INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

Bell & Howell Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®

Université d'Ottawa · University of Ottawa

COMPUTATIONAL ASPECTS IN MODELLING ELECTROMAGNETIC FIELD PARAMETERS IN MICROSTRIPS

by

Arto Chubukjian B.A.Sc., M. Eng., P.Eng.

A Thesis Submitted to the
School of Graduate Studies and Research
in Partial Fulfillment of the Requirements
for the Degree of
Doctor of Philosophy

Ottawa-Carleton Institute for Electrical Engineering
Department of Electrical Engineering
School of Information Technology and Engineering
University of Ottawa
May 2000

ABSTRACT

This thesis introduces the results of a detailed investigation and analysis into the key aspects of the Vector Finite Element Method, the critical relationships between the VFEM zone discretization and the associated solution matrix equations, the discovery of the original fill-in laws and the fill-in prediction methods, and their impact on the computational aspects and processes. Furthermore, the work included the design and implementation of accelerated matrix solution models, and their successful implementation and application to various electromagnetic problems and the presentation of the excellent performance results obtained. Also, one of the problem examples used in the investigation contributed to the validation of a method to reduce the characteristic impedance of a microstrip by means of grooves. Finally, the modelling of the electromagnetic problems by Artificial Neural Networks, and the successful investigation of an original concept – training by decimation – led to the validation of Artificial Neural Networks as a real-time modelling tool which completed the work of this thesis.

ACKNOWLEDGEMENTS

I would like to express my sincere gratitude and appreciation to my supervisors Dr George Costache and Dr Emil Petriu for their support, wise guidance, and encouragement that they always gave me during the entire period of this work. My special thanks are due to my advisory committee members, my colleagues at the University of Ottawa, the departmental teaching and technical staff, the ever-efficient and friendly support staff, all of whom who always helped and encouraged me in many ways, and contributed to the success of my work. Finally, my thanks are due to my employer, my colleagues at work, and to my family.

Table of Contents

Chapter 3 – Relationship of VFEM Zone Discretization and Solution Matrix Structure

Chapter 4 – Computational Aspects of Vector Finite Element Method

List of Symbols

∇ – Del Operator

∇_t – Del Operator (transverse components)

μ_0 – Permeability of free space – $4\pi \times 10^{-7}$ henry/metre

ε_0 – Permittivity of free space – $(36\pi \times 10^9)^{-1}$ farad/metre

μ_r – Relative permeability of a medium

ε_r – Relative permittivity of a medium

η_0 – Intrinsic impedance of free space – ~377 ohms

η – Intrinsic impedance of a medium

γ_e – Parameter of boundary S2

ANN – Artificial Neural Network

CPU – Central Processing Unit

e – Neural network error vector, also 2.718282

E – Electric field intensity in volts/metre

f – frequency

FDTD – Finite Difference Time Domain

FEM – Finite Element Method

GHz – Gigahertz

H – Magnetic field intensity in amperes/metre

HL – Hidden Layer

I – Current in amperes

I/O – Input/Output

$j - (-1)^{0.5}$

J – Electric current density in amperes/metre^2

k_0 – Propagation constant of free space

MHz – Megahertz

$\hat{n}$ – Unit normal vector

RAM – Random Access Memory

TEM – Transverse Electromagnetic

TLM – Transmission Line Matrix

V – Voltage in volts

VFEM – Vector Finite Element Method

$\hat{x}, \hat{y}, \hat{z}$ – Unit vectors in rectangular coordinate system

Z – Impedance in ohms

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Motivation

When this work was undertaken, the aim of the research effort was to develop an appropriate paradigm to predict electromagnetic field interaction with and penetration into inhomogeneous materials or media and irregularly shaped structures, both organic (human body) and inorganic (buildings). Since analytical techniques were unsuitable for such a task, computational electromagnetics were the means by which the above goal could have been achieved. The model selected for the task was the Vector Finite Element Method, recently developed at the time [1]-[17].

While the Finite Element Method has been around for a long time, and it was widely used in many disciplines, the initial applications in electromagnetics problems used the existing model, the so-called Nodal Finite Element Method, which employs nodal basis functions. According to Jin [18], when used in electromagnetics, this method is known to generate non-physical or spurious solutions as a result of the serious problem involving the correct application of the boundary

conditions at the dissimilar media interfaces, and also "... studies showed that the solution does not satisfy the divergence condition for the electric and the magnetic field in a source-free region." [18].

The Vector Finite Element Method, by contrast, uses edge-centred basis functions and the boundary conditions are satisfied naturally, as the tangential field components are continuous across element or material boundaries, and as a result, no spurious solutions are generated. Moreover, the divergence conditions are satisfied naturally. Therefore, the Vector Finite Element Method was chosen as the most suitable vehicle for the intended research.

However, due to unforeseen circumstances, this research effort had to be redirected in the recent past. The initial goal was modified and the emphasis was shifted towards the computational concepts as a paradigm and their possible refinements, rather than the specific application of the computational technique to the electromagnetic problem defined above. In conjunction with this new goal and for particular reasons, the problem examples of the investigation were redefined, namely, microstrip structures and their electromagnetic parameters.

While recognizing that the Vector Finite Element Method has now gained some maturity, as attested by the many papers and publications during the last few years, it also must be recognized that modern electromagnetic problems require larger and larger domains and extremely fine discretizations, which, in turn, impose solution requirements that challenge the available computational resources. It should be mentioned here that the Vector Finite Element Method is not unique in this respect, other types of computational/numerical methods are equally susceptible to this issue.

This situation is then the prime motive behind the current work. Its goal is a thorough analysis of the characteristics of the solution methodology, and the keen understanding of its structural attributes, aspects, and relationships that would enable the development of:

a) a vastly improved electromagnetic problem solution performance,

b) use of a modelling approach such as an Artificial Neural Network that could further enhance the results obtained in a),

c) combination of a) and b) into a real-time electromagnetic solution paradigm described graphically by Figure 1.1.

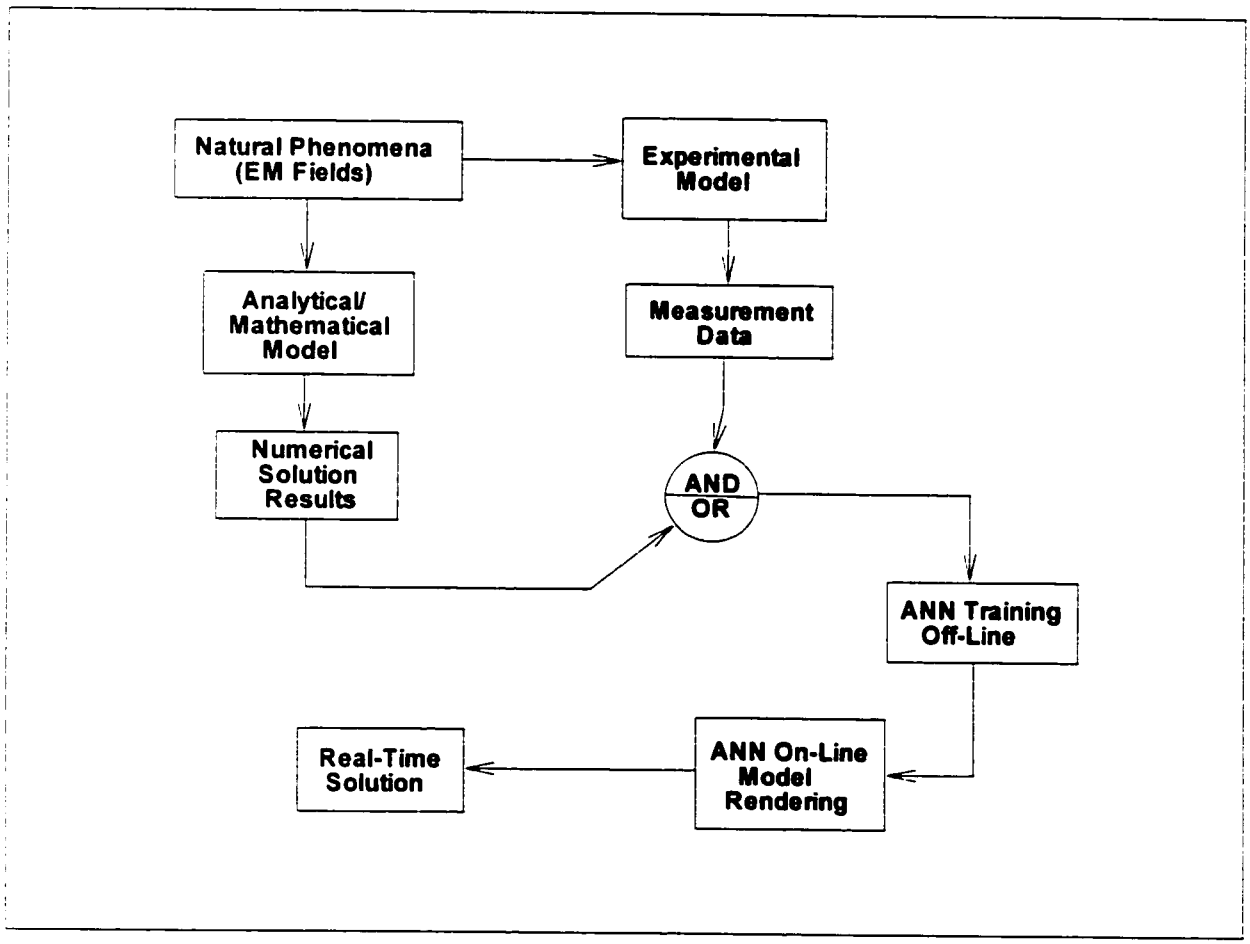


Figure 1.1 – Artificial Neural Network Modelling Process (Analytical and Experimental)

The above enumerated goals have been reached during the course of this work and the results are outlined in this document.

1.2 Organization of the Thesis

The thesis has eight chapters covering the work carried out. Chapter 1 is a brief introduction. Chapter 2 presents an outline of the Vector Finite Element Method. Chapter 3 covers the results involving the critical characteristics and relationships of the Vector Finite Element Method zone discretization and the resulting matrix equation structure, including the matrix fill-in laws and fill-in prediction methods. Chapter 4 presents a brief overview of the matrix solution techniques, and then describes the matrix decomposition aspects, the memory storage, and the matrix solution acceleration techniques, and finally some other speed-up techniques implemented in the course of this work. Chapter 5 presents various application examples such as microstrip problems and the overall performance measure of the Vector Finite Element Method as implemented in this work. Chapter 6 presents a brief outline of the Artificial Neural Network and its various characteristics and architectures. Chapter 7 presents the application of Artificial Neural Networks to microstrip problems, and the resulting performance. Finally, Chapter 8 incorporates the conclusions and summarizes the original work accomplished. An extensive bibliography is located at the end of the thesis.

1.3 Contributions of the Work

During the course of the work carried out in developing a real-time solution prototype for

electromagnetic problems, the following achievements and contributions were realized.

a) a detailed analysis of the VFEM zone discretization and the identification and definition of the critical relationships with the associated solution matrix equations;

b) a thorough, and as far as is known for the first time, analysis of the matrix structures and the matrix-VFEM zone relationships, and the discovery of their many key attributes, and their impact on the computational aspects;

c) design and implementation of accelerated matrix solution models, and their successful application to various problems;

d) adoption and successful application of the VFEM methodology together with the use of the above-mentioned concepts to several electromagnetic problems, and the obtained excellent computational performance results;

e) inception and successful demonstration of the concept of *training by decimation* of the Artificial Neural Networks and the achieved speed-up in training while maintaining the integrity of the error performance;

f) successful modelling of electromagnetic problems by means of Artificial Neural Networks, and the implementation of a rapid solution prototype having excellent error performance.

CHAPTER 2

AN EXAMPLE OF NUMERICAL ELECTROMAGNETICS:

THE VECTOR FINITE ELEMENT METHOD

2.1 Introduction

Electromagnetic analysis can be described as a problem of solving Maxwell's equations subject to the applicable boundary conditions. In the physical environment, analytical techniques are not always the most practical approach to solve boundary-value problems, as analytical solutions can be obtained in a few cases only. As a result, an entire field of numerical techniques, called Computational Electromagnetics, has evolved to obtain approximate solutions to the boundary-value problems. The Vector Finite Element Method (VFEM) is one such numerical technique and has been successfully applied to a large variety of electromagnetic problems. In this Chapter, an overview of the Vector Finite Element Method is presented.

2.2 Vector Finite Element Method

The finite element method has been applied to time-harmonic electromagnetic field problems extensively and successfully. A problem is described by a vector wave equation

$$\nabla \times (\frac{1}{\mu_r} \nabla \times E) - k_0^2 \varepsilon_r E = -jk_0 \eta_0 J \tag{2-1}$$

in the case of the electric field, and by

$$\nabla \times (\frac{1}{\varepsilon_r} \nabla \times H) - k_0^2 \mu_r H = \nabla \times (\frac{1}{\varepsilon_r} J) \tag{2-2}$$

in the case of the magnetic field.

The applicable boundary conditions usually encountered are given by:

$$\hat{n} \times E = 0 \qquad \text{(Dirichlet)} \tag{2-3}$$

$$\hat{n} \times \nabla \times H = 0 \qquad \text{(Neumann)} \tag{2-4}$$

for electrically conducting surfaces, and by

$$\hat{n} \times \nabla \times E = 0 \qquad \text{(Neumann)} \tag{2-5}$$

$$\hat{n} \times H = 0 \qquad \text{(Dirichlet)} \tag{2-6}$$

for magnetically conducting surfaces.

At the interfaces separating the two different media 1 and 2, the continuity conditions are applied:

$$\hat{n} \times E_1 = \hat{n} \times E_2 \tag{2-7}$$

$$\hat{n} \times H_1 = \hat{n} \times H_2 \tag{2-8}$$

Alternately, these may be written as

$$\frac{1}{\varepsilon_{r1}} \hat{n} \times (\nabla \times H_1) = \frac{1}{\varepsilon_{r2}} \hat{n} \times (\nabla \times H_2) \tag{2-9}$$

$$\frac{1}{\mu_{r1}} \hat{n} \times (\nabla \times E_1) = \frac{1}{\mu_{r2}} \hat{n} \times (\nabla \times E_2) \tag{2-10}$$

Before the system of equations is solved, the Dirichlet boundary conditions such as (2-3), must be imposed on the conducting surface. The boundary conditions to be satisfied by the magnetic field on the conducting surface is (2-4), which is equivalent to $\hat{n} \bullet H = 0$ [18].

"In the case when the computational domain contains different media interfaces, the continuity conditions (2-7) and (2-8) have to be imposed explicitly, and leave (2-9) and (2-10) to be satisfied automatically as the natural conditions." [18]. To satisfy the continuity conditions (2-7) and (2-8), a single value can be assigned to the finite element attribute at the media interface.

In nodal finite elements, the element attribute is the node, and assigning a single value to it also enforces the continuity of normal fields. This situation is in conflict with the actual boundary conditions

$$\hat{n} \bullet (\varepsilon_{r1} E_1) = \hat{n} \bullet (\varepsilon_{r2} E_2) \tag{2-11}$$

$$\hat{n} \bullet (\mu_{r1} H_1) = \hat{n} \bullet (\mu_{r2} H_2) \qquad (2\text{-}12)$$

which are not exactly satisfied. This problem has been shown to be alleviated by using very fine discretization near the interface. The idea, tested by Ferrari and Naidu [18],[19], yields rapidly (rather than abruptly) changing normal components of the field across the media boundary, and such a change agrees well with the requirement of (2-11) and (2-12). This approach has major disadvantages. Also, it is known that several other techniques have been tried, such as "....from the use of special interpolatory functions to the use of penalty functions, ..." [20].

In the case of vector or edge finite elements, these naturally satisfy the boundary conditions, due to the use of vector elements where the degrees of freedom are assigned to the edges rather than the nodes of the finite elements. The vector finite elements are free of the disadvantages of the nodal finite elements such as:

a) the existance of non-physical or spurious solutions due to the lack of enforcement of the divergence conditions;

b) the imposing of the boundary conditions at media interfaces as well as conducting surfaces; and

c) the treatment of conducting and dielectric edges and corners due to field singularities associated with these structures.

2.3 Finite Element Method Solution Applied To Boundary Value Problems

An electromagnetic field problem is a boundary-value problem. A typical boundary-value problem can be defined by a governing differential equation in a problem domain Ω,

$$L\phi = f \tag{2-13}$$

together with the boundary conditions on the boundary Γ that encloses the domain Ω,. L is the differential operator, f is the excitation or the forcing function, and ϕ is the unknown quantity. To obtain the solution to this boundary-value problem, we apply the Ritz Method.

The inner product is defined as:

$$\langle \phi, \varphi \rangle = \int_{\Omega} \phi \varphi^{*} d\Omega \tag{2-14}$$

where * denotes the complex conjugate. If the operator L is self-adjoint, i.e. ,

$$\langle L\phi, \varphi \rangle = \langle \phi, L\varphi \rangle \tag{2-15}$$

and positive definite, i.e.,

$$\langle L\phi, \phi \rangle = \begin{cases} > 0 & \phi \neq 0 \\ = 0 & \phi = 0 \end{cases} \tag{2-16}$$

then the solution to (2-13) can be obtained by minimizing the functional [18]:

$$F(\phi) = \tfrac{1}{2}\langle L\phi, \phi \rangle - \tfrac{1}{2}\langle \phi, f \rangle - \tfrac{1}{2}\langle f, \phi \rangle \tag{2-17}$$

with respect to ϕ where ϕ denotes the trial function.

2.3.1 Interpolation Functions

The trial function ϕ can be represented by a first (linear) order polynomial consisting of the products of expansion functions defined over the entire domain Ω and constant coefficients to be determined.

$$\phi^e = \sum_{j=1}^{n} N_j^e \phi_j^e = \{N^e\}^T \{\phi^e\} = \{\phi^e\}^T \{N^e\} \tag{2-18}$$

where n is the number of nodes in the finite element, ϕ_j^e and N_j^e are the value of ϕ and the interpolation function (also known as the expansion or basis function) respectively, at node j of the element e.

Note: { } denotes a column vector; superscript T denotes the transpose of a vector;
[ ] denotes a square matrix.

The highest order of N_j^e is referred to as the order of the element, thus if N_j^e is a linear function, then element e is a linear element. An important feature of the functions N_j^e is that they are nonzero only within each element e, and outside the element, they vanish.

Equation (2-17) can be expressed as

$$F(\phi) = \sum_{e=1}^{M} F^e(\phi^e) \tag{2-19}$$

where M is the number of elements comprising the entire domain Ω and the functional for each element e as

$$F^e(\phi^e) = \tfrac{1}{2}\langle L\phi^e, \phi^e\rangle - \tfrac{1}{2}\langle \phi^e, f\rangle - \tfrac{1}{2}\langle f, \phi^e\rangle \qquad \text{or}$$

$$F^e(\phi^e) = \tfrac{1}{2}\langle \phi^e, L\phi^e\rangle - \langle f, \phi^e\rangle \tag{2-20}$$

$$F^e(\phi^e) = \tfrac{1}{2}\int_{\Omega^e} \phi^e L\phi^e d\Omega - \int_{\Omega^e} f\phi^e d\Omega \tag{2-21}$$

Substituting (2-18) into (2-21) we obtain,

$$F^e(\phi^e) = \tfrac{1}{2}\int_{\Omega^e} \{\phi^e\}^T \{N^e\} L \{N^e\}^T \{\phi^e\} d\Omega - \int_{\Omega^e} f \{\phi^e\}^T \{N^e\} d\Omega$$

$$F^e(\phi^e) = \tfrac{1}{2}\{\phi^e\}^T \int_{\Omega^e} \{N^e\} L \{N^e\}^T d\Omega \{\phi^e\} - \{\phi^e\}^T \int_{\Omega^e} f\{N^e\} d\Omega \tag{2-22}$$

which can be written in matrix form as

$$F^e = \tfrac{1}{2}\{\phi^e\}^T [K^e] \{\phi^e\} - \{\phi^e\}^T \{b^e\} \tag{2-23}$$

where $[K^e]$ is an (n x n) matrix and $\{b^e\}$ is an (n x 1) column vector with their elements given by:

$$K_{ij} = \int_{\Omega^e} N_i^e L N_j^e d\Omega \tag{2-24}$$

$$b_i^e = \int_{\Omega^e} f N_i^e d\Omega \tag{2-25}$$

It should be noted that the element matrix $[K^e]$ is symmetric since L is self-adjoint.

Substituting (2-23) into (2-19) we obtain

$$F(\phi) = \sum_{e=1}^{M} \left(\frac{1}{2} \{\phi^e\}^T [K^e] \{\phi^e\} - \{\phi^e\}^T \{b^e\} \right) \tag{2-26}$$

By performing the summation and adopting global node numbers, this can be written as

$$F = \frac{1}{2} \{\phi\}^T [K] \{\phi\} - \{\phi\}^T \{b\} \tag{2-27}$$

where [K] is an (N x N) symmetric matrix with N being the total number of unknowns or nodes, $\{\Phi\}$ is an (N x 1) unknown vector whose elements are the unknown expansion coefficient, and $\{b\}$ is an (N x 1) known vector.

The system of equations is then obtained by imposing the stationarity requirement $\delta F = 0$ or equivalently, by setting the partial derivative of F with respect to Φ_i to zero. Thus,

$$\frac{\partial F}{\partial \phi_i} = \frac{1}{2} \sum_{j=1}^{N} K_{ij} \phi_j + \frac{1}{2} \sum_{j=1}^{N} K_{ji} \phi_j - b_i = 0 \qquad i = 1,2,3,\ldots,N$$

$$\frac{\partial F}{\partial \phi_i} = \frac{1}{2}\sum_{j=1}^{N}\left(K_{ij} + K_{ji}\right)\phi_j - b_i = 0 \qquad i = 1,2,3,....,N$$

Since [K] is symmetric, Kij = Kji, thus

$$\frac{\partial F}{\partial \phi_i} = \sum_{j=1}^{N} K_{ij}\phi_j - b_i = 0 \qquad i = 1,2,3,....,N \qquad (2\text{-}28)$$

or in matrix form:

$$[K]\{\phi\} = \{b\} \qquad (2\text{-}29)$$

2.3.2 Application of Variational Technique to Electromagnetic Field Problem

Consider the problem defined previously by the electromagnetic wave equation [18]

$$\nabla \times \left(\frac{1}{\mu_r}\nabla \times E\right) - k_0^2 \varepsilon_r E = -jk_0 \eta_0 J \qquad (2\text{-}30)$$

The operator L can be written as

$$L = \nabla \times \left(\frac{1}{\mu_r}\nabla \times\right) - k_0^2 \varepsilon_r \qquad (2\text{-}31)$$

Then, by the definition of inner product we have

$$\langle LE,F\rangle = \iiint_V F^* \bullet (\nabla \times (\frac{1}{\mu_r} \nabla \times E) - k_0^2 \varepsilon_r E) dv \tag{2-32}$$

The second vector Green's Theorem is given by

$$\iiint_V [b \bullet (\nabla \times u \nabla \times a) - a \bullet (\nabla \times u \nabla \times b)] dv = \oiint_S u(a \times \nabla \times b - b \times \nabla \times a) \bullet \hat{n} ds$$

(2-33)

Applying (2-33) into (2-32) yields

$$\langle LE,F\rangle = \iiint_V E \bullet \{\nabla \times (\frac{1}{\mu_r} \nabla \times F^*) - k_0^2 \varepsilon_r F^*\} dv + \oiint_S \frac{1}{\mu_r} \{E \times (\nabla \times F^*) - F^* \times (\nabla \times E)\} \bullet \hat{n} ds \tag{2-34}$$

Since by virtue of a vector identity we have

$$(E \times (\nabla \times F^*)) \bullet \hat{n} = (\hat{n} \times E) \bullet (\nabla \times F^*) = -E \bullet (\hat{n} \times (\nabla \times F^*)) \tag{2-35}$$

Now, if both E and F satisfy the homogeneous Dirichlet boundary condition on S1

$$\hat{n} \times E = 0 \tag{2-36}$$

and the homogeneous Neumann boundary condition of the third kind (also known as Cauchy boundary condition) on S2

$$\frac{1}{\mu_r}\hat{n}\times(\nabla\times\mathbf{E})+\gamma_e\hat{n}\times(\hat{n}\times\mathbf{E})=0 \tag{2-37}$$

with S1 + S2 = S, then the surface integral in (2-34) vanishes provided that μ_r and γ_e, the known parameters of boundary S2, are real. If this is so, then

$$\langle L\mathbf{E},\mathbf{F}\rangle = \langle \mathbf{E}, L\mathbf{F}\rangle \tag{2-38}$$

and therefore L is self-adjoint

The conditions for (2-31) to be self-adjoint are:

a) ε_r, μ_r, γ_e are real numbers or functions,

b) the boundary conditions are homogeneous

Under these conditions, the functional can be constructed by substituting (2-31) into (2-17).

$$F(\mathbf{E}) = \frac{1}{2}\iiint_v \mathbf{E}^*\bullet\left[\nabla\times\frac{1}{\mu_r}(\nabla\times\mathbf{E})-k_0^2\varepsilon_r\mathbf{E}\right]dv - j\frac{k_0\eta_0}{2}\iiint_v(\mathbf{E}\bullet\mathbf{J}^*-\mathbf{E}^*\bullet\mathbf{J})dv \tag{2-39}$$

The first vector Green's Theorem is given by:

$$\iiint_V\left[u(\nabla\times\mathbf{a})\bullet(\nabla\times\mathbf{b})-\mathbf{a}\bullet(\nabla\times u\nabla\times\mathbf{b})\right]dv = \oiint_S u(\mathbf{a}\times\nabla\times\mathbf{b})\bullet\hat{n}ds \tag{2-40}$$

Applying (2-36), (2-37), and (2-40) into (2-39) we obtain

$$F(E) = \frac{1}{2}\iiint_v \frac{1}{\mu_r}(\nabla \times E^*) \bullet (\nabla \times E)dv - \frac{1}{2}\iiint_v k_0^2 \varepsilon_r E \bullet E^* dv + j\frac{k_0\eta_0}{2}\iiint_v (E \bullet J^* - E^* \bullet J)dv - \frac{1}{2}\oiint_s \frac{1}{\mu_r}(E^* \times \nabla \times E) \bullet \hat{n}ds \quad (2\text{-}41)$$

Since S = S1 + S2, we separate the surface integral. First applying (2-35) and (2-36) to S1 yields

$$-\iint_{s1} \frac{1}{\mu_r}(E^* \times \nabla \times E) \bullet \hat{n}ds = -\frac{1}{\mu_r}\iint_{s1} (\hat{n} \times E^*) \bullet (\nabla \times E)ds = 0 \quad (2\text{-}42)$$

since E* and E satisfy (2-36).

Next, applying (2-35) and (2-37) to S2 yields

$$-\iint_{s2} \frac{1}{\mu_r}(E^* \times \nabla \times E) \bullet \hat{n}ds = -\frac{1}{\mu_r}\iint_{s2} -E^* \bullet (\hat{n} \times \nabla \times E)ds = \iint_{s2} E^* \bullet (-\gamma_e(\hat{n} \times \hat{n} \times E))ds \quad (2\text{-}43)$$

Applying the vector identity $a \bullet (b \times c) = c \bullet (a \times b)$ yields

$$E^* \bullet (\hat{n} \times (\hat{n} \times E)) = (\hat{n} \times E) \bullet (E^* \times \hat{n}) = -(\hat{n} \times E) \bullet (\hat{n} \times E^*) \quad (2\text{-}44)$$

We can then modify (2-43) using (2-44) to get

$$-\iint_{s2}\frac{1}{\mu_r}(E^* \times \nabla \times E)\bullet \hat{n}ds = \iint_{s2}\gamma_e(\hat{n}\times E)\bullet(\hat{n}\times E^*)ds \qquad (2\text{-}45)$$

Therefore, the functional can be written as

$$F(E) = \frac{1}{2}\iiint_v \frac{1}{\mu_r}(\nabla \times E^*)\bullet(\nabla \times E)dv - \frac{1}{2}\iiint_v k_0^2\varepsilon_r E\bullet E^* dv + j\frac{k_0\eta_0}{2}\iiint_v (E\bullet J^* - E^*\bullet J)dv + \frac{1}{2}\iint_{s2}\gamma_e(\hat{n}\times E)\bullet(\hat{n}\times E^*)ds \qquad (2\text{-}46)$$

2.4 Recent Computationally Suitable Vector Finite Element Method

In recent years, a lot of research has been carried out in applying the Finite Element Method to a myriad of electromagnetic problems [20] – [32]. The Vector Finite Element Method (VFEM), as indicated previously, is the latest development of Finite Element Method and is eminently suitable to solve many types of these problems.

One of most versatile of the VFEMs is the one developed by Koshiba and Inoue [14]. Briefly, it can be described as follows.

The source-free vectorial wave equation propagating in the z-direction has the form:

$$\nabla \times ([p]\nabla \times \Phi) - k_0^2[q]\Phi = 0 \qquad (2\text{-}47)$$

where Φ denotes either **E** or **H**, and [p] and [q] are given by:

$$[p] = \begin{bmatrix} p_x & 0 & 0 \\ 0 & p_y & 0 \\ 0 & 0 & p_z \end{bmatrix} \quad \text{and} \quad [q] = \begin{bmatrix} q_x & 0 & 0 \\ 0 & q_y & 0 \\ 0 & 0 & q_z \end{bmatrix}$$

The components of [p] and [q] are:

$$\Phi = E$$
$$p_x = p_y = p_z = 1$$
$$q_x = \varepsilon_{rx} = n_x^2$$
$$q_y = \varepsilon_{ry} = n_y^2$$
$$q_z = \varepsilon_{rz} = n_z^2$$

$$\Phi = H$$
$$p_x = \frac{1}{\varepsilon_{rx}} = \frac{1}{n_x^2}$$
$$p_y = \frac{1}{\varepsilon_{ry}} = \frac{1}{n_y^2}$$
$$p_z = \frac{1}{\varepsilon_{rz}} = \frac{1}{n_z^2}$$
$$q_x = q_y = q_z = 1$$

where ε's are the relative permittivities and n's are the refractive indices in the coordinate axes directions. The functional for the wave equation (2-47) is given by the modified equation (2-46) (source free region and applied boundary condition):

$$F(\Phi) = \frac{1}{2} \iint_{\Omega} [(\nabla \times \Phi)^* \bullet ([p]\nabla \times \Phi) - k_0^2 [q]\Phi^* \bullet \Phi] dxdy \qquad (2\text{-}48)$$

where Ω is the cross-section over which the VFEM is applied.

