A TRANSIENT SOLVER FOR CURRENT DENSITY IN THIN CONDUCTORS FOR MAGNETQUASIOSTATIC CONDITIONS

by

TODD H. PETERSEN

B.S., Kansas State University, 2000
M.S., Kansas State University, 2003

AN ABSTRACT OF A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Electical and Computer Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2009

Abstract

A computer simulation of transient current density distributions in thin conductors was developed using a time-stepped implementation of the integral equation method on a finite element mesh. A study of current distributions in thin conductors was carried out using AC analysis methods. The study of the AC current density distributions was used to develop a circuit theory model for the thin conductor which was then used to determine the nature of its transient response. This model was used to support the design and evaluation of the transient current density solver.

A circuit model for strip lines was made using the Partial Inductance Method to allow for simulations with the `SPICE` circuit solver. Magnetic probes were designed and tested that allow for physical measurements of voltages induced by the magnetic field generated by the current distributions in the strip line. A comparison of the measured voltages to simulated values from `SPICE` was done to validate the `SPICE` model. This model was used to validate the finite-integration model for the same strip line.

Formulation of the transient current density distribution problem is accomplished by the superposition of a source current and an eddy current distribution on the same space. The mathematical derivation and implementation of the time-stepping algorithm to the finite element model is explicitly shown for a surface mesh with triangular elements. A `C++` computer program was written to solve for the total current density in a thin conductor by implementing the time-stepping integral formulation.

Evaluation of the finite element implementation was made regarding mesh size. Finite element meshes of increasing node density were simulated for the same structure until a smooth current density distribution profile was observed. The transient current density solver was validated by comparing simulations with AC conduction and transient response simulations of the `SPICE` model. Transient responses are compared for inputs at different frequencies and for varying time steps. This program is applied to thin conductors of irregular shape.

A TRANSIENT SOLVER FOR CURRENT DENSITY IN THIN CONDUCTORS FOR MAGNETOQUASISTATIC CONDITIONS

by

TODD H. PETERSEN

B.S., Kansas State University, 2000
M.S., Kansas State University, 2003

A DISSERTATION

submitted in partial fulfillment of the
requirements for the degree

DOCTOR OF PHILOSOPHY

Department of Electrical and Computer Engineering
College of Engineering

KANSAS STATE UNIVERSITY
Manhattan, Kansas
2009

Approved by:

Major Professor
Kenneth H. Carpenter

Copyright

Abstract

A computer simulation of transient current density distributions in thin conductors was developed using a time-stepped implementation of the integral equation method on a finite element mesh. A study of current distributions in thin conductors was carried out using AC analysis methods. The study of the AC current density distributions was used to develop a circuit theory model for the thin conductor which was then used to determine the nature of its transient response. This model was used to support the design and evaluation of the transient current density solver.

A circuit model for strip lines was made using the Partial Inductance Method to allow for simulations with the `SPICE` circuit solver. Magnetic probes were designed and tested that allow for physical measurements of voltages induced by the magnetic field generated by the current distributions in the strip line. A comparison of the measured voltages to simulated values from `SPICE` was done to validate the `SPICE` model. This model was used to validate the finite-integration model for the same strip line.

Formulation of the transient current density distribution problem is accomplished by the superposition of a source current and an eddy current distribution on the same space. The mathematical derivation and implementation of the time-stepping algorithm to the finite element model is explicitly shown for a surface mesh with triangular elements. A `C++` computer program was written to solve for the total current density in a thin conductor by implementing the time-stepping integral formulation.

Evaluation of the finite element implementation was made regarding mesh size. Finite element meshes of increasing node density were simulated for the same structure until a smooth current density distribution profile was observed. The transient current density solver was validated by comparing simulations with AC conduction and transient response simulations of the `SPICE` model. Transient responses are compared for inputs at different frequencies and for varying time steps. This program is applied to thin conductors of irregular shape.

Table of Contents

List of Figures

List of Tables

List of Symbols

Alphanumeric symbols

Symbol	Definition
$\vec{A}$	Magnetic vector potential (Wb/m)
$\vec{B}$	Magnetic flux density (T)
$\vec{D}$	Electric flux density (C/m^2)
$\vec{E}$	Electric field (V/m)
$\vec{H}$	Magnetic field (A/m)
$\vec{J}$	Current density (A/m^2)
$\vec{r}$	Position vector from origin to point x
t	Time instant
$\mathbb{E}^3$	Three-dimensional oriented Euclidian space

Table 1: Definition of terms and symbols: Alphanumeric Symbols

Greek Symbols

Ω	Bounded open set of $\mathbb{E}^3$
ϕ	Electric scalar potential (V)
σ	Electric conductivity (S/m)
μ	Magnetic permeability (H/m)
μ_0	Magnetic permeability of free space ($= 4\pi \times 10^{-7}$ H/m)
ϵ	Electric permittivity (F/m)
ϵ_0	Electric permittivity of free space ($\approx 8.854 \times 10^{-12}$ F/m)

Table 2: Definition of terms and symbols: Greek Symbols

Abbreviations	
FEM	Finite Element Method
IEM	Integral Equation Method
MQS	Magnetoquasistatic
TLM	Transmission Line Method
FDTD	Finite Difference Time Domain
SOR	Successive Overrelaxation
ϕ_i	Linear Nodal basis function on node i
$\vec{N}_i$	Linear Edge basis function on edge i

Table 3: Definition of terms and symbols: Abbreviations

Operators	
∂	Boundary operator
$\partial_x, \partial_y, \partial_z$	Spacial derivatives
∂_t	Time derivative
∇	Del operator $= [\partial_x \ \partial_y \ \partial_z \ ]^T$
∇V	Gradient of V
$\nabla \times A$	Curl of A
$\nabla \cdot A$	Divergence of A
$[X]$	A matrix X
$\{X\}$	A column vector X
$[X]^T$	Transpose of matrix X
$[X]^{-1}$	Inverse of matrix X

Table 4: Definition of terms and symbols: Operators

Time-stepping eddy current solution terms	
J_0	Source current derived from electrostatic solution
J_e^*	Prediction current
δV	Incremental change in electrical 'eddy' potential
δJ	Correction current
J_T	Total current
X^t	Quantity X at time step t
$X^{t-\tau}$	Quantity X at time step $t - \tau$
Ψ	Free Space Green's Function = $\frac{1}{4\pi\lvert\vec{r}-\vec{r'}\rvert}$
θ	Time step parameter; $\frac{1}{2}$ = Crank-Nicholson scheme; 1 = implicit integration
τ	Time step size, $(t_n - t_{n-1})$
$\sigma(r,t)$	Conductivity at a specific point and time

Table 5: Definition of terms and symbols: Time-stepping solution terms

Acknowledgments

The funding for this research was provided as part of Subcontracts Number B507741, B554637, and B543310, from Lawrence Livermore National Laboratory. Special thanks to Dr. Ronald S. Lee, Dr. Franklin Roeske, Jr., and Chadd M. May for their support as well as the rest of the LLNL staff for their help and hospitality during my summer internships there.

I would also like to thank Dr. Kenneth H. Carpenter for his guidance, patience, and other innumerable contributions as my adviser for this research, as well as my Ph.D. program. I would like to provide specific acknowledgment of his efforts in optimization of my obtuse computer coding, and his implementation of the SOR routine and `LAPACK` library in our research project.

I wish to acknowledge Professor James DeVault for his help in setting up the power amplifiers used in the magnetic probe measuring experiments and for serving on my committee. Thanks are also due to the rest of my doctoral committee, Dr. Amitabha Chakrabarti, Dr. Bill Kuhn, Dr. Ronald S. Lee, and Dr. Charles Moore.

Finally, I wish to thank my friends and family for their help and support during this time in my life.

Chapter 1

Introduction

The impetus of this research is the desire to be able to simulate transient current density distributions in thin conductors of arbitrary shape and size. Additionally, we would like a self-consistant solver capable of calculating the current density allowing for changes in material properties. This research is directed at pulsed power applications, and the effects of fast current pulses to the conductors in question. To accomplish this we first investigate the electrodynamics underpinning the current density and then implement a numerical solution for the governing equations. There are numerous numerical methods used to solve electromagnetic fields problems and existing computer programs which implement these methods. Solving any interesting problem will require significantly dense finite element meshes consisting of thousands of nodes. Any implementation will be limited by available computing resources. In order to reduce the number of computational steps and the memory requirements that are placed on the simulation platform, we want to solve directly for the current density, $\vec{J}$. This lessens the computational overhead of solving fields, then calculating the current density as a secondary solution. We find that in order to satisfy our desire for the transient current density solver which will meet our requirements, a custom program is required.

1.1 Preliminary investigation of current density

It is useful to begin an investigation into transient current distribution studies by first examining DC and AC current conduction. We examine a direct solution for the current density from Maxwell's equations, and look at the skin depth in flat conductors at different frequencies.

1.1.1 Partial differential equation for current density

One approach to solving for the current density in a thin conductor is solve directly for $\vec{J}$ from Maxwell's equations. Starting with writing Maxwell's equations[1] we have

$$\nabla \cdot \vec{D} = \rho_v \tag{1.1}$$

$$\nabla \cdot \vec{B} = 0 \tag{1.2}$$

$$\nabla \times \vec{E} = -\frac{\partial \vec{B}}{\partial t} \tag{1.3}$$

$$\nabla \times \vec{H} = \vec{J} + \frac{\partial \vec{D}}{\partial t}. \tag{1.4}$$

A starting assumption is that we have a constant conductivity and that we are in a magnetoquasistatic[2] case. Magnetoquasistatic * (MQS) approximations can be made when the frequencies are low enough that the time delays due to electromagnetic wave propagation are unimportant to the solution. This means that the distances over which we want to calculate the current distributions are much much shorter than the wavelength of the driving source; therefore we can treat the fields as constant over the region. This approximation is common in eddy current solutions. When we are in an isotropic medium we also have the auxiliary material relationships

$$\vec{D} = \epsilon \vec{E} \tag{1.5}$$

*Referencing the definition by Jackson in Chapter 5.18 of *Classical Electrodynamics*,

> "Quasi-static" refers to the regime for which the finite speed of light can be neglected and fields treated as if they propagated instantaneously. Said in other, equivalent words, it is the regime where the system is small compared with the electromagnetic wavelength associated with the dominant time scale of the problem.

$$\vec{B} = \mu\vec{H} \tag{1.6}$$

$$\vec{J} = \sigma\vec{E}. \tag{1.7}$$

If we are in a good conductor, $\rho_v = 0$, and we can neglect the displacement current. With these conditions we can begin to solve Maxwell's equations in terms of $\vec{J}$ alone for time independent μ, and homogeneous σ and ϵ; eq.(1.3) and eq.(1.4) give

$$\nabla \times (\nabla \times \vec{J}) = -\mu\sigma\frac{\partial}{\partial t}\left(\vec{J} + \frac{\partial(\frac{\epsilon}{\sigma}\vec{J})}{\partial t}\right). \tag{1.8}$$

Rewriting $\nabla \times \nabla \times \vec{J}$

$$\nabla(\nabla \cdot \vec{J}) - \nabla^2\vec{J} = -\sigma\mu\frac{\partial}{\partial t}\vec{J} - \epsilon\mu\frac{\partial^2\vec{J}}{\partial t^2}. \tag{1.9}$$

From eq.(1.1) and eq.(1.7)

$$\nabla \cdot \left(\frac{\epsilon}{\sigma}\vec{J}\right) = 0. \tag{1.10}$$

We can neglect the displacement current term, $\frac{\partial^2\vec{J}}{\partial t^2}$ due to the MQS approximation[1,2]. To determine if we are operating at a frequency low enough where we are in the MQS state, we can compare the conduction and displacement current terms and set them equal. If we treat the material properties as constants (which for high frequencies we know that they are not) we have

$$\epsilon\mu\omega^2\vec{J} = \sigma\mu\omega\vec{J}. \tag{1.11}$$

For relative permittivity of 1, permeability being μ_0, and a good conductor like copper ($\sigma \approx 5.8 \times 10^7$), we find that the terms will be equal when $\omega \approx 6 \times 10^{18}$ Hz. This limit is actually much lower than 10^{18} Hz due to the plasma frequency in metals[1] which is near to the ultraviolet wavelength. We are operating at a much lower frequencies than even optical so the above approximation holds true. Therefore eq.(1.9) reduces to

$$\nabla^2\vec{J} = \mu\sigma\frac{\partial\vec{J}}{\partial t}, \tag{1.12}$$

From the continuity equation for charge and the definition of current

$$\begin{aligned} \oint \vec{J} \cdot d\vec{S} &= -\frac{d}{dt}\iiint \rho_v dv \\ \nabla \cdot \vec{J} &= -\frac{d}{dt}\rho_v, \end{aligned} \tag{1.13}$$

which is valid for nonlinear ϵ and σ. When $\rho_v = 0$ we have $\nabla \cdot \vec{J} = 0$.

We can add Gauss's law and make ϵ and σ constant to get the relaxation equation:

$$\nabla \cdot \sigma \vec{E} = \frac{d}{dt}\rho_v \tag{1.14}$$

$$\nabla \cdot \vec{D} = \frac{\epsilon}{\sigma}\frac{d}{dt}\rho_v = \rho_v \tag{1.15}$$

$$\rho_v = \rho_{vo} e^{-\frac{\sigma}{\epsilon}t} \tag{1.16}$$

$$\frac{\sigma}{\epsilon} \approx \frac{10^{-11}}{10^{7}} = 10^{-18} s. \tag{1.17}$$

In actuality the frequency limit for the charge density to be considered zero is much less than 10^{18}Hz and closer to the optical limit ($\approx$ 500THz) which again, is the plasma frequency for metal. Refer to Chapter 7 in Jackson's *Classical Electrodynamics*[1] for more details on the plasma frequency. The frequencies we are operating at are well below the optical range, so we have

$$\nabla \cdot \vec{J} = 0. \tag{1.18}$$

We are left with a diffusion type equation for $\vec{J}$ with the divergence of $\vec{J}$ also equal to zero. A direct numerical solution of a vector diffusion equation is not easy to implement, among other reasons, due to the mixed boundary conditions required. We know that there is no current leaving the surface of the conductor except for the terminals. For each edge, that condition only sets the boundary condition for one of the vector components. In general, attempts to solve the vector diffusion equation, with $\nabla \cdot \vec{J} = 0$ as a condition are not fruitful.

1.1.2 Vector Laplacian

Rather than to try and solve the more involved diffusion equation, we take a step back and attempt to solve the simpler problem of the magnetostatic case where the current density is not changing in time. The simplified problem is known as the Vector Laplacian.

$$\nabla^2 \vec{J} = 0 \tag{1.19}$$

The method to solving the vector Laplacian[3] when $\vec{J}$ is confined to a plane is to define a scalar field W such that the curl of W times a unit vector in the $\hat{z}$ direction is defined as the current density vector components.

$$\nabla \times \hat{z}W = \frac{\partial W}{\partial y}\hat{x} + \frac{\partial W}{\partial x}\hat{y} = \vec{J} \tag{1.20}$$

Then $\nabla \cdot \vec{J} = 0$ implies $\nabla \cdot W = 0$ and *vice versa*. For a two dimensional surface, we only need to define the boundary conditions in terms of W. On edges where J_x should be zero, we only need to guarantee that the derivative of W with respect to y is zero, and likewise for J_y. Where we want the current entry and exit to the plate to be a constant, the associated function for W will be linear. An illustration of this formulation for a flat plate is presented in Figure 1.1. It doesn't matter what the value of the function for W on the edges is as we

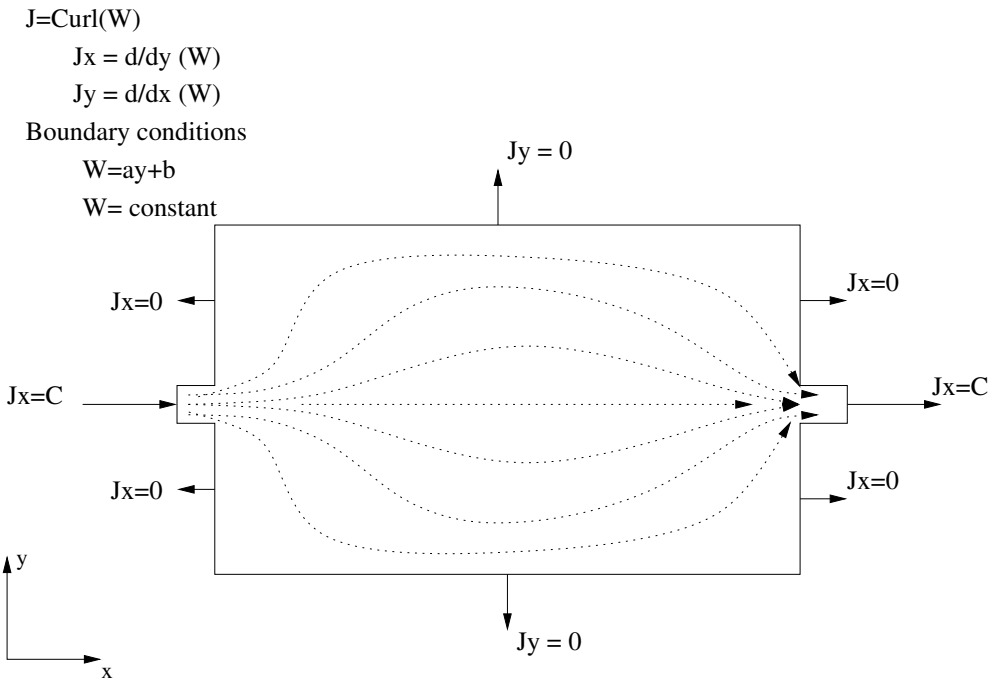

Figure 1.1: Illustration of the W solution of the vector Laplacian with associated boundary conditions

can add any constant to all edges and arrive at the same solution. We could now continue to persue the diffusion equation, but with close observation we note a problem with the scalar

solution to the vector Laplacian. We could add any solution that satisfies zero boundary conditions and not invalidate the approach. An eddy current could be implemented on top of the vector Laplacian solution since by definition, the current density leaving the surface is zero, thus the boundary conditions are satisfied. An illustration of this superposition of currents is presented in Figure 1.2. The implication with this finding is that any solution for the current density in a conductor needs to include magnetic effects, such as those that would produce eddy currents. The W method only works when $\vec{J}$ is not a function of time, and hence no eddy currents exist. The W method will give the same results for a steady state solution of the current as simply solving $\nabla^2 V = 0$ and then using the material relationships and multiplying the conductivity times the gradient of the voltage ($\vec{J} = -\sigma \nabla V$).

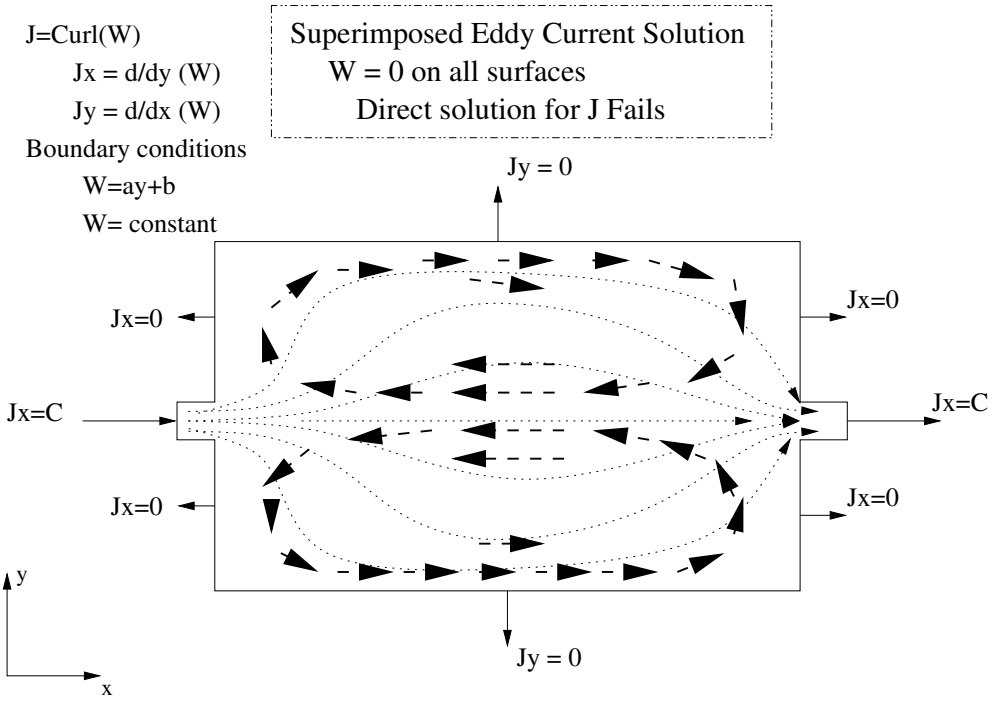

Figure 1.2: Addition of an eddy current to the vector Laplacian solution. An addition of an eddy current to the solution surface doesn't effect the boundary conditions for the W solution of the vector Laplacian due to the boundary condition being met on all surfaces. Any eddy current can be added to this solution, thus the W solution is inadequate to describe the current density. Any solution for the dynamic current density in a conductor needs to include magnetic effects.

1.1.3 Skin depth

Another avenue of investigation for solving for current distributions in conductors is to examine skin effect in thin conductors. This has been an active area of research with regard to the impedance of conductors[4,5,6,7,8,9]. The assumption here was that we know that magnetodynamic effects cause the skin effect in conductors. If we have a relationship for the skin effect in a flat conductor as a function of frequency, then *via* Fourier synthesis we should be able to construct a transient current distribution. This approach makes several assumptions as with the vector Laplacian. We assume that we are in a good conductor, and that it is has constant conductivity. To use skin effect formulas we also must require that we have regular shaped conductors, such as strip lines.

We implemented a formula for the skin depth in a thin rectangular conductor from Ney[5] in `C++` code called `skin.cpp`. Ney's paper is on 'striction' effects to the current density in strip lines. If the conducting strip is fed from a smaller line, then the current distribution will have effects from the constriction of the current into and out of the strip. This 'striction' effect was examined and mathematically defined. Verification of Ney's formulas was conducted via comparison of total impedance of the line to calculated values. We are uninterested in these 'striction' effects here, but the unaltered form for the skin effect can be used for our purposes. In Ney's formulation the cross section of the conductor lies in the x-y plane, and the direction of the current flow is in the z direction. The expression for the z directed electric field in the conductor is

$$E_z(x,L) = \frac{I\sqrt{j\omega\mu\sigma}\cosh(\sqrt{j\omega\mu\sigma}x)}{2\sigma ac\sinh(\sqrt{j\omega\mu\sigma}a)} \tag{1.21}$$

where 'a' is half the width of the conductor, and 'c' is the thickness.

The code reads an input file containing time and voltage data for an input waveform, and then performs a Fast Fourier Transform on the waveform and extracts the Fourier series coefficients A_n and B_n for n harmonics of the fundamental frequency. The number of harmonics is a user entered parameter. The code then uses those coefficients with eq.(1.21)

to construct the transient distribution of the current in the conductor. A single frequency sinewave is presented in Figure 1.3. In it we see the magnitude E_z versus the distance from the center to the edge, and versus time. For a single frequency sinewave, we see the skin effect as expected. The magnitude of the electric field, and by extention the current density, is larger at the edge than at the center. The transient distribution due to a step function

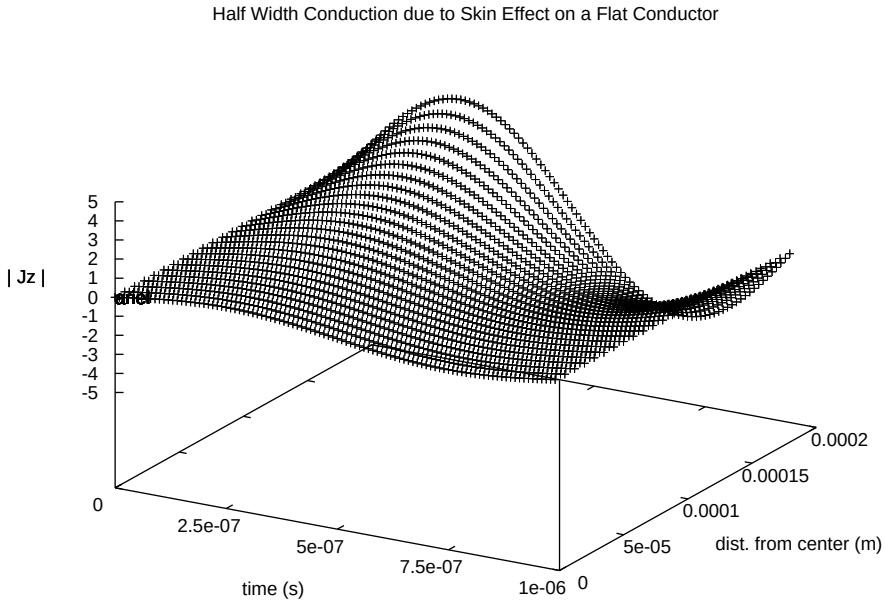

Figure 1.3: Current density reconstruction for a single frequency component in a thin conductor

was simulated next. In Figure 1.4 we see a square wave that has been reconstructed using only 5 harmonic components. This is obviously too few harmonics to accurately represent the nature of the square wave, but illustrates the functionality of the test code. We see that the higher harmonic components push the electric field distribution further to the edge of the conductor. This is congruent with what we expect to see with a square wave. The same square wave was reconstructed using 50 harmonics. A picture of the output of this reconstructed wave can be found in Figure 1.5. Quite clearly we can see that at the

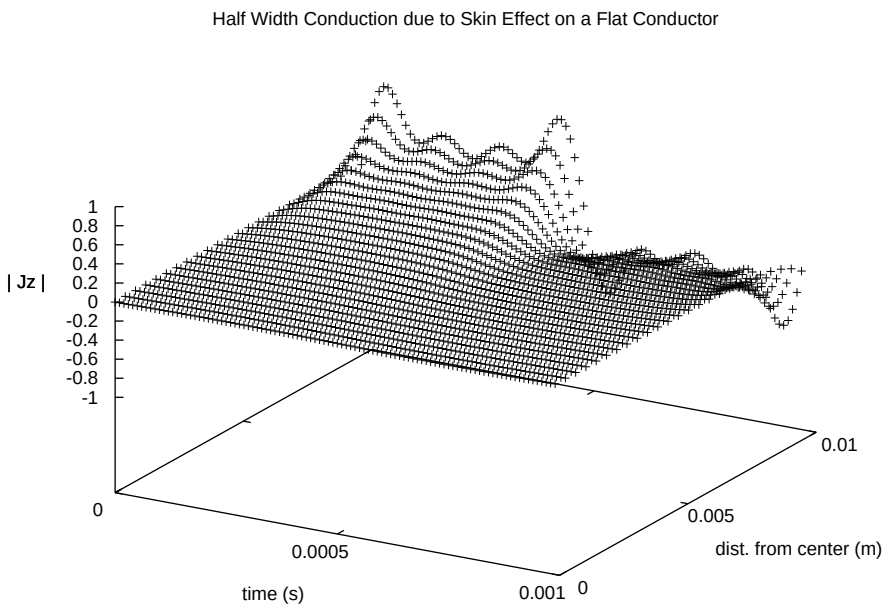

Figure 1.4: Current density reconstruction via Fourier Synthesis for a thin conductor with a squarewave input using five harmonics

transition edges of the square wave the high frequency components of the wave force the field distribution to the edge of the conductor. As the wave reaches it's top level, the lower harmonics become more prominent in the reconstructed wave, and the distribution relaxes towards the middle of the conductor. In this figure we also see that the reconstruction is anticipating the fall time. This is purely an artifact of the Fourier Series reconstruction. A line plot for the edge and the center magnitudes of the electric field are shown in Figure 1.6. Here we can clearly see the differences between the edge and the center. The jagged nature in the edge plot is due to the finite number of harmonics used in the reconstruction. The slow diffusion of the current from the edge to the center can be seen as the wave reaches a steady state.

This test using skin effect and Fourier reconstruction to examine transient waveforms in flat conductors is useful to gain an understanding of the general behavior of the current

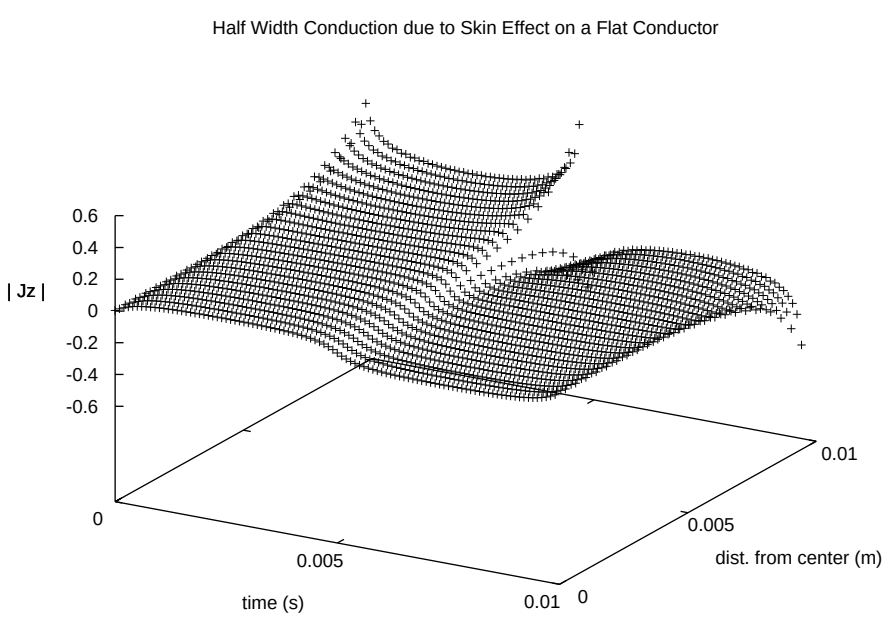

Figure 1.5: Current density reconstruction via Fourier Synthesis for a thin conductor with a squarewave input using fifty harmonics

in the strip lines, but is less useful for actual applications as we want a robust solver not confined to regular conductor shapes, and with the potential for inhomogeneous materials and changing conductivities.

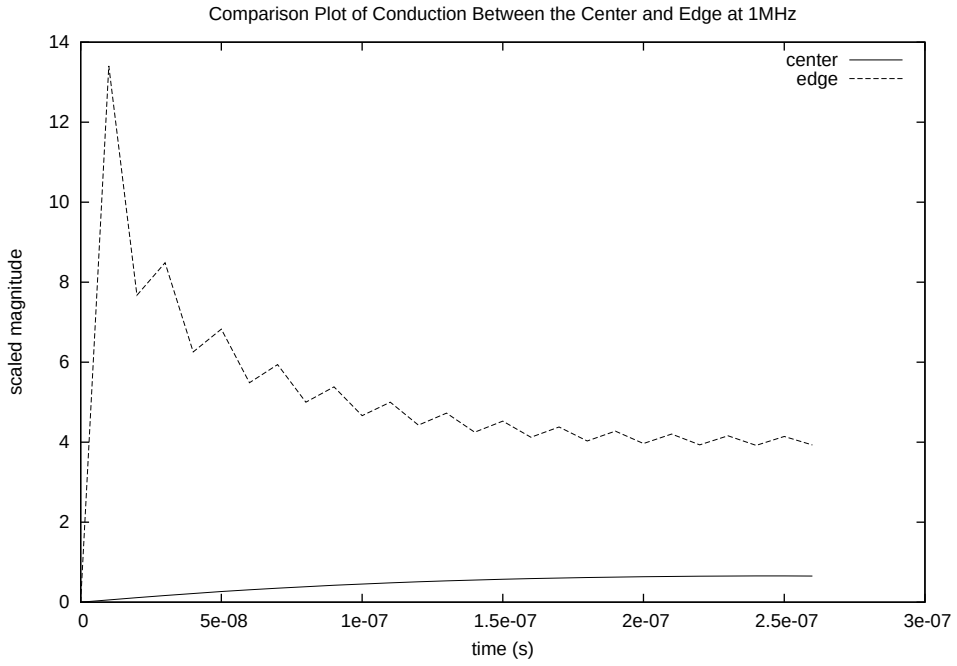

Figure 1.6: Selected edge and center current density from Fourier Synthesis of a squarewave input using 50 harmonics for reconstruction

1.2 Existing electromagnetic solvers

1.2.1 Quickfield

Quickfield[10] is a commercial finite element program for solving field problems. Quickfield contains solvers to perform many types of electromagnetic field computations, but they are restricted to DC and AC fields. The transient solver packages are limited to a thermal diffusion and magnetic diffusion packages. Quickfield's market is for fast and easy EM field calculations in electrostatic or magnetostatic cases. Their application is not suited to performing transient current density calculations.

1.2.2 Maxwell 3D

Maxwell 3D[11] is a commercial EM solver produced by Ansoft. An initial exploration of Maxwell3D's capabilities was performed at Lawrence Livermore National Laboratory as part of a summer internship. Despite claims of transient simulation capability, output for a transient current conduction in a strip line returned non physical results. An example of a simulation run to evaluate Maxwell 3D is shown in figures 1.7 through 1.10. A simple conducting strip was specified using Maxwell's built-in meshing program. A series of tests was then performed on the strip. Figure 1.8 shows the AC conduction solution for the simple strip. Analytical solutions[5] for the strip show that as the frequency increases, the current density crowds to the outside edges of the strip. This plot clearly shows an imbalance along an axis of symmetry. Traces along the top and bottom surfaces of the strip are in Figure 1.9. Here we can see that the current density is almost uniform in the thickness, as should be the case, but the asymmetry along the long axis is not a physical solution for a strip in open space. A transient current conduction simulation was performed on the same strip with Maxwell 3D's numerical solver. The transient response also shows errors. The current density in the strip was visibly uniform the entire length and width of the strip, gradually increasing with the source driving function. A careful examination of the magnetic field produced by the current showed that the current density was again not

physical. A plot of this magnetic field is in Figure 1.10. There is a random nature to the field which can not be explained in any way as a physical phenomenon. This then is the result of the numerical solver being used by Maxwell 3D. Since Maxwell 3D is a commercial product, its source code is unavailable for examination. An educated guess would suggest that the finite element solver is trying to find a minimum energy state for the current, and will eventually iterate down to a lowest numerical error state which is only bound by the boundary conditions. Another example was used to evaluate the suitability of Maxwell 3D for transient current density conduction simulations. A multiply-connected thin conductor with voids was built and simulated. The mesh for the top plane of the conductor is in Figure 1.11. A solid return path was included with the mesh, but is not present in the figure. The transient current density solution for this mesh is in Figure 1.12. As with the case of the simple strip, this picture shows that the transient simulation shows a non-physical solution. The current should be flowing from the top of the figure to the bottom of the figure. We would expect to see some symmetry present, as well as a constant magnitude from top to bottom. Maxwell 3D was deemed ill suited to perform the transient current simulations.

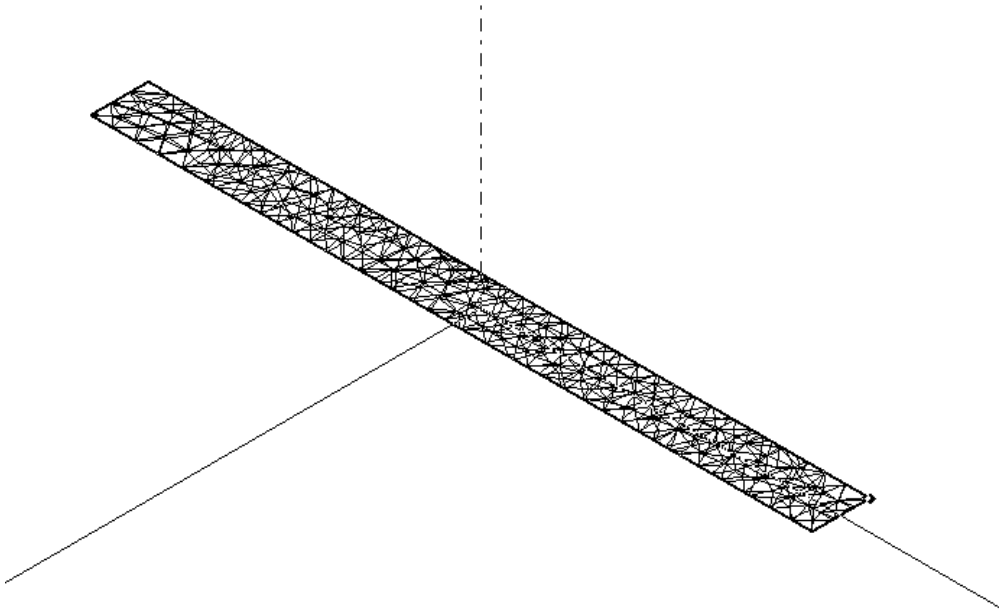

Figure 1.7: Finite element mesh of a long conducting strip made in Maxwell 3D. The strip is 1mm thick by 40mm wide by approximately 500mm in length.

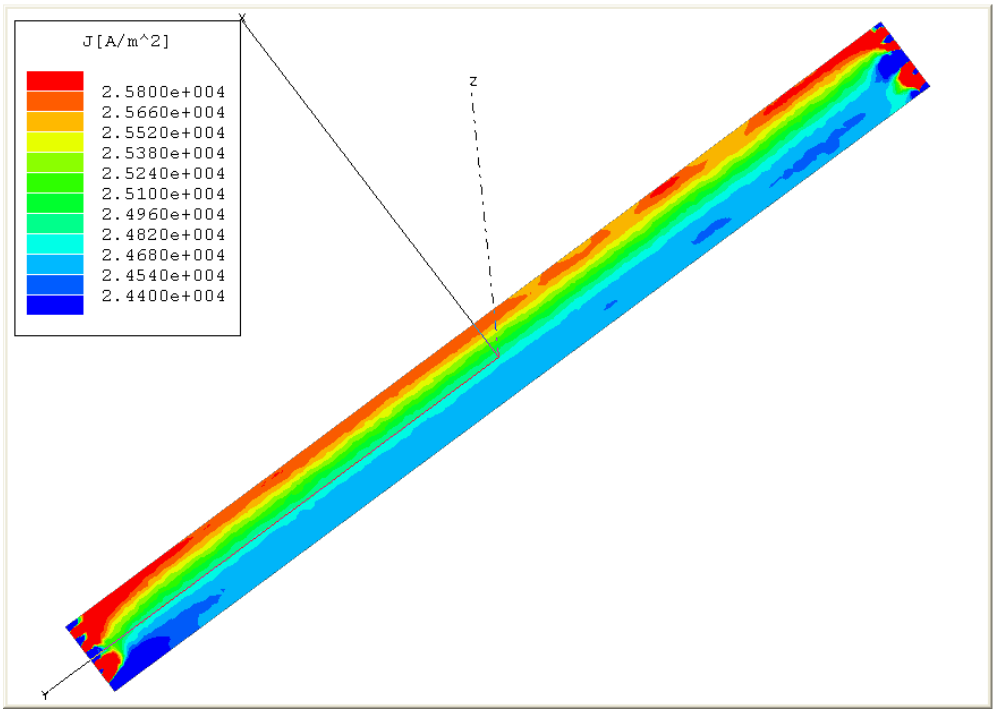

Figure 1.8: AC current conduction solution of a long conducting strip made in Maxwell 3D. The strip was driven at 1 MHz. The AC current solution as provided by Maxwell 3D is invalid as the simulation shows a non-symmetric nature which isn't physical

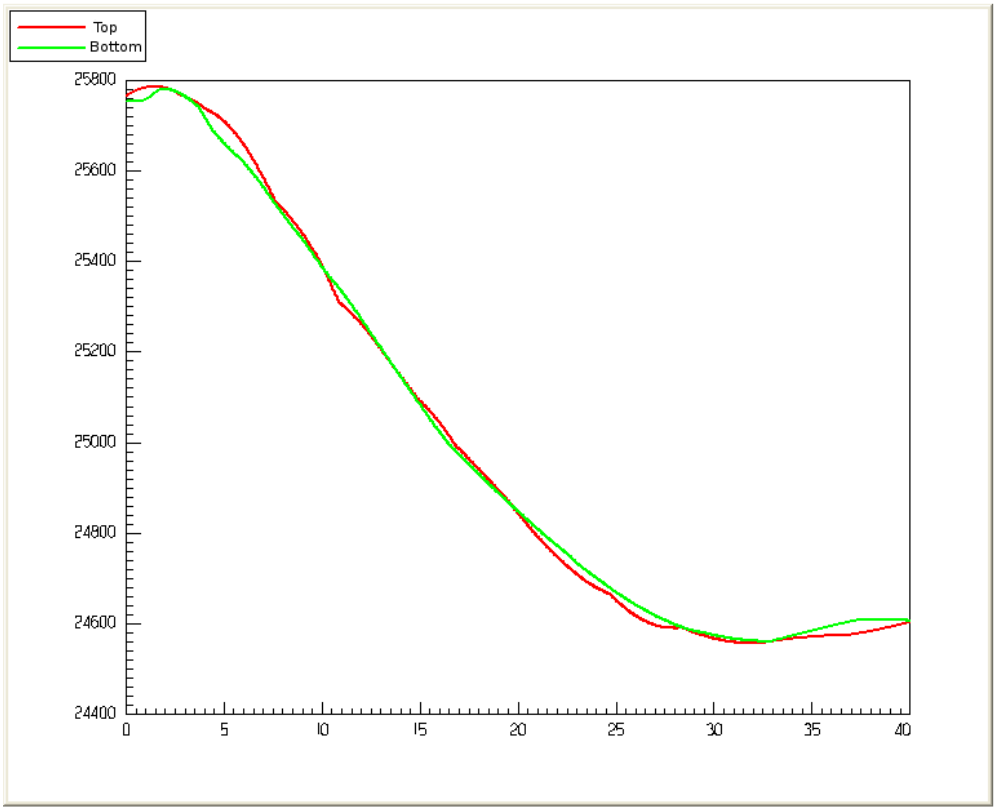

Figure 1.9: Traces of the current density made for the long conducting strip in Figure 1.7. This highlights the non-symmetric nature of Maxwell 3D's solution shown in Figure 1.8.

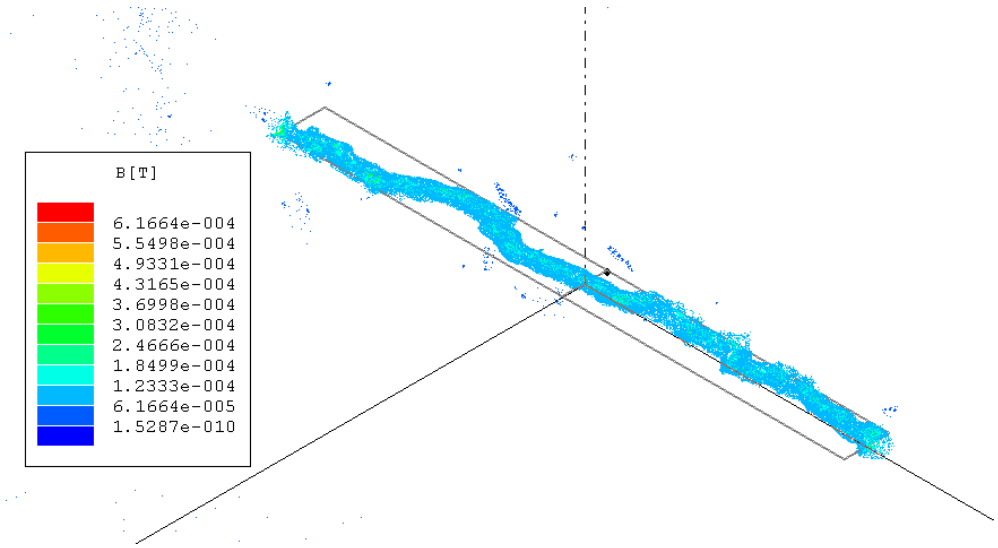

Figure 1.10: This figure shows the magnetic field calculated from the current density produced by Maxwell 3D's transient current solution. The random nature of the B-Field is produced by non-physical current densities from the Maxwell 3D numerical solver.

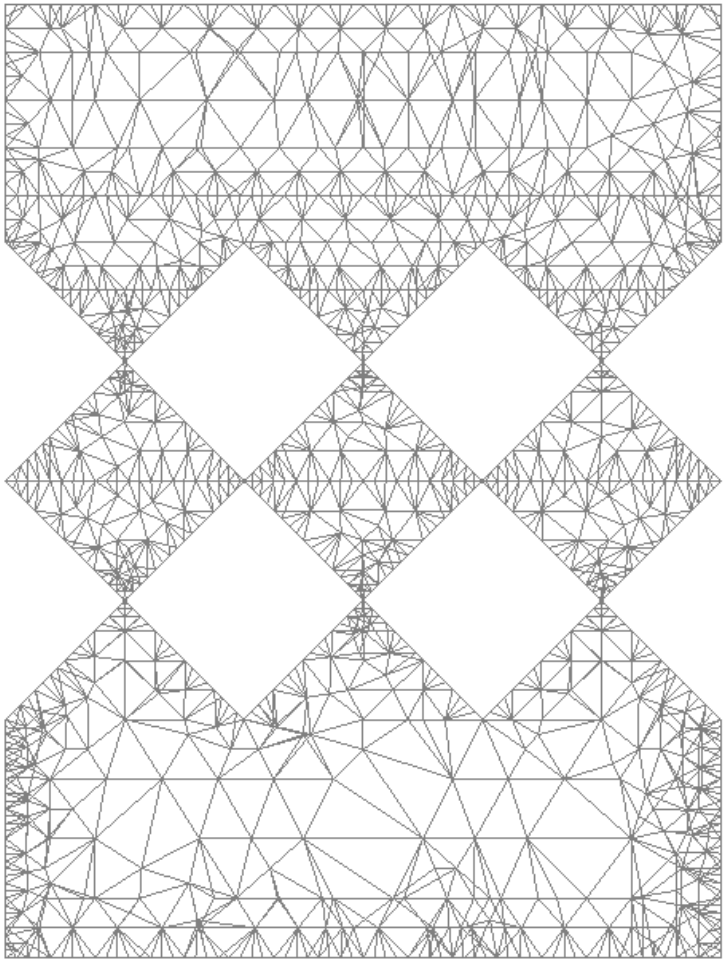

Figure 1.11: The top side of a mesh generated with Maxwell 3D for a thin conductor with voids. The backplane is a solid conductor and is not shown.

