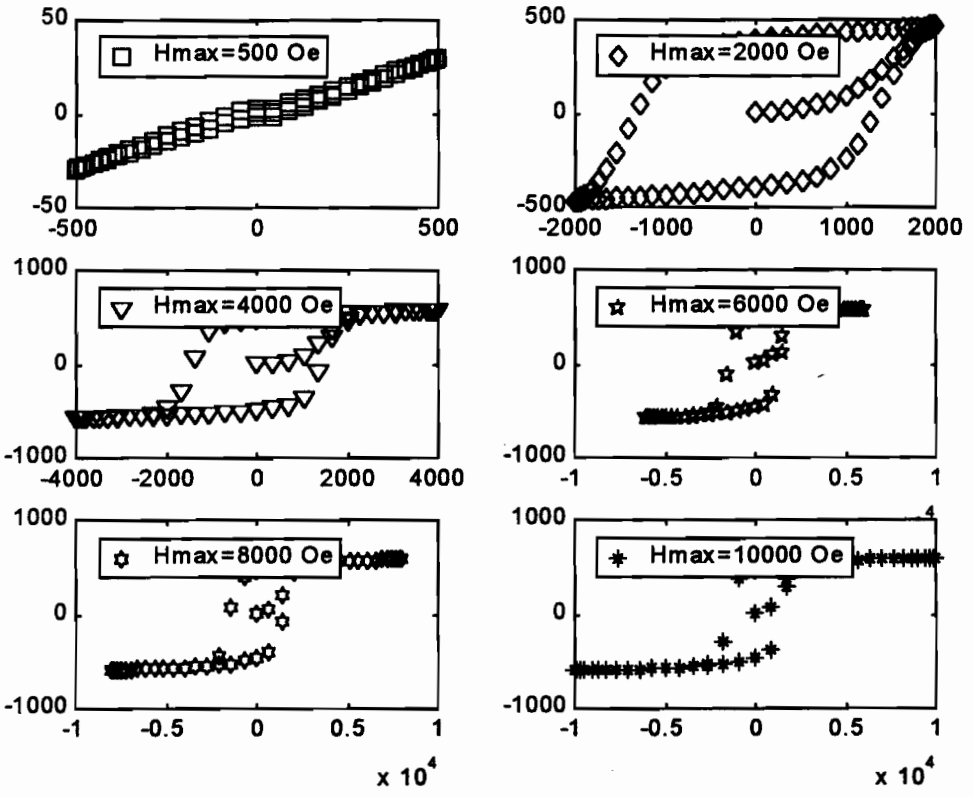


Figure 4.2 Generated M-H Loops from Preisach Model

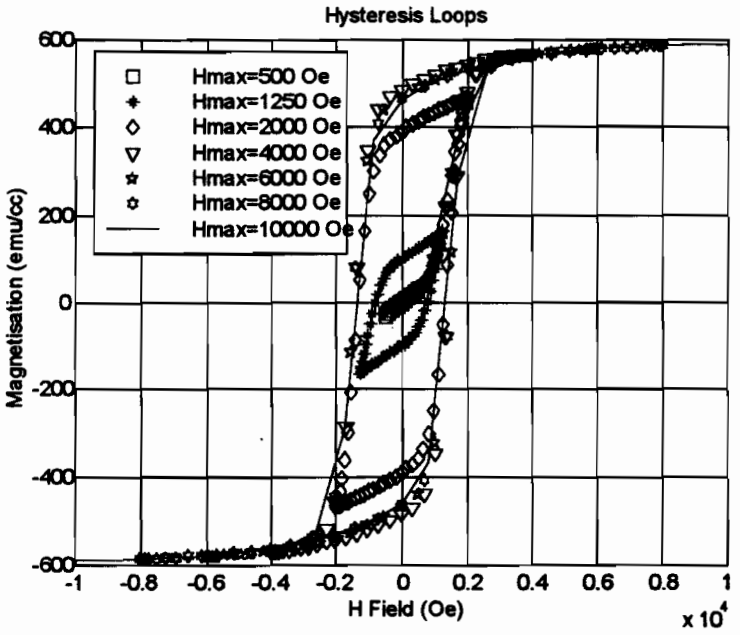


Figure 4.3 Nested Hysteresis Loops

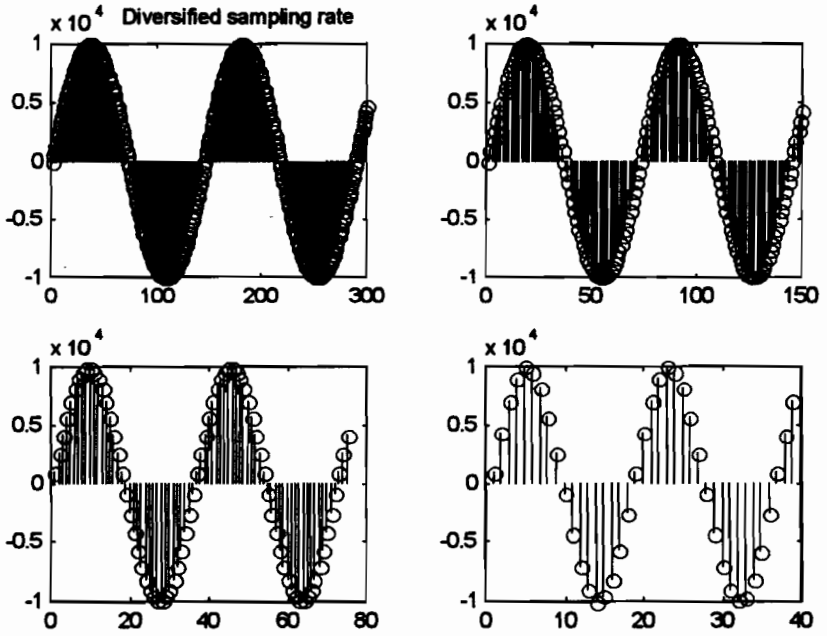


Figure 4.4 Example of Sampled Values of the H used to interpolate M

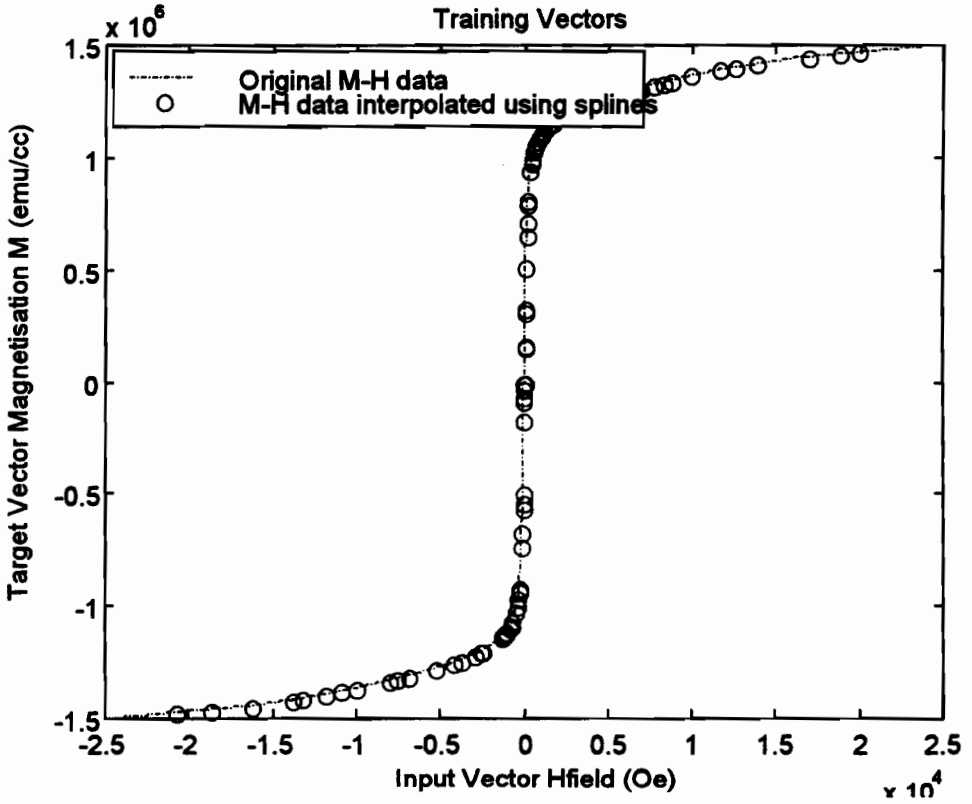


Figure 4.5 Training Data for Anhysteretic Material

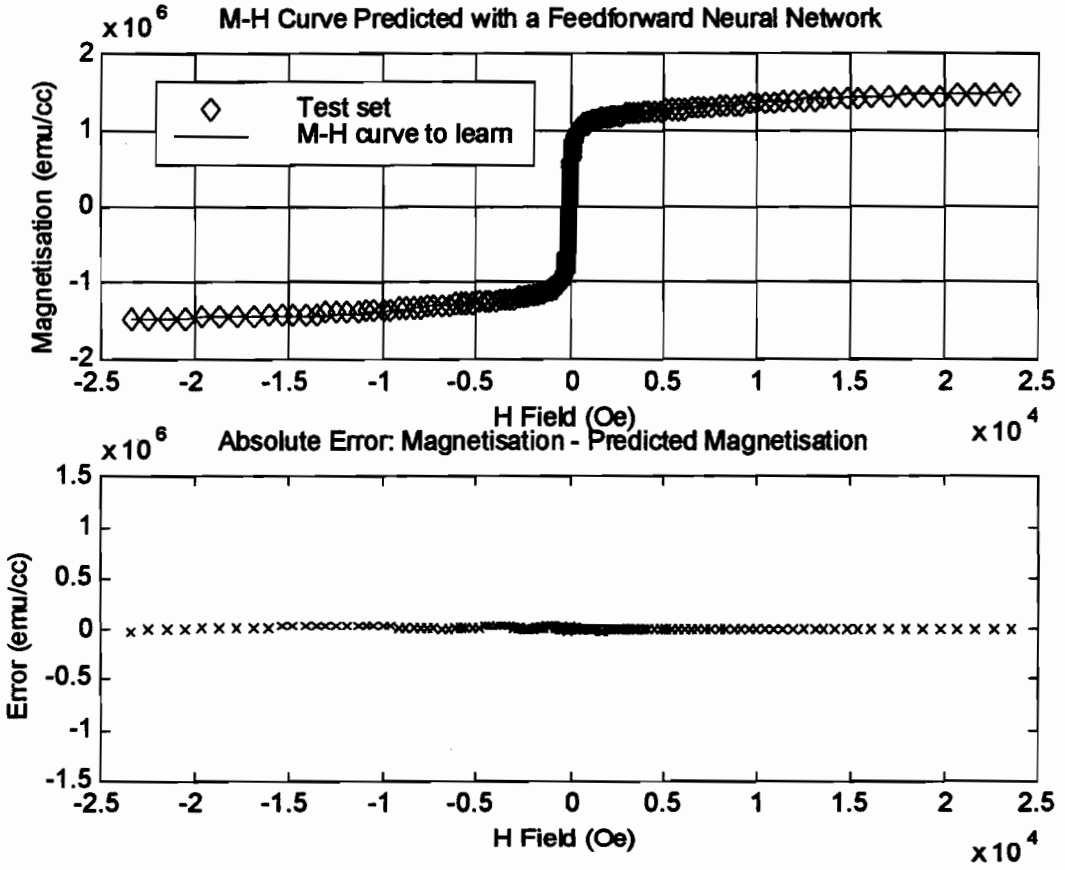


Figure 4.6 Curves Predicted with a (5-6-1) Feedforward Neural Network with Hyperbolic Tangent Activation Functions for the Hidden Layer and a Linear one for the Output, Trained with Levenberg-Marquardt method. Exhibiting an infinitesimal margin of error

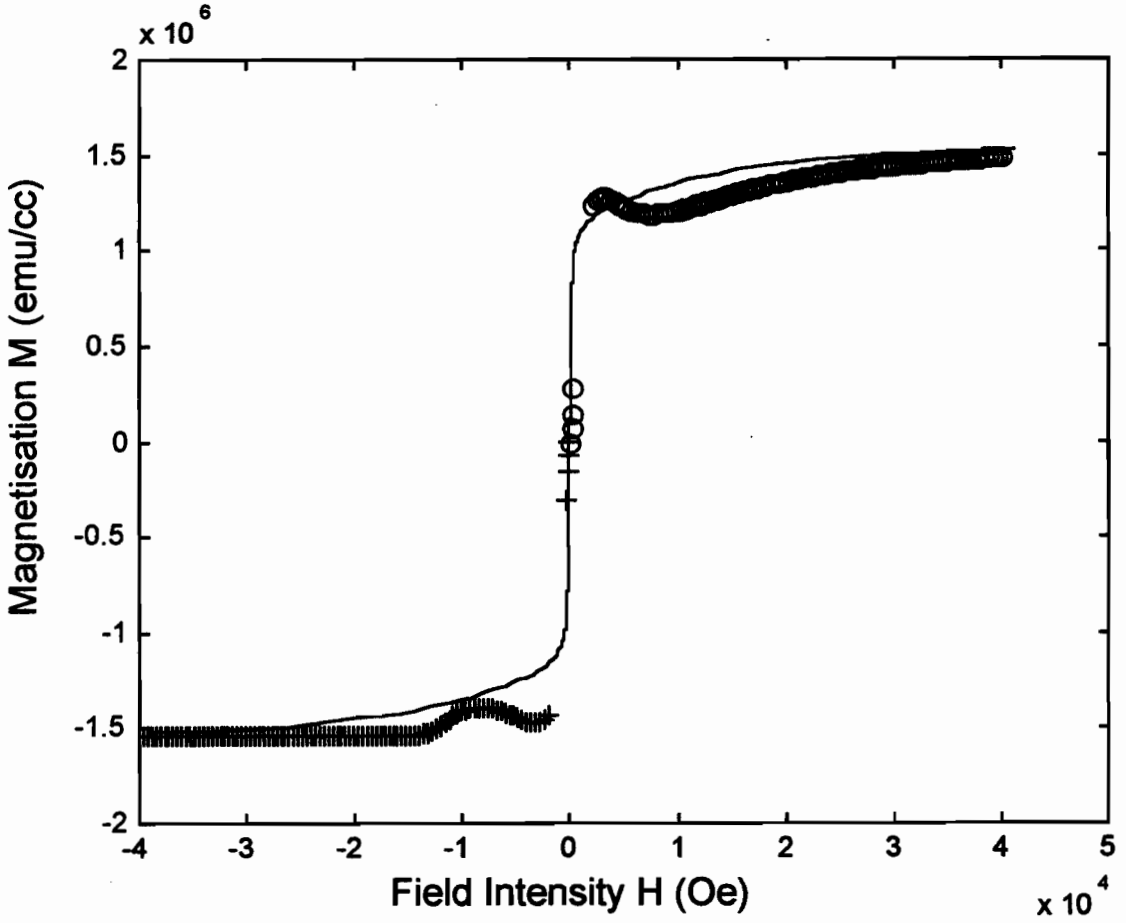


Figure 4.7 Badly represented Anhysteretic M-H Curve with a (5-3-1) Feedforward Neural Network with Hyperbolic Tangent Activation Functions for the Hidden Layer and Linear one for the Output, Trained with Levenberg-Marquardt method and needing more Neurons in the Hidden Layer

