Advances in Numerical Modelling of Waveguide Discontinuities in a Microwave Receiver Protector

by Andrew Morley, MEng

Thesis submitted to the University of Nottingham
for the degree of Master of Research, September 2017



Abstract

Microwave communication systems are used worldwide in applications as varied as weather forecasting, satellite navigation and air traffic control.

Microwave receivers located in Radio Frequency (RF) communication systems contain sensitive components which need to be protected against the potential damage caused by incident electromagnetic (EM) fields. Computational modelling of the devices that protect such a receiver is therefore essential to verify their effectiveness and improve their design. However, current modelling techniques are not capable of providing sufficiently accurate data at all frequencies of interest to be sure that the level of protection is satisfactory.

In this thesis, the capabilities of two novel, numerical modelling techniques - the Unstructured Transmission Line Modelling (UTLM) method and the Boundary Element Method (BEM) - are assessed to determine to what extent they can provide this sufficient level of modelling accuracy concerning the impact of waveguide discontinuities that characterise the behaviour of these protection devices.

Acknowledgements

The completion of the work presented in this thesis would not have been possible without the support and assistance of a great many people:

- Firstly, my sincere thanks go to my supervisors, Dr Christopher Mellor, Professor Trevor Benson and Dr Kristof Cools. Thank you all for your patience and clarity when advising and guiding me through a large number of complex areas in this wide-ranging project, and for facilitating so well both my learning in the field and the progress I have made towards the project's goals. Thanks also for your insightful suggestions for and proof-reading of this thesis.

- Special thanks go to Professor Phillip Sewell and Dr Ana Vukovic at the George Green Institute for Electromagnetics Research for their guidance and assistance in applying UTLM during this project.

- I am indebted to Gary Stimson and Keith Newsome at Teledyne E2V in Lincoln for their time spent explaining the purpose and operation of receiver protector devices to me, and for giving me a firm understanding of the industrial relevance of this project - thank you both.

- To all my friends at the George Green Institute for Electromagnetics Research - both past and present - thank you all for your friendship and support over the past two years. All the best for the future!

- A fond farewell to my fellow students in the School of Physics & Astronomy - I have greatly appreciated your companionship over the past year and the endless stream of fascinating discussions we have had in the tearoom! I'll miss you all.

- My heartfelt thanks go to my parents, sister and my Christian family here in Nottingham for their constant love and support through the highs and lows of the past year. In particular, thank you Dad for your special attention to detail when proof-reading this thesis.

- A debt of thanks is also due to my friends Simon & Abi Evans and Emily & Uche Udeh for allowing me to use their Internet and the peace and quiet of their homes to make the necessary corrections to this thesis.

And above all, my thanks and praise are to God, who sustains, directs and loves me unconditionally. To Him I can give nothing, but from Him I have received everything.

Contents

Abstract **1**

Acknowledgements **2**

List of Symbols **6**

List of Abbreviations **8**

1 Introduction **9**

1.1 Problem Statement 9

1.2 Background to Microwave Communications 10

1.3 Solid State Receiver Protectors (SSRPs): Design and Operation 10

1.4 SSRP Modelling: Current Inadequacies & Motivation for Work 12

1.5 Outline of Thesis 13

2 Waveguides, Discontinuities & Analytical Modelling **15**

2.1 Waveguide Theory 15

2.1.1 Wave Propagation & Waveguide Modes 16

2.1.2 Transmission Line Theory 19

2.2 Analytical Modelling of Inductive Post 22

2.2.1 Derivation of Scattering Parameters 22

2.2.2 MATLAB™Implementation 24

2.3 Chapter Summary 27

3 Literature Review of Numerical Modelling Techniques 28

3.1 Introduction to Numerical Techniques 28

3.2 Common Numerical Techniques 30

3.2.1 Finite Difference Time Domain (FDTD) Method 30

3.2.2 Transmission Line Modelling (TLM) 31

3.2.3 Finite Element Method (FEM) 31

3.3 Proposed Numerical Techniques 32

3.3.1 Unstructured Transmission Line Modelling (UTLM) 32

3.3.2 Boundary Element Method (BEM) 32

3.4 Chapter Summary 34

4 Unstructured Transmission Line Modelling (UTLM) 35

4.1 UTLM Theory 35

4.1.1 1D TLM: The Scatter-Connect Principle 36

4.1.2 2D and 3D TLM: Modelling TE, TM and TEM Propagation 38

4.1.3 Unstructured Meshes 39

4.2 Existing Implementation and Application to Waveguide-Post 41

4.2.1 Geometry Creation 41

4.2.2 Mesh Generation 41

4.2.3 Solver Setup and Simulation 42

4.3 Chapter Summary 45

5 Boundary Element Method (BEM) 47

5.1 BEM Theory 47

5.1.1 Derivation of the Electric Field Integral Equation (EFIE) 48

5.1.2 Basis Functions 50

5.1.3 Solving EFIE Matrix for Unknown Coefficients 52

5.2 Existing Implementation and Application to Waveguide-Post 53

5.2.1 Mesh Generation and Interface with UTLM 53

5.2.2 Basis Functions, Excitations and Solving 55

5.2.3 Post-Processing 56

5.3 Chapter Summary 57

6 Numerical Modelling Results & Comparison 58

6.1 Summary of Simulation Parameters 58

6.2 CST Microwave Studio™Results 59

6.3 UTLM Results 61

6.4 BEAST Results 64

6.5 Chapter Summary 65

7 Thesis Summary, Conclusions & Future Work 66

7.1 Thesis Summary 66

7.2 Conclusions 68

7.3 Proposed Framework for Subsequent Work 70

A MATLAB Implementation of Analytical Model 71

B UTLM Implementation of Waveguide-Post Geometry 77

C UTLM-BEAST Mesh Modification Script 88

D BEAST Implementation of Waveguide-Post Simulation 106

D.1 Example: Waveguide-Post Model 106

D.2 Function: Parsing UTLM Surface Mesh into BEM 113

D.3 Function: Scattering Parameter Computation 115

References 118

List of Symbols

β	Phase constant
Δx	Maximum cubic UTLM mesh cell size
Δz	Slice thickness of UTLM modal excitation
ϵ	Electric permittivity
η_{eff}	Effective refractive index
λ	Wavelength
λ_0	Wavelength at maximum frequency of UTLM simulation
λ_g	Guide wavelength
λ_g	Guide wavelength
μ	Magnetic permeability
ω	Angular frequency
$\mathbf{B}$	Magnetic flux density
$\mathbf{D}$	Electric flux density
$\mathbf{E}$	Electric field intensity
$\mathbf{H}$	Magnetic field intensity
$\mathbf{J}$	Electric current density

$\mathbf{M}$	Magnetic current density
$\mathbf{r}$	Source position vector
$\mathbf{r}'$	Observation position vector
a	Waveguide width
b	Waveguide height
c	Speed of electromagnetic propagation in a vacuum ($= 3.0 \times 10^8$ ms^{-1})
C_d	Capacitance per unit length
f_{modal}	Frequency at which mode is simulated in UTLM
$I(z)$	Spatially-varying current propagating in the z-direction along a 1D transmission line
j	Unit imaginary number
k	Wavenumber
k_c	Cutoff wavenumber
L_d	Inductance per unit length
m	Mode order number, along axis of waveguide width
n	Mode order number, along axis of waveguide height
q_e	Electric charge
q_m	Magnetic charge
s	UTLM clustering threshold
$V(z)$	Spatially-varying voltage propagating in the z-direction along a 1D transmission line
Z_c, Z_0	Characteristic Impedance

List of Abbreviations

ABC	Absorbing Boundary Condition
BEAST	Boundary Element Analysis and Simulation Toolkit
BEM	Boundary Element Method
DE	Differential Equation
EFIE	Electric Field Integral Equation
EM	Electromagnectic
FD	Frequency Domain
FDTD	Finite Difference Time Domain
FEM	Finite Element Method
GGIEMR	George Green Institute for Electromagnetics Research
IE	Integral Equation
KPI	Key Performance Indicator
MFIE	Magnetic Field Integral Equation

PEC	Perfect Electric Conductor
PML	Perfectly Matched Layer
RF	Radio Frequency
RT	Raviart-Thomas
SSRP	Solid State Receiver Protector
TD	Time Domain
TE	Transverse Electric
TEM	Transverse Electromagnetic
TLM	Transmission Line Modelling
TM	Transverse Magnetic
UTLM	Unstructured Transmission Line Modelling

Chapter 1

Introduction

In this chapter, the motivation for the work undertaken is established, by introducing microwave communications as a field of research, exploring the function of microwave receiver protector devices and identifying inadequacies in the way that these devices are modelled during the design process, as per the problem statement.

1.1 Problem Statement

Microwave receivers located in radio frequency (RF) communication systems contain sensitive components which need to be protected against the potential damage caused by incident electromagnetic (EM) fields. Computational modelling of the devices that protect such a receiver is therefore essential to verify their effectiveness and improve their design. However, current modelling techniques are not capable of providing sufficiently accurate data at all frequencies of interest to be sure that the level of protection is satisfactory [1].

1.2 Background to Microwave Communications

Microwave communication systems are used worldwide in applications as varied as weather forecasting, satellite navigation and air traffic control. The RF signals used to convey information in these systems are often transmitted over thousands of kilometres, over which the signal strength is significantly reduced. Very sensitive components are therefore incorporated into microwave receivers to process these small-amplitude, low-power signals.

Consequently, the possibility of signals with a relatively high power level arriving at and propagating through the receiver presents a considerable risk to the system - these signals have the potential to significantly damage the sensitive components of the receiver. To prevent such damage, modules known as receiver protectors are fitted in front of the receiver stage of a microwave system.

1.3 Solid State Receiver Protectors (SSRPs): Design and Operation

The particular configuration of receiver protector being considered in this project is the Solid State Receiver Protector (SSRP). The SSRP concept was first conceived in the 1960s as a more rugged, compact and reliable design to supplement the original gas discharge tube implementation, and which contributed the novel capability of blocking a specific band of frequencies rather than a crude broadband response[2, 3, 4].

A SSRP consists of a series of diode limiter stages within a short length of waveguide. Figure 1.1 shows a cross-section of one such stage.

As shown in Figure 1.1, a single SSRP stage consists of an inductive post placed vertically in a section of hollow rectangular waveguide, at the bottom of which a PIN diode is attached, and two threaded bosses secure the post in place. Below a threshold level of voltage across upon the post, the diode remains unbiased (i.e. non-conducting)

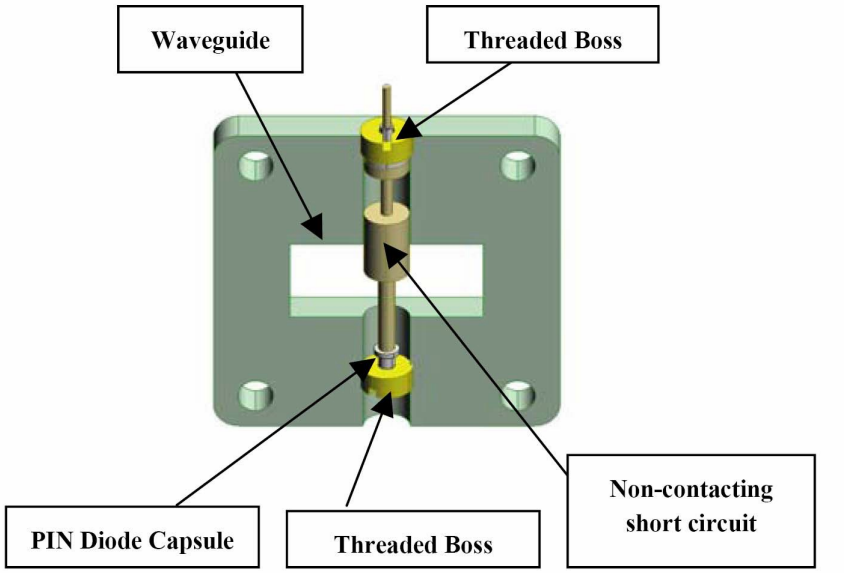

Figure 1.1: A cross-section of one stage of a receiver protector [2]

and therefore the post is open circuited. Therefore, the impedance of the waveguide is not affected and the wave can propagate past the post uninhibited; this is known as the 'OFF' state [5].

However, when the diode's threshold voltage is exceeded, the RF signals begin to be rectified, resulting in a net injection of carriers into the diode's intrinsic (I) region - that is, the semiconductor layer sandwiched between the p- and n-doped layers [6]. Consequently, the diode becomes forward-biased and the post becomes a short circuit to ground, causing a mismatch in the waveguide's impedance; the high reflection coefficient associated with such a mismatch causes a large proportion of the wave to be reflected back along the waveguide instead of propagating through it. This is known as the 'ON' state, by which the receiver is protected [5]. To preserve the binary nature of both states, a non-contacting short circuit (aka RF choke) is integrated into the post to ensure only direct current (DC) flows in the ON state, rather than multiple higher frequencies [7].

A single diode limiter stage can provide approximately 20-30dB of isolation of super-threshold power levels[5], but SSRPs often require multiple stages (as shown in Figure 1.2) to cope with different power ranges.

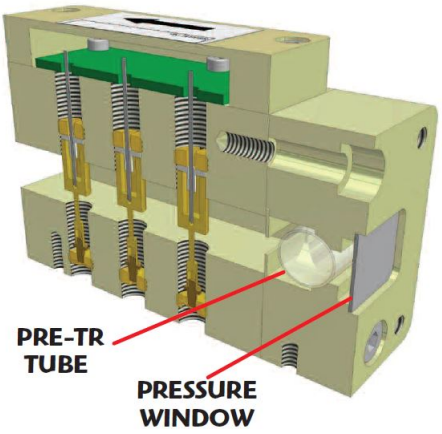

Figure 1.2: A lengthwise cross-section of a multi-stage SSRP [2]

Diodes with a relatively thick I-region are positioned closest to the input of the SSRP. These diodes have a higher threshold voltage, and therefore activate in the presence of higher power signals and stop them propagating further through the system. Such diode stages are known as 'coarse' stages. Conversely, diodes with a thinner I-region are positioned progressively closer to the output of the SSRP, and they activate at lower power thresholds. These diode stages are known as 'clean-up' stages [5].

SSRPs manage signals with power levels in the order of 10kW, whereas other implementations of receiver protector, such as microstrip, are used to cater for lower power levels (e.g. ∼100W)[8].

1.4 SSRP Modelling: Current Inadequacies & Motivation for Work

In order to more comprehensively understand the complex EM response of SSRPs, attempts have been made previously to model an entire SSRP structure numerically - that is, by discretising the geometry into a mesh and applying a particular algorithm to simulate the electric and magnetic fields scattered through the structure according to a given excitation.

To date, numerical models of SSRPs have been constructed using commercial 3D modelling software packages such as Ansoft HFSS™and CST Microwave Studio™[9]. Whilst these packages can accurately model the frequency response of an SSRP *within* its operating band (typically 8 - 12GHz), their capability yield accurate data for its behaviour at higher, out-of-band frequencies (up to ∼40GHz) - and to do so efficiently - is hampered by their inability to combine the capture of complex, multiscale geometrical features with the simulation of a transient response [1]. Consequently, microwave engineers are currently unable to use numerical modelling tools to design SSRPs to protect against the propagation of high-frequency signals through subsequent stages of the microwave receiver, which could cause severe damage to the system [1].

Therefore, in order to improve the design of SSRPs and their behaviour at high frequencies, a more accurate modelling technique must be employed. This thesis outlines the work done to evaluate the capabilities of two candidate modelling techniques in this regard.

1.5 Outline of Thesis

In this chapter, the background of the problem statement has been explored; the role and operation of SSRPs in microwave systems has been described, and the need for current modelling techniques to be supplanted by one that can accurately capture the broadband behaviour of SSRPs has been established.

The remainder of this thesis is structured as follows:

- **Chapter 2** establishes an appropriate analytical model to represent an SSRP - as well as its key performance indicators - by reviewing the fundamental theory and classical models of waveguides and discontinuities (i.e. alterations to the waveguide dimensions or addition of physical obstructions inside - of which a diode stage is an example);

[1]The inadequacies of modelling high-frequencies in traditional methods such as FDTD and TLM are expounded upon in Chapter 3

- **Chapter 3** explains the selection of two candidate numerical modelling techniques - the Unstructured Transmission Line Modelling (UTLM) method and the Boundary Element Method (BEM) - by distinguishing numerical models from analytical models and reviewing the advantages and drawbacks of each category of numerical modelling technique;

- **Chapter 4** describes the principles and existing implementation of the UTLM method, outlines the methodology followed in applying it to the problem in question, details the novel contributions made to the method and compares the results obtained to those of the analytical model;

- **Chapter 5** describes the principles and existing implementation of the BEM method, outlines the methodology followed in applying it to the problem in question, details the novel contributions made to the method (especially in the area of geometry capture and meshing) and explains why the results obtained were not found to be comparable to those of the UTLM and analytical models;

- **Chapter 6** reintroduces the existing CST modelling approach and compares the results obtained from it to those of the analytical and UTLM models. Comparisons are then drawn between each of the four approaches (analytical, CST, UTLM and BEM) and each is evaluated for its ability to deliver the high-frequency models of SSRPs that are required.

- **Chapter 7** outlines the future work necessary to improve modelling of SSRPs using the UTLM and BEM techniques;

- **Chapter 8** reviews the work outlined in the thesis and summarises the progress made in simplifying the methodologies of the UTLM and BEM techniques.

Chapter 2

Waveguides, Discontinuities & Analytical Modelling

In this chapter, the fundamental theory and classical models of waveguides and discontinuities are reviewed, informing the selection of an appropriate simplified analytical model of the solid state receiver protector (SSRP) that acts as a benchmark against which the subsequent numerical results are compared. The MATLAB™implementation of this analytical model is also described.

2.1 Waveguide Theory

In order to understand the nature of high-frequency propagation in SSRPs, it is necessary to first introduce the physics behind electromagnetic (EM) propagation in waveguides in general.