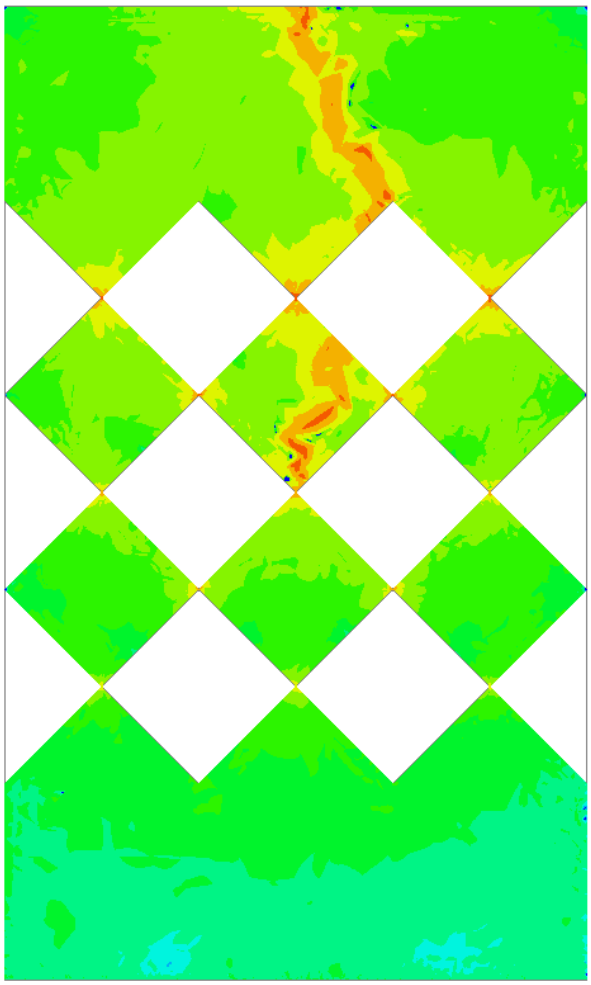

Figure 1.12: The transient current solution for the multiply-connected thin conductor with voids as numerically solved for with Maxwell 3D.

1.2.3 MAFIA 4

`MAFIA`[12] (the solution of **MA**xwell's equations by the **F**inite **I**ntegration **A**lgorithm) is a set of solver packages made by CST :Computer Simulation Technology. This company was founded by Thomas Weiland, who has been very prolific in research with EM fields, Integral Equation Techniques,and the Finite Integral Method. The `MAFIA 4` solvers are sold with CST's commercial products, a few of which are CST: Design Studio, CST: EM Studio, and CST: Microwave Studio. Their published material claims that they can solve transient problems, thermal problems, current conduction problems (assuming both AC conduction and transient conduction problems). The Finite Integration Technique[13] (FIT) has been applied to eddy current problems[14] as well as other discrete EM problems[15,16,17] among many other applications associated with the solution of Maxwell's equations[18]. The FIT is a solution method where a set of dual meshes are interlaced much like the Finite Difference Time Domain[19] (FDTD) methods. The electric and magnetic fields are then discretized on the edges of the dual meshes. For a single cube element in a mesh, the electric field would be discretized quantities on the edges of the cube. the magnetic field would be on dual mesh spaced a half width off in all directions such that an edge containing the discretized magnetic field would pass through a face of an the electric field's cube. In doing so we can take the curl of the electric field by calculating the values as we go around one face of the square side to the cube. The result would be the magnetic field that is quantized on an edge passing through the center of that face. This pattern repeats throughout the entirety of the mesh and allows for calculation of propagating EM fields on the mesh.

`MAFIA` and it's associated solver utilities use a transient pulse response to simulate AC responses of structures to EM fields. They stimulate a given structure with an impulse and simulate the propagation of the wave. Then they perform an Inverse Fourier Transform on the waves and can extract a full spectrum frequency response from the object. This is a selling point of their software. It is faster to perform one transient simulation and process the output than repeatedly simulate full field propagation for multiple independent

frequencies over the range required the individual problem. It is possible that this oftware package can solve the problem we are interested in of transient current distributions. However the CST family of codes was cost prohibitive so an evaluation was not able to be performed. An observation of their published material suggests that this code would be a full field simulation and would require more computer resources than a direct solution of $\vec{J}$.

1.2.4 GetDP

`GetDP`[20](**G**eneral **E**nvironment for the **T**reatment of **D**iscrete **P**roblems) is an open source program for solving a wide variety of partial differential equation problems. `GetDP` was developed by Christophe Guezaine as part of his Ph.D. dissertation work titled "High order hybrid finite element schemes for Maxwell's equations taking thin structures and global quantities into account" [21]. The code was designed with another open source and free program called `GMSH`[22, 23] which is a mesh generation program with a graphic user interface (GUI), and modules for displaying data on a finite element mesh. `GetDP` claims that it is capable of solving transient problems, integral equations, and coupled problems, such as thermo-electrical problem formulations. It is a work in progress and an effort to debug and enhance the software is an ongoing process. `GetDP` takes mesh files made with `GMSH` and uses them as inputs to its solver package. The solver for `GetDP` is a text based input formated file where the problem is specified in weak-form notation. After an extensive investigation, `GetDP` failed to be capable of implementing a transient, integral equation formulated problem. We were able to implement a single step solution for an eddy current density but were unable to loop back and use that output as an input for a next step. We were also unable to implement a superposition solution of a source current density and an eddy current density solution. We believe this is due to how the integral equation is implemented. It was possible to perform an integration of a quantity produced by the finite element solver in the post processing phase, but one could not be included in the direct formulation of the problem

statement. The amount of effort required to adapt GetDP to our needs was deemed better spent creating one whereby we could implement the functionality we desired directly.

While GetDP is not directly useful to us, the bundled geometry and meshing program GMSH is a very convenient utility for the generation of finite element meshes. The advantage of the open format of GMSH is we have access to the mesh formatting style. It is a small task to make use of that mesh file format when building our current density solver.

Other electromagnetic software exits. There are many commercial products and many more open source options available, however none of these are suitable for performing the transient current distribution calculations. Evaluation of these existing numerical solvers has demonstrated the need to create a custom solver for the transient current distribution in thin conductors. The major reasons that this functionality is not available in existing packages is due to the high requirements in computational resources and the lack of interest in this specific problem. Most current conduction problems do not require the level of detail we wish to have. Therefore efforts were directed into the creation of this solver.

1.3 Survey of other transient methods

A survey of existing research into transient current density distributions was performed. Transient eddy current research has been performed by a great number of people. Most of these approaches are not suitable for application to solving for the total current density in a thin conductor but they can give us an indication of where the research needs to look for a total solution. The eddy current solutions generally make starting assumptions which we wish to avoid such as uniform and homogeneous material properties. Eddy current solutions are by definition AC and have sources external to the body of integration. Transient eddy current research performed by Lee[24] applies a finite element implementation to solving for the magnetic field vector as the state variable then uses that to compute the eddy current. His approach was tested on sparse meshed and with external sources. Pichon[25] applies a hybrid finite element and boundary element method to solve for transient eddy

currents, but in axisymmetric problems. He uses a magnetic vector potential approach to his solution. Clemens[26] working with Weiland has published many papers about field calculations with the Finite Integration Technique(FIT). Weiland has published extensively over his FIT method for many years and it's implementation into a family of computer codes `MAFIA`[12]. The FIT as presented in Section 1.2 offers some insights into edge elements in the solution. Work on the FIT has been directed toward EM field propagation, and thus is a full field solver. Another approach by Leonard[27] performs a hybrid and integral formulations for 3-dimensional eddy current problems relies on assumptions of constant conductivity and exterior sources. An advantage of the integral formulation is that the solution space may be confined to the conductor. By doing this, we can omit the meshing of the free space surrounding the conductor, which we would have to include in a full field solution. The integral approach is also applied by K. H. Carpenter[28]. Research by Tsuboi[29] implemented an eddy current analysis with a superposition solution of source current density and eddy current density. This approach arrives at a total current density distribution in a thin conductor but it is restricted to AC conditions. Many other eddy current solutions have been presented[30, 31, 32, 33, 34, 35], that offer solutions to eddy current problems with each contributing some aspect to what we require to satisfy our demands. An integral formulation provides computational savings by restricting the solution space to the conductor as does performing a direct solution for $\vec{J}$. We can use superposition to find the total current density in a conductor due to source currents and an impressed eddy current.

A specific subset of these magnetic vector potential effected current density problems are skin effect problems. Internal fields which cause skin effect can be viewed as a superposition problem with source currents and secondary field currents caused by the magnetodynamic actions of the sources. Skin effect studies have also been performed solving for the total current distributions in the conductors. Most of this research has been directed to the impedance of strip lines. More on this topic is covered in Chapter 2.

Chapter 2

Partial Inductance skin effect studies utilizing SPICE

2.1 Introduction

Any transient current distribution solver will require verification of it's accuracy and precision. We built a circuit theory model for a strip line using a partial inductance method. With the SPICE circuit simulator, we simulate current density distributions for AC and transient excitation. To verify the SPICE model, use was made of magnetic probes invented at Lawrence Livermore National Laboratory[36] which measure the induced voltage produced by the magnetic field between the strip line's conducting paths. These probes were also simulated with SPICE providing a direct link between the simulated induced voltages in the probes due to the current distribution, and measured values observed at different frequencies.

Current distributions in flat rectangular conductors have been an area of study for some time. Many different approaches to calculation have been used, and have been verified by matching calculated impedance values with measured ones[4,5,6,7,8,9]. We have taken a different approach to both the calculation and verification. While impedance measurements give a global indication of current distribution, they do not provide information on local current density. A magnetic current probe has been invented by Lawrence Livermore National Laboratory[36] (LLNL) which can detect local current density, albeit in a convoluted form. The

testing of this probe requires a simple geometry where the current density may be simulated for comparison. We have made measurements using these probes to find the local induced voltages due to the magnetic field of the currents, and hence the current distribution, for a case of a strip line separated sufficiently to accommodate the probes and long enough to provide essentially infinite length for simulation. The simulation has been done by applying the partial inductance method to produce a net list for a `SPICE`[37] circuit simulator. The `SPICE` results directly simulate the probe readings for comparison to measurements without the need to deconvolute the current from the probe values.

These `SPICE` models can then be used with transient simulations to compare with any transient solution by any new transient current density solver.

2.2 Partial inductance

Partial inductance methods have been used to calculate impedance and current distributions in flat conductors by Weeks, *et. al.*,[4] and others[38]. They find current density as a function of frequency and location over the cross section of straight conductors having lengths sufficiently long compared to the cross section dimensions that they may be treated as infinitely long. Other methods exist to model the electrical systems such as the partial element equivalent circuits (PEEC)[39] and the transmission line method (TLM)[40]. Because the frequencies of operation are low for this test case, capacitive effects have little impact on the nature of the current distribution and an expression for the partial capacitance is not needed. Furthermore, our equivalent circuit has a short circuit load which further reduces the effect of capacitive elements. At higher frequencies such as those where the length of the conducting strip approaches the wavelength of the source signal, models like the PEEC or the TLM would be necessary to accurately model the system. We have chosen to apply the partial inductance procedure to cases of straight conductors of finite length. This method provides values for self and mutual inductance between finite divisions of the conductor which may then be used in a circuit theory model of the conductor. This circuit model is

then solved by the SPICE simulator to obtain the current distribution.

2.2.1 Partial inductance for finite length conductors

Mutual and self inductance can be found by the energy method[1]. The mutual inductance between conductors i and j is

$$M_{ij} = \frac{(\mu_0/4\pi)}{I_i I_j} \iiint_i \iiint_j \frac{\vec{J}_i(\vec{r}) \cdot \vec{J}_j(\vec{r'})}{|\vec{r} - \vec{r'}|} d^3r' d^3r. \tag{2.1}$$

[Note that eq.(2.1) shows that $M_{ij} = M_{ji}$.]

When $i = j$ then $L_i = M_{ii}$ of eq.(2.1) is the self inductance.

A conductor, formed into a closed loop, may be separated into subdivisions in which the current can be considered uniform, and the inductance integral separated into a sum of integrals over the geometry of these regions. The terms in the sum are the partial inductances. One then makes a circuit model for the conducting loop, using the partial inductances along with the resistances of the conductor subdivisions. The circuit model is solved to find the current distribution over the conductor. A simple illustration of applying this method to a flat conductor in a strip line configuration is shown in Figure 2.1

The partial inductance between subdivision m of conductor i and subdivision n of conductor j is given by

$$M_{ij}^{(nm)} = \frac{\mu_0}{4\pi} \iiint_{im} \iiint_{jn} \frac{\hat{J}_j(\vec{r}) \cdot \hat{J}_i(\vec{r'})}{S_i(\vec{r}) S_j(\vec{r'}) |\vec{r} - \vec{r'}|} d^3r' d^3r \tag{2.2}$$

where $\hat{J}$ is a unit vector in the direction of the current density, assumed uniform over the subdivision, $S_i(\vec{r})$ is the local cross section area of subdivision i, and where integration is over the volumes of the subdivisions.

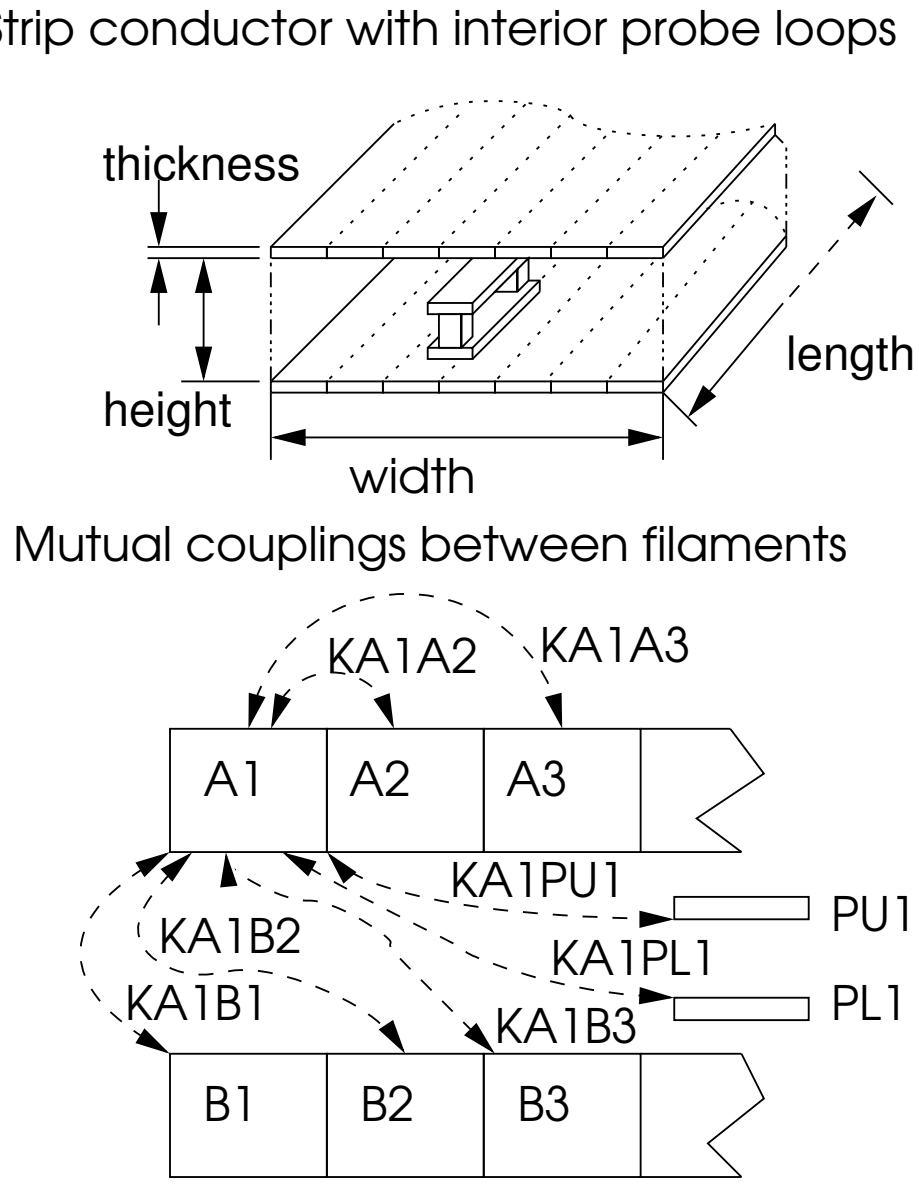

Figure 2.1: Not to scale representation of a strip line consisting of two thin, flat conductors in close proximity separated by a distance much smaller than their width, subdivided into rectangular filaments. The line has a finite length, width, thickness, and a separation height between the current paths. Probe loops are also represented as rectangular filaments between the two planes of the conductors and having an upper and a lower section. Each filament has an inductive coupling to each other filament, as indicated by coupling coefficients (mutual inductance divided by geometric mean of self inductances). (*e.g.*, KA1A2 being the coefficient between filaments A1 and A2).

2.3 Numerical confirmation of experimental values for current distributions

2.3.1 Calculation of partial inductances

For the geometry of Figure 2.1, the subdivisions of the conductor are all parallel, and when chosen small enough, meet the criteria for eq.(2.2) to be used. In this case all the $\hat{J}$ have the same direction and all the $S_i(\vec{r})$ are constant with $\vec{r}$ and have the same value. There are only two conductors, so i and j can be only 1 or 2. Use x and y as coordinates of the cross section and z along the length. Then eq.(2.2) becomes

$$M_{ij}^{(nm)} = \frac{\mu_o/(4\pi)}{(\tau\delta)^2} \int_0^{\zeta} \int_{0+h}^{\tau+h} \int_{(n-m)\delta}^{(n-m+1)\delta} \int_0^{\zeta} \int_0^{\tau} \int_0^{\delta} \frac{1}{\sqrt{(x-x')^2+(y-y')^2+(z-z')^2}} dx'dy'dz'dxdydz, \tag{2.3}$$

where the thickness of the flat conductor is τ, the total width of the conductor is $N\delta$, where N is the number of subdivisions of the width, the length of the conductor is ζ, and the vertical separation of the conductor subdivisions is h, which is zero if $i = j$ and the "height" in Figure 2.1 if $i \neq j$.

The six-dimensional integral of eq.(2.3) has an integrable singularity when the subdivisions touch (or for the case of self inductance of a subdivision).When numerical integration is used to evaluate eq.(2.3), it must be robust enough to handle the singularities. We have combined a Monte-Carlo technique[41] along with the "`dcuhre`" multidimensional integration function[42] to obtain robust results in minimum time. (A second way to find the value of these integrals is to use the "`fasthenry`" program[43] for a single frequency and obtain inductance values from the calculated impedances.)

The magnetic probes, used to detect the current distribution in the strip line simulated by these calculations, may also be represented in a circuit model by inductances which are mutually coupled to the subdivisions of the line conductors. The integrals for these couplings are the same as those of eq.(2.3) except for the limits on the z integrals and the values of h.

2.3.2 SPICE simulation

A circuit model for the flat cable (or strip line) of Figure 2.1 is shown in Figure 2.2. With this circuit model for a strip line, even with as few as 10 subdivisions of each conductor, there would be 400 inductances, not including the magnetic probe inductances and couplings. In order to use the SPICE circuit simulator for this problem, a netlist is required for the equivalent circuit. A computer code was written to take the calculated filament inductances, resistances, and inductive coupling coefficients from the partial inductance method and output a netlist suitable for input to SPICE3[44]. Obviously, this netlist can be very long when a large number of conductor subdivisions is used.

For the netlist, filament resistance is calculated directly from user-entered parameters for strip thickness, width, length, and the material conductivity. The width is divided by the number of desired segments to give the filament width. Self and mutual inductance are calculated by numerical integration of eq.(2.3). Mutual inductances are converted into coupling coefficients, as is required by SPICE . The SPICE netlist formatting program also calculates the self and mutual inductances of the probe loops used for experimental verification.

From the SPICE results, as long as the skin depth was greater than the width of an individual filament, adding more filaments to the model didn't improve the performance of the simulation. At frequencies where the skin depth was less than one filament in width, the SPICE simulation results were non-physical. At 200 filaments, the frequency where the skin depth approached the filament with was near 100 MHz. The physical experiments used to test the probes were conducted at frequencies less than this limit.

When SPICE3 is run with the input generated by our program, it provides currents for each filament, which gives us the current distribution in the conductors. The simulated voltages across the resistances in the magnetic probes, labeled RP1 and RPm in Figure 2.2, are the values we compare to measurements to verify the method. Figure 2.3 shows the results from a SPICE simulation of current distribution in a strip line over a range of frequencies. This simulation used 200 filaments per conductor.

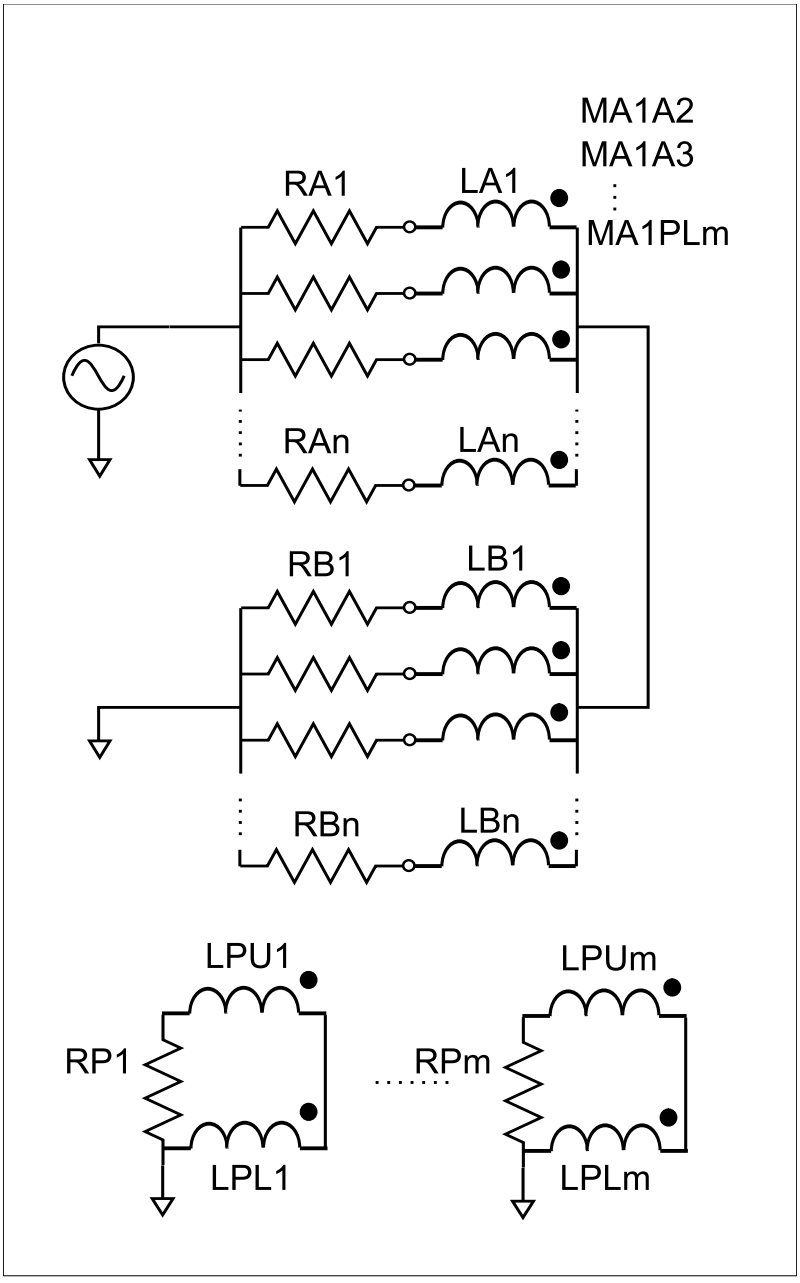

Figure 2.2: Circuit model representation for the strip line segmented into 'n' filaments and 'm' probes. Resistances RP1 and RP2 represent the internal resistance of the test equipment reading the loop voltage.

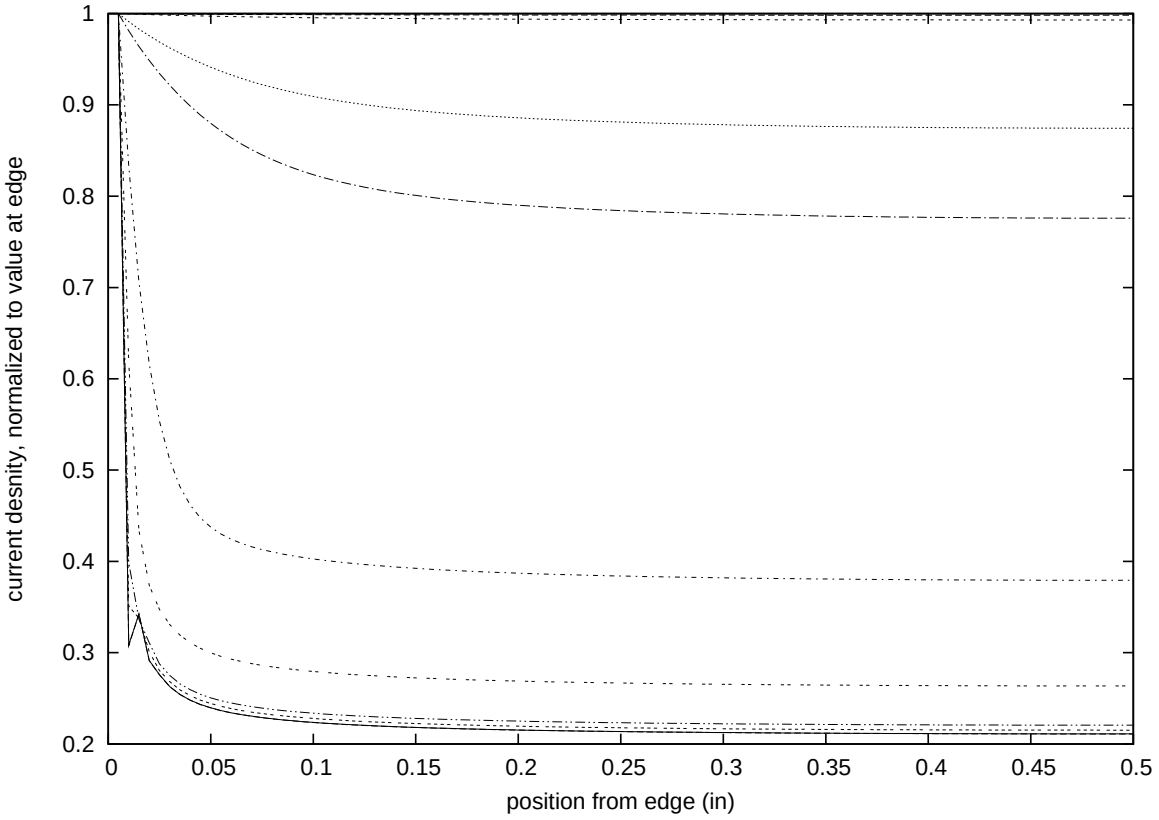

Figure 2.3: Simulated current density profile for a strip line, normalized to the value at the line edge. Only half of one conductor shown due to symmetry. The family of curves cover frequencies 1 kHz (top curve), 5 kHz, 10 kHz, 50kHz and so forth, ending at 100 MHz (bottom curve) This simulation used 200 filaments per conductor. This simulation shows smooth current density profiles until 100MHz where the skin depth is less than the width of one filament.

The currents have been normalized to one at the outside edge of the conductor. Only half of one conductor's current is shown, since the others may be obtained from it by symmetry. There is a non-physical notch near the outside in the current plot at the highest frequency. This is because at that frequency the number of subdivisions is too small.

2.3.3 Experimental observation

The magnetic current probes

The magnetic current probe used to measure the magnetic field inside the strip line was invented at Lawrence Livermore National Laboratory (LLNL) by May and Petersen (patent pending[36]). The probe consists of a loop made in a printed circuit board by connecting two vias on the top and bottom of the board with traces to the edge of the board for mounting connectors to the instrumentation. The experimental setup consists of two boards with a centered 1 in wide strip along it's 8 in length, and a probe board. Two versions of the probe board were fabricated and used in testing (see Figs. 2.4, 2.5, and 2.6). The strip line boards are connected to the driving source by coaxial cable. The other end is shorted with a full width shunt. The center tapped connection of the coax is sufficiently far from the probe loops that we can neglect any current density effects due to striction[5]

Evaluation of the magnetic current probes from a previous design showed parasitic capacitive effects from the surface traces on the probe boards were a significant source of error at higher (RF) frequencies. The presence of the extra copper traces contributed excessive capacitance to the system changing the nature of the current distribution rather than being a passive monitor of the magnetic field.

Two probe board designs were used for this series of measurements. One board, used for higher frequency measurements, has a single probe loop in the center position. The other, used for lower frequency measurements, has an array of probe loops offset from the center in increments of 0.1 in with the final loop at 0.5 in, which is the edge position for the strip line. The single probe loop board design has less copper in the traces to minimize the capacitance for higher frequency measurements. This board achieves spatial resolution by being physically moved and mounted inside the strip boards via a series of measured mounting holes. The mounting holes are spaced 0.125 in apart allowing for measurements at 0.0, 0.125, 0.25, and 0.375 in offsets from center. The lower frequency probe array board is mounted centered to the strip boards, and the spatial measurements are taken from the

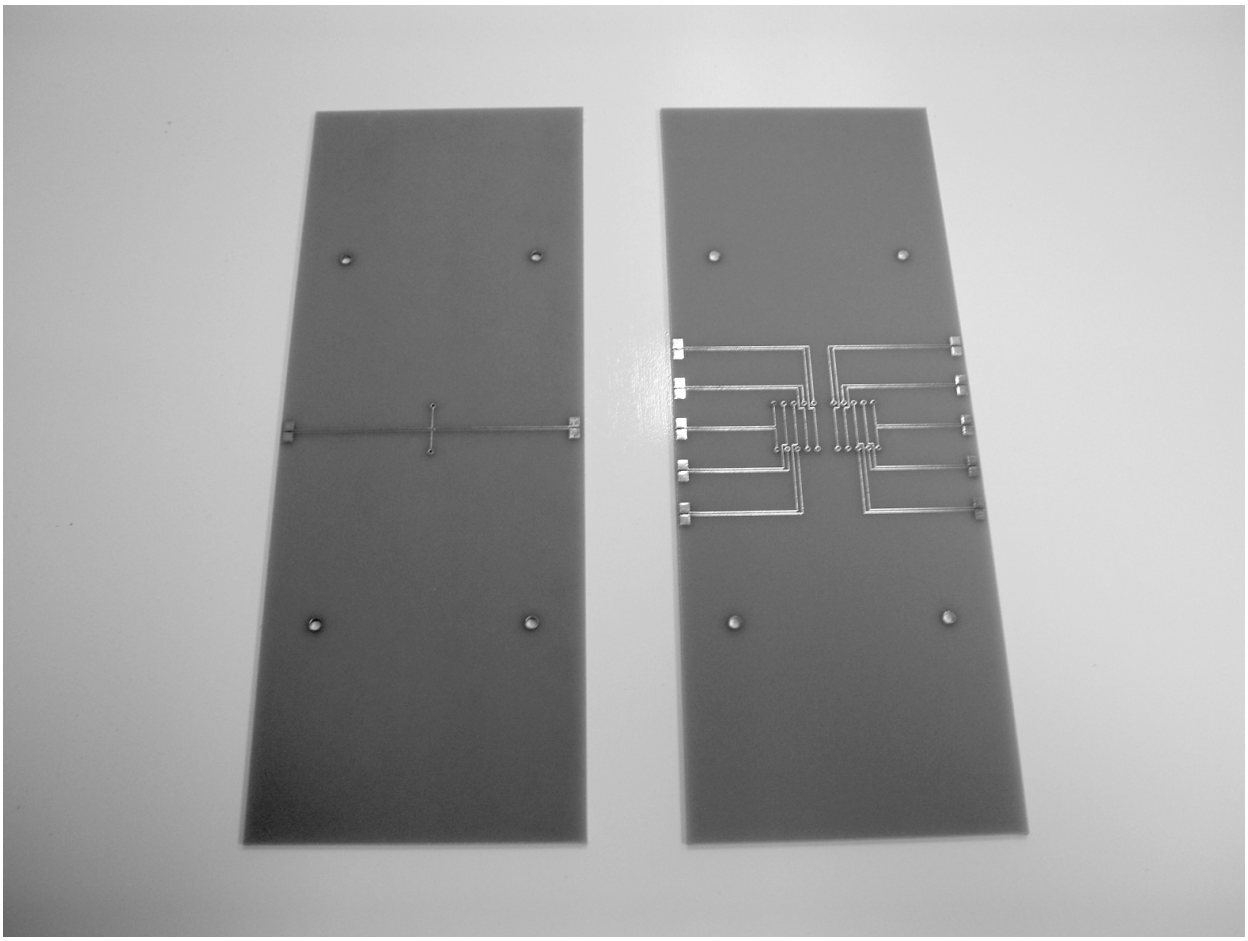

Figure 2.4: Probe boards used in experimental measurements. The board on the top has a single probe loop while the other board has an array of probe loops. Both are made to be able to measure the magnetic flux inside a flat strip line at different positions along the width.

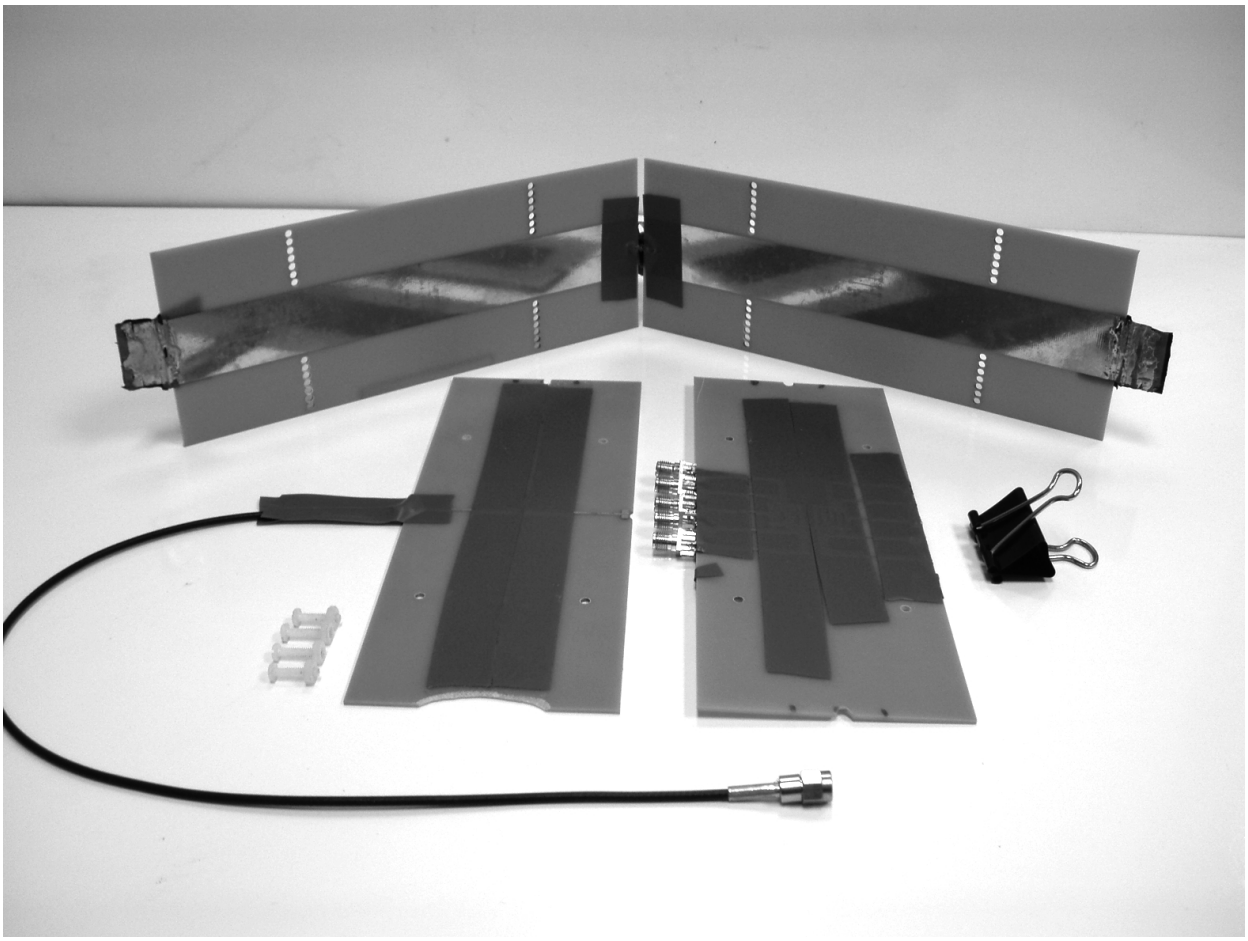

Figure 2.5: Disassembled view of the entire probe and strip line assembly. Interchangeable probe boards present for photo. The strip line feed is from a coaxial line attached to the center of the strip end. Circuit continuity at the shorted end is achieved with clamped copper strips the width of the strip line. The probe location is sufficiently far from the center-fed end that the we can ignore any striction effects to the current density distribution[5].

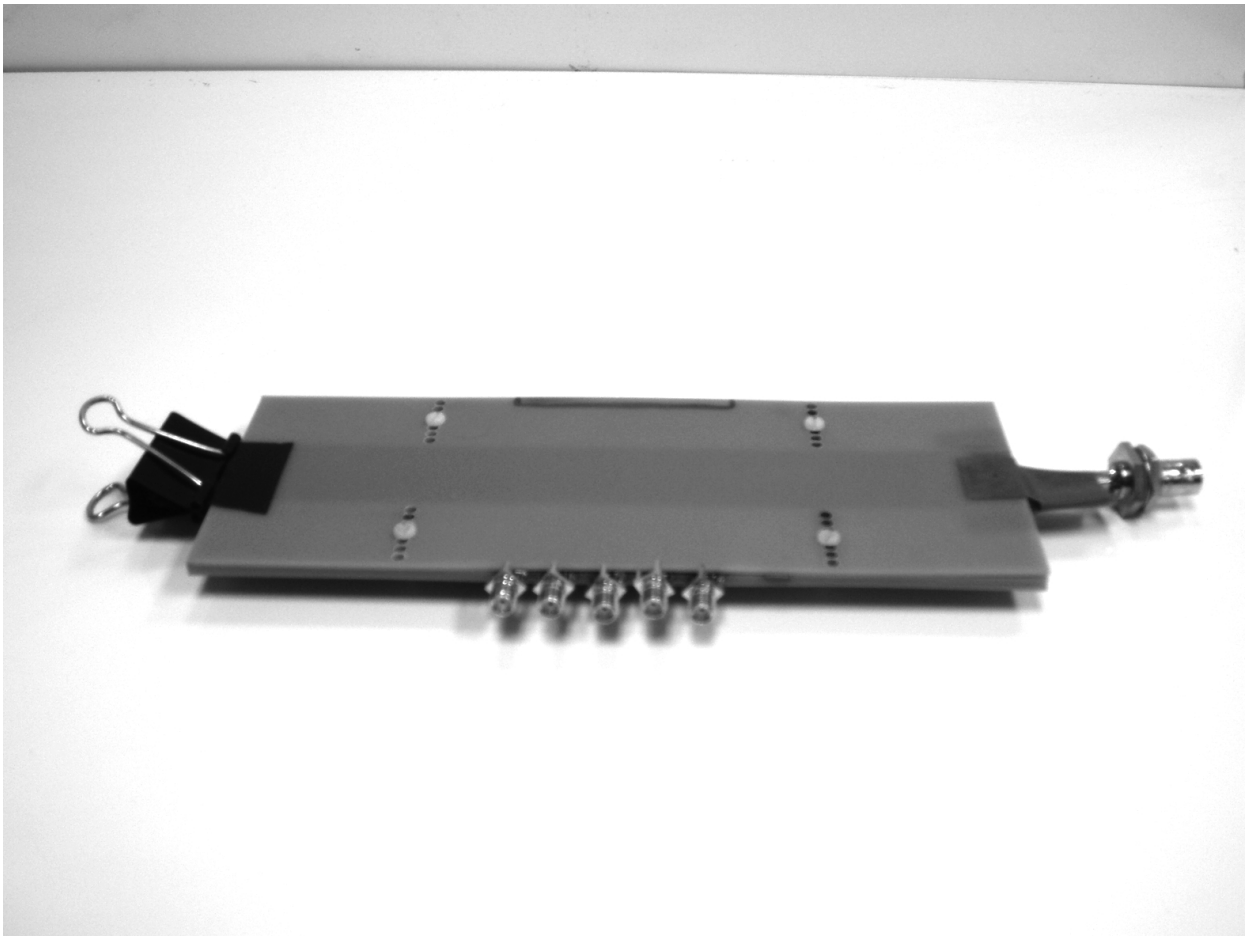

Figure 2.6: Final assembly of the strip line and the array probe board.

respective loops. This board allows for measurements at 0.1, 0.2, 0.3, 0.4, and 0.5 in offsets from the center.

Both probe boards have their output traces connected to SMA connectors for interfacing to a spectrum analyzer.

The instrumentation

The first set of data was sampled at 2 points per decade starting at 50 kHz and ending at 100 MHz. This set was taken with the movable probe board to minimize the capacitive contributions by the traces on the probe board. A HP 8656B Signal Generator was used to drive the system. Induced voltages in the probe loop were measured with a Advantest R3131A Spectrum Analyzer. The second set of measurements was sampled at 1 kHz, 5 kHz, 50 kHz, and 100 kHz. A PA04 power operational amplifier configured with a gain of 10 was driven by a HP 33120A Function Generator. The output of the amplifier was observed with a HP digital storage oscilloscope (DSO). The output of the power amplifier drives the strip line. A HP 4195A Spectrum Analyzer was used to measure the induced voltages in the probe loops. A student-made power supply was used to power the PA04 at ± 26 V and deliver the required current to the strip line.

For the high frequency measurements, the Spectrum Analyzer was configured to read in volts. The window was set to the center frequency of the driving signal and a window span of 50 kHz. The resolution bandwidth was 1 kHz. The HP 8656B was set to output a 1 V signal at the desired frequency.

The low frequency measurements were recorded from the HP4195A Spectrum Analyzer, again, configured to read in volts. The window was centered on the driving frequency, with a 100 Hz span and 30 Hz Resolution Bandwidth (RBW). The reference attenuation on the input of the spectrum analyzer was reduced to 0 dB. This lowers the noise floor for the instrument. The driving signal from the HP 33120A was adjusted such that the output of the PA04 was $1V_{p-p}$. The oscilloscope was used to verify the amplifier's output voltage under load. The DC impedance of the strip line assembly was measured to be 0.33Ω. Induced

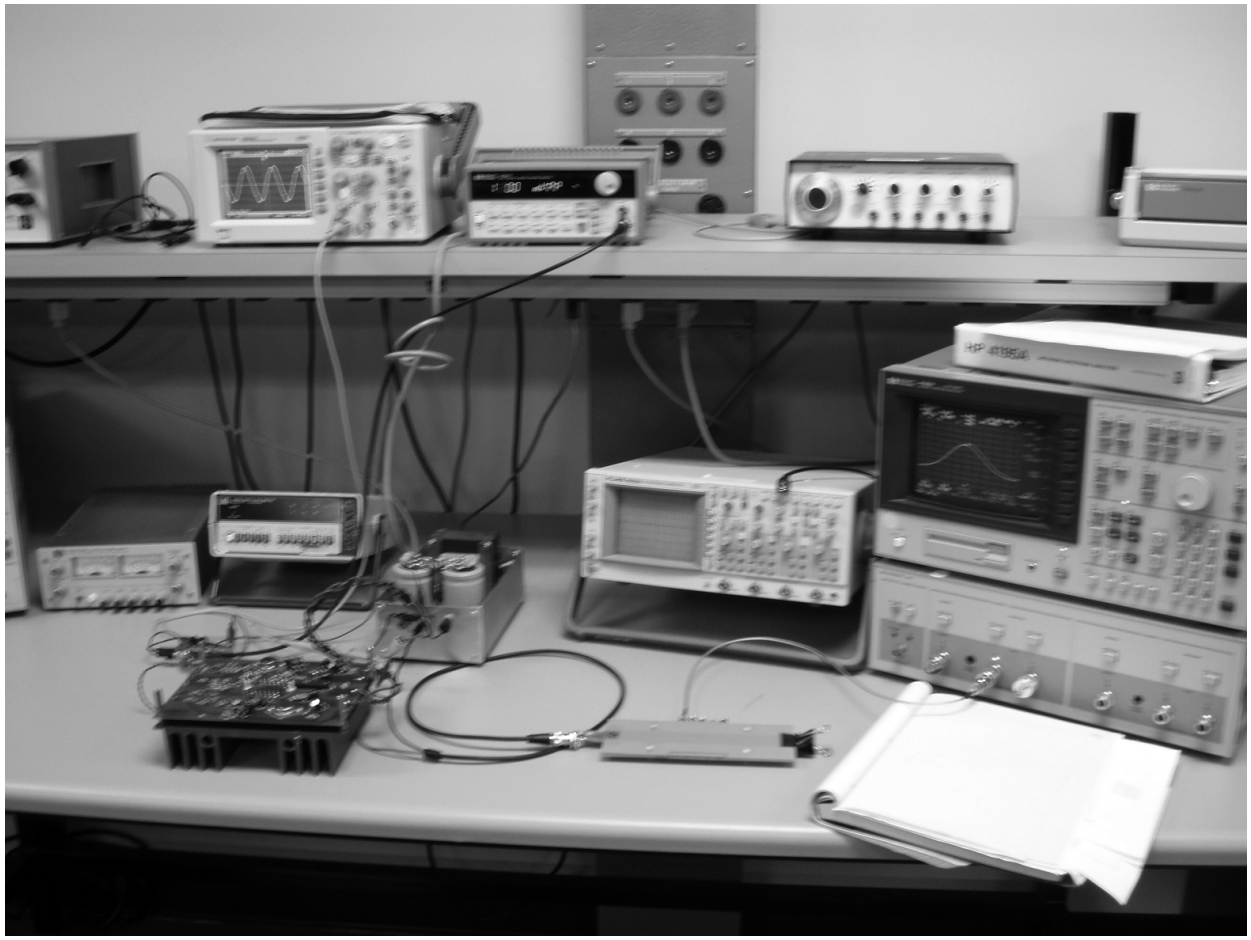

Figure 2.7: Test bench setup for the low frequency measurements.

voltages are proportional to both the current magnitude and the frequency.