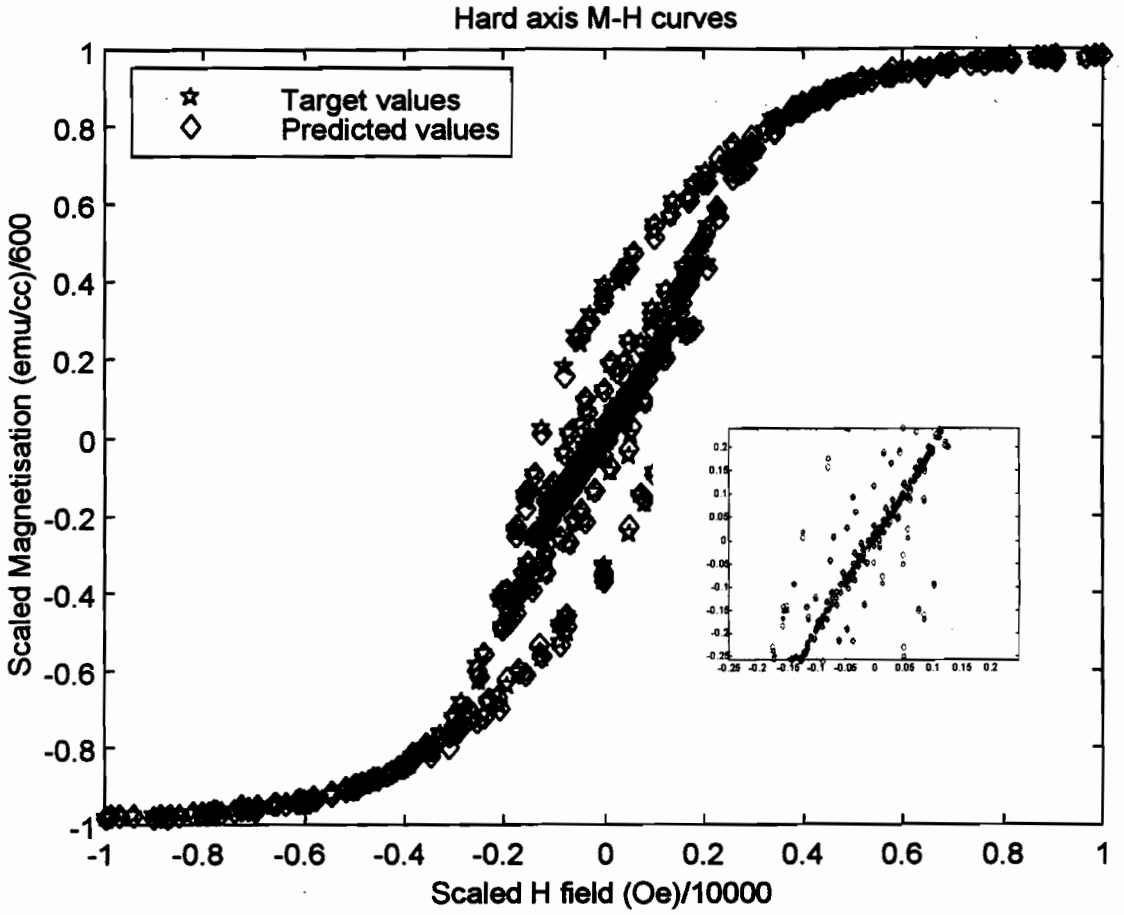


Figure 4.8 M-H Loops Hard Axis Predicted with a (5-12-1) Feedforward Neural Network with Hyperbolic Tangent Activation Functions for the Hidden Layer and Linear one for the Output, Trained with Levenberg-Marquardt method

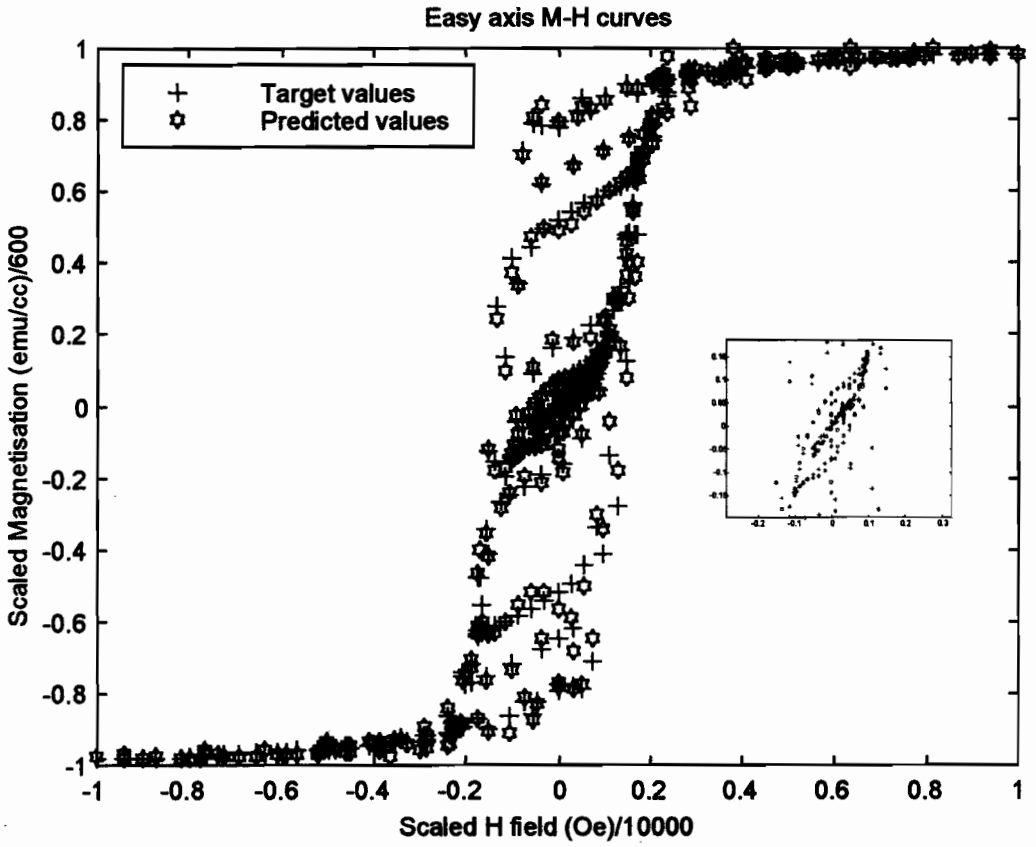


Figure 4.9 M-H Loops Easy Axis Predicted with a [5-12-1]-Feedforward Neural Network with Hyperbolic Tangent Activation Functions for the Hidden Layer and Linear one for the Output, Trained with Levenberg-Marquardt method

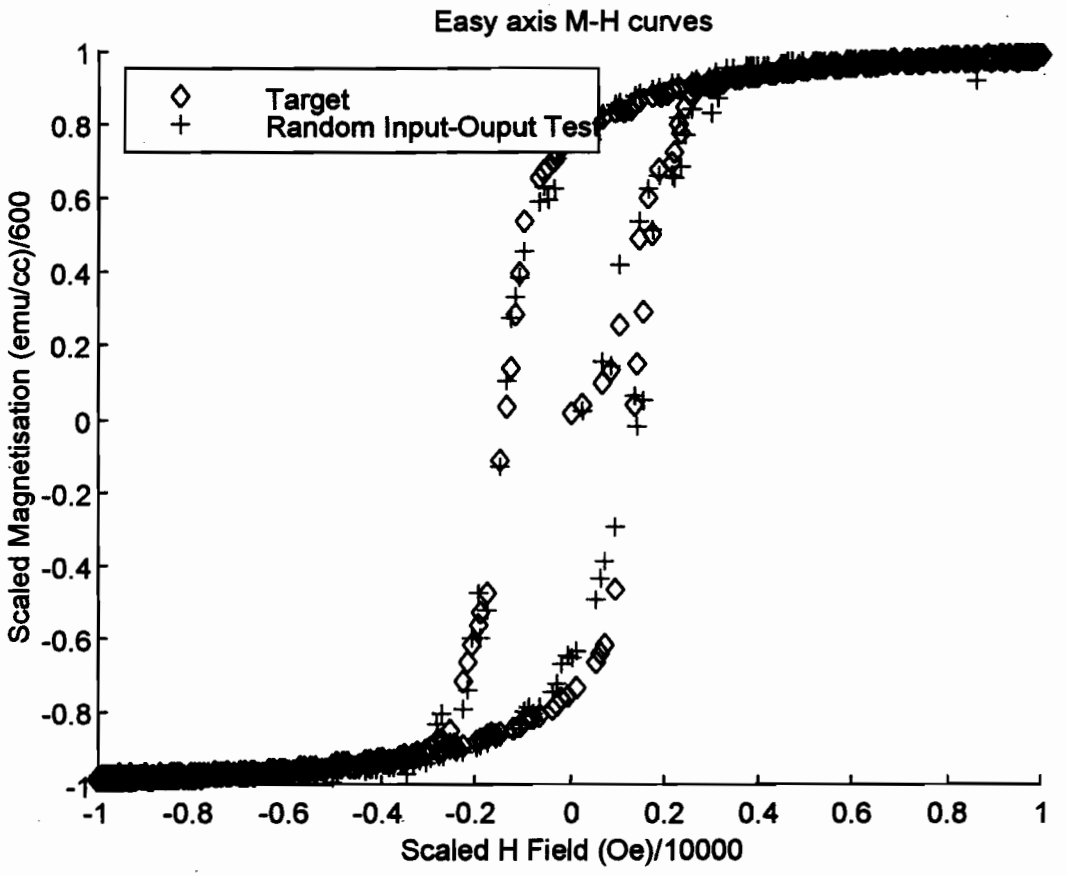


Figure 4.10 The Use of a BFGS Algorithm to Train an Initial Magnetisation Curve and a Limiting Hysteresis Loop [5-12-1], Hyperbolic Tangent Activation Functions for both the Hidden and the Output Layer

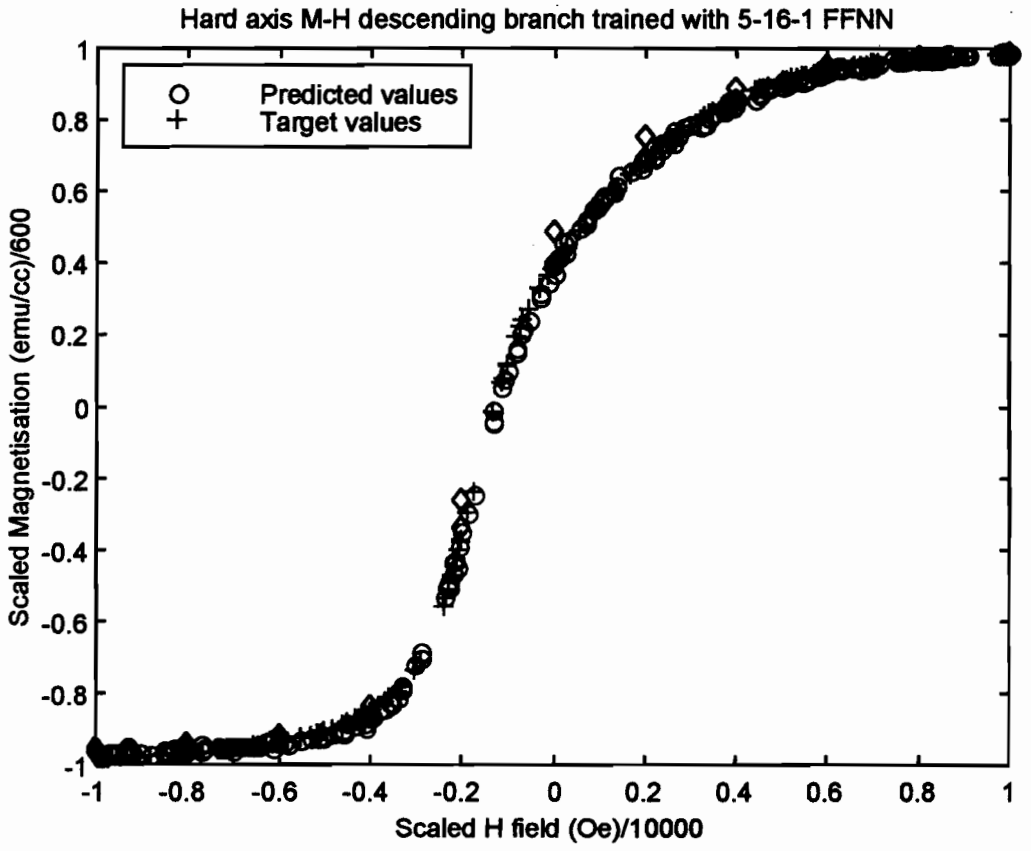


Figure 4.11 Simulating Newton-Raphson Arbitrary Excursions on the Descending Limiting Hysteresis Loop (◊ represents the random check points)