A waveguide is a type of transmission line, i.e. a channel along which EM waves propagate. Typically, waveguides have a metallic boundary and can either be hollow and held in a vacuum or filled with a dielectric such as air [10, 11]. In a SSRP, the channel along which EM waves propagate is rectangular in shape, therefore only the theory governing

the operation of rectangular waveguides is considered in this thesis.

2.1.1 Wave Propagation & Waveguide Modes

In a source-free transmission line (of which a waveguide is an example), sinusoidal wave propagation obeys Maxwell's equations for a source-free region:

$$\nabla \times \mathbf{E} = -j\omega\mu\mathbf{H}, \tag{2.1}$$

$$\nabla \times \mathbf{H} = j\omega\epsilon\mathbf{E}, \tag{2.2}$$

where $\mathbf{E}$ is the electric field intensity, $\mathbf{H}$ is the magnetic field intensity, j is the uni imaginary number, ω is the angular frequency, μ is the magnetic permeability and ϵ is the electric permittivity.

When fully expanded, equations 2.1 and 2.2 describe the relationships between all six transverse (E_x, E_y, H_x, H_y) and longitudinal (E_z, H_z) field components, as follows [11]:

$$\frac{\partial E_z}{\partial y} + j\beta E_y = -j\omega\mu H_x, \tag{2.3}$$

$$-\frac{\partial E_z}{\partial x} - j\beta E_x = -j\omega\mu H_y, \tag{2.4}$$

$$\frac{\partial E_y}{\partial x} - \frac{\partial E_x}{\partial y} = -j\omega\mu H_z, \tag{2.5}$$

$$\frac{\partial H_z}{\partial y} + j\beta H_y = j\omega\epsilon E_x, \tag{2.6}$$

$$-\frac{\partial H_z}{\partial x} - j\beta H_x = j\omega\epsilon E_y, \tag{2.7}$$

$$\frac{\partial H_y}{\partial x} - \frac{\partial H_x}{\partial y} = j\omega\epsilon E_z, \tag{2.8}$$

where β is the phase constant given by $\beta = \sqrt{k^2 - k_c^2}$, k is the wavenumber given by $k = \omega\sqrt{\mu\epsilon}$, and k_c is the cutoff wavenumber (to be explained later).

In waveguides, only Transverse Electric (TE) waves (i.e. with no E_z component) or Transverse Magnetic (TM) waves (i.e. with no H_z component) can propagate[1] [10]. Therefore, when equations 2.3 - 2.8 are rearranged, each of the transverse components can be expressed as dependent on only one of the longitudinal components. For example, by eliminating E_z, the TE field profile can be expressed as dependent on H_z [11]:

$$E_x = -\frac{j\omega\mu}{k_c^2}\frac{\partial H_z}{\partial y}, \tag{2.9}$$

$$E_y = \frac{j\omega\mu}{k_c^2}\frac{\partial H_z}{\partial x}, \tag{2.10}$$

$$H_x = -\frac{j\beta}{k_c^2}\frac{\partial H_z}{\partial x}, \tag{2.11}$$

$$H_y = -\frac{j\beta}{k_c^2}\frac{\partial H_z}{\partial y}. \tag{2.12}$$

Therefore, all that remains to complete the analytical expressions of the TE field profile is to determine H_z, which can be achieved by solving the homogeneous wave (or Helmholtz) equation that can be derived from 2.1 and 2.2 [11]:

$$\left(\frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2} + \frac{\partial}{\partial z^2} + k^2\right) H_z = 0. \tag{2.13}$$

With $H_z = h_z(x, y)e^{-j\beta z}$, equation 2.13 can be reduced to

$$\left(\frac{\partial}{\partial x^2} + \frac{\partial}{\partial y^2} + k^2\right) h_z(x, y) = 0. \tag{2.14}$$

In the case of a rectangular waveguide, equation 2.14 has a solution of the form

$$h_z(x, y) = A\cos\frac{m\pi x}{a}\cos\frac{n\pi y}{b} \tag{2.15}$$

[1]Transverse Electromagnetic (TEM) wave propagation is not possible in a waveguide since only one conductor is present, meaning longitudinal electric and magnetic field components cannot *both* be supported.

where A is an arbitrary constant, a and b are the width and height of the waveguide respectively, and m and n are integers (i.e. $m, n = 0, 1, 2...$) which denote the number of wave maxima in the x and y directions (respectively) of the particular shape of field profile which propagates, known as the mode. The field profiles of some low order modes (i.e. those with fewer maxima) are shown in Figure 2.1.

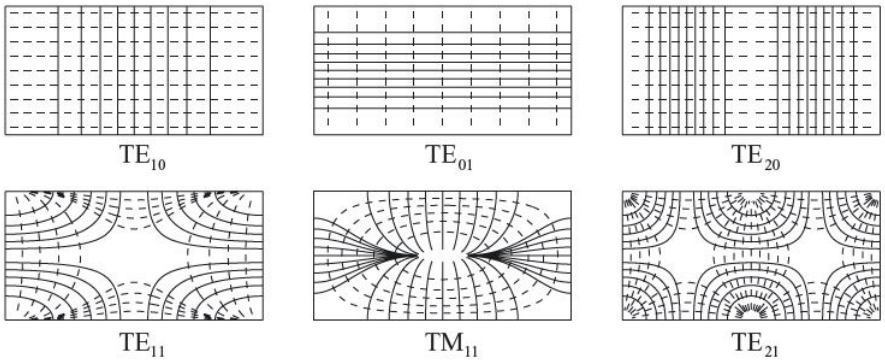

Figure 2.1: Low-order waveguide mode profiles (solid lines are electric field lines, dashed lines are magnetic field lines)[12]

By substituting equation 2.15 into equations 2.9 - 2.12, the field profiles of all four transverse field components of a TE wave can be obtained. Similarly, the TM field profiles can be obtained by eliminating H_z from equations 2.3 to 2.8 and solving for E_z instead [11].

The particular TE or TM modes that can propagate depend on the wave's frequency. For each mode, the waveguide exhibits the properties of a high-pass filter; above a certain cutoff frequency, f_c, the wave can propagate in that mode, but below f_c it cannot, and the mode is described as evanescent. f_c is governed by the following relationship [11]:

$$f_c = \frac{c}{2\pi}\sqrt{\left(\frac{m\pi}{a}\right)^2 + \left(\frac{n\pi}{b}\right)^2}, \tag{2.16}$$

where c is the speed of electromagnetic propagation in a vacuum ($= 3.0 \times 10^8$ ms^{-1}).

Thus, for given waveguide dimensions then the higher the order of the mode, the higher the cutoff frequency and the higher the wave frequency must be in order for the propagation of that mode to be observed [10], and the shorter the guide wavelength, λ_g (that is, the

distance between points of equal phase of the propagating mode [12]). Recalling from Chapter 1 that it is at higher frequencies where advances in SSRP modelling accuracy are needed most, it follows that because SSRPs are waveguide devices, this high frequency propagation manifests itself as higher order TE and TM modes, and it is these modes that candidate techniques need to be able to simulate if SSRP modelling is to be improved in this regard.

In a waveguide that is free of any discontinuities, TE and TM modes can propagate freely. However, in the case of a SSRP, propagating waves encounter each of the diode stages (i.e. discontinuities), resulting in complex reflection and transmission responses. The most natural approach to take when seeking to model this phenomena analytically is to apply transmission line theory - that is, to represent the introduction of discontinuities in a waveguide as changes in the impedance of a transmission line. This approach is now outlined.

2.1.2 Transmission Line Theory

Firstly, the link between wave propagation and transmission line theory is introduced and the impedance of a lossless transmission line (i.e. one with no discontinuities, of which a hollow waveguide is an example) is considered.

The propagation of EM waves through a medium is analogous to the propagation of voltages and currents in a circuit. This is exemplified by the fact that the homogeneous Helmholtz equations that represent the dynamic response of voltages and currents (albeit in a single dimension) are akin to the 3D wave equation (equation 2.13). For example, the voltage along a lossless line oriented in the z-direction is governed by the equation:

$$\frac{d^2V(z)}{dz^2} + \beta^2 V(z) = 0, \tag{2.17}$$

where β is still termed the phase constant and is equivalent to k and is defined as $\beta =$

$\omega\sqrt{L_d C_d}$[2], with L_d and C_d denoting the inductance and capacitance per unit length of the transmission line.

The solutions of equation 2.17, and its equivalent for current, are, respectively,

$$V(z) = V_A e^{j\beta z} + V_B e^{-j\beta z} \tag{2.18}$$

and

$$I(z) = I_A e^{j\beta z} + I_B e^{-j\beta z}, \tag{2.19}$$

where each solution comprises forward and backward propagating terms (z and -z respectively) and V_A, V_B, I_A, and I_B are constants.

The characteristic impedance, Z_c, of a lossless line can be defined [13] as

$$Z_c = \frac{V(z)}{I(z)} \left(= \sqrt{\frac{L_d}{C_d}} \right). \tag{2.20}$$

Equation 2.20 is used to parameterise the first and second terms of each solution in terms of the voltage ($V(0)$) and current ($I(0)$) at the line's origin respectively, yielding:

$$V(z) = V(0)\cos\beta z - I(0) j Z_c \sin\beta z; \tag{2.21}$$

$$I(z) = -V(0) j \frac{\sin\beta z}{Z_c} + I(0)\cos\beta z. \tag{2.22}$$

After rearranging 2.21 and 2.22 to find expressions for $V(0)$ and $I(0)$, and replacing z with the length of the line, l, to denote the voltages and currents at the end of the line, the pair of equations can be represented in matrix form:

[2]This definition of the phase constant is specific to the lossless case. In general (i.e. accounting for losses), the phase constant is complex and is denoted $\gamma = \alpha + j\beta$

$$\begin{bmatrix} V(0) \\ I(0) \end{bmatrix} = \begin{bmatrix} \cos\beta l & jZ_c \sin\beta l \\ j\frac{\sin\beta x}{Z_c} & \cos\beta l \end{bmatrix} \begin{bmatrix} V(l) \\ I(l) \end{bmatrix}. \tag{2.23}$$

Expressing the equations in this matrix form characterises the transmission line as a network of input and output ports, whose responses are interrelated. The particular formulation in 2.23 is known as the transmission (or ABCD) matrix, where the layout of matrix elements in 2.23 equates to the following:

$$\begin{bmatrix} V(0) \\ I(0) \end{bmatrix} = \begin{bmatrix} A & B \\ C & D \end{bmatrix} \begin{bmatrix} V(l) \\ I(l) \end{bmatrix}. \tag{2.24}$$

It is appropriate to extract the necessary scattering (or S) parameters from the ABCD parameters, as the former are the most common key performance indicators (KPIs) of the response of passive microwave networks. The two elements in the S-matrix that are most useful are S_{11} and S_{21} which represent the reflection loss and transmission loss of the line respectively, and henceforth these quantities are the KPIs used to examine the relative quality of each model. These parameters are derived via the following conversions [11]:

$$S_{11} = \frac{A + \frac{B}{Z_c} - \frac{C}{Z_c} - D}{A + \frac{B}{Z_c} + \frac{C}{Z_c} + D}, \tag{2.25}$$

$$S_{21} = \frac{2}{A + \frac{B}{Z_c} + \frac{C}{Z_c} + D}. \tag{2.26}$$

When the ABCD parameters for a lossless line are entered into 2.25 and 2.26, S_{11} and S_{21}, reduce to

$$S_{11} = 0, \tag{2.27}$$

$$S_{21} = e^{-j\beta l}. \tag{2.28}$$

This is as might be expected, since with no losses, the reflection coefficient (i.e. the

proportion of the input signal that is reflected) is 0 and the magnitude of the transmission coefficient (i.e. the proportion of the input signal that is transmitted) is 1.

However, when discontinuities such as those found in a SSRP are introduced, the effect is to introduce inductive and capacitive components to the total impedance of the waveguide, which consequently alter the ABCD parameters and the reflection and transmission response of the device. The procedure followed to integrate these changes in the transmission line model are described in the following section.

2.2 Analytical Modelling of Inductive Post

2.2.1 Derivation of Scattering Parameters

The most appropriate way, geometrically speaking, of representing a single diode stage during the SSRP's 'ON' state of operation is as a cylindrical inductive (i.e. metallic) post oriented vertically in the waveguide.

An inductive post is a canonical example of a waveguide discontinuity, and the change in impedance that results when introducing it to a hollow waveguide can be represented in the transmission line model as Marcuvitz' equivalent circuit, which is shown on the right-hand side of Figure 2.2, with a sketch of the inductive post positioning in the waveguide on the left-hand side of that figure.

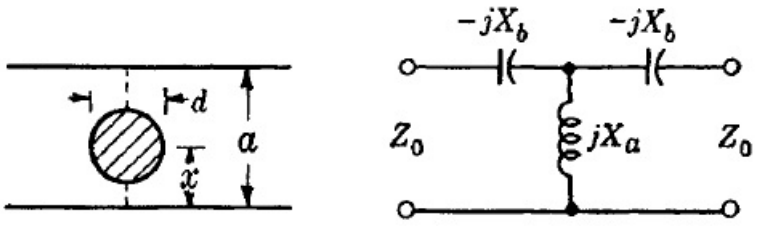

Figure 2.2: Marcuvitz' equivalent circuit of an inductive post [14]

In Figure 2.2, d is the diameter of the post, x is the distance of the post centre from the waveguide wall, $Z_0 = Z_c$, and X_a and X_b are the reactances associated with the inductive and capacitive components respectively.

Via a variational method [14, 15], X_a and X_b can be defined for a post centred in the middle of the waveguide (i.e. $x = \frac{a}{2}$) using the following simultaneous equations [14]:

$$\frac{X_a}{Z_c} - \frac{X_b}{2Z_c} = \frac{a}{2\lambda_g}\left[S_0 - \left(\frac{\pi d}{2\lambda}\right)^2 - \frac{5}{8}\left(\frac{\pi d}{2\lambda}\right)^4 - 2\left(\frac{\pi d}{2\lambda}\right)^4\left(S_2 - 2S_0\frac{\lambda^2}{\lambda_g^2}\right)^2\right], \tag{2.29}$$

$$\frac{X_b}{Z_c} = \frac{4a}{\lambda_g}\frac{\left(\frac{\pi d}{a}\right)^2}{1 + \frac{1}{2}\left(\frac{\pi d}{\lambda}\right)^2\left(S_2 + \frac{3}{4}\right)}, \tag{2.30}$$

where λ_g is the guide wavelength, λ is the signal wavelength and S_0 and S_2 are convergent terms, defined thus [14]:

$$S_0 = \ln\frac{4a}{\pi d} - 2 + 2\sum_{n=3,5\ldots}^{\infty}\left[\frac{1}{\sqrt{n^2 - \left(\frac{2a}{\lambda}\right)^2}} - \frac{1}{n}\right], \tag{2.31}$$

$$S_2 = \ln\frac{4a}{\pi d} - \frac{5}{2} + \frac{11}{3}\left(\frac{\lambda}{2a}\right)^2 - \left(\frac{\lambda}{a}\right)^2\sum_{n=3,5\ldots}^{\infty}\left[\frac{1}{\sqrt{n^2 - \left(\frac{2a}{\lambda}\right)^2}} - n + \frac{2}{n}\left(\frac{a}{\lambda}\right)^2\right]. \tag{2.32}$$

By substituting equation 2.30 into 2.29 and rearranging, X_a can be obtained. In turn, when X_a and X_b are substituted into Figure 2.2, the complete set of impedances in the network can be known. The network in Figure 2.2 is a special case of the 'T-shape' equivalent circuit for a general discontinuity shown in Figure 2.3.

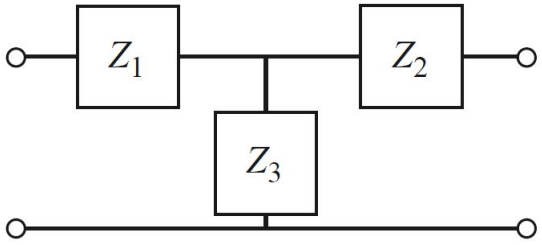

Figure 2.3: 'T-shape' equivalent circuit for a general discontinuity [11]

From Figure 2.2 it is known that $Z_2 = Z_1$; therefore, the ABCD parameters for such a network, given in [11], can be simplified as follows:

$$\begin{bmatrix} V(o) \\ I(o) \end{bmatrix} = \begin{bmatrix} 1 + \frac{Z_1}{Z_3} & 2Z_1 + \frac{Z_1^2}{Z_3} \\ \frac{1}{Z_3} & 1 + \frac{Z_1}{Z_3} \end{bmatrix} \begin{bmatrix} V(l) \\ I(l) \end{bmatrix}. \tag{2.33}$$

Finally, using equations 2.24, 2.25 and 2.26, $S_{11}$and S_{21} can be derived:

$$S_{11} = \frac{2Z_1Z_3 + Z_1^2 - Z_c^2}{2Z_1Z_3 + 2Z_1Z_c + 2Z_3Z_c + Z_1^2 + Z_c^2}, \tag{2.34}$$

$$S_{21} = \frac{2Z_3Z_c}{2Z_1Z_3 + 2Z_1Z_c + 2Z_3Z_c + Z_1^2 + Z_c^2}. \tag{2.35}$$

The implementation of these formulae in a MATLAB™script is outlined in the following section, followed by a discussion of the results yielded.

2.2.2 MATLAB™Implementation

To implement this model, a MATLAB™script was written[3] which defined a series of constants and combined the computation of the reactances and S-parameters into an iterative loop across a range of frequencies. Figure 2.4 shows the process followed by the script.