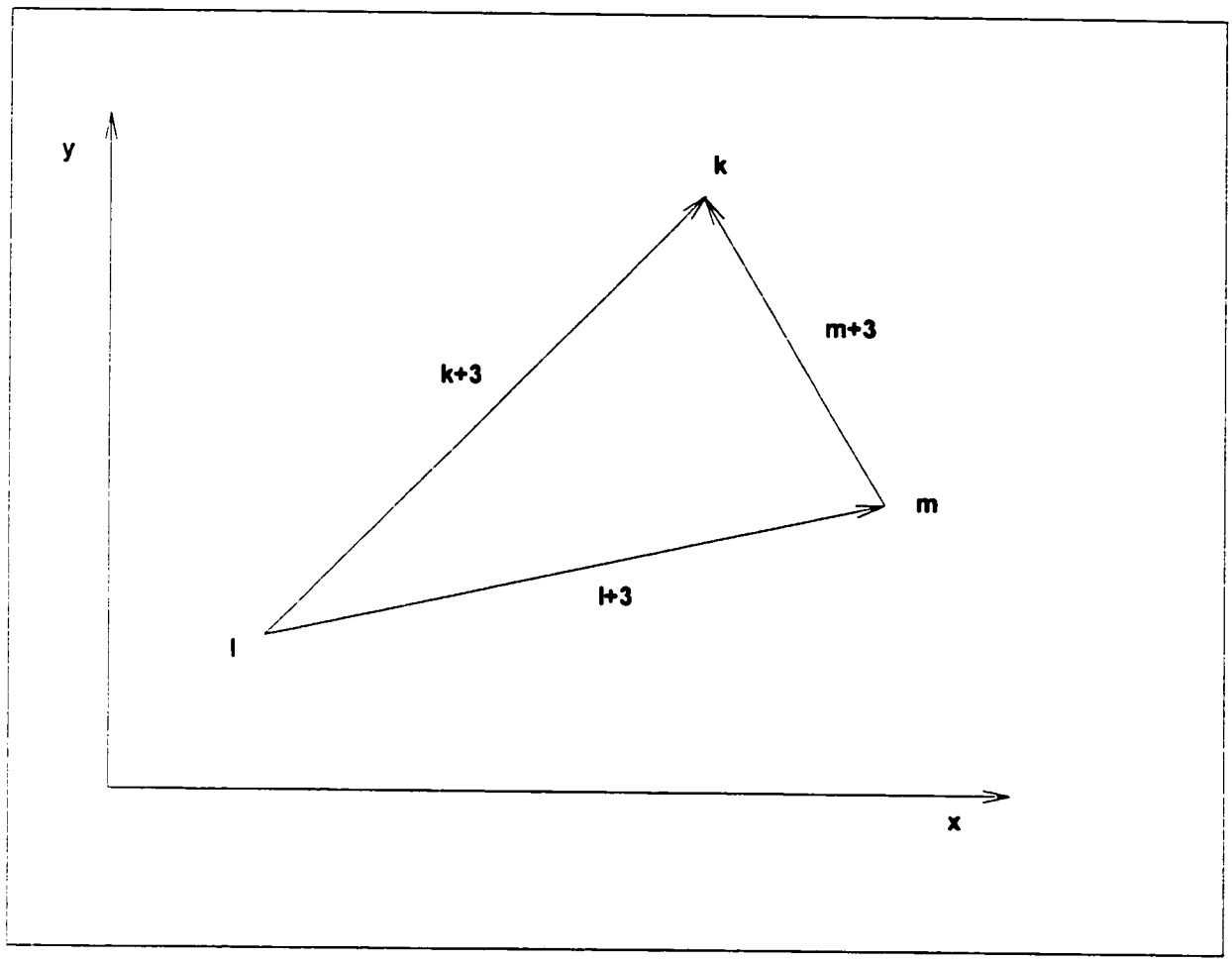


Figure 2.1 – The Vector Finite Element

The vertex points k, l, m are where the longitudinal (z direction) components of Φ are located, while the transverse (x and y direction) components of Φ are located along the edges. The mid-points of the edges are identified by k+3, l+3, m+3.

The longitudinal component of Φ is given by:

$$\Phi_z = j\{N(x, y)\}^T \{\Phi_z\}_e = j\{N\}^T \{\Phi_z\}_e \qquad (2\text{-}49)$$

where e denotes the finite element, and N denotes the interpolation function, which is given by:

$$\{N\} = \frac{1}{\det}\begin{bmatrix} a_k' & b_k' & c_k' \\ a_l' & b_l' & c_l' \\ a_m' & b_m' & c_m' \end{bmatrix}\begin{bmatrix} 1 \\ x \\ y \end{bmatrix} = \begin{bmatrix} \alpha_k \\ \alpha_l \\ \alpha_m \end{bmatrix} \tag{2-50}$$

a' ... c' are obtainable through modulo 3 operation as shown below:

$$\begin{aligned} a_k' &= x_l y_m - x_m y_l & a_l' &= x_m y_k - x_k y_m & a_m' &= x_k y_l - x_l y_k \\ b_k' &= y_l - y_m & b_l' &= y_m - y_k & b_m' &= y_k - y_l \\ c_k' &= x_m - x_l & c_l' &= x_k - x_m & c_m' &= x_l - x_k \\ \det &= \left[a_k' + a_l' + a_m'\right] \end{aligned}$$

Thus,

$$\Phi_z = j\begin{Bmatrix} \alpha_k & \alpha_l & \alpha_m \end{Bmatrix}\begin{Bmatrix} \Phi_{zk} \\ \Phi_{zl} \\ \Phi_{zm} \end{Bmatrix} = j\sum_{i=k}^{m} \Phi_{zi}\alpha_i(x,y) \tag{2-51}$$

where Φ_{zi} is either E_{zi} or H_{zi} on element e.

The transverse component Φ_t lies in the x-y plane, and is given by the vectorial sum of its x and y components,

$$\vec{\Phi}_t = \Phi_x \hat{x} + \Phi_y \hat{y}$$

The transverse component along the edge of the element, namely the edge tangential field is given by:

$$\vec{\Phi}_t\big|_{edge} = \Phi_x\big|_{edge} \cos\theta + \Phi_y\big|_{edge} \sin\theta$$

where θ is the angle between the edge and the positive x - axis.

The transverse components Φ_x and Φ_y within the triangular element e are approximated by a linear function of y and x respectively. In the normal convention for a triangular finite element, x and y are the centroid coordinates.

$$\Phi_x = a + cy \qquad\qquad \Phi_y = b - cx$$

After extensive derivations, we get

$$\Phi_x(y) = \begin{bmatrix} a_k + c_k y \\ a_l + c_l y \\ a_m + c_m y \end{bmatrix}^T \begin{bmatrix} \Phi_{tk+3} \\ \Phi_{tl+3} \\ \Phi_{tm+3} \end{bmatrix} = \{U\}^T \{\Phi_t\}_e$$

$$\Phi_y(x) = \begin{bmatrix} b_k - c_k x \\ b_l - c_l x \\ b_m - c_m x \end{bmatrix}^T \begin{bmatrix} \Phi_{tk+3} \\ \Phi_{tl+3} \\ \Phi_{tm+3} \end{bmatrix} = \{V\}^T \{\Phi_t\}_e$$

The matrices U and V are the shape function vectors for the triangular edge element and the matrix Φ_t represents the edge variables in the transversal plane for each element in the VFEM zone. The field within the finite element is fully described by:

$$\Phi_e = [N^\circ]^T \{\Phi\}_e \tag{2-51}$$

where

$$[N^\circ] = \begin{bmatrix} \{U\} & \{V\} & \{0\} \\ \{0\} & \{0\} & j\{N\} \end{bmatrix} \qquad \{\Phi\}_e = \begin{Bmatrix} \{\Phi_t\}_e \\ \{\Phi_z\}_e \end{Bmatrix}$$

Over the VFEM zone, the functional (equation (2-48)) can be written as:

$$F(\Phi) = \sum_{e=1}^{\#TR} \frac{1}{2} \iint_{\Omega_e} [(\nabla \times [N^\circ]^T \{\Phi\}_e)^{*T} \bullet ([p]\nabla \times [N^\circ]^T \{\Phi\}_e) - k_0^2([q]([N^\circ]^T \{\Phi\}_e)^{*T} \bullet ([N^\circ]^T \{\Phi\}_e))] \, dxdy \tag{2-52}$$

where #TR is the total number of triangular finite elements in the VFEM zone, and e is the element number. Next, the functional is minimized with respect to $\{\Phi\}_e$. It should be noted that

while $\{\Phi\}_e$ represents the nodal (longitudinal) and edge (transverse) values for each finite element, $\{\Phi\}$ represents the same for the entire VFEM zone. After extensive manipulation, the result is an eigenvalue equation:

$$[K]\{\Phi\} - k_0^2[M]\{\Phi\} = \{0\} \tag{2-53}$$

where [K] and [M] are given by:

$$[K] = \begin{bmatrix} [K_{tt}] & [K_{tz}] \\ [K_{zt}] & [K_{zz}] \end{bmatrix} \qquad [M] = \begin{bmatrix} [M_{tt}] & [0] \\ [0] & [M_{zz}] \end{bmatrix}$$

In terms of transverse and longitudinal components, we can write fully as:

$$[K_{tt}]\{\Phi_t\} + [K_{tz}]\{\Phi_z\} - k_0^2[M_{tt}]\{\Phi_t\} = \{0\} \tag{2-54}$$

$$[K_{zt}]\{\Phi_t\} + [K_{zz}]\{\Phi_z\} - k_0^2[M_{zz}]\{\Phi_z\} = \{0\} \tag{2-55}$$

Further simplifications yield the following eigenvalue equations:

$$[K_{tt}]\{\Phi_t\} - \beta[K_{tz}]\{\Phi_z\} - \beta^2[M_{tt}]\{\Phi_t\} = \{0\} \tag{2-56}$$

$$-\beta[K_{zt}]\{\Phi_t\} + [K_{zz}]\{\Phi_z\} = \{0\} \tag{2-57}$$

In the case of deterministic problems, we obtain the appropriate expression by putting $\beta = 0$.

Thus,

$$[K_{tt}]\{\Phi_t\} = \{0\} \tag{2-58}$$

$$[K_{zz}]\{\Phi_z\} = \{0\} \tag{2-59}$$

The above expressions were utilized in the solution of various electromagnetic problems, described in this thesis.

CHAPTER 3

RELATIONSHIP OF VFEM ZONE DISCRETIZATION AND SOLUTION MATRIX STRUCTURE

3.1 Introduction

In investigating the real-time numerical solution of electromagnetic problems, substantial effort was made to determine the key relationships between the VFEM zone discretization and the structure of its related solution matrix. The intent behind this investigation and research was to identify attributes, aspects, and causality between these two elements of the VFEM methodology outlined in Chapter 2. A successful outcome would, in turn, allow the formulation of techniques to improve rapid solution of such problems. Knowing the relationship between the VFEM zone geometry and the solution matrix would lead to more efficient coding and better utilisation of computational resources. In this Chapter, the results of this investigation are summarized.

As far as it can be determined, such an investigation was not carried out elsewhere.

3.2 VFEM Zone Discretization and Matrix Equation Relationship

In a rectilinear VFEM discretized zone, the total number of finite element triangles and edges is obtained by the following expressions:

$$\text{Total Triangles} = 2 \bullet X\_\text{divisions} \bullet Y\_\text{divisions}$$

$$\text{Total Edges} = X\_\text{divisions} + (Y\_\text{divisions} \bullet ((3 \bullet X\_\text{divisions}) + 1))$$

where X_divisions and Y_divisions are the number of divisions along the x-coordinate and y-coordinate axes of the VFEM zone. The VFEM zone is enclosed by Dirichlet boundaries. Depending on the problem type, Dirichlet boundaries may also be present within the VFEM zone. The sum of the Dirichlet Edges (i.e. edges that lie along Dirichlet boundaries) and the edges whose values are unknown is equal to the total edges within the VFEM zone.

Using the VFEM methodology of Chapter 2, in the process of the forming of the solution matrix, only the unknown edges are taken into account, as the values of Dirichlet Edges are already known. For the entire discretized zone, a fixed number S of non-zero values are generated by the process, each identified by row and column indices, prior to the formation of the matrices. This fixed number of non-zero values is always equal to:

$$S = 3! \bullet \text{unknown edges} = 6 \bullet \text{unknown edges}$$

for a zone discretized into triangular finite elements such as shown in Chapter 2, Figure 2.1. These

non-zero values will be used in forming the matrix equation, namely the coefficient (left hand) matrix, and the excitement (right hand) matrix. The number of non-zero values that will go into forming the right hand matrix is labelled R_count (R_count $\in$ S), while the number of non-zero values that will go into forming the left hand matrix is labelled L_count (L_count. $\in$ S). Also, all finite element triangles in the VFEM zone that have one or more edges that coincide with Dirichlet boundaries are named Dirichlet Triangles (DTs).

In the following sub-sections, some of the key factors that enable *a priori* determination of memory storage requirements, computer resource needs, and solution efficiency are presented.

3.2.1 Determination of R_Count

R_count can be determined two ways:

Method 1:

$$R\_count = \sum \text{Dirichlet Edges (n)} \bullet \text{non-Dirichlet Edges(n)}$$

where n represents the number of the Dirichlet Triangle ($1 \leq n \leq M$)

or alternately by a much simpler way,

Method 2:

$$R\_count = 2! \bullet \text{Total Number of Dirichlet Triangles}$$

Thus, R_count can be obtained either by summing the products of the Dirichlet Edges (DEs) and non-Dirichlet Edges (NDEs) of the Dirichlet Triangles in the FEM zone, or by taking twice the total number of the Dirichlet Triangles in the FEM zone.

Example: We have a 4 x 3 VFEM zone. The total number of finite element triangles is 24. The Dirichlet Triangles are the triangles numbered 0, 3, 4, 7, 8, 15, 16, 17, 18, 21, 22, and 23 for a total of 12 triangles. Triangles 0 and 3 have two edges that form Dirichlet boundaries, while the remaining 10 have only one edge that forms a Dirichlet boundary.

Using Method 1:

$$R\_count = \Sigma\ (DE(0) \bullet NDE(0)) + (DE(3) \bullet NDE(3)) + \ldots + (DE(23) \bullet NDE(23))$$

$$R\_count = \Sigma\ (2 \times 1) + (2 \times 1) + (1 \times 2) + (1 \times 2) + \ldots + (1 \times 2) = 24$$

Using Method 2:

$$R\_count = (2 \times 12) = 24$$

3.2.2 Determination of L_Count

In the case of L_count, this is obtained in a much simpler way from:

$$L\_count = S - R\_count = 6 \bullet \text{Unknown Edges} - R\_count$$

3.2.3 Determination of Right Matrix Non-Zeros

For the right hand matrix, the total number of non-zero elements (R_matrix Non-Zeros) is obtained by either of three methods.

Method 1:

R_matrix Non-Zeros = Include all Dirichlet Triangles of the FEM zone and sum the diagonal edges of each Dirichlet Triangle plus non-diagonal shared edges between the Dirichlet Triangles.

Method 2:

R_matrix Non-Zeros = R_count – {sum of all Dirichlet Triangle shared edges (diagonal and non-diagonal) plus the total number of Dirichlet Triangles having more than one Dirichlet Edge}

Method 3: (better option)

R_matrix Non-Zeros = Include all Dirichlet Triangles of the FEM zone and sum all of their non-Dirichlet Edges.

3.2.4 Determination of Left Matrix Non-Zeros

Once L_count is determined, then the total number of non-zero elements of the left hand matrix (L_matrix Non-Zeros) is readily given by:

L_matrix Non-Zeros = L_count – Unknown Edges

This is expected and is directly related to the reason as to why the matrix is diagonally dominant (explained in Chapter 4, Section 4.2).

It should be noted that the theoretical sizes for the left and right hand matrices are (Unknown Edges)^2 and (Unknown Edges) respectively. Also, R_count is not necessarily equal to Unknown Edges.

Obviously,

R_count + L_Matrix Non-Zeros = 5 • Unknown Edges

Finally, the simplified expression to predict the L_matrix Non-Zeros is:

L_matrix Non-Zeros = 5 • Unknown Edges – 2 • Dirichlet Triangles

3.3 Relationship of Edge Numbering and Fill-In Numbers

One more point that should be made is that even if one obtains a smaller number of fill-ins by manipulating the edge numbering of the VFEM zone, at the end, due to the larger matrix bandwidth, the memory storage requirements will be much larger, as shown below. On the other hand, using reverse Cuthill-McKee algorithm to reduce the bandwidth may lessen the memory storage requirement, which will enable solutions of problems with larger number of unknowns, but at the expense of additional processing time, due to the extensive row and column swapping.

Therefore, a matter of trade-off is involved.

NOTE: All Figures are located at the end of this Chapter.

Figure 3.1 shows a VFEM zone discretized into 32 triangular finite elements, whose edges are numbered in accordance with a sequential numbering system developed for this work, and which is applied in all the VFEM problems solved throughout. Figure 3.2 shows the remapped discretized zone non-boundary (unknown) edges (more on this in the next Chapter), and Figure 3.3 shows the coefficients matrix template prior to decomposition, including the future fill-ins. The matrix contains 176 non-zero elements, and 218 fill-ins that will occur in the matrix during its decomposition, therefore 425 memory storage spaces (unidimensional array) will be required, excluding index vectors.

Figures 3.4 and 3.5 are the counterparts of Figures 3.1 and 3.2 respectively, where a different edge numbering system was utilized. The resulting coefficients matrix template is shown in Figure 3.6. This matrix contains 176 non-zero elements as before, but only 192 fill-ins will occur during the decomposition. However, the bandwidth of this matrix is larger, thus necessitating a memory storage of 541 spaces, excluding index vectors.

It is possible to obtain an optimized edge numbering system that would minimize the fill-ins, but given the degree of sparsity of the VFEM coefficients matrix, the benefits would be moot with respect to the efforts involved. The difference in fill-in numbers due to an optimized edge numbering system and any other orderly edge numbering system would not affect computational performance appreciably. Rather, the memory storage requirements will be the overriding concern, because

these are the critical resources that eventually place an upper bound on the size of the VFEM zone of an electromagnetic problem.

3.4 Fill-In Laws

In the investigation into the VFEM zone discretization and solution matrix structure relationship, the following matrix fill-in characteristics were discovered which may be stated as laws controlling fill-ins. Left hand matrix is defined as containing coefficients of non-Dirichlet Edges. Also, adjacent edges are defined as edges where the angle subtended is less than 180 degrees. The edge numbers refer to the remapped discretization zone edges.

Law 1. *An edge will produce a non-zero value (coefficient, matrix element etc) at its own matrix location, namely on the matrix diagonal where row number is equal to column number.*

Law 2. *Any two adjacent non-Dirichlet Edges will produce a non-zero matrix elements at their respective [row, column] and [column, row] locations.*

Example: Figures 3.2 and 3.3; edges 15 and 20 will create non-zero values at matrix locations [15,20] and [20,15].

Law 3. *When two edges are adjacent but do not have consecutive numbers, as in 15 and 20, fill-ins will be generated during matrix decomposition, at the matrix locations of*

the in-between non-Dirichlet Edges.

Example: Figures 3.2 and 3.3; adjacent edges 15 and 20. Fill-ins will be generated at the following matrix locations: [20,16], [20,17], [20,18], [20,19], [16,20], [17,20], [18,20], [19,20]. Also, there will be non-zero values at locations [15,20] and [20,15] as per Law 2. above.

Law 4. *If an edge with a higher number is adjacent to two edges with consecutive numbers, then the larger of the two lower numbers is used in determining the fill-in locations.*

Example: Figures 3.2 and 3.3; edge 20 is adjacent to edges 14 and 15. Fill-ins will be generated exactly as in the example for Law 3. above. Also, there will be non-zero values at locations [14,20], [15,20], [20,15], and [20,14] as per Law 2. above.

Law 5. *The converse of Law 4. is not true. If two edges with consecutive higher numbers are adjacent to an edge with a lower number, the fill-ins will be generated for each higher edge number with respect to the lower edge number in turn.*

Example: Figures 3.2 and 3.3; edges 15 and 16 are adjacent to edge 9. The fill-ins will be generated at the following matrix locations: [16,11], [16,12], [16,13], [16,14], [11,16], [12,16], [13,16], [14,16], [15,10], [15,11], [15,12], [15,13], [10,15], [11,15], [12,15], [13,15], and if edges 15 and 16 are non-adjacent, at the locations [16,15] and [15,16]. Also, there will be non-zero values at locations [9,15], [9,16], [16,9], and [15,9] as per Law 2. above.

3.5 Graphical Techniques for Identifying Edges Contributing to Fill-Ins

In a VFEM discretized zone, to be able to determine the total number of edges producing fill-ins during matrix decomposition is very advantageous for several reasons, and this quality was discovered in the course of this work by detailed analysis. This graphical technique can be described as follows (Note: Edge numbering convention as shown in Figure 3.2):

Starting from the highest numbered non-Dirichlet Edge, count the edges that are adjacent to lower tier edges. At the matrix rows and columns corresponding to these edge numbers, the fill-ins will be present.

Figure 3.7 at the end of this Chapter illustrates this graphical technique.

Figure 3.7 a) : Non-boundary edges, in three tiers:

Tier 1: 3, 4, 5; Tier 2 : 7, 8; Tier 3: 10, 11, 12;

Adjacent edges are: 12 and 8; 10 and 7; 8 and 5; 7 and 3.

Therefore, there are four edges in total contributing to fill-ins.

Figure 3.7 b) : Non-boundary edges, in three tiers:

Tier 1: 4, 5, 6, 7, 8; Tier 2 : 10, 11, 12; Tier 3: 14, 15, 16, 17, 18;

Adjacent edges are: 18 and 12; 16 and 11; 15 and 10; 14 and 10; 12 and 8;

11 and 6; 10 and 5.

Therefore, there are seven edges in total contributing to fill-ins.

Figure 3.7 c) : Non-boundary edges, in five tiers:

Tier 1: 5, 6, 7, 8, 9, 10, 11; Tier 2 : 13, 14, 15, 16;

Tier 3: 18, 19, 20, 21, 22, 23, 24; Tier 4: 26, 27, 28, 29;

Tier 5: 31, 32, 33, 34, 35, 36, 37;

Adjacent edges are: 37 and 29; 35 and 28; 34 and 27; 33 and 27; 31 and 26; 29 and 24; 28 and 22; 27 and 20; 26 and 18; 24 and 16; 23 and 15; 22 and 15; 20 and 14; 19 and 13; 18 and 13; 16 and 11; 15 and 9; 14 and 7; 13 and 5.

Therefore, there are 19 edges in total contributing to fill-ins.

3.6 Non-Graphical Techniques for Predicting the Number of Edges Generating Fill-Ins

The capability to predict the total number of edges contributing to the generation of fill-ins prior to the solution matrix formation and decomposition is considered very important to the computation process. Accordingly, an additional non-graphical methodology was developed here to predict the number of edges within the VFEM zone that would contribute to fill-ins within the solution matrix. The procedure is directly related to the attributes of the VFEM zone. The methodology was tested on various discretizations and it was proven successful in all cases. While this outcome does not confer the mantle of absoluteness on this methodology, – for such a qualification requires rigorous proof – it nevertheless denotes high confidence in its quality. This non-graphical methodology was tested against the graphical technique outlined above.

Note: X_div and Y_div are integers and refer to the number of divisions along the x and y coordinate

directions respectively that are utilized in the discretization of the VFEM zone.

3.6.1 Case 1: VFEM Zones with No Internal Dirichlet Boundary Edges

a) when the following logical condition is met

$$\{(X\_div + Y\_div = EVEN) \text{ AND } (X\_div \bullet Y\_div = EVEN)\} = TRUE$$

then F, the total number of edges contributing to fill-ins in the VFEM zone is given by

$$F = \{2 \bullet X\_div \bullet (Y\_div - 1)\} + \{[(X\_div - 1) \bullet (Y\_div - 1) + 1] / 2\}$$

b) when the following logical conditions are met

$$\{(X\_div + Y\_div = ODD) \text{ AND } (X\_div \bullet Y\_div = EVEN)\} = TRUE$$

OR

$$\{(X\_div + Y\_div = EVEN) \text{ AND } (X\_div \bullet Y\_div = ODD)\} = TRUE$$

then F, the total number of edges contributing to fill-ins in the VFEM zone is given by

$$F = \{2 \bullet X\_div \bullet (Y\_div - 1)\} + \{[(X\_div - 1) \bullet (Y\_div - 1)] / 2\}$$

3.6.2 Verification/Validation of Case 1

Figure 3.8 a)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 5 in this case. Applying the applicable formula:

$X\_div = 2$; $Y\_div = 2$; $X\_div + Y\_div = 4$; $X\_div \bullet Y\_div = 4$.

$F = (2 \bullet 2 \bullet (2-1)) + ((2-1) \bullet (2-1) + 1)/2 = 5$.

Figure 3.8 b)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 7 in this case. Applying the applicable formula:

$X\_div = 3$; $Y\_div = 2$; $X\_div + Y\_div = 5$; $X\_div \bullet Y\_div = 6$.

$F = (2 \bullet 3 \bullet (2-1)) + ((3-1) \bullet (2-1))/2 = 7$.

Figure 3.8 c)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 19 in this case. Applying the applicable formula:

$X\_div = 4$; $Y\_div = 3$; $X\_div + Y\_div = 7$; $X\_div \bullet Y\_div = 12$.

$F = (2 \bullet 4 \bullet (3-1)) + ((4-1) \bullet (3-1))/2 = 19$.

Figure 3.8 d)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 14 in this case. Applying the applicable formula:

$X\_div = 3$; $Y\_div = 3$; $X\_div + Y\_div = 6$; $X\_div \bullet Y\_div = 9$.

$F = (2 \bullet 3 \bullet (3-1)) + ((3-1) \bullet (3-1))/2 = 14$.

Figure 3.8 e)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 29 in this case. Applying the applicable formula:

$X\_div = 4$; $Y\_div = 4$; $X\_div + Y\_div = 8$; $X\_div \bullet Y\_div = 16$.

$F = (2 \bullet 4 \bullet (4-1)) + ((4-1) \bullet (4-1) + 1)/2 = 29$.

Figure 3.8 f)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 36 in this case. Applying the applicable formula:

$X\_div = 5$; $Y\_div = 4$; $X\_div + Y\_div = 9$; $X\_div \bullet Y\_div = 20$.

$F = (2 \bullet 5 \bullet (4-1)) + ((5-1) \bullet (4-1))/2 = 36$.

Figure 3.8 g)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 38 in this case. Applying the applicable formula:

$X\_div = 4$; $Y\_div = 5$; $X\_div + Y\_div = 9$; $X\_div \bullet Y\_div = 20$.

$F = (2 \bullet 4 \bullet (5-1)) + ((4-1) \bullet (5-1))/2 = 38$.

Figure 3.8 h)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins,

which is 48 in this case. Applying the applicable formula:

$$X\_div = 5 ; \quad Y\_div = 5 ; \quad X\_div + Y\_div = 10 ; \quad X\_div \bullet Y\_div = 25 .$$

$$F = (2 \bullet 5 \bullet (5-1)) + ((5-1) \bullet (5-1))/2 = 48 .$$

Figure 3.8 i)

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 44 in this case. Applying the applicable formula:

$$X\_div = 6 ; \quad Y\_div = 4 ; \quad X\_div + Y\_div = 10 ; \quad X\_div \bullet Y\_div = 24 .$$

$$F = (2 \bullet 6 \bullet (4-1)) + ((6-1) \bullet (4-1) + 1)/2 = 44 .$$

As can be seen, this prediction methodology for obtaining the number of edges contributing to the fill-ins shows excellent agreement with its graphical counterpart.

3.6.3 Case 2: VFEM Zones Containing Internal Dirichlet Boundary Edges

When solving stripline, microstrip etc type of problems, the discretized VFEM zone will contain Dirichlet Edges within the zone. The prediction algorithm to obtain the total number of edges contributing to fill-ins in the solution matrix is modified with an additional factor N.

Let P be the total number of internal Dirichlet Edges along the x coordinate axis and let Q be the number of lines along the x coordinate on which the internal Dirichlet Edges are located.

Then, depending on the values of P and Q, N is given by:

Q = ODD ; P = EVEN ; $N = 2 \cdot P + (P/2)$

Q = ODD ; P = ODD ; $N = 2 \cdot P + ((P-1)/2)$

Q = EVEN ; P = EVEN ; $N = 2 \cdot P + (P/2) - 1$

Q = EVEN ; P = ODD ; $N = 2 \cdot P + ((P-1)/2)$

Then, let H be the total number of edges contributing to fill-ins within the solution matrix. H is obtained by:

$H = F - N$ where F is obtained as shown above in Section 3.6.1.

3.6.4 Verification/Validation of Case 2

Figure 3.9 a) Stripline

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 102 in this case.

X_div = 8 ; Y_div = 8 ; P = 14 ; Q = 3 ;

X_div + Y_div = 16 ; X_div • Y_div = 64 ;

$F = (2 \cdot 8 \cdot (8-1)) + ((8-1) \cdot (8-1) + 1)/2 = 137$

$N = 2 \cdot 14 + (14/2) = 35$

$H = F - N = 137 - 35 = 102$.

Figure 3.9 b) Dialectric

Applying the Fill-In Laws, we obtain from the graph, the number of edges contributing to fill-ins, which is 64 in this case.

X_div = 8 ; Y_div = 5 ; P = 6 ; Q = 2 ;

X_div + Y_div = 13 ; X_div • Y_div = 40 ;

F = (2 • 8 • (5-1)) + ((8-1) • (5-1))/2 = 78

N = 2 • 6 + (6/2) - 1 = 14

H = F - N = 78 - 14 = 64 .

Other VFEM zone combinations were tried and the algorithm matched the results from the graphical technique.

3.7 Advantages of Fill-In Laws and Fill-In Edge Determination Techniques

The advantages of utilizing the fill-in laws and the fill-in edge determination techniques to identify the edges contributing to fill-ins has very beneficial outcomes, such as:

i) the capability to determine the overall number of fill-ins and the associated memory storage requirements, because for each edge contributing to fill-ins, the actual number of fill-ins associated with that edge can be obtained;

ii) the contribution to accelerated matrix solution by identifying the entries of the look-ahead decomposition table, described in Chapter 4.

3.8 Contributions of the Identified Relationships to Computational Efficiency

In this Chapter, several key relationships between the VFEM zone discretization and the solution matrix were identified and formulated into a paradigm which is easy to understand and to apply. While this paradigm looks deceptively simple, when used properly, its potential usefulness as a powerful computational tool is undeniable. In this work, this paradigm was utilized successfully.

In summary, the work outlined in this Chapter realized the following contributions:

a) the VFEM edge numbering methodology has great impact on the structure of the solution matrix (including initial non-zero elements plus fill-ins);

b) the ability to compute the initial non-zero value populations of the solution and excitation matrices, plus to be able to determine the actual number of fill-ins to enable the identification of required computational resources;

c) by means of a combination of fill-in laws and graphical / non-graphical techniques, readily identify those edges contributing to fill-ins, and prepare the formation of the decomposition table entries;

d) the implementation of a suitable matrix solution technique for efficient problem processing.