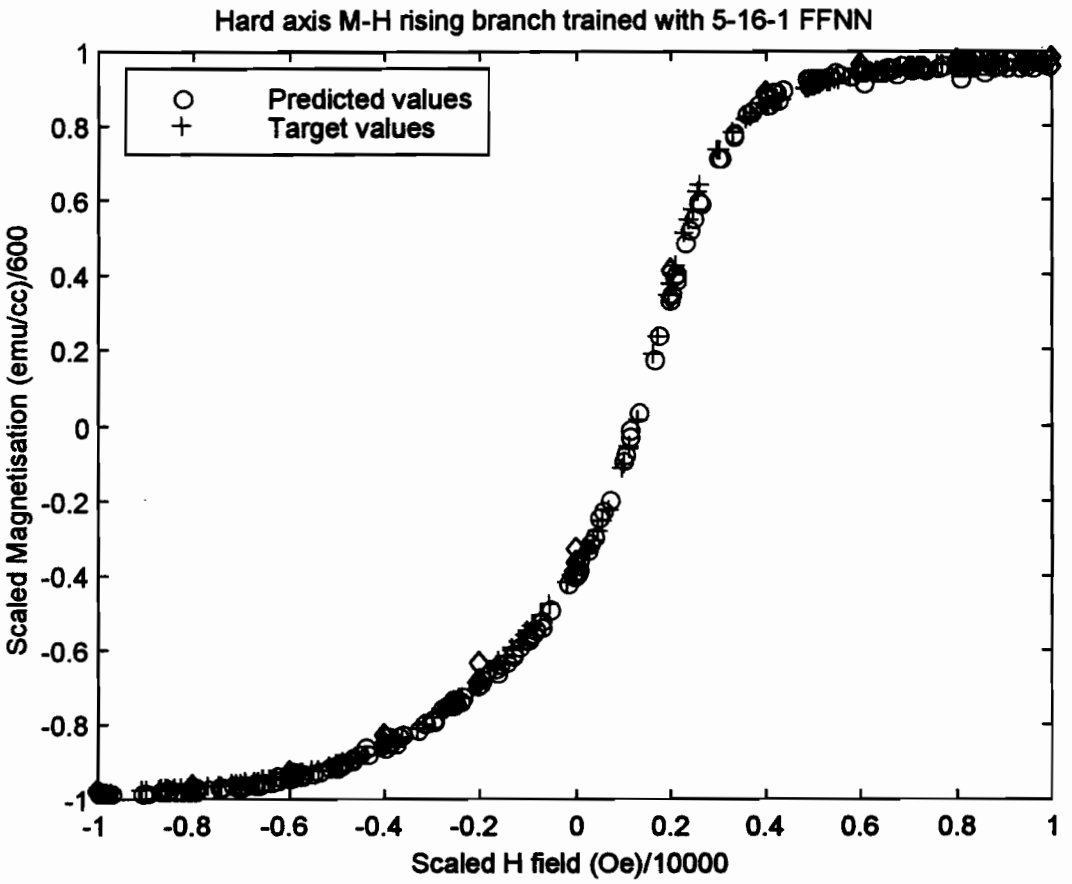


Figure 4.12 Simulating Newton-Raphson Arbitrary Excursions on the Rising Limiting Hysteresis Loop (◊ represents the random check points)

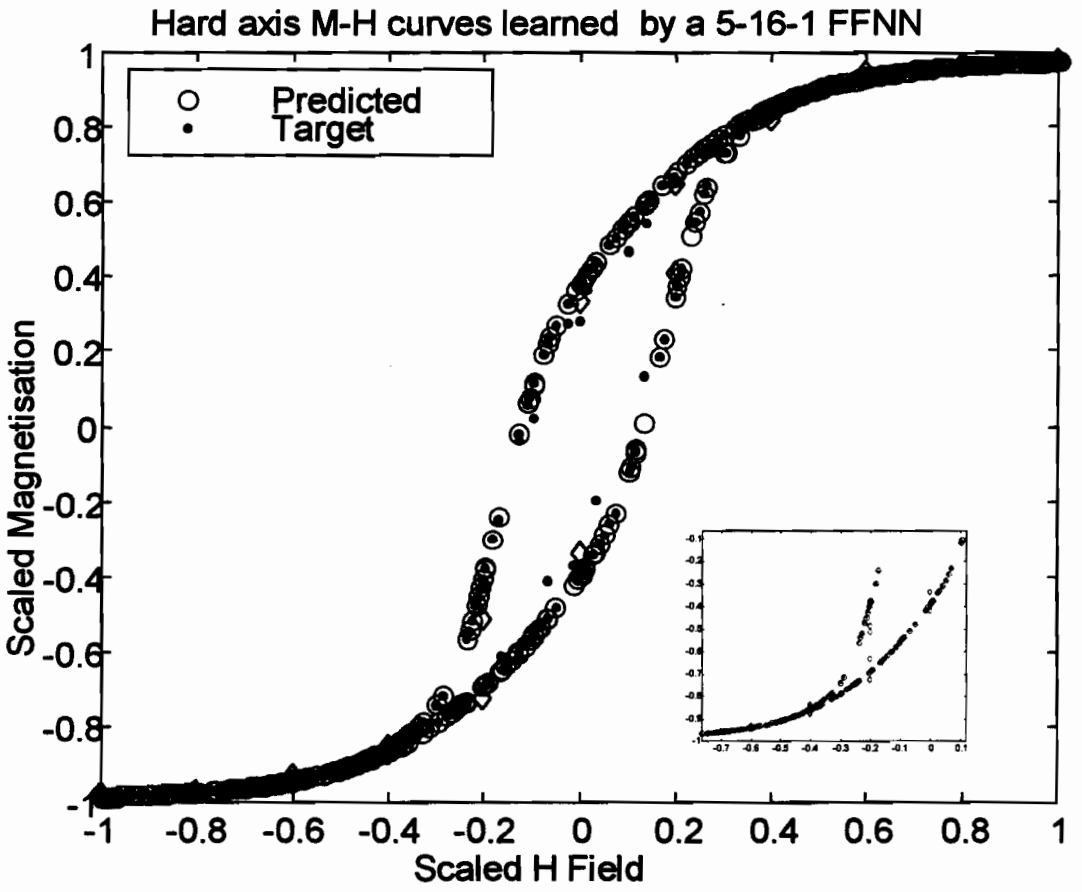


Figure 4.13 Simulating Newton-Raphson Arbitrary Excursions on both the Rising and Descending Limiting Hysteresis Loop (◊ represents the random check points)

4.3 Neural Network Based Models for Finite Element Solvers

4.3.1 Introduction

It has been established in the previous section that magnetic material behaviours can be modelled using artificial neural networks, but the real test of the system is its use within a finite element code. Several authors have considered the use of an Artificial Neural Network (ANN) as a method of providing a functional representation of the hysteresis curve while minimising the storage and time requirements [Nafalski et al, 1996; Mandayam, 1994; Xu-Refsum, 1993]. However, while these papers show that it is possible to model the hysteresis curves with a high

legree of accuracy, they do not attempt to use these models to solve realistic magnetic field roblems by embedding them within a finite element analysis. Alternate models, based on the reisach structure, have been embedded in finite element analysis systems and have shown good esults [Miano, 1995; Bottauscio, 1995]. The aim of this work is to demonstrate that neural etwork material models can be used in a finite element based analysis with comparable results, ut with considerable savings in terms of computer memory and execution time.

Here, we will describe in more detail, how these models are effectively implemented within a finite element program. We will also discuss an experimental implementation of the universal material model within a finite element program capable of solving magnetostatic problems by accounting for anisotropy with and without hysteresis. The results obtained by test cases are presented as well. Both a typical electrical steel (M19) trained as if it were a "*generic magnetic material*" (i.e. isotropic or anisotropic with or without hysteresis) and an hysteretic Preisach-based material model [Davidson, 1996] are used to test our solution.

4.4 Material Model Requirements for Finite Element Solvers

In order to build an interface for neural network material models that takes into account isotropic, anisotropic and anhysteretic or hysteretic behavior, one must create a set of routines. Most commercial analysis programs such as MagNet [MagNet 5.2, 1996] allow the user to provide their own material models through these typical set of routines. In the 3D permanent magnet magnetostatic problem solver used as a test bed in this thesis, eight routines are required to complete this task. The various steps involved in this process are outlined in detail in Appendix 4 and a sample code for H - M retrievals is shown in Appendix 5. The code should work with one component of H and M at a time. The implementation supports two previous states with the X, Y and Z components decoupled.

4.4.1 A Finite Element Implementation

Consider a problem in which the goal is to track the magnetisation of a device as the current is increased from zero and then reduced to zero. If the material is initially unmagnetised, it will move up its initial magnetisation curve until the current peaks, and then will track down the appropriate hysteresis loop. At each current value, the system solves a non-linear magnetic problem using a Newton-Raphson process. Keeping track of the magnetic field at any point so as to identify whether it is on the rising or falling direction is a major concern. During the iterations of this method, the starting state of the material magnetisation must be kept constant.

To generate a complete curve, there is a second iterative process in which the current is stepped up and the magnetisation is dependent on the previous state. In actual fact, once the magnetisation curves have been defined, the process is identical to that used for a standard magnetostatic solution. Therefore, provided that the time variation of the sources is not fast, the evolution of the magnetisation of an hysteretic material is similar to a succession of quasi-static states. At the point at which the field begins to decrease, the data regarding the state of the material are stored and form the basis for the next part of the hysteresis curve to be used.

The calculation of material properties takes place during the matrix assembly. The material model receives the values of the magnetic field, H, at each step and immediately returns the permeability values. In order to help the convergence of the Newton-Raphson technique, these routines also compute the slope of the curve at the current operating point. To do so, the following linear material relation is considered in each element:

$$M = Q \cdot H + M_0 \qquad (4.1)$$

where M is the vector magnetisation and H is the magnetic field vector, Q is a 3x3 tensor that represents the permeability and M_0 is the initial magnetisation vector expressing the state of a permanent magnetic material for example. During the non-linear steps inside the finite element solver, the following events take place:

(i) The material modelling routines provide the initial values of Q and M_0 in each element.

(ii) The finite element solver solves the linear problem.

(iii) Based on the solution H, the material modelling routines provide new values for Q and M_0.

(iv) Steps(ii) and(iii) are repeated until the field H stabilises.

If (H_1, M_1) are the latest values of the magnetic field and magnetisation, then the following calculations in the material modelling routines provide the data that the FEA solver and the Newton-Raphson method require:

$$Q_{ij} = \frac{dM_i}{dH_j} \tag{4.2}$$

$$M_0 = M_1 - Q.H_1 \tag{4.3}$$

4.4.2 A C-Core Test Problem For Speed and Memory Requirement

The neural network was trained with data representing the M and H values for a typical electrical steel (M19) and an interface, as illustrated in Appendix 4 (Figures A4.1 to A4.3), was developed to suit the finite element solver, so as to allow this model to be used instead of the built-in magnetisation curve model based on a set of hermite polynomials.

The topology used is a standard feedforward artificial neural network with five inputs and one output. The inputs to the system are the current value of the magnetic field and the previous values of the magnetic field and magnetisation, while the output is the new value of magnetisation. The system is trained using error backpropagation [Rumelhart et al, 1986;

Aleksander, 1995], by presenting it with the M and H field described in section 4.2 and illustrated in Figures 4.1 to 4.5. For a conventional network based on simple neurons implementing a weighted sum of its inputs, a hidden layer is required in order to model the non-linear behaviour. In the implementation within a finite element based analysis system, each element has access to the defining parameters of the neural network, namely the weights of the interconnections. In addition, each element has to store the values of M and H for the last two time steps. This is the minimum amount of data needed and thus the approach should address the minimum memory requirement stated at the outset of this work. The time to evaluate the required value of M for a given H is minimised, since the output of the neural network is just a weighted sum of the inputs.

The two M19 models , the neural network and the conventional polynomial, were then compared on the same problem, as shown in Figure 4.14, in terms of the accuracy of the results, the amount of memory used for each model and the operation count in the code. The basic process in each case is that a value of H is provided to the curve model and a new value of M is returned. The problem was non-linear and was solved using a Newton-Raphson process. The intention was to look at testing the generalisation of the neural network over a range of H values.

The polynomial model for the material has the advantage of being capable of returning the value of M, the permeability, μ and $\partial\mu/\partial H$. At present, the neural network returns only M; the calculation of μ and $\partial\mu/\partial H$ is done externally. Clearly, it would be appropriate to train the network to generate μ as well as M. Since these calculations are performed using a finite difference approximation, there is likely to be some error introduced in the calculations. Figure

·.14 shows the error field, i.e. the difference between B values computed, using the two different :urve models on the same finite element discretization. The errors have been quantized into about } levels of grey and show an average error in the core of less than 1.4%. For the solution using a Newton convergence tolerance of 0.01%, flux densities in the core ranged from 0.9 Tesla (below :aturation) to around 2 Tesla. Differences in the solutions are generated because of the lifferences exhibited in the two material curves shown in Figure 4.15. However, the inductance :omputed from the two solutions, a measure of the total energy in the system, concurred to within 0.3%. It should be also remembered that the polynomial curve used is only an approximation of the measured data and thus the error in the core flux densities between predicted and measured data could well be of the same order as the errors between the two models.