The specific values chosen for each of the quantities in the initialisation stage of Figure 2.4, are summarised below:

- The WG16 waveguide standard was selected, with dimensions $a = 22.86$mm and $b = 10.86$mm, because of the similarity to an SSRP in terms of its operating range of frequencies (8.2 - 12.4 GHz [16]), and because of the precedent of its use as a test case for UTLM;

[3] See Appendix A for full script

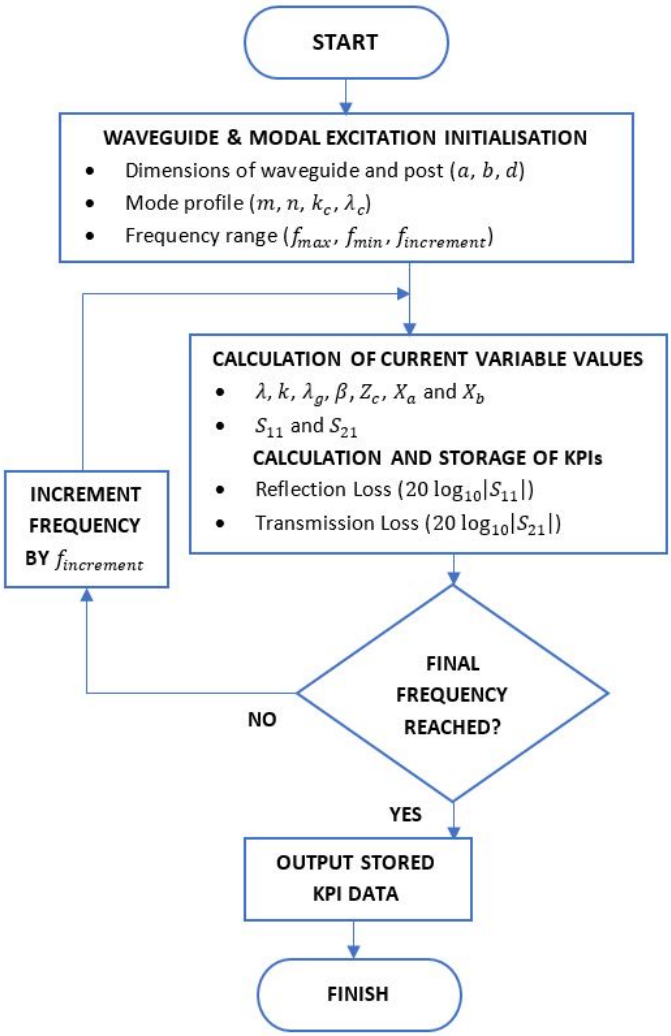

Figure 2.4: Flowchart of MATLAB™implementation of waveguide with inductive post [11]

- The post diameter, d, was selected to be 0.2286mm, obeying the recommended ratio of $\frac{d}{a} \leq 0.02$ for analytical models of inductive posts to be valid [17];
- The fundamental TE mode, TE_{10}, was selected as the excitation - i.e. with $m = 1$ and $n = 0$, hence the notation TE_{10}. This is the most common mode of excitation in a microwave system [11];
- The frequency range was set to be from the cutoff frequency of the TE_{10} in a WG16 waveguide - 6.5GHz - to the upper limit of the frequencies of interest when modelling a SSRP - 40GHz - with an increment of 0.5GHz.

The S_{11} and S_{21} characteristics yielded by the model above the cutoff frequency are shown in Figure 2.5, where it can be observed that the post exhibits relatively simple resonant behaviour at 19.5GHz and 32.5GHz, where transmission is maximum and reflection is minimum. In a SSRP the response is more complex, but using this model initially facilitates a clearer observation of the differences between the outputs of different modelling techniques.

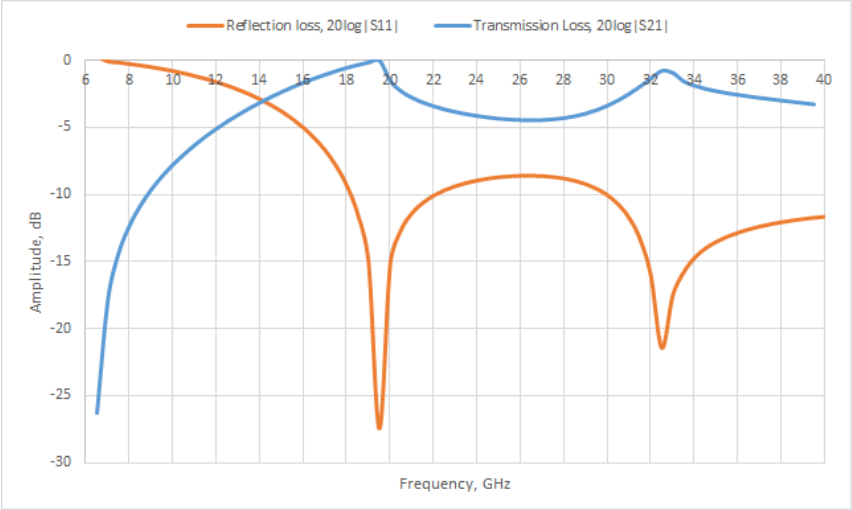

Figure 2.5: Transmission and reflection loss characteristics of the MATLAB™waveguide-post model

In Figure 2.6, the main plot shows the variation of two weighted reactance parameters with the ratio $\frac{d}{a}$, taken from [14]. By calculating $\frac{X_a}{Z_c}\frac{\lambda_g}{2a}$ for $\frac{\lambda}{a} = 1.0$ in the MATLAB™model, plotting against $\frac{d}{a}$ and overlaying the plot (red trace) on the original Marcuvitz figure [14], it can be observed that for $\frac{\lambda}{a} = 2.0$ the two traces are practically identical. This indicates that the model was replicated correctly from the original text.

Having established the reliability of the analytical results in Figure 2.5, this KPI data is then used as the benchmark against which the results of numerical modelling techniques are compared in later chapters.

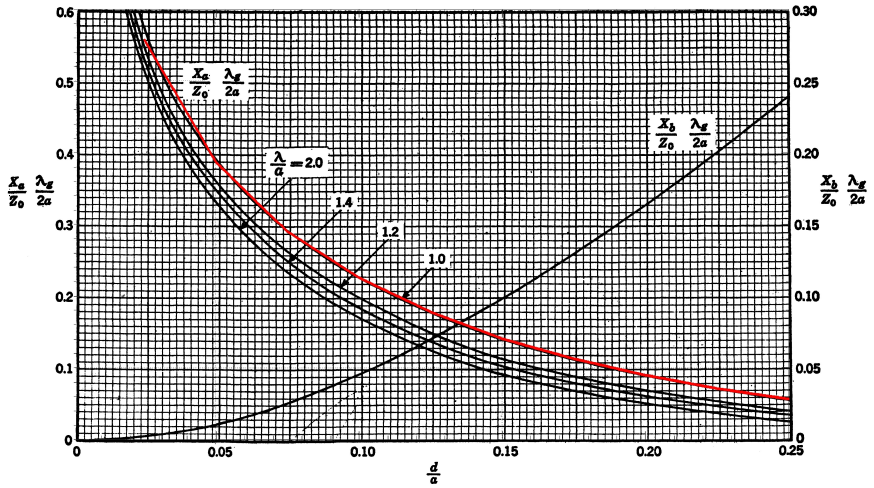

Figure 2.6: Validation of the analytical waveguide-post model by overlaying the MATLAB™plot of $\frac{X_a}{Z_c}\frac{\lambda_g}{2a}$ (red trace) on Marcuvitz original [14]

2.3 Chapter Summary

In this chapter, the theory of EM propagation in waveguides has been described and the application of transmission line theory to modelling this type of propagation has been explained. An approximation of a SSRP diode stage - a waveguide discontinuity in the form of an inductive post - has been selected and an analytical model of this arrangement has been identified, implemented and validated. Although this analytical model is an approximate FD method and cannot be said to represent the behaviour of the waveguide-post configuration comprehensively, it exhibits some expected behaviour that will provide a useful barometer for later numerical results.

In the following chapter, numerical modelling techniques are introduced as an alternative approach to modelling SSRPs.

Chapter 3

Literature Review of Numerical Modelling Techniques

In this chapter, numerical modelling in electromagnetics (EM) is introduced as a concept, different types of traditional numerical techniques are outlined along with their benefits and drawbacks, and rationale behind the choice of two advanced techniques, the Unstructured Transmission Line Modelling (UTLM) method and Boundary Element Method (BEM), is explained in relation to the application of solid state receiver protector (SSRP) modelling.

3.1 Introduction to Numerical Techniques

In all electromagnetic modelling problems, the objective is to find the unique solution to Maxwell's equations, which describe electric and magnetic field scattering behaviour in any given space; these are shown below. [9]:

$$\nabla \times \mathbf{E} = -\mathbf{M} - j\omega\mu\mathbf{H}, \tag{3.1}$$

$$\nabla \times \mathbf{H} = \mathbf{J} + j\omega\epsilon\mathbf{E}, \tag{3.2}$$

$$\nabla \cdot \mathbf{D} = q_e, \tag{3.3}$$

$$\nabla \cdot \mathbf{B} = q_m, \tag{3.4}$$

where $\mathbf{E}$ is the electric field intensity, $\mathbf{M}$ is the magnetic current density, ω is the angular frequency, μ is the magnetic permeability, $\mathbf{H}$ is the magnetic field intensity, $\mathbf{J}$ is the electric current density, ϵ is the electric permittivity, $\mathbf{D}$ is the electric flux density, q_e is the electric charge, $\mathbf{B}$ is the magnetic flux density and q_m is the magnetic charge[1].

Solving these equations analytically (i.e. determining the exact mathematical solution) is possible for problem spaces that are geometrically (and therefore electrically) simple, such as the waveguide-post combination modelled in Chapter 2. However, deriving exact solutions for propagation through the more complex geometrical features of a SSRP is impractical. Therefore, modelling complex structures is generally achieved using *numerical* techniques which work by discretising all or part of the problem space into a mesh (i.e. a network of adjacent geometrical cells representing the geometry), setting certain boundary conditions and excitations and applying a chosen algorithm to solve the resulting matrix of equations.

Numerical techniques are categorised using two main criteria:

- **Type of domain: Time Domain (TD) vs. Frequency Domain (FD) techniques** - this criterion distinguishes between the dependent variables in which domain Maxwell's equations are solved. TD methods are best applied to modelling broadband, transient and non-linear problems, whereas FD methods are more appropriate when modelling steady-state, narrowband problems [13, 18, 19];
- **Type of operator: Differential Equation (DE) vs. Integral Equation (IE) techniques** - this criterion indicates the type of operator employed when solving Maxwell's equations. Using a differential operator (such as in the configuration of equations 3.1 - 3.4 requires the equations to be solved at every point in the volume of the problem space, which is computationally intensive but in principle allows for

[1] In practice, $\mathbf{M}$ and q_m do not physically exist and can be neglected - see section 5.1.1 in Chapter 5

effect of complex geometrical features to be captured. In contrast, using an integral operator requires for the equations only to be solved on the surface of the problem space, which is more efficient computationally but cannot capture complex geometries or inhomogeneous materials [13, 18, 19];

In the next section, some popular techniques that fall into different combinations of these categories are outlined.

3.2 Common Numerical Techniques

3.2.1 Finite Difference Time Domain (FDTD) Method

The FDTD method is a TD-DE technique, in which Maxwell's equations are solved using the leapfrog method - that is, by determining the electric field and magnetic field components alternately at successive time steps [12, 13]. This technique is employed by the transient solver of the popular CST Microwave Studio™software package [20], which is the package used in Chapter 6 to provide a numerical simulation of the waveguide-post geometry, against which the UTLM results are compared.

The main limitation of this technique is that the geometry of the problem is described using cuboidal mesh cells; consequently, curved and/or small-scale regions cannot accurately represented, leading to 'staircase' approximations that can cause errors in simulation [21, 22].

In addition, the method does not exhibit stable convergence, which limits its efficiency as a TD method [23, 24]. Plus, the total simulation time is increased by the requirement to place a particular type of absorbing boundary condition (ABC) known as a perfectly matched layer (PML) to avoid the introduction of reflections at the edge of the problem space [12, 13, 18, 25].

3.2.2 Transmission Line Modelling (TLM)

The TLM method is an alternative TD technique. In structured TLM (as opposed to unstructured TLM (UTLM), which is considered later), a cuboidal volume mesh of the problem space in question is generated, and each discrete mesh cell is modelled as an equivalent network of transmission lines. As the transient response of these transmission lines can be represented by equations analogous to Maxwell's equations, the electric and magnetic fields propagating through the problem space can be mapped onto the equivalent voltages and currents in the transmission line network [10, 26].

The drawbacks of TLM are that similar to FDTD, in that it requires ABCs and suffers from the same staircasing errors. Unlike FDTD however, it does have guaranteed stability, which increases its efficiency and accuracy [23, 25].

3.2.3 Finite Element Method (FEM)

In contrast, the FEM is a family of IE techniques, which are generally formulated in the FD. The FEM employs a weighted residual method, in which the unknowns in Maxwell's equations are replaced with weighted basis functions and then integrated over the discretised surface of the problem space [12, 22]. A popular software package that uses this type of FD technique is Ansoft HFSS ™[27]; it is using this package that early numerical models of SSRPs were created [9].

Although the sparsely populated matrix that results from a purely surface based formulation is computationally efficient, the matrix may grow larger when the problem's boundary is truncated [22, 28].

3.3 Proposed Numerical Techniques

3.3.1 Unstructured Transmission Line Modelling (UTLM)

The UTLM method is an advanced form of the structured TLM method. UTLM removes the limitations associated with a structured cuboidal mesh by implementing a mesh that:

- consists of tetrahedral cells, which can better capture the complex geometrical features inherent in receiver protector designs [21];
- is unstructured (i.e. that can adapt the size of its cells to suit the dimensions of the geometry), thereby having the ability to capture multiscale geometries accurately [21];
- has the option to rationalise tetrahedral cells into cuboids in more uniform areas of the mesh, minimising the computational resources required to model these areas of the structure [21].

Additionally, UTLM has the ability to combine low quality cells into clusters, which increases computational efficiency by allowing a larger time step to be used, whilst maintaining the unconditional stability of structured TLM [26].

These capabilities in high modelling resolution and efficiency combine to make UTLM a superior candidate for modelling SSRPs compared to other TD techniques.

3.3.2 Boundary Element Method (BEM)

The BEM is a FD technique that belongs to the FEM family, whose primary capability is modelling the scattering of EM fields incident upon Perfect Electric Conductor (PEC) objects. This is achieved by discretising and solving the boundary integral equations - that is, the Electric Field Integral Equation (EFIE) and Magnetic Field Integral Equation

(MFIE) - which relate the scattered EM fields to the currents induced on the surface of a PEC object by the incident field [24, 29]. The BEM approximates the unknowns of the EFIE and MFIE as a product of the weighted basis functions and a known Green's function [12].

The fact that the Green's function for 3D EM propagation is known makes BEM particularly powerful: it means that the volume of a region of the problem space does not need to be meshed (and therefore not simulated) in order to compute the scattered field there. This reduces the order of the problem, saving time and computational resources compared to UTLM and other TD methods, which require a mesh of the entire problem space volume to run any simulation. In the context of SSRPs this is especially useful, as it allows the far-field response - a key performance indicator (KPI) in the field of RADAR communications - to be determined efficiently.

Another advantage of the Boundary Element method is that in the weighted residuals that characterise it as a FEM technique, the basis functions themselves are used as the testing functions - known as Galerkin's method [12]. This enables the boundary conditions to be enforced at all points in the problem space, and thus allows a more comprehensive solution to be found [18, 30].

Three drawbacks of the BEM are: firstly, that in its traditional FD formulation it does not provide the broadband response needed to model higher frequencies, and by doing so improve upon existing SSRP models (however, TD formulations have been developed [29]). Secondly, the boundary truncation (mentioned when introducing FEM) occurs in the BEM, increasing the size of the matrix and hence the simulation time [22, 28]. Thirdly, BEM can only model homogeneous materials such as free space, not inhomogeneous ones such as carbon fibre composites (CFCs) [18].

3.4 Chapter Summary

In this chapter, numerical modelling as a concept has been introduced, some existing types of popular numerical techniques have been considered and their drawbacks highlighted. Two alternative, novel techniques have been presented, and the relative merits of each with regard to the application of SSRP modelling have been outlined: UTLM, with its strengths of stability and high-resolution meshes, and BEM with its strengths of efficiency and comprehensive far-field computation.

In the following two chapters, the existing implementations of both BEM and UTLM are detailed, and the progress in using each method to produce waveguide-post models comparable to the analytical model of Chapter 2 is outlined.

Chapter 4

Unstructured Transmission Line Modelling (UTLM)

In this chapter, the basic principles of the UTLM theory are explained, from its foundations in the structured TLM method to the implementation of unstructured meshes. The implementation of UTLM currently under development at the University of Nottingham is then outlined, and in doing so its application to the modelling of the waveguide-post geometry described in Chapter 2 is detailed.

4.1 UTLM Theory

Both TLM and UTLM are based upon the analogous relationship between the propagation of EM fields through a medium and the propagation of voltages and currents along a network of transmission lines, as outlined in the previous two chapters. This section details how this principle is applied to simulation through both structured and unstructured meshes.

4.1.1 1D TLM: The Scatter-Connect Principle

Firstly, a one-dimensional (1D) TLM network is considered. This can be constructed by dividing a 1D problem space into n nodes separated by $n-1$ segments, each of a discrete length, Δx. Each segment consists of a section of transmission line of the form shown in Figure 4.1.

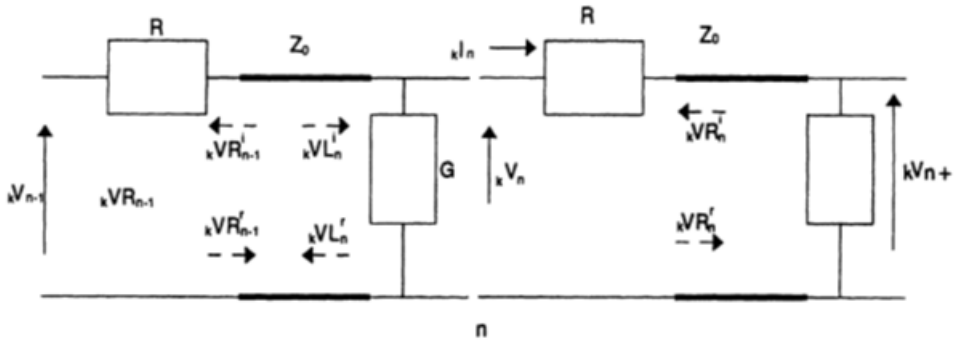

Figure 4.1: Diagram of a single 1D transmission line segment, with labels indicating impedances and incident and reflected voltages and currents [13]

The properties governing the characteristics of the segment are:

- Series resistance, R;
- Shunt admittance, G;
- Series inductance, L;
- Shunt capacitance, C,

where L and C are equal to L_d and C_d respectively, and are combined in Figure 4.1 in the characteristic impedance Z_0, as defined in equation 2.20.