Data reduction

For each frequency and probe location, 10 to 15 discrete samples of the probe voltage were recorded. Mean and standard deviations for the sets were recorded for data presentation. Data for each frequency is normalized to the induced voltage at that frequency at the center position. Standard deviation data is similarly normalized. In the case of the probe array board, which has no center position, the normalization voltage is taken by setting the voltage observed in the loop offset 0.254 cm from the center to match the percentage given in the `SPICE` simulation at that location (99.78% at 1 kHz; 99.97% at 100 kHz). This generates a value for the center position to divide into the values for the other locations to achieve the normalization. (This difference between the simulated center value and the simulated value at the first offset is less than the relative error in the measurement process.)

2.3.4 Comparison of observations and simulations

Measured voltages from the magnetic probes agree with the `SPICE` simulated induced voltages within the margin of error for the signal noise for the lower frequency data set. This is shown in Figure 2.8. Measurements made at 1 kHz confirm the uniform current distribution in the strip line. As the frequency increases, the simulations show skin effect altering the uniform nature of the current distribution and crowding the current to to the outside edges of the strip line. This results in the ratio of the probe voltages changing. The measured values follow this expected change in ratios; thus we can confirm that the `SPICE` model and solution for the current distribution is valid for these frequencies.

Figure 2.9 gives the data from the higher frequency data set. The higher frequency measurements show the limits of that particular measurement setup. At the low end of the frequencies (50 kHz), the magnitude of the induced voltages is very close to the noise floor of the Spectrum Analyzer, thus the large error. At the high end of the measurement frequencies, the strip line begins to experience wavelength effects. Capacitive contributions

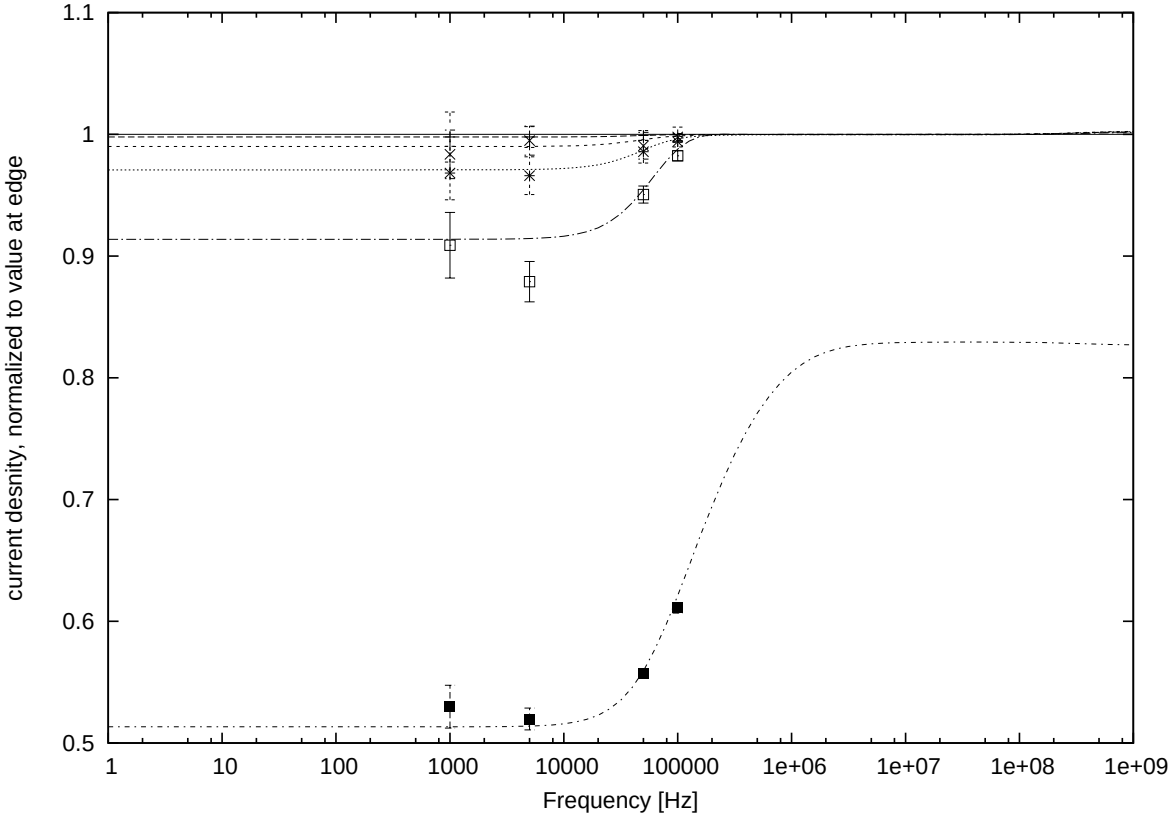

Figure 2.8: Comparison of simulated induced voltages in probes to values observed with the lower frequency instrumentation. Voltages have been normalized to the value at the center probe location. The family of curves show probe voltages as generated by `SPICE` with the center probe at the top and the probe location furthest from center at the bottom. Probe loops have 1/10 inch spacing. Experimental measurements of actual probe voltages are placed with error bars at the measured frequencies.

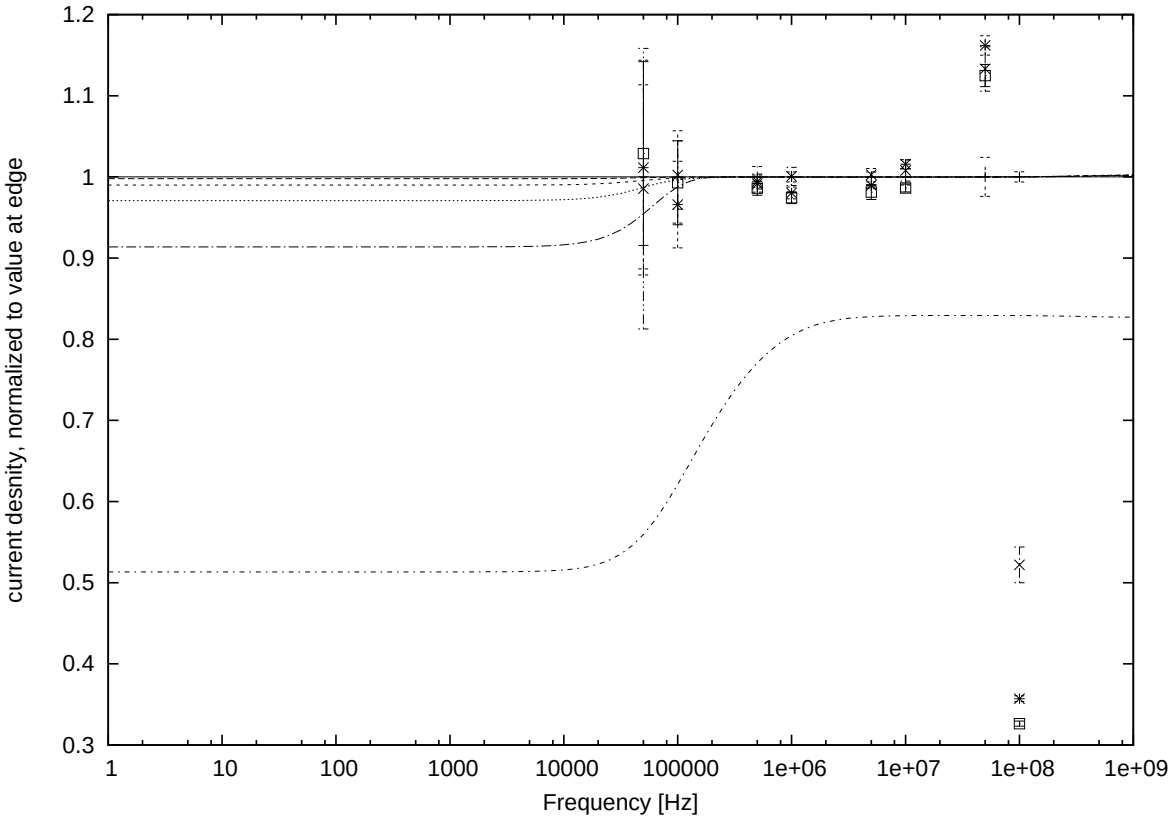

Figure 2.9: Comparison of simulated induced voltages in probes to values observed with the higher frequency instrumentation. Voltages have been normalized to the value at the center probe location. The family of curves show probe voltages as generated by `SPICE` with the center probe at the top and the probe location furthest from center at the bottom. Probe loops have 1/8 inch spacing. Experimental measurements of actual probe voltages are placed with error bars at the measured frequencies.

to the system can't be ignored, thus our resistive and inductive model fails at these high frequencies. The experimental setup produced good measurements between 100 kHz and 10 MHz. These measurements agree with the `SPICE` simulations for the induced voltages in the probes, again validating the `SPICE` model and simulations.

Because we have simulated probe voltages to compare to experimentally measured voltages, a current density reconstruction was not performed. To back-calculate current density distribution from the probe voltages requires assuming a functional form for the current density in terms of position across the conducting strip(s), and then fitting the parameters to the probe voltages by means of magnetic field calculations. The accuracy of the fitted curve improves with more data points from additional measurement locations. The nine probe locations used in these experimental measurements, along with comparison to the simulated probe voltages, are sufficient to validate the probe design and utility without performing a full current density distribution reconstruction.

Limits on measurable frequencies

These models work for cases where the length of the strip line is much less than the wavelength of the driving signal. Once the wavelength of the driving signal is near the length of the test strip, capacitance and wave effects would need to be considered for a complete simulation.

Practical limits on these measurements are bound by the amount of electrical current available to pass through the strip line. The PA04 operational amplifier was able to drive sufficient current at 1 kHz and 5 kHz to induce a voltage in the probe loop large enough to exceed the noise floor of the spectrum analyzer. The power amplifier is limited to lower frequency operation. Beyond this limit we are able to rely on the frequency term to drive the induced voltage in the probe loops. At higher frequencies in and above the megahertz range, a smaller current is required, eliminating the need for the power amplifier. At these frequencies, the strip line can be driven directly by an RF signal generator. This setup is sufficient to measure the magnetic flux through the loops until the frequency is such that

the wavelength of the signal in the strip is on the order of the length of the test strip.

2.3.5 Modifying the net list for pulse excitation

The magnetic probes are capable of giving signals from transient current pulses which can be reconstructed in the same way to get transient current distribution. However the instrumentation for observing the probe voltages must be compatible with the signal levels and frequency content of the pulses. The `SPICE` simulation is capable of simulating a pulse excitation by replacing the AC source with a pulse of the desired shape and modifying the analysis type from AC to Transient. These simulated probe voltages can be compared to measured probe voltages.

2.4 Summary

The magnetic current probes designed at LLNL allow localized observation of current distribution. Comparison to `SPICE` simulations based on the partial inductance method show consistency between the measurements and theory. The probe voltage measurements agree with those produced by the `SPICE` simulation. This method was tested against AC skin effect profiles. The results of the experimental verification show the induced voltages to be within the margin of error for the instrumentation used. The same `SPICE` netlist could now be expected to predict behavior of the strip line with pulse inputs and different loads, however different instrumentation would be required to experimentally test and verify the probe voltages. The method will now allow us a benchmark to compare with our customized current density solver.

Chapter 3

Theoretical basis for transient time-stepping solution

3.1 Introduction

In order to solve for a transient current density, the partial differential equation (1.12) is reformulated as an integrodifferential equation based on the concept of magnetic vector potential. This formulation for the total current density is solved by superposition of a source current density in the conductor and solving for an eddy current solution on the same space. The difference between the classic eddy current problem formulation and using a superposition of the source and eddy currents to solve for the total dynamic current density in a conductor is illustrated in figures 3.1 and 3.2. General eddy current studies have a driving field produced by currents outside the body of integration as in Figure 3.1. The source term is a separate and usually specified term. We wish to solve for both the driving term and the eddy solution. This approach gives us an expression for the total current density with the addition of the two solutions.

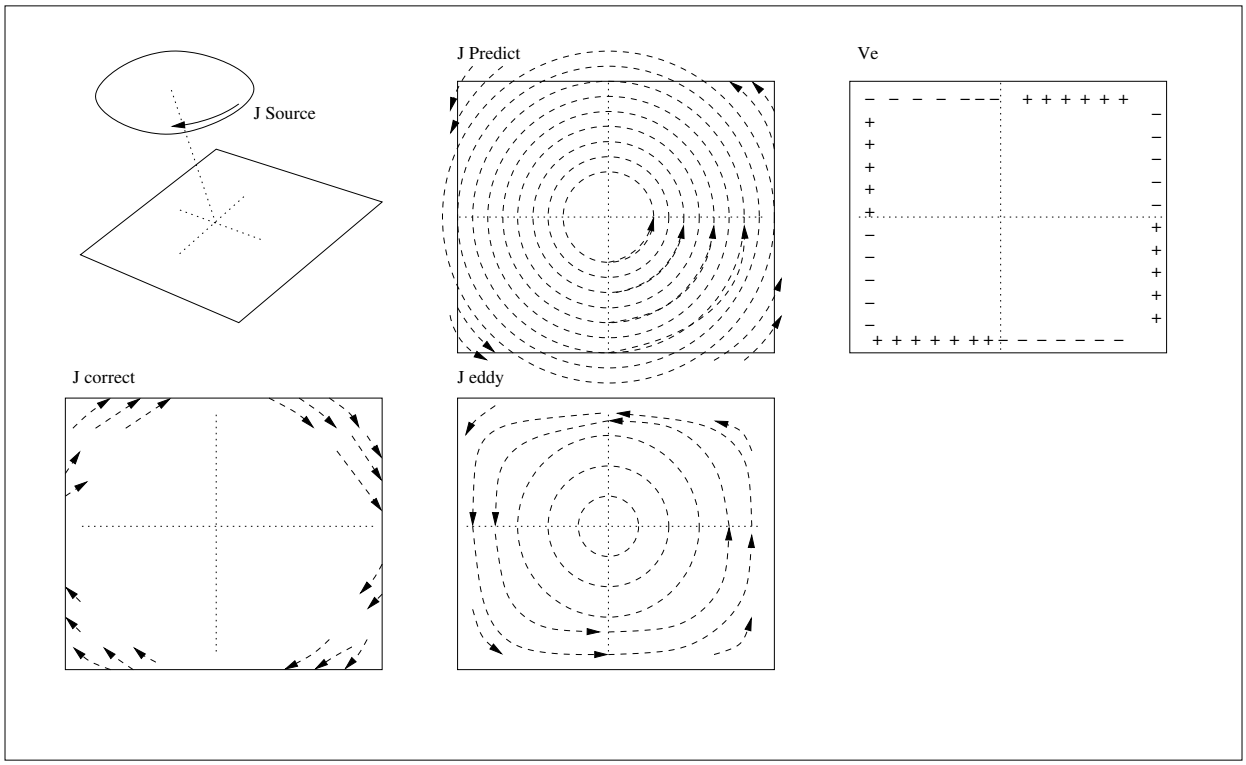

Figure 3.1: Eddy current problem with source current loop outside the body of integration.

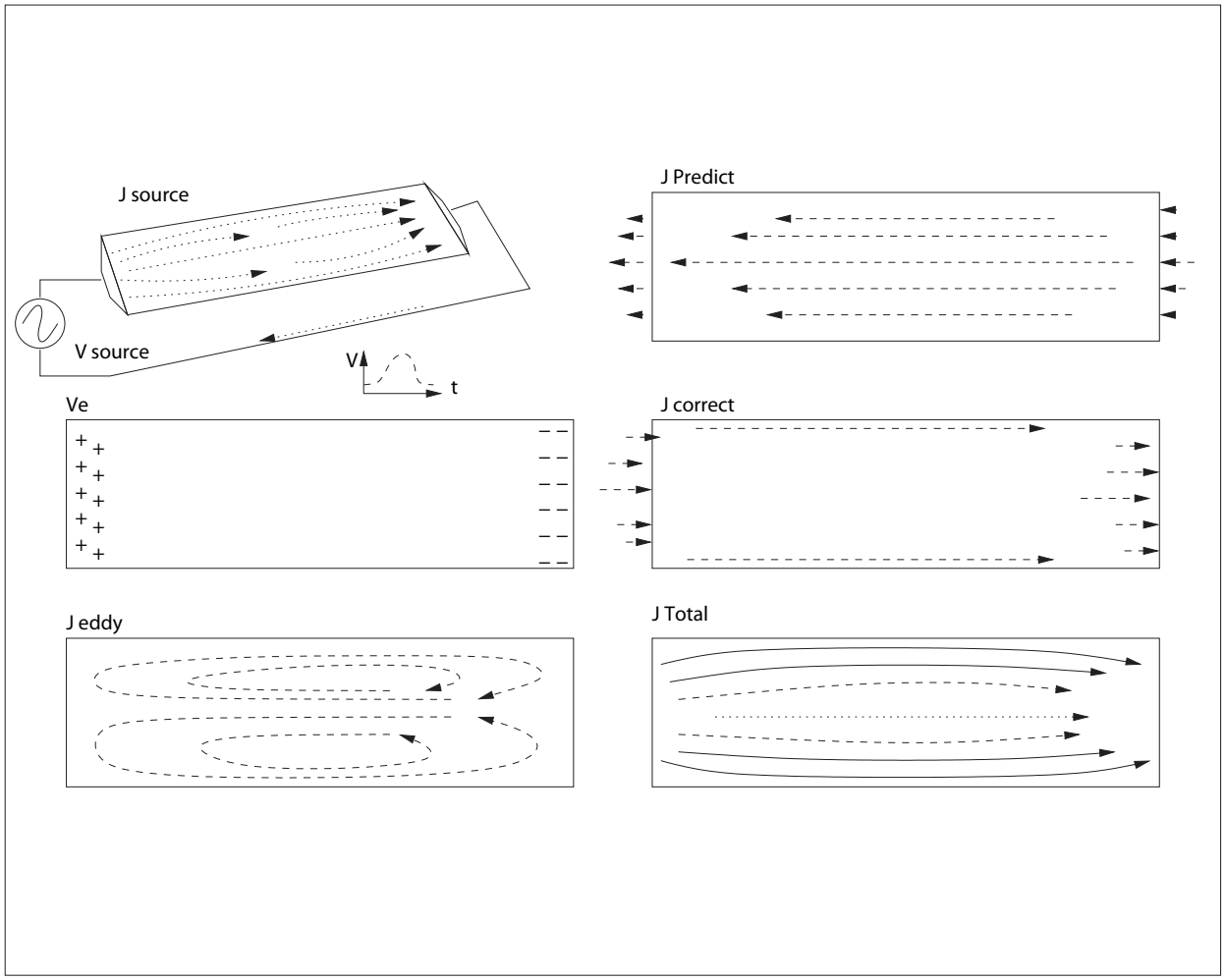

Figure 3.2: Superposition eddy current problem with source currents and eddy current occupying the same volume.

3.2 Derivation

Beginning with Maxwell's Equations[1], an integrodifferential equation system may be derived which is capable of producing a solution for the current density $\vec{J}$. Maxwell's Equations for electric and magnetic fields in a conductor where σ, ϵ are scalars and μ is constant, are

$$\nabla \cdot \vec{E}(\vec{r},t) = 0 \tag{3.1}$$

$$\nabla \cdot \vec{B}(\vec{r},t) = 0 \tag{3.2}$$

$$\nabla \times \vec{E}(\vec{r},t) = -\partial_t \vec{B}(\vec{r},t) \tag{3.3}$$

$$\nabla \times \vec{B}(\vec{r},t) = \mu\vec{J}(\vec{r},t) + \mu\partial_t \epsilon \vec{E}(\vec{r},t) \tag{3.4}$$

with auxiliary equations and potential definitions

$$\vec{J}(\vec{r},t) = \sigma(\vec{r},t)\vec{E}(\vec{r},t) \tag{3.5}$$

$$\vec{E}(\vec{r},t) = -\nabla V(\vec{r},t) - \partial_t \vec{A}(\vec{r},t) \tag{3.6}$$

$$\nabla \times \vec{A}(\vec{r},t) = \vec{B}(\vec{r},t) \tag{3.7}$$

$$\nabla \cdot \vec{A}(\vec{r},t) = 0. \tag{3.8}$$

As in Chapter 1.1.1 we restrict our derivation to good conductors ($\sigma >> 0$), there $\rho_v \approx 0$, and $\nabla \cdot \vec{J}(\vec{r},t) = 0$. We now solve Maxwell's equations for $\vec{J}$ in terms of potentials. Starting with Ampere's Law in differential form eq.(3.4) we substitute for the magnetic flux density in terms of the magnetic vector potential.

$$\nabla \times (\nabla \times \vec{A}(\vec{r},t)) = \mu\vec{J}(\vec{r},t) + \mu\partial_t \epsilon \vec{E}(\vec{r},t) \tag{3.9}$$

It is standard practice in eddy current problems to neglect the displacement current term, $\mu\epsilon\frac{\partial \vec{E}}{\partial t}$, at the frequencies of interest. This is the magnetoquasistatic state which we assumed in the earlier derivation. Applying vector identities to the curl of the curl of the magnetic vector potential we have

$$\nabla(\nabla \cdot \vec{A}(\vec{r},t)) - \nabla^2 \vec{A}(\vec{r},t) = \mu\vec{J}(\vec{r},t). \tag{3.10}$$

With the application of the Coulomb Gauge, eq.(3.8) , we are then able to write

$$\nabla^2 \vec{A}(\vec{r},t) = -\mu \vec{J}(\vec{r},t); \tag{3.11}$$

eq.(3.11) has the solution, in unbounded space

$$\vec{A}(\vec{r},t) = \frac{\mu}{4\pi}\int \frac{\vec{J}(\vec{r'},t)}{|\vec{r}-\vec{r'}|}d\vec{r'}, \tag{3.12}$$

or, with the integral restricted to a finite region, $\vec{A}_0$ is added to account for effects of other currents outside of the volume, which gives us a final integral form of

$$\vec{A}(\vec{r},t) = \frac{\mu}{4\pi}\int \frac{\vec{J}(\vec{r'},t)}{|\vec{r}-\vec{r'}|}d\vec{r'} + \vec{A}_0(\vec{r},t). \tag{3.13}$$

To solve for the total current density in a conductor we make use of the principle of superposition. First define the total current density as the sum of a source current density and that of the 'eddy' current density.

$$\vec{J}(\vec{r},t) = \vec{J}_s(\vec{r},t) + \vec{J}_e(\vec{r},t) \tag{3.14}$$

For our purposes the source current density will consist of the current density created by the application of the electrical potential to the driving edges and treating the instantaneous applied potential as if it was (DC) steady state. The eddy current density consists of all currents created by the time changing magnetic vector potential. The electric field defined by potentials is eq.(3.6). Rewriting eq.(3.6) in terms of $\vec{J}$ for both the source and eddy portions of the total current density, we arrive at

$$V(\vec{r},t) = V_s(\vec{r},t) + V_e(\vec{r},t) \tag{3.15}$$

$$\vec{A}(\vec{r},t) = \vec{A}_s(\vec{r},t) + \vec{A}_e(\vec{r},t) \tag{3.16}$$

$$\frac{\vec{J}(\vec{r},t)}{\sigma(\vec{r},t)} = -\nabla V(\vec{r},t) - \partial_t \vec{A}(\vec{r},t) \tag{3.17}$$

$$\vec{J}_s(\vec{r},t) = -\sigma(\vec{r},t)\nabla V_s(\vec{r},t). \tag{3.18}$$

$\vec{A}_0(\vec{r},t) = 0$ in the equation for the source current density because magnetic effects from current densities outside the conductor are negligible. Collecting the terms,

$$\vec{J}_e(\vec{r},t) = -\sigma(\vec{r},t)\nabla V_e(\vec{r},t) - \sigma(\vec{r},t)\partial_t(\vec{A}_e(\vec{r},t) + \vec{A}_s(\vec{r},t)) \tag{3.19}$$

$$\vec{J}_e(\vec{r},t) = -\sigma(\vec{r},t)\nabla V_e(\vec{r},t) - \sigma(\vec{r},t)\partial_t\left(\frac{\mu}{4\pi}\int\frac{\vec{J}_e(\vec{r'},t)}{|\vec{r}-\vec{r'}|}d\vec{r'} + \frac{\mu}{4\pi}\int\frac{\vec{J}_s(\vec{r'},t)}{|\vec{r}-\vec{r'}|}d\vec{r'}\right) \tag{3.20}$$

where V_e is the potential required to ensure the boundary condition of $\vec{J}_e \cdot \hat{n} = 0$ on the conductor edges.

3.3 Discretization in time

To solve this equation for the electrical current density in the conductor in a transient sense, we need a method for time stepping. We have selected the Fractional Step Method as demonstrated by Koizumi[45] and which has been used in computational fluid dynamics problems for many years. The process separates the solution into several steps. The first step is to calculate a 'prediction' current density which is due to the source current density, previous time eddy current density, and the previous time 'eddy potential'. This prediction current density will not be constrained within the conductor as it knows nothing about the boundaries. The next step is to calculate the change in the eddy potential. If the prediction current density were trying to leave the conductor, there would be a build up of charges on the edges of the conductor. This charge build up creates a potential, which we call $V_e(r,t)$. The eddy potential serves to enforce the boundary condition. The eddy potential is then used to calculate the 'correction current'. This current density when added to the prediction current density ensures correct current density boundary conditions normal to the conductor edge. The prediction combined with the correction current density becomes the total eddy current density for the new time step. Then the quantities are advanced in time and we repeat the process. In this specific case, to find the total current density, we would add the source current density and the eddy solution at the end of each time step.

The following definitions show the relationship between the time stepping parameters:

$$\vec{J}_e^t(\vec{r}) = \vec{J}_e^*(\vec{r}) + \delta J(\vec{r}) \tag{3.21}$$

Eddy current density at time t equals prediction plus correction current densities. Next the eddy voltage at time t equals previous time value plus incremental change

$$V_e(\vec{r}, t) = V_e^t(\vec{r}) = V_e^{t-\tau}(\vec{r}) + \delta V(\vec{r}). \tag{3.22}$$

Finally, from the definition of $\vec{J}$ we write an expression for the differential change in current

$$\delta J(\vec{r}) = -\sigma(\vec{r})\nabla\delta V(\vec{r}). \tag{3.23}$$

$\nabla \cdot \vec{J}(\vec{r}, t) = 0$ and $\nabla \cdot \vec{J}_s(\vec{r}, t) = 0$ lead to

$$\nabla \cdot (\sigma(\vec{r})\nabla\delta V(\vec{r})) = \nabla \cdot \vec{J}_e^*(\vec{r}). \tag{3.24}$$

To advance the time by one step in eq.(3.20) by the fractional step method, we make substitutions with definitions eq.(3.21) through eq.(3.23) and arrive at expressions for the prediction current density, the differential change in the eddy voltage, and the correction current density. The fractional step method allows us to replace the current density term with a mixture of current density from the new and previous time steps. The fractional step parameter θ, can be adjusted if stability issues arise in the integrations. The Crank-Nicholson[46] scheme has a θ value of 0.5 and for a full implicit time integration scheme θ equals unity.

$$\vec{J}_e(\vec{r}, t) = \theta\vec{J}_e^t(\vec{r}) + (1-\theta)\vec{J}_e^{t-\tau}(\vec{r}) \tag{3.25}$$

We will also use a backward differencing scheme for time derivatives,

$$\frac{\partial X}{\partial t} = \frac{X^t - X^{t-\tau}}{\tau}. \tag{3.26}$$

Making the substitutions for the time derivatives in eq.(3.20), and using Ψ to represent the Free Space Green's Function inside the integrals, we have

$$\vec{J}_e(\vec{r}, t) = -\sigma(\vec{r})\nabla V_e(\vec{r}, t) - \sigma(\vec{r}, t)\left(\mu\int\frac{\vec{J}_e^t(\vec{r'}) - \vec{J}_e^{t-\tau}(\vec{r'})}{\tau}\Psi d\vec{r'} + \mu\int\frac{\vec{J}_s^t(\vec{r'}) - \vec{J}_s^{t-\tau}(\vec{r'})}{\tau}\Psi d\vec{r'}\right) \tag{3.27}$$

Next we substitute the fractional stepped form of the eddy current density into eq.(3.27) on terms outside the time derivatives, and multiply through by the time step size τ.

$$\begin{aligned}\tau\theta\vec{J}_e^{\,t}(\vec{r})+\tau(1-\theta)\vec{J}_e^{\,t-\tau}(\vec{r}) = &-\tau\sigma(\vec{r},t)\nabla V_e(\vec{r},t)\\ &-\left(\sigma(\vec{r},t)\mu\int(\vec{J}_e^{\,t}(\vec{r'})-\vec{J}_e^{\,t-\tau}(\vec{r'}))\Psi d\vec{r'}+\sigma(\vec{r},t)\mu\int(\vec{J}_s^{\,t}(\vec{r'})-\vec{J}_s^{\,t-\tau}(\vec{r'}))\Psi d\vec{r'}\right)\end{aligned} \tag{3.28}$$

Now we collect the eddy current density at time t on the left hand side of the equation and the previous time on the right hand side of the equation.

$$\begin{aligned}\tau\theta\vec{J}_e^{\,t}(\vec{r})+\sigma(\vec{r},t)\mu\int\vec{J}_e^{\,t}(\vec{r'})\Psi d\vec{r'} = &-\tau(1-\theta)\vec{J}_e^{\,t-\tau}(\vec{r})-\tau\sigma(\vec{r},t)\nabla V_e(\vec{r},t)+\sigma(\vec{r},t)\mu\int\vec{J}_e^{\,t-\tau}(\vec{r'})\Psi d\vec{r'}\\ &-\sigma(\vec{r},t)\mu\int(\vec{J}_s^{\,t}(\vec{r'})-\vec{J}_s^{\,t-\tau}(\vec{r'}))\Psi d\vec{r'}\end{aligned} \tag{3.29}$$

We must now implement the prediction and correction steps to solve for $\vec{J}_e^{\,t}$. We do not yet have a self consistant eddy voltage and eddy current density. Replacing the ∇V_e in eq.(3.29) with the eddy voltage from the last time step will give us a prediction of what the new time step's eddy current density could be

$$\begin{aligned}\tau\theta\vec{J}_e^{\,*}(\vec{r})+\mu\sigma(\vec{r},t)\int_c\vec{J}_e^{\,*}(\vec{r'})\Psi d\vec{r'} =&\mu\sigma(\vec{r},t)\int_c\vec{J}_e^{\,t-\tau}(\vec{r'})\Psi d\vec{r'}-\mu\sigma(\vec{r},t)\int_s(\vec{J}_s^{\,t}(\vec{r'})-\vec{J}_s^{\,t-\tau}(\vec{r'}))\Psi d\vec{r'}\\ &-\tau(1-\theta)\vec{J}_e^{\,t-\tau}(\vec{r})-\tau\sigma(\vec{r},t)\nabla V_e^{\,t-\tau}(\vec{r}).\end{aligned} \tag{3.30}$$

With the exception of $\sigma(\vec{r},t)$,all the known quantities are on the right hand side of the equation, and the unknown quantity, $\vec{J}_e^{\,*}$ is on the left. For simple cases we can assume $\sigma(\vec{r},t)$ is known. In future versions where time changing conductivity is added, we would need to substitute in the conductivity at the previous time step, $\sigma(\vec{r},t-\tau)$ in these equations and ensure that the change in conductivity per time iteration is small. Once we have a solution for the prediction current density, we calculate the new eddy potential from the incremental change in the voltage that would be produced by the charges deposited on the conductor edge by the prediction current density and enforce

$$\nabla\cdot(\nabla\delta V(\vec{r})) = \nabla\cdot\vec{J}_e^{\,*}(\vec{r}) \tag{3.31}$$

where the boundary conditions are $\vec{J}_e^* \cdot \hat{n} = 0$ on $\partial\Omega$. The total eddy voltage for the new time step becomes

$$V_e^t(\vec{r}) = V_e^{t-\tau}(\vec{r}) + \delta V(\vec{r}). \tag{3.32}$$

The correction current density is generated by the incremental change in the eddy voltage. This correction term when added to the prediction term will guarantee the boundary condition on the edges of the conductor, and that $\nabla \cdot \vec{J} = 0$.

$$\delta\vec{J}(\vec{r}) = -\sigma(\vec{r}, t)\nabla\delta V(\vec{r}) \tag{3.33}$$

We are now able to calculate the eddy current density for the new timestep as the sum of the prediction and correction current densities.

$$\vec{J}_e^t(\vec{r}) = \vec{J}_e^*(\vec{r}) + \delta\vec{J}(\vec{r}) \tag{3.34}$$

The total current density in the conductor is then the sum of the eddy current distribution and the source current distribution for the same time step t.

$$\vec{J}^t(\vec{r}) = \vec{J}_e^t(\vec{r}) + \vec{J}_s^t(\vec{r}) \tag{3.35}$$

3.4 Finite element implementation of the transient eddy current problem

Symbol	Definition
$\sum_m$	Sum over all elements m.
$\int_{\Delta\Omega_m}$	Integral over sub-volume m.
$d\Omega$	Volume integral over the domain Ω.
$\hat{e}_{pi}$	Unit vector parallel to edge i from which $\lvert\vec{r}-\vec{r'}\rvert$ is measured in the Free Space Green's Function
$[E]$	Identity matrix
$\vec{N}_i$	Edge element shape function on edge 'i'
ϕ_i	Nodal basis function on node 'i'

Table 3.1: Definition of terms found in the construction of the Coefficient Matrices

Now that the solution for the eddy current density has been discretized in time by the application of the fractional step method, the equations need to be rewritten for implementation with a Finite Element Method.

The Finite Element Method[19] allows us to represent a continuous function on a mesh. For the case of a 2-dimensional surface, the mesh will consist of many triangular elements joined at nodes. Scalar values, such as voltages, are evaluated on the nodes, and the function for the voltage anywhere within a particular element is the interpolation of the nodal values using basis functions associated with the solution. Edges of the triangular elements then can represent vector quantities associated with the direction of the particular edge. Different basis functions are used to represent the vector field within a triangular element as sums of the edge values and those edge element shape functions. The derivation of the edge element shape functions is shown in Section 3.5 Again, a vector function can be evaluated anywhere within a particular triangular element by the summation of the edge values of the triangular element and its associated edge shape functions. Magnetic vector potential and current

density are represented by:

$$\vec{A} = \sum_i \vec{N}_i A_i \tag{3.36}$$

$$\vec{J} = \sum_i \vec{N}_i J_i \tag{3.37}$$

where $\vec{N}_i$ is the linear edge element shape function for a triangle. The scalar electrical potential has the standard scalar finite element definition of

$$V = \sum_i \phi_i V_i, \tag{3.38}$$

where ϕ_i is the standard linear shape function.

The nodal basis functions for triangular elements and the edge element shape functions are discussed in more detail in Section 3.5. This general form can be extended to other mesh elements such as square, or rectilinear elements in two-dimensions, or prisms, pyramids, cubes, and others in three-dimensional meshes. For our purposes of solving for the current density in thin conductors, we are able to use two-dimensional representations for the surfaces.

Recall that the Green's function integrals in eq.(3.30) are the magnetic vector potential $\vec{A}$ due to the source and eddy current densities. Before we make the finite element approximation substitutions we have

$$\begin{aligned} \tau\theta\vec{J}_e^*(\vec{r}) + \mu\sigma(\vec{r})\vec{A}_e^*(\vec{r}) = \mu\sigma(\vec{r})\vec{A_e^{t-\tau}}(\vec{r}) - \mu\sigma(\vec{r})(\vec{A_s^t}(\vec{r}) - \vec{A_s^{t-\tau}}(\vec{r})) \\ - \tau(1-\theta)\vec{J_e^{t-\tau}}(\vec{r}) - \tau\sigma(\vec{r})\nabla V_e^{t-\tau}(\vec{r}). \end{aligned}, \tag{3.39}$$

where

$$\vec{A}_e^t(\vec{r}) = \frac{\mu}{4\pi}\int \frac{\vec{J}_e^t(\vec{r'})}{|\vec{r} - \vec{r'}|} d\vec{r'}, \tag{3.40}$$

etc. Dividing through by conductivity reduces the number of terms dependent on the material property. To simplify the notation, resistivity is used rather than the inverse of conductivity. Moving this term is useful when this formula is implemented in computer code

because if we allow for changing conductivity, every term with an updated conductivity will require recalculation. This minor alteration greatly reduces computation time.

$$\begin{aligned}\tau\theta\rho(\vec{r},t)\vec{J}_e^*(\vec{r})+\mu\vec{A}_e^*(\vec{r}) = \mu\vec{A_e^{t-\tau}}(\vec{r}) - \mu(\vec{A_s^t}(\vec{r}) - \vec{A_s^{t-\tau}}(\vec{r})) \\ - \tau(1-\theta)\rho(\vec{r},t)\vec{J}_e^{t-\tau}(\vec{r}) - \tau\nabla V_e^{t-\tau}(\vec{r})\end{aligned} \tag{3.41}$$

To express eq.(3.30) with the finite element approximation, we substitute the relations eq.(3.37), eq.(3.36) and eq.(3.38).

$$\begin{aligned}\tau\theta\rho(\vec{r},t)\sum_i^E \vec{N}_i J_{ei}^* + \mu\sum_i^E \vec{N}_i A_{ei}^* = \mu\sum_i^E \vec{N}_i A_{ei}^{t-\tau} \\ - \mu\sum_i^E \vec{N}_i(A_{si}^t - A_{si}^{t-\tau}) - \tau\rho(\vec{r},t)(1-\theta)\sum_i^E \vec{N}_i J_{ei}^{t-\tau} - \tau\sum_i^N \nabla\phi_i V_{ei}^{t-\tau}\end{aligned} \tag{3.42}$$

Superscripts on the summations 'E' refers to the number of edges and 'N' is the number of nodes in the finite element mesh. Next we substitute for the discrete edge quantity A_i. In its vector form, $\vec{A}$ is a three component vector which is constructed by the integration of current densities at point $\vec{r'}$. We can treat each individual vector components separately. The x component of $\vec{A}$ can be written as

$$A_x(\vec{r}) = \frac{\mu}{4\pi}\int \frac{J_x(\vec{r'})}{|\vec{r}-\vec{r'}|}d\vec{r'}. \tag{3.43}$$

In the finite element approximation for $\vec{A}$, A_i is the discretized value on an edge of the triangular element. It is no longer a three component vector, but is directed in a single direction associated with the unit vector of that edge. In the finite element approximation, this unit vector is contained within the normal linear shape function $\vec{N}_i$. More details on the nature of $\vec{N}_i$ are given in Section 3.5 below. To represent this single direction in the integral of the current density, we dot product the current density vector with the unit vector of the discrete edge, $\hat{e}_{pi}$. Only current densities parallel to the edge contribute to the discrete magnetic vector potential component at that point. Furthermore, in the discrete approximation for $\vec{A}$, we set the quantity on the edge to be constant along the edge. We choose the center point of the edge as the evaluation point $\vec{r}$ for the integral. A more exact

approximation would be to perform a six-dimensional integral evaluating the contributions of the current at $\vec{r'}$ to the line at $\vec{r}$. This is the same calculation performed in the partial inductance method in Chapter 2. Because we treat the edge value A_i as a constant on the edge, we would have to average the calculated contributions of the integral along the edge which is approximately equal to evaluating the integral $d\vec{r'}$ with $\vec{r}$ fixed at the center point of the edge. Therefore we have the general form for the calculation of the magnetic vector potential component on a discrete edge as

$$A_i = \frac{\mu}{4\pi} \int \frac{\vec{J}(\vec{r'}) \cdot \hat{e}_{pi}}{|\vec{r}_c - \vec{r'}|} d\vec{r'} \tag{3.44}$$

where point $\vec{r}_c$ is located at the center of edge i, which is bounded by nodes i and j.

$$\vec{r}_c = (\frac{x_j + x_i}{2}\hat{x} + \frac{y_j + y_i}{2}\hat{y} + \frac{z_j + z_i}{2}\hat{z}) \tag{3.45}$$

Inserting eq.(3.44) into eq.(3.42) we have

$$\begin{aligned}\tau\theta\rho(\vec{r},t)\sum_i^E \vec{N}_i J_{ei}^* + \mu\sum_i^E \vec{N}_i \left(\int_c \frac{\vec{J}_e^* \cdot \hat{e}_{pi}}{4\pi|\vec{r} - \vec{r'}|} d\vec{r'}\right) = \mu\sum_i^E \vec{N}_i \left(\int_c \frac{\vec{J}_e^{t-\tau} \cdot \hat{e}_{pi}}{4\pi|\vec{r} - \vec{r'}|} d\vec{r'}\right) \\ - \mu\sum_i^E \vec{N}_i \left(\int_s \frac{(\vec{J}_s^t - \vec{J}_s^{t-\tau}) \cdot \hat{e}_{pi}}{4\pi|\vec{r} - \vec{r'}|} d\vec{r'}\right) - \tau\rho(\vec{r},t)(1-\theta)\sum_i^E \vec{N}_i J_{ei}^{t-\tau} - \tau\sum_i^N \nabla\phi_i V_{ei}^{t-\tau}.\end{aligned} \tag{3.46}$$

To simplify this equation, we again use the substitution $\Psi = \frac{1}{4\pi|\vec{r} - \vec{r'}|}$.

$$\begin{aligned}\tau\theta\rho(\vec{r},t)\sum_i^E \vec{N}_i J_{ei}^* + \mu\sum_i^E \vec{N}_i \left(\int_c \vec{J}_e^* \cdot \hat{e}_{pi}\Psi d\vec{r'}\right) = \mu\sum_i^E \vec{N}_i \left(\int_c \vec{J}_e^{t-\tau} \cdot \hat{e}_{pi}\Psi d\vec{r'}\right) \\ - \mu\sum_i^E \vec{N}_i \left(\int_s (\vec{J}_s^t - \vec{J}_s^{t-\tau}) \cdot \hat{e}_{pi}\Psi d\vec{r'}\right) - \tau\rho(\vec{r},t)(1-\theta)\sum_i^E \vec{N}_i J_{ei}^{t-\tau} - \tau\sum_i^N \nabla\phi_i V_{ei}^{t-\tau}\end{aligned} \tag{3.47}$$

The current density terms inside the integrals can be replaced with their finite element approximations. At every edge term i, we have the contribution of the magnetic vector potential, which is generated from all the other current densities. This is represented as a

sum with the index of j to indicate a separate iterator from i.

$$\tau\theta\rho(\vec{r},t)\sum_{i}^{E}\vec{N}_i J^*_{ei}+\mu\sum_{i}^{E}\vec{N}_i\left(\int_c\sum_{j}^{E}\vec{N}_j J^*_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)=\mu\sum_{i}^{E}\vec{N}_i\left(\int_c\sum_{j}^{E}\vec{N}_j J^{t-\tau}_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)$$
$$-\mu\sum_{i}^{E}\vec{N}_i\left(\int_s\sum_{j}^{E}\vec{N}_j(J^t_{sj}-J^{t-\tau}_{sj})\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)-\tau\rho(\vec{r},t)(1-\theta)\sum_{i}^{E}\vec{N}_i J^{t-\tau}_{ei}-\tau\sum_{i}^{N}\nabla\phi_i V^{t-\tau}_{ei} \tag{3.48}$$

Multiply every term by every other shape function as a dot product.

$$\begin{aligned}&\tau\theta\rho(\vec{r},t)\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i J^*_{ei}+\mu\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i\left(\int_c\sum_{j}^{E}\vec{N}_j J^*_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)=\\&\mu\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i\left(\int_c\sum_{j}^{E}\vec{N}_j J^{t-\tau}_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)\\&-\mu\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i\left(\int_s\sum_{j}^{E}\vec{N}_j(J^t_{sj}-J^{t-\tau}_{sj})\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)\\&-\tau\rho(\vec{r},t)(1-\theta)\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i J^{t-\tau}_{ei}\\&-\tau\sum_{i}^{N}\vec{N}_k\cdot\nabla\phi_i V^{t-\tau}_{ei}\end{aligned} \tag{3.49}$$