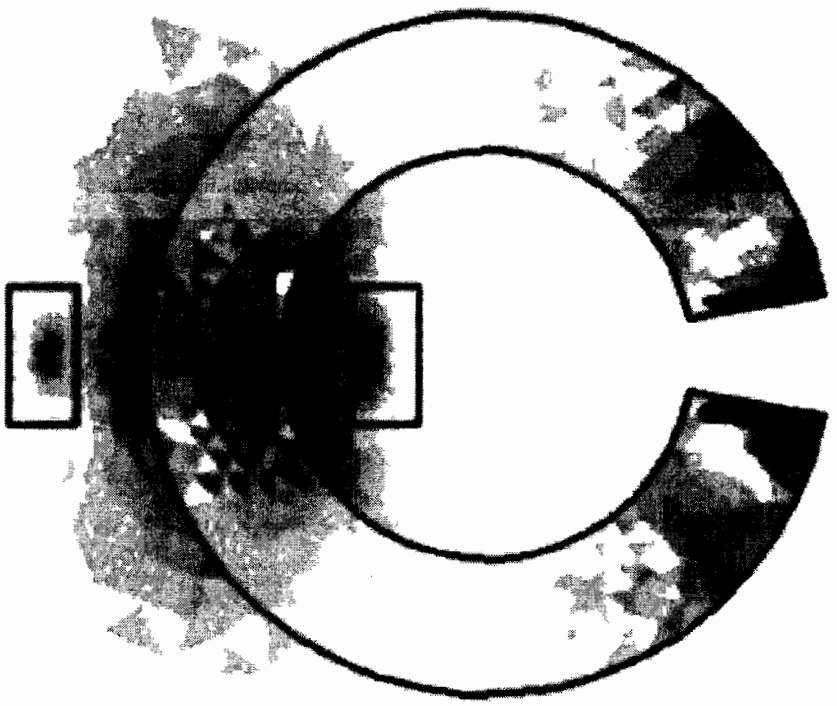


Figure 4.14 Error Plot between Solutions for M19 using Hermite Cubics and the Neural Network

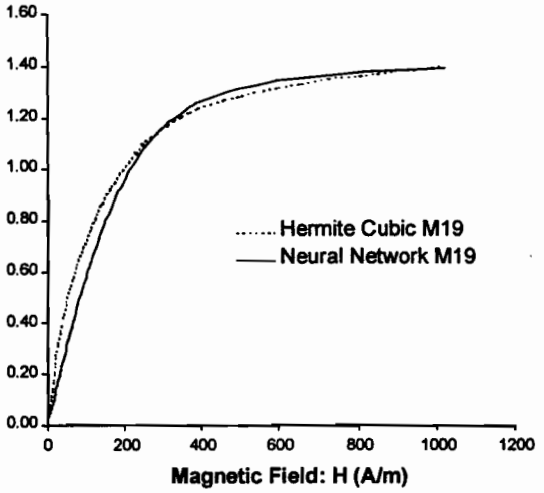


Figure 4.15 M-H Curve for M19 around the Initial Magnetisation Region

In terms of the memory required for each model, the polynomial model currently requires 83 floating point numbers to store the data, whereas the neural network requires 19 for the isotropic case - a saving of almost a factor of 4. The operation counts for both methods are given in table 4.1 for each model.

It is true that the operation count suggests that the neural network model is likely to cost about 20% more than is the case for the polynomial equivalent on each access in terms of CPU time. However, this is only a small component of the total overall solution process and the neural network system was measured as being 10% slower than the polynomial approach. In terms of Newton convergence, the polynomial model required 25 steps, whereas the neural network took 20. The difference here is probably due to the computation of μ as described earlier. The average

ıumber of conjugate gradient steps for each Newton step differed by 5 in 100 (the network being arger).

Table 4.1 Comparison of Operation Counts

	Polynomial	Neural Network
Additions	12	14
Multiplications	18	17
Divisions	3	6
Exponentials	0	2
Comparisons	4	0

Other sample experiments with various computational tolerance have shown that both Newton and successive substitution solution techniques can be successfully used for the two M19 models. When Using Newton's approach, as shown in Table 4.2 for a computational tolerance of 0.01 %, the neural M19 model appears to be faster than the Hermite polynomial model, with convergence steps of 20 and 25 steps respectively, as we have said earlier. However, when the tolerance increased, the number of convergence steps is almost the same. Results in Table 4.2 show as well that the computed energies are very close and the overall largest errors for the flux density are within the acceptable limit of experimental measurement errors.

Overall, it appears that the neural network model performs as well as more conventional models and is somewhat more efficient in terms of memory requirements, although the amount of memory required for isotropic, non-hysteretic materials is fairly small, in both cases.

Table 4.2: A C-Core Test Problem for M19 Models using a Finite Element Analysis Solver. Testing for Convergence and Solution Techniques

Newton Solution Method						
Tolerance	0.01 %			0.01 %		
	Energy (Joule)	Co-energy (Joule)	Largest B Error	Energy (Joule)	Co-energy (Joule)	Largest B Error
Hermite M19	683	690	5.6 %	688	691	3.4 %
Neural M19	645	690		688	691	
Successive Substitution Method						
Tolerance	1%			0.01 %		
	Energy (Joule)	Co-energy (Joule)	Largest B Error	Energy (Joule)	Co-energy (Joule)	Largest B Error
Hermite M19	689	691	0.1 %	686	690	1.1 %
Neural M19	688	691		678	691	

Similarly, Figures 4.16 to 4.18 show results obtained from tests carried out on a C-core problem to make sure that anisotropic material properties can be handled as well. Both solutions for a neural network approach and the hermite polynomials M19 model are used in this case. We have

treated the C-core as an anisotropic material by assuming the permeability of M19 in the y-direction and the permeability of air in the x-direction. The results obtained are quite similar. We can see as expected that the H-field is stronger along the horizontal path of the flux lines and weak along the vertical paths.

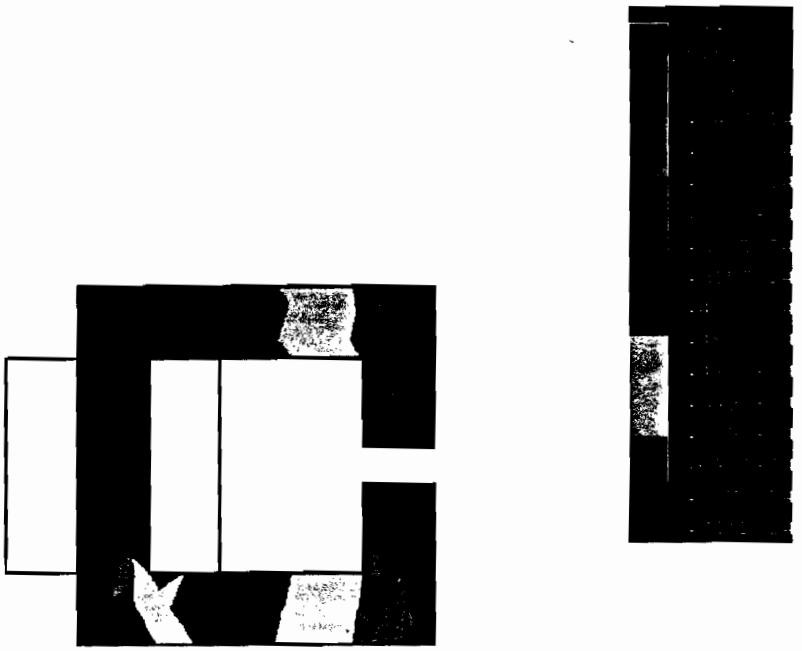


Figure 4.16 A C-Core H-Field Plots Results from MagNet [MagNet 5.2, 1996] with the Neural Network Based Material Model Interface.

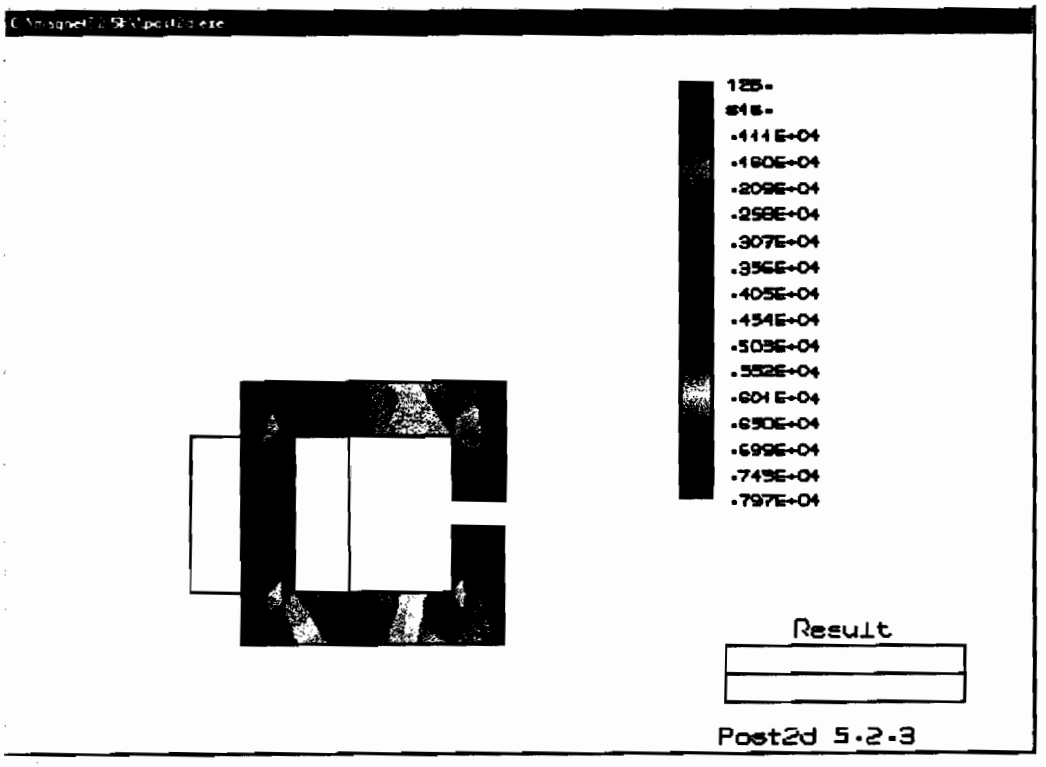


Figure 4.17 A Rectangular C-Core Problem H-Fields Plots. The Problem is Solved using an Hermite Polynomial Anisotropic Material Model

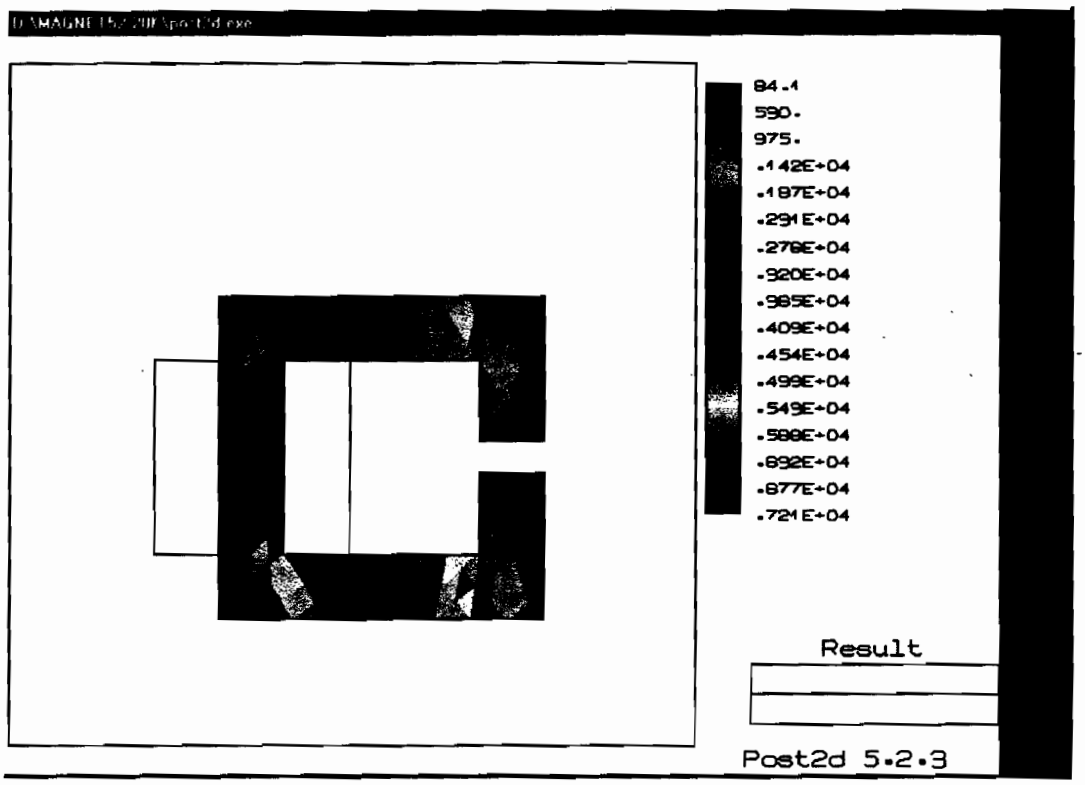


Figure 4.18 A Rectangular C-Core Problem H-Fields Plots; The same Problem (Figure 4.17) but Solved using an [5-6-1] Artificial Neural Network Anisotropic Material Model

4.4.3 The Advantage of Neural Networks

In the approach outlined in this thesis, each magnetic material to be used in a particular problem would have its own weight set, which of course characterises its response, but would use a network architecture common to all magnetic materials. The network for a non-hysteretic material would be the same as that for a material exhibiting hysteresis, thus making the representation uniform and universal.

In addition, in a magnetic device, a material is likely to be operating not only under a distributed set of H values, but also within a given range of temperatures. With models currently in use, a whole range of curves must be constructed to represent temperature variations as shown in Figure 4.19. The solution system then interpolates between them to determine an appropriate value for

the permeability and/or M . The neural network architecture described above can be modified to include an extra input that represents the temperature of the material. It is then trained with information that includes both the magnetic field and temperature level. The final network will be able to generate the appropriate response without the need for explicit interpolation. This is likely to both simplify the system, reduce the amount of memory required to represent a material and reduce the problem-solving time.