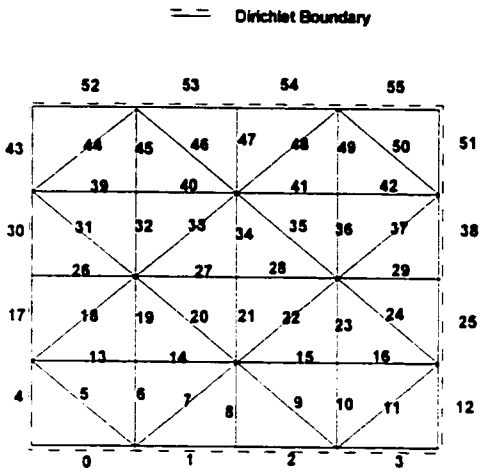


Figure 3.1 VFEM Zone Edge Numbering

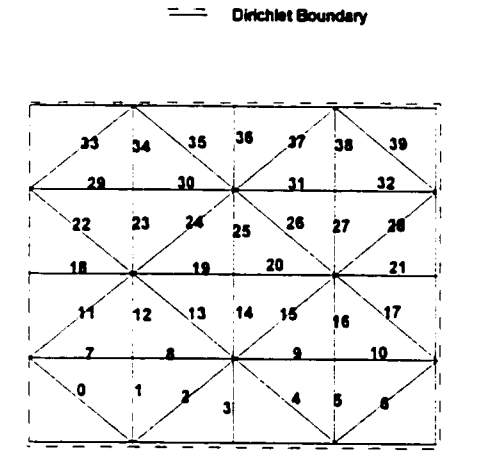
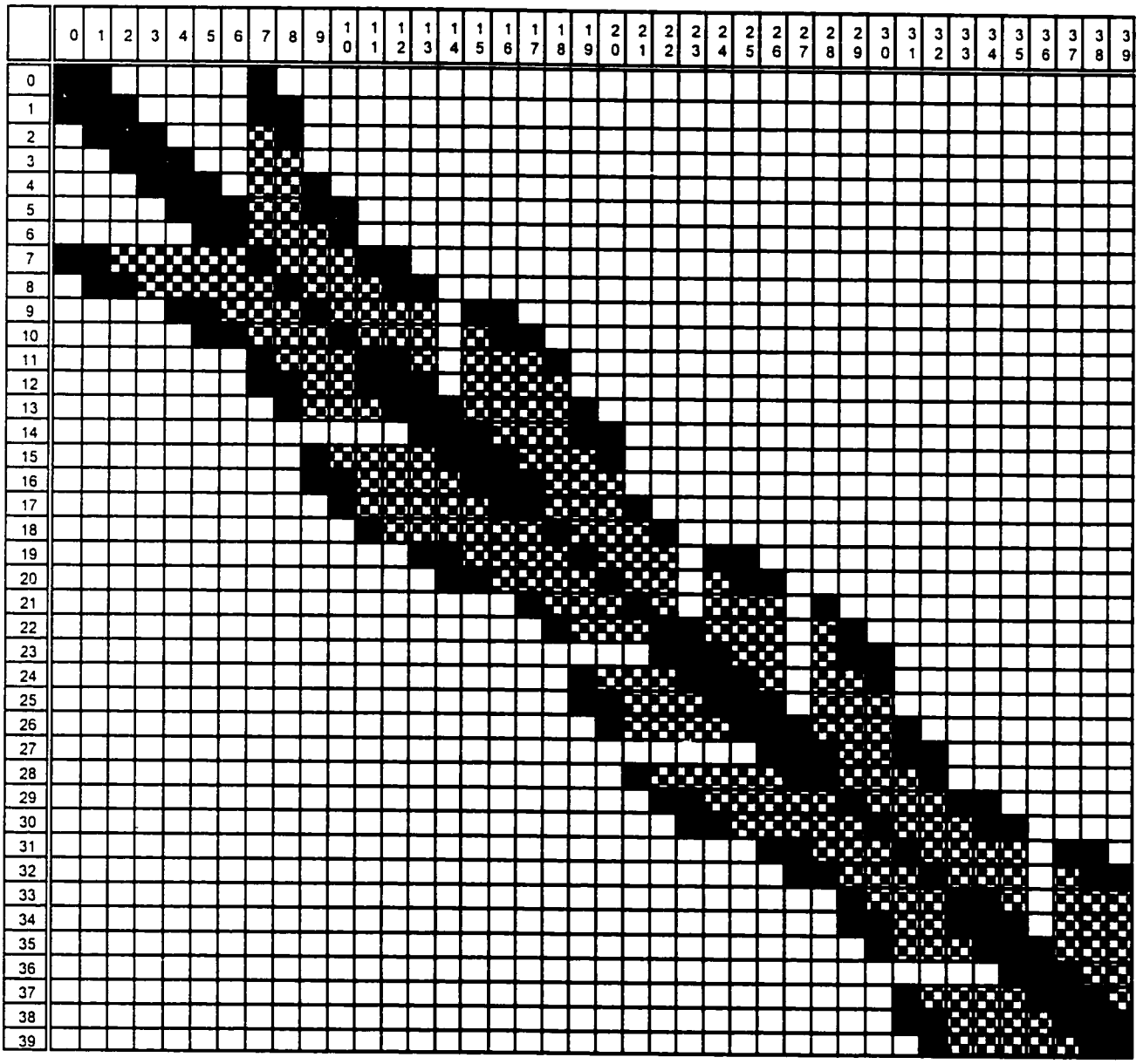


Figure 3.2 Remapping of Unknown (Non-Boundary) Edges



Solid Square – Non-Zeros(176) Checkered Square – Fill-Ins(218)

Figure 3.3 – Matrix Non-Zero Template For Figure 3.2

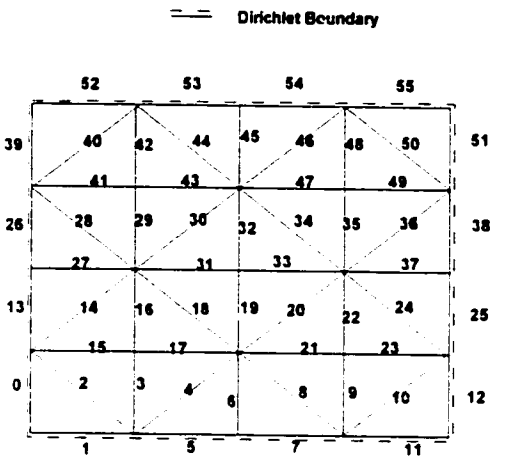


Figure 3.4 VFEM Zone Edge Numbering (Alternate Way)

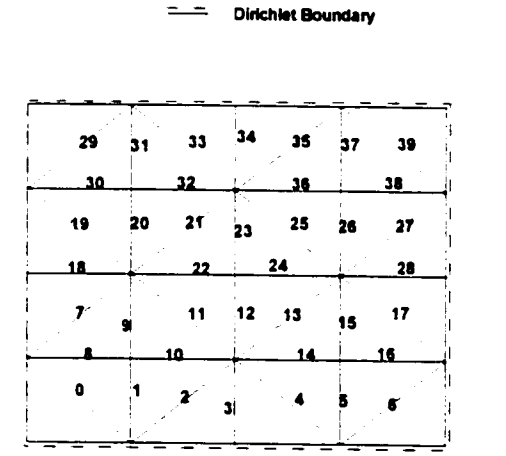
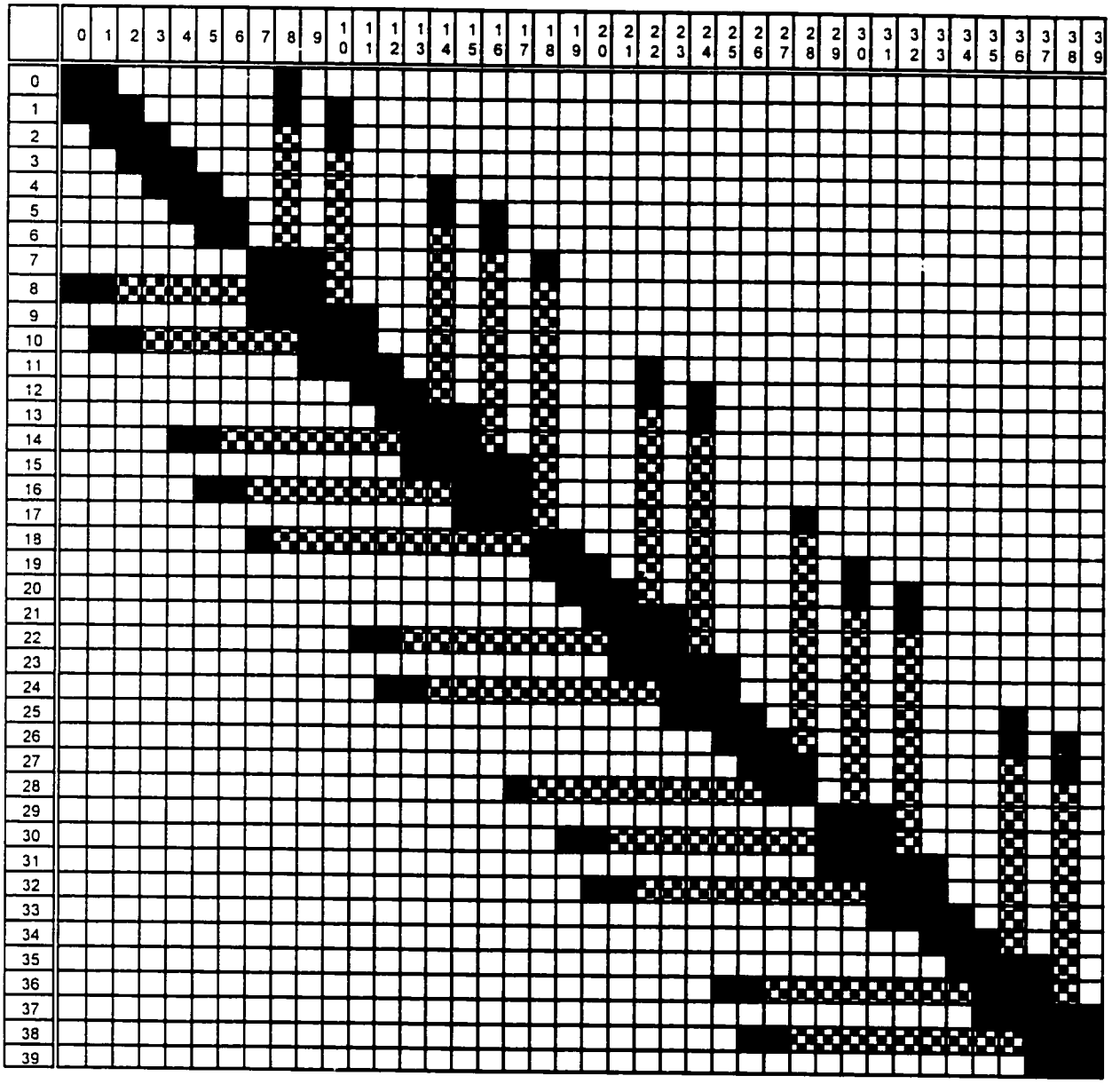


Figure 3.5 Remapping of Unknown (Non-Boundary) Edges (Alternate Way)



Solid Squares – Non-Zeros (176) Checkered Squares – Fill-Ins (192)

Figure 3.6 – Matrix Non-Zero Template For Figure 3.5

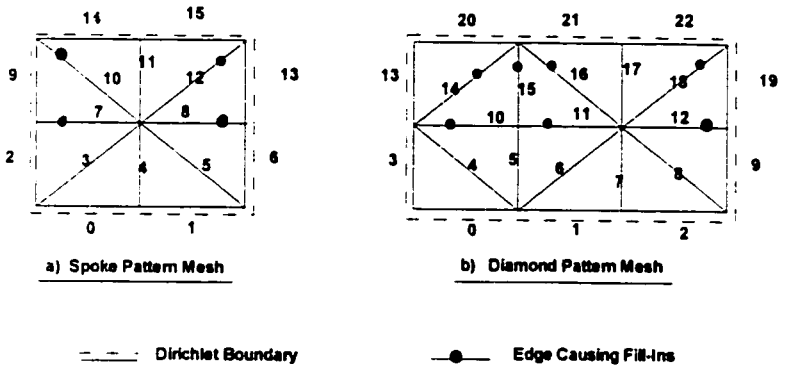


Figure 3.7 a) & b) -- Edges Contributing To Fill-Ins

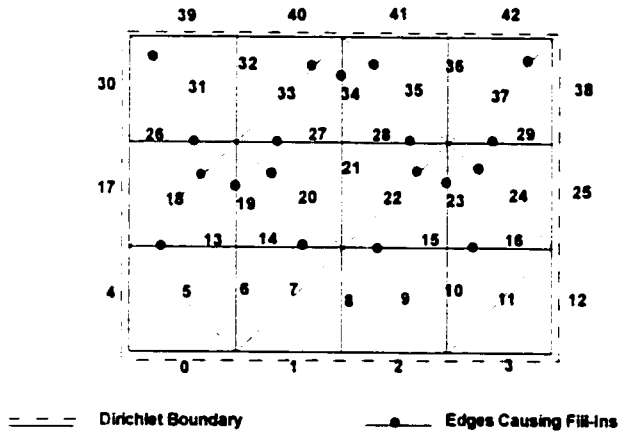


Figure 3.7 c) -- Edges Contributing To Fill-Ins

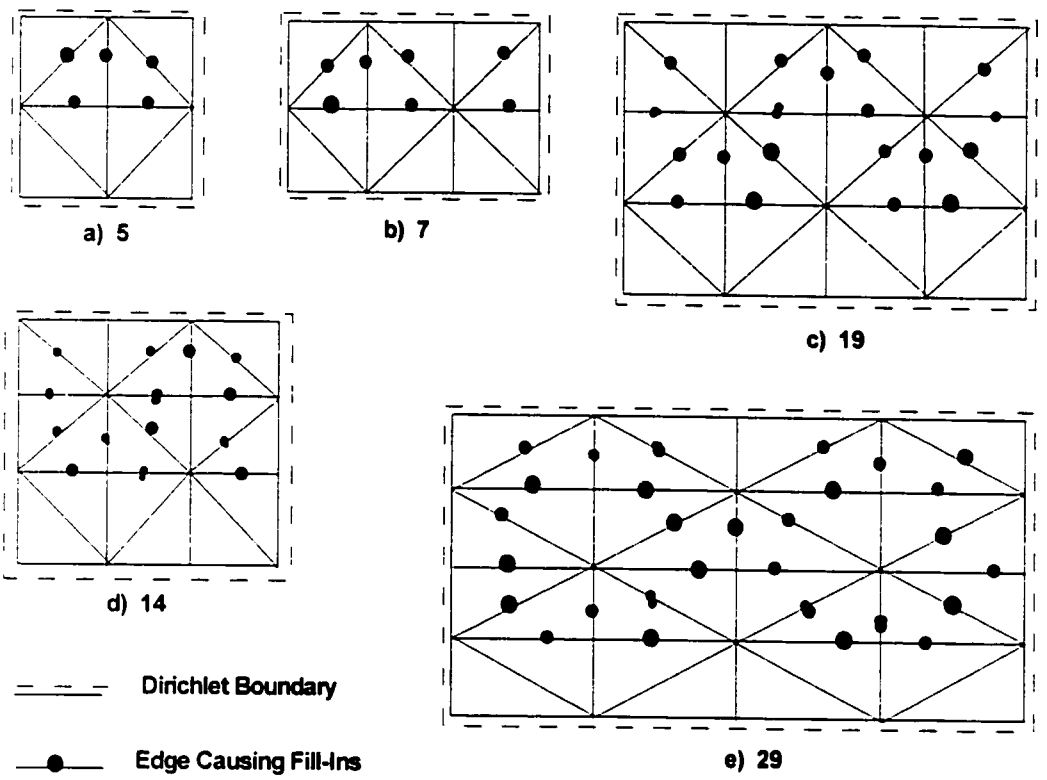
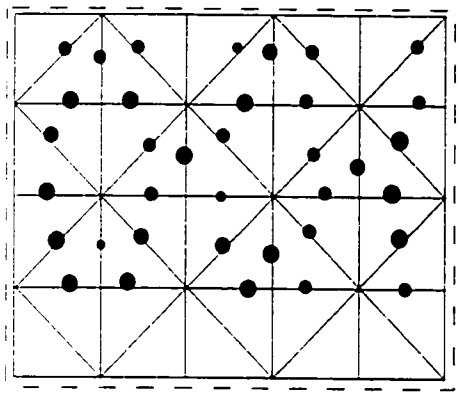
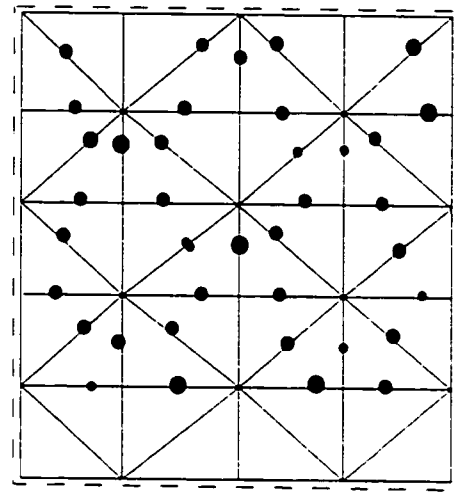


Figure 3.8 a) to e) -- Examples of Fill-In Edge Determination



f) 36

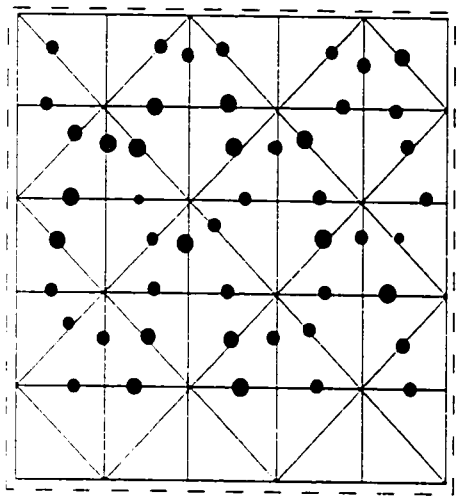


g) 38

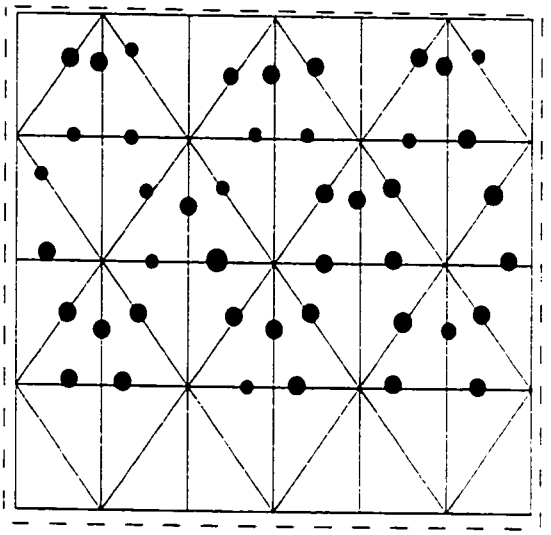
Dirichlet Boundary

Edge Causing Fill-Ins

Figure 3.8 f) to g) -- Examples of Fill-In Edge Determination



h) 48

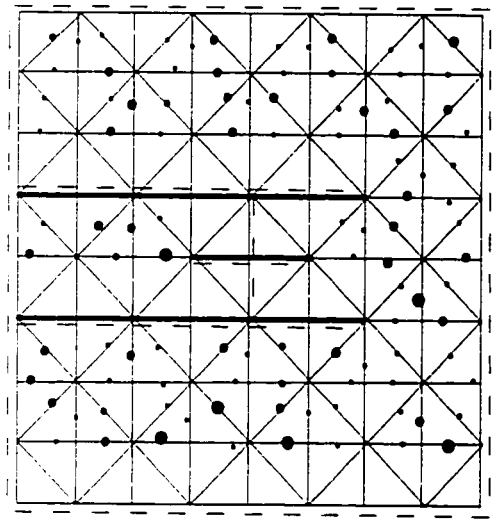


i) 44

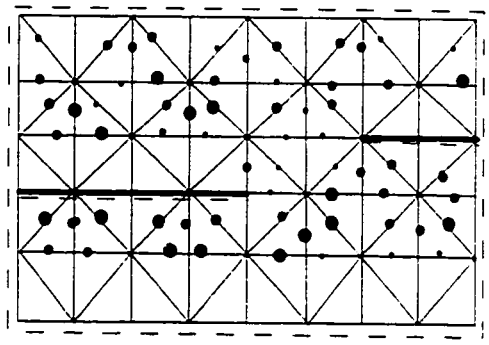
Dirichlet Boundary

Edge Causing Fill-Ins

Figure 3.8 h) to i) -- Examples of Fill-In Edge Determination



a) Stripline -- 102 Fill-In Edges



b) Dielectric -- 64 Fill-In Edges

Dirichlet Boundary

Edge Causing Fill-Ins

Figure 3.9 a) & b) -- Examples of Internal Dirichlet Boundaries

CHAPTER 4

COMPUTATIONAL ASPECTS OF VECTOR FINITE ELEMENT METHOD

4.1 VFEM Problem Formulation

In a problem formulated by means of the Vector Finite Element Method (VFEM), the system of equations describing the problem could either be in the form of a generalized eigenvalue equation,

$$[A]\{\phi\} - \lambda[B]\{\phi\} = \{0\} \tag{4-1}$$

or in the form of a deterministic equation,

$$[A]\{\phi\} = \{0\} \tag{4-2}$$

where $[A]$ and $[B]$ are known matrices and λ and $\{\phi\}$ are the unknowns.

In the research into the computational aspects of VFEM, a bounded problem described by a system of deterministic equations is selected as the vehicle, as the findings can be readily extended to a problem described by a generalized eigenvalue equation. The difference between

the solutions of the two problem types is one of computational scale rather than computational complexity. Stated another way, both problems are described by matrices, solution techniques of which are the same, albeit in the case of an eigenvalue problem, there are more matrices to solve, thus additional computational resources are required.

4.2 Structural Attributes of the VFEM Solution Matrix

The VFEM zone discretization and edge numbering system detailed in Chapter 3 results in a solution (coefficients) matrix that has the qualities of symmetry, sparsity, bandedness, diagonal dominance and low bandwidth. Figure 3.3 in Chapter 3 illustrates many of these qualities.

Symmetry

The symmetry is present relative to the matrix diagonal and the contributing factor is the solution matrix formation process itself, in that, in a finite element triangle, each non-boundary edge gets processed cyclically against all other adjacent edges, including itself. Here, adjacent edges are defined as edges where the angle subtended is less than 180 degrees. The result is a non-zero value at the respective matrix locations, and at their reciprocals. For example, edge 9 processed with edge 16 will result in a non-zero value at row 9 column 16, and at row 16 column 9 of the matrix.

Sparsity

The sparsity of the solution matrix is a direct result of the zone discretizition, in that, each edge is in contact with a maximum of four non-boundary edges. Therefore, in each matrix row

referring to an edge, there will be a maximum of five non-zero values, four due to adjacent edges and one due to the edge itself. While the number of unknown edges control the dimension of the matrix, the number of initial row non-zero values will be constant at five. Even with many fill-ins, the VFEM solution matrix will always have a low sparsity profile.

Bandedness

The solution matrix also exhibits the quality of bandedness that is symmetric with respect to the matrix diagonal. This bandedness is also concentrated along the matrix diagonal.

Diagonally Dominant

The solution matrix is also diagonally dominant because an edge is processed with itself twice, because each non-boundary edge is shared by two finite element triangles. As an example, triangle #11 formed of edges 13, 14, 19, while triangle #12 formed of edges 14, 15, 20. Therefore, on the matrix diagonal at row 14, column 14, the non-zero value will always be larger than any other non-zero value on row 14. This is true for all shared edges, and the VFEM solution (coefficient) matrix will always be diagonally dominant.

Low Bandwidth

The bandwidth of the solution matrix is directly related to the difference between the highest and the lowest edge numbers within a particular finite element triangle. The VFEM solution matrix bandwidth will be approximately twice the number of finite element zone divisions along the x-coordinate. As to the total extent of a matrix row in terms of non-zero values, this is approximately triple the number of finite element zone divisions along the x-coordinate. For instance, assuming

a VFEM zone is divided into 12 divisions along the x-coordinate, at row [i] the first non-zero value occurs at column [j] and the final non-zero value occurs at no further than column [j+36].

4.3 Techniques of Fill-In Minimization

Fill-ins are defined as certain left hand matrix locations which initially contain values equal to zero, but during the matrix decomposition process, their values get converted to non-zero values. From a computational efficiency aspect, the more non-zero values a sparse matrix contains, the longer would be its solution time, – normally, judicious programming practices avoid operating on zero values – thus minimizing the number of fill-ins would be beneficial.

There are several techniques in minimizing the fill-ins in a sparse matrix [33],[34]. These are described briefly as follows.

Matrix Reordering for Diagonal Dominance

This is carried out to keep the coefficients (left hand) matrix diagonally dominant, by swapping its rows and columns. Keeping the matrix diagonally dominant improves the accuracy of the solution. Given the deterministic matrix equation $[A]\{x\} = \{b\}$, the matrix reordering is accomplished by swapping rows in [A] and {b} together and by swapping columns in [A] and {x} together. There are two variants: full pivoting and partial pivoting.

a) In full pivoting, at each step of the matrix decomposition, the largest value element within the coefficient matrix (or sub-matrix) is chosen and placed on the diagonal to become the

pivot.

b) In partial pivoting, the coefficients matrix diagonal is reordered in the order of the largest value to the smallest value.

Reordering to Reduce Bandwidth

To reduce the bandwidth, the use of reverse Cuthill-McKee ordering is quite beneficial, as this technique reorders the solution matrix to a minimum bandwidth and concentrates all the non-zero elements near the diagonal. This not only reduces fill-ins, but it also makes matrix solution by direct decomposition very practical. The penalty for this benefit is the additional processing steps taken by the implementation.

Minimum Degree Rule

The coefficients matrix is reordered by placing as first row, the row containing the fewest non-zero elements. Then, in the remaining part of the matrix, the row with the fewest non-zero elements is placed as second row, and the process is continued until all of the rows are reordered in this fashion. This technique yields less fill-ins and sparser matrices, and is also called Least Non-zero Row Reordering. However, the additional processing overhead makes this technique less practical for VFEM matrix solutions.

Nodal Graph Method

Another technique that is available, especially in circuit analysis and SPICE type solutions, is the nodal graph procedure. However, unlike the nodal structure of a circuit, the VFEM nodal graph so generated (not to be confused with the VFEM zone nodes!) does not lend itself to this

procedure. When this procedure is tried out on a simple VFEM discretized zone, the resulting nodal graph is an enclosed curve and contains interior nodes, which will yield fill-ins when the graph nodes are eliminated. The characteristics of the procedure, namely the gradual elimination of the nodes, points to the reordering of the matrix so that it would yield a minimum number of fill-ins.

4.4 VFEM Matrix Reordering Option

The structural attributes of the VFEM solution matrix enumerated above – such as symmetry, sparsity, diagonal dominance, bandedness, and low bandwidth – impart to it the quality of a naturally ordered matrix structure, which, in turn, is the outcome of the VFEM zone discretization system detailed in Chapter 3.

Therefore, given this naturally ordered matrix structure with its built-in advantages, it is possible to forego many of the reordering options mentioned in the previous section, in consideration of the need for faster solutions, yet still be able to maintain accuracy, while avoiding the introduction of additional software coding complexity and overhead. However, if matrix reordering is preferred, the Minimum Degree Rule reordering is a good option.

For the VFEM sparse solution matrix, the best approach is to utilize its structural advantages. Furthermore, the structure of the left hand matrix is conducive to additional speeding up of its solution time by the careful analysis of its structure and by means of efficient software design, and finally, by determining how and where and by which techniques these speed ups can be implemented and obtained. These are treated in more detail below.

4.5 Solution of VFEM Matrices – Decomposition Methods

The matrix equations obtained from the VFEM formulation can be solved by means of two methods: direct methods (such as the LU and LDL^T decompositions), and iterative methods (such as the biconjugate gradient method) [18].

Normally, the direct methods make use of the Gaussian elimination technique. The Finite Element systems lend themselves to direct method solutions, and therefore matrix decomposition methods are very suitable as solution techniques. In the following sections, a brief description is given of these matrix decomposition methods.

4.5.1 LU Decomposition

A problem described by

$$[A]\{x\} = \{b\} \tag{4-3}$$

where [A] is the coefficient matrix, {x} is the unknowns, and {b} is the excitation matrix, can be reduced to the following form by decomposing [A] such that we have [18],[34],

$$[L][U]\{x\} = \{b\} \tag{4-4}$$

where [L] and [U] are the lower and upper triangular matrices. Then these can be restated as:

$$[U]\{x\} = \{z\} \tag{4-5}$$

$$[L]\{z\} = \{b\} \tag{4-6}$$

and [L] and [U] have the form:

$$[L] = \begin{vmatrix} l_{11} & 0 & 0 & \cdot & 0 \\ l_{21} & l_{22} & 0 & \cdot & 0 \\ l_{31} & l_{32} & l_{33} & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & 0 \\ l_{n1} & l_{n2} & l_{n3} & \cdot & l_{nn} \end{vmatrix} \tag{4-7}$$

$$[U] = \begin{vmatrix} 1 & u_{12} & u_{13} & \cdot & u_{1n} \\ 0 & 1 & u_{23} & \cdot & u_{2n} \\ 0 & 0 & 1 & \cdot & u_{3n} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ 0 & 0 & 0 & 0 & 1 \end{vmatrix} \tag{4-8}$$

The solution is arrived at by using **Forward Substitution on (4-6)**

$$z_1 = \frac{b_1}{l_{11}} \tag{4-9}$$

$$z_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=1}^{i-1} l_{ik} z_k \right) \qquad i > 1 \tag{4-10}$$

and by using Backward Substitution on (4-5)

$$x_n = \frac{z_n}{u_{nn}} \tag{4-11}$$

$$x_i = \frac{1}{u_{ii}}\left(z_i - \sum_{k=i+1}^{n} u_{ik} x_k \right) \qquad i < n \tag{4-12}$$

The procedure is as follows:

Step 1: Divide the first row by the pivot while leaving first column unchanged. The pivot is the row element on the matrix diagonal.

Step 2: The remainder of the coefficients matrix bounded by the by the first row and first column forms a sub-matrix. Process each element m_{ij} by subtracting the product of the corresponding elements in the pivot row and pivot column and replace by the result, namely, $m_{ij} \Leftarrow (m_{ij} - p_i \times p_j)$.

Step 3: Repeat Steps 1 and 2 until entire matrix is decomposed.

Step 4: Separate decomposed [A] into [L] and [U] for solution by Forward and Backward Substitutions.

In general, when [A] is sparse, so are [L] and [U]. The number of zeros in [L] and [U] does not have to be equal to that of [A].

In this work, the LU Decomposition was the method of choice for solving VFEM problems.

4.5.2 Crout's Decomposition

A modified form of LU Decomposition is attributed to P.D. Crout, and is called Crout's Decomposition Method [18],[34]. This method can be summerized as follows.