Integrate eq.(3.49) with respect to $\vec{r}$ for the entire domain Ω (but not $\vec{r'}$). This leaves us with only the non-zero terms.

$$\begin{aligned}&\tau\theta\int_\Omega\rho(\vec{r},t)\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i d\vec{r}J^*_{ei}+\mu\int_\Omega\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i d\vec{r}\left(\int_c\sum_{j}^{E}\vec{N}_j J^*_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)=\\&\mu\int_\Omega\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i d\vec{r}\left(\int_c\sum_{j}^{E}\vec{N}_j J^{t-\tau}_{ej}\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)\\&-\mu\int_\Omega\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i d\vec{r}\left(\int_s\sum_{j}^{E}\vec{N}_j(J^t_{sj}-J^{t-\tau}_{sj})\cdot\hat{e}_{pi}\Psi d\vec{r'}\right)\\&-\tau(1-\theta)\int_\Omega\rho(\vec{r},t)\sum_{i}^{E}\vec{N}_k\cdot\vec{N}_i d\vec{r}J^{t-\tau}_{ei}\\&-\tau\int_\Omega\sum_{i}^{N}\vec{N}_k\cdot\nabla\phi_i d\vec{r}V^{t-\tau}_{ei}\end{aligned} \tag{3.50}$$

We are able to move the discrete variables outside the summations and integrals and rewrite them as column vectors. The order of the integral and the sum can be reordered, so we are taking the sum of integral quantities, which is the same as an integral of a sum. Here the square brackets denote a matrix, $[X]$, and the curly brackets denote a column vector, $\{X\}$.

$$\begin{aligned}
&\tau\theta\left[\sum_i^E \int_\Omega \rho(\vec{r},t)\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\{J_e^*\} + \mu\left[\sum_i^E \int_\Omega \vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left\{\sum_j^E \int_c \vec{N}_j\cdot\hat{e}_{pi}J_{ej}^*\Psi d\vec{r'}\right\} = \\
&\mu\left[\sum_i^E \int_\Omega \vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left\{\sum_j^E \int_c \vec{N}_j\cdot\hat{e}_{pi}J_{ej}^{t-\tau}\Psi d\vec{r'}\right\} \\
&-\mu\left[\sum_i^E \int_\Omega \vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left\{\sum_j^E \int_s \vec{N}_j\cdot\hat{e}_{pi}(J_{sj}^{t}-J_{sj}^{t-\tau})\Psi d\vec{r'}\right\} \\
&-\tau(1-\theta)\left[\sum_i^E \int_\Omega \rho(\vec{r},t)\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\{J_e^{t-\tau}\} \\
&-\tau\left[\sum_i^N \int_\Omega \vec{N}_k\cdot\nabla\phi_i d\vec{r}\right]\{V_e^{t-\tau}\}
\end{aligned} \tag{3.51}$$

The Green's function finite element approximations are a column vector of values of A_i on the edges. These column vectors are themselves formed by the matrix multiplication of their respective current density column vectors and the matrix formed by the evaluation of the sums of the Green's function integrals. We next perform the same extraction of the discrete variables from inside the Green's function integrals and are left with the matrix equation

form for calculating the prediction current density.

$$\tau\theta\left[\sum_{i}^{E}\int_{\Omega}\rho(\vec{r},t)\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\{J_e^*\}+\mu\left[\sum_{i}^{E}\int_{\Omega}\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left[\sum_{j}^{E}\int_{c}\vec{N}_j\cdot\hat{e}_{pi}\Psi d\vec{r'}\right]\{J_e^*\}=$$
$$\mu\left[\sum_{i}^{E}\int_{\Omega}\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left[\sum_{j}^{E}\int_{c}\vec{N}_j\cdot\hat{e}_{pi}\Psi d\vec{r'}\right]\{J_e^{t-\tau}\}$$
$$-\mu\left[\sum_{i}^{E}\int_{\Omega}\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\left[\sum_{j}^{E}\int_{s}\vec{N}_j\cdot\hat{e}_{pi}\Psi d\vec{r'}\right]\{J_s^t-J_s^{t-\tau}\}$$
$$-\tau(1-\theta)\left[\sum_{i}^{E}\int_{\Omega}\rho(\vec{r},t)\vec{N}_k\cdot\vec{N}_i d\vec{r}\right]\{J_e^{t-\tau}\}$$
$$-\tau\left[\sum_{i}^{N}\int_{\Omega}\vec{N}_k\cdot\nabla\phi_i d\vec{r}\right]\{V_e^{t-\tau}\} \tag{3.52}$$

Provided the resistivity of the solution space is constant, we can express eq.(3.52) in a simplified matrix form[45].

$$[S]\{\theta\tau\rho[E]+\mu_0[P_e]\}\{J_e^*\}=\mu_0[S]\{[P_0]\{J_e^{t-\tau}\}-[P_e]\{J_s^t-J_s^{t-\tau}\}\}-(1-\theta)\tau\rho[S]\{J_e^{t-\tau}\}-\tau[G]\{V^{t-\tau}\}. \tag{3.53}$$

Eq.(3.31) and eq.(3.23) become:

$$[D]\{\delta V\}=[G]^T\{J_e^*\} \tag{3.54}$$

$$[S]\{\delta J\}=-\sigma[G]\{\delta V\} \tag{3.55}$$

where $[X]$ denotes a matrix, and $\{X\}$ denotes a column vector. Elements of coefficient matrices are then defined as

$$[E]=identity\ matrix \tag{3.56}$$

$$S_{ij}=\sum_{m}\int_{\Delta\Omega_m}\vec{N}_j\cdot\vec{N}_i d\Omega \tag{3.57}$$

$$P_{0ij}=\sum_{m}\int_{\Delta\Omega_m}\vec{N}_j\cdot\hat{e}_{pi}\Psi d\Omega_0 \tag{3.58}$$

$$P_{eij}=\sum_{m}\int_{\Delta\Omega_m}\vec{N}_j\cdot\hat{e}_{pi}\Psi d\Omega_e \tag{3.59}$$

$$G_{ij} = \sum_m \int_{\Delta\Omega_m} \vec{N}_i \cdot \nabla\phi_j d\Omega \tag{3.60}$$

$$D_{ij} = \sum_m \int_{\Delta\Omega_m} \sigma(\nabla\phi_j \cdot \nabla\phi_i) d\Omega \tag{3.61}$$

with the free space Green's Function being represented by

$$\Psi = \frac{1}{4\pi|\vec{r} - \vec{r'}|}. \tag{3.62}$$

Terms used in the matrix construction definitions are summarized in Table 3.1. Matrices $[P_0]$ and $[P_e]$ are the same form but they can have different volumes over which they operate resulting in separate matrices, of possibly different sizes if source current density exist outside the conductor. In the case of the self consistant current density solver, the source and the eddy current densities are superimposed on the same geometrical mesh, resulting in the same $[P]$ matrix for both terms. For the case of nonuniform resistivity, we can not extract $\rho(\vec{r}, t)$ from inside the integrand over each element. We rewrite eq.(3.51) as a modification of eq.(3.53),

$$[\theta\tau[pS]+\mu_0[S][P_e]]\{J_e^*\} = \mu_0[S]\{[P_0]\{J_e^{t-\tau}\}-[P_e]\{J_s^t-J_s^{t-\tau}\}\}-(1-\theta)\tau[pS]\{J_e^{t-\tau}\}-\tau[G]\{V^{t-\tau}\}. \tag{3.63}$$

Alterations to eq.(3.64) are also required.

$$[S]\{\delta J\} = -[cG]\{\delta V\} \tag{3.64}$$

with the addition of the matrix definitions for $[pS]$ and $[cG]$ as

$$pS_{ij} = \sum_m \int_{\Delta\Omega_m} \rho(\vec{r}, t)\vec{N}_j \cdot \vec{N}_i d\Omega \tag{3.65}$$

$$cG_{ij} = \sum_m \int_{\Delta\Omega_m} \sigma(\vec{r}, t)\vec{N}_i \cdot \nabla\phi_j d\Omega \tag{3.66}$$

Now that these terms are defined for general sums of general basis functions over a non-specific region, we need to examine what the basis functions are for edge elements and nodal elements and what is meant by the summation of the integral of these basis function combinations and how to get eq.(3.53) from eq.(3.30).

3.5 Basis functions

3.5.1 Linear nodal basis function

The normal linear basis function for a node consists of a function that evaluates to one on that node and zero on all adjacent nodes[19, 47, 48]. An illustration of a nodal basis function on a triangular element is presented in Figure 3.3.

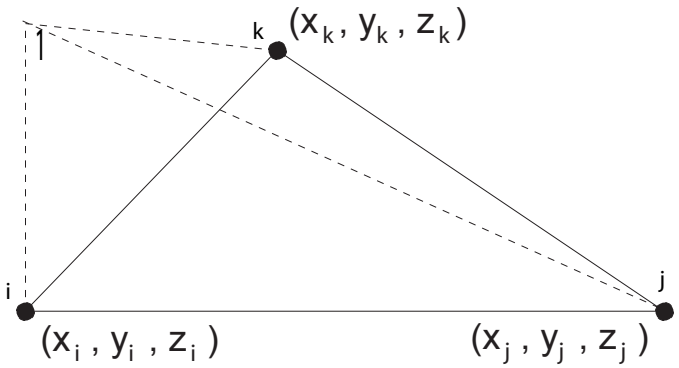

Figure 3.3: Normal linear basis function for node i

The construction for a nodal basis function on a triangle with general vertices in the x-y plane at (x_i, y_i) , (x_j, y_j), and (x_k, y_k) can be constructed via the matrix expression[19, 49]

$$\phi_i(x,y) = \frac{1}{Den}\begin{bmatrix}1 & x & y\end{bmatrix}\begin{bmatrix}x_j y_k - x_k y_j & x_k y_i - x_i y_k & x_i y_j - x_j y_i \\ y_j - y_k & y_k - y_i & y_i - y_j \\ x_k - x_j & x_i - x_k & x_j - x_i\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} \tag{3.67}$$

where the denominator Den is twice the area of the triangular element.

$$Den = x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)y_i \tag{3.68}$$

The last column vector defines for which node we are evaluating the basis function. In the case above it is i. For node j or node k, the last vector would be $\begin{bmatrix}0\\1\\0\end{bmatrix}$ or $\begin{bmatrix}0\\0\\1\end{bmatrix}$ respectively.

From this we get three nodal basis functions on the triangular vertices of a triangular element with node i on the origin, node j on the x axis, and node k in the positive x-y plane. Using an affine transformation, we can transform any triangle in three dimensional space to be in the x-y plane.

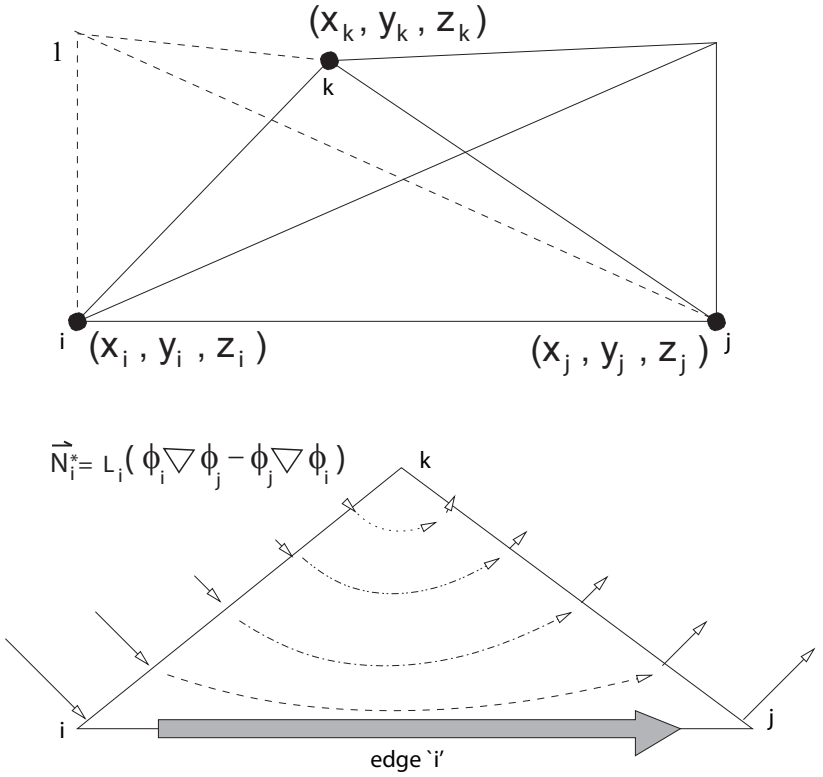

Figure 3.4: Edge element shape function for edge i (connecting nodes i-j)

The nodal basis functions are:

$$\phi_i(x,y) = \frac{((x_j y_k - x_k y_j) + x(y_j - y_k) + y(x_k - x_j))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)y_i} \tag{3.69}$$

$$\phi_j(x,y) = \frac{((x_k y_i - x_i y_k) + x(y_k - y_i) + y(x_i - x_k))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)y_i} \tag{3.70}$$

$$\phi_k(x,y) = \frac{((x_i y_j - x_j y_i) + x(y_i - y_j) + y(x_j - x_i))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)y_i} \tag{3.71}$$

3.5.2 Gradient of the linear nodal basis function

The gradient of the nodal basis function can be found for the same triangle by taking the gradient of eq.(3.67) , which results in replacing the row vector $\begin{bmatrix}1 & x & y\end{bmatrix}$ with $\begin{bmatrix}0 & \hat{x} & \hat{y}\end{bmatrix}$

$$\nabla\phi_i(x,y) = \frac{1}{Den}\begin{bmatrix}0 & \hat{x} & \hat{y}\end{bmatrix}\begin{bmatrix} x_j y_k - x_k y_j & x_k y_i - x_i y_k & x_i y_j - x_j y_i \\ y_j - y_k & y_k - y_i & y_i - y_j \\ x_k - x_j & x_i - x_k & x_j - x_i \end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} \tag{3.72}$$

This is the case for $\nabla\phi_i$. To obtain the gradient of the basis function for nodes j and k we again replace the last column vector with the appropriate term as we did for the nodal basis function in the previous section.

The gradient of the nodal basis functions are:

$$\nabla\phi_i(x,y) = \frac{(\hat{x}(y_j - y_k) + \hat{y}(x_k - x_j))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)yi} \tag{3.73}$$

$$\nabla\phi_j(x,y) = \frac{(\hat{x}(y_k - y_i) + \hat{y}(x_i - x_k))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)yi} \tag{3.74}$$

$$\nabla\phi_k(x,y) = \frac{(\hat{x}(y_i - y_j) + \hat{y}(x_j - x_i))}{x_j y_k - x_k y_j + (y_j - y_k)x_i - (x_j - x_k)yi} \tag{3.75}$$

3.5.3 Coordinate transformations for integration

In order to have only a single case for integration over triangles, the affine transformation is also made to map $x_i \to 0$, $y_i \to 0$, $y_j \to 0$. Node k is mapped to the x-y plane. In transformed coordinates eq.(3.67) becomes

$$\phi_i(x,y) = \frac{1}{Den}\begin{bmatrix}1 & x & y\end{bmatrix}\begin{bmatrix}x_j y_k & 0 & 0\\ -y_k & y_k & 0\\ x_k - x_j & -x_k & x_j\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix} \tag{3.76}$$

with the denominator Den transformed to

$$Den = x_j y_k \tag{3.77}$$

This transformation will allow integration of basis functions over a triangular element, for any orientation of the global coordinates, as a single integral. Our integral surface can consist of surfaces in any 3-dimensional orientation with terms in the finite element matrices evaluated over the transformed 2-dimensional triangle.

3.5.4 Linear edge element shape function

The edge element shape function[48] is defined as

$$\vec{N}_i = L_i(\phi_i \nabla\phi_j - \phi_j \nabla\phi_i) \tag{3.78}$$

where L_i is the length of the edge i, and ϕ_i is the nodal basis function on node i and ϕ_j is the nodal basis function of node j. The edge i is from nodes i to j. $\vec{N}_i$, as given, is used to interpolate vectors on a triangle along with the other edge shape functions $\vec{N}_j$ and $\vec{N}_k$ for a triangular element with edges i, j, and k. Within triangle$_{(i,j,k)}$ a vector is given by

$$\vec{V} = \vec{N}_i V_i + \vec{N}_j V_j + \vec{N}_k V_k \tag{3.79}$$

To begin solving for general edge element shape functions $\vec{N}$ we make a labeling notation change in order to ease in the solution process. We redefine the inner matrix from eq.(3.67) and give the individual terms alphanumeric identifiers. It is useful to present this in the generalized 2-dimensional form for a general case.

$$\begin{bmatrix}A & B & C\\ D & E & F\\ G & H & I\end{bmatrix} = \begin{bmatrix}x_j y_k - x_k y_j & x_k y_i - x_i y_k & x_i y_j - x_j y_i\\ y_j - y_k & y_k - y_i & y_i - y_j\\ x_k - x_j & x_i - x_j & x_j - x_i\end{bmatrix} \tag{3.80}$$

In the transformed coordinates that we use, eq.(3.80) is replaced by

$$\begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} = \begin{bmatrix} x_j y_k & 0 & 0 \\ -y_k & y_k & 0 \\ x_k - x_j & -x_k & x_j \end{bmatrix} \tag{3.81}$$

Evaluating eq.(3.78) can be done starting at the matrix formulation and reducing terms until we arrive at an expression for each edge's shape function. In the generalized notation we have*

$$\begin{aligned}
\vec{N}_i &= L_i(\phi_i \nabla\phi_j - \phi_j \nabla\phi_i) \\
&= L_i \left(\frac{1}{Den}[1\ x\ y] \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \frac{1}{Den}[0\ \hat{x}\ \hat{y}] \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \right. \\
&\qquad \left. - \frac{1}{Den}[1\ x\ y] \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \frac{1}{Den}[0\ \hat{x}\ \hat{y}] \begin{bmatrix} A & B & C \\ D & E & F \\ G & H & I \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \right) \\
&= \frac{L_i}{Den^2} \left((A + Dx + Gy)(E\hat{x} + H\hat{y}) - (B + Ex + Hy)(D\hat{x} + G\hat{y}) \right) \\
&= \frac{L_i}{Den^2} \left(\hat{x}(AE + DEx + GEy - BD - DEx - HDy) \right. \\
&\qquad \left. + \hat{y}(AH + DHx + GHy - BG - EGx - HGy) \right) \\
&= \frac{L_i}{Den^2} \left((((AE - BD) + y(GE - HD))\hat{x} + ((AH - BG) + x(DH - EG))\hat{y} \right)
\end{aligned} \tag{3.82}$$

Thus evaluating the expression for

$$\vec{N}_i = L_i(\phi_i \nabla\phi_j - \phi_j \nabla\phi_i)$$

and repeating for the case of edges j and k we end up with the expressions:

$$\vec{N}_i = \frac{L_i}{Den^2}(((AE - BD) + y(GE - HD))\hat{x} + ((AH - BG) + x(DH - EG))\hat{y}) \tag{3.83}$$

$$\vec{N}_j = \frac{L_j}{Den^2}(((BF - CE) + y(HF - IE))\hat{x} + ((BI - CH) + x(EI - FH))\hat{y}) \tag{3.84}$$

$$\vec{N}_k = \frac{L_k}{Den^2}(((CD - AF) + y(ID - FG))\hat{x} + ((CG - AI) + x(FG - DI))\hat{y}) \tag{3.85}$$

*Note: In a local triangle with nodes i, j, k, the edge element $\vec{N}_i$ goes from node i to node j. In the global sums in the matrix definitions, the subscript on $\vec{N}_i$ is a counting index and not necessarily the same as the index of the first node of the edge.

3.5.5 Example of linear edge element shape function for canonical triangle

When specialized to the case of a triangle having node i at $(0,0,0)$, node j at $(x_j,0,0)$, node k at $(x_k,y_k,0)$, equations (3.83), (3.84), and (3.85) become

$$\vec{N}_i = (1-\frac{y}{y_k})\hat{x} + \frac{(x-x_k)}{y_k}\hat{y} \tag{3.86}$$

$$\vec{N}_j = \frac{\sqrt{(x_j-x_k)^2+y_k^2}}{x_j y_k}(-y\hat{x}+x\hat{y}) \tag{3.87}$$

$$\vec{N}_k = \frac{\sqrt{x_k^2+y_k^2}}{x_j y_k}(-y\hat{x}+(x-x_j)\hat{y}). \tag{3.88}$$

Examination of $\vec{N}_i$ shows that the component of $\vec{N}_i$ parallel to edge i, between nodes i and j is just $\hat{x}$ on that edge where $y=0$. Examination of $\vec{N}_j$ shows that it is entirely in the $\hat{y}$ direction along edge j, between nodes j and k, where $y=0$. The same is true for $\vec{N}_k$. Thus the value of any vector field given in terms of linear edge elements:

$$\vec{A} = \sum A_i \vec{N}_i, \tag{3.89}$$

when taken on an edge, has the component parallel to the edge equal to the $\vec{N}_i$ coefficients

$$\hat{e}_{pi} \cdot \vec{A}|_{on\ edge\ i} = A_i. \tag{3.90}$$

3.6 Constructing the coefficient matrices

3.6.1 The $[S]$ matrix

The $[S]$ matrix is a 'stiffness' matrix which relates how individual edges are linked together. Elements of $[S]$ are composed of the sum over the differential sub-volumes m of the shape function for edge 'j' times the shape function for edge 'i' with the dot product of their terms being evaluated. The integral is over all space, but the only terms that are non-zero are where the edges share the same sub-volume $\Delta\Omega_m$. This means terms are only generated for triangular elements where both edges are members of the triangle. The notation used by Koizumi[45] can be better expressed with the knowledge that the linear edge shape functions are vector field quantities and is thus written as

$$S_{ij} = \sum_m \int_{\Delta\Omega_m} \vec{N}_j \cdot \vec{N}_i d\Omega \tag{3.91}$$

For a triangular mesh, which we are using in our implementation, we can see that there will be some coupling for every edge of the triangular element. Figure 3.5 shows an example of a section of a triangular mesh. We see that we should have a term for S_{de} as both edge 'e' and 'f' belong to triangle A. Coupling term $S_{df} = 0$ due to edges 'd' and 'f' not belonging to the same triangular element. Coupling term S_{ee} would contain contributions from both triangle 'A' and triangle 'B'.

To solve for the general matrix element, we begin with the multiplication of the shape functions for the general triangle.

$$S_{ii} = \int_{triA} \vec{N}_i \cdot \vec{N}_i d\Omega_A + \int_{triB} \vec{N}_i \cdot \vec{N}_i \Omega_B \tag{3.92}$$

For a general triangular element, the self term, S_{ii} could have contributions from both triangles for which it serves as an edge. In the reference figure, edge B would have an integral term from both triangles. As we are interested in constructing a general form to code for these elements, we will reduce the solution to an individual triangle. This is also the case for the general terms S_{ij} as there will be only one triangular region to integrate over.

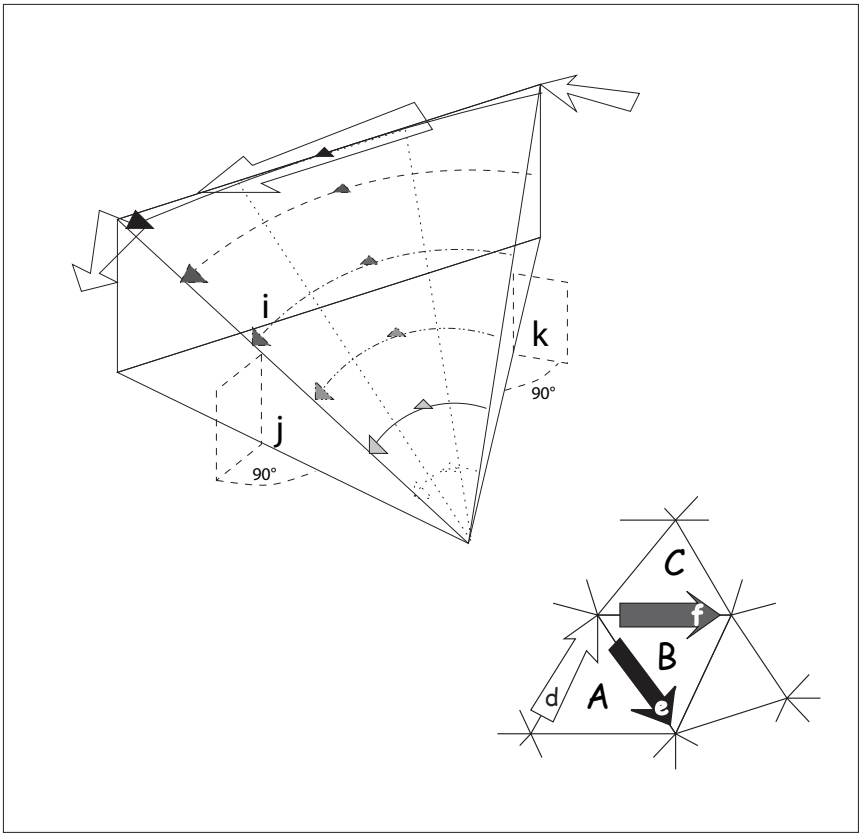

Figure 3.5: Edge element shape function for edge i (connecting nodes $i-j$) and an example mesh showing multiple edges. There will be coupling between edges that are part of the same triangular element, but none for edges that have no common elements.

$$S_{ij} = \int_{triA} \vec{N}_i \cdot \vec{N}_j d\Omega_A \tag{3.93}$$

Expressing the shape functions in the manner previously listed and carrying out the multiplication we arrive at

$$\begin{aligned} S_{ii} = \int_A (\frac{L_i}{Den^2}(((AE-BD)+y(GE-HD))\hat{x} + ((AH-BG)+x(DH-EG))\hat{y})) \\ \cdot (\frac{L_i}{Den^2}(((AE-BD)+y(GE-HD))\hat{x} + ((AH-BG)+x(DH-EG))\hat{y}))d\Omega_A + \int_B \dots \end{aligned} \tag{3.94}$$

$$\begin{aligned} S_{ii} = \int_{triA} \frac{L_i^2}{Den^4}\{&(AE-BD)(AE-BD) + 2y(GE-HD)(AE-BD) \\ &+ y^2(GE-HD)(GE-HD) + (AH-BG)(AH-BG) \\ &+ 2x(AH-BG)(DH-EG) + x^2(DH-EG)(DH-EG)\}d\Omega_A + \int_B \dots \end{aligned} \tag{3.95}$$

Here we will make use of another substitution of terms. The multiplied shape function has the form of a polynomial of x and y, so we collect the terms and define

$$M = (AE-BD)(AE-BD) \qquad N = 2(GE-HD)(AE-BD) \qquad P = (GE-HD)(GE-HD)$$

$$Q = (AH-BG)(AH-BG) \qquad R = 2(AH-BG)(DH-EG) \qquad S = (DH-EG)(DH-EG)$$

which results in a cleaner expression for the integral over the triangular element

$$S_{ii} = \frac{L_i^2}{Den^4} \int_0^{y_3} \int_{x_l}^{x_r} \left[M + yN + y^2P + Q + xR + x^2S\right] dxdy + \int_B \dots \tag{3.96}$$

The integral which requires evaluation is

$$I_{ii} = \int_0^{y_3} \int_{x_l}^{x_r} \left(x^0y^0M + x^0y^1N + x^0y^2P + x^0y^0Q + x^1y^0R + x^2y^0S\right) dxdy \tag{3.97}$$

which we can break up as

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3}\int_{x_l}^{x_l} x^0 y^0 (M+Q)dxdy \\
&+ \int_0^{y_3}\int_{x_l}^{x_r} x^0 y^1 N dxdy \\
&+ \int_0^{y_3}\int_{x_l}^{x_r} x^0 y^2 P dxdy \\
&+ \int_0^{y_3}\int_{x_l}^{x_r} x^1 y^0 R dxdy \\
&+ \int_0^{y_3}\int_{x_l}^{x_r} x^2 y^0 S dxdy
\end{aligned}
\tag{3.98}
$$

From Calculus we know that the integral over a triangle in the $x-y$ plane is

$$
I(m,n) = \int_0^{y_3}\int_{x_l}^{x_r} x^m y^n dxdx \tag{3.99}
$$

$$
x_l = y(x_3/y_3) \qquad x_r = x_2 + y(x_3 - x_2)/y_3 \tag{3.100}
$$

$$
I(m,n) = \int_0^{y_3} \frac{y^n}{m+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{m+1} - \left[y\frac{x_3}{y_3} \right]^{m+1} \right\} \tag{3.101}
$$

What we now have are five separate cases of this integral to combine to obtain I_{ii}:.

$$
I_{ii} = (M+Q) \times I(0,0) + N \times I(0,1) + P \times I(0,2) + R \times I(1,0) + S \times I(2,0) \tag{3.102}
$$

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3} (M+Q)\frac{y^0}{0+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{0+1} - \left[y\frac{x_3}{y_3} \right]^{0+1} \right\} dy \\
&+ \int_0^{y_3} N\frac{y^1}{0+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{0+1} - \left[y\frac{x_3}{y_3} \right]^{0+1} \right\} dy \\
&+ \int_0^{y_3} P\frac{y^2}{0+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{0+1} - \left[y\frac{x_3}{y_3} \right]^{0+1} \right\} dy \\
&+ \int_0^{y_3} R\frac{y^0}{1+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{1+1} - \left[y\frac{x_3}{y_3} \right]^{1+1} \right\} dy \\
&+ \int_0^{y_3} S\frac{y^0}{2+1} \left\{ \left[x_2 + y\frac{x_3 - x_2}{y_3} \right]^{2+1} - \left[y\frac{x_3}{y_3} \right]^{2+1} \right\} dy
\end{aligned}
\tag{3.103}
$$

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3} (M+Q)\left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right] - \left[y\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} N y^1 \left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right] - \left[y\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} P y^2 \left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right] - \left[y\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} R\frac{1}{2}\left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right]^2 - \left[y\frac{x_3}{y_3}\right]^2\right\} dy \\
&+ \int_0^{y_3} S\frac{1}{3}\left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right]^3 - \left[y\frac{x_3}{y_3}\right]^3\right\} dy
\end{aligned} \tag{3.104}
$$

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3} (M+Q)\left\{\left[x_2 + y\frac{x_3-x_2}{y_3}\right] - \left[y\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} N\left\{\left[yx_2 + y^2\frac{x_3-x_2}{y_3}\right] - \left[y^2\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} P\left\{\left[y^2x_2 + y^3\frac{x_3-x_2}{y_3}\right] - \left[y^3\frac{x_3}{y_3}\right]\right\} dy \\
&+ \int_0^{y_3} R\frac{1}{2}\left\{\left[x_2^2 + 2x_2\frac{y}{y_3}(x_3-x_2) + \frac{y^2}{y_3^2}(x_3-x_2)^2\right] - \left[\frac{y^2}{y_3^2}x_3^2\right]\right\} dy \\
&+ \int_0^{y_3} S\frac{1}{3}\left\{\left[x_2^3 + 3x_2^2\frac{y}{y_3}(x_3-x_2) + 3x_2\frac{y^2}{y_3^2}(x_3-x_2)^2 + \frac{y^3}{y_3^3}(x_3-x_2)^3\right] - \left[\frac{y^3}{y_3^3}x_3^3\right]\right\} dy
\end{aligned} \tag{3.105}
$$

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3} (M+Q)\left\{x_2 + y\frac{x_3-x_2}{y_3} - y\frac{x_3}{y_3}\right\} dy \\
&+ \int_0^{y_3} N\left\{yx_2 + y^2\frac{x_3-x_2}{y_3} - y^2\frac{x_3}{y_3}\right\} dy \\
&+ \int_0^{y_3} P\left\{y^2x_2 + y^3\frac{x_3-x_2}{y_3} - y^3\frac{x_3}{y_3}\right\} dy \\
&+ \int_0^{y_3} R\frac{1}{2}\left\{x_2^2 + 2x_2\frac{y}{y_3}(x_3-x_2) + \frac{y^2}{y_3^2}(x_3-x_2)^2 - \frac{y^2}{y_3^2}x_3^2\right\} dy \\
&+ \int_0^{y_3} S\frac{1}{3}\left\{x_2^3 + 3x_2^2\frac{y}{y_3}(x_3-x_2) + 3x_2\frac{y^2}{y_3^2}(x_3-x_2)^2 + \frac{y^3}{y_3^3}(x_3-x_2)^3 - \frac{y^3}{y_3^3}x_3^3\right\} dy
\end{aligned} \tag{3.106}
$$

$$
\begin{aligned}
I_{ii} = &\int_0^{y_3} (M+Q)\left\{x_2 - y\frac{x_2}{y_3}\right\} dy \\
&+ \int_0^{y_3} N\left\{yx_2 - y^2\frac{x_2}{y_3}\right\} dy \\
&+ \int_0^{y_3} P\left\{y^2x_2 - y^3\frac{x_2}{y_3}\right\} dy \\
&+ \int_0^{y_3} R\frac{1}{2}\left\{x_2^2 + 2x_2\frac{y}{y_3}(x_3-x_2) + \frac{y^2}{y_3^2}((x_3-x_2)^2 - x_3^2)\right\} dy \\
&+ \int_0^{y_3} S\frac{1}{3}\left\{x_2^3 + 3x_2^2\frac{y}{y_3}(x_3-x_2) + 3x_2\frac{y^2}{y_3^2}(x_3-x_2)^2 + \frac{y^3}{y_3^3}((x_3-x_2)^3 - x_3^3)\right\} dy
\end{aligned} \tag{3.107}
$$

$$
\begin{aligned}
I_{ii} =&(M+Q)\left\{x_2y_3-\frac{1}{2}y_3x_2\right\}\\
&+N\left\{\frac{1}{2}y_3^2x_2-\frac{1}{3}y_3^2x_2\right\}\\
&+P\left\{\frac{1}{3}y_3^3x_2-\frac{1}{4}y_3^3x_2\right\}\\
&+R\frac{1}{2}\left\{y_3x_2^2+x_2y_3(x_3-x_2)+\frac{1}{3}y_3((x_3-x_2)^2-x_3^2)\right\}\\
&+S\frac{1}{3}\left\{y_3x_2^3+\frac{3}{2}x_2^2y_3(x_3-x_2)+x_2y_3(x_3-x_2)^2+\frac{1}{4}y_3((x_3-x_2)^3-x_3^3)\right\}
\end{aligned}
\tag{3.108}
$$

$$
\begin{aligned}
I_{ii} =&(M+Q)\left\{\frac{1}{2}x_2y_3\right\}\\
&+N\left\{\frac{1}{6}y_3^2x_2\right\}\\
&+P\left\{\frac{1}{12}y_3^3x_2\right\}\\
&+R\left\{\frac{1}{6}(x_2x_3+x_2^2)y_3\right\}\\
&+S\left\{\frac{1}{12}(x_2x_3^2+x_2^2x_3+x_2^3)y_3\right\}
\end{aligned}
\tag{3.109}
$$

All possible combinations of shape functions are $\vec{N}_i\cdot\vec{N}_i$, $\vec{N}_i\cdot\vec{N}_j$, $\vec{N}_i\cdot\vec{N}_k$, $\vec{N}_j\cdot\vec{N}_k$, $\vec{N}_j\cdot\vec{N}_j$, and $\vec{N}_k\cdot\vec{N}_k$. It should be noted that the combinations are symmetric, so $\vec{N}_i\cdot\vec{N}_j=\vec{N}_j\cdot\vec{N}_i$. By following the same derivation for integral terms in $[S]$, we see that the integral of the triangle for any combination of edges results in the same polynomial form of

$$M+Q+Ny+Py^2+Rx+Sx^2.$$

Therefore to solve these different cases we only need derive the coefficients for the polynomial and substitute into the integral solved above. These combinations result in the collected terms in Table 3.2

Table 3.2: Polynomial Coefficients for shape function combinations in the formulation of the $[S]$ Matrix

$\vec{N}_i \cdot \vec{N}_i$	$\vec{N}_j \cdot \vec{N}_j$	$\vec{N}_k \cdot \vec{N}_k$
$M = (AE - BD)(AE - BD)$	$M = (BF - CE)(BF - CE)$	$M = (CD - AF)(CD - AF)$
$N = 2(GE - HD)(AE - BD)$	$N = 2(BF - CE)(HF - IE)$	$N = 2(CD - AF)(ID - FG)$
$P = (GE - HD)(GE - HD)$	$P = (HF - IE)(HF - IE)$	$P = (ID - FG)(ID - FG)$
$Q = (AH - BG)(AH - BG)$	$Q = (BI - CH)(BI - CH)$	$Q = (CG - AI)(CG - AI)$
$R = 2(AH - BG)(DH - EG)$	$R = 2(BI - CH)(EI - FH)$	$R = 2(CG - AI)(FG - DI)$
$S = (DH - EG)(DH - EG)$	$S = (EI - FH)(EI - FH)$	$S = (FG - DI)(FG - DI)$

$\vec{N}_i \cdot \vec{N}_j$
$M = (AE - BD)(BF - CE)$
$N = (AE - BD)(HF - IE) + (GE - HD)(BF - CE)$
$P = (GE - HD)(HF - IE)$
$Q = (AH - BG)(BI - CH)$
$R = (AH - BG)(EI - FH) + (BI - CH)(DH - EG)$
$S = (DH - EG)(EI - FH)$
$\vec{N}_i \cdot \vec{N}_k$
$M = (AE - BD)(CD - AF)$
$N = (AE - BD)(ID - FG) + (CD - AF)(GE - HD)$
$P = (GE - HD)(ID - FG)$
$Q = (AH - BG)(CG - AI)$
$R = (AH - BG)(FG - DI) + (CG - AI)(DH - EG)$
$S = (DH - EG)(FG - DI)$
$\vec{N}_j \cdot \vec{N}_k$
$M = (BF - CE)(CD - AF)$
$N = (BF - CE)(ID - FG) + (CD - AF)(HF - IE)$
$P = (HF - IE)(ID - FG)$
$Q = (BI - CH)(CG - AI)$
$R = (BI - CH)(FG - DI) + (CG - AI)(EI - FH)$
$S = (EI - FH)(FG - DI)$

3.6.2 The $[G]$ matrix

The $[G]$ matrix is defined as

$$G_{ij} = \sum_m \int_{\Delta\Omega_m} \vec{N}_i \cdot \nabla\phi_j d\Omega \tag{3.110}$$

where the $[G]$ matrix will be the number of edges by the number of nodes in size, or m edges times n nodes. Using the same notation for the matrix in eq.(3.80), it is convenient to rewrite the definitions of the gradients of the nodal functions as;

$$\nabla\phi_i = \frac{1}{Den}(D\hat{x} + G\hat{y})$$
$$\nabla\phi_j = \frac{1}{Den}(E\hat{x} + H\hat{y})$$
$$\nabla\phi_k = \frac{1}{Den}(F\hat{x} + I\hat{y})$$

Possible combinations are $\vec{N}_i \cdot \nabla\phi_i$, $\vec{N}_i \cdot \nabla\phi_j$, $\vec{N}_i \cdot \nabla\phi_k$, $\vec{N}_j \cdot \nabla\phi_i$, $\vec{N}_j \cdot \nabla\phi_j$, $\vec{N}_j \cdot \nabla\phi_k$, $\vec{N}_k \cdot \nabla\phi_i$, $\vec{N}_k \cdot \nabla\phi_j$, $\vec{N}_k \cdot \nabla\phi_k$.

Solving for $\vec{N}_i \cdot \nabla\phi_i$, as an example,:

$$\begin{aligned}
&\vec{N}_i \cdot \nabla\phi_i = \\
&= \frac{L_i}{Den^2}(((AE - BD) + y(GE - HD))\hat{x} + ((AH - BG) + x(DH - EG))\hat{y}) \cdot \frac{1}{Den}(D\hat{x} + G\hat{y}) \\
&= \frac{L_i}{Den^3}(((AE - BD) + y(GE - HD))D + ((AH - BG) + x(DH - EG))(G)) \\
&= \frac{L_i}{Den^3}(D(AE - BD) + yD(GE - HD) + G(AH - BG) + xG(DH - EG))
\end{aligned} \tag{3.111}$$

Comparing this derivation for element G_{ii} with the derivations for $[S]$, we note that we have the same polynomial form for the integrand, with the quadratic terms equal to zero.

$$\frac{L}{Den^3}(D(AE - BD) + yD(GE - HD) + 0y^2 + G(AH - BG) + xG(DH - EG) + 0x^2)$$

Furthermore we adopt the same lettering strategy as in the [S] formulation. This will cause less confusion during code implementation.

$$M + Ny + Py^2 + Q + Rx + Sx^2$$

With direct comparison it is easy to extract the polynomial coefficients. The polynomial coefficients for the other cases of edge and node combinations are easily derived and the resultant coefficients for all cases are listed in Table 3.3

Table 3.3: Polynomial Coefficients for shape function combinations in the formulation of the $[G]$ Matrix

	$\vec{N_i} \cdot \nabla \phi_i$	
$M = D(AE - BD)$	$N = D(GE - HD)$	$P = 0$
$Q = G(AH - BG)$	$R = G(DH - EG)$	$S = 0$
	$\vec{N_i} \cdot \nabla \phi_j$	
$M = E(AE - BD)$	$N = E(GE - HD)$	$P = 0$
$Q = H(AH - BG)$	$R = H(DH - EG)$	$S = 0$
	$\vec{N_i} \cdot \nabla \phi_k$	
$M = F(AE - BD)$	$N = F(GE - HD)$	$P = 0$
$Q = I(AH - BG)$	$R = I(DH - EG)$	$S = 0$
	$\vec{N_j} \cdot \nabla \phi_i$	
$M = D(BF - CE)$	$N = D(HF - IE)$	$P = 0$
$Q = G(BI - CH)$	$R = G(EI - FH)$	$S = 0$
	$\vec{N_j} \cdot \nabla \phi_j$	
$M = E(BF - CE)$	$N = E(HF - IE)$	$P = 0$
$Q = H(BI - CH)$	$R = H(EI - FH)$	$S = 0$
	$\vec{N_j} \cdot \nabla \phi_k$	
$M = F(BF - CE)$	$N = F(HF - IE)$	$P = 0$
$Q = I(BI - CH)$	$R = I(EI - FH)$	$S = 0$
	$\vec{N_k} \cdot \nabla \phi_i$	
$M = D(CD - AF)$	$N = D(ID - FG)$	$P = 0$
$Q = G(CG - AI)$	$R = G(FG - DI)$	$S = 0$
	$\vec{N_k} \cdot \nabla \phi_j$	
$M = E(CD - AF)$	$N = E(ID - FG)$	$P = 0$
$Q = H(CG - AI)$	$R = H(FG - DI)$	$S = 0$
	$\vec{N_k} \cdot \nabla \phi_k$	
$M = F(CD - AF)$	$N = F(ID - FG)$	$P = 0$
$Q = I(CG - AI)$	$R = I(FG - DI)$	$S = 0$

3.6.3 The $[D]$ matrix

The $[D]$ matrix is the global coefficient matrix[19] which contains information from the connection of the gradients of nodal basis functions within a triangular element.

$$D_{ij} = \sum_m \int_{\Delta\Omega_m} \sigma_{\Delta\Omega_m} \nabla\phi_j \cdot \nabla\phi_i d\Omega \tag{3.112}$$

This matrix will be n nodes by n nodes and symmetric. All the possible combinations of basis functions are ii, ij, ik, jk, jj, kk. In this case all of the combinations will have no dependence on x or y, thus only the 'M' term from the integral will be solved for each combination. The conductivity term, which can be assigned as a zero order basis function for each triangular element can be moved outside of the integral with the 'Den' term. The following are the definitions of the gradients of the nodal functions in terms of the matrix defined by eq.(3.80).