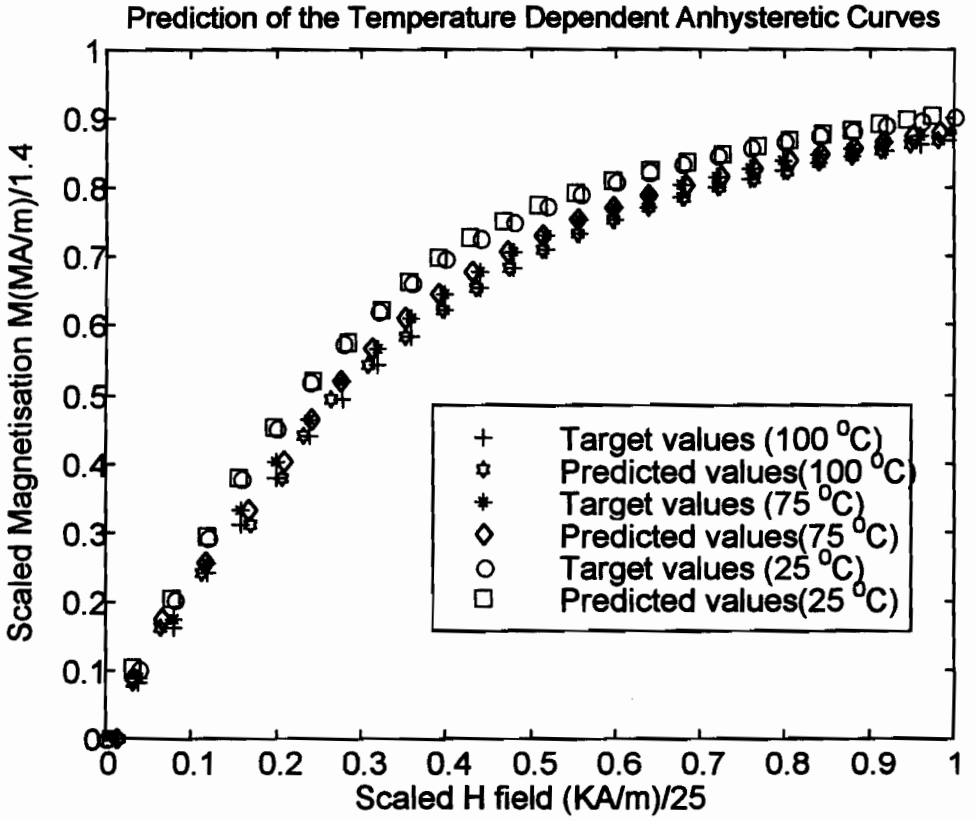


Figure 4.19 Neural Network Prediction of the Parametrized vs Temperature of the Modified Jiles-Atherton Model described in section 2.3

Chapter 5

5 Future Work and Conclusions

5.1 Future Work

Future work on solving electromagnetic problems using the proposed model should follow these lines:

Improvements need to be made toward a more efficient automated knowledge acquisition process, particularly the investigation of a way of implementing other schemes of pre-processing algorithms, or using artificial neural networks to classify and cluster the generated data beforehand to gather the best sequences of training data and their number. Such a technique would reduce the required training time and remove the burden on a user not familiar with artificial neural network methods.

In a magnetic device, a material is likely to be operating not only under a distributed set of H values at various frequencies but also at various temperatures, not to mention stress effects. It is clear that artificial neural networks are more appropriate since they can, in this sense, allow a straightforward inclusion of diverse factors and considerations.

Material representations can be stored in neural database systems [Bower-Beeman, 1994] so that they can be retrieved when required during a magnetic device optimisation or simulation task. A number of problems have to be solved and analysed within an optimisation or optimization process many parameters are to be tried. Thus, for example, getting a neural based material model to quickly replace another could be useful.

Today, both in the market place and in the research field, the provider, be it the manufacturer or the vendor, must furnish the functional characteristics of materials. Thus, research in this area is a key concern for all parties. An approach combining the use of functional material characteristics instead of simple structural behaviour, together with soft computation techniques such as neural, genetic and fuzzy systems, could be viable in many areas. In other words, what is needed are "intelligent", multi-purpose materials representations.

As we all know, the finite element method provides a technique involving the transformation of field equations into energy functionals to be minimised using some optimisation algorithms. An approach to be considered in the future could be to use the energy functional information as a priori knowledge to train the selected artificial neural network. The appropriate knowledge source or theoretical base can be, rather than a Preisach-based phenomenological model, grounded on energy density based $B-H$ curve modelling, as suggested by Silvester and Gupta [Silvester-Gupta, 1991] who show that a full description is available if the stored energy density is known for any given magnetic state of the material. These new considerations should help bring about another approach in solving electromagnetic problems using finite element or boundary element methods. The $B-H-M$ behaviour will then not be the sole and ultimate modelling goal.

5.2 Conclusions

The research reported here is an example of using a neural-based approach to model and utilise magnetic materials. The neural network serves as the memory component for

complex non-linear dependencies, while the less computationally intensive tasks are accomplished with conventional analysis techniques. The approach provides a uniform method of handling all materials, rather than using different representations for different properties. The performance of the system is heavily dependent on the architecture of the network chosen as well as on the input-output data. This seems to be particularly promising for a wide range of practical applications.

The capabilities of various artificial neural network capabilities for modelling non-linear systems with unknown dynamics and their identification have been presented and verified through numerical examples. The experiments carried throughout the frame of this thesis clearly show that supervised multilayer non recurrent feed-forward neural networks best suit both the memory and speed requirements to run large finite elements problems. Recurrent networks have to perform more difficult task than that of a feed-forward network. In fact, a recurrent network not only has to make the prediction, it has to figure out what information it needs to keep about past data points in order to make a prediction. Thus it needs more neurons in its hidden layers so that there is enough variation in weights and biases to form a precise enough representation of past inputs for the problem. In our case, the past data information, the two previous states, are known beforehand. Finally, the major problems concerning the recurrent networks is that related to their stability.

The utility of CMAC and RBF networks in modelling have been shown in our previous work. However, the experiments with CMAC networks show the need for finding techniques in adaptively selecting the quantization level in order to cope with the function to be learned so as to reduce the required memory space and the speed of the learning

process. In the case of the standard RBF networks another drawback is their inability to deal with duplicated input data that cause singularity problems. This drawback is not avoidable without using a number of preprocessing strategies such as clustering the input data. Standard RBF networks suffer, as well, from the curse of dimensionality; i.e. when the number of inputs increases, the number of neurons in the hidden layer of a RBF network grows exponentially.

Discussions and experimental work in this thesis suggest as well, that to achieve a generalised magnetic material model, multiple established material models identifying clearly the influence of some factors such as the temperature, the frequency and stress should be explicitly used. Modular artificial neural networks architectures can be thought of for this purpose.

In this thesis, we have shown a possible approach for integrating neural network based magnetic materials models, both isotropic or anisotropic and with or without hysteresis within a large-scale finite element analysis system in an efficient manner - the computation of magnetisation values being reduced to a simple weighted sum calculation. The results obtained have demonstrated that the neural network model can provide a computational model that has a cost in terms of time comparable to that of more conventional polynomial-based systems, but that does have a reduced requirement in terms of memory. However, while the basic process has been proven, there is still more work to be done in producing a complete system handling major and minor loops.

Preprocessing the input-output information is not a simple task. To take into account the broad range of data needed to train the neural network, special care has been taken to ensure that both the material physical model that can be acquired using fixed sequences

of states and the non-linear solver, the Newton-Raphson method which uses random search techniques, are satisfied. For this purpose, we have used sequential spline interpolation technique and filtering methods to respectively grow the number of training data points and appropriately distribute them as described in section 4.2.4.

Provided that a great number of experimentally validated data points are available for a large number of materials and that measurement techniques can isolate differing factors such as temperatures, frequencies and stresses, the use of techniques based on neural networks for modelling magnetic materials can be automated with a higher degree of accuracy.

References

[Adly-Abd-El-Hafiz, 1998] Adly, A. A. and Abd-El-Hafiz, S. K. , "Using Neural Networks in the Identification of Preisach-Type Hysteresis Models", *IEEE Transactions on Magnetics*, Vol. 34, No. 3, pp. 629-635, 1998.

[Albus, 1975a] Albus, J.S., "A New Approach to Manipulator Control: The Cerebellar Model Articulation Controller (CMAC)", *Transactions ASME J. DSMC,* Vol.97 No.3, pp 312-324, 1975.

[Albus, 1975b] Albus, J.S., "Data Storage in the Cerebellar Model Articulation Controller", *Transactions ASME J. DSMC,* Vol. 97 No.3, pp 228-233, 1975.

[Aleksander-Morton, 1995] Aleksander, I. and Morton, H., *Neural Computing*, Second Edition, International Thomson Computer Press, London, 1995.

[Armstrong et al. 1991] Armstrong W.W., Dwelly, A., Liang, J.D., Lin, D. and Reynolds, S., " Learning and Generalization in Adaptive Logic Networks, in Artificial Neural Networks ", *Proceedings of the 1991 International Conference on Artificial Neural Networks* (*ICANN'91*), Espoo, Finland, June 24-28, 1991.

[Atree 3.0, 1996] Atree 3.0 Educational Kit, Dendronic Decisions Limited , Edmonton, Canada, 1996.

[Benbouzid, 1998] Benbouzid, M. E. H., "Artificial Neural Networks for Finite Element Modeling of Giant Magnetostrictive Devices", *IEEE Transactions on Magnetics*, Vol. 34, No. 6, pp. 3853-3856, 1998.

[Berger, 1994] Berger, C. S., "Linear splines with adaptive mesh sizes for modeling non-linear dynamic systems", *IEE Proceedings: Control Theory & Applications*. Vol. 141, p.p. 277-84, 1994.

[Bertotti, 1998] Bertotti, G., *Hysteresis in Magnetism*, Academic Press, Boston 1998.

[Bottauscio et al. 1995] Bottauscio, O., Chiampi, M. and Repetto, M., "Modelling Dynamical Hysteresis in Ferromagnetic Sheets under Time-Periodic Supply Conditions", *Proceedings of the 7th International IGTE Symposium*, Graz, Austria, pp. 313-318, 1995.

[Bouc, 1969] Bouc R.,"Modèle mathématique d'hystérésis. Application aux systèmes à un degré de liberté", *Thèse*, Marseille 1969, C.N.R.S (AO 3078).

[Bouc, 1971] Bouc R.,"Modèle mathématique d'hystérésis", *Acustica*, Vol. 24, p.16-25, 1971.

[Bower-Beeman, 1994] *The Book of GENESIS*: Exploring Realistic Neural Models with the GEneral NEural SImulation System, Springer-Verlag, New York, 1994

[Brown et al. 1993] Brown, M., Harris, C.J. and Parks, P.C., "The interpolation capabilities of the binary CMAC", *Neural Networks*, Vol.. 6, No. 3, pp. 429-40, 1993.

[Brown-Harris, 1994] Brown M. and Harris C.J., *Neurofuzzy Adaptive Modeling and Control*, Prentice Hall, Hemel Hempstead, 1994.

[Buchanan et al. 1983] Buchanan, B.G., Barstow, D., Bechal, R., Bennett, J., Clancey, W., Kulikowski, C., Mitchell, T. and Waterman, D.A., "Constructing an expert system", in *Building Expert Systems*, F. Hayes-Roth, D.A. Waterman and D.B. Lenat (ed.), Addison-Wesley, Reading, Massachusetts, pp 127-68, 1983.

[Burgin, 1992] Burgin, G., "Using Cerebellar Arithmetic Computers", *AI Expert*, p.p. 32-41, 1992.

[Chiang-Lin, 1995] Chiang, Ching-Tsan and Lin, Chun-Shin, "Integration of CMAC and radial basis function techniques", *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, Vol. 4, 1995. IEEE, Piscataway, New Jersey, U.S., 95CB35767, pp. 3263-3268, 1995.

[Cincotti-Daneri, 1997] Cincotti, S. and Daneri, I., "Neural Network Identification of a non-linear circuit model of hysteresis", *Electronics Letters*, Vol. 33, No. 13, 1997.

[Cortial et al. 1997] Cortial, F., Ossart, F., Albertini, J.B. and Aid, M., "An Improved Analytical Hysteresis Model and its Implementation in Magnetic Recording Modeling by the Finite Element Method", *IEEE Transactions on Magnetics*, Vol. 33, pp. 1592-1595, 1997.

[Coyne et al. 1989] Coyne, R.D., Rosenman, M.A., Radford, A.D., Balachandran, M. and Gero, J.S., *Knowledge-Based Design Systems*, Addison-Wesley, 1989.

[Cybenko, 1989] Cybenko, G., "Approximation by Superpositions of a Sigmoidal Function", *Mathematics of Control, Signals, and Systems*, Vol. 2, pp. 303-314, 1989.

[Davidson, 1996] Davidson R., " Vector Preisach Hysteresis Models for Simulation of Recording Media ", *Ph.D. Thesis*, Carnegie Mellon University, 1996

[Della Torre-Vajda, 1996] Della Torre, E. and Vajda, F., "Vector Hysteresis Modeling for Anisotropic Recording Media", *IEEE Transactions on Magnetics*, MAG-35, pp.1116-1119, 1996.

[Durkin, 1991] Durkin J., "Designing an Induction Expert System", *AI EXPERT*, pp.29-35, 1991.

[Dym-Lewitt, 1991] Dym, C.L. and Lewitt R.E., *Knowledge Based Systems in Engineering*, McGraw-Hill, New York 1991.

[Edgar et al. 1991] Edgar A. P.C., Miller, W.T., and Parks, P.C., "Design Improvements in Associative Memories for Cerebellar Model Articulation Controllers (CMAC)", *Proceedings of ICANN*, pp 1207-1210, 1991.

[EL-Sherbiny ,1973] EL-Sherbiny, M.K., "Representation of the Magnetization Characteristic by a Sum of Exponentials", *IEEE Transactions on Magnetics*, Vol. 9, No. 1, pp. 60-61, 1973.

[Fletcher, 1987] *Practical Methods of Optimization*, John Wiley, New York, 1987.

[Forghani et al. 1982] Forghani, B., Freeman, E.M., Lowther, D.A. and Silvester, P.P., "Interactive modelling of magnetisation curves" , *IEEE Transactions on Magnetics*, Vol. 18, pp. 1070-1072, 1982.

[Funahashi, 1989] Funahashi, K., "On the Approximate Realization of Continuous Mappings by Neural Networks", *Neural Networks*, Vol. 2. pp. 183-192, 1989.