Decomposition Algorithm

$$u_{ii} = 1 \qquad i = 1, 2, 3, \ldots, n \tag{4-13}$$

$$l_{ij} = \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} u_{kj} \right) \qquad i \geq j \tag{4-14}$$

$$u_{ij} = \frac{1}{l_{ii}} \left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right) \qquad i < j \tag{4-15}$$

where u_{ii} is a diagonal element of [U] and l_{ij}, u_{ij}, and a_{ij} are matrix elements of [L], [U], and [A].

Forward Substitution Algorithm is given by:

$$z_1 = \frac{b_1}{l_{11}} \tag{4-16}$$

$$z_i = \frac{1}{l_{ii}}\left(b_i - \sum_{k=1}^{i-1} l_{ik} z_k \right) \qquad i > 1 \tag{4-17}$$

Finally, Backward Substitution Algorithm is given by:

$$x_n = z_n \tag{4-18}$$

$$x_i = \left(z_i - \sum_{k=i+1}^{n} u_{ik} x_k \right) \qquad i < n \tag{4-19}$$

4.5.3 LDL^T Decomposition

When [A] is a symmetric matrix, i.e. $a_{ij} = a_{ji}$, equation (4-15) is modified accordingly.

$$u_{ij} = \frac{1}{l_{ii}}\left(a_{ij} - \sum_{k=1}^{i-1} l_{ik} u_{kj} \right) = \frac{l_{ji}}{l_{ii}} \qquad i < j \tag{4-20}$$

Therefore, calculation of u_{ij} is not needed in the decomposition process. Furthermore, [L] can be put in the form

$$[L] = [\bar{L}][D] \tag{4-21}$$

where

$$[\bar{L}] = \begin{vmatrix} 1 & 0 & 0 & \cdot & 0 \\ \bar{l}_{21} & 1 & 0 & \cdot & 0 \\ \bar{l}_{31} & \bar{l}_{32} & 1 & \cdot & 0 \\ \cdot & \cdot & \cdot & \cdot & 0 \\ \bar{l}_{n1} & \bar{l}_{n2} & \bar{l}_{n3} & \cdot & 1 \end{vmatrix} \tag{4-22}$$

$$\bar{l}_{ij} = \frac{l_{ij}}{l_{jj}} \qquad i > j \tag{4-23}$$

[D] is the diagonal matrix where

$$d_{ii} = l_{ii} \tag{4-24}$$

$$d_{ij} = 0 \qquad i \neq j \tag{4-25}$$

Due to the symmetry condition, we have

$$[U] = [\bar{L}]^T \tag{4-26}$$

and therefore [L] [U] can be written as

$$[L][U] = [\bar{L}][D][\bar{L}]^T \tag{4-27}$$

The solution to the matrix is then obtained by using Forward Substitution to solve

$$[\bar{L}]\{z\} = \{b\} \tag{4-28}$$

and using Backward Substitution to solve

$$[D][\bar{L}]^T\{x\} = \{z\} \tag{4-29}$$

4.5.4 Cholesky Decomposition

The procedure described above is a slightly modified version of the Cholesky Decomposition [18]. The method can be described algorithmically as follows (omitting the bar over l_{ij}).

Decomposition Algorithm

$$d_{11} = a_{11} \tag{4-30}$$

$$d_{jj} = \left(a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2 d_{kk} \right) \qquad j > 1 \tag{4-31}$$

$$l_{ij} = \frac{1}{d_{jj}} \left(a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk} d_{kk} \right) \qquad i > j \tag{4-32}$$

Forward Substitution Algorithm will yield z_i values,

$$z_1 = b_1 \tag{4-33}$$

$$z_i = \left(b_i - \sum_{k=1}^{i-1} l_{ik} z_k \right) \qquad i > 1 \tag{4-34}$$

Finally, Backward Substitution Algorithm will yield x_i values,

$$x_n = \frac{z_n}{d_{nn}} \tag{4-35}$$

$$x_i = \frac{1}{d_{ii}} \left(z_i - \sum_{k=i+1}^{n} l_{ki} d_{ii} x_k \right) \qquad i < n \tag{4-36}$$

4.6 Domain Discretization and Mesh Types

In the course of this work, it was found that the mesh type selected for domain discretization into finite elements, did yield slight differences in values for edges or points that were located along equipotential lines. While the differences were not large enough to cause errors in the final results, the selection of the ultimate mesh type was made on the merits of whichever geometry yielded greater accuracy.

Initially, a unidirectional diagonal mesh composed of identical triangular finite elements was used to discretize the domain Ω. While the final results obtained were accurate, due to the unidirectional nature of the diagonal edges the interpolated edge values at the centroid of the finite element gave slight variances (~10E-7) along an equipotential line.

On the other hand, the use of a symmetrical mesh whose diagonal edges alternated in direction eliminated these variances, and this was expected since any computational artifact due to a particular diagonal direction cancelled out. The symmetrical meshes as shown in Chapter 3, Figures 3.7 a) and 3.7 b) were thus utilized throughout this work on account of their greater accuracy.

4.7 VFEM Matrix Equation Solution Process

The method described in Chapter 2, Section 2-4 was successfully applied to several electromagnetic closed-boundary problems, such as:

a) field distribution within a semi-infinite dielectric slab, mainly as a verification vehicle for testing the software. Excellent match of results between analytical and computational results;

b) stripline, which simulated electric field distribution within printed circuit board from buried conductor (track) and electric field leakage at card edges;

c) microstrip in a shielded enclosure and the determination of its characteristic impedance;

d) grooved microstrip in a shielded enclosure and its characteristic impedance behaviour with respect to groove depth and frequency.

All of these problems were deterministic two dimensional problems, where the transverse field components were the required unknowns. In the case of the microstrips, quasi-TEM mode propagation was assumed (the standard approach).

The solution process of the method is two-fold: firstly, all the edge values were obtained from the solution of the type of matrix equation described by Equation (4-3); secondly, the electric field values at the centroids of each triangular finite element were obtained by the interpolation of the three edge values of the finite element as specified in the method of Chapter 2, Section 2-4.

4.7.1 VFEM Solution (Coefficient) Matrix Decomposition Choice

As mentioned before, the VFEM is very amenable to direct decomposition techniques. In the process, one of the major issues is the memory storage requirements of the matrices, especially the coefficients (solution) matrix, which gets sparser in terms of non-zero values as the VFEM zone gets more refined. Therefore, it is very advantageous from a computational resources aspect, to implement a memory storage technique that will conserve memory as much as possible. There are several techniques, all of which involve storing only the non-zero values.

4.7.2 Unknowns Only Matrix Equation

Since the VFEM zone edges have known values as they form the Dirichlet Boundaries, these do not have to be included in the solution process for obvious reasons. Therefore, these values can be removed from the coefficients matrix, while simultaneously the excitation matrix is modified accordingly, to maintain the proper mathematical relationship.

The resulting matrix equation contains only the unknown edges' coefficient values, and the modified excitation values matrices (left and right hand matrices respectively). To have an orderly process, the left and right hand matrices are remapped to a new set of consecutive edge numbers; at the same time, a restoration vector is generated containing initial references. After the solution of the matrix equation, the restoration vector allows the recovery of the original VFEM zone edge numbering.

4.7.3 Memory Storage

To conserve memory, one dimensional array storage was utilized throughout. The actual implementation had two steps: an initial storage stage and a final storage stage.

In the initial stage, while being formed, both the left hand and right hand matrices' non-zero values were stored into their respective arrays, and their row and column matrix locations were stored in the index vectors. Once completed, the index vectors were sorted numerically while the left hand and right hand matrices' non-zero values were reordered accordingly.

In the final storage stage, a new left hand matrix storage space was formed to allow storage space for the fill-ins inevitably generated during the decomposition process. At the same time, new row and column pointer vectors plus a pivot vector were generated, all for use in the decomposition process.

4.8 VFEM Numerical Solution Acceleration Techniques

The numerical solution of a VFEM matrix equation in a quick and efficient manner is one of the most important requirements, if not the most important.

One of the goals of this work was to devise a matrix equation solution technique that was both rapid and efficient – while remaining accurate – and was conservative in its use of computer resources (e.g. CPU, memory, I/O, disk access, etc).

In many technical papers, it is stated that computational resources were quickly exhausted during numerical solutions, be it FDTD, FEM, TLM, etc, without alluding to any effort undertaken to improve the performance of the code. In some cases, lengthy solution times are mentioned in solving unknowns numbering under 2000, and sometimes much less.

In this present work, using the developed technique, total solution times for the solution of electromagnetic problems such as microstrips mentioned above, including processes such as VFEM zone mesh forming, zone discretization, matrix formation, matrix decomposition, unknown edge solutions, element field computation, and results storage into files, were one minute for ~5500

unknowns and 9.75 minutes for ~ 49000 unknowns, running on a 150 MHz ULTRA/Sun Solaris multiuser machine. All in all, an excellent performance was realized, and the goal of obtaining a high speed solution paradigm was reached. The contributing developments are described in the following sub-sections.

4.8.1 VFEM Solution Matrix Decomposition Process Acceleration

A VFEM zone that is discretized and has its edges numbered in a sequential order along the x-direction as described in Chapter 3, its coefficients matrix has the attributes mentioned in Section 4.2 above.

When the template of the coefficients (left hand) matrix (Chapter 3, Figure 3.3) is carefully studied, it can be seen that, at each level of the decomposition, only certain rows in the remainder of the matrix need to be processed, namely those whose first non-zero element column number is less than or equal to that level's pivot's column number.

Accordingly, to speed up the decomposition process considerably, a look-ahead decomposition table is created during the matrix formation process. This table identifies for each pivot in a given row, the set of rows below the pivot row where decomposition has to take place. For each edge, only those rows in which the decomposition should be carried out are treated, and unnecessary operations and testing algorithms are avoided during the solution process.

To elaborate the last part further, each row of the left-hand matrix represents an edge. In

the course of this work, it was discovered that for each edge, the entry of the look-ahead decomposition table is composed of all the row numbers denoting the fill-in edges in between itself and its higher numbered adjacent edges inclusive. Therefore, the look-ahead decomposition table could be formed readily, and utilized to control the matrix solution process.

Example: Chapter 3, Figures 3.2 and 3.8 e); Edge 21. Its higher numbered adjacent edge is edge 28. In between these edges, there are edges contributing to fill-ins, namely edges 22, 24, 25, and 26. Therefore, the decomposition table entry corresponding to edge 21 will contain row numbers 22, 24, 25, 26 and 28. The decomposition for level 21 will be carried out at only those row numbers.

This is the most important attribute of the VFEM solution matrix in addition to its qualities mentioned in Section 4.2 above, and the recognition and utilization of this attribute in accelerating the solution process is one of the original contributions of this work.

Also, a parallel process to the above, the end of each pivot row is identified during the decomposition stage, in order to provide a delimiter to the decomposed row. Once this end is reached, the process jumps to the next row on the set as indicated by the look-ahead table. This delimiting also has the additional benefit of determining dynamic memory allocation requirements.

In the event that the VFEM zone edges are numbered in a manner different from that outlined in Chapter 3, *a priori* determination or identification of edges contributing to fill-ins may not be possible. An alternative way to form the look-ahead decomposition table still exists and is described as follows.

Specifically, at step [m], pivot[m] is located at row [m], column [p], then all rows numbered greater than [m] whose first element column number is less than or equal to [p] are identified, and reference vectors are formed containing the row counts and the row numbers for step[m], where the decomposition will be carried out.

Implementation of the above techniques yielded excellent speed up in solution times, while retaining the accuracy of the solution.

4.8.2 VFEM Solution Acceleration by Stepwise Refinement of the VFEM Zone

In another technique designed and implemented for solution acceleration, the VFEM zone is initially discretized into m by n sections, forming ($m \times n$) cells, ($2 \times m \times n$) triangular finite elements, the edge values are computed as before. Then, instead of proceeding with the computation of the transverse components of the cell finite elements, the cell diagonal edge values are discarded, and each cell formed by the m by n divisions is further processed as a mini-VFEM zone by subdividing it into p by q subdivisions and ($2 \times p \times q$) finite elements, whose Dirichlet boundaries are the previously computed rectilinear edge values of the cell. Then, the edge values of the mini-VFEM zones and the transverse components of the mini-finite elements are calculated. This two step coarse and fine process is extremely fast and its memory storage requirements are much more modest. The net result is similar to a VFEM zone discretized to ($m \times p$) by ($n \times q$) sections having ($2 \times (m \times p) \times (n \times q)$) triangular finite elements. This VFEM solution technique is a new approach and was developed during the course of this work. It was the fastest solution technique used in this work by far.

With respect to accuracy, the stepwise refinement results are slightly less accurate than those obtained by the conventional finely discretized VFEM zone, for edges that are next to the VFEM zone boundaries when the Dirichlet Boundary values are small, and the VFEM zone is quite large (The reason is that discarding coarse mesh diagonal edges near a zone Dirichlet Boundary, will give rise to slight inaccuracies in the computation of fine mesh edges). Elsewhere, the results were very close to those obtained with conventional discretization. Further work is indicated to improve the accuracy near Dirichlet boundaries.

4.8.3 Irregular Mesh VFEM Zone Technique

In this case, the VFEM zone is discretized into a coarser mesh in areas where the electric or the magnetic field is changing slowly, and into a finer mesh where the field is changing much faster. This was tried out on the same test problem, one with a coarse-fine mesh and the other with fine mesh throughout, the fine mesh dimensions being the same. The results obtained in the area of the fine mesh were comparable to those of a full-zone fine mesh, and the computations were carried out faster due to the smaller number of finite elements within the test zone, hence smaller number of unknown edges. The sole advantage of this technique is in enabling the extension of the physical dimensions of the VFEM zone for a given number of edges. There is nothing original about this technique, but it was tried out for comparison, but not pursued further due to time limitations.

4.8.4 Equivalent Doubling of VFEM Zone Mesh Density – Dual Mesh Approach

During his work, one other efficient and rapid VFEM solution technique was devised and

applied to actual problems very successfully. In this technique, a solution precision can be readily implemented that is quasi-equivalent to that obtained from a VFEM zone mesh density double to the original one, at a cost of solving the problem twice.

In the two dimensional VFEM, after the unknown edges are solved for, the x and y components of the electromagnetic field components at the centroid of each finite element are computed. These field components in each finite element represent the final profile of the electromagnetic field within the VFEM zone bounded geometry.

Solving the VFEM problem twice, once with the Spoke Pattern mesh (Chapter 3, Figure 3.7.a)), and again with the Diamond Pattern mesh (Chapter 3, Figure 3.7.b)), and then overlaying the solutions on the VFEM zone geometry, yields results that are almost equivalent in precision to the electromagnetic field values of a VFEM zone solved by the use of a mesh having double the density of the original mesh.

In practical terms, a four-fold saving is obtained in the number of unknowns, and approximately a sixteen-fold saving is obtained in matrix storage requirements, not to mention an untold saving in computational time that will have to be taken to solve the much larger number of unknowns. This is the advantage of the dual-mesh geometry solution, a concept which is simple, yet like all such simple solutions, very powerful and useful.

As far as is known, this concept of dual mesh solution was not used before elsewhere,

4.9 Summary of Contributions

The concepts outlined in this Chapter made significant contributions towards the overall objective of the work undertaken in this thesis. These are:

a) Determination and utilization of the structural attributes of the VFEM matrix to realize an efficient solution process;

b) Beneficial use of these attributes by the conception of a look-ahead decomposition table to accelerate the VFEM matrix solution;

c) Recognition of the utility of pre-determined fill-in edges in the formation of the look-ahead decomposition table entries;

d) Cell level VFEM zone discretization technique for very fast solution options;

e) Dual mesh VFEM zone discretization approach to double zone precision at modest cost of processing.

CHAPTER 5

VFEM APPLICATION TO VARIOUS MICROSTRIP PROBLEMS

5.1 Introduction

In this Chapter, the results of the various problem solutions are presented. The VFEM methodology applied was that described in Chapter 2. All the coding was written by the author, including mesh generation, domain discretization, matrix formation, matrix decomposition, etc. For the purpose of brevity, some of the obtained results are not presented here. These include the semi-infinite dialectric slab, and the stripline problem.

In the case of the semi-infinite dialectric slab, this problem was used as a benchmark for both the VFEM methodology's validity and the correctness of the coding. It was solved both analytically and numerically, and the results showed excellent agreement.

The stripline problem (Chapter 3, Figure 3.9a)) was the first electromagnetic problem solved

that had an actual application, namely, a simulated printed circuit card and field leakage through its edge. Again, the VFEM methodology solved the problem quickly and accurately. Over a large frequency range, the behaviour of the field distribution within the dialectric and outside it was observed and compiled, and plots were generated, both static and dynamic. These are not included here, due to the extensive amount of data involved.

5.2 Microstrip Characteristic Impedance

The characteristic impedance of a microstrip line is an important parameter, as it has a bearing on the amount of loss a signal is exposed to during propagation on the line. The characteristic impedance can be obtained by applying the VFEM methodology to obtain the electromagnetic field distribution around the microstrip, and then computing the voltage and the current from these. The usual problem geometry has the microstrip line within a shielded metal box, and the standard approach is to treat the microstrip as a quasi-TEM mode device. To accommodate this assumption, the capacitance and the relative permittivity have to be modified accordingly. The appropriate expressions are given by Collin [35] as follows.

Assuming an isotropic dialectric substrate and negligible line thickness, the characteristic impedance Z_c of the microstrip line is given by:

$$Z_c = \sqrt{\frac{\mu_0 \varepsilon_0}{\varepsilon_{e(0)}}} \frac{1}{C_a} = \frac{1}{c C_a \sqrt{\varepsilon_{e(0)}}} \qquad (5\text{-}1)$$

where C_a is the capacitance of the unscaled air-filled line, $\varepsilon_e(0)$ is the effective permittivity quasistatic value, and c is the speed of light in vacuum. Assuming zero line thickness (T = 0),

$$\varepsilon_e(0) = \frac{\varepsilon_r + 1}{2} + \frac{\varepsilon_r - 1}{2}(1 + 12\frac{H}{W})^{-0.5} + F(\varepsilon_r, H) \tag{5-2}$$

$$\begin{aligned} F(\varepsilon_r, H) &= 0.02(\varepsilon_r - 1)(1 - \frac{W}{H})^2 \qquad \frac{W}{H} < 1 \\ &= 0 \qquad\qquad\qquad\qquad\quad \frac{W}{H} \geq 1 \end{aligned} \tag{5-3}$$

$$C_a = \frac{2\Pi\varepsilon_0}{\ln\left(8\frac{H}{W} + \frac{1}{4}\frac{W}{H}\right)} \qquad \frac{W}{H} \leq 1 \tag{5-4}$$

$$C_a = \varepsilon_0\left(\frac{W}{H} + 1.393 + 0.667\ln\left(\frac{W}{H} + 1.444\right)\right) \qquad \frac{W}{H} > 1 \tag{5-5}$$

The frequency dependence of the effective permittivity is given by Kobayshi [35],

$$\varepsilon_e(f) = \varepsilon_r - \frac{\varepsilon_r - \varepsilon_e(0)}{1 + \left(\frac{f}{f_a}\right)^m} \tag{5-6}$$

$$f_a = \frac{f_b}{0.75 + (0.75 - 0.332\varepsilon_r^{-1.73})\frac{W}{H}} \tag{5-7}$$

$$f_b = \frac{47.746}{H(\varepsilon_r - \varepsilon_{e(0)})^{0.5}} \tan^{-1}\left(\varepsilon_r\left(\frac{\varepsilon_{e(0)} - 1}{\varepsilon_r - \varepsilon_{e(0)}}\right)^{0.5}\right) \tag{5-8}$$

$$m = m_0 m_c \leq 2.32 \tag{5-9}$$

$$m_0 = 1 + \frac{1}{1 + \left(\frac{W}{H}\right)^{0.5}} + 0.32\left(1 + \left(\frac{W}{H}\right)^{0.5}\right)^{-3} \tag{5-10}$$

$$m_c = 1 + \frac{1.4}{1 + \left(\frac{W}{H}\right)}\left(0.15 - 0.235e^{-0.45\frac{f}{f_a}}\right) \qquad \frac{W}{H} \leq 0.7 \tag{5-11}$$

$m_c = 1$ for all other values,

Substrate height H and microstrip width W are in mm, f is in GHz, angle is in radians.

5.3 Frequency Dependence of Characteristic Impedance

The characteristic impedance will change with frequency. The direction of change is dependent on the method used in obtaining it.

According to Edwards [36], "there are five ways of computing the characteristic impedance:

$$Z_{c1}(f) = \frac{V}{I}$$

$$Z_{c2}(f) = \frac{V_c}{I}$$

$$Z_{c3}(f) = \frac{P}{II^*} \qquad (5\text{-}12)$$

$$Z_{c4}(f) = \frac{V_c V_c^*}{P}$$

$$Z_{c5}(f) = \frac{VV^*}{P}$$

The general theoretical conclusion is that Z_{c2} and Z_{c4} always rise with frequency whereas the remaining fall. V_c is the voltage at the centre of the microstrip, while V is the mean voltage of the microstrip" [36].

On the other hand, Gupta et al [37] say that "...Z_c should decrease with the rise of $\varepsilon_e(f)$ as

frequency rises, but the rising $\varepsilon_e(f)$ implies that the fields are concentrated below the microstrip, which in turn implies that the effective width W is decreased, hence Z_c rises slightly as frequency increases" [37].

To obtain the frequency-dependent value of characteristic impedance of the microstrip, Equation (5-1) should be modified as follows:

$$Z_c = \sqrt{\frac{\mu_0 \varepsilon_0}{\varepsilon_{e(f)}}} \frac{1}{C_a} = \frac{1}{cC_a\sqrt{\varepsilon_{e(f)}}} \tag{5-13}$$

5.4 Microstrip In Shielded Box

The quasi-TEM mode of the microstrip can be solved as follows [38]:

$$\begin{aligned} E_z &= 0 \\ H_z &= 0 \\ \nabla_t^2 E &= 0 \\ \nabla_t^2 H &= 0 \end{aligned} \tag{5-14}$$

Thus, for a wave travelling in a positive direction, we have,

$$H_x = -\frac{E_y}{\eta} \qquad H_y = \frac{E_x}{\eta} \tag{5-15}$$

The problem is solved for the transverse components of E, from which the transverse components of H can be obtained by means of Equation (5-15). The current I can be obtained by summing the transverse components of H around a circuital path enclosing the cross-section of the microstrip line. The voltage V can also be readily obtained from the transverse components of E. Therefore, the characteristic impedance Z_c can then be computed and compared to the value obtained by means of the above formulas.

5.4.1 Case 1:

The parameters of the microstrip were: W = H = 0.635 mm, ε_r = 10.2, f = 10 GHz. Z_c was computed using Equation (5-1) which gave a value of 48.285 ohms. The fields were computed by means of VFEM, with an initial excitation of 1 volt/metre between the microstrip line and the ground plane. The current was obtained by two separate summation techniques.

$$I = \sum_{path} H_k \Delta d_k \tag{5-16}$$

where k represents either x or y, and Δd represents the discrete path interval.

The results for Z_c were 50.796 ohms and 51.394 ohms, which translates into differences

of 5.2% and 6.4% respectively, in comparison with the theoretical value of the characteristic impedance.

5.4.2 Case 2:

The parameters of the microstrip were: W = 2.4 mm H = 0.8 mm, ε_r = 2.22, f = 2 GHz. Again, Z_c was computed using Equation (5-1) which gave a value of 50.993 ohms. The fields were computed by means of VFEM, with an initial excitation of 1 volt/metre between the microstrip line and the ground plane. Once again, the current was obtained by two separate summation techniques. The results for Z_c were 50.932 ohms and 51.141 ohms, which translates into differences of 0.1% and 0.29% respectively, in comparison with the theoretical value of the characteristic impedance.

As to why Case 1 results weren't as good as in the second case, many times the dimensions of the shielded box (which happens to be the VFEM domain or zone) and the microstrip characteristics are conducive to resonance and dispersion which will affect the results. Also, in the VFEM solution, actual dielectric constant values were used, rather than the effective dielectric constants.

5.5 Grooved Microstrip

Another application of the VFEM methodology to a microstrip in a shielded box was the grooved microstrip. A paper by Li and Atsuki [39] describes the concept. The dielectric substrate

has side grooves of rectangular cross-section, and the authors state that "..theoretical results for these (microstrip) lines demonstrate significant effects on the conductor loss reduction by the (air-filled) side grooves and inlaid dielectric". The paper presents results that indicate lower conductor losses due to the groove architecture. The authors utilized a numerical computational method called the partial-boundary element method (p-BEM).

In order to replicate this result, this concept of a groove was applied to a microstrip using the VFEM methodology. The problem was formulated as follows: relative permittivity ε_r of 10.2, microstrip width W of 0.6 mm, substrate height H of 0.63 mm, groove width of 0.14 mm, groove depth varying from 0 mm to 0.21 mm in steps of 0.015 mm, and frequency varying from 10 GHz to 40 GHz. However, the box dimensions caused resonance around 28 GHz to 34 GHz, therefore only the results between 10 GHz and 28 GHz are dealt with.

Given the extremely fine discretization used in the problem for great accuracy, increasing the box dimensions was not an option as the memory requirements became appreciable. As the frequency increased and the box approached resonance, the characteristic impedance became smaller, as expected. However, the depth of the groove also had some contribution in affecting the field distributions.

The main finding was that within a groove depth of approximately 5% of the substrate height, the groove generated lower values for the characteristic impedance, in the frequency range 10 GHz to 18 GHz, as compared to a plain microstrip. After that, resonance effects took over, and the plain microstrip provided lower values for the characteristic impedance.

What remains to be investigated is to see what effect the groove width variations will have. Also, it is expected that lowering the frequency range below 10 GHz would enable the desired groove depth range to be larger than 5% of the substrate height determined above, probably to about 7%.

The results obtained in the solution of this grooved microstrip problems were utilized in training Artificial Neural Networks, the next phase of this work which investigated alternate methods of modelling and solving electromagnetic problems.

Figure 5.1 shows the family of curves for the characteristic impedance values of the grooved microstrip versus frequency, except for the curve that is different, which represents the plain (ungrooved) microstrip.

At any given frequency, the effect of the groove depth is evident. The shallow groove reduces the concentration of current at the edge of the conductor strip, thus lowering the overall current in the conductor. However, as the groove deepens past a depth equal to approximately 5% of the substrate thickness, the fields are attenuated less in the air dielectric within the groove; consequently, the characteristic impedance value is higher.

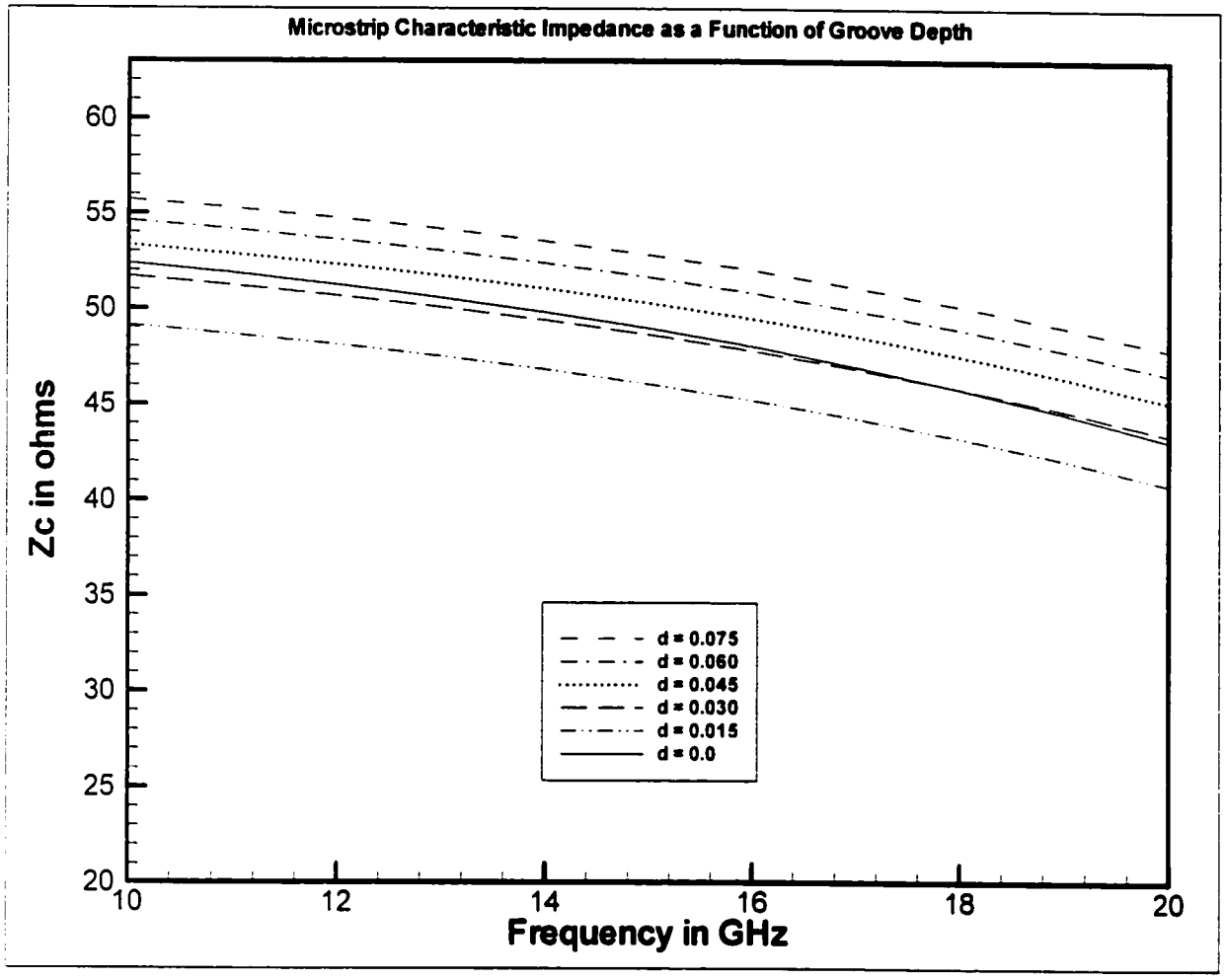


Figure 5.1 – Microstrip Characteristic Impedance as a Function of Groove Depth

Table 5.1 below shows the behaviour of the characteristic impedance for a plain (ungrooved) dielectric between 10 and 20 GHz.

Frequency GHz	Z_c (VFEM) Ω	Z_c (f) Ω (Eq 5-13)	Z_c (0) Ω (Eq 5-1)
10.0	52.38	48.31	49.44
11.0	51.84	48.16	49.44
12.0	51.24	48.00	49.44
13.0	50.56	47.85	49.44
14.0	49.80	47.69	49.44
15.0	48.96	47.54	49.44
16.0	48.03	47.39	49.44
17.0	46.98	47.23	49.44
18.0	45.82	47.08	49.44
19.0	44.53	46.94	49.44
20.0	43.08	46.79	49.44

Table 5.1 Characteristic Impedance Behaviour of Plain Microstrip

Table 5.2 below shows the behaviour of the characteristic impedance for grooved dielectric microstrip at five different groove depths between 10 and 20 GHz.