The voltage at node n at a given timestep k ($_kV_n$) is calculated as a superposition of the incident voltages from the adjacent nodes on the left ($_kVL^i_n$) and right ($_kVR^i_n$) [13].

$_kVL^i_n$ is equal to the voltage scattered to the right from node $n-1$ at the previous timestep. Similarly, $_kVR^i_n$ is equal to the voltage scattered to the left from node $n+1$ at the previous timestep [13, 23].

Therefore, ${}_kV_n$ is calculated by reducing the incident voltages to their Thevenin equivalents and applying the parallel generator theorem (or Millman's theorem) [13]. The equivalent circuit that results from applying this theorem is shown in Figure 4.2.

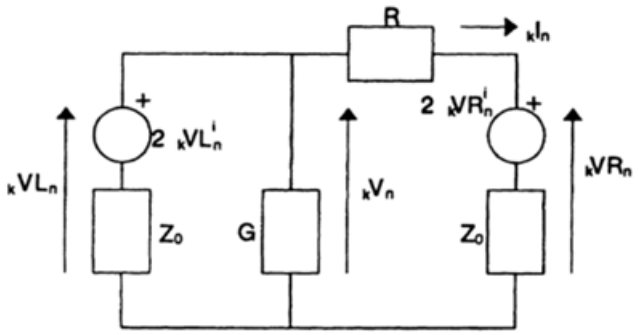

Figure 4.2: Thevenin equivalent circuit of a single TLM segment after applying Millman's theorem [13]

From Figure 4.2, the formula for ${}_kV_n$ can be derived:

$$ {}_kV_n = \frac{\frac{2_kVL_n^i}{Z_0} + \frac{2_kVR_n^i}{Z_0+R}}{\frac{1}{Z_0} + \frac{1}{Z_0+R} + G}. \tag{4.1} $$

By substituting ${}_kV_n$ into Figure 4.2, the voltages scattered to the left (${}_kVL_n^r$) and to the right (${}_kVR_n^r$) at the current timestep can also be computed:

$$ {}_kVL_n^r = {}_k\,VL_n - {}_k\,VL_n^i, \tag{4.2} $$

$$ {}_kVR_n^r = {}_k\,VR_n - {}_k\,VR_n^i, \tag{4.3} $$

The computation of the scattering response of node n is then complete, and when replicated for all other nodes, the response of the transmission line is complete for the current timestep. By increasing the number of timesteps, the transient scattering response of the line can be computed. This is known as the scatter-connect algorithm, and is the fundamental iterative method by which voltage and current propagation are computed in TLM.

4.1.2 2D and 3D TLM: Modelling TE, TM and TEM Propagation

The scatter-connect method can be extended to two-dimensional (2D) space by using the concept of nodes to model TE and TM modes of propagation.

In a TE mode, three out of six EM field components feature: E_x, E_y and H_z. Through the analogy between voltages and electric fields discussed in [13, 22, 23], the propagation of a TE mode can be represented by an arrangement of four transmission lines known as a series node, which is depicted in Figure 4.3(a).

Likewise, the remaining three EM field components that constitute the TM mode - H_x, H_y and E_z - are represented by the shunt node transmission line arrangement in Figure 4.3(b).

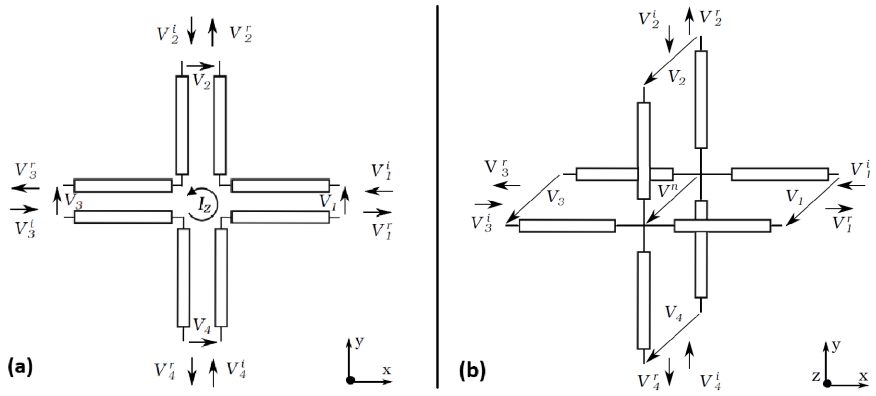

Figure 4.3: 2D TLM of TE and TM propagation as (a) series node and (b) shunt node configurations, respectively [22]

This concept can be further extended to three-dimensional (3D) space by combining the series and shunt nodes into a single Symmetrical Condensed Node (SCN) shown in Figure 4.4.

By combining all six field components, the modelling of TEM propagation is also made possible, as explained in detail in [13]. Whilst TEM modes do not propagate through waveguide devices such as SSRPs, the implementation of this node is nevertheless of use in the SSRP application as it facilitates the modelling of the propagation of different types of excitation through the problem space *prior* to reaching the waveguide, thus allowing a

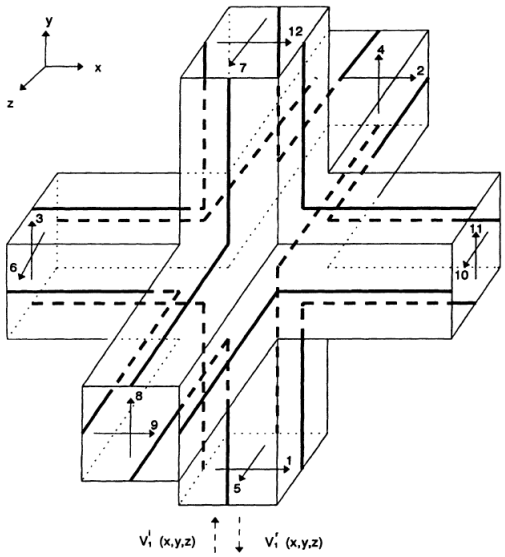

Figure 4.4: Symmetrical Condensed Node used in structured 3D TLM [13]

wide variety of input signals into a SSRP to be mimicked.

4.1.3 Unstructured Meshes

The tessellation of the SCN results in a structured mesh of cuboids. Now, despite the advent of adaptive structured meshes, in which the dimensions of the cuboids are variable [20], it is ultimately not possible to accurately capture even the simplest of curved geometries with purely cuboidal cells, let alone the complex geometries of SSRP components like the PIN diode capsule.

However, by introducing a *tetrahedral* unit mesh cell and combining it with an adaptive meshing capability as the UTLM method does, both curved and multiscale geometrical characteristics can be more accurately captured. An example of a single cell of such a mesh is illustrated in Figure 4.5.

The formulae on the right of Figure 4.5 show that the magnitudes of the equivalent voltage (V_i) and current (I_i) propagating along each of the four transmission lines connecting the cell to the rest of the network (also known as link lines) are dependent upon:

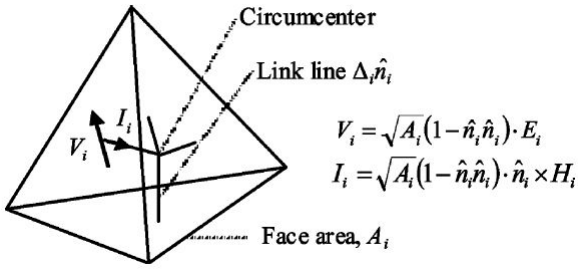

Figure 4.5: Diagram of a single tetrahedral mesh cell, showing the derivation of the equivalent transmission line voltages and currents [26]

- The area of each cell face (A_i);
- The incident electric field ((E_i)) (in the case of (V_i));
- The incident magnetic field ((H_i)) (in the case of (I_i)).

The fact that the modelled response (i.e. V_i and I_i) of the geometrical portion contained within the cell's volume is proportional to both the cell dimensions (A_i) is crucial to the power of the unstructured mesh. It means that if smaller cells are required to model a particularly small or complex geometrical feature, the proportional response of this small feature to the propagating EM wave can be measured [26].

Furthermore, the cell shown is that of a Delaunay mesh - that is one in which each of the link lines emanate from the circumcentre of the cell, thus maximising the timestep and minimising the total simulation time. [26].

Having reviewed the theory behind the UTLM meshing capability, its application to the problem of SSRP modelling using the UTLM implementation developed at the University of Nottingham is now described.

4.2 Existing Implementation and Application to Waveguide-Post

At the George Green Institute for Electromagnetics Research (GGIEMR), UTLM model creation has been implemented in a three stage workflow:

- Geometry creation;
- Mesh generation;
- Solver setup and simulation.

The steps followed at each of these stages to implement a waveguide-post model are now detailed.

4.2.1 Geometry Creation

Using a library of predefined C++ functions, the two geometrical surfaces that constitute the waveguide-post arrangement (i.e. a cuboid and a cylinder) were created and a Boolean operation was performed on them to replicate the post going 'through' the waveguide (i.e. the cylinder through the cuboid). The values used for waveguide width, a, waveguide height, b, position of post, x, and diameter of post, d, were the same as those used for the MATLAB™model created in Chapter 2; additionally a waveguide length, l was defined to be 100mm. The code written to achieve this is listed in Appendix B, and compiling this code yielded the 3D geometry shown in Figure 4.6.

4.2.2 Mesh Generation

Next, a tetrahedral volume mesh of the structure was created, with cuboidal gridding reintroduced in uniform areas of the mesh. To minimise dispersion errors in the mesh, a maximum length of a side of a cubic cell, Δx, was chosen to meet the condition $\Delta x \leq \frac{\lambda_0}{10}$,

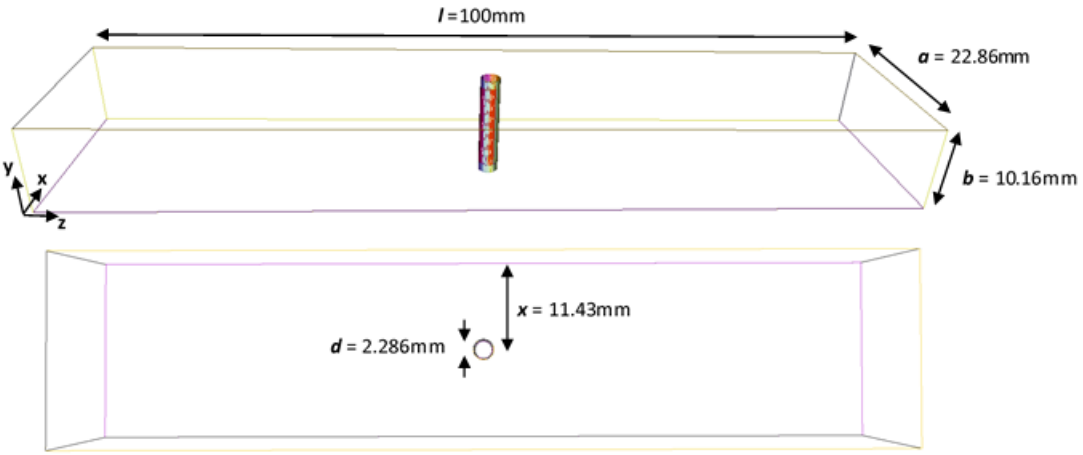

Figure 4.6: Side-on and bird's eye views (top and bottom respectively) of 3D waveguide-post geometry created in UTLM, with dimensions labelled

where λ_0 is the wavelength at the maximum frequency of interest [31]. As this frequency is 40GHz, $\lambda_0 = 7.5$mm and Δx was set to be 0.75mm.

On the left of Figure 4.7, a cross-section at the centre of the volume mesh is shown. It can be observed that in the uniform areas of the cross section (i.e. the areas of free space on either side of the post), the tetrahedral mesh cells have been rationalised into cubes. Meanwhile, on the right of Figure 4.7 is a section of the surface mesh of the waveguide-post, and the inset image shows that at the junction between the curved post and the flat waveguide wall, the unstructured and tetrahedral properties of the mesh combine to produce an accurate discretisation of that more complex part of the geometry, as discussed previously in sections 3.3.1 and 4.1.3.

4.2.3 Solver Setup and Simulation

The final stages of pre-processing required were the setup of several simulation parameters concerning boundary conditions, modal excitation, clustering and the timestep.

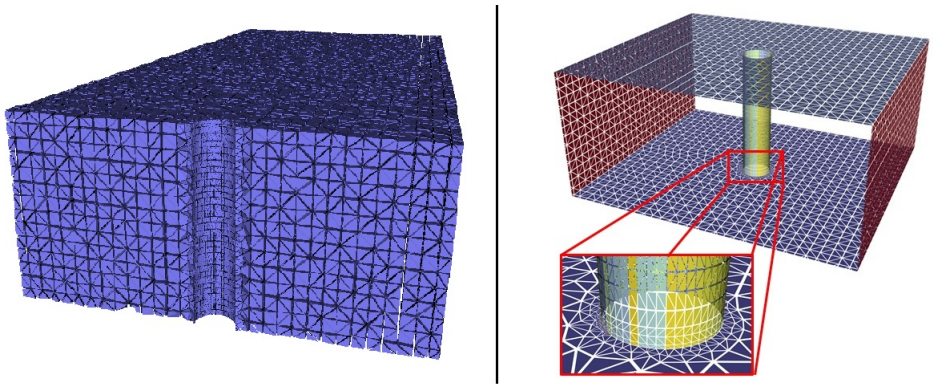

Figure 4.7: UTLM tetrahedral, gridded volume mesh and surface mesh (left and right respectively) of the WG16 waveguide-post geometry

Boundary Conditions

As discussed in section 3.2, TLM (and, by extension, UTLM) requires the setting of absorbing boundary conditions (ABCs) at any boundaries of the volume mesh at which the modelling of non-reflective behaviour is desired. Therefore, for the waveguide-post mesh, the cells at both the input and output of the waveguide were set to have a reflection coefficient of 0 (indicating the seamless propagation of EM fields across that boundary, as would be expected at the input and output of a SSRP. Meanwhile, the reflection coefficient assigned to the waveguide walls and the surface of the inductive post was set to be -1, indicating the full reflection of EM fields expected from these metallic surfaces.

TE_{10} Modal Excitation

In order to set up the fundamental TE modal excitation (TE_{10}) on the 2D collection of mesh cell faces at the input to the waveguide shown in Figure 4.8, the following parameters were set:

- **Modal frequency**, f_{modal} - that is, the frequency at which the mode is excited. In order to propagate along the waveguide, this has to be comfortably above the cutoff frequency of the TE_{10} mode, $f_{c_{10}} = 6.56$GHz, therefore a value of **14GHz** was chosen;

- **Effective refractive index**, η_{eff} - that is the proportion by which the speed of EM propagation changes when passing from one medium into another [10], at f_{modal}. This was calculated via the formula [32]:

$$\eta_{eff} = \frac{c\beta}{2\pi f_{modal}}, \tag{4.4}$$

  where c is the speed of EM propagation in a vacuum, β is the propagation constant (≈ 259 for the TE_{10} mode)). This produces a value for η_{eff} of **0.88**.

- **Slice thickness**, Δz - the length of waveguide extruded from the collection of mesh cell faces in order to compute the mode's field distribution [32]. To obey the UTLM condition [32]

$$\Delta z \leq \frac{0.01c}{2\pi f_{modal}\eta_{eff}}, \tag{4.5}$$

  a value of **0.03mm** was chosen.

In Figure 4.8, the E_y field distribution yielded across the plane of mesh cells at the input to the waveguide by using these parameters is shown. The scale bar on the left-hand side of the figure indicates that the region at the centre has a higher magnitude of field strength than toward the left- and right-hand sides; by observing the similarities with the TE_{10} sketch in Figure 2.1 in Chapter 2, it can be seen that parameters produce the correct excitation.

Timestep and Clustering Threshold

The final parameters to set centre around achieving the optimum length of time for the whole simulation.

The UTLM timestep is the time taken for the voltage scattering and reflection to take place through a single UTLM mesh cell. This is the same for all cells in the mesh. Thus, if a *large* number of *small* mesh cells exists, the total time for simulation will be very long.

This motivated the introduction of cell clustering - the rationalisation of small mesh

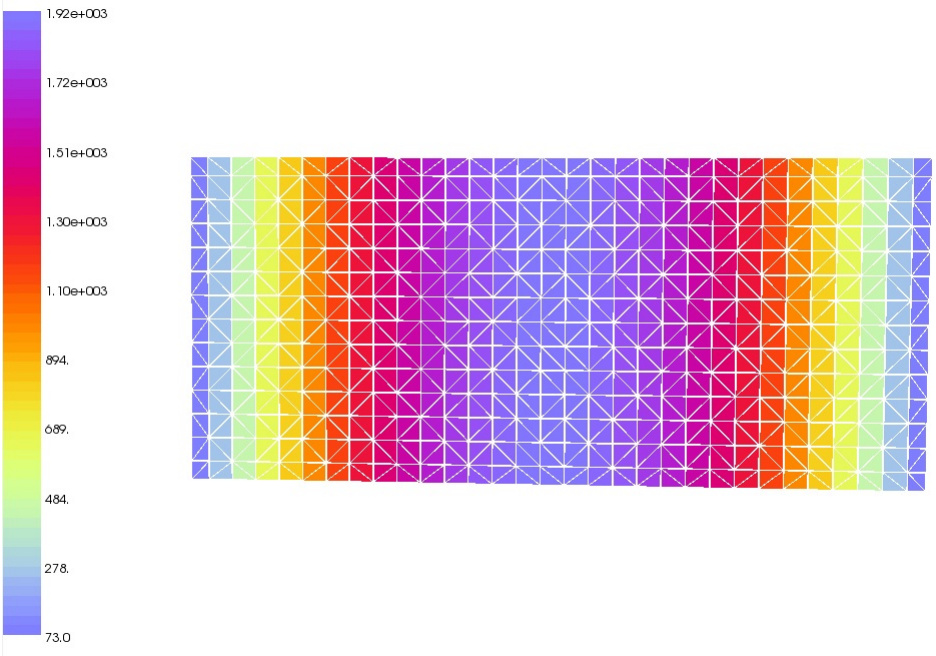

Figure 4.8: E_y field distribution of the TE_{10} modal excitation input to the UTLM simulation of the waveguide-post mesh, where the scalebar on the left indicates magnitude of E_y

cells (i.e. those with link lines below a certain length) into larger cells which were closer in size to the larger original mesh cells [26], thus allowing the use of a larger timestep to be justified. To achieve the recommended level of clustering (below 200 cells per cluster, in order to conserve computational efficiency [33]), the link line length below which cells would be clustered together - i.e. the cluster threshold, s - chosen for this particular mesh was $9.5\mu\mathbf{m}$.