[Gill et al. 1981] Gill, P. E., Murray, W. and Wright, M. H., *Practical Optimization*, Academic Press, London, 1981.

[Hagan-Menhaj, 1994] Hagan, M. and Menhaj, M., " Training Feedforward Networks with the Marquardt Algorithm ", *IEEE Transactions on Neural Networks*, Vol. 5, No. 6, pp 989-993, 1994.

[Hannalla-McDonald, 1980] Hannalla, A. Y. and McDonald, D. C., "Representation of soft magnetic materials", *Proceedings of IEE*, Vol.127 Pt. A, No.6, pp. 386-391, 1980.

[Hassibi-Stork, 1993] Hassibi, B. and Stork, D. G., "Second order derivatives for networks pruning: optimal brain surgeon", *NISP 5*, S.J.Hanson et al (ed.), 164, San Mateo, Morgan Kaufman, 1993.

[Henrotte et al. 1992] Henrotte, F., Nicolet, A., Delince, F., Genon, A. and Legros, W.,"Modelling of ferromagnetic materials in 2D finite element problems using Preisach's model", *IEEE Transactions on Magnetics*, Vol.28, No. 5, pp.2614-2616, 1992.

[Hodgdon, 1988a] Hodgdon, M., "Application of a Theory of Ferromagnetic Hysteresis", *IEEE Transactions on Magnetics*, Vol. 24, No. 1, pp. 218-221, 1988.

[Hodgdon, 1988b] Hodgdon, M., "Mathematical Theory and Calculations of Magnetic Hysteresis Curves", *IEEE Transactions on Magnetics*, Vol. 24, No 6, pp. 3120-3122, 1988.

[Hopfield, 1982] Hopfield, J. J., "Neural networks and physical systems with emergent collective computational abilities", *Proceedings of the National Academy of Science,* vol. 79, pp.2554-2558, U.S.A., 1982.

[Hornick et al. 1989] Hornick, K, Stinchcombe, M., and H. White, "Multilayer Feedforward Networks and Universal Approximators", *Neural Networks*, Vol. 2, pp 359-366, 1989.

[Hui-Zhu, 1995] Hui, S.Y.R. and Zhu, J., "Numerical modelling and simulation of hysteresis effects in magnetic cores using transmission-line modelling and Preisach

theory", *IEE Proc-Electr. Power Appl.*, Vol.142, No. 1, p.57-62, 1995.

[Hwang-Lord, 1976] Hwang, J. H.and Lord W. J., "Exponential series for B-H curve modelling", *Proceedings of IEE*, Vol.123, No. 6, pp. 559-560, 1976.

[Jiles, 1992] Jiles, D.C., "A self Consistent Generalized Model for the Calculation of Minor Loop Excursions in the Theory of Hysteresis", *IEEE Transactions on Magnetics*, Vol. 28, No. 5, pp. 2602-2604, 1992.

[Jiles et al. 1992] Jiles, D.C., Thoelke, J.B. and Devine, M.K., "Numerical Determination of Hysteresis Parameters for Modeling of Magnetic Properties Using the Theory of Ferromagnetic Hysteresis" *IEEE Transactions on Magnetics*, Vol. 28, No, 1, pp. 27-35, 1992.

[Jiles-Atherton, 1984] Jiles, D.C. and Atherthon, D.L., "Theory of Ferromagnetic Hysteresis", *Journal of Applied Physics*, Vol. 55, No. 6, Pt. 2B, pp. 2115-2120 1984.

[Kodratoff-Bares, 1991] Kodratoff, Y.and Bares M., *Base terminologique de l'Intelligence Artificielle*, Technique et Documentation-Lavoisier, 1991.

[Kohonen, 1982] Kohonen, T., "Self-Organized Formation of Topologically Correct Feature Maps", *Biological Cybernetics*, Vol. 43, pp. 59-69, 1994.

[Kvasnica-Kundracik, 1996] Kvasnica, B., Kundracik, F. "Fitting Experimental Anhysteretic Curves of Ferromagnetic Materials and Investigation of the Effect of Temperature and Tensile Stress", *Journal of Magnetism and Magnetic Materials*, Vol. 162, pp43-49, 1996.

[Lin-Chiang, 1998] Chiang, Ching-Tsan and Lin, Chun-Shin, "Integration of CMAC and Weighed Regression for Efficient Learning and Output Differentiability", *IEEE Transactions on Systems, Man and Cybernetics,* Vol. 28, No. 2, pp. 231-237, 1998.

[Lowther-Silvester, 1986] Lowther, D.A. and Silvester P.P., *Computer Aided Design in Magnetics*, Springer-Verlag, New York, 1986.

[MagNet 5.2, 1996] MagNet 5.2, Reference Manual, Infolytica Corporation, Montreal, Canada, 1996.

[Mandayam et al. 1994] Mandayam, D., Udpa, L., and Udpa, S. S., "Parametric Models for Representing Hysteresis Curves", *Nondestuctive. Testing and Evaluation*, Vol. 11, pp. 235-245, 1994.

[Masters, 1993] Masters, T., *Practical Neural Network Recipes in C++*, Academic Press Inc., Boston, 1993.

[Masters, 1995] Masters, T., *Advance Algorithms for Neural Networks A C++ Sourcebook*, John Wiley & Sons Inc., New York, 1995.

[Mayergoyz, 1991] Mayergoyz, I.D., *Mathematical Models of Hysteresis*, Springer-Verlag, New York, 1991.

[McCarthy, 1949] McCarthy, J., "Programs with common sense", in Minsky, M. (ed), *Semantic Information Processing*, MIT Press, MIT, Cambridge, 1968.

[McCulloch-Pitts, 1943] McCulloch, Warren and Pitts, Walter, "A logical calculus of the ideas immanent in nervous activity," *Bulletin of Mathematical Biophysics*, No. 7, 1943.

[Miano et al. 1995] Miano, G., Serpico, C., Verlolino, L. and Visone, C., "Comparison of Different Hysteresis Models in FE Analysis of Magnetic Field Diffusion", *IEEE Transactions on Magnetics*, Vol. 31, pp.1789-1792, 1995.

[Michalski, 1993] Michalski, R.S., "LEARNING = INFERENCING + MEMORISING Basic Concepts of Inferential Theory of Learning and Their Use for classifying Learning Processes", in *FOUNDATIONS OF KNOWLEDGE ACQUISITION: Machine Learning*", Meyrowitz, A. L. and Chipman, S., (ed.), Kluwer Academic Publisher, London, 1993.

[Michalski-Kodratoff, 1990] Michalski, R.S. & Kodratoff, Y., "Research in Machine Learning: Recent Progress, Classification of Methods and Future Directions"; in *Machine Learning: An Artificial Intelligence Approach*, Vol. III, Kodratoff, Y. and Michalski, R.S. (eds), Morgan Kaufmann Publishers, Inc. 1990.

[Mohammed et al. 1993] Mohammed, O. A., Merchant, R. S. and Üler, F. G., "Utilizing Hopfield neural networks and an improved simulated annealing procedure for design optimization of electromagnetic devices", *IEEE Transactions on Magnetics*, Vol. 29, pp. 2404-2406, 1993.

[Mohammed et al. 1994] Mohammed, O., Merchant, R. and Uler, F., "An Intelligent System for Design Optimization of Electromagnetic Devices", *IEEE Transactions on Magnetics*, Vol. 30, pp. 3633-3636, 1994.

[Nafalski et al. 1996] Nafalski, A., Hoskins, B. G., Kundu, A.and Doan, T., "The Use of Neural Networks in Describing Magnetisation Phenomena", *Journal of Magnetism and Magnetic Materials*, Vol. 160, 84-86, 1996.

[Naidu, 1990] Naidu S.R.,"Simulation of the hysteresis phenomenon using Preisach's

theory", *IEE Proc.* Vol.137, Pt. A, No. 2, p.73-79, 1990.

[Nussbaum et al. 1996] Nussbaum, C., Booth, T., Ilo A. and Pfützner, H., "A Network for the Prediction of Performance Parameters of Transformer Cores", *Journal of Magnetism and Magnetic Materials*, Vol. 160, pp. 81-83, 1996.

[Ossart et al. 1995] Ossart, F., Davidson, R. and Charap, S. H., " A 3D Moving Vector Preisach Hysteresis Model ", *IEEE Transactions on Magnetics*, Vol. 31, pp. 1785-1788, 1995.

[Oti, 1990] Oti, J.O., "Vector Characterisations of Magnetic Recording Medium", *Ph.D. Thesis*, George Washington University, 1990.

[Pao, 1989] Pao, You-Han, *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, Massachusetts, 1989.

[Peng et al. 1990] Peng, T.M., Hubele, N. F. and Karady, G. G., "Conceptual Approach to the Application of Neural Networks for Short-term Load Forecasting", *Proceedings of the 1990 IEEE International Symposium on Circuits and Systems*, pp. 2942-2945, 1990.

[Potter, 1970] Potter, R. I., "Analysis of saturation magnetic recording based on arctangent magnetic transitions", *Journal of Applied Physics*, Vol.41, pp.1647-1651, 1970.

[Preisach, 1935] Preisach, F., "Uber die magnetische nachwirkung", *Zeitschrift fur Physik*, Vol. 94, pp.277-302, 1935.

[Rivas et al. 1981] Rivas, J., Zamaro, J. M., Martin, E. and Pereira, C., "Simple Approximation for Magnetization Curves and Hysteresis Loops", *IEEE Transactions on Magnetics*, Vol. 17, No. 4, pp.1498-1501, 1981.

[Rosenblatt, 1962] Rosenblatt, F., Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms, Spartan Books, New York, 1962.

[Rouve et al. 1995] Rouve, L-L, Waeckerle, T. and Kedous-Lebouc, A., "Determination of the Parameter k of the Generalized Dynamic Preisach Model", *IEEE Transactions on Magnetics,* Vol.32, No. 3, p.1124-1127, 1996.

[Saliah et al. 1997] Saliah, H.H., Lowther, D.A. and Forghani, B., " A Neural Network Model of Magnetic Hysteresis for Computational Magnetics ", *IEEE Transactions on Magnetics*, Vol. 33, No. 5, pp. 4146-4148, 1997.

[Saliah et al. 1998] Saliah, H.H., Lowther, D.A. and Forghani, B., " Modeling Magnetic Materials using Artificial Neural Networks ", *IEEE Transactions on Magnetics*, Vol. 34, No. 5, pp. 3096-3059, 1998.

[Saliah et al. 1999] Saliah, H.H., Lowther, D.A. and Forghani, B., " Generalised Material Models for Coupled Magnetic Analysis ", Submitted to COMPUMAG *99*, Sapporo, Japan, 1999.

[Saliah-Lowther, 1995] Saliah, H. H. and Lowther, D.A., " An Automated Learning System for the design of Magnetic Devices " , Non-linear Electromagnetic Systems, A. J. Moses and A. Basale (Editors) IOS Press, Amsterdam, pp302-305, 1996.

[Saliah-Lowther, 1997a] Saliah, H. H. and. Lowther, D.A., " The Use of Neural Networks in Magnetic Hysteresis Identification " , *Physica B Condensed Matter*, Vol. 233, No. 4, pp. 318-323, 1997.

[Saliah-Lowther, 1997b] Saliah, H. H. and Lowther, D.A., " Magnetic Material Property Identification Using Neural Networks " , *Applied Computational Electromagnetics Society Journal (ACES)*, Vol. 12, No. 2, pp. 44-49, 1997.

[Serpico-Visione, 1998] Serpico, C. and Visone, C., "Magnetic Hysteresis Modeling via Feed-Forward Neural Networks", *IEEE Transactions on Magnetics*, Vol. 34, No. 3, pp. 623-628, 1998.

[Shimodaira, 1996] Shimodaira, H., "A method for selecting Learning Data in The Prediction of Time Series", *Proceedings of the Ninth International Conference*, Fukuoka, Japan, June 4-7, 1996.

[Silvester-Gupta, 1991] Silvester, P.P and Gupta R.P., "Effective computational models for anisotropic soft B-H curves", *IEEE Transactions on Magnetics*, Vol. 27, No. 5, pp. 3804-3807, 1991.

[Silvester-Ferrari, 1996] Silvester, P.P. and Ferrari, R.L., *Finite Elements for Electrical Engineers*, 3rd ed.,Cambridge University Press, New York, 1996.

[Specht, 1990] Specht, D., "Probalistic Neural Networks", *Neural Networks*, Vol. 3, pp.109-118, 1990.[Freeman, 1994] Freeman, J. A., *Simulating Neural Networks with Mathematica*, Addison-Wesley, Reading, Massachusetts, 1994.

[Stoner-Wohlfarth, 1948] Stoner, E.C. and Wohlfarth, E.P, "A mechanism of magnetic hysteresis in heterogeneous alloys", *Phil. Trans. Roy. Soc.*,Vol. A240, pp.194-204, 1948.

[Takahashi et al. 1998] Takahashi, N., Miyabara, S. and Fujiwara,K., " Problems in Practical Finite Element Analysis Using Preisach Hysteresis Model ", *IEEE Transactions on Magnetics*, Vol.35, No. 3, pp.1243-1246, 1999.

[Vajda-Della Torre, 1993] Vajda, F. and Della Torre, E., "Efficient Numerical Implementation of Complete-Moving-Hysteresis Models", *IEEE Transactions on Magnetics*, Vol. 29, No. 2, pp.1532-1537, 1993.

[Vajda-Della Torre, 1994] Vajda, F. and Della Torre, E., "Identification of Parameters of an Accommodation Model", *IEEE Transactions on Magnetics*, Vol. 30, No. 6, pp. 4371-4373, 1994.

[Vajda-Della Torre, 1996] Vajda, F.and Della Torre, E., "Hierarchy of Scalar Hysteresis Models for Magnetic Recording Media", *IEEE Transactions on Magnetics,* Vol. 32, No. 3, pp. 1112-1115, 1996.