Frequency GHz	Zc (VFEM) d = 0.0 mm	Zc (VFEM) d = 0.015 mm	Zc (VFEM) d = 0.030 mm	Zc (VFEM) d = 0.045 mm	Zc (VFEM) d = 0.060 mm	Zc (VFEM) d = 0.075 mm
10.0	52.38	49.13	51.70	53.33	54.62	55.73
11.0	51.84	48.65	51.22	52.85	54.15	55.27
12.0	51.24	48.10	50.68	52.32	53.62	54.76
13.0	50.56	47.49	50.07	51.72	53.04	54.18
14.0	49.80	46.81	49.39	51.05	52.38	53.54
15.0	48.96	46.06	48.64	50.31	51.65	52.82
16.0	48.03	45.22	47.80	49.49	50.84	52.03
17.0	46.98	44.29	46.87	48.57	49.94	51.14
18.0	45.82	43.25	45.84	47.55	48.94	50.16
19.0	44.53	42.10	44.68	46.41	47.82	49.06
20.0	43.08	40.81	43.39	45.13	46.57	47.83

Table 5.2 Characteristic Impedance Behaviour of Grooved Microstrip

5.6 VFEM Computational Performance

As mentioned before, the various problems solved in the course of this work were run on a 150 MHz, 400 MByte RAM, Ultra machine with Sun/Solaris Unix environment, located at the Communications Research Centre, Ottawa, Ontario, Canada. All the coding used for the VFEM methodology was written by the author in the C language.

For the microstrip problem in Case 1 above, the shielded box dimensions were 6.985 mm for length and 5.08 mm for height, the VFEM zone was discretized into 44 divisions along the x-axis and 32 divisions along the y-axis, with uniform interval each of 0.15875 mm. The discretization yielded a total of 2816 triangular finite elements, 4300 edges, 160 Dirichlet boundary edges, and 4140 unknown edges that were solved for. Finally, the transverse components of the field at the centroids of each triangle were computed as per the VFEM methodology. The problem was solved for five separate frequencies, from 8 GHz to 12 GHz, in steps of 1 GHz. The total execution time per frequency for the complete VFEM package was 43 seconds.

For the microstrip problem in Case 2 above, the shielded box dimensions were 12.0 mm for length and 6.4 mm for height, the VFEM zone was discretized into 60 divisions along the x-axis and 32 divisions along the y-axis, with uniform interval each of 0.2 mm. The discretization yielded a total of 3840 triangular finite elements, 5852 edges, 200 Dirichlet boundary edges, and 5652 unknown edges that were solved for. Finally, the transverse components of the field at the centroids of each triangle were computed as per the VFEM methodology. The problem was solved for five separate frequencies, from 1.6 GHz to 2.4 GHz, in steps of 0.2 GHz. The total execution time per frequency for the complete VFEM package was 60 seconds.

For the grooved microstrip problem above, the shielded box dimensions were 2.68 mm for length and 1.575 mm for height, the VFEM zone was discretized into 134 divisions along the x-axis and 105 divisions along the y-axis, with x-axis interval of 0.02 mm, and y-axis interval of 0.015 mm. The discretization yielded a total of 28140 triangular finite elements, 42449 edges, 550 Dirichlet boundary edges, and 41899 unknown edges that were solved for. Finally, the transverse components of the field at the centroids of each triangle were computed as per the VFEM

methodology. The problem was solved for frequencies from 10 GHz to 40 GHz, in steps of 1 GHz. The total execution time per frequency for the complete VFEM package was 585 seconds.

It is difficult to carry out an exact comparison of the above computational performance results with others obtained elsewhere, due to various differences such as, computer parameters, operating system, problem formulation, coding approach, matrix solution process, etc. However, a broad criterion such as the respective proportional population of unknowns and non-zero matrix elements can be utilized to obtain a qualitative assessment. Accordingly, a review of various published material indicated that the computational performance of the VFEM solution methodology developed in this work compared very favourably with, and in some cases exceeded, the performance results obtained elsewhere, thereby achieving one of the objectives of this work.

It should also be mentioned that the solution times listed for the VFEM problems above are total process times. They include times for all of the steps such as VFEM zone discretization, mesh generation, finite element edge formulation, matrix formations, matrix solutions and finite element field component computations. On the other hand, performance examples in the reviewed material were at times unspecific as to what these computation times represented. It is assumed that these are referring to matrix solution times only.

5.7 Computational Complexity

The general aspects of the computational complexity involved in the formulation of a VFEM problem was within the normal bounds for a numerical technique of this type. The coding was

optimized throughout for all the steps included in the formulation and the solution of the VFEM problem, i.e., VFEM zone discretization, mesh generation, finite element edge formulation, matrix formations, matrix solutions and finite element field component computations.

For the most complicated and time consuming process, namely, the matrix solution, the number of operations involved is approximately $c \bullet (UE)^2$, where c is a constant, and UE is the number of matrix rows (or columns), which represent the unknown edges of the VFEM zone. This is arrived at as follows.

Given a VFEM zone, let X and Y denote the number of divisions along the x- and y-coordinate axes. Assuming X is equal to Y for simplicity, then,

(1) No of unknown edges: $UE = (3X^2 - 2X)$

(2) No of matrix decomposition levels: $(UE - 1)$ or $(3X^2 - 2X - 1)$

(3) No of row references in the look-ahead decomposition table entry for each level: ~1.5X

(4) No of non-zero elements between pivot column and row end column: ~1.5X

(5) No of pivot row division operations for each non-zero matrix element:: 1

(6) No of decomposition operations for each non-zero matrix element: 2

(7) No of non-zero elements in each row (including fill-ins): ~3X

(8) No of operations in Forward Substitution: $(UE)^2$

(9) No of operations in Backward Substitution: $\sim((UE)^2 - UE)$

No of total operations involved in matrix solution is:

$$Ops = (2) \bullet \{ [(4) \bullet (5)] + [(6) \bullet (4) \bullet (3)] \} + (8) + (9)$$

$$Ops = (3X^2 - 2X - 1) \bullet \{[1.5X \bullet 1] + [2 \bullet 1.5X \bullet 1.5X]\} + (3X^2 - 2X)^2 + \{(3X^2 - 2X)^2 - (3X^2 - 2X)\}$$

$$Ops = 4.5X^3 - 3X^2 - 1.5X + 13.5X^4 - 9X^3 - 4.5X^2 + 9X^4 - 12X^3 + 4X^2 + 9X^4 - 12X^3 + 4X^2 - 3X^2 + 2X$$

Given that the fourth order terms will predominate, this can be simplified to:

$$Ops = 13.5X^4 + 9X^4 + 9X^4$$

$$Ops = 31.5X^4 \text{ or } \sim 3.5(UE)^2$$

This result is expected as the solution complexity is directly related to the VFEM zone discretization complexity and to the structure of the solution matrix. (For a dense (non-sparse or full) matrix, the total number of operations involved in its solution is $((UE)^3 - UE) / 3$ [34].)

The computational performance obtained is indicative of this relationship. It is necessary to quantify the relationship, therefore, the performance results obtained for the three microstrip problems are plotted in Figure 5. 2, which graphically illustrates the super linearity quality of the VFEM problem solution methodology developed in this work.

Finally, from the information in Figure 5.2, the computational time with respect to the number of unknown edges can formulated analytically as:

VFEM Processing Time = 10^m seconds

where $m = \{(1.1368 \bullet \log_{10}(UE)) - 2.4873\}$

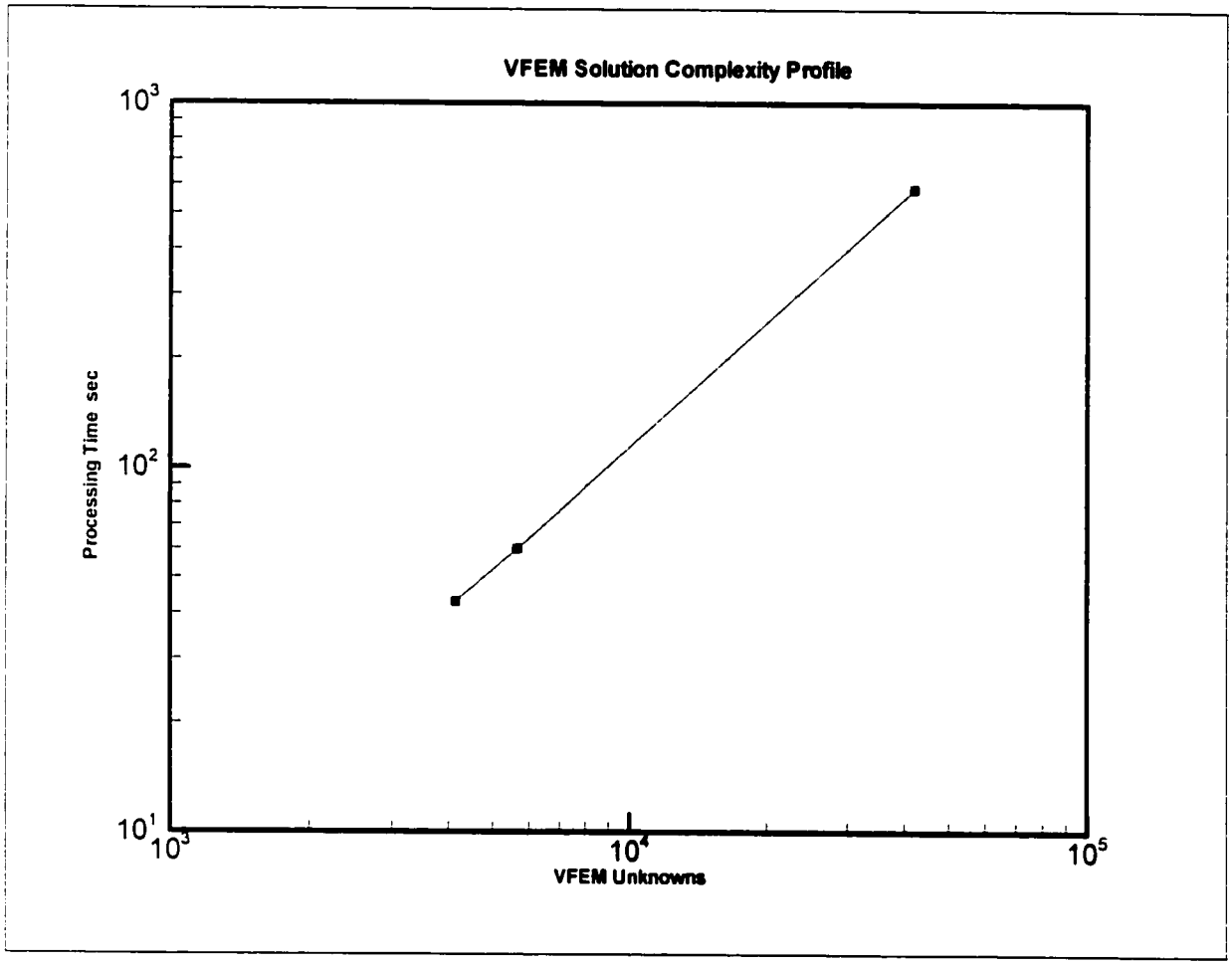


Figure 5.2 – Relationship of Processing Time to Number of Unknown Edges

5.8 Conclusions

The above-listed computational results indicate excellent performance superior to its counterparts in use elsewhere. The application of the results obtained from the research in this work presented in Chapters 3 and 4 enabled this superior performance.

Of course, while every effort was made to optimize the coding as much as possible, additional computational speed-ups and efficiency are always possible with further improvements in the matrix decomposition coding routine, and the application of any new developments along these lines. One promising option is the melding of an efficient implementation of the reverse Cuthill-McKee algorithm with the techniques developed in this work. It is anticipated that such an approach will yield matrix solution acceleration at least one order higher than the currently demonstrated efficiency in this thesis. Another promising option lies in improving the accuracy of the technique outlined in Chapter 4, namely the stepwise (cell level) refinement of the VFEM zone. This is by far the fastest solution method applied in this work, and once its accuracy is improved, it has the potential to become the premier matrix solution method.

CHAPTER 6

ARTIFICIAL NEURAL NETWORK MODELLING

6.1 Introduction

The well-known numerical electromagnetic models such as Finite-Difference Time Domain (FDTD), Finite Element Method (FEM), Transmission Line Matrix (TLM) all use types of iterative and sequential algorithms to arrive at solutions. The classical forms of these techniques require that problems incorporate bounded geometries, in other words, the solution domain be subject to well-defined electromagnetic boundary conditions. The computational resources required, in both hardware and software, are extensive, and this factor more than any other consideration, dictates limitations and places upper bounds on the size of the problem space that can be accommodated.

During the last few years, new concepts were developed such as Absorbing Boundary Conditions (ABC) [18],[30],[40] and Perfectly Matched Layer (PML) Boundaries [41],[42] to refine boundary condition imposition. Application of these techniques allowed "unbounding" of the solution domain which then enabled problems involving open infinite domains to be solved, by providing the

advantage of truncating a large dimensional space to a small dimensional space.

Another technique that allows improvements to the problem dimensions is the use of hybrid electromagnetic methods [18],[43]-[56], such as the combination of Vector Finite Element Method with the Integral Equation Method [18], or some other method. Normally, in problems that involve open infinite domains, the Sommerfeld radiation condition can only be imposed by extending the domain of discretization far from the source region. The net effect of this situation is to increase the VFEM zone discretization and resultant matrices' size. The Integral Equation Method incorporates the Sommerfeld radiation condition through the use of the appropriate Green's function [18]. By combining the two methods, the discretization zone can be made smaller.

Lately, alternate computational methods have been applied successfully to electromagnetic problems. One such method that is quite suitable is the Artificial Neural Networks (ANN) modelling concept [20],[57]-[64]. Artificial Neural Networks are composed of many simple elements operating in parallel inspired by biological nervous systems [65]. ANNs have a high level of adaptability and are capable of performing complex functions in various fields of applications by suitable training. It is this versatility in conjunction with their rapid response-to-stimulus characteristic that makes the ANNs a viable method of choice to model various electromagnetic field problems. Indeed, ANNs have been applied to electromagnetic problems successfully, especially in combination with the Finite Element Method. One such application involved the solution of an axisymmetrical problem configuration consisting of a coaxially driven dielectric ring resonator antenna [20],[60].

6.2 Rationale for Artificial Neural Networks as a Modelling Technique

In view of the fact that an electromagnetic problem represents a natural phenomenon, and is, in turn, represented by an analytical model described by a differential equation, its solution can be arrived at:

a) either by employing techniques associated with an analytical model such as:
   i) a closed form analytical solution,
   ii) a numerical (computational) solution,
   iii) an analog computer solution;

b) or by use of a Artificial Neural Network model trained to emulate the phenomenon's behaviour.

The set of input/output data generated by the analytical model is used to train the Artificial Neural Network, so that it can correctly estimate or interpolate the behaviour of the phenomenon in between the training data points set, thereby yielding an accurate and speedy solution.

To investigate the concept further, the results obtained in the solutions of the microstrip problems by the VFEM methodology were utilized as a set of data points to train various Artificial Neural Network models. The successful implementation of this investigation demonstrated not only the value of the Artificial Neural Network as a valid modelling tool for electromagnetic problems, but also realized a real-time numerical computational prototyping vehicle.

6.3 Artificial Neural Networks – Basics

Nowadays, an Artificial Neural Network (ANN) has come to mean any computing architecture that consists of massively parallel interconnection of simple neurons [66]. Over the years, many researchers have contributed to the advancement of this field, such as McCulloch, Pitts, Hebb, Minsky, Rosenblatt, Widrow, Hoff, Papert, Kohonen, Anderson, Grossberg, Rumelhart, Hopfield, Elman, Hinton, Williams, Steinbuck, McClelland, Rosenfeld, and so on [67]-[89]. For engineers, the main interest in Artificial Neural Networks is as powerful problem solving techniques or vehicles. For instance, Artificial Neural Networks can increase the speed of computation by orders of magnitude, because of their massively parallel nature. Furthermore, due to their adaptive qualities, Artificial Neural Networks can adapt to changes in the data and learn the characteristics of a system from its input signals. Also, due to their non-linear nature, they can perform filtering operations which are beyond the capabilities of conventional linear filtering techniques. Finally, most important of all, their ability to learn and to use a system's underlying complex functional structure coupled to their parallel architecture can overcome some of the limitations of conventional computers and artificial intelligence systems.

Basically, an Artificial Neural Network is a network architecture formed by one or more layers of neurons. A neuron is represented by a set of inputs, a set of associated synaptic weights, a transfer function, and an output as shown in Figure 6.1. The inputs $\mathbf{x}_i$ (i = 1,2, ... , n), are linearly combined through the synaptic weights $\mathbf{w}_i$ (i = 1,2, ... , n), such that the resulting combination **p** is given by:

$$p = \sum_{i=1}^{n} x_i w_i \qquad (6\text{-}1)$$

which is then passed through the transfer function **f** of the neuron before the output **y**, so that:

$$y = f(b + \sum_{i=1}^{n} x_i w_i) \qquad (6\text{-}2)$$

where **b** is a modifier or a bias value.

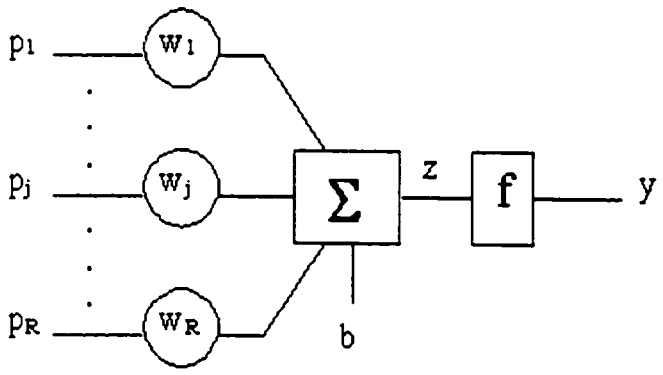


Figure 6.1 McCulloch-Pitts Model of an Artificial Neuron

The characterisation of a system by the general term Artificial Neural Network usually implies an ability to learn. We can describe learning as the process by which a neural system acquires the ability to carry out certain tasks by adjusting its internal parameters according to some learning scheme [66]. Learning can be classed as either *supervised* or *unsupervised*.

Supervised learning involves the type of learning when given the actual sets of input and output values of the system, the ANN eventually determines the correct mapping between the two data sets.

Unsupervised learning involves the type of learning where the output data of the system is not known *a priori*, and the ANN learns by creating associations between individual elements and by correlation of input data, while keeping network parameters within specified bounds.

A general model for an ANN can be described by the following system of coupled non-linear differential equations [66],[80].

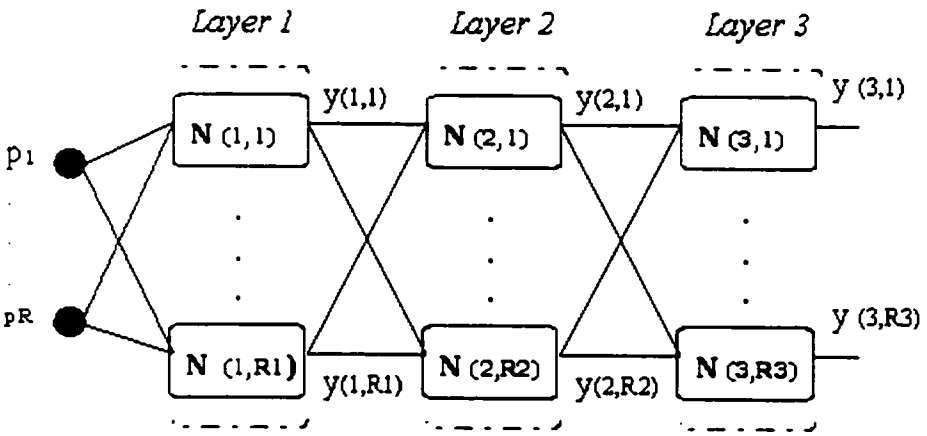
$$\frac{dx_i}{dt} = a_i(x_i)[b_i(x_i) - \sum_{j=1}^{n} c_{ij} d_j(x_j)] \qquad (6\text{-}3)$$

where the matrix $C = [c_{ij}]$ is symmetric, $a_i(.)$ are positive definite functions, and $d_j(.)$ are monotonic functions. The above system of differential equations can be used to represent an ANN. Thus, x_i is the activity level of the i^{th} neuron, $d_j(x_j)$ is the output of the j^{th} neuron, c_{ij} is a synaptic weight which represents the strength of the connection between the i^{th} and j^{th} neurons The term under the summation represents the net input from the neurons which form the network to the i^{th} neuron.

There are three types of ANN architectures: Feed Forward, Feed Back, and Self-Organizing, briefly described below.

6.3.1 Feed Forward ANN Architecture

The Feed Forward ANN (Figure 6.2) consists of one or more layers of non-linear processing elements. The elements belonging to neighbouring layers are connected by sets of synaptic weights. Because the output of each layer is fed to the next layer of elements, the network is called a Feed Forward ANN. These networks are quite popular and are readily trainable to provide a desired response to a given input. The Feed Forward ANN accomplishes this response behaviour by adapting its synaptic weights during the learning phase on the basis of a selected learning rule. The training of these networks is carried out by means of a set of input and output data, namely a *training set*.



Feed-forward architecture with three layers

Figure 6.2 Example of Feed-Forward ANN Architecture

Feed Forward ANNs are frequently trained by first order learning algorithms, derived on the basis of the *gradient descent* method. These algorithms are widely recognized to reach convergence slowly. For multi-layered Feed Forward ANNs, alternative learning algorithms were developed by modifying the optimization strategy and/or employing adaptation rules other than the *gradient descent*, resulting in the type of Feed Forward ANN called Feed Forward with Back Propagation ANN.

The training of the Feed Forward ANN is completed when the discrepancy (or error) between the desired and the estimated responses for all the input-output pairs reaches a minimum criterion,

$$e = \sum_{k=1}^{m} \sum_{i=1}^{n} g(v_{i,k} - y_{i,k}) \qquad (6\text{-}4)$$

where g is a positive definite, convex, continuous, and differentiable everywhere function, and v_k is the desired output, and y_k is the actual response.

The training of a Feed Forward ANN is usually performed in *adaptation cycles* or *epochs*. Each *epoch* involves the adaptation of all the synaptic weights of the ANN with respect to all of the associations and relationships which form the *training set*. During each *epoch*, the synaptic weights are updated by minimizing e_k.

The training algorithm for a Feed Forward ANN with backpropagation can be summarized as follows:

1. Select the ANN architecture and its parameters such as error criterion, neuron layers, neurons per layer, learning rate etc;
2. Initialize the ANN neuron synaptic weights and biases to random values;
3. Input the training set data vector;
4. Compute output data vector;
5. Compute the backpropagation error, and compare to 'end training' criterion;
6. If error meets or exceeds the specified error criterion, end training, otherwise, update the synaptic weights and biases in the direction of error minimization, and adapt learning rates and momentum values to maintain ANN's stability, and return to step 3.

In this work, a Feed Forward ANN with Backpropagation was utilized.

6.3.2 Feed Back ANN Architecture

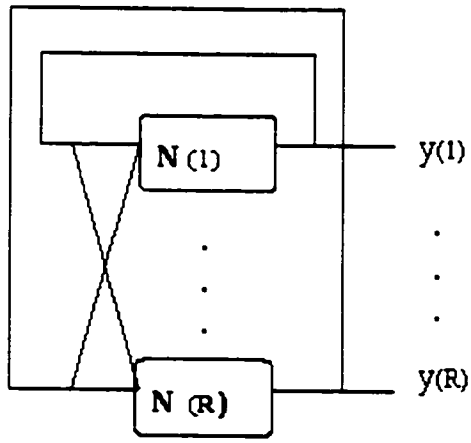
Feed Back ANNs are those neural network structures in which there exists feedback connections from the output layer to the input layers, or networks in which there exists lateral interactions between the neurons and sharing of information from neighbouring output nodes. These architectures are sometimes called recurrent or recursive neural networks. These networks are quite powerful because they are sequential rather than combinatorial. The feedback from output to input allows the network to exhibit temporal behaviour [65].

In Feed Back ANN, the input information defines the initial activity state of the feedback

system. After some state transitions, the asymptotic equilibrium state is identified as the outcome of the computations. The most common types are the Elman and Hopfield Feed Back ANNs.

Elman networks are two-layer backpropagation networks with the addition of a feedback connection from the output of the hidden layer to its input. This feedback path allows Elman networks to learn to recognize and generate temporal patterns, as well as spatial patterns. Main application areas are signal processing and prediction where time plays a dominant role [65],[88].

Hopfield networks (Figure 6.3) can be used to store one or more stable target vectors. These stable vectors can be viewed as memories which the network recalls when provided with similar vectors which act as a cue to the network memory. Hopfield networks can act as vector categorization or error correction networks. Input vectors are used as the initial conditions to the network, which recurrently updates until it reaches a stable output vector.



Recurrent architecture (*Hopfield* NN)

Figure 6.3 An Example of Feed-Back ANN Architecture

6.3.3 Self-Organizing (Associative) ANN Architecture

An associative system has the property that given an input vector p_1, it will produce an associated output vector a_1, given an input p_2, it will produce an output vector a_2, and so on. Various learning rules are involved in these networks such as Hebb, Instar, Outstar, or Kohonen. Self-Organizing ANNs can learn to detect regularities and correlations in their inputs, and adapt their future responses to that input accordingly. These networks are trained by competitive learning and also by the Learning Vector Quantization (LVQ) method, which is competitive learning in a supervised manner [86].

6.3.4 Artificial Neural Network Stability

The stability of a dynamical system is of critical importance if the system represents an ANN. The stability of a dynamical system is guaranteed if there exists a Lyapunov function for this system. A Lyapunov function is a function of the state space which is non-increasing along the trajectories of the state space. Cohen and Grossberg [80] have provided a rigorous proof that there exists a global Lyapunov function for the model described by the differential equation (6-3) above, which is of the form (d' is a derivative):

$$V = -\sum_{i=1}^{n} \int^{x_i} b_i(\xi_i) d_i'(\xi_i) d\xi_i + \frac{1}{2} \sum_{j=1}^{n} \sum_{k=1}^{n} c_{jk} d_j(x_j) d_k(x_k) \qquad (6\text{-}5)$$

The crucial point of this proof is that V satisfies the basic property of a Lyapunov function.

The time derivative of V can be evaluated by:

$$\frac{dV}{dt} = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} \frac{dx_i}{dt} \tag{6-6}$$

It can easily be verified that since $c_{ij} = c_{ji}$

$$\frac{\partial V}{\partial x_i} = -d_i'(x_i)[b_i(x_i) - \sum_{j=1}^{n} c_{ij} d_j(x_j)] \tag{6-7}$$

or

$$\frac{dV}{dt} = -\sum_{i=1}^{n} a_i(x_i) d_i'(x_i)[b_i(x_i) - \sum_{j=1}^{n} c_{ij} d_j(x_j)]^2 \tag{6-8}$$

Since it was assumed that $a_i(x_i) \geq 0$ and $d_i'(x_i) \geq 0$, then $dV/dt \leq 0$. When $dV/dt = 0$, the system reaches its equilibrium.

6.3.5 Artificial Neural Network Training

A physical system can be represented by an input, an output, and a transfer function. The behaviour of this system is intrinsically described in mathematical form by its transfer function. An ANN can be trained to emulate the transfer function of the system.

The optimization of any training process has two related goals: best possible accuracy and least time to train. The critical entities in training an ANN are the selection of the neuron functions, the number of layers and their neurons, values of the neuron synaptic weights and biases, the error criterion, and the learning algorithms. In order to optimize the ANN training in accordance with the above-stated goals, the underlying relationship, which is the key concept, has to be determined. In fact, this relationship can be considered as the description of the transfer function of the ANN, if one looks at an ANN as a system in its own right.

At initialization of an ANN, its neurons are assigned synaptic weight and bias values. These change throughout the training process until the error criterion is satisfied. At that point, the values of the weights and biases are considered final. Alternately, initialization places the ANN at a specific point on the error topography. At the end of the training, the ANN is at a minimum point on the error topography. Therefore, optimization of an ANN training will occur when the minimum energy trajectory is found on the error topography. The determination of a provable methodology to identify and/or to find this minimum error trajectory for a given ANN architecture – which is a far from simple process – will advance the field considerably, and is the subject of further research.

There are several ANN training optimization techniques having the main goal of fast and accurate way of arriving at fast convergence while minimizing as much as possible the required resources (number of neurons, training epochs, sum squared error goal, and so on). In many engineering problems, function approximation by means of ANN architectures is one of the main applications of ANNs. Some techniques involve spectral analysis [90], while in this work, training by decimation technique is utilized, whereby the ANN gets to be designed and trained for a near-optimum performance by the use of subset of the training set data, and then further trained with

progressively larger subsets of the training set data.

An Artificial Neural Network can be viewed as a universal function approximator or as a transfer function of a non-linear physical process. The optimization of this function can be posed as a *least squares* problem. The Levenberg-Marquardt algorithm is the standard method applied in the minimization of *mean-square error* criteria [78],[91],[92],[93]. Also, the Levenberg-Marquardt algorithm is quite suitable for Feed Forward with Backpropagation ANNs because such ANN networks enable easy calculation of the Jacobian matrix of the Levenberg-Marquardt system [60]. Therefore, the chosen model for the ANN application in this work was a Feed Forward with Backpropagation ANN trained by the Levenberg-Marquardt algorithm.

Looking further into the transfer function characteristics or ANN parameters, we can represent these parameters as the coordinates of a vector $v \in R^L$ where L denotes the number of ANN parameters. Then, the best v in terms of *least squares* could be found [94],[95]:

$$f(x, v) \rightarrow \min_{v \in V}, \qquad V \in R^L \tag{6-9}$$

$$f(x, v) = \sum_{i=1}^{M} e_i(x, v)^2 = e(x, v)^T e(x, v) \tag{6-10}$$

where e(x,v) is the ANN error vector given by

$$e(x, v) = t(x) - a(x, v) \tag{6-11}$$

where $t(x)$ is the desirable or target output for an input x, $a(x,v)$ is the actual output of the ANN as a function of input x and ANN parameter vector v, and M in (6-10) denotes the number of input-output training data pairs.

To improve the iterative procedure, we can exploit the characteristics of the least-squares problem described by (6-9) and (6-10). By substituting expression (6-11), we get,

$$f(v) = \|t - a(v)\|^2 \rightarrow \min_{v \in V}, \qquad V \in R^L \tag{6-12}$$

The gradient of (6-12) is given by,

$$\nabla f(v) \equiv \frac{\partial f}{\partial v} = -2\sum_{i=1}^{M} e_i \frac{\partial a(x_i, v)}{\partial v} \equiv g(v) \tag{6-13}$$

where e_i is as in (6-11). The Hessian of (6-12) is described by:

$$H_{\alpha,\beta} = \frac{\partial^2 f}{\partial v_\alpha \partial v_\beta} = -2\sum_{i=1}^{M} \frac{\partial^2 a_i}{\partial v_\alpha \partial v_\beta} e_i + 2\sum_{i=1}^{M} \frac{\partial a_i}{\partial v_\alpha} \frac{\partial a_i}{\partial v_\beta} \tag{6-14}$$

As v^l approaches v^*, the errors e_i approach 0. Therefore, by the use of the Newton method,

$$v^{k+1} = v^k - \rho^k H(v^k)^{-1} \nabla f(v^k) \tag{6-15}$$

we can approximate the Hessian as,

$$H_{\alpha,\beta} \approx G_{\alpha,\beta} = 2\sum_{i=1}^{M} \frac{\partial a_i}{\partial v_\alpha} \frac{\partial a_i}{\partial v_\beta} \tag{6-16}$$

It can be seen readily that

$$G = 2\left(\frac{\partial a}{\partial v}\right)^T \left(\frac{\partial a}{\partial v}\right) \geq 0 \tag{6-17}$$

This type of modification of the Newton method is, in fact, referred to as the Gauss-Newton method [94],[96].