The recommended timestep, x is two times the clustering threshold [26], and so the value used for this particular simulation was $19\mu\mathbf{m}$.

4.3 Chapter Summary

In this chapter the theory of how the UTLM method models EM propagation has been described in detail and the application of its most recent implementation at the GGIEMR to the present problem - the modelling of a waveguide-post geometry as an approximation of an SSRP - has been outlined.

The results of this simulation are presented and discussed in Chapter 6. Meanwhile in the next chapter, the theory, implementation and attempted application of the Boundary Element Method (BEM) is discussed.

Chapter 5

Boundary Element Method (BEM)

In this chapter, the basic principles of the BEM are explained, from the derivation of the electric field integral equation (EFIE) to the discretisation of the surface current using basis functions and the formulation of the solution matrix. Then, the implementation of the BEM currently under development at the University of Nottingham (primarily) is outlined, the progress in applying it to the SSRP modelling problem is summarised, and the capabilities added to the method over the course of the project, and their benefits, are described.

5.1 BEM Theory

As discussed in Chapter 3, the BEM finds the unique solution to Maxwell's equations by formulating a relationship between the scattered field and incident field in terms of the current flowing across a discretised surface, and then by substituing in an approximation of the current using the weighted residual method.

Firstly, the formulation of the integral equations is explained.

5.1.1 Derivation of the Electric Field Integral Equation (EFIE)

Both the EFIE and magnetic field integral equation (MFIE) are derived from the inhomogeneous solution to Maxwell's equations - that is, the solution given the presence of a current source. This derivation is now summarised, taking the EFIE as an example.

To begin with, Maxwell's equations can be simplified by recognising that magnetic charge (and therefore current) do not physically exist, and thus q_m and $\mathbf{M}$ can be neglected:

$$\nabla \times \mathbf{E} = -j\omega\mu\mathbf{H}, \tag{5.1}$$

$$\nabla \times \mathbf{H} = \mathbf{J} + j\omega\epsilon\mathbf{E}, \tag{5.2}$$

$$\nabla \cdot \mathbf{D} = q_e, \tag{5.3}$$

$$\nabla \cdot \mathbf{B} = 0, \tag{5.4}$$

By applying the Lorenz gauge condition and parameterising $\mathbf{E}$ and $\mathbf{H}$ with the magnetic vector potential, $\mathbf{A}$, Equations 5.1 - 5.4 can be manipulated to evaluate $\mathbf{E}$ and $\mathbf{H}$ as follows [12]:

$$\mathbf{E} = -j\omega\mathbf{A} + \frac{1}{j\omega\mu\epsilon}\nabla(\nabla \cdot \mathbf{A}), \tag{5.5}$$

$$\mathbf{H} = \frac{1}{\mu}\nabla \times \mathbf{A}, \tag{5.6}$$

By employing the constitutive relations and relevant vector identities, Equations 5.5 and 5.6 can be reduced to the following inhomogeneous vector Helmholtz equation [12]:

$$\nabla^2\mathbf{A} + k^2\mathbf{A} = -\mu\mathbf{J}. \tag{5.7}$$

Solving Equation 5.7 for $\mathbf{A}$ by means of an appropriate Green's function, G, yields:

$$\mathbf{A}(\mathbf{r}) = \mu \int \int \int_V \mathbf{J}(\mathbf{r}) G(\mathbf{r}, \mathbf{r}') dV', \tag{5.8}$$

where $\mathbf{r}$ and $\mathbf{r}'$ are the Cartesian position vectors of the source and observation points respectively, and dV' is a discrete element of the volume of the problem.

Substituting this expression into Equation 5.5 yields:

$$\mathbf{E}^s(\mathbf{r}) = -j\omega\mu \int \int_S G(\mathbf{r}, \mathbf{r}') \left[\mathbf{J}(\mathbf{r}') + \frac{1}{k^2} \nabla' \left(\nabla' \cdot \mathbf{J}(\mathbf{r}') \right) \right] d\mathbf{r}', \tag{5.9}$$

where the quantity being calculated, $\mathbf{E}^s(\mathbf{r})$, is the electric field scattered by the induced current on the surface, S, by current $\mathbf{J}$.

Owing to the law of energy conservation, the tangential electric field scattered by a surface is equal and opposite to that which is incident, as captured in the following boundary condition [30, 34]:

$$\hat{\mathbf{n}}(\mathbf{r}) \times \mathbf{E}^s(\mathbf{r}) = -\hat{\mathbf{n}}(\mathbf{r}) \times \mathbf{E}^i(\mathbf{r}), \tag{5.10}$$

where $\hat{\mathbf{n}}$ is the surface normal and $\mathbf{E}^i(r)$ represents the incident tangential field. Substituting the expression in 5.9 for $\mathbf{E}^s(r)$ and rearranging yields:

$$-\frac{j}{\omega\mu} \hat{\mathbf{n}}(\mathbf{r}) \times \mathbf{E}^i(\mathbf{r}) = \hat{\mathbf{n}}(\mathbf{r}) \times \int \int_S G(\mathbf{r}, \mathbf{r}') \left[\mathbf{J}(\mathbf{r}') + \frac{1}{k^2} \nabla' \left(\nabla' \cdot \mathbf{J}(\mathbf{r}') \right) \right] d\mathbf{r}'. \tag{5.11}$$

Equation 5.11 constitutes the aforementioned EFIE; the next stage is to define an approximation of $\mathbf{J}(\mathbf{r}')$.

5.1.2 Basis Functions

Deriving an approximation of surface current firstly involves reconsidering a factorised version of the inhomogeneous vector Helmholtz equation (5.7):

$$(\nabla^2 + k^2)\mathbf{A}(\mathbf{r}) = -\mu\mathbf{J}(\mathbf{r}), \tag{5.12}$$

which is a specific case of the more general inhomogeneous differential equation

$$L(f) = g, \tag{5.13}$$

where the operator L (i.e. $\nabla^2 + k^2$) acting on the unknown function f (i.e. $\mathbf{A}(\mathbf{r})$) should equal the excitation function g (i.e. $\mathbf{J}(\mathbf{r})$) [30].

Using this formulation, a solution for $\mathbf{J}$ is assumed to be approximated as a sum of weighted basis functions, of the general form

$$\mathbf{J}(\mathbf{r}) = \sum_{n=1}^{N} a_n \mathbf{f}_n(\mathbf{r}), \tag{5.14}$$

where N is the number of mesh cells, a_n are unknown coefficients and $\mathbf{f}_n(\mathbf{r})$ are the aforementioned basis functions.

Basis functions represent the behaviour of a vector function over successive elements (or groups of elements) in a discretised space. They are chosen to best represent the characteristics of the particular function in question. In EM modelling, the basis functions of choice are the Raviart-Thomas (RT) basis functions (also referred to as the Rao-Wilton-Glisson basis functions): a pair of functions describing vector function behaviour across two adjacent triangles which share a common edge [12, 30], such as those shown in Figure 5.1(a).

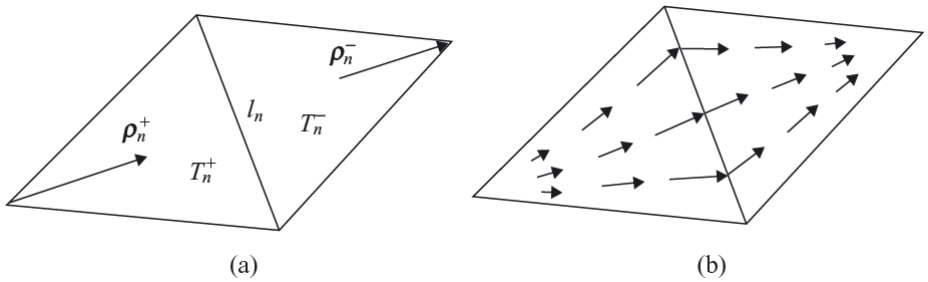

Figure 5.1: A pair of adjacent triangles, sharing a common edge (a), and the vector function behaviour modelled when the Raviart Thomas basis function is applied (b) [12]

The RT functions themselves, which are used to produce the modelled vector behaviour shown in Figure 5.1(b), are as follows:

$$\mathbf{f}_n(\mathbf{r}) = \begin{cases} \frac{l_n}{2A_n^+}\boldsymbol{\rho}_n^+(\mathbf{r}) & \mathbf{r} \in T_n^+ \\ \frac{l_n}{2A_n^-}\boldsymbol{\rho}_n^-(\mathbf{r}) & \mathbf{r} \in T_n^-, \end{cases} \tag{5.15}$$

where l_n is the length of the common edge, T_n^+ and T_n^- are each of the triangles, A_n^+ and A_n^- are the areas of each of the triangles, and $\boldsymbol{\rho}_n^+$ and $\boldsymbol{\rho}_n^-$ are the vectors shown in Figure 5.1(a).

Substituting the approximation for $\mathbf{J}(\mathbf{r})$ (5.14) into the EFIE (5.11) gives:

$$-\frac{j}{\omega\mu}\hat{\mathbf{n}}(\mathbf{r}) \times \mathbf{E}^i(\mathbf{r}) = \hat{\mathbf{n}}(\mathbf{r}) \times \int\int_S G(\mathbf{r},\mathbf{r}') \sum_{n=1}^{N} a_n \mathbf{f}_n(\mathbf{r}') \left[1 + \frac{1}{k^2}\nabla'\nabla'\right] d\mathbf{r}'. \tag{5.16}$$

When paired with the basis functions definitions (in 5.15), all that remains to complete the solution is to determine the unknown coefficients a_n.

In order to appreciate how equation 5.16 is solved for a_n over an entire mesh - thus obtaining the complete solution of the scattered field - the matrix form of the equation is considered next.

5.1.3 Solving EFIE Matrix for Unknown Coefficients

In its most condensed form, the matrix form of equation 5.16 can be expressed as

$$\mathbf{Za} = \mathbf{b}, \tag{5.17}$$

or, in expanded form,

$$\begin{bmatrix} z_{11} & z_{12} & z_{13} & \cdots & z_{1N} \\ z_{21} & z_{22} & z_{23} & \cdots & z_{2N} \\ z_{31} & z_{32} & z_{33} & \cdots & z_{3N} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ z_{N1} & z_{N2} & z_{N3} & \cdots & z_{NN} \end{bmatrix} = \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_N \end{bmatrix} \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_N \end{bmatrix}, \tag{5.18}$$

where:

- $\mathbf{Z}$ is a $N \times N$ matrix in which the elements (z_{mn}) of each row (m) constitute the field components scattered by successive 'source' mesh cells (n) to a single 'observation' cell, with

$$z_{mn} = \int\int_{\mathbf{f}_m} \int\int_{\mathbf{f}_n} \left(\mathbf{f}_m(\mathbf{r}) \cdot \mathbf{f}_n(\mathbf{r}') - \frac{1}{k^2} \left[\nabla \cdot \mathbf{f}_m(\mathbf{r}) \right] \left[\nabla' \cdot \mathbf{f}_n(\mathbf{r}') \right] \right) G(\mathbf{r}, \mathbf{r}') d\mathbf{r}' d\mathbf{r}, \tag{5.19}$$

  where $\mathbf{f}_m(\mathbf{r})$ are the testing functions, which as discussed in Chapter 3 are identical to the basis functions via Galerkin's method;

- $\mathbf{a}$ is a N-length vector in which each element (a_n) is the unknown weighting coefficient valid at successive 'source' cells;

- $\mathbf{b}$ is a N-length vector in which each element (b_m) is the total field incident upon successive 'observation' cells, with

$$b_m = -\frac{j}{\omega\mu} \int \int_{\mathbf{f}_m} \mathbf{f}_m(\mathbf{r}) \cdot \mathbf{E}^i(\mathbf{r}) d\mathbf{r}. \tag{5.20}$$

To obtain $\mathbf{a}$, each side of equation 5.17 is simply multiplied by $\mathbf{Z}^{-1}$.

Therefore, $\mathbf{Za}$ in its expanded form represents an 'exploded view' of every single scattered field component that contributes to the scattering response of the PEC object; by taking a unique observation cell on each row of $\mathbf{Z}$, calculating the contributory scattered field components and summing them accordingly, the total scattered field observed across the entire mesh is calculated.

5.2 Existing Implementation and Application to Waveguide-Post

The BEM has been implemented at the George Green Institute for Electromagnetics Research (GGIEMR) as an open-source package called the Boundary Element Analysis and Simulation Toolkit (BEAST), written in the Julia programming language [35]. In the following sections, each part of this implementation is outlined with respect to the application of the waveguide-post geometry, and new features that have been added to the package are highlighted.

5.2.1 Mesh Generation and Interface with UTLM

Structure capture in the BEAST package does not involve geometry definition and meshing, but instead the definition of a mesh directly or the import of an existing mesh. As the BEM is a integral equation (IE) numerical technique, this mesh is a 2D surface mesh defined using 3D Cartesian coordinates, i.e. of the kind presented on the right of Figure 4.7 in the previous chapter. The type of object into which the mesh is read consists of a list of mesh vertices (i.e. the x-, y- and z-coordinates that define each point of the mesh, plus a unique an ID number) and a list of the mesh faces (i.e. the ID numbers of

the vertices that comprise each face of the mesh).

Originally, two modes of mesh generation existed in the BEAST package: either defining the points and vertices of a mesh in a Julia function directly, or by parsing a MSH file output from the popular meshing tool GMSH [36].

Unfortunately, the former method is unsuitable for producing meshes of anything other than simple geometrical shapes, and the latter requires the extra step of either importing an existing industry-standard 3D geometry file (e.g. IGES or STEP files) or defining the geometry from scratch in the GMSH package. Given that a high-quality mesh of the waveguide-post geometry had already been produced in the form of the unstructured tetrahedral mesh with cuboidal gridding created in UTLM (as described in Chapter 4), it was decided that the optimum use of resources would be to import the UTLM surface mesh (contained in the proprietary TRI file) into the BEAST package.

However, it was found that a fundamental issue with importing the UTLM surface mesh (or, for that matter, any surface mesh generated from a 3D geometry) is that such meshes often include surfaces intended as *non-metallic* (and thus *non-reflective*) that the BEM cannot avoid treating as *reflective*, *metallic* surfaces. In the waveguide-post mesh, this problem manifested itself in the presence of surface mesh cells across the input and output of the waveguide. Although it was necessary to retain these cells in the UTLM method in order to define a modal excitation across them, in the BEM the presence of these cells would cause any excitation to be reflected, and their removal was essential for the propagation of any excitation through the waveguide to be possible.

The C++ script listed in Appendix C was written to enable the removal of these cells by reading in a UTLM TRI file, selecting a cubed region of cells and writing the vertices and faces of all mesh cells *except* these selected ones to an output file (thus removing the selected cells).

After running this script on the waveguide-post mesh by calling the function shown in Appendix D.2 from the example shown in Appendix D.1, and removing the panels of cells at the input and output of the waveguide, the modified mesh was imported into the

BEAST package as a generic object of vertices and faces, yielding the mesh shown on the left and top-right of Figure 5.2.

The top-right and bottom-right images of Figure 5.2 show the BEAST and UTLM meshes of the waveguide-post geometry, respectively. By comparing the regions boxed in red in each of these two images, the removal of the mesh cell faces at the ends of the waveguide in the BEAST mesh image can be clearly observed.

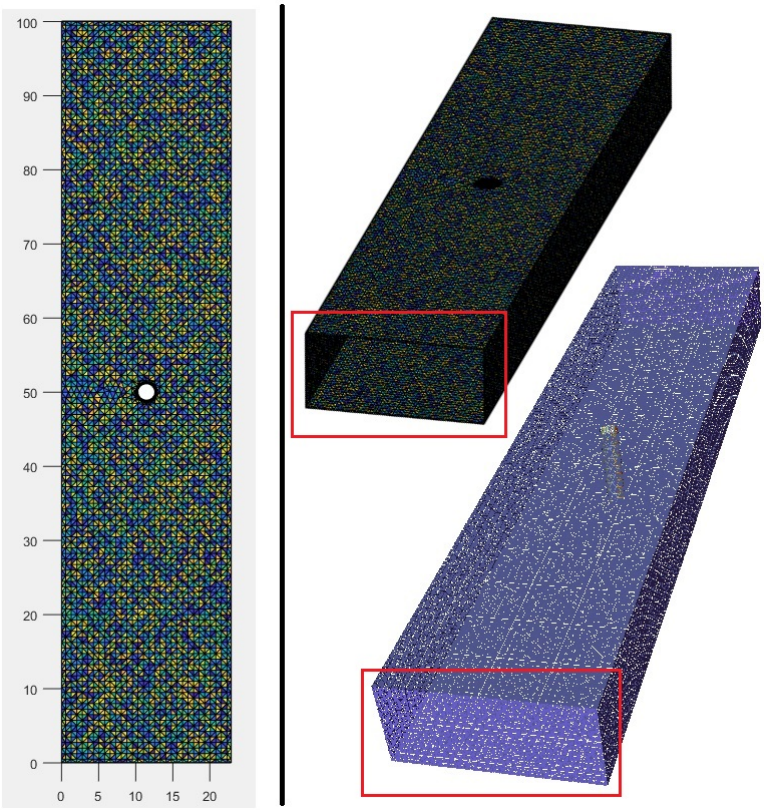

Figure 5.2: A bird's eye view (in the negative y-direction) of the waveguide-post surface mesh imported into BEAST (left - axes in mm), and a comparison of the BEAST surface mesh (top-right) and UTLM surface mesh (bottom-right)

Having successfully imported a mesh suitable for simulation using the BEM method, the setup and solving process for the rest of the BEM simulation is now briefly outlined.