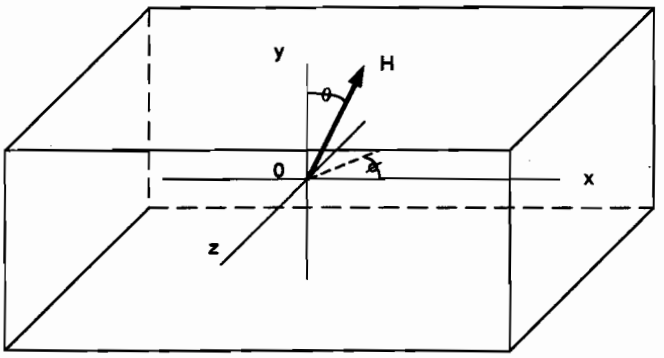
[Vassent et al. 1989] Vassent, E., Meunier, G. and Sabonnadière, J.C., "Simulation of Induction Machine Operation Using Complex Magnetodynamic Finite Element", *IEEE Transactions on Magnetics*, Vol. 25, No. 4, pp. 3064-3066, 1989.

[Verdi-Visintin, 1985] Verdi, C. and Visintin, A., "Numerical Approximation of Hysteresis Problems", *IMA Journal of Numerical Analysis,* No. 5, pp.447-463, 1985

[Visintin, 1994] Visintin, A., *Differential Models of Hysteresis*, Springer, Berlin 1994.

[Wasserman, 1993] Wasserman, P.D., *Advanced Method in Neural Network*, Van Nostrand Reinhold, New York, 1993.

Appendix 1 Principle of construction of the 3D Moving Vector Preisach Model's Lookup Table



$\mathbf{M}_x(\mathbf{H}_x,\theta,\phi)$, $\mathbf{M}_y(\mathbf{H}_y,\theta,\phi)$, $\mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$ are functions for evaluation of magnetization components from first octant data implemented in reference [Oti, 1990].

$\varphi = \cos^{-1}[|\mathbf{H}_x| / \mathbf{H}\sin\theta]$; $\theta = \cos^{-1}[|\mathbf{H}_z| / \mathbf{H}]$

Case 1 : $\mathbf{H}_x \geq 0$

s =1 :

$$\mathbf{M}_x = \mathbf{M}_x(\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

s = -1

if $|\mathbf{H}| < \mathbf{H}_c(\theta,\phi)$

$$\mathbf{M}_x = -\mathbf{M}_x(-\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

else

$$\mathbf{M}_x = \mathbf{M}_x(\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

end if;

Case 2 : $\mathbf{H}_x \leq 0$

s =1 :

if $|\mathbf{H}| < \mathbf{H}_c(\theta,\phi)$

$$\mathbf{M}_x = \mathbf{M}_x(-\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

else

$$\mathbf{M}_x = -\mathbf{M}_x(-\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

s ← -1

end if

s = -1

$$\mathbf{M}_x = -\mathbf{M}_x(-\mathbf{H}_x,\theta,\phi); \mathbf{M}_y = \mathbf{M}_y(\mathbf{H}_y,\theta,\phi); \mathbf{M}_z = \mathbf{M}_z(\mathbf{H}_z,\theta,\phi)$$

Table A1 : Computation of the Components of Magnetisation from the First Octant Data [Oti, 1990]

Appendix 2 Multilayer Feedforward Neural Network Learning Algorithms

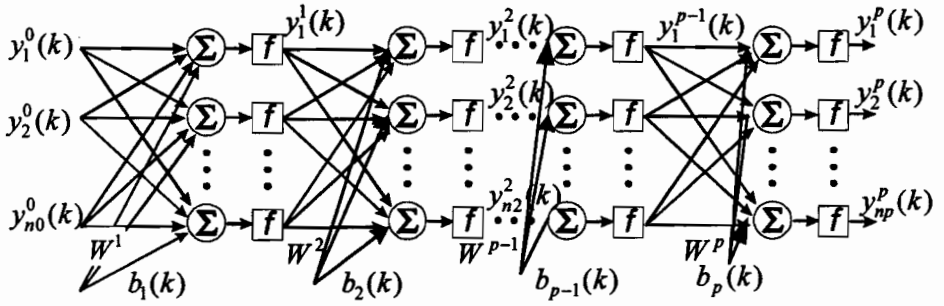


Figure A2.1 Multilayer Feedforward Neural Network Architecture

The figure A2.1 above shows the architecture of a $p+1$ layer feedforward neural network with input and output vectors $y_j^0(j=1,\cdots,n_0)$ and $y_j^p(j=1,\cdots,n_p)$.

The output y_j^p is computed by the equation

$$y_i^p = f_i^p\left(\sum_{j=1}^{n_{p-1}} \omega_{ij}^p y_i^{p-1} + b_i\right) \qquad \text{(A2.1)}$$

where W^p is the weight matrix related to the p^{th} layer, where $b_i(i=1,...,p)$ are the bias vectors for each neuron i and where $f_i^p[.]$ is the transfer function discussed earlier in this thesis.

The learning occurs when the weights stabilise after a cost function minimisation process. The most common cost function is the quadratic error between the computed and the desired ouputs, respectively, y_i^p and y_d, determined by

$$J(W^1 \ldots W^p) = \frac{1}{2} \sum_{k=1}^{n_e} \sum_{s=1}^{n_p} e_s^2(k) \qquad \text{(A2.2)}$$

where $e_s(k) = y_d^s(k) - y^s(k, \ W^1, \ldots, W^p)$, where n_e is the number of samples and n_p is the length of the output vector y.

A number of training strategies are used to achieve the quadratic error minimisation task, such as the gradient descent and backpropagation of error, the Newton and Quasi-Newton methods as well as the Levenberg-Marquardt techniques.

The gradient descent and backpropagation of error technique

The goal is to evaluate the error on the output neurons of the hidden layers by calculating the error on the output layer. For example, the gradient evaluated at the q^{th} layer with n_q neurons is given by the following expression

$$g_{ij}^q = \frac{\partial J}{\partial W_{ij}^q} = \sum_{k=1}^{n_e} \sum_{s=1}^{n_p} \frac{\partial J}{\partial y^s(k)} \frac{\partial y^s(k)}{\partial W_{ij}^q} \qquad \text{(A2.3)}$$

and the weight update formula is given by the expression

$$W(k+1) = W(k) - \eta \, H(k) \nabla(k) \qquad \text{(A2.4)}$$

where $\nabla(k)$ is the gradient, η is the learning coefficient updated at each iteration and $H(k)$ is the estimation of the inverse Hessian , i.e. the matrix of second derivatives of the error function.

Newton Technique

The Newton approach is based on a Taylor series approximation, but truncated after three terms, as given by

$$f(x) = f(x_k) + \nabla f(x_k)^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(x_k)\Delta x + o(\Delta x^3) \quad \text{(A2.5)}$$

After dropping higher order terms , the equation (A2.5) yields

$$\tilde{f}(x; x_k) = f(x_k) + \nabla f(x_k)^T \Delta x + \frac{1}{2}\Delta x^T \nabla^2 f(x_k)\Delta x \quad \text{(A2.6)}$$

where $\tilde{f}(x; x_k)$ is a functional approximated at a given point x_k .

Ans finally, an iterative sequence can be inferred so that the gradient of $\tilde{f}(x; x_k) = 0$ at the next point x_{k+1}. After a few manipulations, the following equation will arrived at:

$$x_{k+1} = x_k - \nabla^2 f(x_k)^{-1} \nabla f(x_k) \quad \text{(A2.7)}$$

Quasi-Newton Method

This approach makes use of line searches, a constrained minimisation along a specific search direction. The Quasi-Newton method uses successive steps to construct an approximation to the inverse Hessian matrix. Brayden, Fletcher, Goldfarb and Shanno (BFGS)[Fletcher, 1987] have shown that $H(k)$ can be evaluated at each iteration using the following simplified relation

$$H(k+1) = H(k) + \left[1 + \frac{\Delta^T H(k)\Delta}{A^T \Delta}\right]\frac{AA^T}{A^T \Delta} - \frac{A\Delta^T H(k) + H(k)\Delta A^T}{A^T \Delta} \quad \text{(A2.8)}$$

where

$$\Delta = \nabla(k+1) - \nabla(k) \quad \text{(A2.9)}$$

and

$$A = -\eta H(k)\nabla(k) \quad \text{(A2.10)}$$

Levenberg-Marquardt Learning

To alleviate the problem of computing a large number of second derivatives in the Hessian, the Levenberg-Marquardt method uses an approximation to the Hessian. This can be quickly calculated during the backpropagation process, that produces the gradient. In this case, the weights are adjusted according to equation

$$W(k+1) = W(k) + \left(J_b^T J_b + \mu I\right)^{-1} J_b^T e \qquad \text{(A2.11)}$$

where :

J_b is the Jacobian , i.e. the derivatives of each error with respect to the weights;

e is the error vector;

μ , a scalar, is adjustable with respect to the error vector e_k . (The smaller the error, the larger μ is.)

Appendix 3 Results from Adaptive Logic Networks

A Sample of Experimental Data BHALN00.dat

H Field (A/m)	B(Tesla)	ΔB (incremental variation)
10000.000000	1.027791	-0.000002746
8000.000000	1.026574	-0.000008936
6000.000000	1.023828	-0.000012117
4000.000000	1.014892	-0.000027627
3000.000000	1.002775	-0.000029194
2000.000000	0.975148	-0.000022002
1500.000000	0.945954	-0.000027504
1250.000000	0.923952	-0.000036454
1000.000000	0.896448	-0.000044760
750.000000	0.859994	-0.000057026
500.000000	0.815234	-0.000074485
250.000000	0.758208	-0.000160722
0.000000	0.683723	-0.000299434
250.000000	0.523001	-0.000384258
500.000000	0.223567	-0.000312167
.	.	.
.	.	.
.	.	.
500.000000	-0.223565	0.000312166
750.000000	0.160691	0.000148942
1000.000000	0.472857	0.000146967
1250.000000	0.621799	0.000157487
1500.000000	0.768766	0.000065134
2000.000000	0.926253	0.000018353
3000.000000	0.991387	0.000013231
4000.000000	1.009740	0.000003603
6000.000000	1.022971	0.000001217
8000.000000	1.026574	0.0
10000.000000	1.027791	0.000001217

Program in ALN Definition Language to Learn an Hysteresis Behavior

```
// BHALN01.ADL
//
// create ALN with 3 variables, third variable is the output
ALN aBHALN01(3, 3)
{
   var 1
   {
   min = -10000;
   max = 10000;
  epsilon = 0.001;
wmin = 0;
 };
   var 2
   {
   min = -1.5;
   max = 1.5;
  epsilon = 0.0001;
wmin = 0;
 };
   var 3
   {
   min = -0.0004;
```

```
  max = 0.0004;
  epsilon = 0.00000005;
  };
// The ALN will have 7 layers, and fanin 2 for the top layers.
// The numbers of elements in successive layers
// (top=output, to above the bottom) are 1, 2, 4, 8, 16, 32.
// The last fanin is also 3, so there are  96 linear-threshold elements
//  in the bottom (= 7th) layer.  If each linear piece matches three data
// points exactly, there are easily enough pieces for all 144 data points.
 tree = OR(7, 2, 3, FULL);
};@echo off
// MULT.SHL
// Program in ALN Shell Language for hystereis path tracking
// First make sure there are no objects in the working space
clear
// load in ALN from BHALN01.ADL
LoadADL " bhaln01.adl "
// Train the tree on the data
// echo is turned on so we can get the train command
// interactively if required high learning rate to start
// with, in order to get.a fast, coarse approximation
@echo on
// Train an ALN, using the data provided in the bhaln01.dat
```

```
// file, for  500 epochs with RMS tolerance for delta B egals
// 0.00001, and a learning rate 0.1
aBHALN01.Train(500, 0.00001, 1.0, " bhaln01.dat ")
@echo off
// continue training with a slower learning rate to " tune "
// the surface for better generalization
@echo on
aBHALN01.Train(500, 0.000005, 0.1, " bhaln01.dat ")
@echo off

// Evaluate on training set
aBHALN01.OutputFmt(3, r)
aBHALN01.Eval(3, 144, " bhaln01.dat ", " bhaln01o.dat ")

// Test generalisation on some fractional numbers
//aMult.Eval(3, 14400, " frac.dat ", " fracout.dat ")

// Create a 4 layers decision tree on variable 3 and save
// for use in multest.shl
DTREE dBHALN01(aBHALN01, 3, 4)
dBHALN01.Write(" BHALN01.dtr ");
```

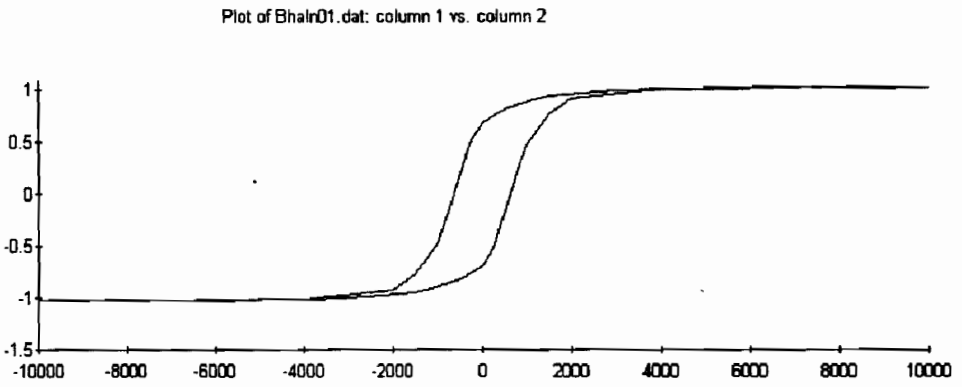


Figure A3.1 B-H Curve Ploted from an ALN

Generated Decision Tree file

// BHALN01.dtr exported on Tue May 19 17:35:46 1998

// ALN Decision Tree file

VARIABLES = 3;

// Maximum and minimum values for the field (A/m)

x0 : [-10000, 10000];

// Maximum and minimum values for the induction (Tesla)

x1 : [-1.5, 1.5];

// Maximum and minimum values for the incremental variation

// of the induction (Tesla)

x2 : [-0.0004, 0.0004];

OUTPUT = x2;

// The training gives 96 piecewise linear functions as shown bellow

LINEARFORMS = 96;

0 : -1 (x2 + 5.525780289462935E-005);

1 : -1 (x2 + 5.314660343819683E-005);

2 : -1 (x2 + 5.509529264766634E-005);

3 : -1 (x2 + 5.968951847731343E-005);

4 : -1 (x2 + 5.719889198657595E-005);

5 : -1 (x2 + 5.866102874427959E-005);

6 : -1 (x2 + 5.70936612947134E-005);

7 : -1 (x2 + 4.544087631198782E-005);

8 : -1 (x2 + 5.784146579023843E-005);

9 : -1 (x2 + 4.144284850500426E-005);

. . . . . .