To obtain the direction of the iteration in (6-12), a(x,v) can be expanded in a Taylor series

$$a(x,v) = a(x,v^k) + \frac{\partial a}{\partial v}\Big|_{v=v^k} (v - v^k) + \cdots \tag{6-18}$$

The next direction of the iteration direction can be found using the sum of the first two terms of (6-18),

$$\arg\min_v \left\| t - \sum_{j=1}^{2} a(x,v) \right\|^2 = \left\| t - a(x,v^k) - \frac{\partial a}{\partial v}(v - v^k) \right\|^2 \tag{6-19}$$

Let g_β denote the βth component of the gradient, then

$$g_\beta = -2\sum_{i=1}^{M} \frac{\partial a_i}{\partial v_\beta}(t - a_i) \tag{6-20}$$

Comparing (6-19) and (6-20) and using (6-17), it can be seen that the solution providing the minimum to (6-19) satisfies the following equation:

$$\left(\frac{\partial a}{\partial v}\right)^T\left(\frac{\partial a}{\partial v}\right)(v - v^k) = \left(\frac{\partial a}{\partial v}\right)^T\left(t - a(x, v^k)\right) \tag{6-21}$$

This is the same solution as in Gauss-Newton method:

$$v - v^k = -G^{-1}(v^k)g(v^k) \qquad G \geq 0 \tag{6-22}$$

Now, if one applies a modification so that instead of G, $G + \lambda P$ is used, where

$$G + \lambda P > 0, \quad \lambda > 0, \quad P > 0 \tag{6-23}$$

It is possible to put $P = I$, where I is the unit matrix, which is Levenberg's option. Then, letting $J(v)$ be the $M \times L$ dimensioned Jacobian matrix of $e(v)$, d^k be the search direction on the kth iteration ($d^k = v - v^k$) , and applying (6-23), a set of linear equations are obtained [91],[92], [96]:

$$\left(J(v^k)^T J(v^k) + \lambda^k I\right)d^k = -J(v^k)e(v^k) \tag{6-24}$$

Equation (6-24) is called the Levenberg-Marquardt Method. The scalar λ^k in (6-24) controls both the magnitude and the direction of d^k. When λ^k is zero, the direction of d^k is identical to that of the Gauss-Newton method. As λ^k tends to infinity, d^k tends towards a vector of zeros and a steepest descent direction. We can describe the algorithm as follows:

1. Create an initial parameter vector v^0 and an initial value λ^0 (e.g. $\lambda^0 = 0.01$);
2. Determine the search direction d^k from (6-24);
3. If $f(v^{k+1}) < f(v^k)$ then λ^{k+1} is reduced by a factor, otherwise it is increased. Here, $f(v)$ is a functional of (6-12);

4. Check the stop criterion. If it is satisfied, then $v^* = v^k$, and stop, else go to 2.

Note: The stop criterion is the following: error goal is met, λ becomes too large, or the maximum number of iterations (epochs) has occurred.

The backpropagation algorithm applies a correction Δw_{ji} to the synaptic weight w_{ji} connecting the output of neuron i to the input of neuron j. When applied to a multielement network, the backpropagation technique adjusts the weights in the direction opposite to the instantaneous gradient

$$\partial e_k^2 / \partial w_{ij}$$

of the sum square error in weight space [65],[82],[87],[97],[98].

6.4 Conclusion

In this Chapter, a brief overview of Artificial Neural Networks was presented, including basic concepts, network architecture types, stability requirements, and training algorithm based on the Levenberg-Marquardt system. Collectively, these make up the foundation for successful modelling of electromagnetic or other types of engineering problems by means of non-analytical techniques.

CHAPTER 7

ARTIFICIAL NEURAL NETWORK MODELLING OF PLAIN AND GROOVED MICROSTRIP

7.1 Introduction

In this Chapter, the successful implementation of the Artificial Neural Network (ANN) model to an electromagnetic problem is presented. The problem selected for this implementation was the grooved microstrip geometry outlined in Chapter 5. This problem was solved by means of the VFEM methodology, and the values of the microstrip characteristic impedance for both plain and grooved geometries were obtained. These values listed in Chapter 5, Table 5.2 form a data set, describing both the frequency-dependent and/or groove-dependent behaviour of each microstrip geometry. This data set was used to train the ANN architectures that were selected to model the characteristic impedance behaviour of the microstrip. In Chapter 6, the rationale for the use of ANNs to model electromagnetic field problems was presented. The benefit of modelling an electromagnetic field problem by ANNs is that, once trained, they can extrapolate accurate solutions in real time thereafter.

7.2 Modelling with Single Input ANN Architecture

The initial step in modelling with ANN was the task of determining the suitable ANN architecture. Given the fact that the microstrip characteristic impedance behaviour as a function of frequency, with groove depth as a parameter, produced a family of curves, the obvious approach was to use the ungrooved geometry (i.e. zero depth groove) microstrip results as the vehicle in the investigation for the suitable ANN architecture, composed of a single input, one or more hidden layers of neurons, and a single output. The results of the initial ANN architecture determination are outlined in the following Sections.

The modelling of the microstrip characteristic impedance behaviour falls under the category of *function approximation*. A Feed Forward network with Backpropagation, having one or more hidden layers having non-linear transfer functions (usually sigmoidal), and one output layer having linear transfer functions, is capable of approximating any function with a finite number of discontinuities with arbitrary accuracy [65]. In addition, the two-layer sigmoid/linear network can represent any functional relationship between inputs and outputs if the sigmoid layer has enough neurons [65]. In the initial step for the determination of a suitable ANN architecture, both ANN with one hidden sigmoid layer and ANN with two hidden sigmoid layers were utilized. The training method selected was the Levenberg-Marquardt approximation. All ANN work was carried out on a 200 MHz Pentium machine, using MATLAB Version 5.1. The Levenberg-Marquardt algorithm was selected because it is the fastest training available in the MATLAB Neural Network Toolbox for Feed Forward networks with Backpropagation [65].

The training plan was to train the ANN with the available characteristic impedance values for the frequency range f_1 to f_n, for each value of the groove depth d. Once trained, the ANN would output a corresponding characteristic impedance value for any new input frequency within the range f_1 to f_n, that will match closely one obtained from any VFEM solution in the same range. Figure 7.1 below shows the implementation of the ANN modelling in this work.

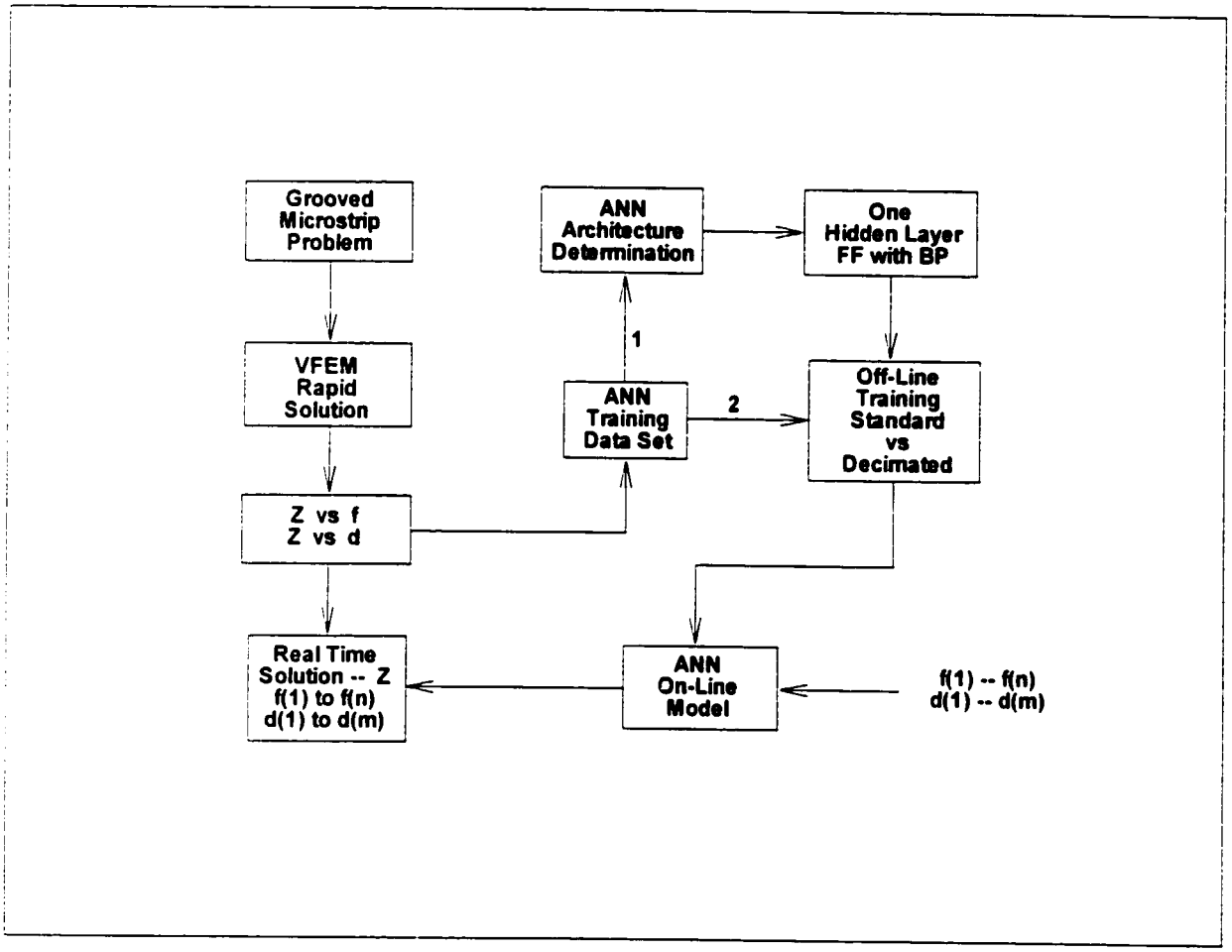


Figure 7.1 ANN Modelling Implementation

7.2.1 ANN with One Hidden Layer

A total of eight ANN architectures with one hidden neuron layer were used and excellent error performance results were obtained. The non-linear transfer function of the neurons in the hidden layer was *tansig* , while the linear transfer function of the neurons in the output layer was *purelin*, as provided by the Matlab Neural Network Toolbox [65]. Figures 7.2 through 7.9 show the relative error obtained (ΔZ/Z), and the desired error goal and the achieved error by the training process for each architecture. In some Figures, the latter two quantities are indistinguishable.

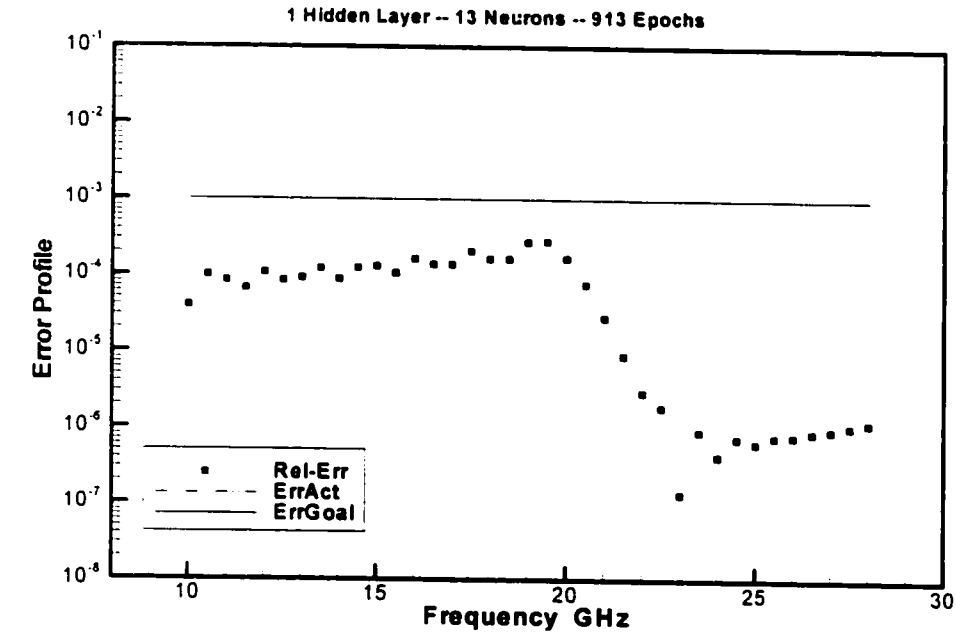


Figure 7.2 Error Performance – 1 Hidden Layer with 13 Neurons

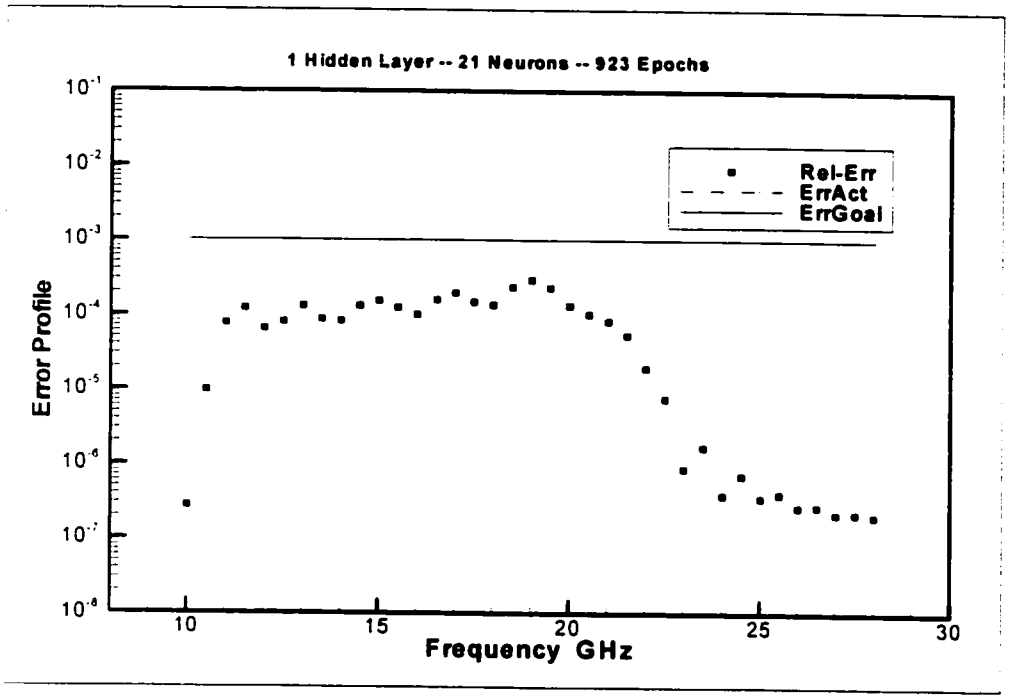


Figure 7.3 Error Performance – 1 Hidden Layer with 21 Neurons

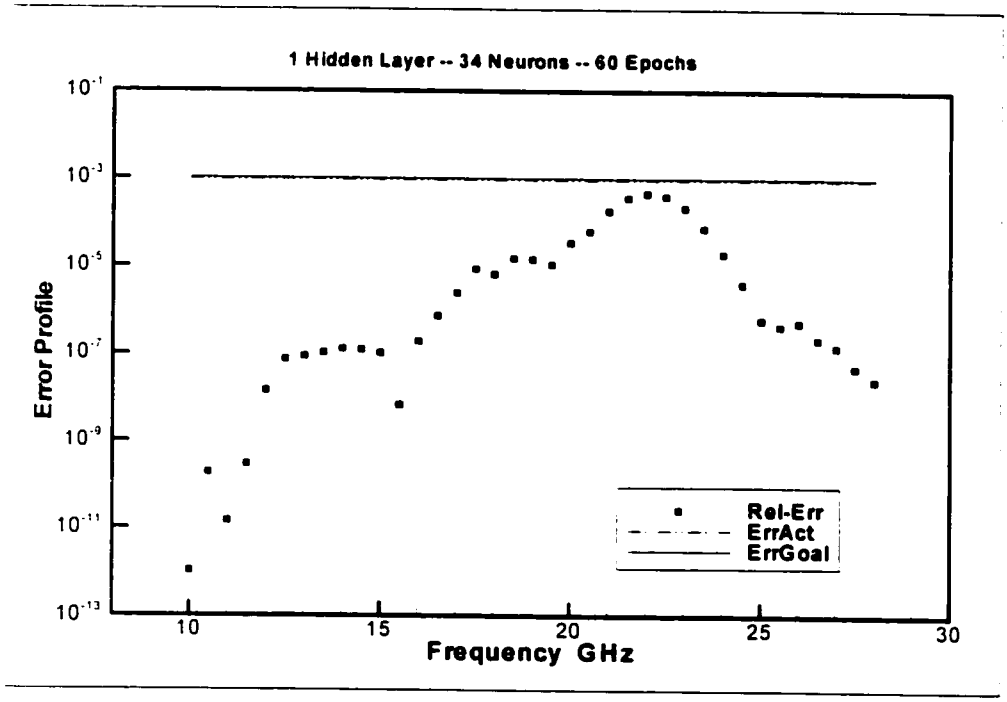


Figure 7.4 Error Performance – 1 Hidden Layer with 34 Neurons

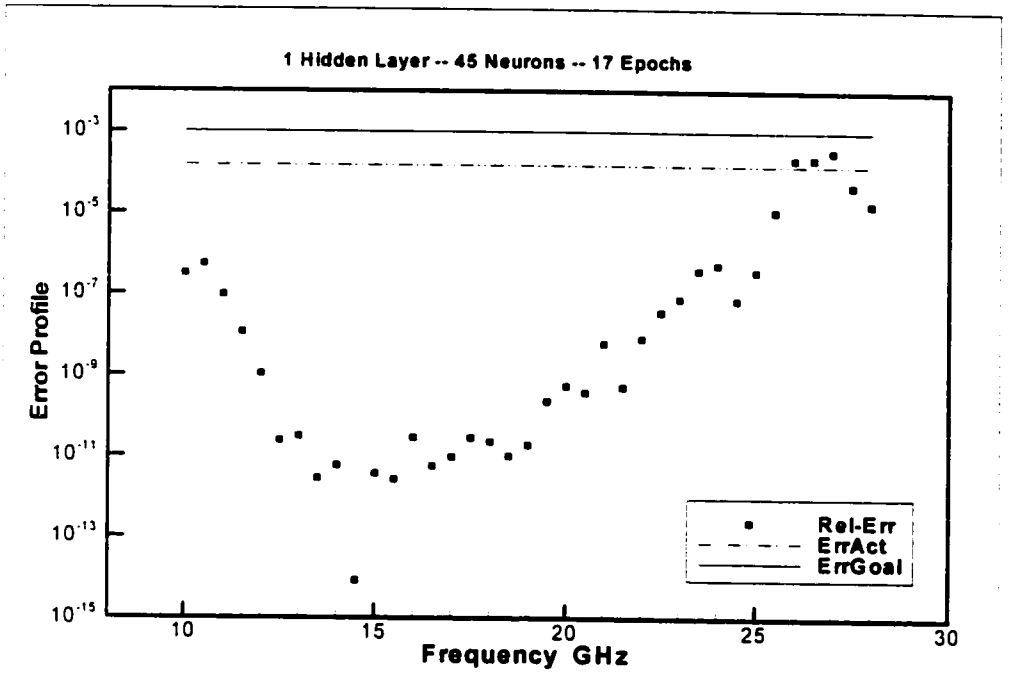


Figure 7.5 Error Performance – 1 Hidden Layer with 45 Neurons

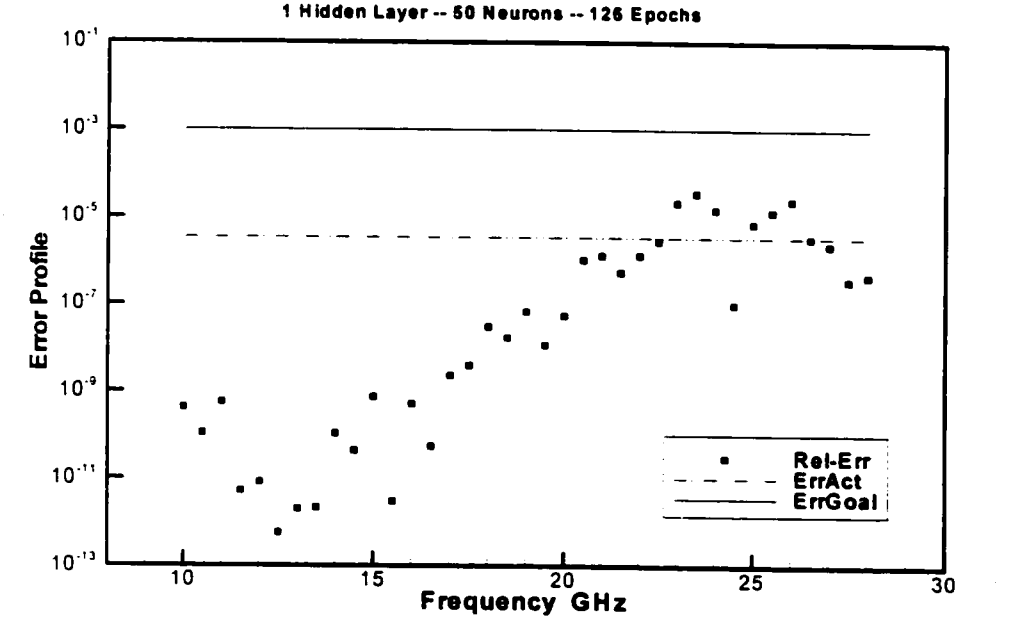


Figure 7.6 Error Performance – 1 Hidden Layer with 50 Neurons

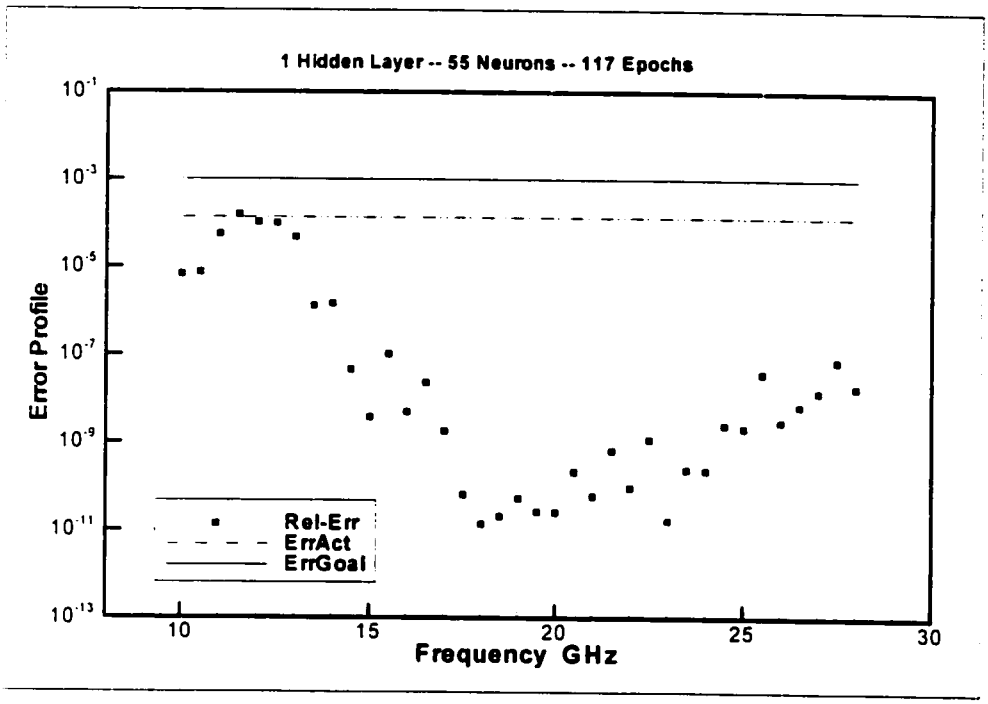


Figure 7.7 Error Performance – 1 Hidden Layer with 55 Neurons

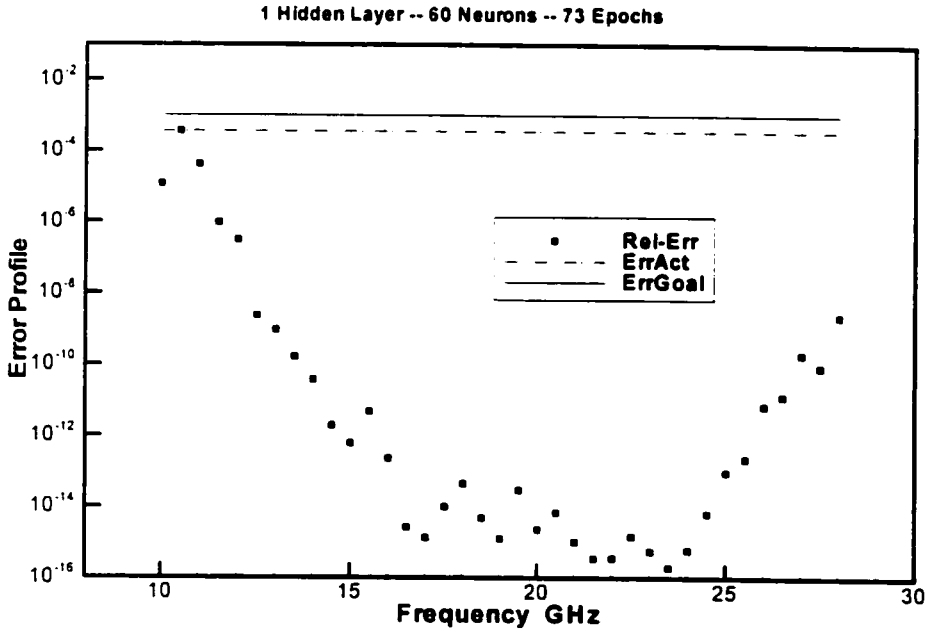


Figure 7.8 Error Performance – 1 Hidden Layer with 60 Neurons

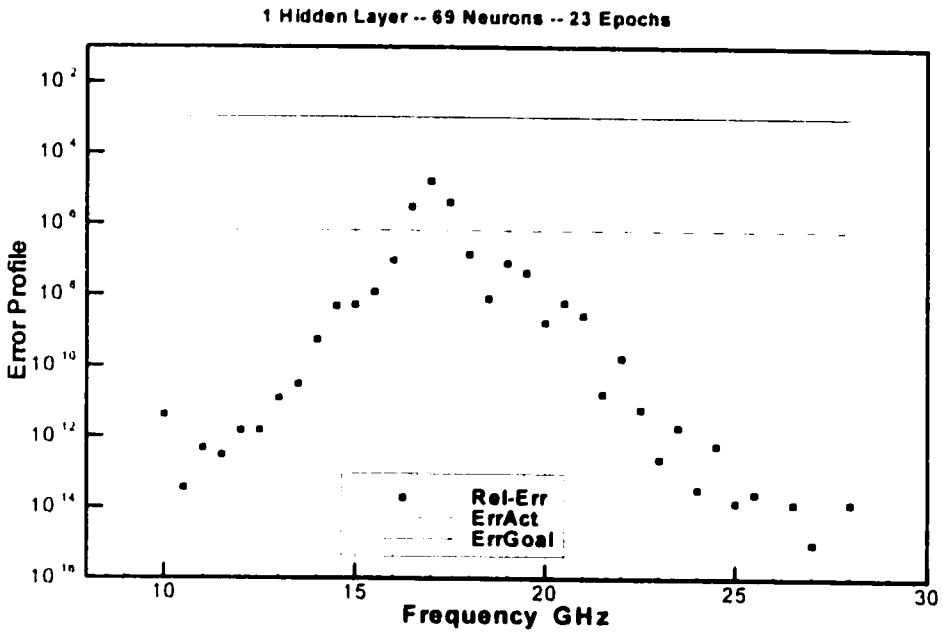


Figure 7.9 Error Performance – 1 Hidden Layer with 69 Neurons

7.2.2 ANN with Two Hidden Layers

In this case, a total of six ANN architectures with two hidden neuron layers were used and once again excellent error performance results were obtained. Again, the non-linear transfer function of the neurons in the hidden layers was *tansig* , while the linear transfer function of the neurons in the output layer was *purelin*, as provided by the Matlab Neural Network Toolbox [65]. Figures 7.10 through 7.15 show the relative error obtained ($\Delta Z/Z$), and the desired error goal and the achieved error by the training process for each architecture. Again, in these Figures, the latter two quantities are indistinguishable. It can also be noted that, training takes longer with two hidden layers.