Starting with the possible gradients of nodal basis functions

$$\nabla\phi_i = \frac{1}{Den}(D\hat{x} + G\hat{y})$$

$$\nabla\phi_j = \frac{1}{Den}(E\hat{x} + H\hat{y})$$

$$\nabla\phi_k = \frac{1}{Den}(F\hat{x} + I\hat{y})$$

and following the format established above, node combination i-i evaluates as:

$$\begin{aligned} \nabla\phi_i \cdot \nabla\phi_i =& (D\hat{x} + G\hat{y}) \cdot (D\hat{x} + G\hat{y}) \\ \nabla\phi_j \cdot \nabla\phi_i =& D^2 + G^2 \\ \nabla\phi_j \cdot \nabla\phi_i =& (D^2 + G^2)x^0 y^0 \\ M =& (DD + GG) \end{aligned} \tag{3.113}$$

For node combination i-j

$$\begin{aligned} \nabla\phi_i \cdot \nabla\phi_j =& (D\hat{x} + G\hat{y}) \cdot (E\hat{x} + H\hat{y}) \\ \nabla\phi_i \cdot \nabla\phi_j =& (DE + GH) \end{aligned} \tag{3.114}$$

$$M = (DE + GH)$$

For node combination i-k

$$\begin{aligned} \nabla\phi_i \cdot \nabla\phi_k =& (D\hat{x} + G\hat{y}) \cdot (F\hat{x} + I\hat{y}) \\ \nabla\phi_i \cdot \nabla\phi_k =& (DF + GI) \\ M =& (DF + GI) \end{aligned} \tag{3.115}$$

For node combination j-j

$$\begin{aligned} \nabla\phi_j \cdot \nabla\phi_j =& (E\hat{x} + H\hat{y}) \cdot (E\hat{x} + H\hat{y}) \\ \nabla\phi_j \cdot \nabla\phi_j =& (E^2 + H^2) \\ M =& (EE + HH) \end{aligned} \tag{3.116}$$

For node combination j-k

$$\begin{aligned} \nabla\phi_j \cdot \nabla\phi_k =& (E\hat{x} + H\hat{y}) \cdot (F\hat{x} + I\hat{y}) \\ \nabla\phi_j \cdot \nabla\phi_k =& (EF + HI) \\ M =& (EF + HI) \end{aligned} \tag{3.117}$$

For node combination k-k

$$\begin{aligned} \nabla\phi_k \cdot \nabla\phi_k =& (F\hat{x} + I\hat{y}) \cdot (F\hat{x} + I\hat{y}) \\ \nabla\phi_k \cdot \nabla\phi_k =& (FF + II) \\ M =& (FF + II) \end{aligned} \tag{3.118}$$

Table 3.4: Polynomial Coefficients for basis function combinations in the formulation of the $[D]$ Matrix

$\nabla\phi_i \cdot \nabla\phi_i$	$\nabla\phi_i \cdot \nabla\phi_j$	$\nabla\phi_i \cdot \nabla\phi_k$
$M = (DD + GG)$	$M = (DE + GH)$	$M = (DF + GI)$
$N = 0$	$N = 0$	$N = 0$
$P = 0$	$P = 0$	$P = 0$
$Q = 0$	$Q = 0$	$Q = 0$
$R = 0$	$R = 0$	$R = 0$
$S = 0$	$S = 0$	$S = 0$
$\nabla\phi_j \cdot \nabla\phi_j$	$\nabla\phi_j \cdot \nabla\phi_k$	$\nabla\phi_k \cdot \nabla\phi_k$
$M = (EE + HH)$	$M = (EF + HI)$	$M = (FF + II)$
$N = 0$	$N = 0$	$N = 0$
$P = 0$	$P = 0$	$P = 0$
$Q = 0$	$Q = 0$	$Q = 0$
$R = 0$	$R = 0$	$R = 0$
$S = 0$	$S = 0$	$S = 0$

3.6.4 The $[P]$ matrix

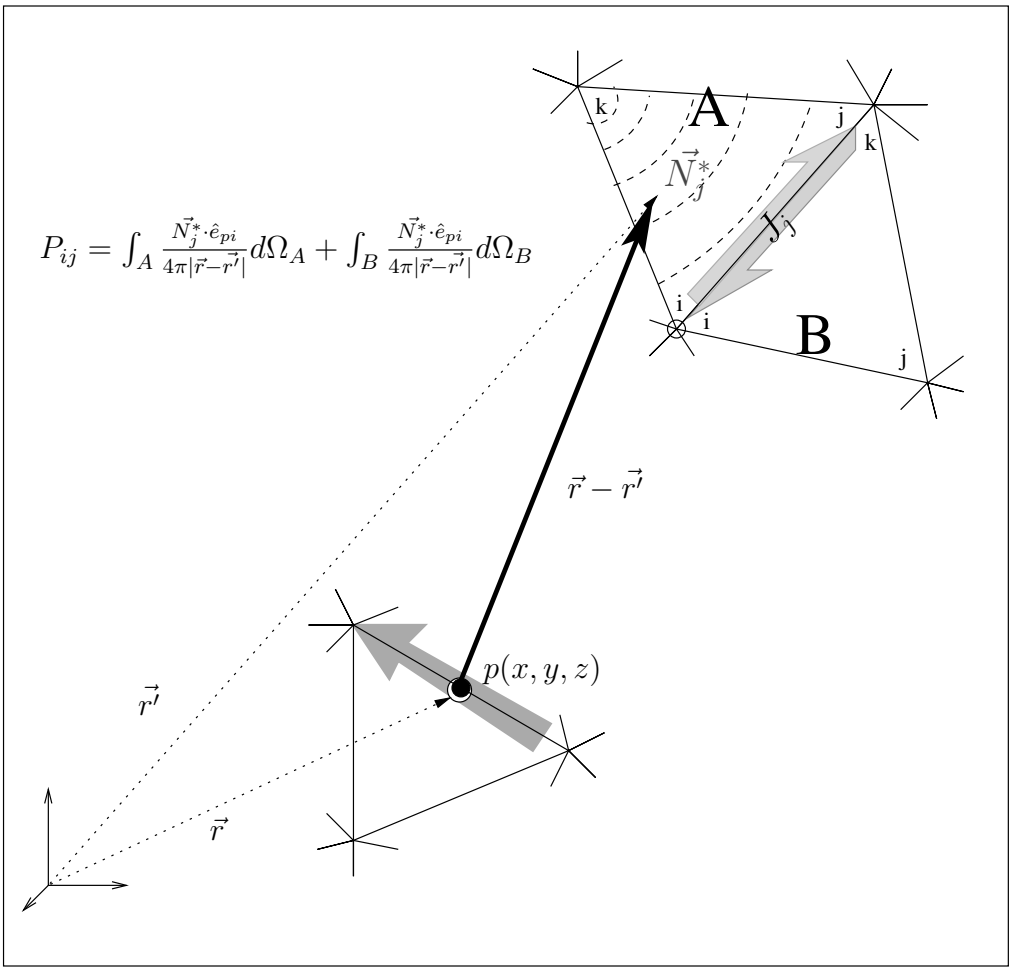

Figure 3.6: Example of a construction for the $[P]$ matrix with observation point 'p' and current density J_j edge being integrated. The contribution of the edge is contained in two separate integrals, one for each triangular element. The shape function $\vec{N}_j$ for the current density on edge 'j' is shown in triangle 'A'.

$$P_{ij} = \sum_m \int_{\Delta\Omega_m} \vec{N}_j \cdot \hat{e}_{pi} \Psi d\Omega$$

The $[P]$ matrix has a more complex construction. The integral includes the free space Green's function, which among other things will present us with a singularity when the triangle of integration contains the observation point. This matrix will be edges by edges in

size, and dense. It will have some symmetry, but it will not necessarily be symmetric. An illustration of the geometries and terms used in construction of terms in the $[P]$ matrix are in Figure 3.6. The observation point of the edge with unit vector $\hat{e}_{pi}$ is located at the edge's center of mass. This was done in accordance to an approximation made by Koizumi[45]. In his formulation for calculation of eddy currents he includes magnetic materials. For these elements he uses the center of mass on a zero order element for the observation point when working with magnetic materials. This is an acceptable approximation provided the triangular elements are small compared to the size of the domain Ω. A more accurate approximation would be to perform a six-dimensional integral, like the ones performed for the partial inductance method in Chapter 2, over the triangular differential sub-volume m. This would integrate the observation 'point' along the entire edge. An unfortunate consequence of this change is that it would greatly increase computation time, and the resultant gain in accuracy is questionable since the representation of the current along an edge is a constant along the entirety of the edge.

A general element P_{ij} can be expressed as

$$P_{ij} = \int_A \vec{N}_j \cdot \hat{e}_{pi} \Psi d\Omega_A + \int_B \vec{N}_j \cdot \hat{e}_{pi} \Psi d\Omega_B \tag{3.119}$$

To evaluate the integral for one region in general terms we note that we have an integral where the local edge is on location i and the observation point has a unit direction of

$$\hat{e}_{pi} = \hat{x} e_{p_x} + \hat{y} e_{p_y} + \hat{z} e_{p_z} \tag{3.120}$$

associated with the observation point's global edge location. The observation edge can be on a different plane as the triangle with the edge of integration. The $\hat{z}$ term will never contribute to the integral due to the dot product with the shape function that has no $\hat{z}$ component. The integral can be expressed and solved in a similar manner as the previous coefficient matrix solutions by reducing the shape function combinations down to a sum of monomial terms.

$$
\begin{aligned}
I &= \int_{triangle} \frac{\vec{N_j} \cdot \hat{e}_{pi}}{4\pi\sqrt{(x-x_p)^2+(y-y_p)^2+(z-z_p)^2}} dxdy|_{z=0} \\
I &= \int_0^{y_3}\int_{x_r}^{x_l} \frac{L_j}{Den^2} \frac{((M+Ny)\hat{x}+(Q+Rx)\hat{y})\cdot(\hat{x}e_{p_x}+\hat{y}e_{p_y})}{4\pi\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} dxdy \\
I &= \int_0^{y_3}\int_{x_r}^{x_l} \frac{L_j}{Den^2} \frac{e_{p_x}(M+Ny)+e_{p_y}(Q+Rx)}{4\pi\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} dxdy \\
I &= \frac{L_j}{4\pi Den^2}\int_0^{y_3}\int_{x_r}^{x_l} \frac{e_{p_x}M+e_{p_x}Ny+e_{p_y}Q+e_{p_y}Rx}{\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} dxdy \\
I &= \frac{L_j}{4\pi Den^2}\int_0^{y_3}\int_{x_r}^{x_l} [\frac{e_{p_x}M}{\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} \\
&+ \frac{e_{p_x}Ny}{\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} \\
&+ \frac{e_{p_y}Q}{\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}} \\
&+ \frac{e_{p_y}Rx}{\sqrt{(x-x_p)^2+(y-y_p)^2+z_p^2}}] dxdy
\end{aligned}
\tag{3.121}
$$

where the limits of integration over the triangle are again defined as

$$x_l = y(x_3/y_3)$$

$$x_r = x_2 + y(x_3 - x_2)/y_3$$

.

We will make use of the following two integrals

$$I_1 = \int \frac{dx}{R}; R = \sqrt{ax^2+bx+c} \tag{3.122}$$

$$\int \frac{dx}{R} = \frac{1}{\sqrt{a}} \sinh^{-1}\left\{\frac{2ax+b}{\sqrt{4ac-b^2}}\right\}; a>0, 4ac-b^2>0 \tag{3.123}$$

$$I_2 = \int \frac{xdx}{R}; R = \sqrt{ax^2+bx+c} \tag{3.124}$$

$$\int \frac{x dx}{R} = \frac{R}{a} - \frac{b}{2a} \int \frac{dx}{R} \tag{3.125}$$

The solution to the integral over x contains a hyperbolic sine term which them must be evaluated numerically. It is more efficient to numerically integrate over both variables with a "cubature" routine. The "cubature" routine "triex" [50, 51, 52] was used to numerically evaluate the Green's function integral.

Chapter 4

Implementation of transient current density solver in C++

To implement the algorithm presented in Chapter 3 a numerical computer code was written in C++. This program was prototyped with the matrix library `NEWMAT`[53] and later the matrix operations were translated into `LAPACK`[54] for more efficient calculation of matrix inverses. For the description of coding algorithms and functions, general terms will be used. It was convenient to use a prepackaged and open-source program to generate geometry meshes for the solution spaces. For this we chose `GMSH`[21,22,23]. By using an already made meshing program, it allowed us to direct more effort into the programming of the EM solver. By being an open source code, the formats for the mesh files were available for reading directly into the transient eddy current solver program by coding a proper file input subroutine. The transient current density solver we coded is called `TRANSEDDY` (**TRANS**ient **EDDY** current solver).

4.1 Using `GMSH`

`GMSH`[22] is a readily available mesh generation program that is available on many Linux distributions as well as having versions compiled to run on most computers running the Microsoft Windows family of operating system. Geometry files and meshes can be made in interactive mode with the Graphic User Interface (GUI) or from direct entry with the

Extention	Description
*.geo	Geometry File
*.msh	Mesh File
*.pos	Parsed Output File

Table 4.1: File extensions used by GMSH

appropriate files. Types of files and their file extentions are listed in Table 4.1. GMSH also has a built in display module for showing output files from its integrated EM solver program GetDP[20]. We will also use GMSH as a means of viewing and processing the output data from the new transient current distribution solving code.

Creating a geometry is as simple as adding points, specified by its coordinates in three dimensions and a characteristic length term. The characteristic length term specifies how far from the point the next nearest mesh nodes should be added by the meshing algorithm. Lines are added by selecting points to connect. Planes are generated by creating loops of lines. To create a mesh, physical surfaces must then be defined from the planes and lines previously defined. This sets the parameters used by the meshing portion of GMSH .

For TRANSEDDY to function properly, both the solution surface and the edges for the current 'entry' and 'exit' must be defined as 'Physical Surfaces' and 'Physical Lines' respectively. This is to identify the boundary nodes that will be fixed in value in the iterative solver for the impressed source voltage.

An example of a geometry file for a simple square test case is below in Table 4.2 and its associated display from GMSH is shown in Figure 4.1

For more detailed information on functionality of the GMSH geometry creation program, refer to the GMSH user's manual[23].

If the .geo file is loaded in GMSH , selecting the 'mesh' tab will bring up the meshing interface in the GUI. We are working with two dimensional surfaces, but they can be orientated in any three dimensional arrangement, so we select the '3D' meshing option. This will generate a mesh on the defined physical surface. The .msh format saves the mesh data

```
-----------------------------------------------------
// Gmsh project created on Mon Apr 14 12:37:35 2008
Point(1) = {0,0,0,0.1};
Point(2) = {1,0,0,0.1};
Point(3) = {1,1,0,0.1};
Point(4) = {0,1,0,0.1};
Line(1) = {1,2};
Line(2) = {2,3};
Line(3) = {3,4};
Line(4) = {4,1};
Line Loop(5) = {1,2,3,4};
Plane Surface(6) = {5};
Physical Surface(100) = {6};
Physical Line(10) = {1};
Physical Line(11) = {2};
Physical Line(12) = {3};
Physical Line(13) = {4};
-----------------------------------------------------
```

Table 4.2: Example .geo file for a square geometry in GMSH

in columns of node data, followed by columns of element data. This will be explained in further detail in Section 4.2. The meshing algorithms implemented by GMSH will generate an unstructured triangular grid. The size of the individual elements will vary according to the characteristic length of the nearest specified geometry point. With this we are able to create variable sized elements within our surface. The computational savings of having unstructured, variable sized mesh elements will be illustrated in Chapter 5. An example of a mesh for a unit square is given in Figure 4.2 while an excerpt from the .msh file is in Table 4.3. The general format for a .msh file consists of the general components as shown in Table 4.4. For the element listing line in Table 4.4, `Elm-number` refers to the element identifier, `elm-type` references the type of element. Type 1 is a line. Type 2 is a triangle[23]. GMSH has 31 distinct types of finite element objects, but we only use the simplest order of lines and triangles. `number-of-tags` is always 3 in this version of GMSH . This sets up the next three numbers to follow. The first digit is the physical identifier tag assigned from the geometry file. The second digit refers to the geometry identifier number again assigned in the geometry file. The third tag is always zero for this version, with possible relevance if there were volumes in the model. After the tags have been listed, the nodes are listed, two for a line and three for a triangle. Node ordering for a triangle is a right handed coordinate

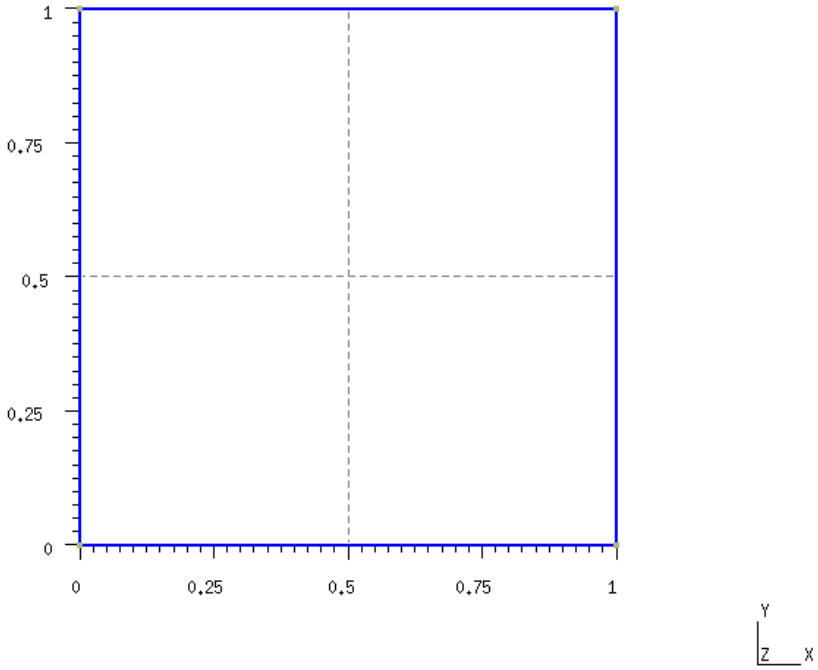

Figure 4.1: Screen capture from GMSH displaying a unit square geometry.

system.

Referencing the GMSH user's manual[23] again, the ordering for nodes for lines and triangular elements are presented in Table 4.5.

4.2 Parsing mesh files

Once the mesh file has been saved it is ready to be read by TRANSEDDY . From the command prompt, the user only needs to enter the file name for the mesh without the file extention. TRANSEDDY will now use the mesh file name as the key for creating output files and folders.

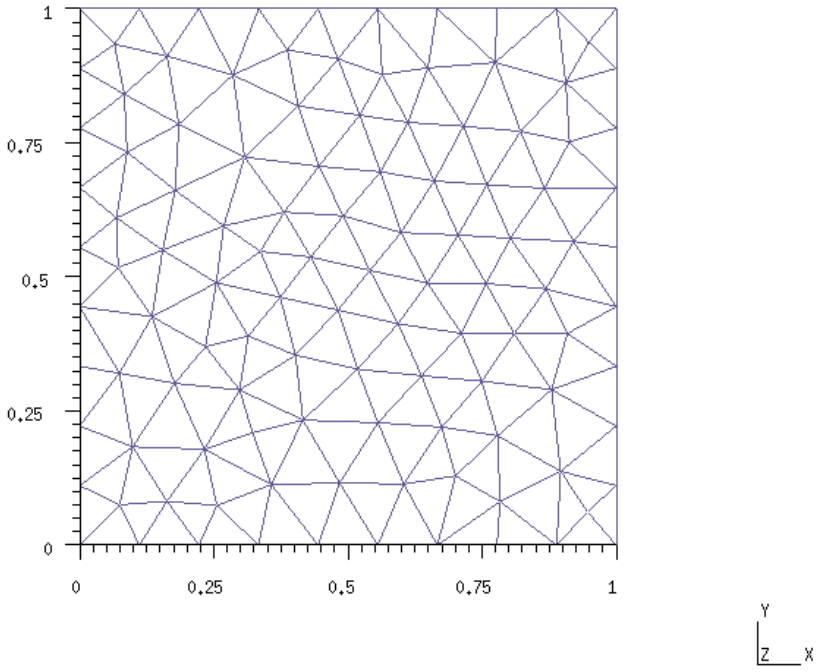

Figure 4.2: Screen capture from `GMSH` showing a meshed unit square.

4.2.1 Reading mesh data

A function was written to read in the .msh file and store the data in arrays in memory. The format for the mesh file helps us in this process as the total number of nodes and elements are stated prior to the actual listings. The function `readmesh` takes a string for the file name, and pointers to matrix data structures for locations to store the node data and the element data as inputs.

```
void readmesh(string inname, Matrix& nodedata, Matrix& elemdata)
```

The function then begins to read the file. Following the format of the .msh file, `readmesh` reads in each comment line beginning with the $ symbol, checks to see if it is proper, then

```
-----------------------------------------------
$MeshFormat
2 0 8
$EndMeshFormat
$Nodes
113
1 0 0 0
2 1 0 0
3 1 1 0
4 0 1 0
5 0.1111111111108444 0 0
...
112 0.9487268238852733 0.9377452106882056 0
113 0.07447428536744624 0.07529042718247041 0
$EndNodes
$Elements
224
1 1 3 10 1 0 1 5
2 1 3 10 1 0 5 6
...
224 2 3 100 6 0 58 113 102
$EndElements
-----------------------------------------------
```

Table 4.3: Example .msh file for a square geometry in GMSH

discards. When the parsing of the file reached the total number of nodes, a function call is made to resize the node data matrix. A loop then cycles through the nodes until the final node has been reached, then stores the node identifier number and it's three coordinates as doubles (doubles are double precision variables in C++).

The function then begins a similar process for storing elements beginning with resizing the storage matrix. The data that is saved consists of the element identifier, element type, physical object and geometry object identifiers, and finally the member nodes. At this point, the data structure of elements contains both lines and triangles. From the line data we will be able to extract which nodes are fixed on the lines designated as the 'current' entry and exit. The triangle information will be used to construct the finite element solution space.

4.2.2 Extracting edge data from triangle data

A function was written to separate the lines and triangles from the elements data structure, and store them in their respective data arrays.

```
void parseElements(const Matrix& elemdata, Matrix& triangles, Matrix& lines)
```

```
-------------------------------------------------
$MeshFormat //comment line to indicate GMSH version to follow
2 0 8         //GMSH version number
$EndMeshFormat //comment line to end GMSH version identifier
$Nodes //comment line to indicate start of node listing
N           //total number of nodes
node_number_1 X_coord Y_coord Z_coord
...
node_number_N X_coord Y_coord Z_coord
$EndNodes //comment line to indicate end of node listing
$Elements //comment line to indicate start of element listing
M //total number of elements
   //Elm-number elm-type number-of-tags < tag > ... < node-number-list >
1 1 3 10 1 0 1 5
   //line 1 belongs to phys-obj 10, geo-obj 3, and is bound by nodes 1 and 5
..
37 2 3 100 6 0 40 29 30
   //triangle 37 belongs to phys-obj 100, geo-obj 6,
   //and is bound by nodes 40, 29 and 30
..
$EndElements //comment line to indicate end of element listing
-------------------------------------------------
```

Table 4.4: General Format for a *.msh file in `GMSH` , adapted from the `GMSH` User's Manual[23]

The function sorts through the elements structure, keeping a running count, until the element type identifier changes from 1 to 2. These correspond to `GMSH` 's built-in element identifier scheme as described in Section 4.1. With the number of lines extracted, the number of triangles can be found taking the total number elements and subtracting the number of lines. The new storage matrices for lines and triangles are now resized to their respective length and then the element matrix is divided among them.

The storage matrix of line data is only useful as a means of identifying nodes that are to be fixed in the numerical Poisson's equation solver. To build the coefficient matrices that will be used in solving Poisson's equation and implementing the integral equation solution for eddy currents, the edges must be extracted from the list of triangles. The functions in Table 4.6 were written to extract the individual edges from the list of triangles. `ExtractEdges` cycles through the list of triangles and checks each edge of the triangle with the list of edges already found. This is accomplished by calling the function `edgealready` which returns a certain value if the edge exists in the list of edges. If it hasn't been found so far, the new edge is added to list.

The edge data consists of the nodes at ends, the three component values of the unit

```
-------------------------------------------------
Line:

0----------1 --> u

Triangle:

v
^
|
2
|‘\
|  ‘\
|    ‘\
|      ‘\
|        ‘\
0----------1 --> u
-------------------------------------------------
```

Table 4.5: Node numbering for lines and triangles in GMSH , taken from the GMSH User's Manual[23]

```
-------------------------------------------------
void ExtractEdges(const Matrix& Triangles, const Matrix& Nodes, Matrix& Edges)
int edgealready(int nodea, int nodeb, const Matrix& Edges)
void edgehat(Matrix& Edges)
-------------------------------------------------
```

Table 4.6: Function prototypes for edge data extraction.

```
-----------------------------------------------------
0.00E+000 0.00E+000
3.00E-006 1.87E-001
6.00E-006 3.68E-001
9.00E-006 5.36E-001
1.20E-005 6.85E-001
1.50E-005 8.09E-001
1.80E-005 9.05E-001
2.10E-005 9.69E-001
2.40E-005 9.98E-001
2.70E-005 9.92E-001
3.00E-005 9.51E-001
3.30E-005 8.76E-001
3.60E-005 7.71E-001
3.90E-005 6.37E-001
4.20E-005 4.82E-001
4.50E-005 3.09E-001
4.80E-005 1.25E-001
5.10E-005 0.00E+000
-----------------------------------------------------
```

Table 4.7: Sample data for a half period of a 10 kHz sinewave as used for

vector of the edge, and a list of the triangles it belongs to. The unit vector of the edge is used later in the formulation of the $[P]$ matrix. When stuffing the edge matrix with values, the unit vector is first filled with the vector from the first ordered node to the second. Then using function `edgehat`, the vector is scaled to unit values.

4.3 Reading time–voltage data for time stepping

To allow for the most flexibility to transient time-voltage input signals, `TRANSEDDY` has been written to read from a file the two column data for the time and voltage values. The time step size for the time-stepping solution is taken directly from the file. Additionally, this method allows for future flexibility if non-linear time steps are desired, as well as the ability to implement a print-step identifier.

```
void readVfromfile(string inname, Matrix& Vsource);
```

A sample data file for half a sine wave at 10 kHz is presented in Table 4.3 in the raw data format and as a plot in Figure 4.3.

The function opens the data file for reading, then pushes the data onto the data structures until the end of file has been reached.

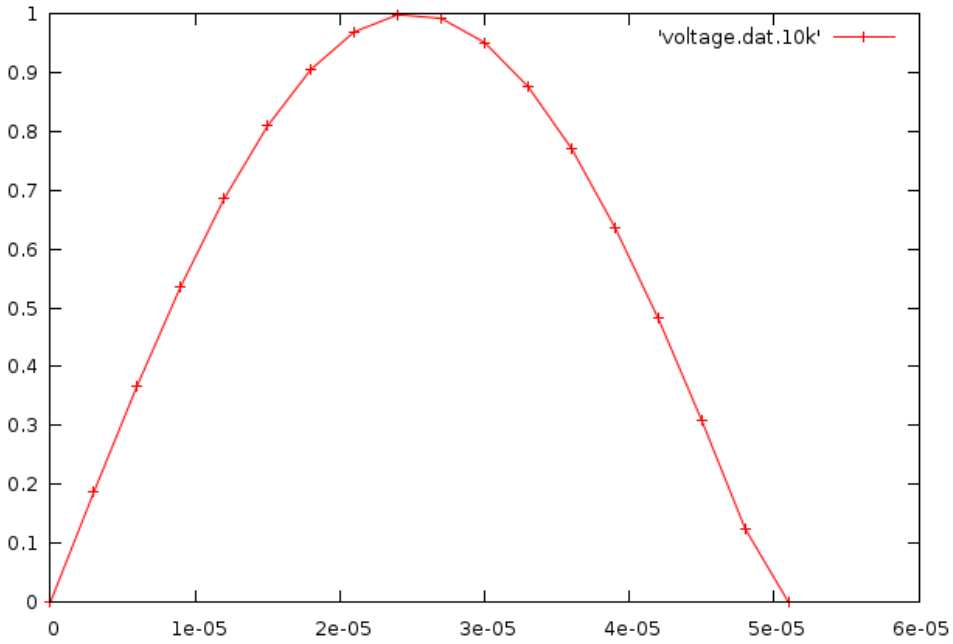

Figure 4.3: Plot of a sample time-voltage data file 'voltage.dat' for a half sinewave at 10 kHz

4.4 Construction of coefficient matrices

The coefficient matrices are described mathematically in Chapter 3, Section 3.4 by eq.(3.57) through eq.(3.62). All the coefficient construction functions share a common structure. They all have a nested for loop structure to cycle through the elements being referenced, either edges or nodes, or both. There is an evaluation section to identify if the nodes or edges in question belong to the same particular triangular element, and finally a function call to an integration function that actually carries out the math to evaluate the coefficient for each matrix location. Each function to evaluate the integrals of basis functions requires the use of the transformation matrix.

```
Matrix Transformation(Matrix coordinates);
```

This function takes in the nine coordinates of the triangle element's nodes and returns a matrix containing the triangle's coordinates transformed to the first quadrant of the x–y plane.

4.4.1 Building the [C] matrix

The $[C]$ matrix, which is used in the iterative solution to Poisson's equation, is nodes, by nodes in size, square, symmetric and sparse. It is constructed with the `buildCmat` function call. The function cycles through three nested 'for' loops. The outer loops cycle through the nodes, and the inner loops cycles through the triangles. For each element of $[C]$, the inner loop checks to see if i and j are both members of the current triangle. The function `ismember` checks as to whether or not the node is a member of the particular triangular element. It returns an integer identifier. This identifier corresponds to the location on the local triangle of which the node is found, or zero if the node is not a member. If both nodes i and j return integer values, then they belong to the same element, and a contribution to the term C_{ij} is produced by the integral over the triangle of the gradients of the basis functions. The function `IntGradPhiGradPhi` performs the integration by inserting the relevant coordinates of the triangle and the node locations into the formulas for the analytic integral derived in Chapter 3 Section 3.6. While there isn't a section for the mathematical derivation of the $[C]$ matrix, careful examination will show that the integration of the gradient of the nodal basis functions is the same as for the $[D]$ matrix formulation. The function prototypes used to construct the $[C]$ matrix are found in Table 4.8. Provided conductivity is constant in the solution space, $[D]$ and $[C]$ can be used interchangeably. Once the conductivity changes due to the problem specification, they are treated separately. For the simplified case of a constant and homogeneous conductivity, only use the $[D]$ matrix to save computation time.

```
-------------------------------------------------
void buildCmat(const Matrix& nodedata,const Matrix& elemdata,
         SymmetricMatrix& Cmat);
int ismember(const Matrix& elementdata, int nodei, int element)
double IntGradPhiGradPhi(int T,const Matrix& tridata,
         const Matrix& nodedata, int Nid1,int Nid2)
-------------------------------------------------
```

Table 4.8: Function prototypes used to construct the $[C]$ matrix in `TRANSEDDY` .

4.4.2 Building the [S] matrices

The $[S]$ matrix is of the size edges by edges and is square, symmetric and sparse. To construct individual elements, S_{ij} another nested 'for' loop structure was coded into `BuildSmat`. In order to save computation time, there is no inner loop to sort through triangles and return edge membership. The storage matrix of edges has been appended to contain membership information. For a single element, S_{ij}, there is no need to again sort through all the triangles and perform comparison checks. Edge i and edge i compare membership data, and where a commonality exists, then the integration of the shape functions over that particular triangle is performed with the function `IntShapeFunction`. The reduction of three loops down to two created a savings in operational time for this function of $O(n^3)$ down to $O(n^2)$.

The function `IntShapeFunction` takes as its inputs the addresses of the storage matrices, as well as identifiers for the specific triangular element and the edges to be evaluated. An integer identifier also is input which informs the function of what combination of edges for the local triangle are to be evaluated. The function then performs the triangle coordinate transformation and evaluates the integral of the shape functions over the triangular region.

There are actually two $[S]$ matrices. The normal one does not include the resistivity in it's formulation, and the other does. This matrix is called the $[pS]$ matrix in code. They are both created in `BuildSmat` by simply adding a multiply statement to the inner loop for the $[pS]$ matrix. When the conductivity of the material is allowed to change with time, the $[pS]$ matrix will require reconstruction taking in to account the new material properties, while the $[S]$ matrix is independent of material properties and is a function of geometry only. The same function can be used to reconstruct only the $[pS]$ matrix by adding an argument to the function call to selectively rebuild only $[pS]$.The function prototypes used to construct the $[S]$ matrix are found in Table 4.9.

```
-------------------------------------------------
void BuildSmat(const Matrix& tridata, const Matrix& edgedata, const Matrix& nodedata,
               SymmetricMatrix& Smat,SymmetricMatrix& pSmat,const Matrix& conductivity);
int edgeoftriangle(int T,const Matrix& tridata,int E, const Matrix& edgedata);
double IntShapeFunction(int T,const Matrix& tridata,int E1,int E2,
               const Matrix& edgedata,const Matrix& nodedata, int Eid1,int Eid2);
-------------------------------------------------
```

Table 4.9: function prototypes used to construct the $[S]$ and $[ps]$ matrices in `TRANSEDDY` .

4.4.3 Building the [G] matrices

The $[G]$ matrix is edges by nodes in size, and sparse. It links edge terms with the gradients of nodal values, as such it is not a square matrix. The $[G]$ matrix, like the $[S]$ in the previous section, also has two forms, one with the material conductivity and one independent of material conductivity. The function `BuildGmat` is a nested for loop consisting of an outer loop of the edges, with an inner loop of the nodes. A third loop is not needed for searching through triangles when edge membership information is included with the edge storage matrix. The inside loop only need to check to see if the current node belongs to the triangle parent of the edge and evaluate the interior integral accordingly. The function `IntShapeGradPhi` performs the integral over the triangle for the combination of edge shape functions and nodal basis functions. The $[cG]$ matrix is a separate matrix of the same information, with the exception that material properties are present inside the evaluation of the integrals for each sub-volume. In the case of constant conductivity, $[cG]$ appears is if we multiplied the conductivity to every term in $[G]$. When conductivity is not constant, each element's conductivity contributes to the individual terms of the matrix separately. The function prototypes used to construct the $[G]$ matrix and the $[cG]$ matrix can be found in Table 4.10. As with the case of the $[pS]$ reconstruction in the case that conductivity is allowed to change with time, $[cG]$ would also need to be rebuilt. The same function can be used, with the addition of an argument to the function call to selectively rebuild only $[cG]$.

```
-------------------------------------------------
double IntShapeGradPhi(int T,int M, const Matrix& tridata,
        const Matrix& edgedata,const Matrix& nodedata, int Eid1,int Nid2);
void BuildGmat(const Matrix& tridata, const Matrix& edgedata,
        const Matrix& nodedata, Matrix& Gmat,Matrix& cGmat,const Matrix& conductivity);
-------------------------------------------------
```

Table 4.10: Function prototypes used to construct the $[G]$ matrix and the $[cG]$ matrix in `TRANSEDDY` .

4.4.4 Building the [D] matrix

The $[D]$ matrix is nodes by nodes in size and is square, symmetric, and sparse. This matrix is used when calculating the "eddy voltage". It contains material conductivity information. For the case of constant conductivity, this matrix is the same as the $[C]$ matrix discussed earlier. `buildDmat` has the now familiar nested 'for' loop structure with the outside loops cycling through nodes and the inner loop through triangles. This function multiplies the result of the integral over the triangle of the nodal basis functions by the material conductivity of each respective element. Function prototypes used to construct the $[D]$ matrix can be found in Table 4.11.

```
-------------------------------------------------
double IntGradPhiGradPhi(int T,const Matrix& tridata,
        const Matrix& nodedata, int Nid1,int Nid2);
void BuildDmat(const Matrix& tridata, const Matrix& nodedata,
        SymmetricMatrix& Dmat, const Matrix& conductivity);
-------------------------------------------------
```

Table 4.11: Function prototypes used to construct the $[D]$ matrix in `TRANSEDDY` .

4.4.5 Building the [P] matrices

The $[P]$ matrix is edges by edges in size, square, but not necessarily symmetric. It is a dense matrix with a non-zero term in every location. The $[P]$ matrix represents the physical interaction of currents in the solution surface with the observation edges electrical current. There could be separate $[P]$ matrices for the source current and the eddy current densities if they are contained in separate bodies. For the cases of self-consistent current density

in a thin conductor, these bodies are the same, therefore we only need one matrix for the conductor geometry. As for the case of the $[S]$ matrix, the $[P]$ matrix consists of a nested 'for' loop structure, looping through every edge in each loop. Each edge in the inner loop adds to each individual term of P_{ij} for every triangular element to which it belongs. This integral has the potential to have a singularity when the observation edge and the target edge share the same element. A function to evaluate the monomial over a triangular region, as described in Chapter 3 Section 3.6.4 was adapted from the numerical quadrature algorithm TOM612[50,51,52] by K. H. Carpenter and implemented in `C++`. A wrapper function was written to generate the proper input form required by the `greenint2dtri` function. Function prototypes used to construct the $[P]$ matrix are listed in Table 4.12.

```
-------------------------------------------------
void BuildPmat(const Matrix& tridata, const Matrix& edgedata,
               const Matrix& nodedata, Matrix& PMat);
double IntShapeGreen(int T,const Matrix& tridata,int P,int Edge,
               const Matrix& edgedata,const Matrix& nodedata,int edgeid);
double greenint2dtri_wrapper(const Matrix& vertex,const Matrix& field,int xp, int yp );
-------------------------------------------------
```

Table 4.12: Function prototypes used to construct the $[P]$ matrix in `TRANSEDDY` .

4.5 Solution of Laplace's equation

The Laplace Equation is solved by means of the iteration method[19]. This makes use of the $[C]$ matrix, which we note is the same as the $[D]$ matrix for cases of constant and uniform conductivity.

To solve Laplace's equation,

$$\nabla^2 V = 0 \tag{4.1}$$

with finite elements, we again start with the basic definitions from the finite element approximation of the function. We can write the value of the voltage $V(\vec{r})$ with in a particular triangular element on the finite element mesh as a sum of the value of V and the nodal basis functions evaluated at the point $\vec{r}$.

$$V(\vec{r}) = \sum_{i=1}^{3} V_i \phi_i(\vec{r}) \tag{4.2}$$

Sadiku[19] states that the energy functional for the Laplace equation is

$$F(V) = \frac{1}{2} \int_s \left[\epsilon |\nabla V|^2 \right] dS \tag{4.3}$$

where $F(V)$ represents the total per energy per unit area within an element. The terms under the integral correspond to the energy density in the electrostatic system, $\frac{1}{2}\mathbf{D} \cdot \mathbf{E} = \frac{\mathbf{1}}{\mathbf{2}}\epsilon|\nabla \mathbf{V}|^{\mathbf{2}}$.

The functional in eq.(4.3) can be written in terms of the finite element approximations for V as

$$F(V) = \frac{1}{2} \sum_{i=1}^{3} \sum_{j=1}^{3} \epsilon V_i \left[\int \nabla \phi_i \cdot \nabla \phi_j dS \right] V_j \tag{4.4}$$

and in matrix form is expressed as

$$F(V) = \frac{1}{2} \epsilon [V]^t [C^{(e)}][V] \tag{4.5}$$

where, similarly to the definition of the $[D]$ matrix from eq.(3.61) and described in Section 4.4.1 the individual elements of the $[C]$ are defined as

$$C_{ij}^{(e)} = \int \nabla \phi_i \cdot \nabla \phi_j dS \tag{4.6}$$

.

A term for the energy functional across all 'N' elements 'e' in a finite element solution region can be expressed as the sum of the individual energy functionals.

$$F(V) = \sum_{e=1}^{N} F(V_e) = \frac{1}{2} \epsilon [V]^t [C][V] \tag{4.7}$$

$[V]$ is a column matrix of values of V_i. The functional is now minimized by differentiating by ϕ_i and setting the result equal to zero.

In order to solve this minimization process, we chose the iterative method. The advantage of the iterative method over the band matrix method[19] is that for sufficiently large matrices

of $[C]$, there are less computer operations in the iterative method due to the necessity to invert the matrices in the band matrix method. Matrix inversion for large matrices can be a lengthy process[19].

The general form of the functional in matrix notation appears as

$$F = \frac{1}{2}\epsilon \begin{bmatrix} V_1 & V_2 & V_3 & \dots & V_n \end{bmatrix} \begin{bmatrix} C_{11} & C_{12} & C_{13} & \cdots & C_{1n} \\ C_{21} & C_{22} & C_{23} & \cdots & C_{2n} \\ \vdots & \vdots & \ddots & \vdots & \\ C_{n1} & C_{n2} & C_{n3} & \cdots & C_{nn} \end{bmatrix} \begin{bmatrix} V_1 \\ V_2 \\ V_3 \\ \vdots \\ V_n \end{bmatrix}. \tag{4.8}$$

We can easily see we can minimize the energy by taking the partial derivative of F by the individual electric potentials and setting that equal to zero,

$$\frac{\partial F}{\partial V_i}, \qquad i = 1, 2, \dots, n. \tag{4.9}$$

What we have left is a general expression for solving the electrical potential at all the nodes based on its neighbors. For a general free node voltage V_k we have

$$V_k = -\frac{1}{C_{kk}} \sum_{i=1;1\neq k}^{n} V_i C_{ki} \tag{4.10}$$

The iterative solution can be implemented by assigning the fixed node voltages, making initial guesses for the free node values, and calculating the energy functional. Then the free nodes are replaced by the summation in eq.(4.10). At this point the energy functional is calculated again by eq.(4.7). A comparison is made between the two energies, and if the change in energy is less than a prescribed error tolerance, then we say that we have converged to a minimum energy state and have the solution to $\nabla^2 V = 0$ on the solution surface. The tolerance is set by the user to the desired accuracy. If the change in energy is not less than the error tolerance, then the free nodes are again calculated from the previous results and the fixed nodes. This cycle repeats until the error is less than the set tolerance.

4.5.1 The addition of Successive Overrelaxation (SOR) to the solution of Laplace's equation

For large meshes, the number of iterations required to reach the minimized error limit can be vary large, requiring a lot of computational time. In an effort to reduce the number of iterations, Successive Overrelaxation[55] (SOR) was implemented in the iterative solver. SOR increases the speed of convergence for the iterative process by the addition of an acceleration term α to eq.(4.10), which then becomes

$$\phi_k^n(t) = -\alpha \frac{1}{C_{kk}} \sum_{i=1;1\neq k}^{e} \phi_i^{n-1} C_{ki} + (1-\alpha)\phi_k^{(n-1)} \tag{4.11}$$

where 'n' is the iteration step and 'e' is is the number of nodal elements. To accelerate the convergence of the solution the value of the acceleration term should be $1 < \alpha < 2$. A value of α larger than 2 can cause the successive iterations to become unstable. The larger the mesh size, the more SOR improves computation time to convergence.

It is possible that α is too large and causes a fluctuation in the convergence rate where the value of the residual increases rather than decreases between steps. For a mesh with 'n' nodes, there will be an optimum value for α where the residual steadily drops to it's minimum value (or the error tolerance limit we set). Without being able to solve for the optimum value for any possible mesh which could be used as an input to the program, we implemented a test for the proper sizing of α and a SOR reduction percentage. This test compares the residual from the last iteration to the newest iteration. If the residual has increased, α is multiplied by a percentage to reduce it's value for the next iteration.

In this way, we do not need to know the optimum value for α to accelerate the iterative solutions convergence. We can overestimate α to begin with, and allow it to be reduced to the approximately the proper size.

To implement the solution of Laplace's equation in `TRANSEDDY` several functions were written, the prototypes for which can be found in Table 4.13.

`InitVT` takes as its inputs the storage matrices of the node data, line data, node voltages, and the numerical value for the voltage at the fixed nodes, and the identifiers of which lines

```
-----------------------------------------------
 void InitVT(const Matrix& nodedata,const Matrix& lines,
          Matrix& V,double voltage,int fixed, int ground);
 void SolveV(const Matrix& Cmat, Matrix& V, double error);
 double CalcW(const Matrix& Cmat,const Matrix& V);
 void iterateV(const Matrix& C, Matrix& V);
-----------------------------------------------
```

Table 4.13: Function prototypes for functions used in the solution of Laplace's equation in `TRANSEDDY` .

are the fixed voltage and ground potentials. The function then sorts through the nodes and assigns an identifier whether it is a fixed or free node. If it is a fixed node, the voltage is then assigned from the source file.

`SolveV` performs the iterative solution to the minimization of the energy functional in eq.(4.7) via the spatial averaging represented by eq.(4.10). First a base line value for the energy is required. An initial value is found via the `CalcW` function. Next it performs the averaging. `CalcW` is called on the processed voltage data, and a comparison of the values from the previous state is made. If this is found to be greater than the prescribed error, the process will repeat until such a time as the change in the energy states is less than the error term. Appended to this function is the process of reducing the SOR factor if the residual has not decreased from the previous step.

`CalcW` performs the calculation of the energy functional in eq.(4.7) via matrix multiplication and returns the result.

`iterateV` implements the averaging of the nodal voltage values according to the $[C]$ matrix coefficients. Successive Overrelaxation is implemented here to accelerate the rate of convergence for the iterative Laplace solver.

4.6 Time-stepping solution for eddy current

After the coefficient matrices have been built we are ready to begin the time-stepping solution for the total current density. The length of the input time/voltage data storage matrix determines the number of steps. Figure 4.4 presents the flow of the time stepping process

graphically, including what steps would be required to add in order to perform conductivity changes due to Ohmic heating of the conductor.

The process begins by initializing the storage vector of source voltages to the new impressed value for the fixed nodes as specified by the source file. Next the iterative solution for the Poisson Equation is performed by the function `SolveV`. To solve for the source current from the source voltage we need to take the gradient of the voltage multiplied by negative one and the conductivity. This step is accomplished via matrix manipulation with the stiffness and the gradient matrices.

```
Jsource=Smat.i()*(-1.0*cGmat*V.column(1));
```

With the new source current provided, and previous time quantities either initialized to zero, or existing from previous steps, we solve for the prediction current. We see that eq.(3.53) reduces to a linear system, $Ax = b$, with x being the vector of edge values of the prediction current, $[A]$ is from the combination the $[S]$ matrices;

$$(\theta\tau[pS]) + (\mu[S][P]).$$

The right hand side of eq.(3.53),

$$\mu_0[S]\{[P_0]\{i_e^{t-1}\} - [P_e]\{i_0^t - i_0^{t-1}\}\} - (1-\omega)\tau\rho[S]\{i_e^{t-1}\} - \tau[G]\{V^{t-1}\}$$

reduces to a column vector, 'b'. To solve $Ax = b$, we perform a LU decomposition[56] on this matrix. The LU Decomposition is accomplished in `NEWMAT` by the declaration of a `Croutmatrix A`. When A is initialized, a LU decomposition is performed automatically. In an optimization revision, the LU decomposition was performed with the `LAPACK` Library.