. . . . . .

91 : -1 (x2 + 5.125856885642749E-006);

92 : -1 (x2 + 3.619435948892792E-006);

93 : -1 (x2 + 2.421819194630696E-006);

94 : -1 (x2 + 1.600902132381746E-006);

95 : -1 (x2 + 3.435025933796655E-006);

Appendix 4 Neural Networks Based Material Model Interface Integration and the Requirements for Finite Element Implementation

Routine: *newh*

The routine handles the neural network materials, that we have called, for our purposes, "PMNN". Any other materials in the problem are handled from the material database in the usual way. The routine uses the latest estimate of the magnetic field to update the state of the material in the tetrahedron for a three dimensional solution or triangles for 2D. In this way, the field and magnetisation components previously computed are provided to the *mline* routine. If the field is converged at the end of a nonlinear iteration, then the flag "Solution_Converged" is set to 1; otherwise, it is set to 0.

Routine: *mhline*

The routine provides a linear $M-H$ model ($M = QH + M_0$) that the finite element (FE) solver will use to take the next step in the iteration. First-order tetrahedra (or triangles) only are considered. (Thus H and M do not vary with position in the tetrahedron.) The matrix Q is symmetric, so only the lower triangle will be used by the FE solver. If this is not one of the tetrahedra in the user's material database, the Not_Present flag should be set to 1. Then, the solver will get the linear model from the material database in the usual way.

Routine: *getm*

The routine provides the magnetisation in a tetrahedron and reports the magnetisation components m_x, m_y, m_z, previously stored by the *newh* routine.

Routine: *initmh*

The routine initialises the user's material model for one tetrahedron. After this initialisation, calls to *mhline* or *getm* for this tetrahedron should return valid answers. If this is not one of the tetrahedra in the user's material database, the Not_Present flag should be set to 1. Then the solver will get the linear model from the material database in the usual way. This routine will be called for each tetrahedron, once for each problem defined. It initialises the input MH. The initialisation could start at 0 and the path is the first magnetisation-saturation-descending curve-ascending curve. Other starting points could be considered (e.g. remanence at the state previous to the former one (delay 2), i.e., $M_{n-2} = M_r$ or at the saturation point where $M_{n-2} = M_s$.

Routine: *openmh*

The routine opens the user's material model database. If the database does not exist, the flag " $M-H$_Database" is set to 0; otherwise, it is set to 1. This routine will be called just once per run of the solver. It reads "PMNN.INI", i.e., the neural network initialisation file, as shown in Figure A4.3. Memory for the neural network weight matrices and biases are allocated and when a neural network material model is considered, a switch is made to a single input single output (anhysteretic) or multiple inputs single output (hysteretic).

Routine: *closmh*

The routine closes the user's material model database. If the database does not exist, the flag " $M-H$_Database" is set to 0; otherwise, it is set to 1. The routine will be called just once per run of the FE solver.

Routine: *inmh*

The routine checks the tetrahedra status. If this is not one of the tetrahedra in the user's material database, the Not_Present flag should be set to 1. Then the solver will get the model from the material database in the usual way.

```
Open

        Initialise neural net weights (read in PMNN.INI)
        and/or read last 2 previous states from the material database

for each Newton step :

{

Matrix Assembly

        For each element {
                For first time
                        call initmh

        Calculate H
        Call newh (input the H vector to material database)
        Call mhline to get Q and μ0 (user defined μ-H
                                relationship)

}

Solve the system of equations


}

closemh (update the state of the material for each element in the
database)
```

Figure A4.1 Implementation Steps

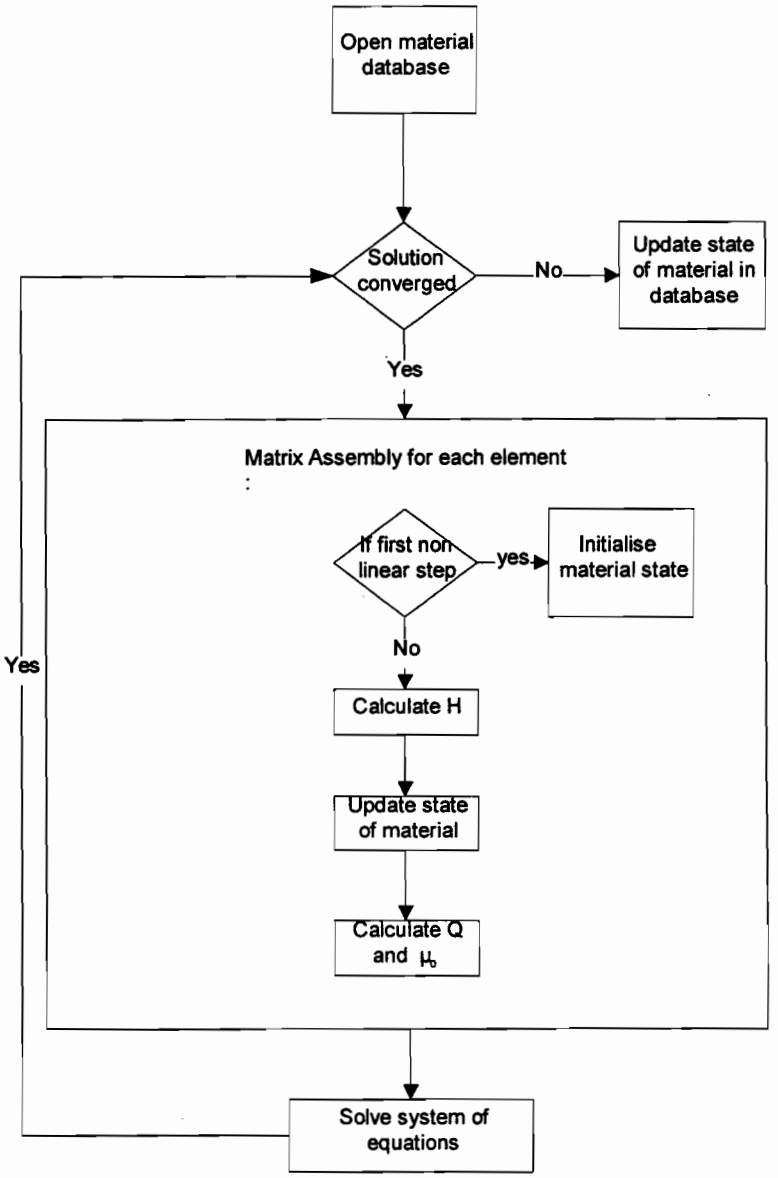


Figure A4.2 Implementation Flowchart

Figure A4.3 The Neural Network Initialisation File (PMNN.INI)

```
# Information switch: Use Artificial Neural Networks Training
# Results and switch to a Multiple Inputs Case (200); or Single
# input single output case (100)

ANN  200

# Matrices (F) and Biases (B) Files to read for each axis

F1X  Neural_Net_First_Layer_Weights_Matrix_for_Axis_X.dat
F2X  Neural_Net_second_Layer_Weights_Matrix_for_Axis_X.dat
F1Y  Neural_Net_First_Layer_Weights_Matrix_for_Axis_Y.dat
F2Y  Neural_Net_second_Layer_Weights_Matrix_for_Axis_Y.dat
F1Z  Neural_Net_First_Layer_Weights_Matrix_for_Axis_Z.dat
F2Z  Neural_Net_second_Layer_Weights_Matrix_for_Axis_Z.dat
B1X  Neural_Net_Input_Units_Bias_Vector_for_Axis_X.dat
B2X  Neural_Net_Hidden_Units_Bias_Vector_for_Axis_X.dat
B1Y  Neural_Net_Input_Units_Bias_Vector_for_Axis_Y.dat
B2Y  Neural_Net_Hidden_Units_Bias_Vector_for_Axis_Y.dat
B1Z  Neural_Net_Input_Units_Bias_Vector_for_Axis_Z.dat
B2Z  Neural_Net_Hidden_Units_Bias_Vector_for_Axis_Z.dat

# Number of Inputs (N)

N    5

# Number of Hidden Units (MN)

MM   12

# Number of outputs (P)

P    1
# Saturation Field Strength (Oe) for X, Y, Z Axis respectively

HXMAX  10000
HYMAX  10000
HZMAX  10000

# Saturation Magnetisation (emu/cc) for X, Y, Z Axis respectively

MXMAX  600
MYMAX  600
MZMAX  600
```

Appendix 5 A Sample Artificial Neural Networks Material Model Interface Routine for FE Solvers

```
/* Author : Hamadou Saliah-Hassane
 *
 * propagate  is called in  the newh, mhline and inmh routines to compute M(H)
 * given the following parameters:
 *
 *       tet                     Tetrehedra no.
 *
 *       idir                    Index of the field direction (1, 2 or 3, standings for x, y or z)
 *       (h1, mg1), (h2 mg2), (h3, mg3) Respectively two previous magnetisation states and the
 *       current state.
 *
 *       matnam                                  Material name
 *       hxmax, mxmax, hymax, mymax, hzmax, mzmax :Saturation magnetisation for x, y and z
 *
 *
 */

void propagate(int tet, int idir, double *h1, double *h2, double *h3, double *mg3,
          double *mg2, double *mg1)
{

    int j, k;
    double B=0;
  char matnam[5];
    double hxmax, hymax, hzmax;
    double mxmax, mymax, mzmax;

    strcpy(matnam, tetmat[tet-1]);

    if( !strncmp(matnam,"PMNN",4) ) {
        hxmax = HXMAX; hymax = HYMAX; hzmax = HZMAX;
        mxmax = MXMAX; mymax = MYMAX; mzmax = MZMAX;
    }
    else if( !strncmp(matnam,"PMN1",4) ) {
        hxmax = 3*HXMAX; hymax = 3*HYMAX; hzmax = 3*HZMAX;
        mxmax = 3*MXMAX; mymax = 3*MYMAX; mzmax = 3*MZMAX;
    }
    else if( !strncmp(matnam,"PMN2",4) ) {
        hxmax = HXMAX; hymax = HYMAX; hzmax = HZMAX;
        mxmax = MXMAX; mymax = MYMAX; mzmax = MZMAX;
    }
```

```
   /* Build the input vector */


if (idir==1) {
    MH[1]=h3[1]/hxmax; MH[2]=mg3[1]/mxmax;MH[3]=h2[1]/hxmax;
    MH[4]=mg2[1]/mxmax; MH[5]=h1[1]/hxmax;
}

if (idir==2) {
    MH[1]=h3[2]/hymax; MH[2]=mg3[2]/mymax;MH[3]=h2[2]/hymax;
    MH[4]=mg2[2]/mymax; MH[5]=h1[2]/hymax;
}

if (idir==3) {
    MH[1]=h3[3]/hzmax; MH[2]=mg3[3]/mzmax;MH[3]=h2[3]/hzmax;
    MH[4]=mg2[3]/mzmax; MH[5]=h1[3]/hzmax;
}

   /* Propagate the input MH through layer M M, N and M
    * with
    * W1(X, Y and Z) and W2(X,Y and Z)   First and second layer weights */
    * B1(X, Y and Z) and B2(X,Y and Z)    Biases */
    *
   /* Layer N->M */
   for (k=1;k<=M;k++) {
      M12[k]=0.0;
      for (j=1;j<=N;j++) {
            if (idir==1)
                  M12[k]+=MH[j]*W1X[k][j];
            else if (idir==2)
                  M12[k]+=MH[j]*W1Y[k][j];
            else if (idir==3)
                  M12[k]+=MH[j]*W1Z[k][j];
      }
      if (idir==1)
         B = B1X[k];
      else if (idir==2)
         B = B1Y[k];
      else if (idir==3)
         B = B1Z[k];

         M12[k]=1-2./(exp(2*M12[k]+B)+1);
   }


   /* Layer M->P */
      /*M12[M+1]=1.0;*/
   for (k=1;k<=P;k++) {
      mg1[idir]=0.0;
      for (j=1;j<=M;j++) {
               if (idir==1)
```

```
                mg1[idir]+=M12[j]*W2X[k][j];
            if (idir==2)
                mg1[idir]+=M12[j]*W2Y[k][j];
            if (idir==3)
                mg1[idir]+=M12[j]*W2Z[k][j];
        }

    if (idir==1)
        B = B2X[k];
     else if (idir==2)
        B = B2Y[k];
     else if (idir==3)
        B = B2Z[k];

     mg1[idir] = 1-2./(exp(2*mg1[idir]+B)+1);
  }
  if (idir == 1)
    mg1[1]*=mxmax;
  else if (idir == 2)
    mg1[2]*=mymax;
  else
    mg1[3]*=mzmax;
}

/*
    nnet with a single input and a single output

*/
void propagateSISO_BP(double *h, double *mg)
{
    int j, k;

  /* Propagate the input given MH */
  h[1]/=HXMAX; h[2]/=HYMAX; h[3]/=HZMAX;
  /* Layer N->M */
    for (k=1;k<=M;k++) {
    M12[k]=0.0;
     for (j=1;j<=N;j++)
        M12[k]+=h[j]*W1X[k][j];
     M12[k]=2./(1 + exp(-2*(M12[k]+B1[k])))-1;
  }

  /* Layer M->P */
  for (k=1;k<=P;k++) {
        mg[k]=0.0;
        for (j=1;j<=M;j++)
            mg[k]+=M12[j]*W2X[k][j];
        mg[k] += B2[k];
  }
  mg[1]*=MXMAX; mg[2] *=MYMAX; mg[3] *=MZMAX;
}
```