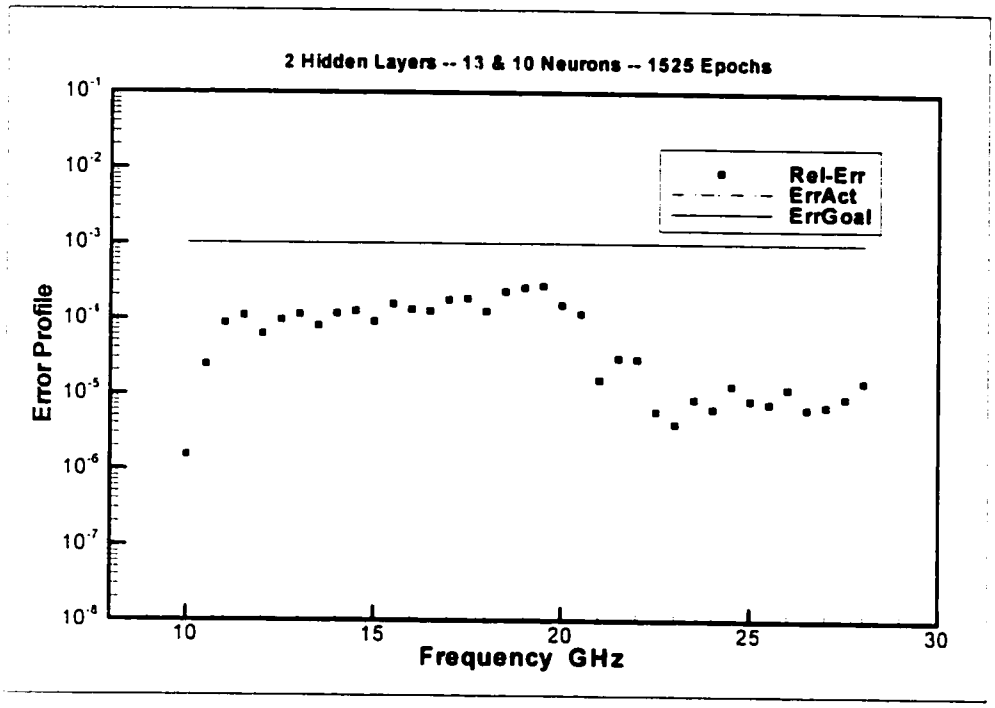


Figure 7.10 Error Performance – 2 Hidden Layers with 13 + 10 Neurons

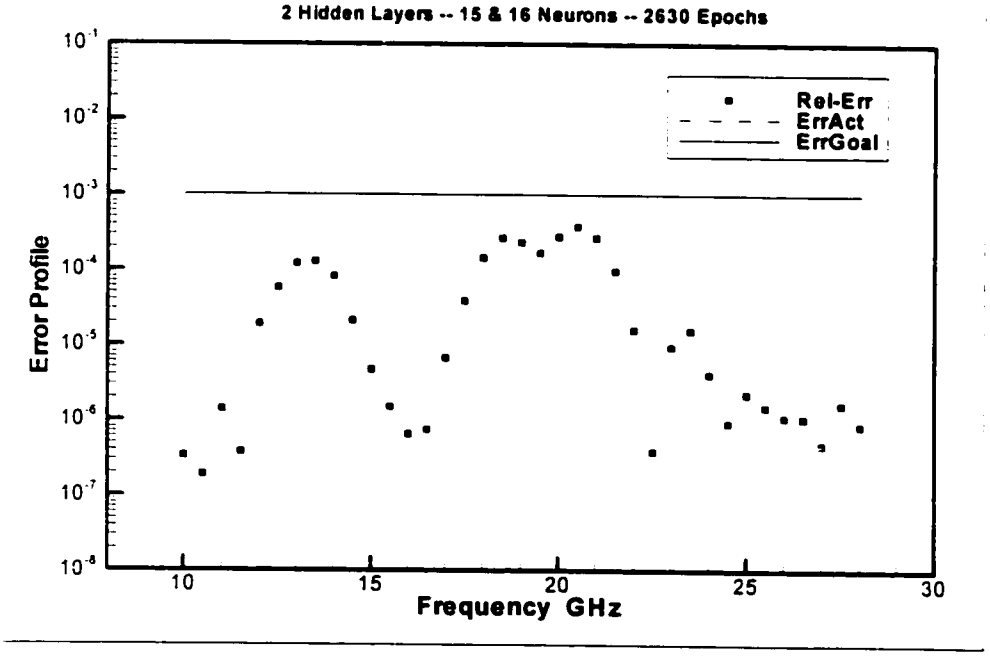


Figure 7.11 Error Performance – 2 Hidden Layers with 15 + 16 Neurons

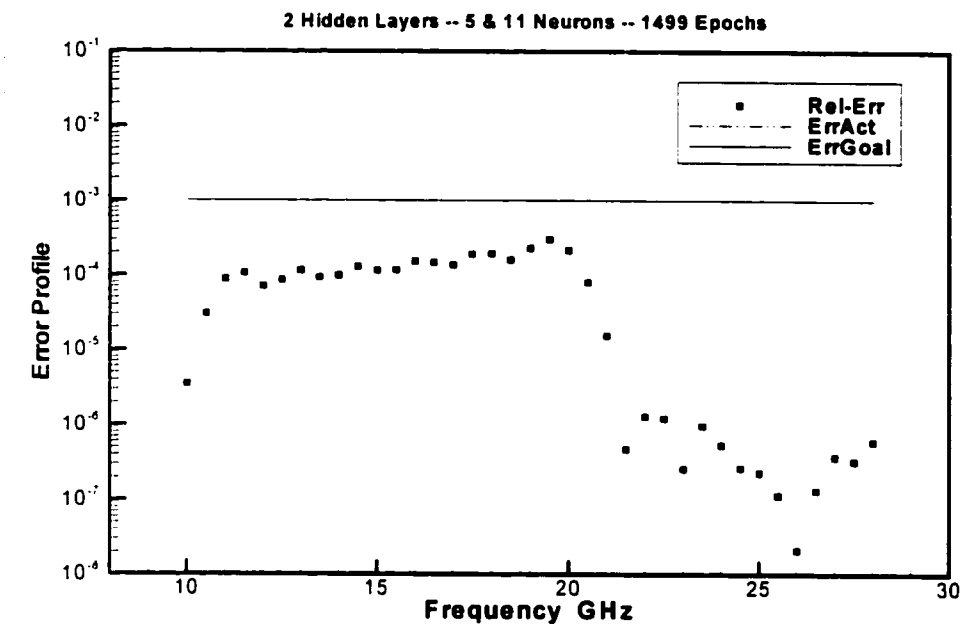


Figure 7.12 Error Performance – 2 Hidden Layers with 5 + 11 Neurons

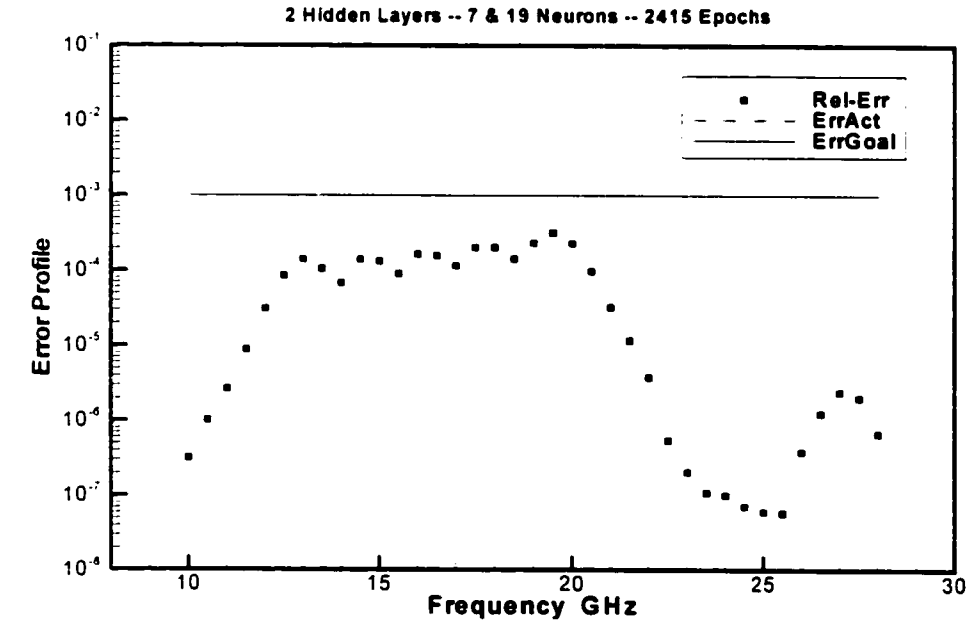


Figure 7.13 Error Performance – 2 Hidden Layers with 7 + 19 Neurons

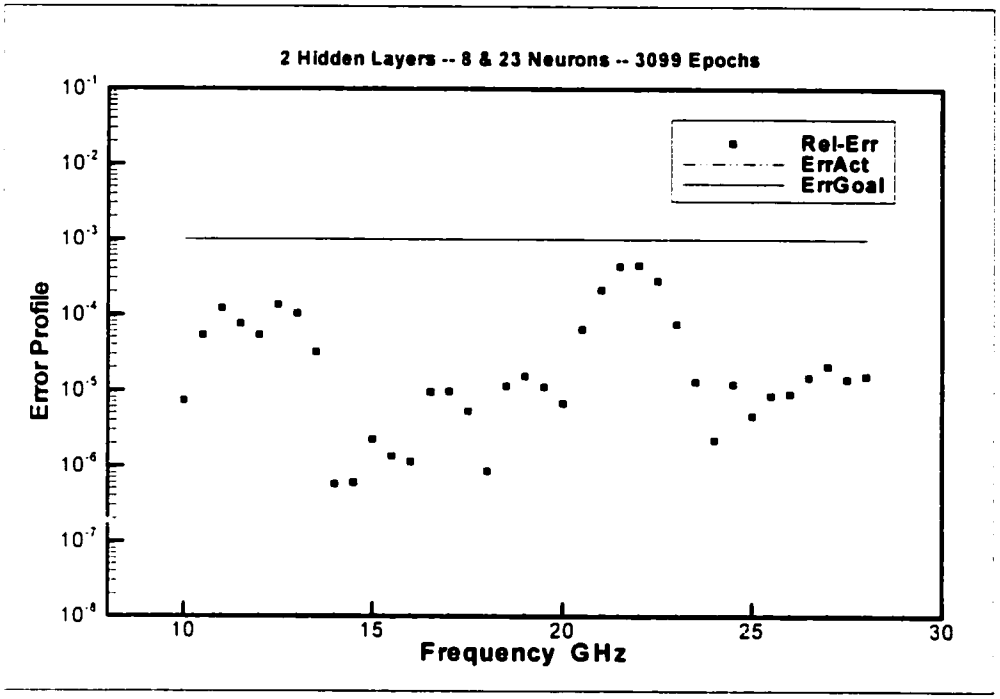


Figure 7.14 Error Performance – 2 Hidden Layers with 8 + 23 Neurons

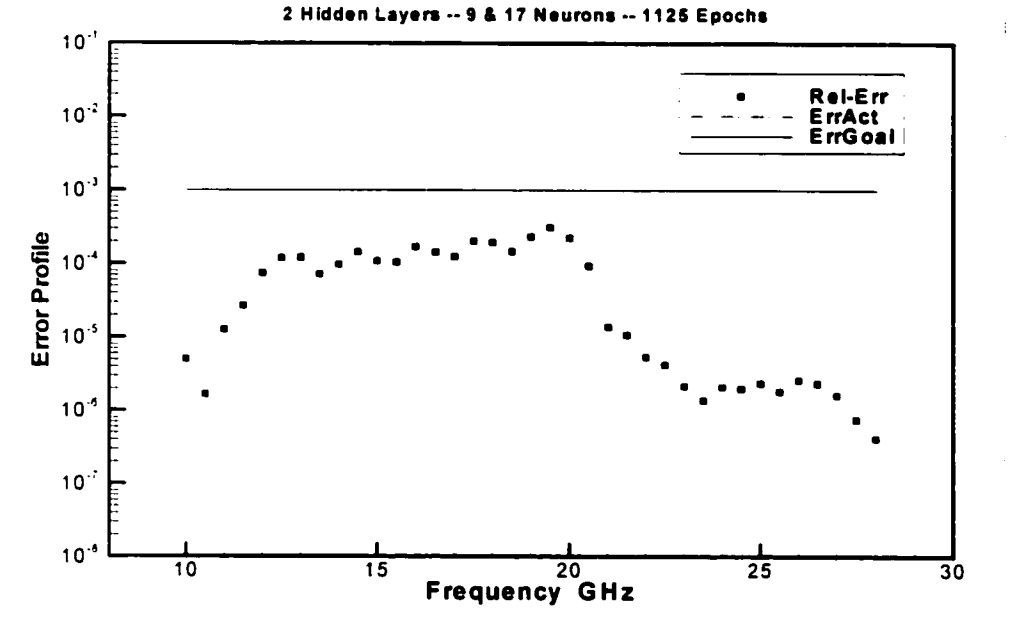


Figure 7.15 Error Performance – 2 Hidden Layers with 9 + 17 Neurons

7.2.3 Error Performance

Table 7.1 below summarizes the error performance of the various ANN architectures used. In all cases, the achieved sum-squared error of the network was lower than the specified sum-squared error goal. Also, in order to carry out a simple comparison between the ANN architectures, the maximum relative error for each architecture is listed. As to which architecture is most suitable, depends on the relative importance of factors such as the total number of neurons in a network, the total number of epochs taken in training the network, the error performance obtained, and so on. In an application such as function approximation, the most critical figure of merit is the relative error performance of the network.

A closer examination of the Figures in Sections 7.2.1 and 7.2.2 shows that the overall error performances of one hidden layer and two hidden layer networks having approximately equal number of neurons, are similar. The numbers in the brackets refer to the number of neurons.

Figure 7.10 (23) is similar to Figure 7.3 (21) and to a lesser degree to Figure 7.2 (13)

Figure 7.11 (31) is similar to Figure 7.4 (34)

Figure 7.12 (16) is similar to Figure 7.2 (13) and to a lesser degree to Figure 7.3 (21)

Figure 7.13 (26) is similar to Figure 7.3 (21) and to a lesser degree to Figure 7.2 (13)

Figure 7.14 (31) is similar to Figure 7.4 (34)

Figure 7.15 (26) is similar to Figure 7.3 (21) and to a lesser degree to Figure 7.2 (13)

Because a one hidden layer network performs just as well as a two hidden layer network, it is preferable on account of its shorter training times, and is used henceforth.

No of Hidden Layers	Neurons Per Hidden Layer	Total No of Neurons	No of Training Epochs	Achieved SSE	Maximum Relative Error
1	13	13	913	9.997E-04	2.79E-04
1	21	21	923	9.991E-04	2.95E-04
1	34	34	60	9.276E-04	4.65E-04
1	45	45	17	1.454E-04	3.32E-04
1	50	50	126	3.452E-06	3.61E-05
1	55	55	117	1.343E-04	1.56E-04
1	60	60	73	3.571E-04	3.60E-04
1	69	69	23	6.215E-07	1.60E-05
2	5 + 11	16	1499	9.993E-04	3.01E-04
2	13 + 10	23	1525	9.995E-04	2.75E-04
2	7 + 19	26	2415	9.997E-04	3.18E-04
2	9 + 17	26	1125	9.993E-04	3.14E-04
2	8 + 23	31	3099	9.963E-04	4.54E-04
2	15 + 16	31	2630	9.991E-04	3.77E-04

Table 7.1 Error Performance Summary of Various ANN Architectures

7.3 Decimation Technique for ANN Training

The concept of decimation tried out in this work, involves training an Artificial Neural Network with progressively larger training data subsets. Decimation has an intuitive quality to it, the expectation being that it could yield a speedier training time, while maintaining similar error performance to the standard (conventional) training method. The training algorithm utilized is the Levenberg-Marquardt algorithm as before.

To implement decimation, the available training data set is subdivided into n data subsets. Thus, for example let n = 3, and let

$$\text{DataSet} = \{ d_1,\ d_2,\ d_3,\ d_4,\ \ldots\ldots,\ d_{17} \}$$

Then,

$$\text{DataSubSet}[1] = \{ d_1,\ d_6,\ d_{10},\ d_{13},\ d_{17} \}$$

$$\text{DataSubSet}[2] = \{ d_4,\ d_8,\ d_{11},\ d_{15} \}$$

$$\text{DataSubSet}[3] = \{ d_2,\ d_3,\ d_5,\ d_7,\ d_9,\ d_{12},\ d_{14},\ d_{16} \}$$

The procedure for training using decimation is as follows. Given:

an ANN architecture with N neurons,

an error goal of EG = ε, where ε is a small number,

initial synaptic weights and biases { W(0) } and { b(0) },

1) The training data sets are prepared as follows:

TrainingSet[1] = {DataSubSet[1]}

TrainingSet[2] = {DataSubSet[1] ∪ DataSubSet[2]}

TrainingSet[3] = {DataSubSet[1] ∪ DataSubSet[2] ∪ DataSubSet[3]} = {DataSet}

2) Initialize ANN with weights and biases { W(0) } and { b(0) }

3) Train ANN with TrainingSet[1] until EG is reached,
the weights and biases are now { W(1) } and { b(1) }

4) Continue training with TrainingSet[2]. Initial weights and biases for this step are { W(1) } and { b(1) }. Stop once EG is reached, the weights and biases are now { W(2) } and { b(2) },

5) Again continue training with TrainingSet[3]. Initial weights and biases for this step are { W(2) } and { b(2) }. Stop once EG is reached, the final weights and biases are now { W(3) } and { b(3) }.

The concept of *training by decimation* proposed here, was investigated by applying it to the various ANN architectures presented in Sections 7.2.1 and 7.2.2 above. In order to validate the concept, the expected outcome for a given architecture was the following criterion:*lower number of training epochs when trained by decimation while retaining an error performance comparable to*

training without decimation.

The available training data set, containing 37 frequencies and corresponding values of characteristic impedance values for the plain (ungrooved) microstrip was subdivided into three subsets as follows:

F_SubSet[1] = {f1, f6, f9, f13, f17, f21, f26, f30, f33, f37},

Z_SubSet[1] = {Z1, Z6, Z9, Z13, Z17, Z21, Z26, Z30, Z33, Z37},

F_SubSet[2] = {f4, f7, f11, f15, f19, f24, f28, f31, f35},

Z_SubSet[2] = {Z4, Z7, Z11, Z15, Z19, Z24, Z28, Z31, Z35},

F_SubSet[3] = {f2, f3, f5, f8, f10, f12, f14, f16, f18, f20, f22, f23,
f25, f27, f29, f32, f34, f36},

Z_SubSet[3] = {Z2, Z3, Z5, Z8, Z10, Z12, Z14, Z16, Z18, Z20, Z22, Z23,
Z25, Z27, Z29, Z32, Z34, Z36},

Then, the decimation training sets were formed as follows:

TS[1] = {F_SubSet[1]; Z_SubSet[1]},

TS[2] = {F_SubSet[1] ∪ F_SubSet[2]; Z_SubSet[1] ∪ Z_SubSet[2]},

TS[3] = {F_SubSet[1] ∪ F_SubSet[2] ∪ F_SubSet[3];
Z_SubSet[1] ∪ Z_SubSet[2] ∪ Z_SubSet[3]}

The starting point for training by decimation is to use the same initialization values of

synaptic weights and biases used in training the ANN the standard way, to enable a proper comparison between training methods. Thus, for each architecture described in Sections 7.2.1 and 7.2.2, the initial weight and bias values were saved, and used again to initialize the ANN for training by decimation. At the completion of the investigation, the following observations were made.

a) Where the number of neurons in an architecture (one hidden layer or two hidden layer) was smaller than or comparable to the number of data points in the full training set TS[3], training by decimation was not successful. Either the initial weights and biases were not suitable to reach the error goal, or the error did not converge to the desired value during the last decimation step.

b) Where the number of neurons were comparable or larger than TS[3], training by decimation was successful, the expected goal was reached, thus validating the concept of *training by decimation*.

Figures 7.16 through 7.20 show the comparative data between standard (conventional) training and training by decimation, for a one hidden layer ANN. It can be seen that *training by decimation* yielded a multi-fold speed up in training with no appreciable deficit in error performance.

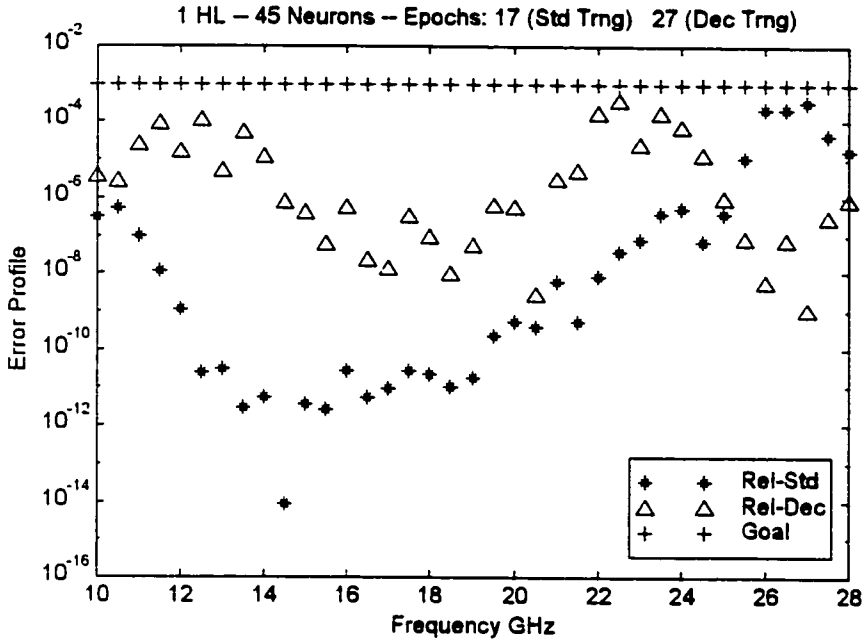


Figure 7.16 Error Performance and Training Comparison – 1 HL – 45 Neurons

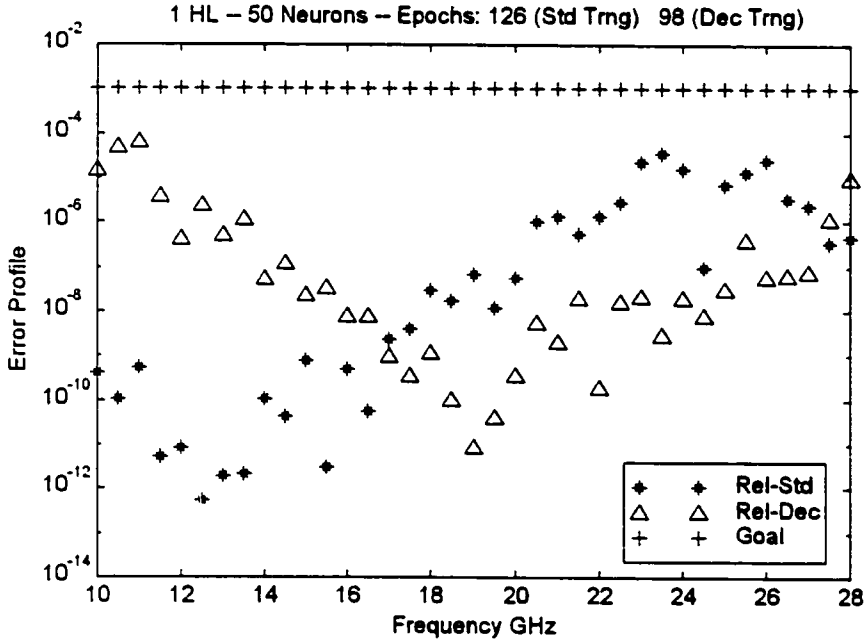


Figure 7.17 Error Performance and Training Comparison – 1 HL – 50 Neurons

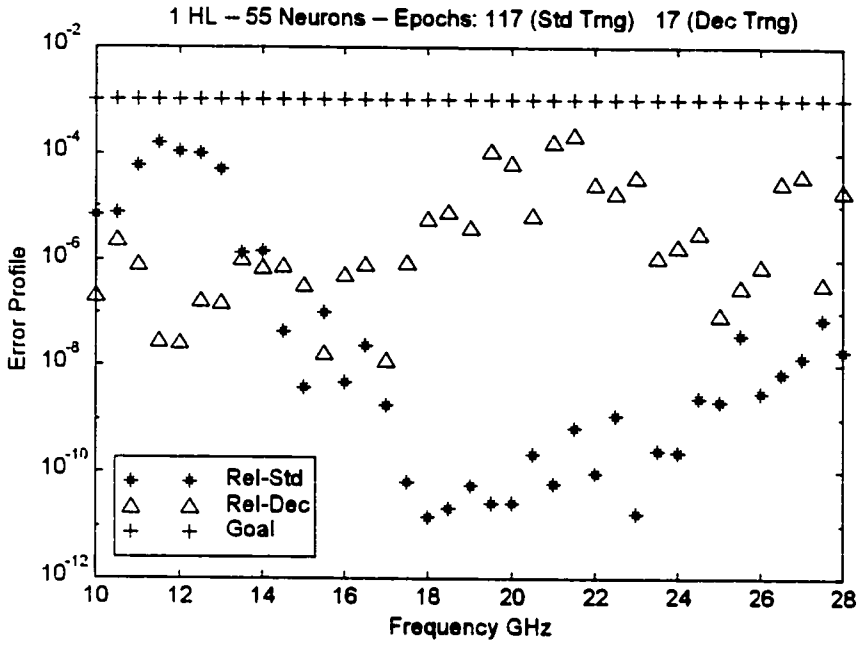


Figure 7.18 Error Performance and Training Comparison – 1 HL – 55 Neurons

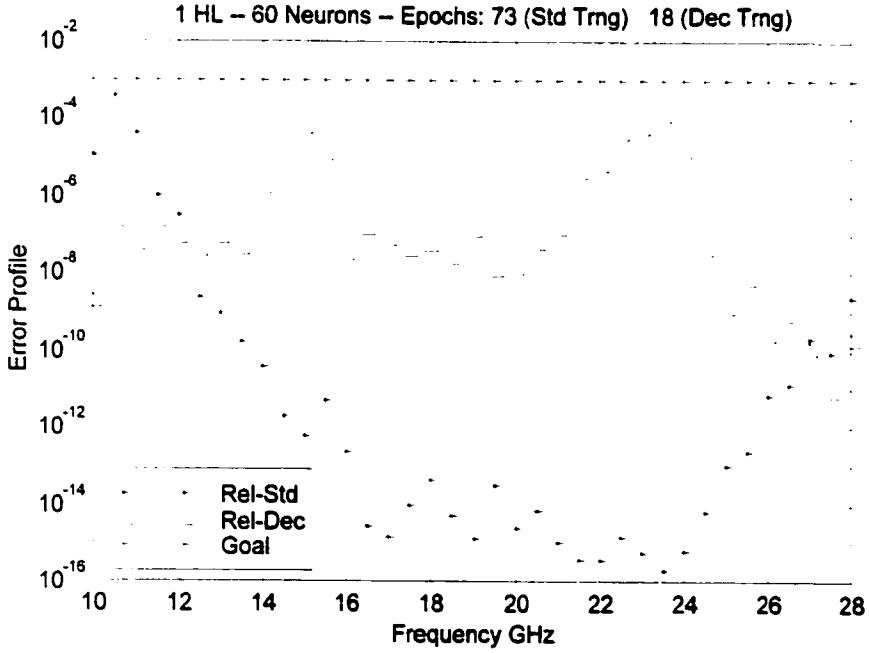


Figure 7.19 Error Performance and Training Comparison – 1 HL – 60 Neurons

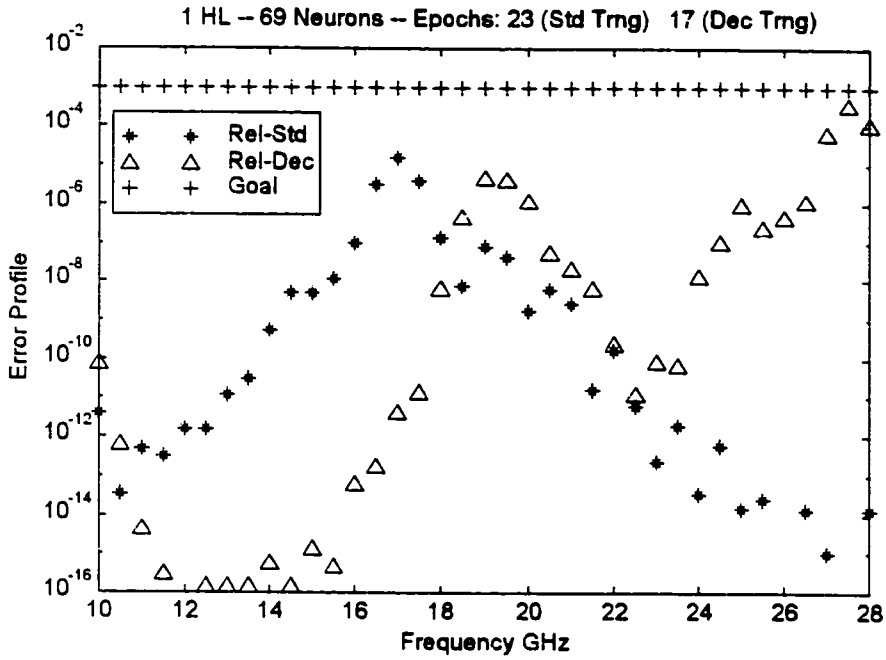


Figure 7.20 Error Performance and Training Comparison – 1 HL – 69 Neurons

7.4 Solution by Decimation: Modelling of Grooved Microstrip

The grooved microstrip was modelled initially by two separate one hidden layer ANN architectures, having 50 and 60 hidden neurons respectively, since these architectures had yielded the best overall error performances for the plain (ungrooved) microstrip case. These networks were trained both by decimation and by the standard way. The resulting error performance obtained by decimation was comparable to that obtained by standard training, and at times, was superior. The networks reached the desired error goal easily, with excellent sum-squared error figures. Nevertheless, the ANN architecture with 60 hidden neurons gave better results compared to the 50 hidden neuron architecture, and it was selected as the vehicle for further modelling.

Figures 7.21 through 7.25 show the error performance comparison for both standard and decimation training methods for the one hidden layer with 60 neurons ANN architecture. The actual comparison of epochs is as follows:

Figure 7.21: Groove Depth 0.015 mm: Standard = 27; Decimation = 18;

Figure 7.22: Groove Depth 0.030 mm: Standard = 1937; Decimation = 30;

Figure 7.23: Groove Depth 0.045 mm: Standard = N/A;* Decimation = 17;

Figure 7.24: Groove Depth 0.060 mm: Standard = 86; Decimation = 24;

Figure 7.25: Groove Depth 0.075 mm: Standard = 34; Decimation = 37;

* Standard training was not successful.