The stepped change in the eddy voltage is solved with a similar matrix algebra. In this case the $[D]$ matrix has the LU decomposition performed on it. The correction current also requires LU decomposition for solution. Now that we have the prediction and correction currents, they are simply added together to arrive at the eddy current for the current time

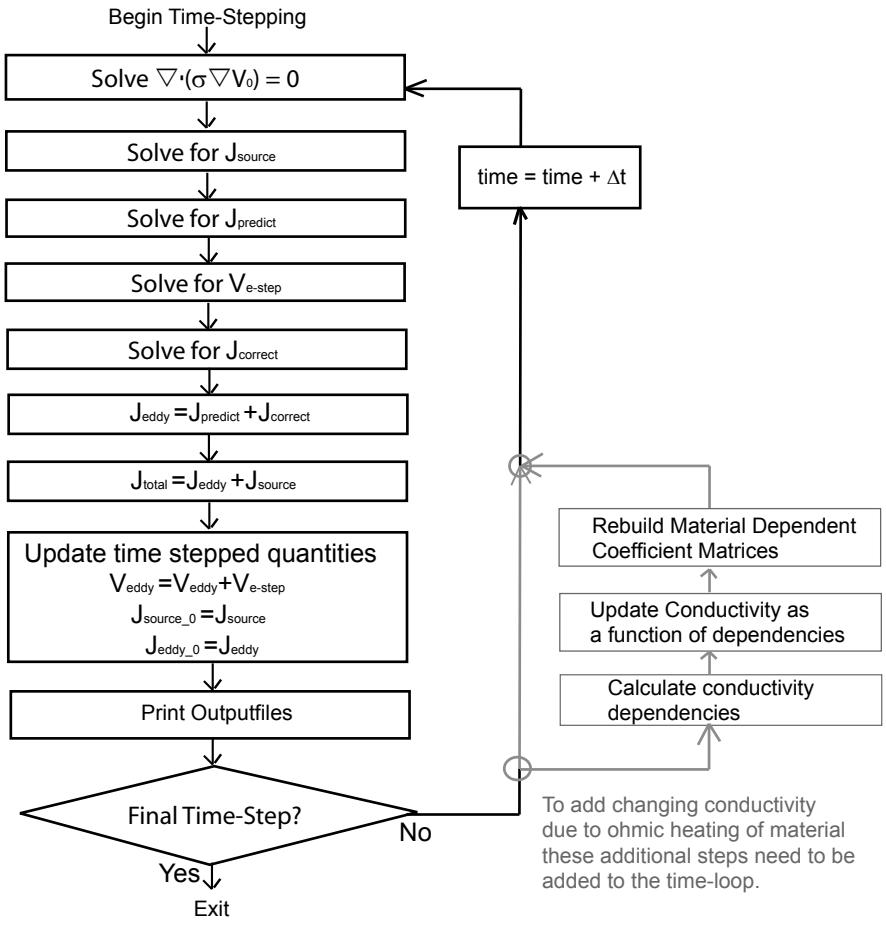

Figure 4.4: Flowchart for the time-stepping solution of the total current density. To implement time changing conductivity due to ohmic heating, additional steps for calculating the conductivity dependencies and then the conductivity as a function of those dependencies would be required.

```
-----------------------------------------------------
for(i=1;i<=Vsource.Nrows();i++){//------Solve given current from SS voltage
InitVT(nodes,lines,V,Vsource(i,2),fixed,ground);
cout<<"|----------time step "<<i<<" -----------|"<<endl;
cout<<"|-----Solving V_0 for t="<<Vsource(i,1) <<endl;
SolveV(Dmat,V,error);
cout<<"|-----Solving for Jsource---------|"<<endl;
Jsource=Smat.i()*(-1.0*cGmat*V.column(1));
//------build and solve for Jpredict Ax=b  x=A.i()*b
cout<<"|-----Solving for Jpredict--------|"<<endl;
CroutMatrix A = (Theta*timestep*pSmat)+(MU*Smat*Pmat);
b=MU*Smat*( (Pmat*Jeddy_0) - (Pmat*(Jsource-Jsource_0)) );
b = b - (1.0-Theta)*timestep*(pSmat*Jeddy_0);
b = b - timestep*(Gmat*Veddy);
Jpredict=A.i()*b;
cout<<"|-----Solving for Vestep----------|"<<endl;
Vestep = Dmat_test.i()*(Gmat.t()*Jpredict);
cout<<"|-----Solving for Jcorrect--------|"<<endl;
Jcorrect=-1.0*Smat.i()*cGmat*Vestep.column(1);
cout<<"|-------Jeddy = Jp + Jc-----------|"<<endl;
Jeddy=Jpredict+Jcorrect;
cout<<"|-----Adding total current--------|"<<endl;
Jtotal=Jsource+Jeddy;
cout<<"|------updating variables---------|"<<endl;
Veddy=Veddy+Vestep;
Jsource_0=Jsource;
Jeddy_0=Jeddy;
outname=innameroot+"/Jtotal_t_"+int2string(i)+".pos";
WritePosVTJ(Jtotal,nodes,triangles,edges,outname);
cout<<"     file "<<outname<<" saved"<<endl;
} //End time stepping loop
-----------------------------------------------------
```

Table 4.14: Sample C++ code for the time-stepping process.

step. The total current density then is the addition of the eddy and source currents.

The next procedures in the time-stepping process are to update the eddy voltage, and promote the source and eddy currents to the previous time step label and write any output files for this time step. A reduced sample of C++ code for the time-stepping process is shown in Table 4.14.

4.7 Generating output files

To get data out of `TRANSEDDY` suitable for plotting with `GMSH` , several file formatting functions were formulated. `GMSH` has a native output file format, `*.pos`. This file format consists of several key words and then series of data in appropriate order. The two formats which

we make use of are scalar triangle, 'ST', and vector triangle, 'VT'.

GMSH recognizes triangular elements and nodal interpolation of functions across the triangular elements. For a scalar function to be plotted on the triangular mesh, all that needs to be generated is a listing of triangles by nodes and values at those nodes. The first line contains the name for the view title in GMSH in quotes. within brackets the triangle listings start with the keyword first. For a scalar triangle, the key is 'ST'. Following this, in parenthesis separated by commas are the x, y, and z coordinates of the nodes in right handed orientation. Finally, in brackets, separated by commas are the scalar values on the nodes, listed in right hand numbering order. A sample section of a formatted scalar triangle .pos can be found in Table 4.15.

```
-----------------------------------------------------
View " Vsource.pos " {
ST(0.04099557342,0.9590680501,0,0,1,0,0,0.9473684211,0)
     {0.03371601599,0,0.04376288132};
ST(0.05263157895,1,0,0,1,0,0.04099557342,0.9590680501,0)
     {0,0,0.03371601599};
...
ST(x1, y1, z1, x2, y2, z2, x3, y3, z3){V1, V2, V3};
}
-----------------------------------------------------
```

Table 4.15: Sample syntax for a formatted scalar triangle .pos output file

To plot vector data on a triangular mesh, we use the keyword, 'VT', for vector triangle. The same format is followed, with the exception that we now have three vector components for each node on the triangular element. GMSH does not have the functionality to plot edge values on a triangular mesh and have the program interpolate the data correctly. To solve this limitation, for each triangle, we evaluate the vector values at the nodes based on the edge values, and report those values to GMSH . A sample listing for vector data can be found in Table 4.16. As the raw information from the solver is actually values associated with the edges of the triangular elements, it may be useful to view the actual edge values. For this a formatting function was written to output the data on the edges alone. GMSH uses the keyword 'VL', or Vector Line to plot the data on the lines of the mesh. This is accomplished by writing out a line for each edge, with the value of the vector component on each node of

```
-------------------------------------------------------
View " Jtotal_t.pos" {
VT(0.08305545794,0.8408118784,0,0,0.8888888889,0,0,0.7777777778,0)
   {0.05030431094,2573.657425,0,0.1578674736,2573.843246,0,-0.09072249495,2573.843246,0};
VT(0.0740327381,0.3206248163,0,0,0.3333333333,0,0,0.2222222222,0)
   {-0.03880048271,2573.883753,0,-0.007712838458,2574.064852,0,-0.2795134492,2574.064852,0};
...
VT(x1, y1, z1, x2, y2, z2, x3, y3, z3){Ex1, Ey1, Ez1, Ex2, Ey2, Ez2, Ex3, Ey3, Ez3};
}
-------------------------------------------------------
```

Table 4.16: Sample syntax for vector data formatted on triangular elements in a `.pos` output file

the line. A sample section of a Vector line formatted .pos file can be found in Table 4.17.

```
-------------------------------------------------------
View " Jsource_e.pos " {
VL(0.08305545794,0.8408118784,0,0,0.8888888889,0)
    {-1115.787348,645.8783242,0,-1115.787348,645.8783242,0}
VL(0,0.8888888889,0,0,0.7777777778,0)
    {-0,2573.451938,-0,-0,2573.451938,-0};
...
VL(x1, y1, z1, x2, y2, z2){Ex1, Ey1, Ez1, Ex2, Ey2, Ez2};
}
-------------------------------------------------------
```

Table 4.17: Sample syntax for vector data formatted on line elements in a `.pos` output file

The last information formatting function that we could make use of would be to plot the value of the conductivity (or resistivity) on the triangular mesh. As the conductivity is treated as a zero-order element (constant on the surface of an individual triangular element) we can use the Scalar Triangle keyword for plotting, and assign the value for the triangle to each node. A sample section of the `.pos` file can be found in Table 4.18.

Function prototypes for output file formatting functions can be found in Table 4.19.

4.8 Coding scheme for changing conductivity as a function of specific action

Conductivity is currently implemented as a constant for the entire mesh. However a few simple changes can be made to allow for inhomogeneous conductivity on the solution surface.

```
-----------------------------------------------------
View " Conductivity.pos " {
ST(0.08305545794,0.8408118784,0,0,0.8888888889,0,0,0.7777777778,0)
   {1000000,1000000,1000000};
ST(0.0740327381,0.3206248163,0,0,0.3333333333,0,0,0.2222222222,0)
   {1000000,1000000,1000000};
...
ST(x1, y1, z1, x2, y2, z2, x3, y3, z3){c,c,c};
}
-----------------------------------------------------
```

Table 4.18: Sample syntax for scalar data formatted on triangular elements in a `.pos` output file.

```
-----------------------------------------------------
void WritePosST(const Matrix& V,const Matrix& nodedata,
             const Matrix& triangledata, string outname );
void WritePosSTZ(const Matrix& conductivity,const Matrix& nodedata,
             const Matrix& triangledata, string outname );
void WritePosVL(const Matrix& E,const Matrix& nodedata,
             const Matrix& edgedata, string outname );
void WritePosVT(const Matrix& V,const Matrix& nodedata,
             const Matrix& Tridata, string outname );
void WritePosVTJ(const Matrix& J,const Matrix& nodedata,
             const Matrix& Tridata,const Matrix& edges, string outname );
-----------------------------------------------------
```

Table 4.19: Function prototypes for output file formatting functions.

Conductivity is stored in a storage vector with reference locations corresponding to the list of triangles. Various models exist for describing the resistivity of a metal as a function of temperature, power dissipation, or specific action[57,58] Depending on the particular solution method, a function can be written to calculate the electrical power in each element. We have to total current at each time step, therefore it would be a short step to evaluate the integral of the square of the currents for each triangle. Power is the square of the current times the resistance. With power per element, we could calculate a thermal rise, and then a new conductivity. Similarly, specific action links the time integral of the square of the currents directly to resistivity[57,58]. If changes in material properties are added, then several of the coefficient matrices require reformulation due to the conductivity falling under the integral for individual elements. This would greatly add to the time required to solve any time-stepped simulation.

4.9 Command-line entry operation

`TRANSEDDY` has been implemented with a command-line entry mode. When the program is executed, the program prompts the user for the name of the target mesh filename without the file extention. The user is then prompted for the name of the voltage data file. Successive overrelaxation (SOR) has been implemented within the iterative Laplace equation solver, and the user is prompted for a value for the SOR value (between 1 and 2). The final user prompt is to identify which lines from the geometry file correspond to the current entry and exit. After this, the program is running without any more user interaction. Time stamp data is printed to the screen for user gratification. Selected output from a test of `TRANSEDDY` is shown in Table 4.20.

```
-----------------------------------------------------
todd@todd-linux:~/transedd/test$ ../transeddy
|-----------Data Entry------------|
Input mesh file name without extension: (of type *.msh)
> test1
Mesh file test1.msh read successfully.

Enter voltage source data file name with extention of text
file type two column data | time voltage |
>voltage.dat
Data file voltage.dat read successfully
Enter SOR value (=>1, < 2):1
Using SOR value = 1
|----Label Dirichlet boundaries---|
Enter integer number identifier for physical line with fixed potential
>1
Enter integer number identifier for physical line with ground reference potential
>3
|---------------------------------|
|     Processing mesh file        |
|     ..Parsing elements..        |
|     ..Extracting edges..        |
|---------------------------------|
|--Building coefficient matrices--|
|---------------------------------|
|                                 |
|      Building [S] matrix        |
     progress time 2 seconds
|      Building [G] matrix        |
     progress time 3 seconds
|      Building [D] matrix        |
     progress time 3 seconds
|      Building [P] matrix        |
     progress time 7 seconds
|---------------------------------|
|----------time step 1 -----------|
|-----Solving V_0 for t=0
         solution took 0 iterations
     progress time 7 seconds
|-----Solving for Jsource---------|
     progress time 7 seconds
|-----Solving for Jpredict--------|
     progress time 7 seconds
|-----Solving for Vestep----------|
     progress time 7 seconds
|-----Solving for Jcorrect--------|
     progress time 7 seconds
|-------Jeddy = Jp + Jc-----------|
|-----Adding total current--------|
|------updating variables---------|
     file test1_outputfiles/Jtotal_t_1.pos saved
...
...
|------updating variables---------|
     file test1_outputfiles/Jtotal_t_2.pos saved
 Transeddy completed, program took 7 seconds
-----------------------------------------------------
```

Table 4.20: Example screen output from `TRANSEDDY` .

Chapter 5

Testing and validation of Transient current conduction code

A series of test cases were run to validate the transient current density solver. Validation is done by verification of adherence to electromagnetic theory and to similar performance with other solvers for simple cases. The test strip line from Chapter 2 was been recreated with `GMSH` and simulated with `TRANSEDDY` . Comparisons of 'AC' conduction skin effects, and of transient performance have been performed and the results are presented below.

5.1 Simple conducting square

The first test to verify proper execution of the time stepping methodology was to apply the program to a unit square with a voltage input signal or a ramp. A coarse mesh for the test can be found in Figure 5.1. This test mesh is fine enough to be able to evaluate the eddy current density solution, yet coarse enough that the program will complete quickly.

The initial test case for the evaluation of the time-stepping functions was a ramp of voltage from zero to one, in one time step. A graphical representation for the input file is presented in Figure 5.2

The source voltage distribution on a square with opposite ends held to zero and one volts respectively, has a trivial analytic solution. The voltage has a constant gradient with respect to the y direction and zero gradient along the x axis. The iterative solver used in

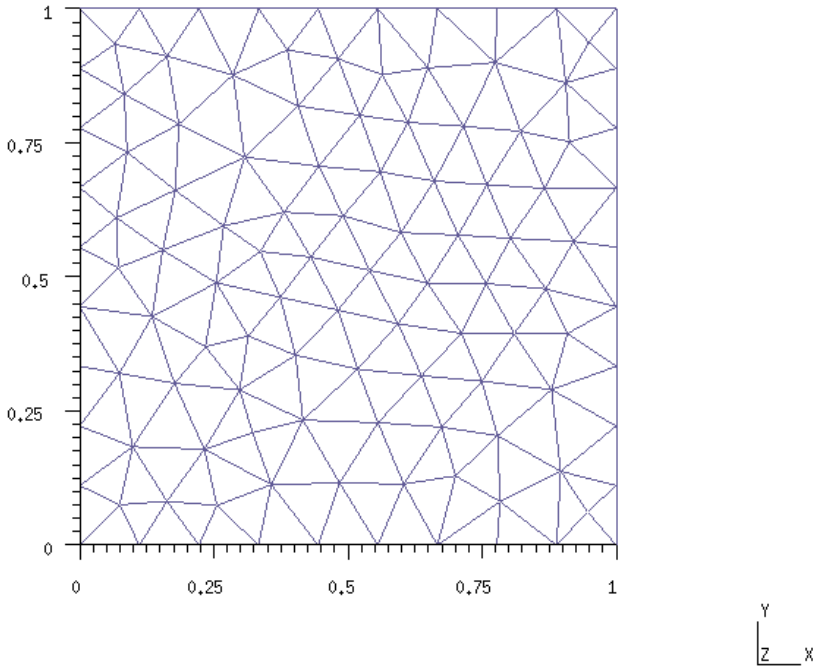

Figure 5.1: Screenshot from `GMSH` of a meshed square used in testing

`TRANSEDDY` produces this result with some residual error which will be more noticeable in the plot of the source current density once the gradient of the voltage has been calculated. A plot of the voltage on the test square mesh can be found in Figure 5.3.

With the application of the matrix algebra which implements the gradient of voltage, we arrive at the source current density, $\vec{J}_s$ on the surface mesh. A plot of $\vec{J}_s$ is presented in Figure 5.4 The current density everywhere should be a constant if this solution is to match the analytic solution of the constant gradient for a square. Here, we can see that the current density has a small ($< 0.1\%$) error though it should be a constant value.

This error is a byproduct of the iterative solution method. An illustration of an iterative solution for Laplace's equation on a one dimensional, ten segment line is shown in Figure

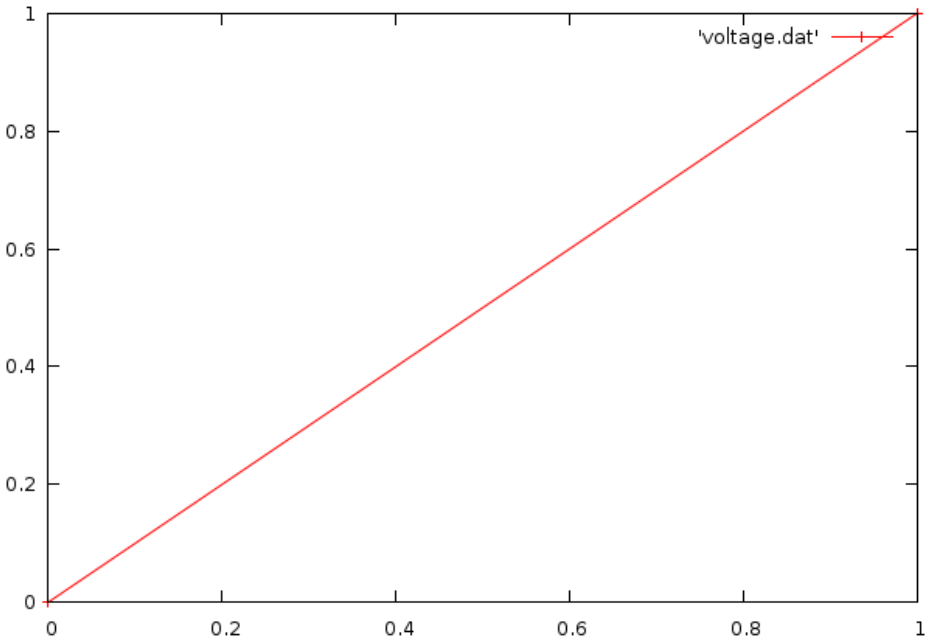

Figure 5.2: Plot of a ramp function used to test performance of solution steps in `TRANSEDDY`

5.5. At several levels of iterations performed we can see the progression of the voltage distribution. We know that the analytic solution is a straight line, and with the successive iterations of the averaging process, we see the voltage distribution approach that limit. The iterative solution will terminate when the relative error between steps drops below some tolerance threshold which we set. In this simple case of a ten segment one dimensional line, after 300 iterations the relative error is less than 1×10^{-8}. This example was performed without SOR. Had SOR been implemented, the number of iterations required to reach the desired threshold in relative error would have been reduced, however the nature of the error will be exactly the same as if we had not used SOR.

A plot of the relative error for this example case can be found in Figure 5.6. For a general solution, the number of iterations required to converge to a solution will increase with the number of nodes along the length. This behavior will show up again when we are observing the simulations of long strip lines like the ones examined in Chapter 2. If we take the gradient of the voltage from the example solution, we see the same type of error present.

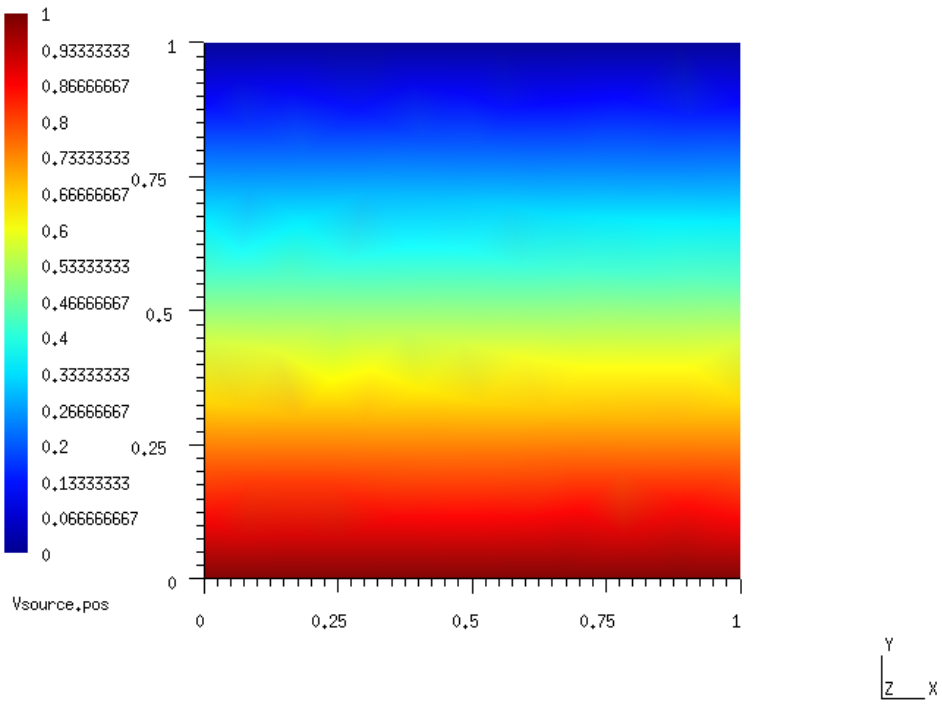

Figure 5.3: `GMSH` output of the Voltage on a test square with opposing edges fixed at one volt and zero.

Figure 5.7 shows the gradient of the one-dimensional voltage from the example problem for several high iteration steps. We see the increase in accuracy for increased iterations. This shows us that when a similar error appears in the $\vec{J}_s$ solution from `transeddy` that we can reduce this error by decreasing the tolerance error limit in the iterative solver.

The data for the current density is associated with the edges. A plot of how these values are actually represented by the code is presented in Figure 5.8. The magnitude of the edge value is stored, along with the direction of the edge. `GMSH` does not have implemented functionality to display and interpolate edge values with the edge shape functions, so for display and data processing, we use the nodal representation.

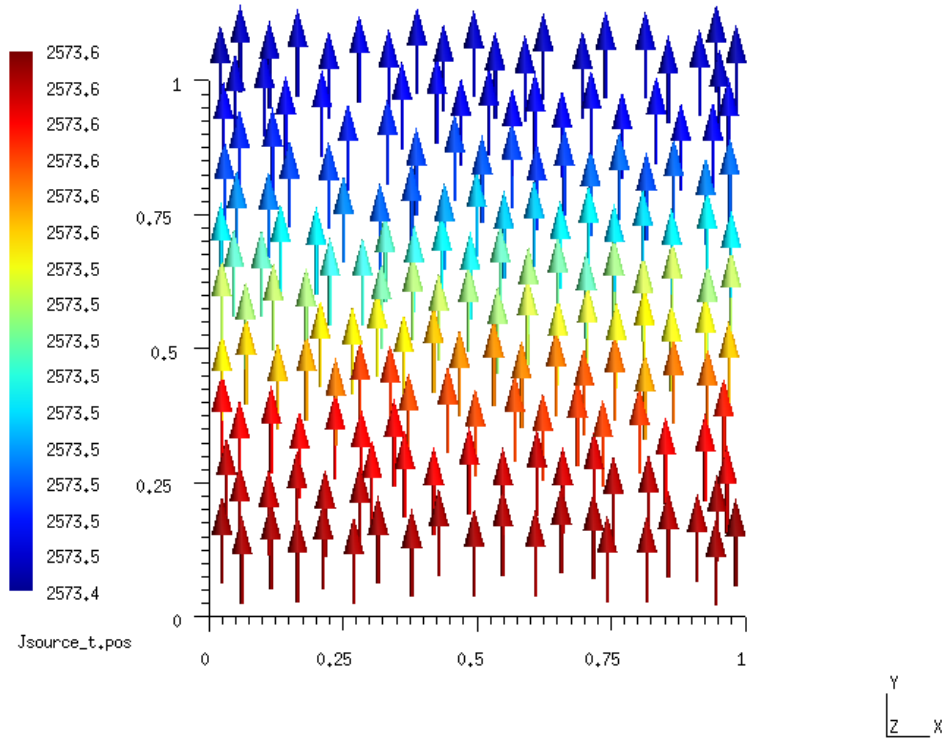

Figure 5.4: $\vec{J}_s$ plot from GMSH for the test square mesh. The source current density is derived from the gradient of the source voltage. Residual error in the iterative solution process for V_s produces the error in the magnitude of the current density. Lowering the tolerance error in the iterative solution will reduce this error.

To verify proper performance of the first step in the time-stepping process, we need to view the prediction current density. We know that the prediction current is produced by the magnetic vector potential. The magnetic vector potential is due to the change in the source and the eddy current densities from the present and previous time steps. It behaves according to eq.(3.12), or as $\frac{1}{distance}$ in general. for a square we would expect to see a concentration of more of the prediction current density in the center of the square where the average distance to every other part of the square is the smallest. In Figure 5.9 we see the GMSH output for the prediction current density stage of the time stepping process. As

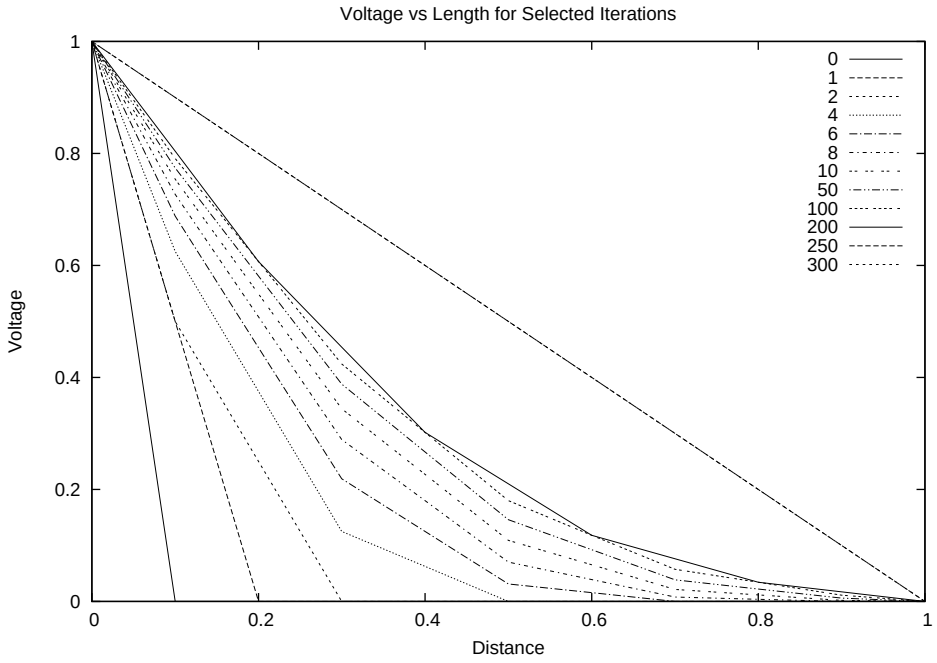

Figure 5.5: This graph represents a sample iterative solution of the Laplace Equation via the Iterative Method for a one-dimensional problem. Each line represents the solution at a number of steps into the solution. By 300 iterations, the ten steps of the voltage can be assumed to be the analytical solution for the voltage. This example was implemented without SOR

expected, it has a maximum at the center and has symmetry in both length and width.

The correction current density for the test square is presented in Figure 5.10. It should guarantee that the eddy current density does not leave the surface of the conductor. We would then expect the correction current density to oppose the direction of the prediction current density. Indeed we see this behavior.

With the addition of the prediction and correction current densities we are able to see the eddy current solution. Figure 5.11 shows the eddy current density. The boundary conditions are preserved (within the margin of error for the coarse mesh) and we see that the majority of the central path of the eddy current flows to oppose the change in the source current, which obeys Lenz's Law[59]

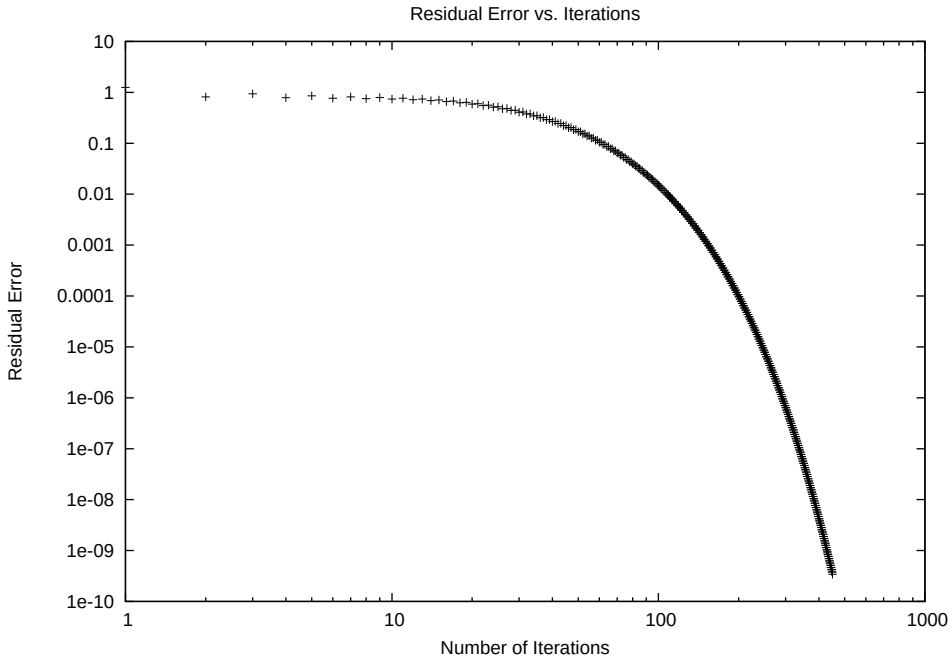

Figure 5.6: This graph represents the residual error in an iterative solution of Laplace's Equation for a one-dimensional sample problem presented in Figure 5.5. We can see that by 300 iterations for the sample problem the change in the relative error is less than 10^{-8}. This example was implemented without SOR.

We must now take one more step to arrive at the total current density for this test square case. The source current density added to the eddy current solution gives us this total current density. The total current density for the test square can be viewed in Figure 5.12. This figure for an example ramp in the source voltage meets expectations of exhibiting a skin effect, or current crowding at the edges of the square. This test case was performed with numbers chosen for convenience in analysis; the square having unit size, the voltage differential of 1 Volt, and the input time step of 1 second. This allowed examination of each stage, and verification of the coefficient matrices.

With evaluation of the basic performance of the time-stepping solution, we turn to solving actual physical problems that can be verified. To do this we will revisit the one inch

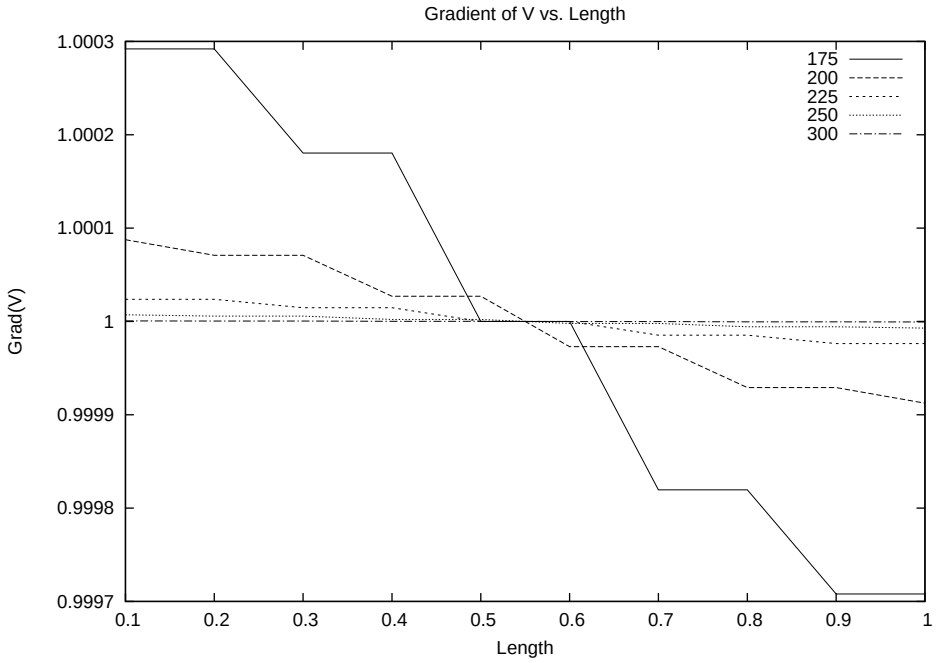

Figure 5.7: Example plot of the gradient of the voltage for the sample one-dimensional Laplace Equation. Higher number of iterations reduces the residual error in the voltage and thus the error in the pseudo-current represented by the gradient. The plot of the gradient for 175 iterations shows the largest error with a 0.03% error. The plot of the gradient of the voltage for 300 iterations shows a constant gradient. The error is less the resolution of the graph. This example was implemented without SOR.

wide strip lines examined in Chapter 2 with the magnetic loop probes, and with `SPICE` *via* the Partial Inductance Method.

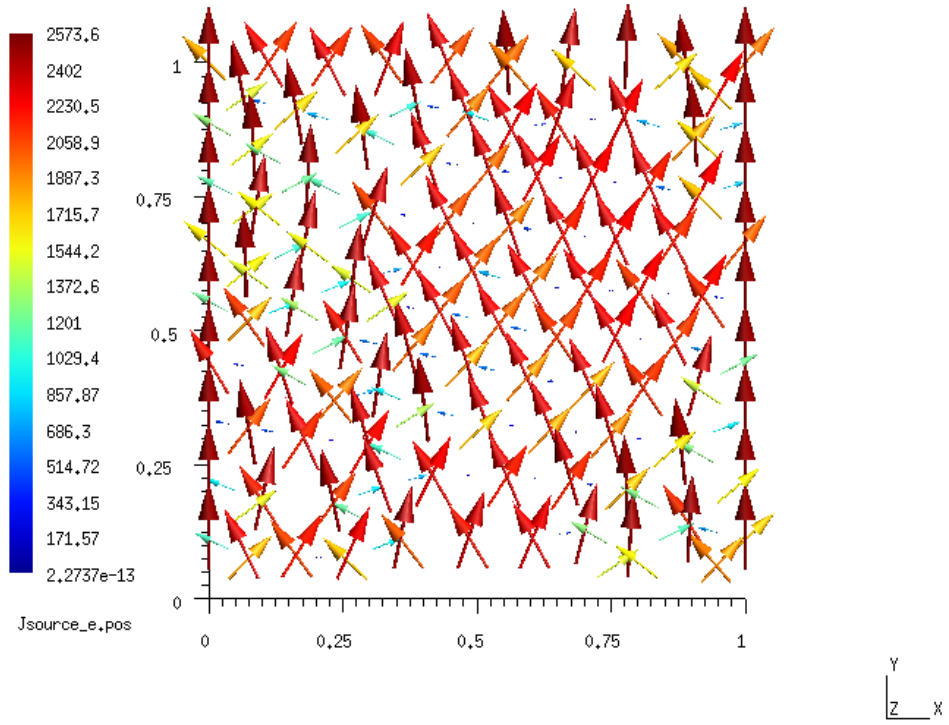

Figure 5.8: GMSH output of $\vec{J}_s$ for the test square displayed as the edge values as they are represented in memory. These values are confined to the edges of the mesh which appear to cross as the magnitude of the edge values are displayed with arrows proportional in size to their value. GMSH lacks the functionality to display and interpolate edge shape functions, so for normal display of functions on the mesh, we translate the vector quantities to nodal values for the three components. This view is useful for determining the how the edge values are represented in memory.

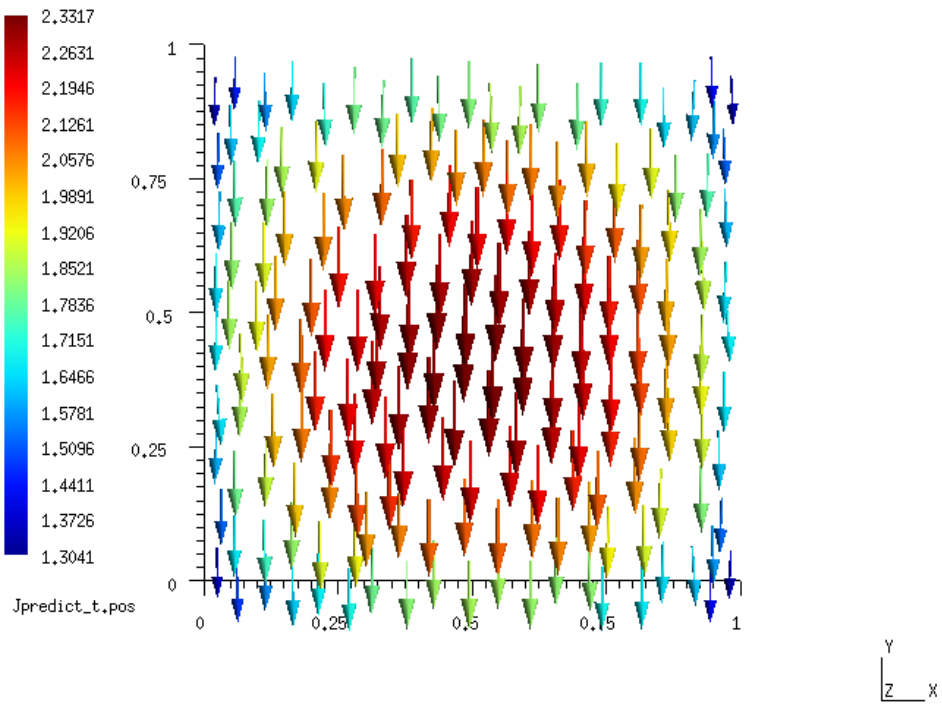

Figure 5.9: `GMSH` output of the prediction current density for the test square case. This is the current density as it would be generated by the magnetic vector potential produced by the source currents. It has no knowledge of the boundaries of the surface.

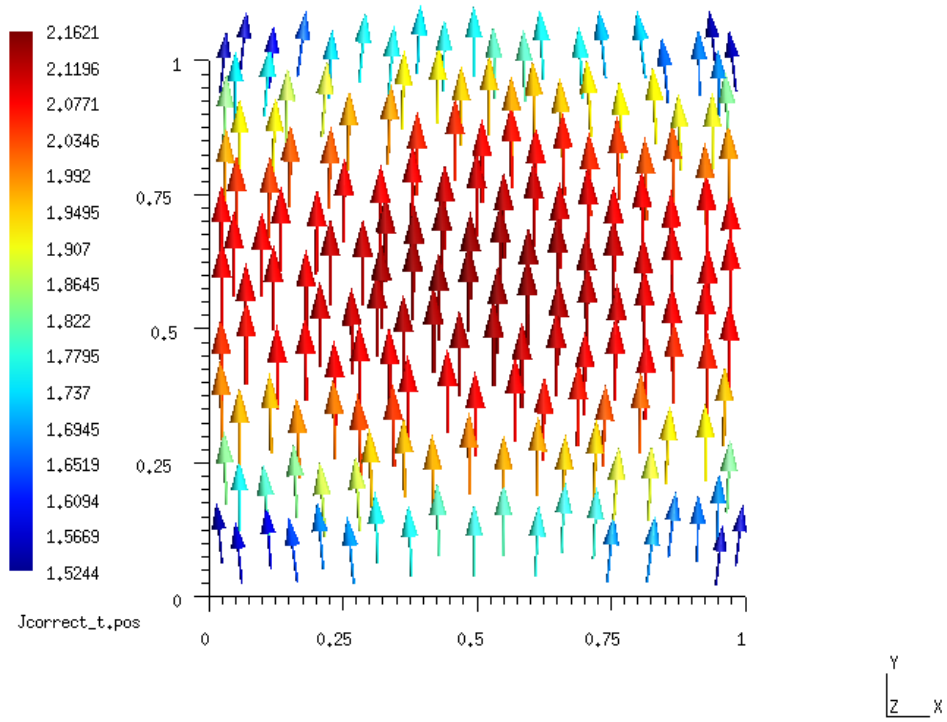

Figure 5.10: GMSH output of the correction current density for the test square case. The correction current density is added to the prediction current density to ensure the boundary condition for the eddy current solution as zero perpendicular current at the surfaces.

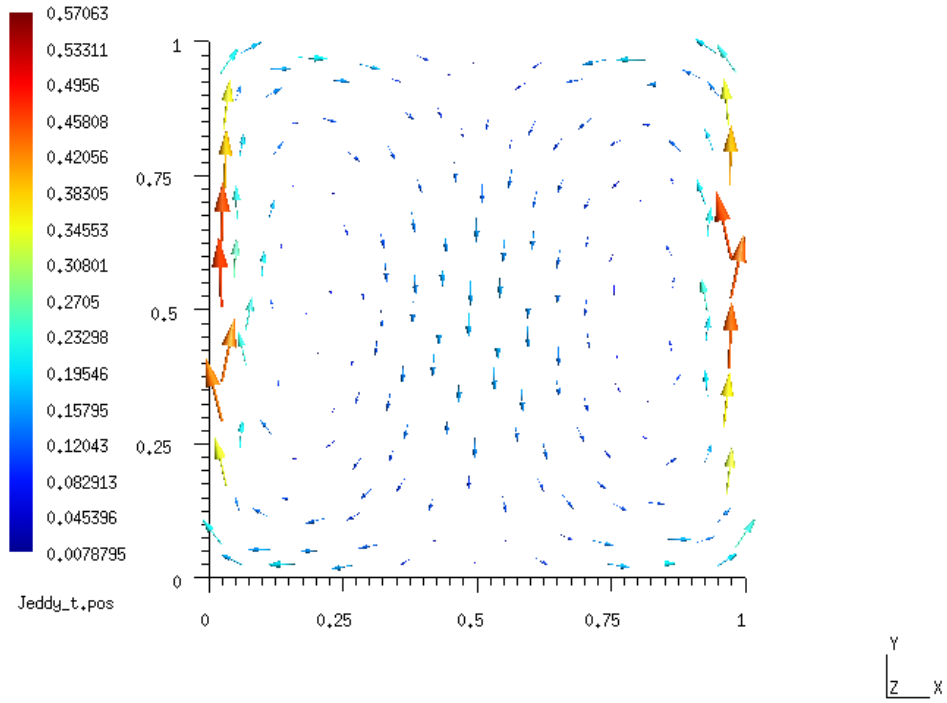

Figure 5.11: GMSH output of the eddy current for the test square. The addition of the prediction and correction current densities produce the eddy current density. The boundary conditions of zero current perpendicular to the surface is met.

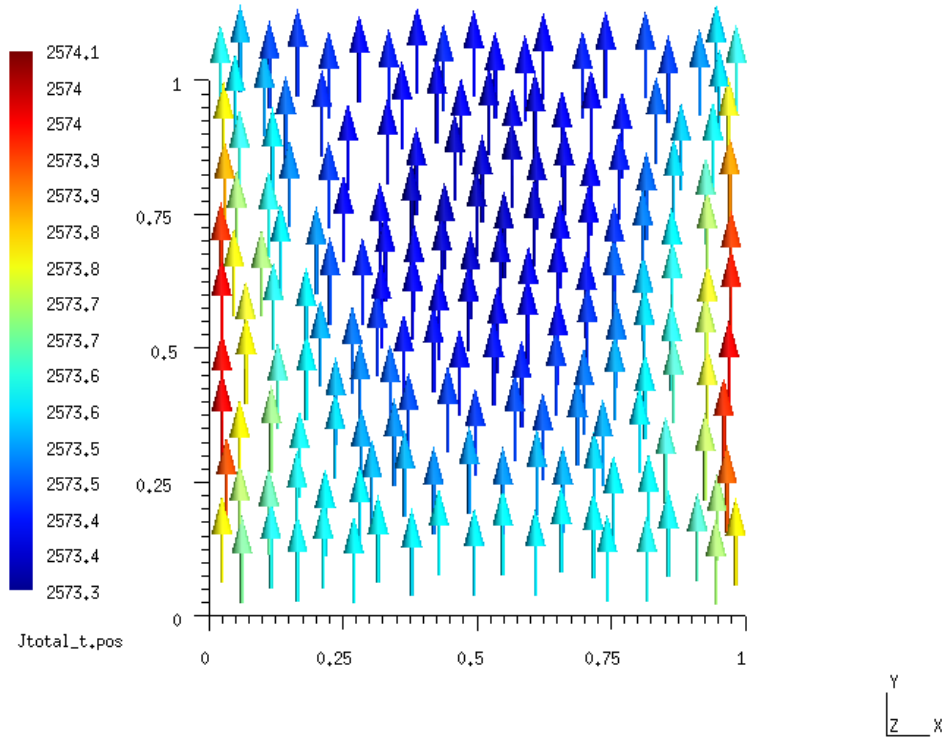

Figure 5.12: GMSH output of the total current density in the test square case. The total current density in this test case exhibits the characteristics of skin effect.