5.2.2 Basis Functions, Excitations and Solving

The BEAST package computes the RT basis functions by:

- Dividing the mesh into pairs of cells which share a common edge;
- Assigning the necessary coefficients for computing the current distribution (for example the positive and negative orientation of cells shown in Figure 5.1), so that when the solver requires to access them they are available.

Following this, an excitation object is created based upon user-defined parameters; for a plane wave excitation (the most appropriate for use with BEM simulations), these are the polarisation, direction and wavenumber. Based upon this information, the EFIE is constructed and discretised to be of the form shown in equation 5.16.

Solving the double integrals in the EFIE equation matrix (i.e. computing each of the elements in the $\mathbf{Z}$ matrix in equation 5.18) is achieved by applying unique Gaussian quadrature rules to each double integral computation, and then multiplying each side of equation 5.17 by $\mathbf{Z}^{-1}$ to obtain $\mathbf{a}$.

Subsequently, $\mathbf{E}^s(\mathbf{r})$ is calculated by computing $\mathbf{Za}$ at a range of points. For instance, to compute $\mathbf{E}^s(\mathbf{r})$ across the $z = 0$ plane at the waveguide input port, a rectangular grid of points of the same dimensions as the plane of mesh cells shown in Figure 4.8) was used.

More detail on how each of these aspects of the simulation are implemented is available in the open-source BEAST package itself [35].

5.2.3 Post-Processing

In contrast to the UTLM package, no capability originally existed in the BEAST package to compute the scattering parameters of a simulation. In the interest of rectifying this, an attempt was made to compute the scattering parameters from the scattered field distribution of $\mathbf{E}^s(\mathbf{r})$ already performed.

An implementation of this approach already existed in the COMSOL Multiphysics® simulation package[1], where the S_{11} and S_{21} are computed as follows:

[1]The web page from which these equations were obtained is no longer available, but the URL was

$$S_{11} = \frac{\int_{port1} \left((\mathbf{E}_c - \mathbf{E}_1) \cdot \mathbf{E}_1 \right) dA_1}{\int_{port1} \left(\mathbf{E}_1 \cdot \mathbf{E}_1 \right) dA_1}, \tag{5.21}$$

$$S_{21} = \frac{\int_{port2} \left(\mathbf{E}_c \cdot \mathbf{E}_2 \right) dA_2}{\int_{port2} \left(\mathbf{E}_2 \cdot \mathbf{E}_2 \right) dA_2}, \tag{5.22}$$

where $\mathbf{E}_c$ is the scattered field computed at the relevant port, $\mathbf{E}_1$ and $\mathbf{E}_2$ are the analytical solutions of the fundamental (TE_{10}) modal field solutions at each of the input and output ports respectively[2], and dA_1 and dA_2 are the discretised elements of the areas of the input and output ports respectively.

Appendix D.3, the function written to apply the Gaussian quadrature method to solving equations 5.21 and 5.22 is listed. In order to replicate the computation of the scattering parameters across a range of frequencies, this function was called within an iterative loop in the example shown in Appendix D.1. To date, however, the data produced when applying this approach to the waveguide-post problem has produced incorrect data; only the computation of $\mathbf{E}^s(\mathbf{r})$, described in section 5.2.2, has been successful.

5.3 Chapter Summary

In this chapter, the main principles of the BEM have been detailed and the implementation of the BEM in the form of the BEAST package has been outlined, with a particular focus on the contributions made to enhancing the structure capture capabilities of the package and the attempts made to enhance post-processing. In the next chapter, the results of the UTLM and BEM simulations of the waveguide-post model are presented (where available) and discussed.

`https://lost-contact.mit.edu/afs/pdc.kth.se/roots/ilse/v0.7/pdc/vol/comsol/3.2/amd64_fc3/doc/em/wwhelp/wwhimpl/common/html/wwhelp.htm?context=&file=modeling-09.html`

[2] computed from modal field distribution formulae given in [11]

Chapter 6

Numerical Modelling Results & Comparison

In this chapter, the results of implementing models of the waveguide-post in the CST Microwave Studio™software package, Unstructured Transmission Line Modelling (UTLM) method and Boundary Element Method (BEM) are presented and discussed, with commentary on the impact upon modelling of devices such as solid-state receiver protectors.

Throughout this chapter, the analytical result obtained in Chapter 2 is used as a benchmark result against which the numerical techniques are compared to assess their relative accuracy, but it is acknowledged that this is an approximate FD method which, though indicative of the kind of behaviour to be expected from the waveguide-post configuration, is nevertheless not a comprehensive model.

6.1 Summary of Simulation Parameters

In Figure 6.1, a summary of the various simulation parameters used in each of the three techniques for which results were obtained [1]. Multiple parameters were mirrored

[1]Where no figures are given for a particular quantity, these were either unavailable or not applicable to the technique in question.

between UTLM and BEM owing to them using the same mesh.

Parameter Name & Symbol	**CST**	**UTLM**	**BEM**
Wavelength at the maximum frequency of interest, λ_0	-	7.5mm	7.5mm
Maximum length of a cubic mesh cell, Δx	0.75mm	0.75mm	0.75mm
Modal frequency, f_{modal}	14 GHz	14 GHz	14 GHz
Effective refractive index, η_{eff}	0.88	0.88	-
Slice thickness, Δz	-	0.03mm	-
Cluster threshold, s	-	9.5μm	-
Timestep, x	90μm	19μm	-

Figure 6.1: Table of simulation parameters used in CST, UTLM and BEM simulations

6.2 CST Microwave Studio™Results

In order to gauge the quality of the UTLM and BEM as *advanced* numerical techniques, the same WG16 waveguide-post model as described in previous chapters was simulated using a *current*, commercially available numerical technique - the Finite Difference Time Domain (FDTD) solver available in the CST Microwave Studio™. The 3D geometry was created in the software package itself, although the dimensions used were identical to those used in the UTLM geometry script (see Appendix B). A TE_{10} modal excitation was also used and, as a time domain (TD) technique was being employed over a finite volume mesh, that same absorbing boundary conditions (ABCs) used in UTLM were applied (see section 4.2.3 in Chapter 4 for more details).

The scattering parameter data (that is, reflection loss (S_{11}) and transmission loss (S_{21})) yielded by the waveguide-post simulations in CST is plotted in Figures 6.2 and 6.3 respectively, in comparison with the equivalent data obtained through the MAT-

LAB™analytical model detailed in Chapter 2.

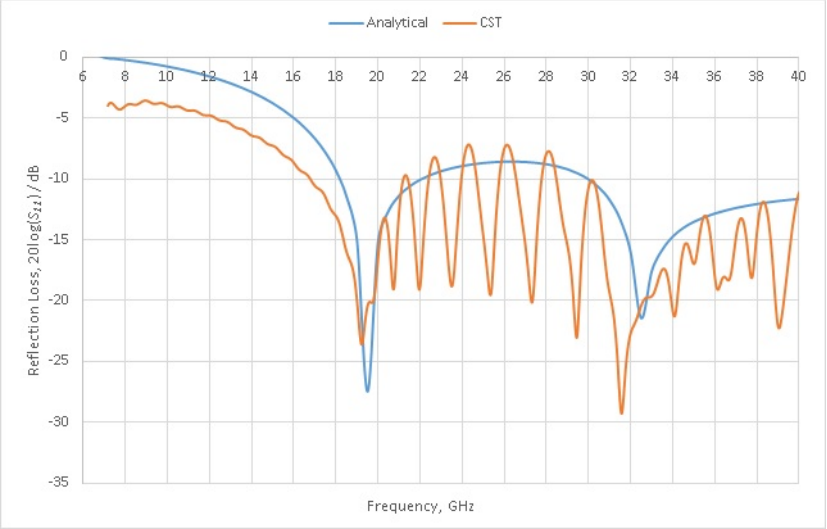

Figure 6.2: Reflection loss (S_{11}) of a TE_{10} modal excitation above cutoff in a WG16 waveguide-post geometry, simulated using CST Microwave Studio™vs. using MATLAB™analytical model, plotted against frequency

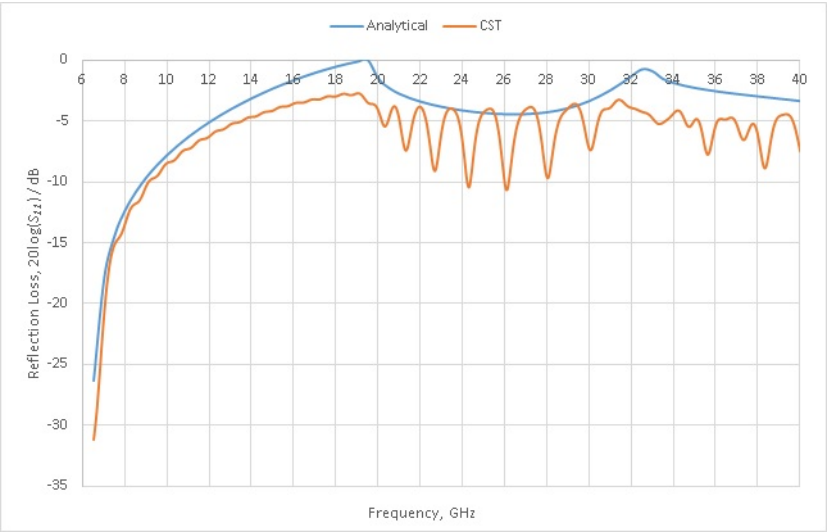

Figure 6.3: Transmission loss (S_{21}) of a TE_{10} modal excitation above cutoff in a WG16 waveguide-post geometry, simulated using CST Microwave Studio™vs. using MATLAB™analytical model, plotted against frequency

It can be observed from both Figure 6.2 and 6.3 that the CST traces (orange) bear little resemblance to the analytical traces (blue); the most notable discrepancies in the CST model are:

- The ≤5dB disparities in amplitude below the lower resonant frequency (i.e. $f \leq$ 19.2GHz) in both figures;
- The presence of large oscillations between resonant frequencies (i.e. 19.2GHz $\leq f \leq$ 31.6GHz);
- The presence of a significant degree of noise above the higher resonant frequency (i.e. $f \geq$ 31.6GHz), which is most likely caused by the interference between the multiple modes reflected and transmitted at the post;
- The mismatch between the amplitudes and positions of the two resonant frequencies of the CST model and those of the analytical model - for example in Figure 6.2 the lower resonant frequency has an error of 0.3GHz, the higher resonant frequency has an error of 0.7GHz, and in both figures the resonances have amplitudes that differ between 5 and 20dB from the analytical equivalents.

The presence of discrepancies in *amplitude* in particular indicates a poor quality of model when considering applications such as SSRPs, for which amplitude responses need to be known accurately and precisely to facilitate better device design and thus prevent damage to the wider microwave system. The fact that the CST model cannot accurately model the amplitude response of the simple waveguide-post geometry - especially at the higher frequencies of interest to microwave engineers - indicates its inadequacy to model its more complex analogue, the SSRP.

6.3 UTLM Results

In contrast, the scattering parameter data yielded by the waveguide-post simulations in UTLM (detailed in section 4.2 of Chapter 4) is plotted in Figures 6.4 and 6.5 respectively, again in comparison with the equivalent data obtained through the MATLAB™analytical model.

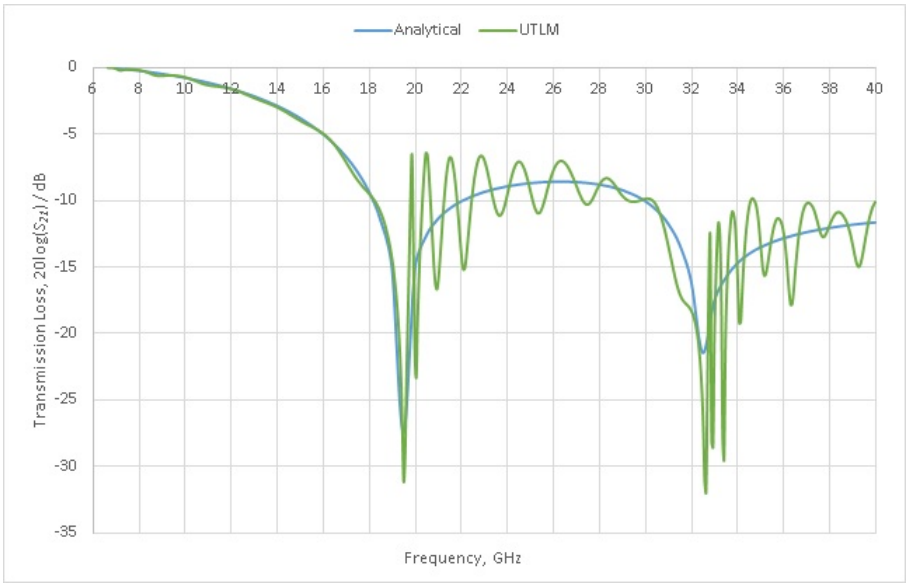

Figure 6.4: Reflection loss (S_{11}) of a TE_{10} modal excitation above cutoff in a WG16 waveguide-post geometry, simulated using UTLM gridded tetrahedral mesh vs. using MATLAB™analytical model, plotted against frequency

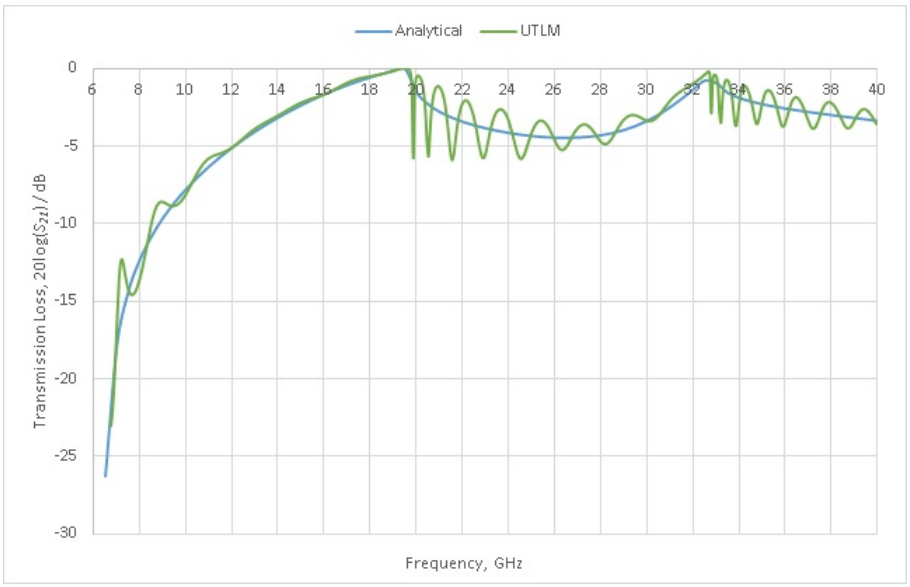

Figure 6.5: Transmission loss (S_{21}) of a TE_{10} modal excitation above cutoff in a WG16 waveguide-post geometry, simulated using UTLM gridded tetrahedral mesh vs. using MATLAB™analytical model, plotted against frequency

It can be observed from both Figure 6.4 and 6.5 that there are a number of improvements in the resemblance between the UTLM traces (green) and the analytical traces (blue)

compared to the CST results in Figures 6.2 and 6.3:

- Firstly, the UTLM traces exhibit a much-reduced disparity in amplitude below the lower resonant frequency (i.e. $f \leq 19.5\text{GHz}$) in both figures; indeed, in Figure 6.4 the difference is negligible;
- Although oscillations are still present between the resonant frequencies (i.e. $19.5\text{GHz} \leq f \leq 32.6\text{GHz}$), these are generally smaller in magnitude and actually occur *about* the analytical trace, rather than being offset by several dB as in Figures 6.2 and 6.3;
- A high degree of modal interference noise is still present above the higher resonant frequency (i.e. $f \geq 32.6\text{GHz}$), but this is limited to the reflection loss only (Figure 6.4). It would appear that the set of modes transmitted by the post in the UTLM simulation do not show as much interference as those that are reflected back.
- There is negligible mismatch between the amplitudes and positions of the two resonant frequencies of the UTLM model and those of the analytical model. However, in Figure 6.4 the oscillations in reflection loss are considerably larger at these resonances than in CST, and large deviations from the resonant amplitudes of the analytical trace are present.

Overall, the characteristics of the UTLM results indicate much closer convergence to the analytical results than the CST results do, in terms of both reduced oscillations that occur about the original amplitude response, and the more precise matching of resonant frequencies. These improvements in modelling of the waveguide-post therefore indicate that UTLM is a more appropriate tool than CST to apply to more complex SSRP modelling

However, the presence of noise at higher frequencies in Figure 6.4 is cause for concern; such discrepancies indicate that UTLM perhaps cannot yet model the higher frequency reflective behaviour of SSRPs to a satisfactory degree of accuracy.

6.4 BEAST Results

As stated in Chapter 5, the attempts made to extract scattering parameter data of the waveguide-post simulation from the Boundary Element Analysis and Simulation Toolkit were unsuccessful. However, the collection of scattered electric field data from the completed simulation was successful, and a sample of this capability is shown in Figure 6.6.