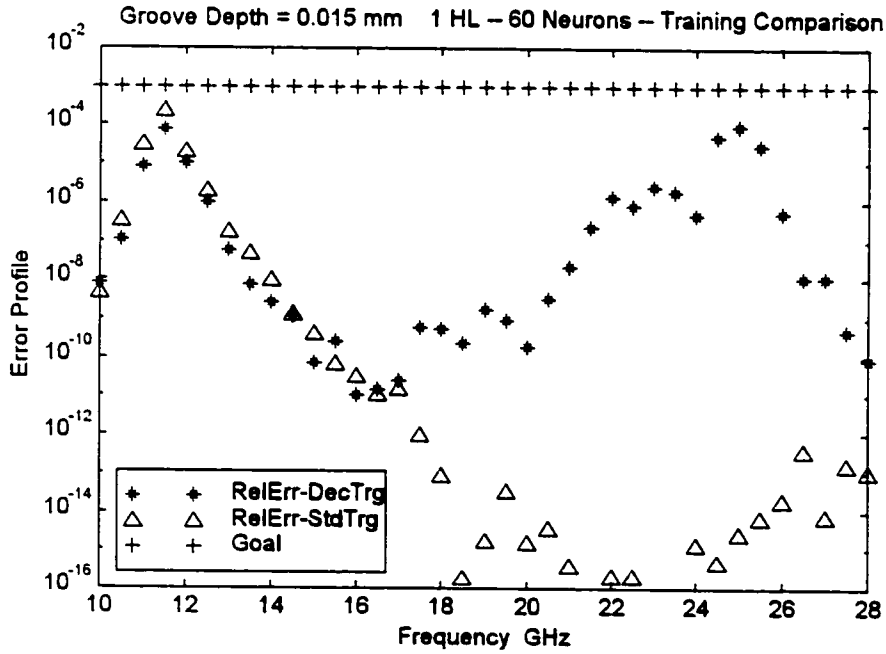


Figure 7.21 Error Performance and Training Comparison – 60 Neurons – 0.015 mm

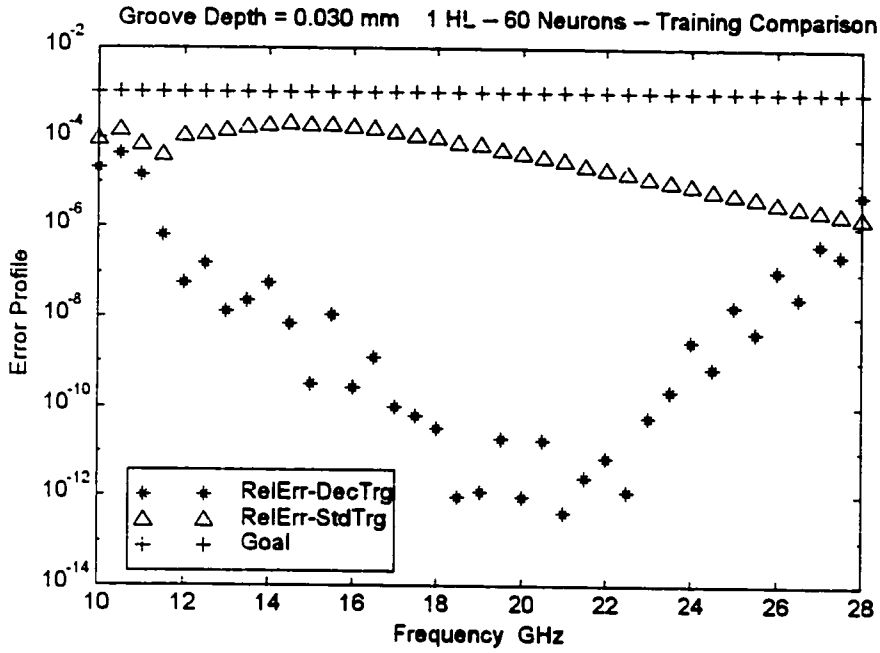


Figure 7.22 Error Performance and Training Comparison – 60 Neurons – 0.030 mm

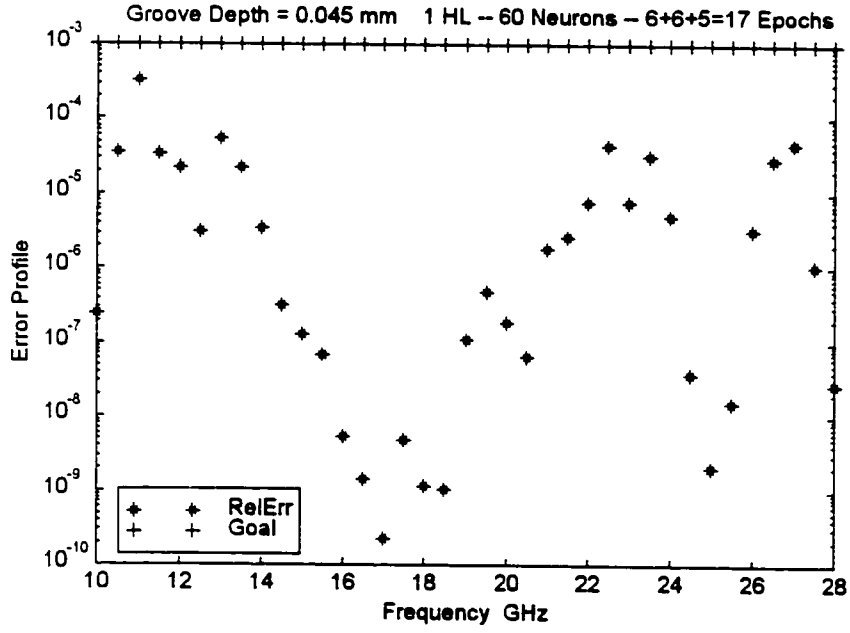


Figure 7.23 Error Performance (Decimation) – 60 Neurons – 0.045 mm

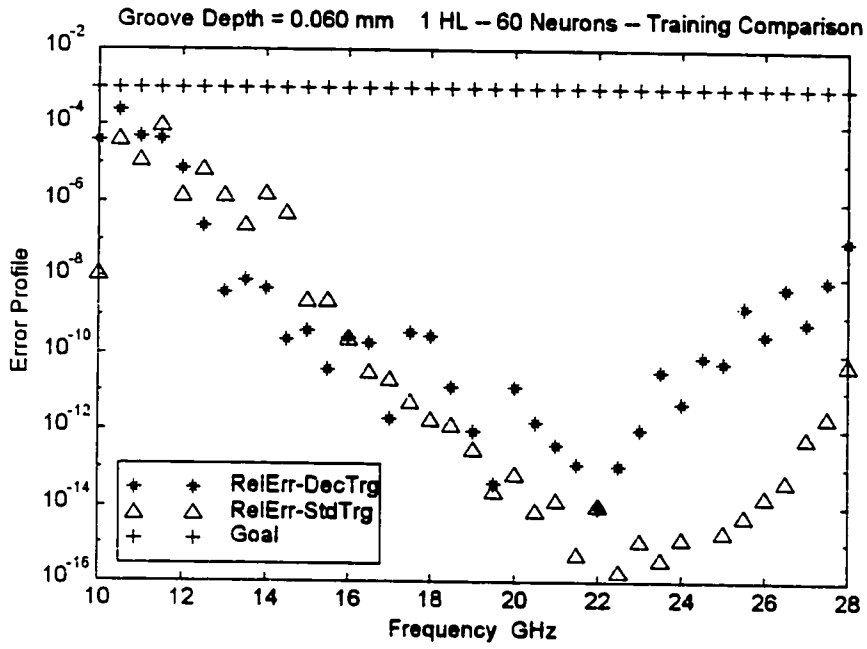


Figure 7.24 Error Performance and Training Comparison – 60 Neurons – 0.060 mm

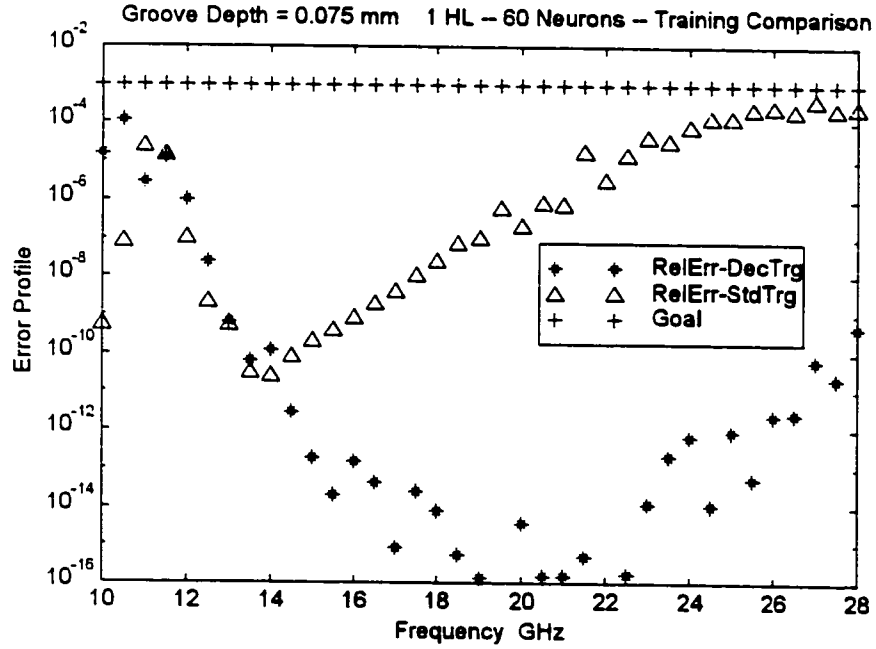


Figure 7.25 Error Performance and Training Comparison – 60 Neurons – 0.075 mm

7.5 Evaluation of Modelling with Artificial Neural Networks

The investigation of Artificial Neural Network (ANN) modelling technique proved fruitful as an alternative and even complementary to the theoretical/numerical solution technique. While ANNs are not a substitute to a numerical method such as VFEM, their advantage lies in the rapid simulation of a given problem with constant geometry almost instantaneously. ANNs are also able to model (capture behaviour) natural phenomena such as electromagnetic fields based on experimental measurement samples of these phenomena. This is quite useful when these phenomena cannot be modelled analytically. The data obtained by numerical means (VFEM or other numerical method) or by experimental means can be used in training the ANN. Once the ANN is trained, it can be used to save considerable time to obtain any desired output for a given input for that specified problem. The modelling process is described schematically in Chapter 1, Figure 1.1, and in Figure 7.1 in this Chapter.

The benefits obtained by the *training by decimation* method investigated in this thesis, and the selection of the particular ANN architecture can be summarized as follows:

a) first of all, the training of the network was in terms of the frequency, the naturally variable quantity, while the groove depth is a naturally constant parameter. A training method (conventional or decimation) based on groove depth as a variable does not make practical sense as there were a total of 14 groove depths as opposed to 37 frequencies;

b) for the case of the plain microstrip (groove depth = 0), the one hidden layer with 60 neuron ANN architecture was the better of all the architectures, in both overall error performance, and in lowest maximum relative error value, for both modes of training. The training epoch ratio for standard vs decimation was greater than one, i.e. 73/18;

c) for the case of grooved microstrip, it was modelled with the same architecture as in b), over various groove depths. Training by decimation showed its advantage over the standard (conventional) mode of training, as it produced lower number of training epochs, while maintaining the error performance fidelity. As mentioned in Section 7.4, the training epoch ratio for standard vs decimation was greater than one, for almost all groove depths investigated.

The performance of the trained ANN was tested with various new input values of frequency and groove depth, different from those in the training data set, yet within the range of defined frequencies and groove depths for the microstrip problem. The ANN test output values for the characteristic impedance were comparable to those obtained from the VFEM solution, the errors being within the set criterion.

The results outlined above, indicate the successful outcome of this ANN modelling technique. Whether trained by the standard way or by decimation, the ANNs definitely yielded accurate data which could be used with confidence.

Finally, *training by decimation* showed an unexpected quality, in that, in some cases where the network trained conventionally was unable to reach the error goal during training, training by

decimation was able to do so. As an example, this occurred for the case of a groove depth of 0.045 mm, for both 50 and 60 hidden neuron networks.

It can be concluded from the contents of this Chapter that, both the modelling of electromagnetic problems by means of ANNs and, the concept of *training by decimation* were proven successful in this work.

CHAPTER 8

CONCLUSIONS

8.1 Contribution of Work

An appraisal of the work carried out in developing a real-time solution prototype for electromagnetic problems, indicates that the following achievements were realized.

a) a detailed analysis of the VFEM zone discretization and the identification and definition of the critical relationships with the associated solution matrix equations, many of them original, such as fill-in laws and fill-in prediction methods, both graphical and algebraic, for edges contributing to fill-ins ;

b) a thorough, and as far as is known for the first time, analysis of the matrix structures and the matrix-VFEM zone relationships, and the discovery of their many key attributes, and their impact on the computational aspects;

c) design and implementation of several accelerated matrix solution models, and their

successful application to various problems;

d) adoption and successful application of the VFEM methodology together with the use of the above-mentioned concepts to several electromagnetic problems, and the obtained excellent computational performance results, especially in comparison with results obtained elsewhere;

e) successful modelling of electromagnetic problems by means of Artificial Neural Networks, and the implementation of a rapid solution prototype having excellent error performance, when the Artificial Neural Network is used in conjunction with computational electromagnetics;

f) inception and successful demonstration of the concept of *training by decimation* of the Artificial Neural Networks and the achieved speed-up in training while maintaining the integrity of the error performance.

Further details on contributions and findings are summarized at the end of Chapters 3, 4, 5, and 7.

8.2 Future Work

There are always improvements that can be obtained by further work, and one of the areas that need further investigations in numerical techniques is the matrix solution acceleration, to build

upon the performance obtained during this work. Some of this work lies in computer coding approaches, while others involve algorithm design. In the area of Artificial Neural Networks, the development of a methodology to optimize the initialisation so that the training process will follow a minimum energy trajectory on the error topography is very important, and has great promise. Next, the concept of *training by decimation* requires further research in order to identify the relationship between data size and network architecture. Finally, efforts are needed to determine the cause of the difficulty a network has, at times, in reaching the error goal during the final step in decimation (extreme shallow descent), and in the latter phases of standard training, where 95% of the epochs occur in the final 5% of the error goal.

BIBLIOGRAPHY

[1] B.M.A. Rahman, J.B. Davies
"Finite-Element Analysis of Optical and Microwave Waveguide Problems",
IEEE Transactions on Microwave Theory and Techniques, Vol 32, No. 1, Jan 1984

[2] Z. Pantic, R. Mittra
"Quasi-TEM Analysis of Microwave Transmission Lines by the Finite-Element Method"
IEEE Transactions on Microwave Theory and Techniques, Vol 34, No. 11, Nov 1986

[3] M.L. Barton, Z.J. Cendes
"New Vector Finite Elements for Three-Dimensional Magnetic Field Computation"
J. Appl. Phys. 61 (8), 15 Apr 1987

[4] G.I. Costache
"Finite Element Method Applied to Skin-Effect Problems in Strip Transmission Lines"
IEEE Transactions on Microwave Theory and Techniques, Vol 35, No. 11, Nov 1987

[5] R.L. Khan, G.I. Costache
"Finite Element Method Applied to Modeling Crosstalk Problems on Printed Circuit Boards"
IEEE Transactions on Electromagnetic Compatibility, Vol 31, No. 1, Feb 1989

[6] A. Bossavit
"Simplicial Finite Elements for Scattering Problems in Electromagnetism"
Computer Methods in Applied Mechanics and Engineering, Vol 76, No. 3, 1989

[7] A. Bossavit, I. Mayergoyz
"Edge-Elements for Scattering Problems"
IEEE Transactions on Magnetics, Vol 25, No. 4, Jul 1989

[8] E. Yamashita, (Editor)
"Analysis Methods for Electromagnetic Wave Problems"
Artech House, 1990

[9] J. D'Angelo, I.D. Mayergoyz
"Finite Element Methods for the Solution of RF Radiation and Scattering Problems"
Electromagnetics, Vol 10, 1990

[10] D.N. Ladd, G.I. Costache
"Computation of the Electromagnetic Field Distribution Inside Enclosures with Apertures Using 3-D FEM"
International Journal of Numerical Modelling, Vol 4, 1991

[11] J-F. Lee, D-K. Sun, Z.J. Cendes
"Full-Wave Analysis of Dielectric Waveguides Using Tangential Vector Finite Elements"
IEEE Transactions on Microwave Theory and Techniques, Vol 39, No. 8, Aug 1991

[12] J. D'Angelo, I.D. Mayergoyz
"Three Dimensional RF Scattering by the Finite Element Method"
IEEE Transactions on Magnetics, Vol 27, No. 5, Sep 1991

[13] B.M.A Rahman, F.A. Fernandez, J.B. Davies
"Review of Finite Element Methods for Microwave and Optical Waveguides"
Proceedings of the IEEE, Vol 79, No. 10, Oct 1991

[14] M. Koshiba, K. Inoue
"Simple and Efficient Finite-Element Analysis of Microwave and Optical Waveguides"
IEEE Transactions on Microwave Theory and Techniques, February 1992

[15] G. Mur
"The Finite Element Modeling of Three-Dimensional Time-Domain Electromagnetic Fields in Strongly Inhomogeneous Media"
IEEE Transactions on Magnetics, Vol 28, No. 2, Mar 1992

[16] J-F. Lee, R. Mittra
"A Note on the Application of Edge-Elements for Modeling Three-Dimensional Inhomogeneously-Filled Cavities"
IEEE Transactions on Microwave Theory and Techniques, Vol 40, No. 9, Sep 1992

[17] A. Chatterjee, J.M. Jin, J.L. Volakis
"Computation of Cavity Resonances Using Edge-Based Finite Elements"
IEEE Transactions on Microwave Theory and Techniques, Vol 40, No. 11, Nov 1992

[18] J. Jin
"The Finite Element Method in Electromagnetics"
John Wiley & Sons, Inc, 1993

[19] R.L. Ferrari, R.L. Naidu
"Finite-Element Modelling of High-Frequency Electromagnetic Problems with Material Discontinuities"
IEE Proceedings A, Vol 137, Nov 1990

[20] H. O. Ali Doctoral Dissertation, 1997
"Near Field Computations Using the Finite Element Model"
University of Ottawa, Ottawa, Ontario, Canada

[21] A. Chatterjee, J.M. Jin, J.L. Volakis
"Edge-Based Finite Elements and Vector ABCs Applied to 3-D Scattering"
IEEE Transactions on Antennas and Propagation, Vol 41, No. 2, Feb 1993

[22] M.V.K. Chari, G. Bedrosian, J. D'Angelo, A. Konrad
"Finite Element Applications in Electrical Engineering"
IEEE Transactions on Magnetics, Vol 29, No. 2, Mar 1993

[23] H.O. Ali, G. Costache
"Finite-Element Time-Domain Analysis of Axisymmetrical Radiators"
IEEE Transactions on Antennas and Propagation, Vol 42, No. 2, Feb 1994

[24] A.F. Peterson
"Vector Finite Element Formulation for Scattering from Two-Dimensional Heterogeneous Bodies"
IEEE Transactions on Antennas and Propagation, Vol 42, No. 3, Mar 1994

[25] W. Sun, C.A. Balanis
"Edge-Based FEM Solution of Scattering from Inhomogeneous and Anisotropic Objects"
IEEE Transactions on Antennas and Propagation, Vol 42, No. 5, May 1994

[26] J-F. Lee
"Finite Element Analysis of Lossy Dielectric Waveguides"
IEEE Transactions on Microwave Theory and Techniques, Vol 42, No. 6, Jun 1994

[27] Z.L. Zhu, J.B. Davies, F.A. Fernandez
"3-D Edge Element Analysis of Dielectric Loaded Resonant Cavities"
International Journal of Numerical Modelling, Vol 7, 1994

[28] S.T. Clegg, K.A. Murphy, W.T. Joines, G. Rine, T.V. Samulski
"Finite Element Computation of Electromagnetic Fields"
IEEE Transactions on Microwave Theory and Techniques, Vol 42, No. 10, Oct 1994

[29] C. J. Reddy, M. D. Deshpande, C. R. Cockrell, F. B. Beck
"Finite Element Method for Eigenvalue Problems in Electromagnetics"
NASA Technical Paper 3485, December 1994

[30] *Electromagnetics* – Special Issue Vol 16, No 4, 1996

[31] J-Y. Wu, R. Lee
"The Advantages of Triangular and Tetrahedral Edge Elements for Electromagnetic Modeling with the Finite-Element Method"
IEEE Transactions on Antennas and Propagation, Vol 45, No. 9, Sep 1997

[32] M.S. Al Salameh, M.R. Owais
"Full-Wave Solutions to the Propagation Characteristics of Buried Microstrip Lines by VFEM and ABCs"
International Journal of Numerical Modelling, Vol 11, 1998

[33] I.S. Duff
"A Survey of Sparse Matrix Research"
Proceedings of the IEEE, Vol 65, No 4, April 1977

[34] J. Vlach, K.Singhal
"Computer Methods for Circuit Analysis and Design"
Van Nostrand Reinhold, 1983

[35] R. E. Collin
"Foundations For Microwave Engineering" 2nd Edition
McGraw-Hill, Inc, 1992

[36] T. C. Edwards
"Foundations for Microstrip Circuit Design"
John Wiley & Sons, Inc, 1981

[37] K. C. Gupta, R. Garg, I. J. Bahl
"Microstrip Lines and Slotlines"
Artech House, 1979

[38] S. Ramo, J. R. Whinnery, T, Van Duzer
"Fields and Waves in Communications Engineering" 2nd Edition
John Wiley & Sons, Inc, 1984

[39] K. Li, K. Atsuki
"Low Conductor Loss Microstrip Lines with Side-Grooves"
IEEE MTT-S Digest, 1996

[40] J.A.M. Svedin
"A Modified Finite Element Method for Dielectric Waveguides Using an Asymptotically Correct Approximation on Infinite Elements"
IEEE Transactions on Microwave Theory and Techniques, Vol 39, No. 2, Feb 1991

[41] J-P. Berenger
"A Perfectly Matched Layer for the Absorption of Electromagnetic Waves"
Journal of Computational Physics, Vol 114, 1994

[42] J-Y. Wu, D.M. Kingsland, J-F. Lee, R. Lee
"A Comparison of Anisotropic PML to Berenger's PML and Its Application to the Finite-Element Method for EM Scattering"
IEEE Transactions on Antennas and Propagation, Vol 45, No. 1, Jan 1997

[43] M.A. Morgan, C.H. Chen, S.C. Hill, P.W. Barber
"Finite Element-Boundary Integral Formulation for Electromagnetic Scattering"
Wave Motion 6, 1984

[44] K.C. Paulsen, D.R. Lynch, J.W. Strohbehn
"Three-Dimensional Finite, Boundary, and Hybrid Element Solutions of the Maxwell Equations for Lossy Dielectric Media"
IEEE Transactions on Microwave Theory and Techniques, Vol 36, No. 4, Apr 1988

[45] T. Onuki
"Hybrid Finite Element and Boundary Element Method Applied to Electromagnetic Problems"
IEEE Transactions on Magnetics, Vol 26, No. 2, Mar 1990

[46] X. Yuan, D.R. Lynch, J.W. Strohbehn
"Coupling of Finite Element and Moment Methods for Electromagnetic Scattering from Inhomogeneous Objects"
IEEE Transactions on Antennas and Propagation, Vol 38, No. 3, Mar 1990

[47] X. Yuan
"Three-Dimensional Electromagnetic Scattering from Inhomogeneous Objects by the Hybrid Moment and Finite Element Method"
IEEE Transactions on Microwave Theory and Techniques, Vol 38, No. 8, Aug 1990

[48] J-M. Jin, J.L. Volakis
"A Finite Element-Boundary Integral Formulation for Scattering by Three-Dimensional Cavity-Backed Apertures"
IEEE Transactions on Antennas and Propagation, Vol 39, No. 1, Jan 1991

[49] C-H. Ahn, B-S. Jeong, S-Y Lee
"Efficient Hybrid Finite Element-Boundary Element Method for 3-Dimensional Open-Boundary Field Problems"
IEEE Transactions on Magnetics, Vol 27, No. 5, Sep 1991

[50] J-M. Jin, J.L. Volakis
"A Hybrid Finite Element Method for Scattering and Radiation by Microstrip Patch Antennas and Arrays Residing in a Cavity"
IEEE Transactions on Antennas and Propagation, Vol 39, No. 11, Nov 1991

[51] M. S. H. Tharf, Doctoral Dissertation, 1993
"Computer Modelling of Electromagnetic Interference, Radiation, and Cross-Talk in Electronic Systems"
University of Ottawa, Ottawa, Ontario, Canada

[52] S. Wakao, T. Onuki
"Electromagnetic Field Computations by the Hybrid FE-BE Method Using Edge Elements"
IEEE Transactions on Magnetics, Vol 29, No. 2, Mar 1993

[53] J-F. Lee, G.M. Wilkins, R. Mittra
"Finite-Element Analysis of Axisymmetric Cavity Resonator Using a Hybrid Edge Element Technique"
IEEE Transactions on Microwave Theory and Techniques, Vol 41, No. 11, Nov 1993

[54] M. S. Tharf, G. I. Costache
"Finite Element Method Solutions of Field Distribution in Large Cavities"
International Journal of Numerical Modelling, 1994

[55] C.J. Reddy, M.D. Deshpande, C.R. Cockrell, F.B. Beck
"Radiation Characteristics of Cavity Backed Aperture Antennas in Finite Ground Plane Using the Hybrid FEM/MoM Technique and Geometrical Theory of Diffraction"
IEEE Transactions on Antennas and Propagation, Vol 44, No. 10, Oct 1996

[56] X-Q. Sheng, J-M. Jin, J. Song, C-C. Lu, W.C. Chew
"On the Formulation of Hybrid Finite-Element and Boundary-Integral Methods for 3-D Scattering"
IEEE Transactions on Antennas and Propagation, Vol 46, No. 3, Mar 1998

[57] I. Ratner, H. Ali, E. Petriu, G. Eatherly
"Neural Network Modelling of Electromagnetic Field Problems"
Proceedings NICROSP'96, International Workshop on Neural Networks for Identification, Control, Robotics, and Signal/Image Processing, Venice, Italy
IEEE Computer Society Press, August 1996

[58] G.L. Creech, B.J. Paul, C.D. Lesniak, T.J. Jenkins, M.C. Calcatera
"Artificial Neural Networks for Fast and Accurate EM-CAD of Microwave Circuits"
IEEE Transactions on Microwave Theory and Techniques, Vol 45, No. 5, May 1997

[59] A. Veluswami, M.S. Nakhla, Q-J. Zhang
"The Application of Neural Networks to EM-Based Simulation and Optimization of Interconnects in High-Speed VLSI Circuits"
IEEE Transactions on Microwave Theory and Techniques, Vol 45, No. 5, May 1997

[60] I. Ratner, H. Ali, E. M. Petriu
"Neural Network Simulation of a Dielectric Ring Resonator Antenna"
Journal of Systems Architecture, Issue 44, Elsevier 1998

[61] A.H. El Zooghby, C.G. Cristodoulou, M. Georgiopoulos
"Neural Network-Based Adaptive Beamforming for One- and Two-Dimensional Antenna Arrays"
IEEE Transactions on Antennas and Propagation, Vol 46, No. 12, Dec 1998

[62] R.K. Mishra, A. Patnaik
"Neural Network-Based CAD Model for the Design of Square-Patch Antennas"
IEEE Transactions on Antennas and Propagation, Vol 46, No. 12, Dec 1998

[63] G. Antonini, S. Cristina
"Performance Analysis of Electromagnetic Shields Using Neural Networks"
Transactions, EMC Zurich, 1999

[64] M. D'Amore, M.S. Sarto
"Diagnostics of Signal Transmission Lines by Using A Neural Network Based Monitoring of the Radiated Emission"
Transactions, EMC Zurich, 1999

[65] H. Demuth, M. Beale
"MATLAB Neural Network Toolbox User's Guide, Version 2"
The MathWorks, Inc. Jan 1994

[66] N.B. Karayiannis, A.N. Venetsanopoulos
"Artificial Neural Networks"
Klower Academic Publishers, 1993

[67] W. McCulloch, W. Pitts
"A Logical Calculus of the Ideas Immanent in Nervous Activity"
Bulletin of Mathematical Biophysics, Vol 5, 1943

[68] D.O. Hebb
"The Organization of Behaviour"
John Wiley & Sons, Inc, 1949

[69] J. Von Neumann
"Probabilistic Logics and the Synthesis of Reliable Organisms from Unreliable Components"
Automata Studies, (C.E. Shannon, Ed.), Princeton University Press, 1956

[70] F. Rosenblatt
"The Perceptron: A Probabilistic Model for Information Storage and Organization in the Brain"
Psychological Review, Vol 65, 1958

[71] B. Widrow, M.E. Hoff
"Adaptive Switching Circuits"
IRE WESCON Convention Record, 1960

[72] F. Rosenblatt
"Principles of Neurodynamics"
Spartan Press, 1961

[73] M. Minski, S. Papert
"Perceptrons"
MIT Press, 1969

[74] J.S. Albus
"A Theory of Cerebellar Function"
Mathematical Biosciences, Vol 10, 1971

[75] T. Kohonen
"Correlation Matrix Memories"
IEEE Transactions on Computing, Vol 21, 1972

[76] J.A. Anderson
"A Simple Neural Network Generating an Interactive Memory"
Mathematical Biosciences, Vol 14, 1972

[77] S. Grossberg
"Adaptive Pattern Classification and Universal Recording: I. Parallel Development and Coding of Neural Feature Detectors"
Biological Cybernetics, Vol 23, 1976

[78] J.J. More
"The Levenberg-Marquardt Algorithm: Implementation and Theory"
Numerical Analysis, Springer-Verlag, 1977

[79] K. Fukushima, S. Miyake, T. Ito
"Neocognitron: A Neural Network Model for a Mechanism of Visual Pattern Recognition"
IEEE Transactions on Systems, Man and Cybernetics, Vol 13, No 5, 1983

[80] M.A. Cohen, S. Grossberg
"Absolute Stability of Global Pattern Formation and Parallel Memory Storage by Competitive Neural Networks"
IEEE Transactions on Systems, Man and Cybernetics, Vol 13, 1983

[81] B. Widrow, S.D. Sterns
"Adaptive Signal Processing"
Prentice-Hall, Inc., 1985

[82] D.E. Rumelhart, G.E. Hinton, R.J. Williams
"Learning Internal Representations by Error Propagation"
Parallel Distributed Processing, Vol 1, Ch. 8, MIT Press, 1986

[83] D.W. Tankard, J.J. Hopefield
"Simple 'Neural' Optimization Networks: An A/D Converter, Signal Decision Circuit, and a Linear Programming Circuit"
IEEE Transactions on Circuits and Systems, Vol 33, No 5, 1986

[84] M.J.D. Powell
"Radial Basis Functions for Multivariable Interpolation"
Algorithms for the Approximation of Functions and Data, Clarendon Press, 1987

[85] G.A. Carpenter, S. Grossberg
"ART2: Self-Organizing of Stable Category Recognition Codes for Analog Input Patterns"
Applied Optics, Vol 26, No 23, 1987

[86] T. Kohonen
"Self-Organization and Associative Memory" 2nd Edition
Springer-Verlag, 1987

[87] B. Widrow, M.A. Lehr
"30 Years of Adaptive Neural Networks: Perceptron, Madaline, and Backpropagation"
Proceedings of the IEEE, Sept 1990

[88] J.L. Elman
"Finding Structure in Time"
Cognitive Science, Vol 14, 1990

[89] M.T. Hagan, M. Menhaj
"Training Feedforward Networks with the Marquardt Algorithm"
IEEE Transactions on Neural Networks, Vol 5, No 6, 1994

[90] C. Citterio, A. Pelagotti, V. Piuri, L. Rocca
"Function Approximation – A Fast-Convergence Neural Approach Based on Spectral Analysis"
IEEE Transactions on Neural Networks, Vol 10, No 4, July 1999

[91] K. Levenberg
"A Method for the Solution of Certain Problems in Least Squares"
Quart. Appl. Math, Vol 2, 1944

[92] D. Marquardt
"An Algorithm for Least-Squares Estimation of Nonlinear Parameters"
SIAM J. Appl. Math., Vol11, 1963

[93] J.E. Dennis
"Nonlinear Least Squares"
State of the Art in Numerical Analysis, Springer, 1977

[94] P.E. Gill, W. Murray, M.H. Wright
"Practical Optimization"
Academic Press, 1981

[95] R. Fletcher
"Practical Methods of Optimization, Volume 1"
John Wiley & Sons, 1980

[96] A. Grace
"Optimization Toolbox"
The Mathworks, Inc, 1993

[97] S. Haykin
"Neural Networks: A Comprehensive Foundation"
Macmillan, 1994

[98] P.J.W. Melsa
"Neural Network: A Conceptual Overview"
Technical Report TRC-89-08,
Tellabs Research Center, 1989