5.2 Test cases of one inch strip lines of variable lengths

To evaluate the performance of the TRANSEDDY code for accuracy and precision, we will compare its transient output to transient output from SPICE using the circuit models of the strip lines as derived in Chapter 2 with the partial inductance method.

5.2.1 Mesh size requirements

In Chapter 2 we implemented a partial inductance method of solving for the skin effect in flat conductors by segmenting the flat conductor into many identically sized filaments. A convergence test was performed, the results of which verified that the the number of filament segments required to represent the total line depended on the frequency of operation of the line. In general, the width of the filaments needed to be less than the skin depth. If the skin depth were shallower than a single filament, the numerical solution was invalid. SPICE will perform its assigned task of balancing the currents in the individual filaments, and adjust the magnitude and phase in each filament until a lowest energy state is reached. If the skin depth is less than one filament, then too much current will be represented in the edge element, requiring the inner filaments to have greater phase errors. With the knowledge of the skin depth for a given frequency, we set the number of filaments such that their the skin depth is greater than several filaments. For the finite element implementation of the transient current solver, we must choose the size of our elements with similar requirements. The size of the elements should be smaller than skin depth.

Figures 5.13 through 5.15 are pictures of the mesh for the top plane of a strip line. The line is one inch wide, and four inches in length. The vertical spacing between the top and return path is the same as for the physical boards used in simulations and testing in Chapter 2, which is approximately 1.75 mm. The mesh density is increased gradually between the different test cases from four elements across the lateral center line to 100 elements. Test meshes are presented for the cases of 4 , 10, 20, 30, 40 ,50 ,60 ,and 100 elements. A half cycle of a 100 kHz sinewave was set as the input signal to the line and a sample of 400 points

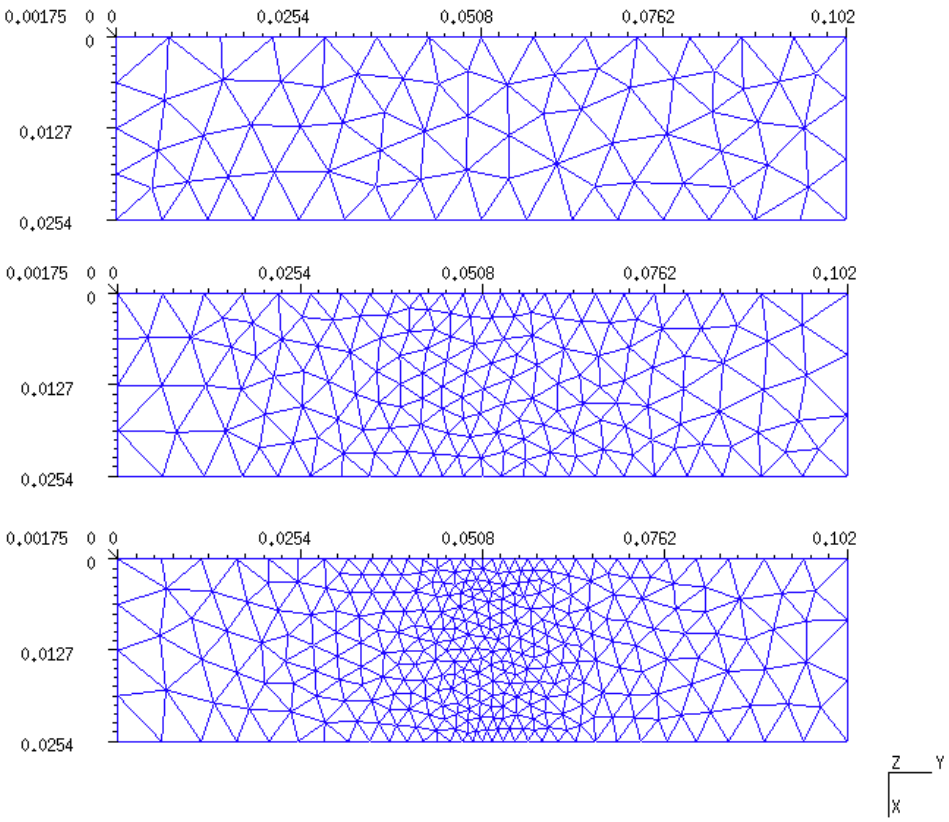

Figure 5.13: Meshes for the test strip line. The line is 4 inches in length, 1 inch wide, and has a return path. A variable mesh has been implemented. The sample tolerances for node spacing across the center of the line for these strip lines are the all referenced to the width. They are the width divided by 4, 10 , and 20 respectively.

was taken at the peak current, across the center line of the top plane.

The total current density for the strip line with 100 elements across the center line at the peak current is in Figure 5.16. A detailed view of the center section of the line is in Figure 5.17.

A comparison of the current distribution between these different mesh sizes was made and can be found in Figure 5.18. In it we can clearly see how the finite element approximation linearly interpolates the function between element edges. This is most evident in the line for the 4 element cross section. By the time the number of segments increases to greater than

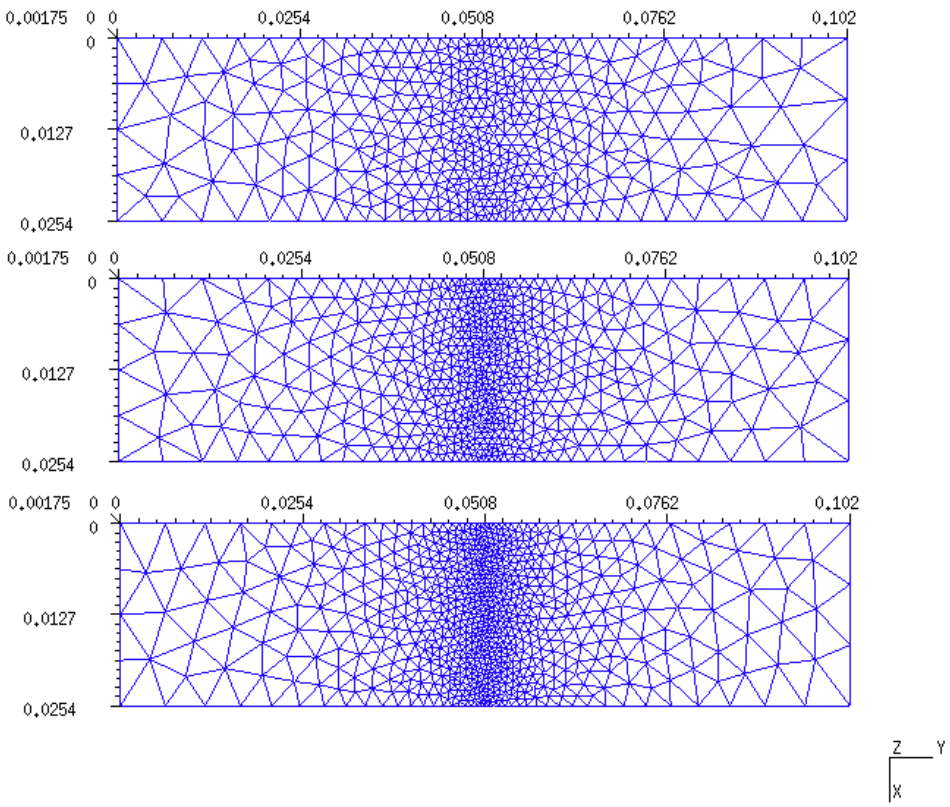

Figure 5.14: Meshes for the test strip line. The line is 4 inches in length, 1 inch wide, and has a return path. A variable mesh has been implemented. The sample tolerances for node spacing across the center of the line for these strip lines are the all referenced to the width. They are the width divided by 30, 40 , and 50 respectively.

50 elements, the curves are almost on top of each other and we see little improvement with increasing elements. For a simulation of a higher frequency source input, the mesh density would need to be increased.

Similar performance was observed with the `SPICE` simulations in Chapter 2 , where the number of filaments used limited the frequency range at which the simulations were valid. Figures 5.19 through 5.21 show the effect of increasing the number of filaments to the current density distributions. By plotting the current density distributions we can see the effect of a coarse filament refinement at higher frequencies where the skin depth is less

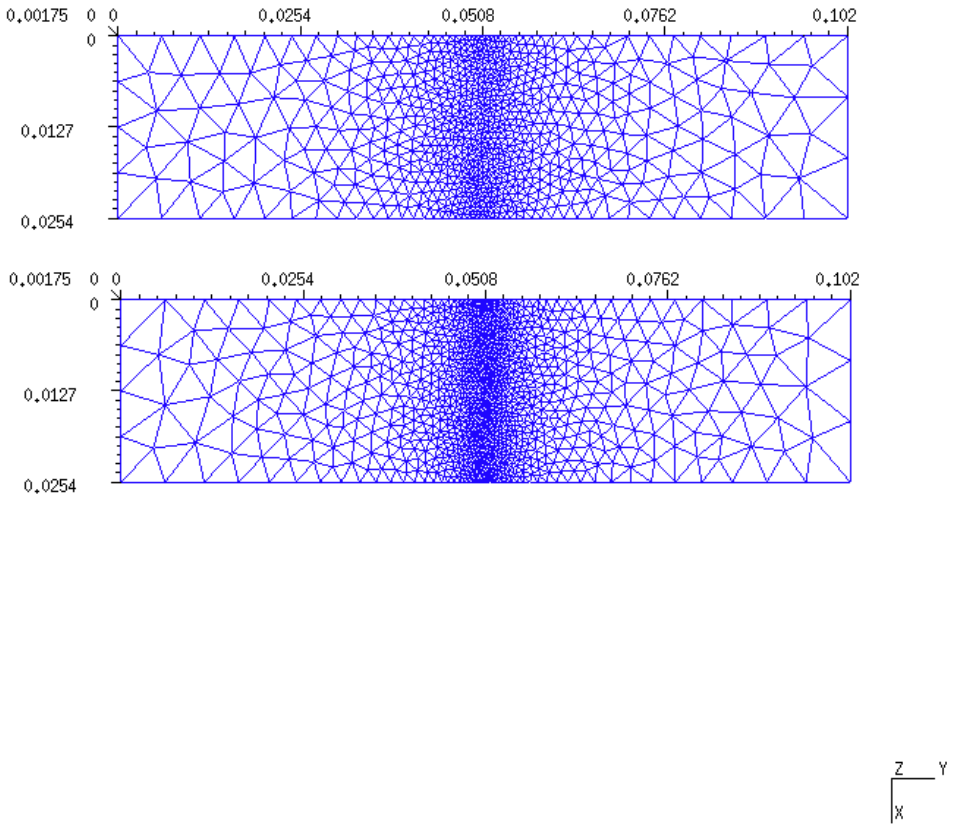

Figure 5.15: Meshes for the test strip line. The line is 4 inches in length, 1 inch wide, and has a return path. A variable mesh has been implemented. The sample tolerances for node spacing across the center of the line for these strip lines are the all referenced to the width. They are the width divided by 60, and 100 respectively.

than that of a single filament width. In Figure 5.19 we see the different distributions in the one inch strip line at 10 kHz. The minor differences in the magnitudes, visible as a vertical shift in the distributions is due to machine precision is solving for the circuit theory model parameters for each filament. The curves are smooth, signifying that at 10 kHz, the skin depth is greater than the width of a single filament. This behavior is also visible in Figure 5.20. The skin effect is more pronounced, but the distributions are still smooth curves. In Figure 5.21 we can see the change in the smooth current density distribution when the skin depth is less than one filament width. the plots for the 15 filament and 20 filament show a

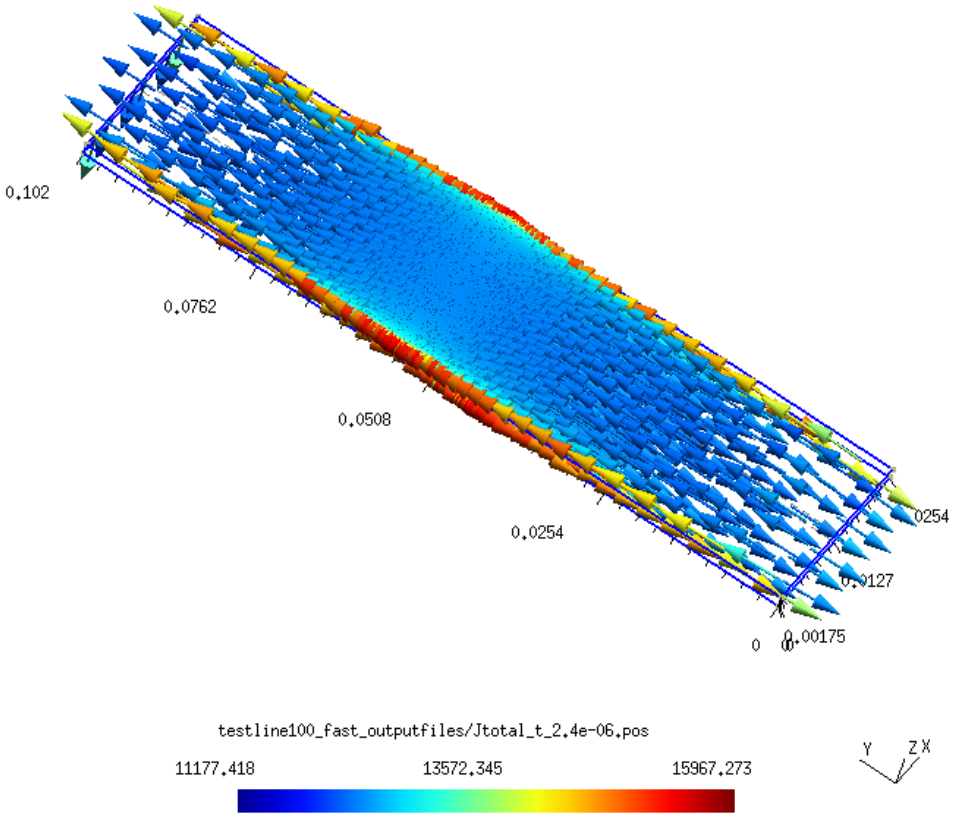

Figure 5.16: GMSH output display for the test strip line with 100 elements across the center line. Total current density taken at peak current.

bump at the edges, and an elevated current density in the center of the distribution. The more dense filament models continue to show the smooth distribution we expect to see when viewing the skin effect in a strip line. This bump in the distributions is caused by SPICE as it numerically solves for the current in each filament.

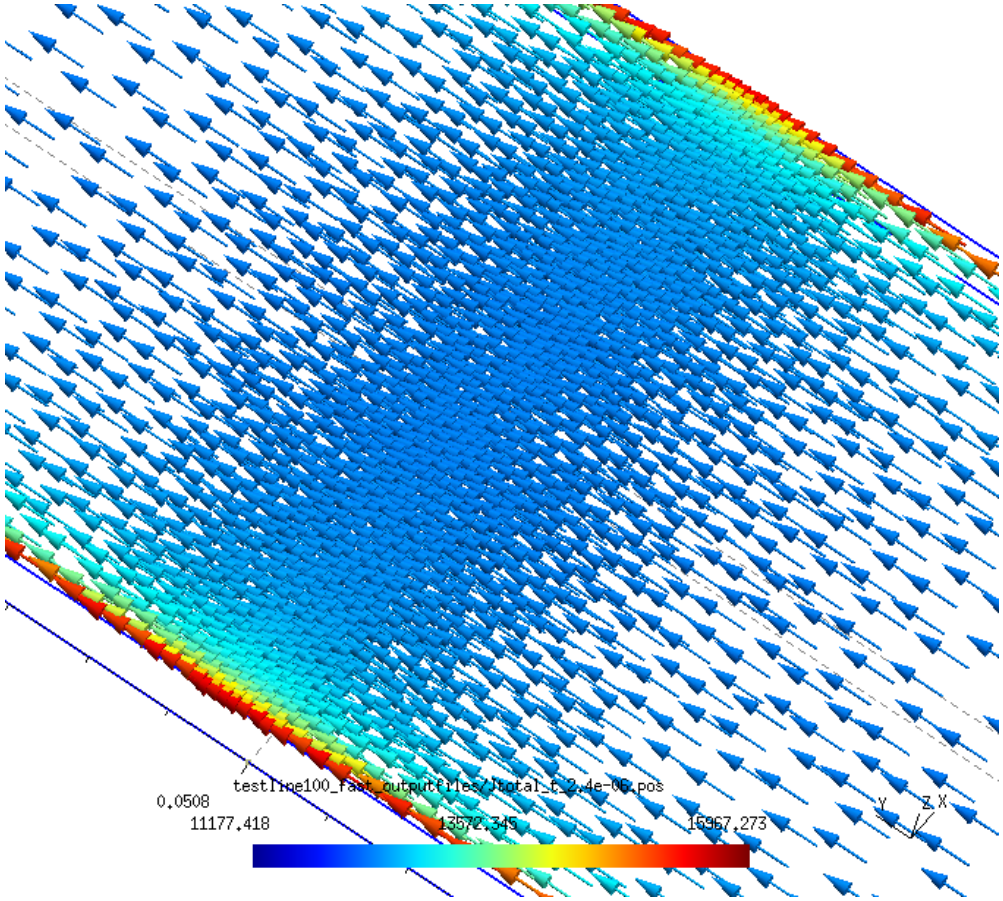

Figure 5.17: Detail view of GMSH output display for the test strip line with 100 elements across the center line. Total current density taken at peak current.

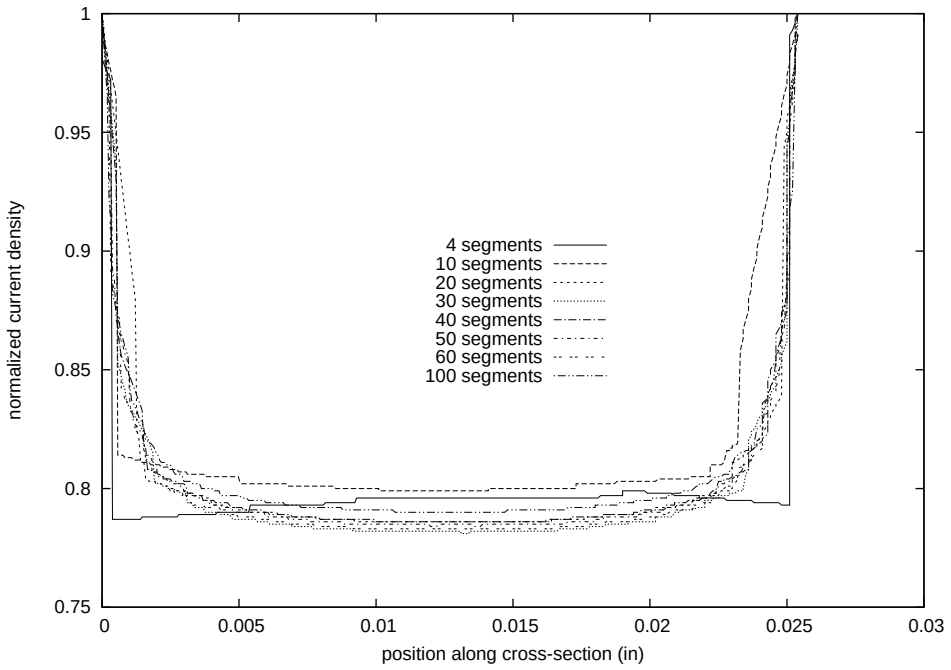

Figure 5.18: Plots of the current density across the centerline of the top plane of the test stripline for different mesh densities. The strip lines were simulating the current flow for a 100 kHz sinewave input. Data was taken a the peak value of the input current. Linear interpolation of data between element edges is apparent in the 4 element line. Current Density distributions converge when the the number of elements across the strip is greater than 50. For higher frequency simulations, greater density in the mesh is required for improvements in distribution accuracy and precision.

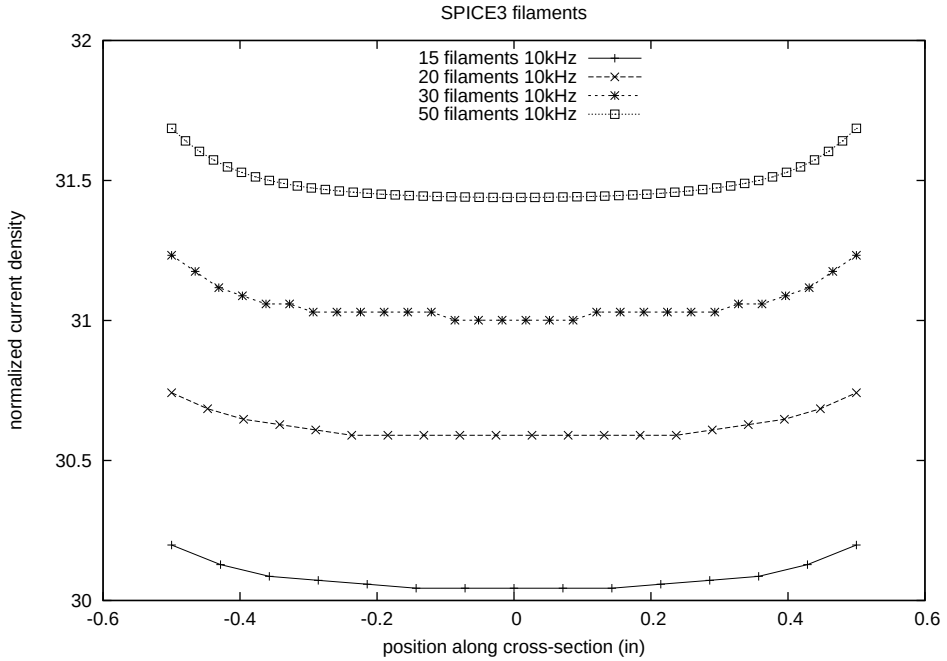

Figure 5.19: Current density distributions for a SPICE simulation of a 1-inch strip line. The strip line was divided into different numbers of segments and an AC simulation was conducted. Presented plots are for a 10 kHz input. Very little skin effect is present. Slight differences in the magnitudes are due to machine precision in solving each individual filaments circuit theory model parameters. This results in a vertical shift in the current density distribution between the different runs.

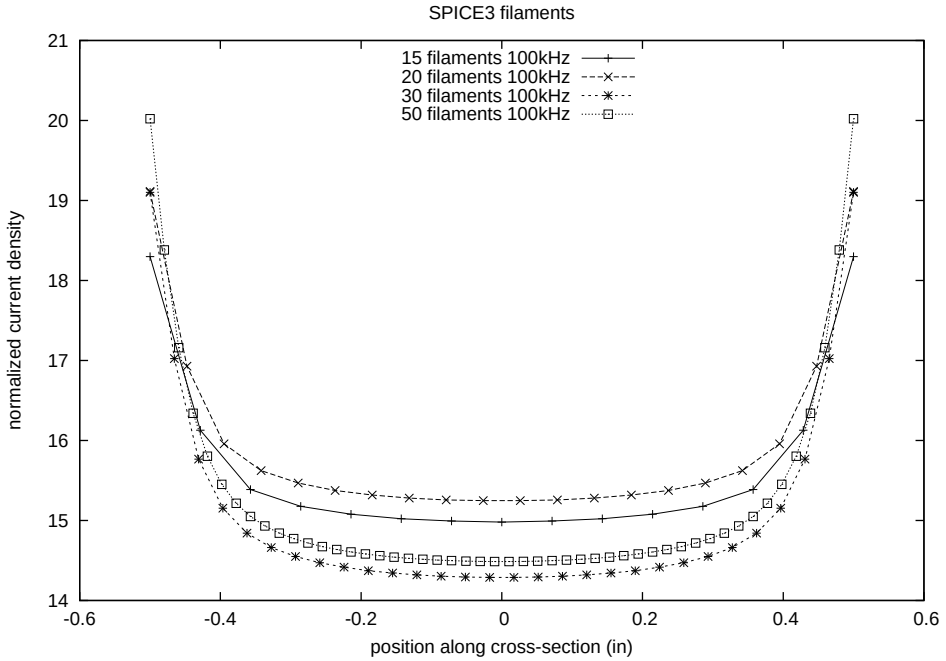

Figure 5.20: Current density distributions for a SPICE simulation of a 1-inch strip line. The strip line was divided into different numbers of segments and an AC simulation was conducted. Presented plots are for a 100 kHz input. Slight differences in the magnitudes are due to machine precision in solving each individual filaments circuit theory model parameters. This results in a vertical shift in the current density distribution between the different runs.

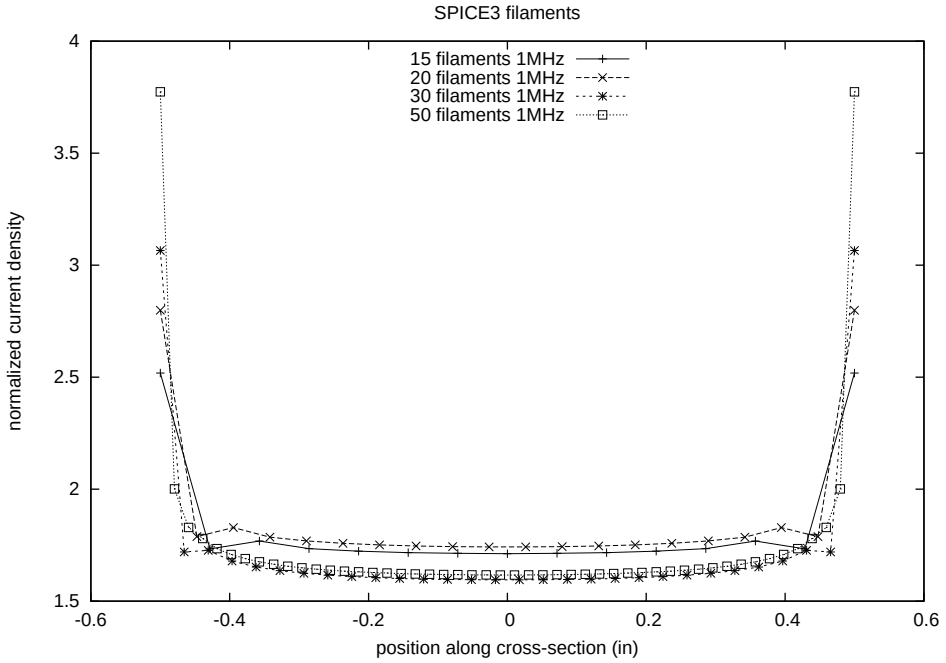

Figure 5.21: Current density distributions for a SPICE simulation of a 1-inch strip line. The strip line was divided into different numbers of segments and an AC simulation was conducted. Presented plots are for a 1 MHz input. From these plots, the effect of the skin depth being less than the width of a filament can be observed. Slight differences in the magnitudes are due to machine precision in solving each individual filaments circuit theory model parameters. This results in a vertical shift in the current density distribution between the different runs. The plots for 15 and 20 filaments show the effect of the skin depth being less than the width of a single filament, resulting in the non physical distribution of the current density. Plots for 30 and 50 filaments show the smooth curve we expect to see for the skin effect. This signifies that the skin depth is greater than the width of a single filament.

A limitation on TRANSEDDY is that it required a large amount of memory to run. This keeps the size of mesh files we are able to simulate to less than approximately 12,000 edges. Through testing, this number of edges will result in almost 8GB of RAM being used. In order to increase the mesh density, and still be able to be executed on available machines, further testing was performed on a single plane rather than a full strip line, and the length was kept short. With use of the variable meshing available in GMSH we constructed a dense mesh for a one inch wide, two inch long plate. A spice model was made of this geometry using the partial inductance method as described in Chapter 2.

The implementation of a single strip versus a full line with return path means that we can simulate at a lower frequency and still observe the skin effect. Without the return path, we are able to use all the available memory size to mesh a single plane, thus increasing our mesh density on the remaining plane, and increasing spatial resolution of the result. A sample output showing the achieved density and the relatively smooth density distribution across the center line can be found in figures 5.23 and 5.24. At this frequency and mesh size, the distribution is a smooth function.

Discrepancies between the SPICE filament model and the FEM model can be seen from the plot of the peak current on the test strip. The filament model guarantees that there is no lateral current flow, that the current can only be contained in one of the long filaments. The only connection is at the ends of the strip, where they are all equally connected to the source node. This model is appropriate for cases where the filaments are very long. Indeed, from Chapter 2 we see that the formulas to derive the partial inductances were originally derived by assuming all currents in the z direction only.

The finite element approximation is superpositioning an eddy current on top of the source current. By Maxwell's equations this is a valid formulation, but we see that the eddy current is completely contained within the conductor. Magnetic effects, or an induced back emf will not be seen by the source. A physical representation of this method is that a source current is being fed to the test conductor, and the resultant current density distribution is

due to that current. Therefore, what we have with the initial solution of the source voltage by the Laplace equation, is actually solving for the source current that is flowing through the sample. For comparison with SPICE , we will use a current source to drive the signals.

To solve for the voltage required to give the desired current in the simulation, one simply needs to back calculate using Ohm's Law. We can calculate the total resistance of the conductor from it's spatial parameters, and the conductivity. Then we choose the voltage appropriately to give the total source current desired.

For data processing and comparison to SPICE simulations, the current distribution was sampled across the center of the strip with 400 samples taken. At this distance from the ends, the current distribution is almost constant in the y direction. Effects from the ends have dissipated and we are seeing all transverse currents. This is as close to the SPICE approximation of filaments with only y directed currents as we have available.

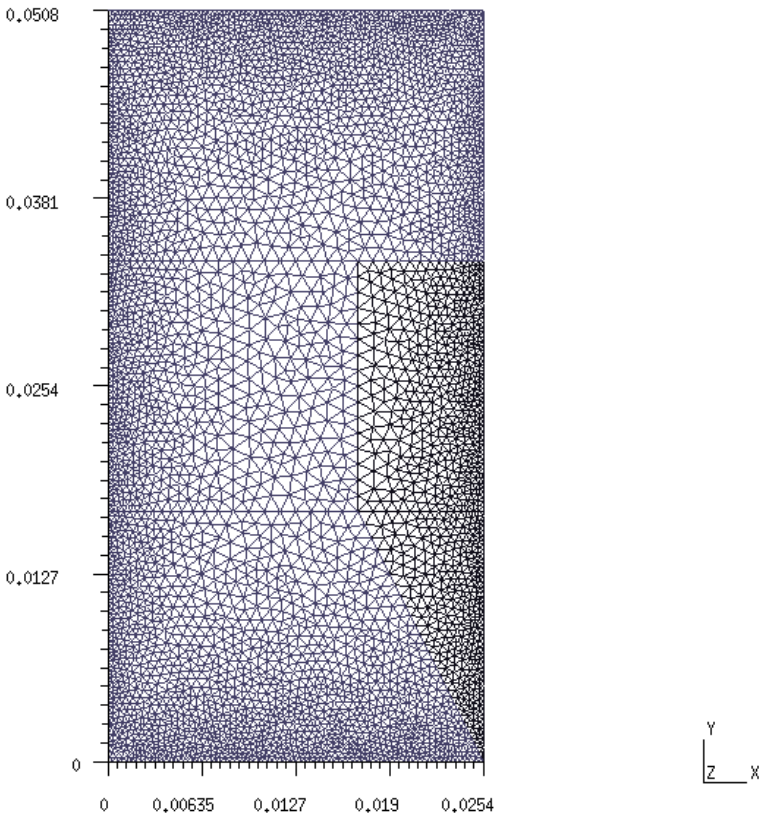

Figure 5.22: Mesh for the test strip. The line is 2 inches in length, 1 inch wide, and has no return path. A variable mesh has been implemented. This mesh has been sized to give a fine spacing along the corners and edges, where the expected skin effects will be large, and coarser near the center, where the current distribution is expected to be regular and well behaved.

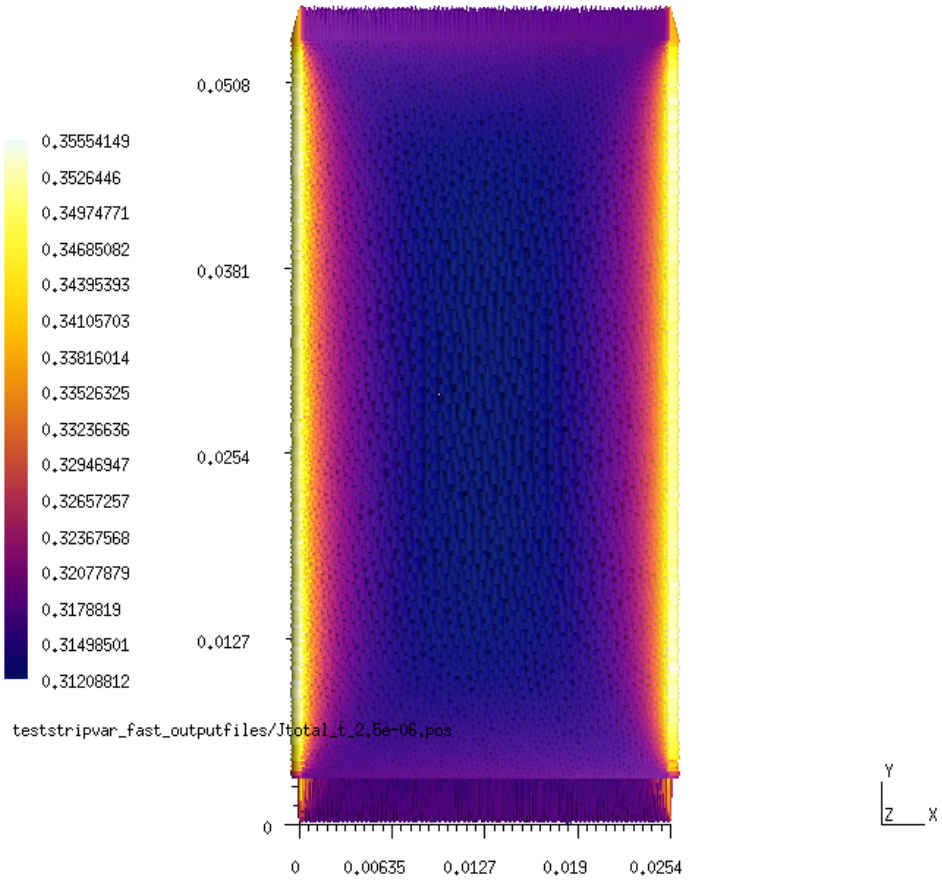

Figure 5.23: GMSH output for the simulation of a 10 kHz input current to the test strip with variable mesh density

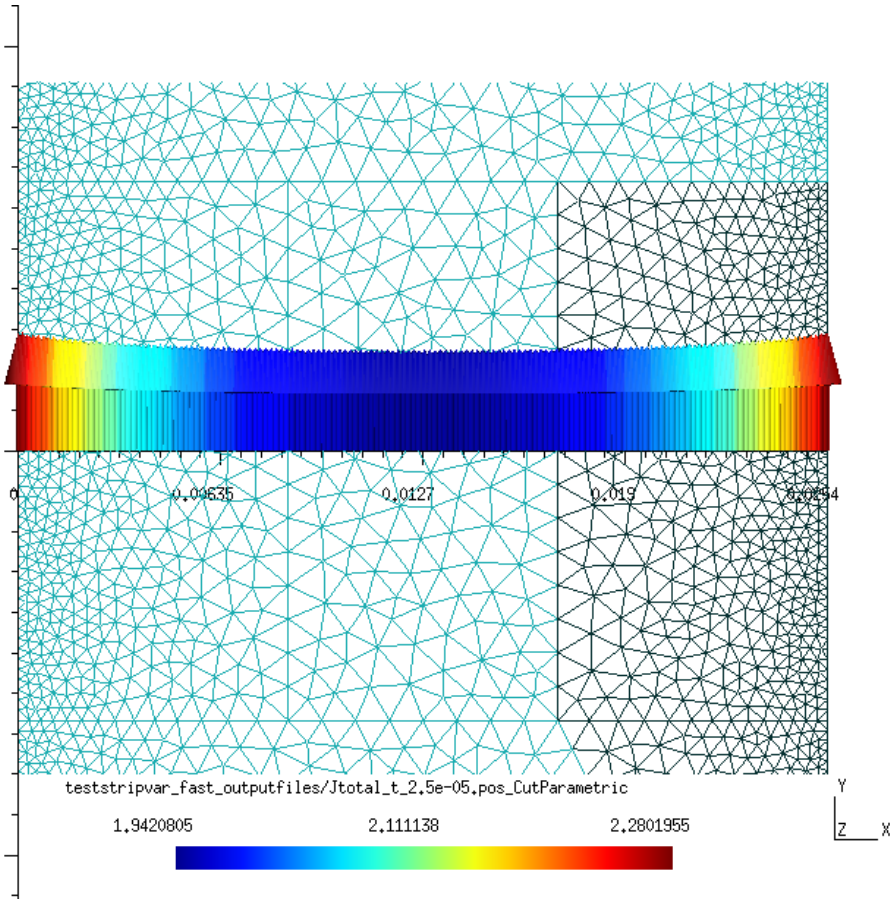

Figure 5.24: GMSH output for the centerline sampled current density in the test strip. 400 samples were extracted across the center of the strip. The mesh on which the data was extracted is presented in the background.

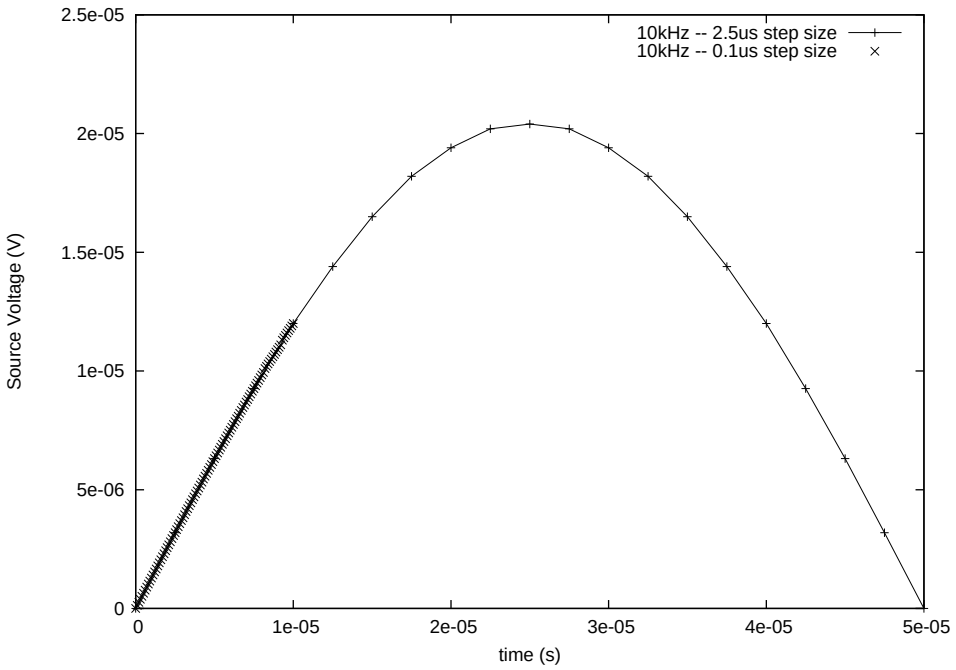

Figure 5.25: Sample time-voltage data files for a 10 kHz signal. Both a full half period with 20 samples and 100 samples taken within $\frac{1}{10}^{th}$ of the period are presented.

5.2.2 Time step size constraints

To evaluate the effect of the time step size on performance a series of simulations were performed. First a baseline was run for a 10 kHz input signal with a stepsize of 2.5μs for 50μs. A second simulation was run for the same 10 kHz signal, but with a step size of 100 ns for 10μs. The example source signals are represented in Figure 5.25.

The simulation results for the test strip are presented in Figure 5.26. Four hundred discrete samples were taken across the centerline of the strip at each time step. The first ten samples of the 100ns step size simulation are plotted, after which every tenth sample is displayed. The graphs are almost identical with deviations well within the margin of error for the machine.

The code was written utilizing the fractional step method. This method is used to

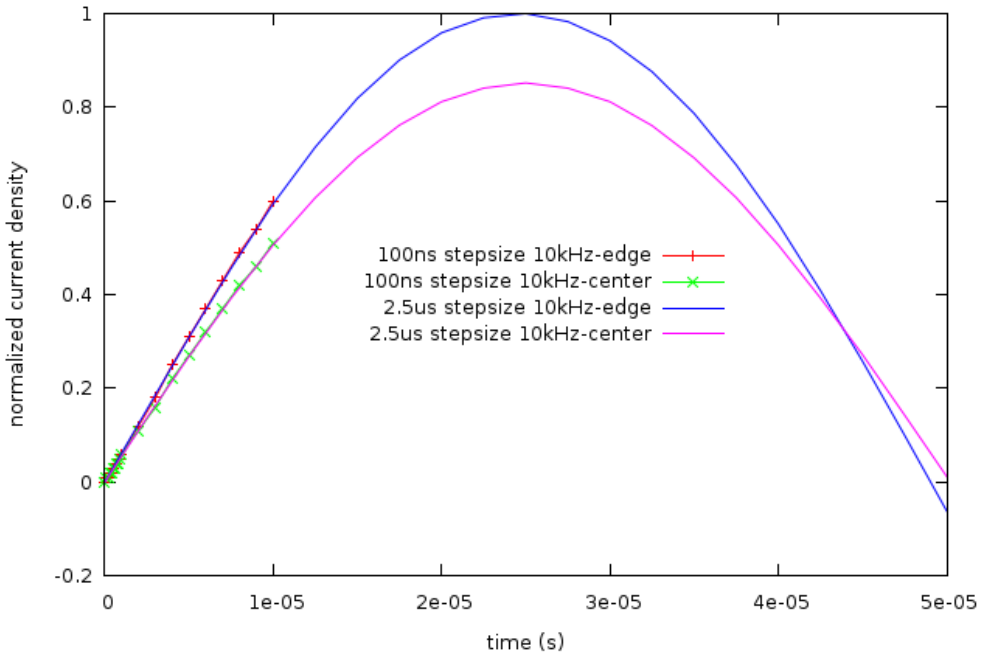

Figure 5.26: Comparison plot of `TRANSEDDY` simulations with different time step sizes. Time step sizes are $100ns$ and $2.5\mu s$. The magnitudes of the current density at the edge and center of the strip are shown. The small time step size graph shows the first 10 samples, followed by every tenth sample there after.

guarantee stability in integrations. For these simulations we used a fractional step parameter of 1, which means we were implementing implicit integration. To test for stability, the 10 kHz signal was also integrated with the fractional step parameter set to 0.5, which is the Crank-Nicholson scheme[45]. A graph comparing the two simulations is in Figure 5.27. From these simulations we see little difference between the Crank-Nicholson scheme and implicit integration. Minor differences in the magnitudes are within the limits of numerical error of the machine.

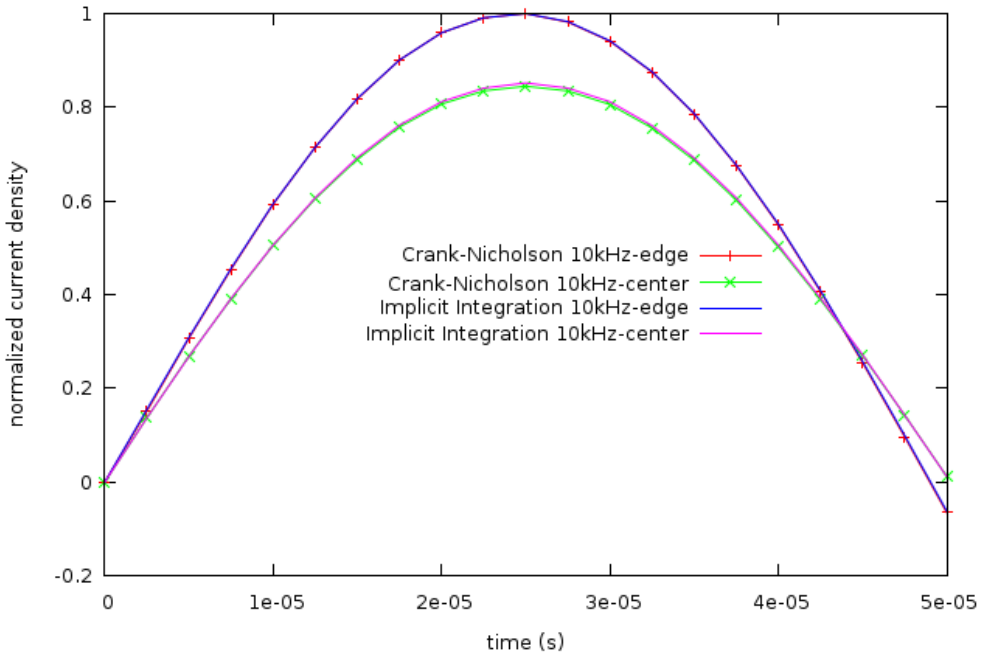

Figure 5.27: Comparison plot of `TRANSEDDY` simulations with different fractional step schemes. Plots for the magnitude of the current density at the edge and at the center of the strip are presented for the Crank-Nicholson Scheme and Implicit Integration.