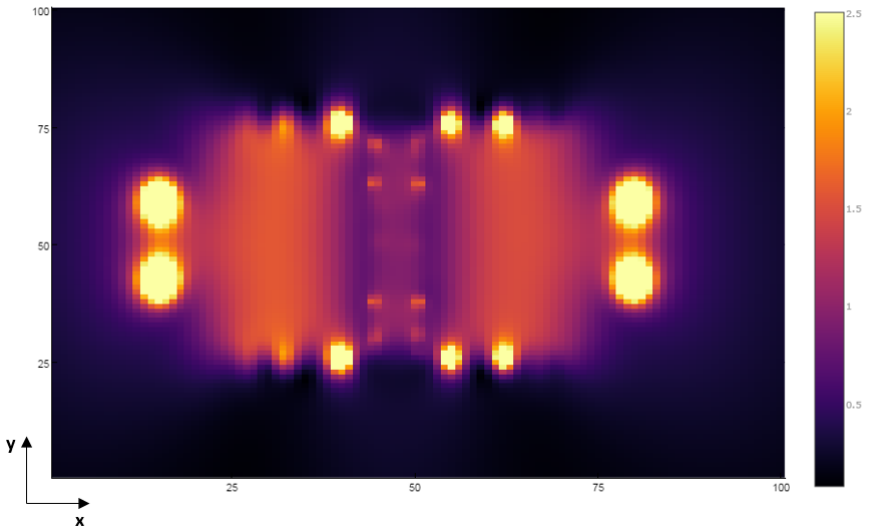

Figure 6.6: Scattered field distribution of a planewave excitation in a WG16 waveguide-post geometry. Image shows cross-section in the x-y plane in the middle of the waveguide ($z = 50$mm). (Axes denote resolution of field capture; scalebar denotes scaled magnitude of field strength)

Figure 6.6 shows the scattered field at a central cross-section of the waveguide post geometry, i.e. through the middle of the post, where the waveguide occupies the region of $25 \leq y \leq 75$ and $7 \leq x \leq 87$ (similar to the cross-section shown on the left-hand side of Figure 4.7 in Chapter 4). It can be observed that near-zero field magnitudes occur within the post (thin section in centre of plot, indicated by orange/purple shade), and higher field magnitudes occur in the areas of the waveguide surrounding the post (bright orange). This is as expected, because when an electric field is incident upon a hollow metallic surface the field scattered inside the surface is zero.

Therefore, although the computation of scattering parameters from the BEM simu-

lation was not successful, Figure 6.6 is nevertheless evidence that the BEM is capable of modelling the scattering behaviour of the waveguide-post geometry, which indicates that is capable of doing likewise for the analogous SSRP application.

6.5 Chapter Summary

In this chapter, the results obtained from the CST, UTLM and BEM simulations of the waveguide-post geometry have been presented and compared against each other where possible, and some judgments about each technique's relative suitability to modelling SSRPs have been made.

In the following chapter, the findings of this thesis are summarised, conclusions are drawn regarding the tangible progress made towards addressing the problem statement in Chapter 1, and a framework for further work in this field is suggested.

Chapter 7

Thesis Summary, Conclusions & Future Work

In this chapter, the findings of this thesis are summarised, conclusions are drawn regarding the tangible progress made towards addressing the problem statement in Chapter 1, and a framework for further work in the field of electromagnetic (EM) modelling of RADAR systems is suggested.

7.1 Thesis Summary

The problem statement made in section 1.1 of Chapter 1 described the need in the microwave engineering sector for a computational technique to more accurately model the behaviour of solid state receiver protectors (SSRPs). The work towards meeting this need presented in each of the chapters of this thesis is summarised as follows:

- In **Chapter 1**, the purpose and operation of the SSRP were introduced and explained, and the characteristic frequency response of this device was highlighted in contrast with the inability of current modelling techniques to accurately model this response;

- In **Chapter 2**, the physical theory of EM propagation in waveguides - the device family to which SSRPs belong - was presented. The impact on propagation of introducing discontinuities (such as the vertical posts characterising SSRP diode stages) into a waveguide was then examined, with respect to the changes in transmission and reflection behaviour that result. From this examination, an analytical model analogous to an SSRP - a waveguide with inductive post - was devised to be used as a test case, and its implementation in MATLAB™was described, providing benchmark key performance indicator (KPI) data against which novel numerical modelling techniques could later be compared.

- In **Chapter 3**, the concept of numerical modelling was introduced, different categories of established numerical techniques were described, and their advantages and disadvantages were compared against those of two novel techniques: the Unstructured Transmission Line Modelling (UTLM) method and the Boundary Element Method (BEM). Thus, the motivations for selecting each for the development of SSRP models were established: UTLM, as a stable, high resolution method capable of accurately modelling the multiscale, complex nature of SSRP geometry, and BEM as an efficient method with strengths in the farfield computation essential in SSRP modelling.

- In **Chapter 4**, the theory behind UTLM was outlined, with regard to how the propagation of EM fields is mapped onto the propagation of voltages through a transmission line network in one-, two- and three-dimensional space; the concept and power of unstructured meshes was also explained. The implementation of UTLM currently under development at the George Green Institute for Electromagnetics Research (GGIEMR) was then explained in each of its three stages - geometry creation, mesh generation and solving - and the parameters chosen in each of these stages to yield an optimised model of the waveguide-post test case were summarised.

- In **Chapter 5**, the mathematical approaches used in the BEM to formulate, discretise and solve the electric field integral equation (EFIE) were presented in detail. The implementation of these principles in the form of the Boundary Element Analysis and Simulation Toolkit (BEAST) package was outlined, and the functionality added to the package during the project was summarised - in particular, the expansions in

structure capture capability and attempts to derive scattering parameter data.

- In **Chapter 6**, the results obtained from the FDTD[1], UTLM and BEM simulations of the waveguide-post model were presented. The differences between the FDTD and UTLM simulations were noted with reference to the analytical model developed in Chapter 2, and it was found that the UTLM method produced amplitude responses more faithful to those produced by the analytical model than those produced by the FDTD method.

7.2 Conclusions

From this body of work, a number of conclusions and achievements on the path towards developing better models of SSRPs can be claimed:

- The strengths and drawbacks of the UTLM and BEM techniques with regard to SSRP modelling are varied, and in principle no single technique is predominant over the other for this application.
- The strengths of the UTLM method lie in its ability to model the multiscale nature of an SSRP, although its broadband response exhibits a questionable degree of noise at the upper end of the frequency spectrum of interest for the waveguide-post test case model. There would appear to be some refinements to the UTLM method that are necessary to reduce the impact of modal interference when simulating reflection behaviour.
- The strengths of the BEM lie in its efficient computation of fields across a far larger problem space than UTLM, although the failure to generate scattering parameter data during this project means the accuracy of this capability with regard to the SSRP application cannot *yet* be fully verified.
- However, efforts made to streamline the parallel workflows of UTLM and BEM have been fruitful, as shown in the 'before and after' comparison exhibited in Figure 7.1.

[1]Using the commercial package CST Microwave Studio™

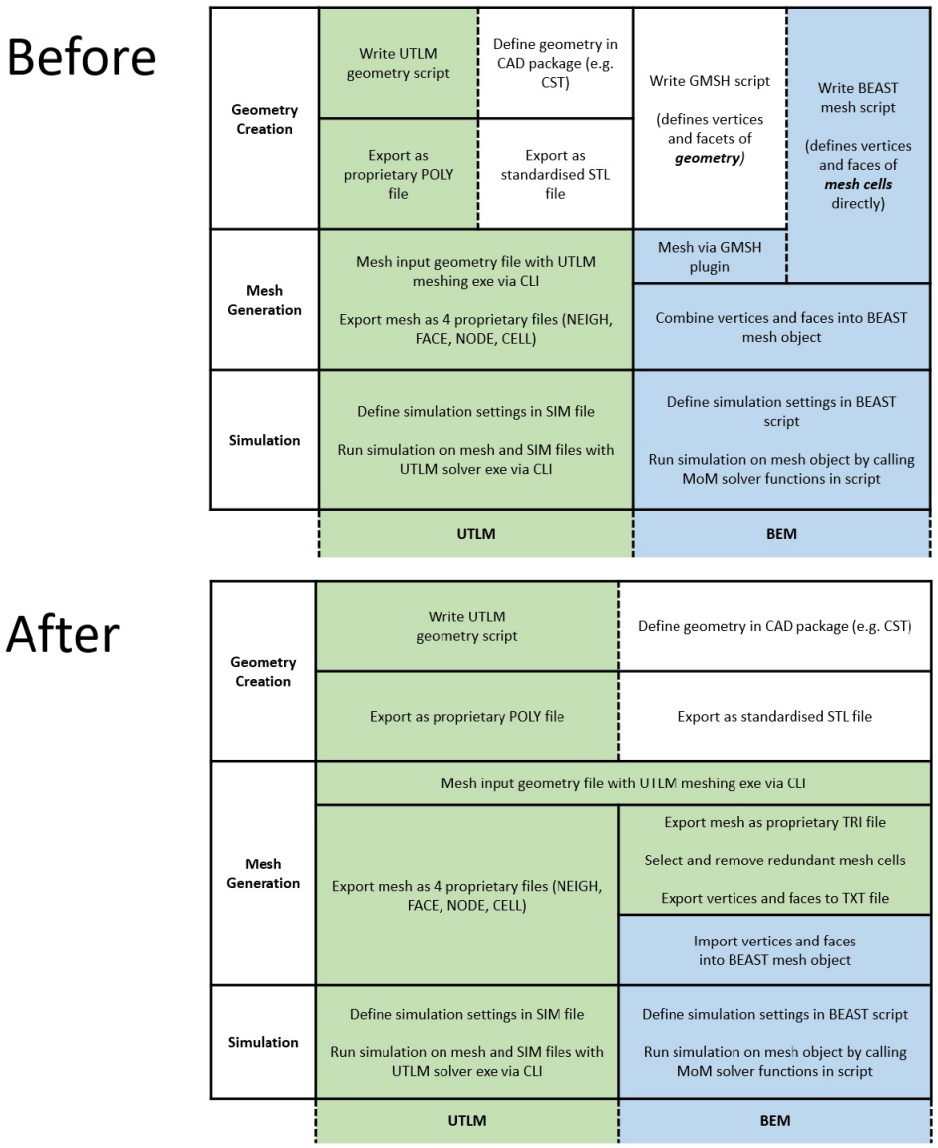

Figure 7.1: Progress made in streamlining the workflows of UTLM and BEM, as shown in 'before and after' diagrams

Figure 7.1 shows that the geometry capture and meshing capabilities of both methods have been merged, meaning that meshes created in UTLM (or via commercial package) can be imported into *either* UTLM or BEM - the addition of the surface mesh transformation tool described in Chapter 5 (and listed in Appendix C) means that meshes need no longer be generated *separately* for the UTLM and BEM methods; instead, the established meshing capability of UTLM can be coupled with BEM to produce higher quality simulations.

7.3 Proposed Framework for Subsequent Work

To progress further towards implementing complete SSRP models in UTLM and BEM, the following framework is suggested:

1. Once the new implementation of the full SSRP model in CST currently being undertaken at Teledyne E2V is complete, a plan to progressively add components (e.g. (dielectric shields, diode contacts, diode capsules etc.) from the SSRP device to the waveguide-post model needs to be devised. Then, by importing these incrementally-altered models into UTLM and BEM, the effect of these components on the frequency response modelled in each technique will be observable, and the strengths and weaknesses of each technique in modelling different types of components will be able to be deduced.

2. The problems preventing the scattering parameter computation code in Appendix D.3 from accurately calculating scattering parameters need to be solved.

3. More comparable simulations need to be run in UTLM and BEM; for instance, modelling planewave excitations in UTLM (as in BEM) and using the time-domain (TD) implementation of BEM in conjunction with the purely TD implementation of UTLM.

4. A similar capability to that in UTLM for modelling different waveguide excitation modes needs to be devised for BEM, in order for the high-frequency capability of the BEM method to be more easily examinable.

Appendix A

MATLAB Implementation of Analytical Model

The MATLAB code that follows was written to implement the analytical model of the waveguide post, as depicted in Figure 2.4 in Chapter 2.

```
%This script calculates the S11 and S12 scattering parameter
%exhibited by an analytical model of a WG16 waveguide with
   inductive post based
%on the physical parameters and impedance calculations in the
   following references:

% Waveguide Handbook, N. Marcuvitz, pp. 257-263
% Microwave Engineering, D. M. Pozar, pp. 190,192

% This function works by:
% - Initialising the waveguide and modal excitation constants, a
   nd
% frequency range;
% - Running through a frequency loop to calculate and store the
```

```
   reflection and transmission
% response of the inductive post at each frequency;
% - Plotting the S11 and S21 parameters against frequency.

%----Andrew Morley, September 2017----

%--------Physical constants----------
speed_light = 3 * (10^8); %Speed of light, m * s^-1
permeability_vacuum = 4 * pi * (10^-7); %Vacuum permeability, H
   * m^-1
permittivity_vacuum = 8.85 * (10^-12); %Vacuum permittivity, F *
    m^-1

%--------Waveguide and mode constants--------
waveguide_width = (0.9 * 2.54) * (10^-2);
waveguide_height = (0.4 * 2.54) * (10^-2);
post_diameter = (0.09 * 2.54) * (10^-2);

mode_m = 1; %Mode number in x-direction
mode_n = 0; %Mode number in y-direction

wavenumber_cutoff = sqrt((((mode_m * pi) / waveguide_width) ^2)
   + (((mode_n * pi) / waveguide_height) ^2));
wavelength_cutoff = (2 * pi) / wavenumber_cutoff; % Correct for
   TE10 mode

%-------Frequency scanning loop: limits and step definitions
   ------
frequency_min = 5 * (10^9);
frequency_max = 40 * (10^9);
```

```
frequency_step = 0.5 * (10^9);
frequency_no = ((frequency_max - frequency_min) / frequency_step
   ) + 1;

%-------Results arrays: dynamic sizing and initialisation
   --------
frequency_storage = zeros(frequency_no, 1);
S_11_results = zeros(frequency_no, 1);
S_21_results = zeros(frequency_no, 1);

frequency_no_current = 1; %Initialises row position of arrays

%Frequency scanning loop
for frequency = frequency_min:frequency_step:frequency_max

    frequency_angular = 2 * pi * frequency;
    wavelength_free_space = speed_light / frequency;
    wavenumber = sqrt(((2 * pi * frequency)^2) * permeability_va
       cuum * permittivity_vacuum);
    phase_constant = sqrt((wavenumber^2) - (wavenumber_cutoff^2)
       );
    wavelength_guide = wavelength_free_space / sqrt(1 - ((wa
       velength_free_space / wavelength_cutoff)^2)); %Should equ
       al approx. 6.8GHz

    impedance_characteristic = (wavelength_guide / wa
       veguide_width) * (waveguide_height / pi) * frequency_a
       ngular * permeability_vacuum;
    impedance_wave = (frequency_angular * permeability_vacuum) /
        phase_constant;
```

```
%Calculation of S_0 and S_2 coefficients (see Marcuvitz page
    263)
sum_component_0 = 0;
sum_component_2 = 0;

for n = 3:2:300

    sum_component_0 = sum_component_0 + ((1 / sqrt((n^2) -
       ((2 *waveguide_width) / wavelength_free_space)^2)) -
       (1/n));
    sum_component_2 = sum_component_2 + (sqrt((n^2) - ((2 *
       waveguide_width) / wavelength_free_space)^2) - n +
       ((2 / n) * ((waveguide_width / wavelength_free_space)
       ^2)));

end

s_coefficient_0 = log((4* waveguide_width) / (pi * post_diam
   eter)) - 2 + (2 * sum_component_0);
s_coefficient_2 = log((4* waveguide_width) / (pi * post_diam
   eter)) - 2.5 + ((11/3) * ((wavelength_free_space / (2 * w
   aveguide_width))^2)) - (((wavelength_free_space / wa
   veguide_width)^2) * sum_component_2);

%Impedance calculations
XA_halfXB_normalised = (waveguide_width / (2 * wa
   velength_guide)) * (s_coefficient_0 -((pi * post_diameter
    / (2 * wavelength_free_space))^2) - ((5 / 8) * ((pi *
   post_diameter / (2 * wavelength_free_space))^4)) - ((2 *
```

```
    ((pi * post_diameter / (2 * wavelength_free_space))^4)) *
     (s_coefficient_2 - (2 * s_coefficient_0 * (wa
    velength_free_space^2) / (wavelength_guide^2)))^2));

XB_normalised = (waveguide_width / wavelength_guide) * (((pi
    * post_diameter) / waveguide_width)^2) * (1 / (1 + (0.5
    * (((pi * post_diameter) / wavelength_free_space)^2) * (
    s_coefficient_2 + 0.75))));
XB = XB_normalised * impedance_characteristic;
ZB = - 1i * XB;

XA_normalised = (XA_halfXB_normalised + (0.5 * XB_normalised
    ));
XA = XA_normalised * impedance_characteristic;
ZA = 1i * XA;

impedance_sum = ZA + ZB;

%Calculation of S11 and S21 (as per derivation in Chapter 2)
S_11_results(frequency_no_current, 1) = 20 * log10(abs(((2 *
    ZA * ZB) + (ZB^2) - (impedance_characteristic^2)) / ((2
    * impedance_characteristic * ZA) + (2 * impedance_chara
    cteristic * ZB) + (2 * ZA * ZB) + (ZB^2) + (impedance_cha
    racteristic^2))));
S_21_results(frequency_no_current, 1) = 20 * log10(abs((2 *
    impedance_characteristic * ZA) / ((2 * impedance_chara
    cteristic * ZA) + (2 * impedance_characteristic * ZB) +
    (2 * ZA * ZB) + (ZB^2) + (impedance_characteristic^2))));

%Stores current frequency and iterates
```

```
    frequency_storage(frequency_no_current, 1) = frequency;
    frequency_no_current = frequency_no_current + 1;

end;


%Plotting of traces for each frequency value
plot(frequency_storage(:, 1),S_11_results(:, 1));
hold on;
plot(frequency_storage(:, 1),S_21_results(:, 1));
hold on;

ax = gca;
ax.XAxis.Exponent = 9;
xlabel('Frequency, GHz') % x-axis label
ylabel('Transmission/Reflection Loss, dB') % y-axis label
ax.XAxis.MinorTick = 'on';
ax.YAxis.MinorTick = 'on';
set(gca,'fontsize',16);
axis([frequency_min frequency_max -50 0]); % Axis limits

grid on;

datacursormode on;
```