5.2.3 Comparison of transient responses

Once we have established that TRANSEDDY is performing the time stepping correctly with regard to spacial and temporal step sizes, we need to compare the results of a transient simulation to ones calculated with SPICE . Figure 5.28 shows the transient response to a 10 kHz input current pulse for a single conducting strip two inches in length, one inch wide and approximately 0.0017 inches thick. The magnitude of the response is close to agreement between the SPICE filament model and the TRANSEDDY finite element implementation. Obvious disagreements are in the phase of the current density between the center and the edge of the strip. This is due to the limitations in the models. The SPICE filament model is accurate

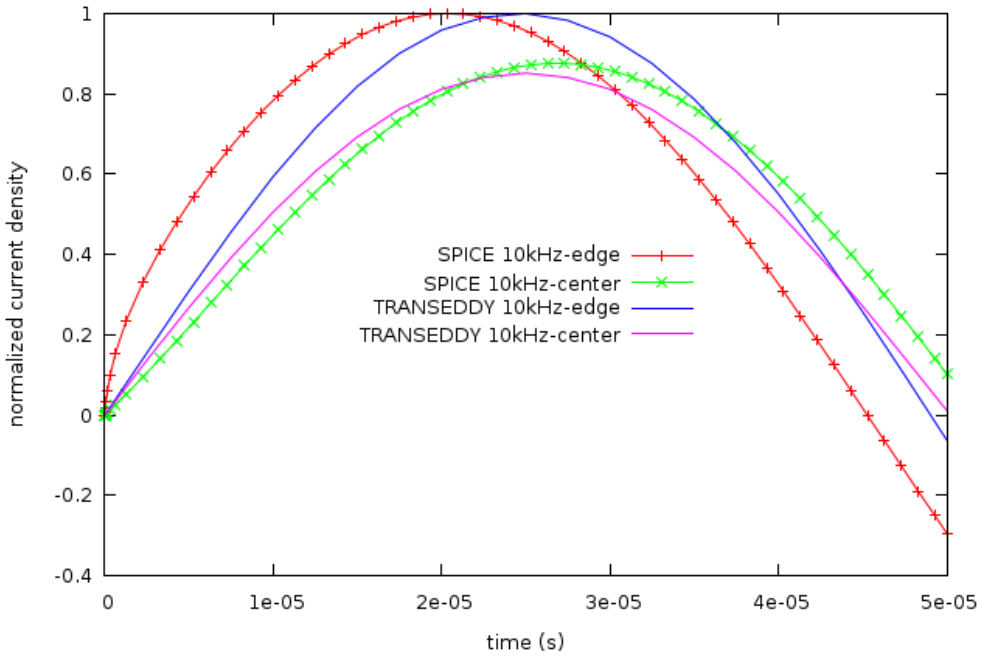

Figure 5.28: Comparison of AC conduction for the test strip line for SPICE and TRANSEDDY simulations

for very long conductors where the current will all be directed in the forward direction and parallel. The SPICE model forces the all the currents parallel the entire length of the strip line while the finite element implementation allows current flow in any direction along the surface. There is a phase delay between the center and the edge of the TRANSEDDY model;

it is not as pronounced at 10 kHz. At higher frequency we see the phase shift, but it is still not to the magnitude that SPICE produces. From this comparison we must conclude that the SPICE model's limitations are producing a phase shift that is not physical in nature, but due to the numerical constraints of its implementation. The magnitude of the responses are in close relative agreement.

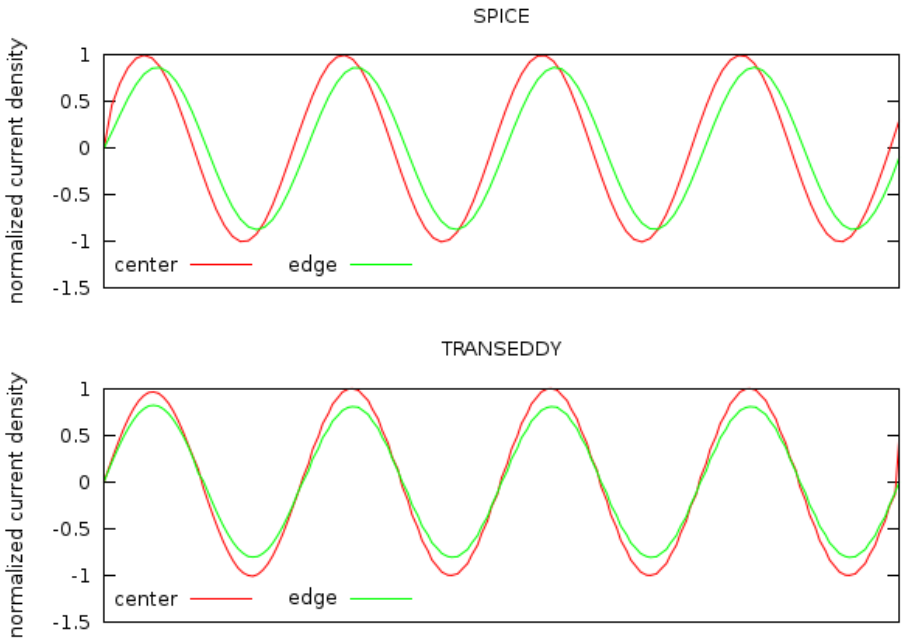

Figure 5.29: Comparison of transient conduction for the test strip line for SPICE and TRANSEDDY simulations over 4 periods of an input sinewave. AC steady state is reached by the first cycle for both simulations. Plots for the center and the edge of the strip line are presented for each simulation.

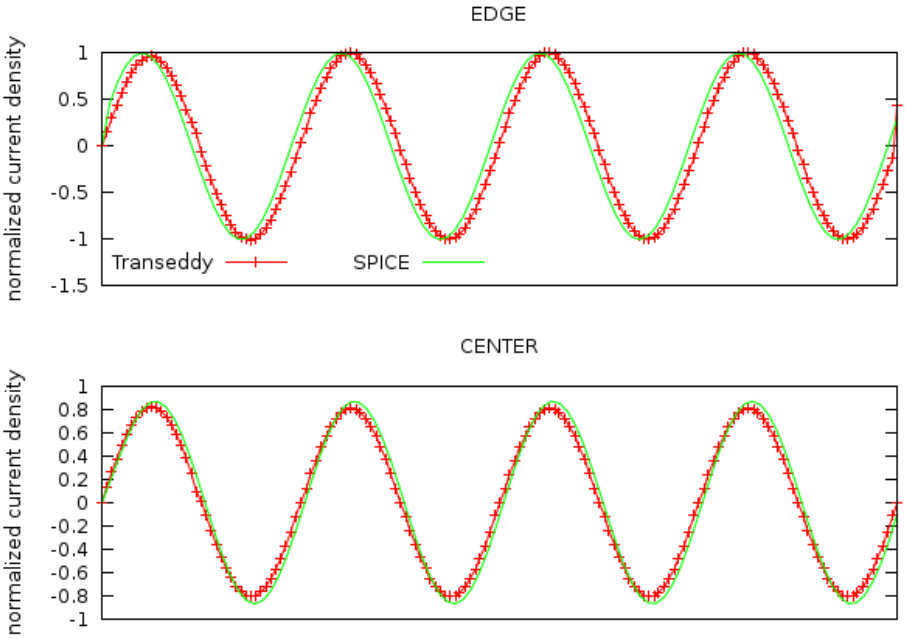

Figure 5.30: Comparison of transient conduction for the test strip line for `SPICE` and `TRANSEDDY` simulations over 4 periods of an input sinewave. AC steady state is reached by the first cycle for both simulations. Plots for the center and the edge of the strip line are presented with comparison between individual selections for `SPICE` and `TRANSEDDY` .

Chapter 6

Application and future work

This method of calculating transient current density in thin conductors has many applications. It serves as a foundation on which to build for cases where a transient solution is required for cases where the conductive surfaces are either irregular or have the potential to alter material properties. With minor additions, functions could be written to calculate Ohmic heating due to the current density. A model for conductivity of the material can them be read in to adjust the properties due to the thermal rise in temperature. Following this step, several of the coefficient matrices would require recalculation due to changes in the material. The time-stepping algorithm is already suitable for the addition of inhomogeneous conductivity.

6.1 Thin conductors of irregular shape

One example of this application is for thin conductors with voids. If the distribution of the current density is important to the devices' performance, a transient analysis can be used to predict the device behavior.

6.1.1 Conductor with a hole

The first example problem is a simple square conductor with a central square hole. Figures 6.1, 6.2, and 6.3 show the geometry model as implemented with `GMSH` , the meshed surface, and the current density from an instant in time. This example was simulated with a ramp

input signal. It demonstrates the ability of the finite element implementation to operate on structures with holes in the mesh. We can see the symmetric flow of current, and see where the current is crowding.

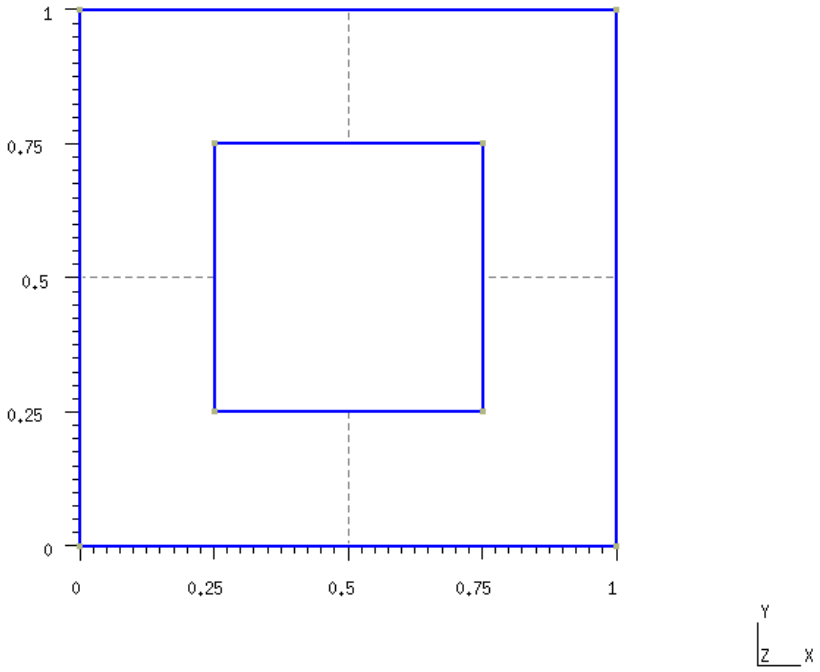

Figure 6.1: Test geometry file for a square conductor with a central square hole.

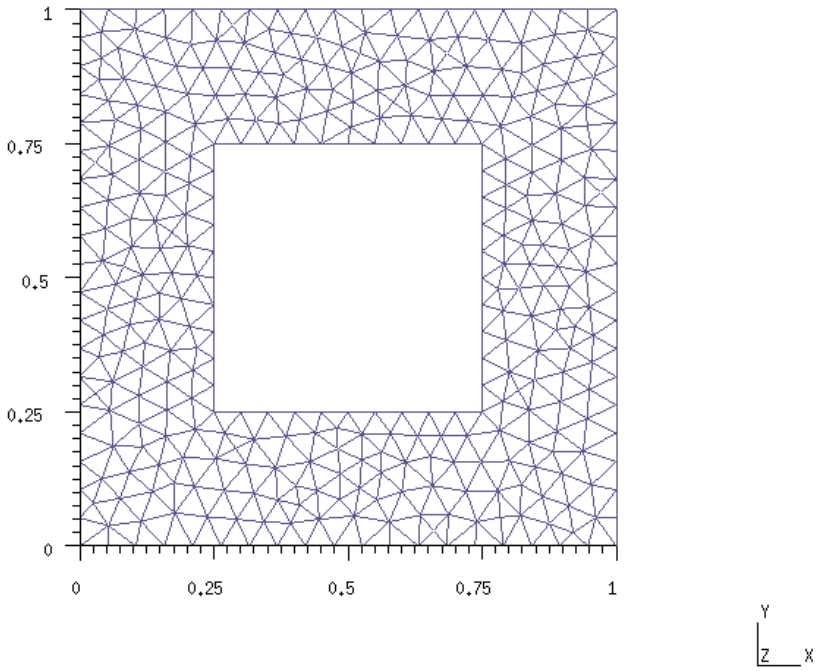

Figure 6.2: Test mesh for a square conductor with a hole.

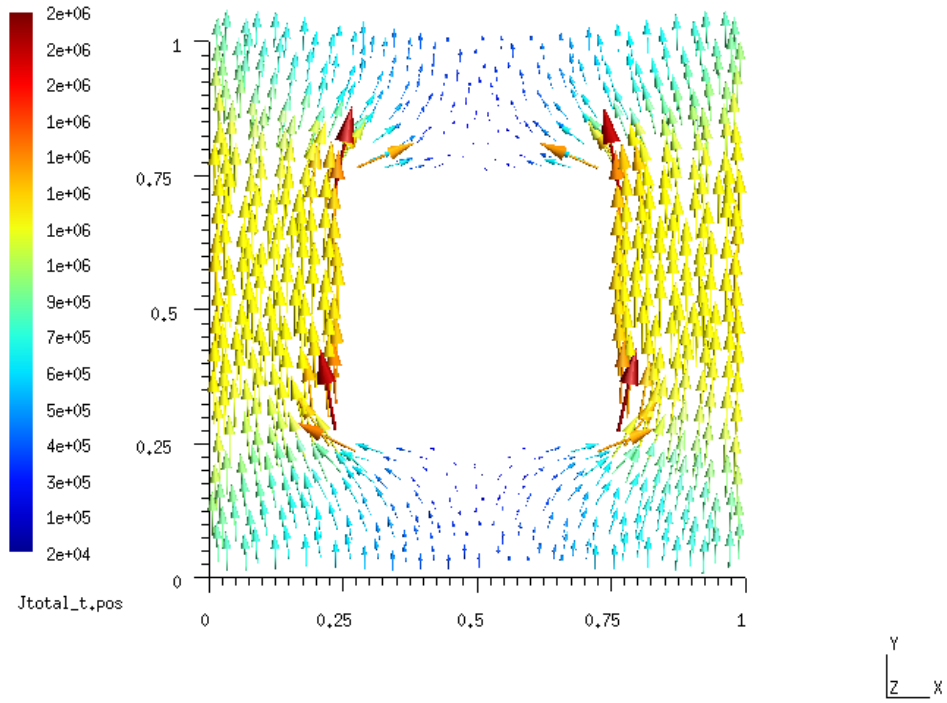

Figure 6.3: Current density distribution produced by `TRANSEDDY` for a test square conductor with a centralized square hole. Conduction is shown for the peak current with a sinusoidal input.

6.1.2 Conductor with constricted entry points

Another example refers to the paper by Ney[5] where he was interested in finding an expression for the effect of 'striction' on strip line impedance. Figure 6.4 shows a simple square conductor, but with narrow entry points for the current. This example was also simulated with a ramp input signal. The effective frequency is too low and the conductor too short to be able to see any skin effect in the current density distribution, but the 'striction' effect is very noticeable.

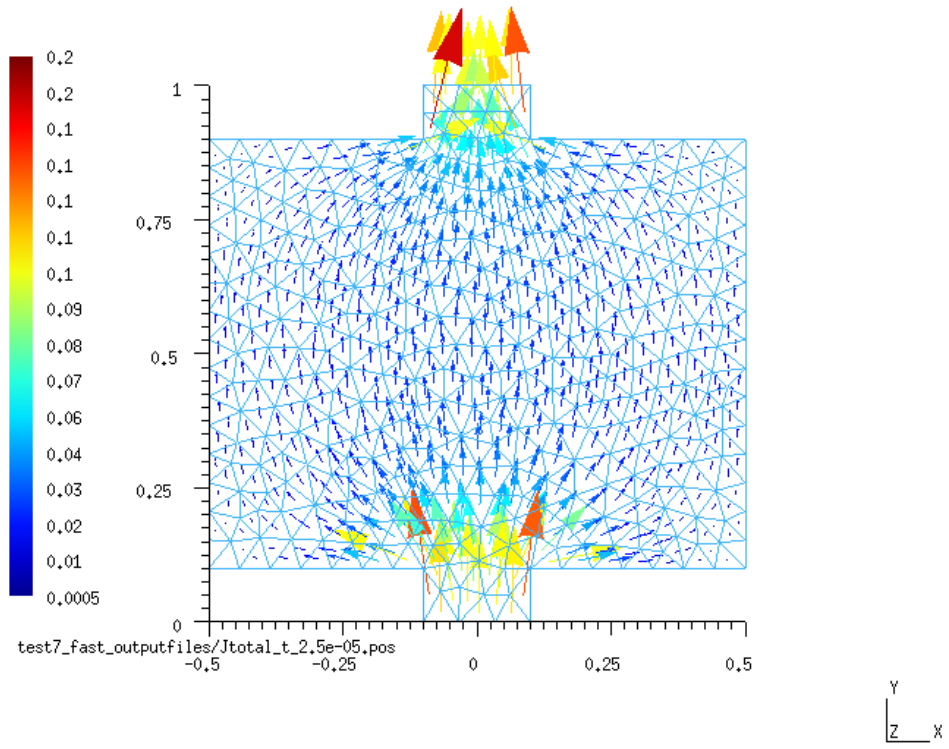

Figure 6.4: The current density is shown for a square test conductor with constricted current entry due to a small tab entry and exit. The 'striction' effect as noted by Ney[5] is visible in the current density distribution.

6.1.3 Conductor with constriction point

The example in figures 6.5 and 6.6 is slightly more complex than the previous two. In it we have a square conductor with a constricted center, and the addition of a return path for the current as a backplane. This example was simulated with a sinusoidal current. This example shows that constricting the current density will produce local "hot spots". This behavior was also predicted and observed by Logan[57] *et.al.*. The current density on the bottom plane also images the path of the upper conductor.

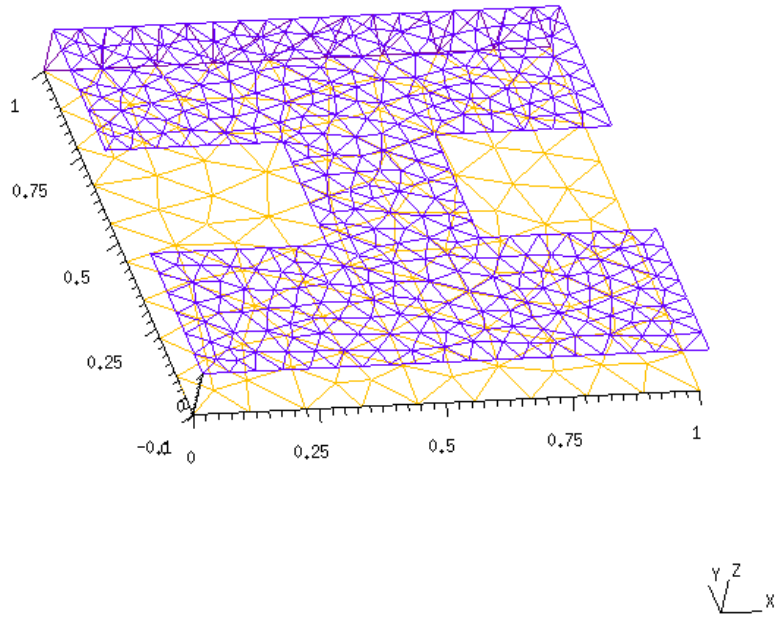

Figure 6.5: Test mesh for a conductor with a constricted center region with a solid return path.

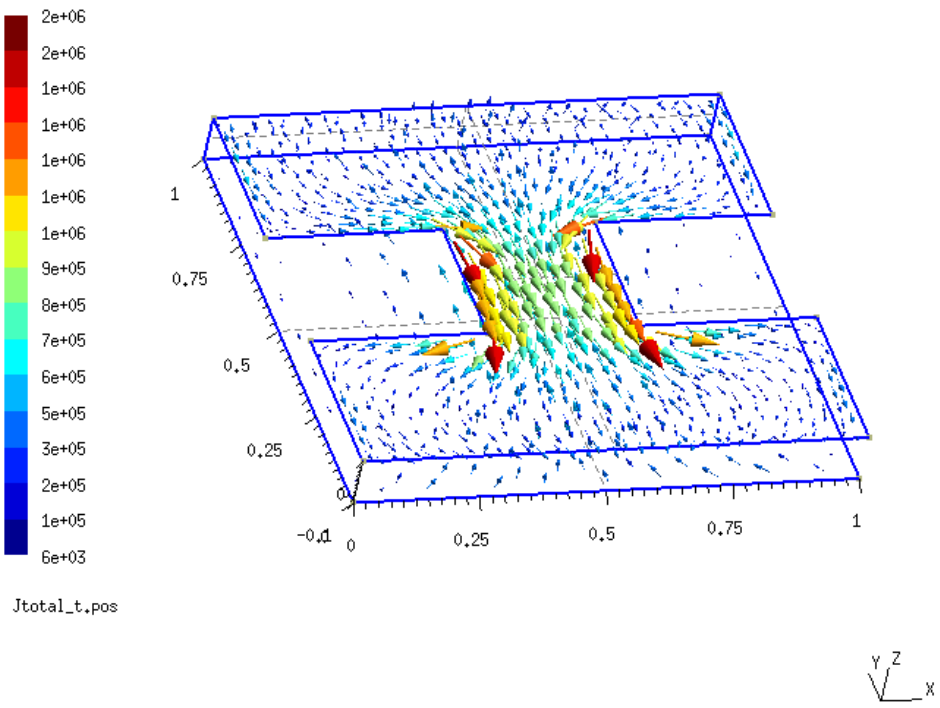

Figure 6.6: The current density distribution for the center constricted test mesh with return path. The current density is crowded on the inside corners of the constricted points as demonstrated by Logan, *et.al*[57]

6.1.4 Conductor with multiple holes and constriction points

Figures 6.7, 6.8, and 6.9 show an example problem consisting of a large conducting plane with multiple holes, and constriction points. The model also includes a backplace. This simulation shows the current distribution due to a sinusoidal pulse in current. The distribution shows the effects of constricting the current as well as skin effect. The center path has a lower current density than the outside paths. The element sizes in the mesh are not sufficiently fine enough to provide much detail in the distribution, however we can see that the current density on the exterior edges are larger than the interior of the conducting paths. The backplane mesh is not sufficiently fine enough to display much detail in the current distribution. By setting the scale for the current density on the back plane to only consider that plane, we can see the effect of the current path on the top plane magnetically coupling to the currents in the back plane. This rough example displays some of the functionality of `TRANSEDDY` for simulating current density distributions in irregular conductors.

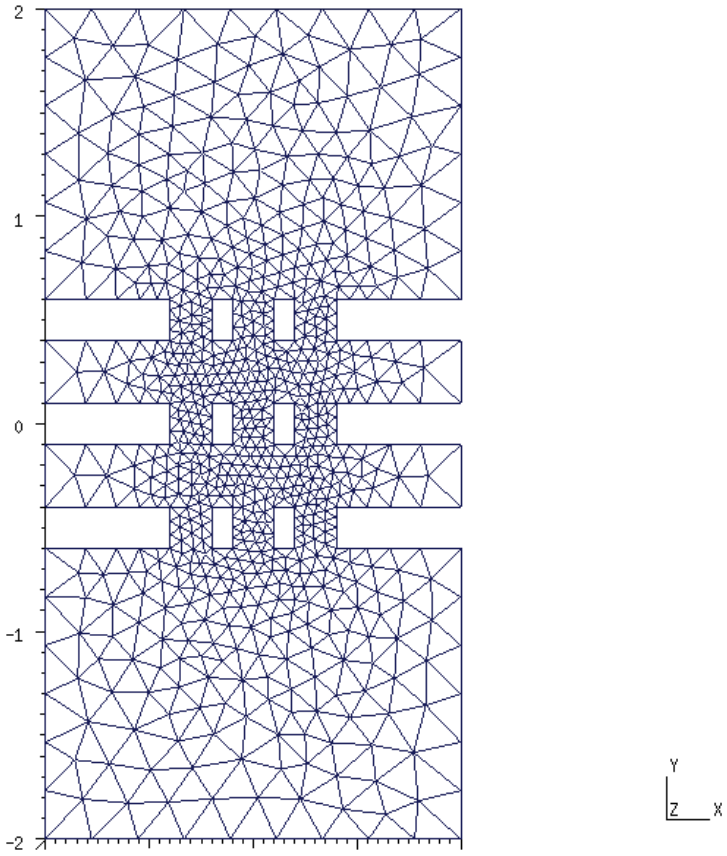

Figure 6.7: Mesh for a conductor with multiple evenly spaced holes with return path.

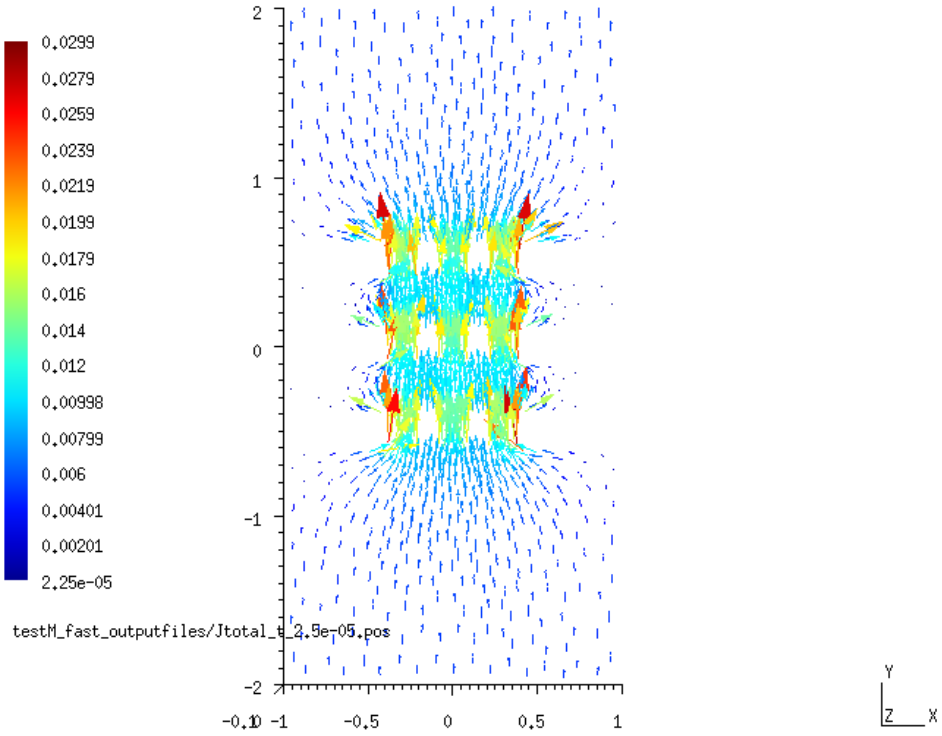

Figure 6.8: Current density distribution for the top plane of the test conductor with multiple holes.

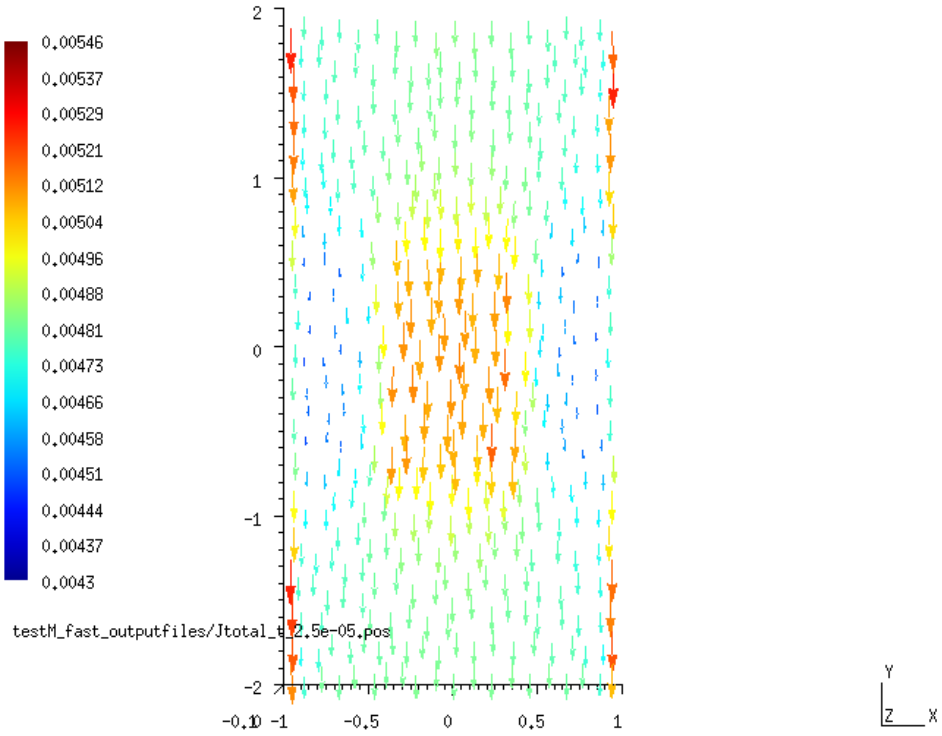

Figure 6.9: Current density distribution for the bottom plane of the test conductor with multiple holes.

6.1.5 Flat, spiral wound inductors

Another application is a spiral inductor[60,61]. An example spiral wound inductor is shown in Figure 6.10. This example was simulated for an overly large inductor subjected to a sinusoidal pulse in current. The simulation shows the current crowding to the interior walls of the inductor and at the interior corners.

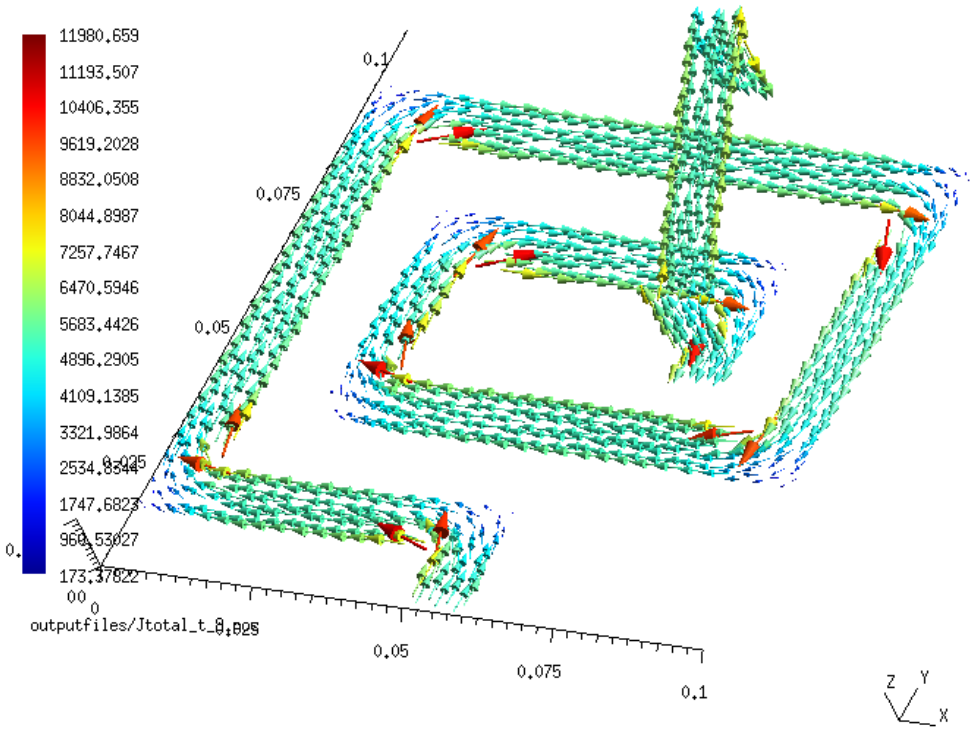

Figure 6.10: Example of the current density distribution in a flat spiral wound inductor.

6.2 Future work: Implementation of conductivity dependencies

To implement conductivity dependencies to `TRANSEDDY` a few minor modifications are required. A model for the conductivity is required. The simplest way to add this functionality is to make the conductor model a separate file to be read in. The file would contain values for conductivity versus whatever parametric the specific model requires. The user then has greater control of the simulation. The addition of a look-up-table would not significantly increase the memory demands of the program.

Once a model has been entered, the dependencies of the conductor would require calculation. Whether it be a thermal model or more complex specific action[58] form, the total current density in each element would have to be calculated. A function to integrate the total current density on each triangular element would have to be added. The integrals over the triangles have already been solved, and a summation of integrals for the edge element basis function and the discrete edge elements is a trivial addition to make.

After calculation of the dependency is done and the conductivity updated from the look-up-table, several of the coefficient matrices would have to be recalculated. The time-stepping loop then continues as usual. The source current density is calculated with the new material properties. The integral formulation for the eddy current density is performed with the updated coefficient matrices.

The conductivity model is limited to the cases where the material maintains it's spatial parameters. Thermal expansion is not covered with this formulation, nor are moving conductors.

6.3 Future work: User-interface

Implementation of `TRANSEDDY` was done primarily as a proof of concept prototype. Robust user friendly interaction was an item with a low priority, proper operation and validation of the time stepping algorithm being the area of paramount importance. A graphic user inter-

face could be created, or additions could be made to GMSH 's GUI, allowing for specification of mesh files, matrix storage files, output file locations, etc. Features of this GUI would be identified upon further use of TRANSEDDY and from feedback from other users.

6.4 Conclusions

What we have accomplished with this research is the creation of a transient current density solver for use with thin conductors of arbitrary shape. The solver is restricted to magneto-quasistatic approximations for size and frequency. This solver does not make assumptions of constant or homogeneous conductivity, and has allowances for the addition of time changing conductivity. We accomplished this task by using the principle of superposition to add a source current density and an eddy current density on the same conductor. The solutions are implemented with a fractional step time stepping algorithm of an integral formulation for the eddy current density and a finite element solution to Laplace's equation for the source current density.

A circuit theory model for a strip line was implemented using the partial inductance method, and SPICE was used to generate both AC and transient solutions for the current density in the strip. This model was verified by the use of custom magnetic pickup probes designed at LLNL. The circuit theory model was used to verify the transient finite element model for the current distribution. The code, TRANSEDDY , was written to accomplish the implementation of the time-stepped finite element model. Testing of TRANSEDDY was performed by comparison of transient current density distributions with the results from the SPICE model.

This method can solve for the transient current density distributions in conductors of irregular shape. Thus we have developed a tool for analyzing current density flow and distributions in conductors which can't be modeled by analytical means.

Bibliography

[1] J. D. Jackson, *Classical Electrodynamics*, 3rd ed. New Jersey: John Wiley and Sons, Inc, 1999.

[2] H. A. Haus and J. R. Melcher, *Electromagnetic Fields and Energy.* New Jersey: Prentice-Hall, 1989.

[3] W. R. Smythe, *Static and Dynamic Electricity*, 3rd ed. McGraw-Hill, 1968.

[4] W. T. Weeks, L. L. Wu, M. F. McAllister, and A. Singh, "Resistive and inductive skin effect in rectangular conductors," *IBM J. Res. Develop*, vol. 23, no. 6, pp. 652–660, November 1979.

[5] M. M. Ney, "Striction and skin effects on the internal impedance value of flat conductors," *IEEE Trans. Electromagn. Compat.*, vol. 33, no. 4, pp. 321–327, November 1991.

[6] D. M. Sheen, S. M. Ali, D. E. Oates, R. S. Withers, and J. A. Kong, "Current distribution, resistance, and inductance for superconducting strip transmission lines," *IEEE Trans. Appl. Supercond.*, vol. 1, no. 2, pp. 108–115, 1991.

[7] G. Antonini, A. Orlandi, and C. R. Paul, "Internal impedance of conductors of rectangular cross section," *IEEE Trans. Micro. Theory Techn.*, vol. 47, no. 7, pp. 979–985, July 1999.

[8] L. J. Giacoletto, "Frequency and time-domain analysis of skin-effects," *IEEE Trans. Mag.*, vol. 32, pp. 220–229, January 1996.

[9] Z. Zhihua and M. Weiming, "Ac impedance of an isolated flat conductor," *IEEE Trans. Electromagn. Compat.*, vol. 44, no. 3, pp. 482–486, August 2002.

[10] "Quickfield," `http://www.quickfield.com/`, Tera Analysis Ltd.

[11] "Maxwell 3d," `http://www.ansoft.com/maxwell/`, Ansoft L.L.C.

[12] "Mafia 4," `http://www.cst.com/`, CST:Computer Simulation Technologies.

[13] T. Weiland, "A discretization method for the solution of maxwell's equations for six component fields," *Electronics and Communications AEU*, vol. 31, no. 3, pp. 116–120, 1977.

[14] M. Clemens, M. Wilke, and T. Weiland, "Advanced $\mathtt{FI}^2\mathtt{TD}$ algorithms for transient eddy current problems," *COMPEL:The International Journal for Computation and Mathematics in Electrical and Electrinic Engineering*, vol. 20, no. 2, pp. 365–379, 2001.

[15] M. Clemens and T. Weiland, "Discrete electromagnetism with the finite integration technique," *Progress in Electromagnetic Research*, vol. 32, pp. 65–87, 2001.

[16] R. Schuhmann and T. Weiland, "Conservation of discrete energy and related laws in the finite integration technique," *Progress in Electromagnetic Research*, vol. 32, pp. 301–316, 2001.

[17] T. Weiland, "Time domain electromagnetic field computation with finite difference methods," *International Journal of Numerical Modelling: Electronic Networks, Devices and Fields*, vol. 9, pp. 295–319, 1996.

[18] M. Bartsch and *et.al.*, "Solution of maxwell's equations," *Computer Physics Communications*, vol. 72, pp. 22–39, 1992.

[19] M. N. O. Sadiku, *Numerical Techniques in Electromagnetics*, 2nd ed. Florida: CRC Press, 2001.

[20] P. Dular and C. Geuzaine, "Getdp: a general environment for the treatment of discrete problems," `http://geuz.org/getdp/`

[21] C. Geuzaine, "High order hybrid finite element schemes for maxwell's equations taking thin structures and global quantities into account," Ph.D. dissertation, University of Lige, Belgium, December 2001.

[22] C. Geuzaine and J. F. Remacle, `http://geuz.org/gmsh/`

[23] C. Geuzaine and J. F. Remacle, "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities," `http://geuz.org/gmsh/doc/texinfo/gmsh.html`

[24] J. S. Lee, "A direct field formulation for transient eddy current calculations in thin conductors," *IEEE Transactions on Magnetics*, vol. 27, no. 5, September 1991.

[25] L. Pinchon, "Hybrid finite-element method and boundary-element method using time-stepping for eddy-current calculation in axisymmetric problems," *IEEE Proceedings*, vol. 136, no. 4, July 1989.

[26] M. Clemens and T. Weiland, "Transient eddy-current calculation with the FI-Method," *IEEE Transactions on Magnetics*, vol. 35, no. 3, pp. 1163–1166, May 1999.

[27] P. J. Leonard and D. Rodger, "Some aspects of two- and three-dimensional transient-eddy-current nodeling using finite elements and single-step time-marching algorithms," *IEE Proceedings*, vol. 135, no. 3, pp. 159–166, March 1998.

[28] K. H. Carpenter and H. T. Yeh, "Eddy current calculations for thin cylinders of finite length with driving fields of ramp time dependence," *IEEE Transactions on Magnetics*, vol. 12, no. 6, pp. 1059–1061, November 1976.

[29] H. Tsuboi and K. Kunisue, "Eddy current analysis of thin plates taking account of the source current distributions and it's experimental verifications," *IEEE Transactions on Magnetics*, vol. 27, no. 5, pp. 4020–4023, September 1991.

[30] P. Robert, M. Ito, and T. Takahashi, "Numerical solution of three dimensional transient eddy current problems by the **A**-Φ Method," *IEEE Transactions on Magnetics*, vol. 28, no. 2, pp. 1166–1169, March 1992.

[31] Z. Badics, "Transient eddy current field of current forced three-dimensional conductors," *IEEE Transactions on Magnetics*, vol. 28, no. 2, pp. 1232–1234, March 1992.

[32] C. J. Carpenter and M. Djurović, "Three-dimensional numerical solution of eddy currents in thin plates," *PROC. IEE*, vol. 122, no. 6, pp. 681–688, June 1975.

[33] C. J. Carpenter, "Finite-element network models and their applications to eddy-current problems," *PROC. IEE*, vol. 122, no. 4, pp. 455–462, April 1975.

[34] ——, "Computation of magnetic fields and eddy currents," in *Proceedings of the Fifth Int. Conf. on Magnetic Technology, Rome*, 1975, pp. 147–158.

[35] J. H. McWhirter, "Computation of three-dimensional eddy currents in thin conductors," *IEEE Transactions on Magnetics*, vol. 18, no. 2, pp. 456–460, March 1982.

[36] T. H. Petersen and C. M. May, patent disclosure by May and Petersen being processed by LLNL.

[37] T. L. Quarles, "Analysis of performance and convergence issues for circuit simulation," Ph.D. dissertation, Electronics Research Laboratory, College of Engineering, University of California, Berkeley, April 1989, a Ph.D. dissertation, with Appendices, UCB/ERL Memoranda M89/42, M89/43, M89/44, M89/45, M89/46, M89/47.

[38] M. W. Beattie and L. T. Pileggi, "Inductance 101: Modeling and extraction," in *DAC 2001.* Los Vegas, NV, USA: Design Automation Conference, June 18-22 2001, pp. 323–328, section IV.

[39] A. E. Ruehli, "Equivalent circuit models for three-dimensional multiconductor systems," *IEEE Trans. Micro. Theory Tech.*, vol. 22, no. 3, pp. 216–221, March 1974.

[40] C. Christopoulos, *The Transmission-Line Modeling Method: TLM.* IEEE Press, 1995.

[41] M. T. Heath, *Scientific Computing, An Introductory Survey*, 2nd ed. McGraw-Hill, 2002, section 8.4.4.

[42] J. Berntsen, T. O. Espelid, and A. Genz, "Algorithm 698: DCUHRE: An Adaptive Multidimensional Integration Routine for a Vector of Integrals," *ACM Transactions on Mathematical Software*, vol. 17, no. 4, pp. 452–456, 1991.

[43] M. Kamon, M. J. Tsuk, and J. White, "FastHenry: A multipole accelerated 3-D inductance extraction program," *IEEE Trans. Micro. Theory Techn.*, vol. 42, no. 9, pp. 1750–1758, September 1994.

[44] T. L. Quarles, information on various versions of SPICE, and links to sites where they can be obtain, can be found at `//bwrc.eecs.berkeley.edu/Classes/IcBook/SPICE/`.

[45] M. Koizumi and M. Onizawa, "Computational method of three dimensional eddy current by using volume integral equation method," *IEEE Transactons on Magnetics*, vol. 27, no. 5, pp. 4077–4080, 1991.

[46] M. N. O. Sadiku, *Numerical Techniques in Electromagnetics*, 2nd ed. Florida: CRC Press, 2001.

[47] J. C. Nedelec, "Mixed Finite Elements in $\mathbb{E}^3$," *Numerische Mathematik*, vol. 35, pp. 315–341, 1980.

[48] A. Bossavit, "Whitney forms: A class of finite elements for three-dimensional computations in electromagnetim," *IEE Proc.*, vol. 135, pp. 493–499, 1988.

[49] S. C. Brenner and L. R. Scott, *The Mathematical Theory of Finite Element Methods*, 2nd ed. New York:Springer-Verlag, Inc., 2002.

[50] K. H. Carpenter, "On efficient computation of integrals of the free space green's function for laplace's equation over triangular finite elements," in *Proceedings of the 2007 International Conference on Scientific Computing*, H. R. Arabnia, J. Y. Yang, and M. Q. Yang, Eds. CSREA Press, 2007, pp. 76–80.

[51] I. Robinson, "An algorithm for automatic integration over a triangle using nonlinear extrapolation," *ACM Transactions on Mathematical Software (TOMS)*, vol. 10, no. 1, pp. 1–16, March 1984.

[52] E. de Doncker and I. Robinson, "Algorithm 612: Triex: Integration over a triangle using nonlinear extrapolation," *ACM Transactions on Mathematical Software (TOMS)*, vol. 10, no. 1, pp. 17–22, March 1984.

[53] R. Davies, "Newmat c++ matrix library," (1990-2006) Newmat C++ matrix library `http://www.robertnz.com/ol_doc.htm`

[54] E. Anderson, Z. Bai, J. Dongarra, A. Greenbaum, A. McKenney, J. Du Croz, S. Hammerling, J. Demmel, C. Bischof, and D. Sorensen, "Lapack: a portable linear algebra library for high-performance computers," in *Supercomputing '90: Proceedings of the 1990 conference on Supercomputing.* Los Alamitos, CA, USA: IEEE Computer Society Press, 1990, pp. 2–11.

[55] D. M. Young Jr, "Iterative Method for Solving Partial Difference Equations of Elliptic Type," Ph.D. dissertation, Harvard University, Cambridge, MA, May 1950.

[56] M. T. Heath, *Scientific Computing: An Introductory Survey*, 2nd ed. McGraw-Hill, 2002.

[57] J. D. Logan, R. S. Lee, R. C. Weingart, and K. S. Yee, "Calculation of heating and burst phenomena in electrically exploded foils," *Journal of Applied Physics*, vol. 48, no. 2, pp. 621–628, February 1977.

[58] V. S. Sedoi, G. A. Mesyats, V. I. Oreshkin, V. V. Valevich, and L. I. Chemezova, "The current density and the specific energy input in fast electrical explosion," *IEEE Transactions on Plasma Science*, vol. 27, no. 4, pp. 845–850, August 1999.

[59] D. A. DeWolf, *Essentials of Electromagnetics for Engineering.* Cambridge University Press, 2001.

[60] W. B. Kuhn and N. M. Ibrahim, "Analytical modeling of current crowding effects in multi-turn spiral inductors," *IEEE Transactions on Microwave Theory and Techniques*, vol. 49, pp. 31–38, January 2001.

[61] N. M. Ibrahim and W. B. Kuhn, "An approach for the calculation of magnetic field within square spiral inductors at low frequency," *International Journal of Numerical Modeling: Electronic Networks, Devices and Fields*, vol. 15, pp. 339–354, July 2002.