Appendix B

UTLM Implementation of Waveguide-Post Geometry

The C++ code that follows was written to implement the UTLM waveguide-psot geometry shown in Figure 4.6 in Chapter 4.

```
#include <map>
#include <math.h>

#include "topology.h"
#include "geometry_scene.h"

#include <algorithm>
#include <functional>
#include <numeric>
#include <ext/functional>
#include <bits/stl_function.h>

#include "testing.h"
```

```
using namespace std;

using __gnu_cxx::compose1;
using __gnu_cxx::compose2;
using __gnu_cxx::identity;
using __gnu_cxx::select1st;
using __gnu_cxx::select2nd;

///--------------------------------SETUP AND RUN INSTRUCTIONS
   -------------------------------------------

/// 1. Install the UTLM geometry kernel (gk) (and MinGW if not a
   lready installed)
/// 2. Copy this text into a file, name it 'ssrp_testing' and sa
   ve it as a .cpp file in the 'gk' folder
/// 3. Add the line 'ssrp_testing.cpp' to the SOURCES list in
   the file 'gk/makefile'
/// 4. Add the following lines to the list in the file 'gk/
   dos_version.cpp':
///     - 'examples.push_back(pair<string,scene_generator>("
   centred_post_in_waveguide",centred_post_in_waveguide));'
///     - 'examples.push_back(pair<string,scene_generator>("
   centred_post_in_waveguide_HS",centred_post_in_waveguide_HS))
   ;'
/// 5. Add the following lines to the list in the file 'gk/
   testing.h':
///     - 'void centred_post_in_waveguide(scene* sc);'
///     - 'void centred_post_in_waveguide_HS(scene* sc);'
/// 6. In the MinGW command line, navigate to the gk folder and
   run the command 'make' followed by either
```

```
///     - './gk centred_post_in_waveguide' for a geometry suita
   ble for MODAL excitations, or
///     - './gk centred_post_in_waveguide_HS' for a geometry
   suitable for PLANE WAVE excitations
/// 7. Once the run is complete, open either the POLY, plottable
    POLY or TRI files in the MeshViewer from the 'gk' folder



//------------------------------------------------------------------------

void centred_post_in_waveguide(scene* sc){

///This function renders a UTLM geometry of a WG16 waveguide
   with a circular, inductive post placed partway along the wa
   veguide. The dimensions used were taken from the following
   reference:

///Waveguide Handbook, N. Marcuvitz, pp. 257-263

///The function:
/// - Creates the surface of the waveguide
/// - Creates the surface of the inductive post and pushes it
   through the waveguide material;
/// - Imposes reflective boundary conditions on the waveguide wa
   lls in order to represent their metallic nature.

///All dimensions in mm (unless otherwise stated)

///Andrew Morley - September 2017
```

```
///-----------------------------------WAVEGUIDE CONSTRUCTION
   -------------------------------------------

    ///Dimensions and unit conversions
    const double waveguide_width_inches(0.9);///'a' from litera
       ture
    const double waveguide_width_mm(waveguide_width_inches
       *2.54*10); ///Conversion to metric
    const double waveguide_height_inches(0.4);///'b' from litera
       ture
    const double waveguide_height_mm(waveguide_height_inches
       *2.54*10); ///Conversion to metric

    const double waveguide_length(100); ///Selected arbitrarily

    ///Construction of waveguide

    const vertex waveguide_origin(0, 0, 0);
    const geom_vector waveguide_inner_cuboid_diagonal(wa
       veguide_width_mm, waveguide_height_mm, waveguide_length);
    int waveguide=sc->make_cuboid(waveguide_origin, wa
       veguide_inner_cuboid_diagonal);

        ///Tags below enable other material properties of wa
           veguide to be assigned in .sim file
   sc->set_volume_tag(waveguide,"WAVEGUIDE");
   sc->create_surface_property(waveguide,"SURFACE_OF_WAVEGUIDE")
      ;
```

```
///---------------------------------------POST CREATION
   ------------------------------------------------

    ///Dimensions and unit conversions
80  const double post_diameter_inches(0.09);
    const double post_diameter_mm(post_diameter_inches*2.54*10);
        ///Conversion to metric

    const geom_vector post_height(0, waveguide_height_mm, 0);

    ///Construction of post
    const vertex post_origin(waveguide_width_mm/2, 0, wa
       veguide_length/2);
    const int post_number_of_sides(50);
    int post=sc->make_cylinder(post_origin, post_height, post_di
       ameter_mm/2, post_number_of_sides);

90  ///Metallic property of post recognised; will be removed in
       mesher
    sc->set_remove_in_mesher_and_create_surface_property(post,"r
       =-1");

    ///Tags below enable other material properties of post to be
        assigned in .sim file
   sc->set_volume_tag(post,"POST");
   sc->create_surface_property(post,"SURFACE_OF_POST");

    sc->through(waveguide,post);

///-------------------------------------BOUNDING BOX CREATION
```

```
   ------------------------------------------------

    int all = sc->assemble();

    int bounding_box=sc->set_bounding_box
       (-1,-1,-1,-1,0,0,0,0,0,0,0,0); ///Sets reflective
       properties of metallic waveguide walls

}
//---------------------------------------------------------------------

void centred_post_in_waveguide_HS(scene* sc){

///This function renders a geometry of a WG16 waveguide with a
   circular, inductive post placed partway along the waveguide.
   The
///dimensions used were taken from the following reference:

///Waveguide Handbook, N. Marcuvitz, pp. 257-263

///The function:
/// - Creates the surface of the waveguide
/// - Creates the surface of the inductive post and pushes it
   through the waveguide material;
/// - Imposes reflective boundary conditions on the waveguide wa
   lls in order to represent their metallic nature;
/// - Encloses the problem in a Huygens surface for the purposes
    of a subsequent plane wave excitation.

///All dimensions in mm (unless otherwise stated)
```

```
///Andrew Morley - September 2017

///-----------------------------------------------WAVEGUIDE
   CONSTRUCTION
   ------------------------------------------------------

    ///Dimensions and unit conversions
    const double waveguide_width_inches(0.9);///'a' from litera
       ture
    const double waveguide_width_mm(waveguide_width_inches
       *2.54*10); ///Conversion to metric
    const double waveguide_height_inches(0.4);///'b' from litera
       ture
    const double waveguide_height_mm(waveguide_height_inches
       *2.54*10); ///Conversion to metric

    const double waveguide_length(100); ///Selected arbitrarily

    ///Construction of waveguide

    const vertex waveguide_origin(0, 0, 0);
    const geom_vector waveguide_inner_cuboid_diagonal(wa
       veguide_width_mm, waveguide_height_mm, waveguide_length);
    int waveguide=sc->make_cuboid(waveguide_origin, wa
       veguide_inner_cuboid_diagonal);

        ///Tags below enable other material properties of wa
           veguide to be assigned in .sim file
   sc->set_volume_tag(waveguide,"WAVEGUIDE");
```

```
   sc->create_surface_property(waveguide,"SURFACE_OF_WAVEGUIDE")
      ;



///-------------------------------------------------POST
   CREATION
   ------------------------------------------------------

    ///Dimensions and unit conversions
    const double post_diameter_inches(0.09); ///Calculated from
       'd/a' in literature
    const double post_diameter_mm(post_diameter_inches*2.54*10);
        ///Conversion to metric

    const geom_vector post_height(0, waveguide_height_mm + 2, 0)
       ;

    ///Construction of post
    const vertex post_origin(waveguide_width_mm/2, -1, wa
       veguide_length/2); ///Insertion depth will be specified l
       ater
    const int post_number_of_sides(50);
    int post=sc->make_cylinder(post_origin, post_height, post_di
       ameter_mm/2, post_number_of_sides);

    ///Metallic property of post recognised; will be removed in
       mesher
    sc->set_remove_in_mesher_and_create_surface_property(post,"r
       =-1");
```

```
    ///Tags below enable other material properties of post to be
        assigned in .sim file
   sc->set_volume_tag(post,"POST");
   sc->create_surface_property(post,"SURFACE_OF_POST");

    sc->through(waveguide,post);

///-------------------------------------HUYGENS SURFACE
   CONSTRUCTION-----------------------------------------------

    ///Calculating operating wavelength
    const double speed_of_light(3 * (10^8));
    const double frequency_operating(10 * (10^9)); ///Operating
       frequency estimated as middle of X-band (i.e. 8-12 GHz)
    const double wavelength_operating(1000*speed_of_light/
       frequency_operating);

    cout<<"\nOperating wavelength: "<<wavelength_operating<<"mm\
       n"<<endl;

    int whole_structure=sc->assemble();

   ///Inner box - dimensions set to be maximum dimensions of
      problem plus 1 operating wavelength on either side
    const double inner_box_width(waveguide_width_mm+(2*wa
       velength_operating));
    const double inner_box_height(waveguide_height_mm+(2*wa
       velength_operating));
    const double inner_box_depth(waveguide_length+(2*wa
       velength_operating));
```

```
const vertex inner_box_origin(-wavelength_operating,-wa
   velength_operating,-wavelength_operating);
 const geom_vector inner_box_diagonal(inner_box_width,
    inner_box_height,inner_box_depth);
 int inner_box=sc->make_cuboid(inner_box_origin,inner_box_dia
    gonal);

     ///Tags below enable inner box properties to be assigned
         in .sim file
sc->set_volume_tag(inner_box,"INNER_BOX");
sc->create_surface_property(inner_box,"SURFACE_OF_INNER_BOX")
   ;

sc->through(inner_box,whole_structure); ///Antenna array,
   substrate and coaxial cable all put through inner box

 ///Outer box - to be removed in mesher
 const int outer_box_to_inner_box_gap(100); ///Gap between
    boxes is arbitrary

 const int outer_box_width(inner_box_width+(2*
    outer_box_to_inner_box_gap));
 const int outer_box_height(inner_box_height+(2*
    outer_box_to_inner_box_gap));
 const int outer_box_depth(inner_box_depth+(2*
    outer_box_to_inner_box_gap));

 const vertex outer_box_origin(-(wavelength_operating+
    outer_box_to_inner_box_gap),-(wavelength_operating+
```

```
      outer_box_to_inner_box_gap),-(wavelength_operating+
      outer_box_to_inner_box_gap));
   const geom_vector outer_box_diagonal(outer_box_width,
      outer_box_height,outer_box_depth);
   int outer_box=sc->make_cuboid(outer_box_origin,outer_box_dia
      gonal);

   ///Tags below allow referencing of outer box in .sim file, a
      nd removal in mesher
  sc->set_volume_tag(outer_box,"OUTER_BOX");
  sc->set_remove_in_mesher_and_create_surface_property(
     outer_box,"SURFACE_OF_OUTER_BOX");

   sc->through(outer_box,inner_box); ///Inner box put through
      outer box

}
```

Appendix C

UTLM-BEAST Mesh Modification Script

The C++ code that follows was written to implement the UTLM surface mesh to BEAST surface mesh conversion described in section 5.2 of Chapter 5.

```
#include <stdlib.h>
#include <string.h>
#include <iostream>
#include <fstream>
#include <vector>
#include <algorithm>
#include <stdint.h>

///-----------------------UTLM SURFACE MESH --> BEAST SURFACE
   MESH CONVERTER-----------------------------

///This function allows the user to remove selected mesh cells
   from a UTLM surface mesh such that any non-desired reflective
    surfaces are absent, thus pre-processing the mesh ready for
```

```
   import into the BEAST software package:

///The function:
/// - Reads in a UTLM surface mesh (TRI file);
/// - Accepts user input of the dimensions of a cube (in mm)
   within which mesh cells will be removed;
/// - Writes the vertices and faces of all cells EXCEPT those
   selected to an output buffer;
/// - Repeats previous two steps successively, allowing user to
   remove multiple regions of mesh;
/// - Writes output buffer to TXT file when user indicates mesh
   cell removal is complete.

///All dimensions in mm (unless otherwise stated)

///Phillip Sewell, Andrew Morley - September 2017

///--------------------------------SETUP AND RUN INSTRUCTIONS
   ---------------------------------------------

/// 1. Install MinGW if not already installed.
/// 2. Copy this text into a file, name it 'binary_tri_selector'
    and save it as a .cpp file
/// 3. Copy the TRI file to convert ('name_of_input_binary_file.
   tri') to the same folder where 'binary_tri_selector.cpp' is
   stored.
/// 3. In the MinGW command line, navigate to the folder where '
   binary_tri_selector.cpp' is stored
/// 4. Compile with the command 'g++ -o name_of_input_bina
   ry_file.tri binary_tri_selector.cpp'
```

```
/// 5. Run with the command './binary_tri_selector nam
   e_of_input_binary_file.tri'
/// 6. The output TXT file is automatically named the same the
   TRI file, but preceded by 'from_'; copy this file to the root
    of the BEAST package.

using namespace std;

typedef int FILE_INT;

typedef int32_t OP_FILE_INT;

template<typename T> struct triplet {

   T a,b,c;

   triplet(T _a=0,T _b=0,T _c=0):a(_a),b(_b),c(_c){}

   T get_a(){return(a);}
   T get_b(){return(b);}
   T get_c(){return(c);}
};

template<typename T> struct quadlet {

   T a,b,c,d;

   quadlet(T _a=0,T _b=0,T _c=0,T _d=0):a(_a),b(_b),c(_c),d(_d)
      {}
```

```
   T get_a(){return(a);}
   T get_b(){return(b);}
   T get_c(){return(c);}
   T get_d(){return(d);}
};



int main(int argc, char* argv[]) {

   char* filename(argv[1]);

// double xmin(atof(argv[2]));
// double xmax(atof(argv[3]));
//
// double ymin(atof(argv[4]));
// double ymax(atof(argv[5]));
//
// double zmin(atof(argv[6]));
// double zmax(atof(argv[7]));

/// Note the alternative way of opening/reading binary files
   used in type_1_binary_to_ascii.cpp

/// Sometimes portability my prefer one over the other

   FILE *pfile=fopen(filename,"rb");

   cout<<"\n\nOpen the BINARY (ele expected) file named {"<<
      filename<<"}";cout.flush();
```

```
/// Read the point data

   FILE_INT no_points;

90       if(fread(&no_points,sizeof(FILE_INT),1,pfile)<1) {

      cout<<"\n\nThe file does not appear to have any data - the
          'no_points' field has a value of "<<no_points<<"\n";
         cout.flush();

      return(0);
   }

   cout<<"\nno_points="<<no_points;cout.flush();

   FILE_INT no_dimensions;

100

       if(fread(&no_dimensions,sizeof(FILE_INT),1,pfile)<1) {

      cout<<"\n\nThe file does not appear to have any data - the
          'no_dimensions' field has a value of "<<no_dimensions
         <<"\n";cout.flush();

      return(0);
   }

   cout<<"\nno_dimensions="<<no_dimensions;cout.flush();

110   FILE_INT no_point_attributes;  /// Not used but in standard
```

```
    file format

     if(fread(&no_point_attributes,sizeof(FILE_INT),1,pfile)
        <1) {

   cout<<"\n\nThe file does not appear to have any data - the
       'no_point_attributes' field has a value of "<<
      no_point_attributes<<"\n";cout.flush();

   return(0);
}

cout<<"\nno_point_attributes="<<no_point_attributes;cout.
   flush();

vector<triplet<double> > point_coords;

double* point_attributes(NULL); /// Not used but in standard
   file format

FILE_INT gg(0);

     point_coords.resize(no_points);

 if((gg=fread(&point_coords[0],sizeof(double),3*no_points,
    pfile))<3*no_points)
 {
     cout<<"\n\nThe file does not appear to have valid data -
         the 'point_coords' data, expected "<<3*no_points<<"
        read "<<gg;
```

```
        cout.flush();

        return(0);
    }

    if(no_point_attributes>0)
    {
        point_attributes=new double[no_points*no_point_a
           ttributes];

        if((gg=fread(point_attributes,sizeof(double),no_points*
           no_point_attributes,pfile))<no_points*no_point_a
           ttributes)
        {

            cout<<"\n\nThe file does not appear to have valid da
               ta - the 'point_attributes' data, expected "<<
               no_points*no_point_attributes<<" read "<<gg;
            cout.flush();

            return(0);
        }
    }

/// Read the cell geometry data

   FILE_INT no_cells(0);

   FILE_INT no_vertices_per_cell(0);
```

```
FILE_INT no_records(0);

     if(fread(&no_cells,sizeof(FILE_INT),1,pfile)<1) {

   cout<<"\n\nThe file does not appear to have any data - the
       'no_cells' field has a value of "<<no_cells<<"\n";cout
      .flush();

   return(0);
}

cout<<"\nno_cells="<<no_cells;cout.flush();

     if(fread(&no_vertices_per_cell,sizeof(FILE_INT),1,pfile)
        <1) {

   cout<<"\n\nThe file does not appear to have any data - the
       'no_vertices_per_cell' field has a value of "<<
      no_vertices_per_cell<<"\n";cout.flush();

   return(0);
}

cout<<"\nno_vertices_per_cell="<<no_vertices_per_cell;cout.
   flush();

     if(fread(&no_records,sizeof(FILE_INT),1,pfile)<1) {

   cout<<"\n\nThe file does not appear to have any data - the
       'no_records' field has a value of "<<no_records<<"\n";
```

```
        cout.flush();

    return(0);
}

cout<<"\nno_records="<<no_records;cout.flush();

 vector<triplet<FILE_INT> >cell_vertices;

cell_vertices.resize(no_cells);

if((gg=fread(&cell_vertices[0],sizeof(FILE_INT),
   no_vertices_per_cell*no_cells,pfile))<no_vertices_per_cell
   *no_cells) {

   cout<<"\n\nThe file does not appear to have valid data -
      the 'cell_vertices' data, expected "<<
      no_vertices_per_cell*no_cells<<" read "<<gg;cout.flush
      ();

   return(0);
}


 double xmin(0);
 double xmax(0);
 double ymin(0);
 double ymax(0);
 double zmin(0);
 double zmax(0);
```

```
vector<int> selected_points;
FILE_INT no_selected_points(0);
vector<int> selected_cells;
FILE_INT no_selected_cells(0);

vector<int> selected_points_buffer;
vector<int> selected_cells_buffer;

int answer(1);
int counter(0);

do
{
    cout<<counter<<"\n";
    cout<<"\n\nSETTING OF BOUNDARIES OF CUBE CONTAINING
       CELLS TO BE REMOVED:\n";
    cout<<"\nPlease enter the min x-coordinate\n";
    cin>>xmin;
    cout<<"\nPlease enter the max x-coordinate\n";
    cin>>xmax;
    cout<<"\nPlease enter the min y-coordinate\n";
    cin>>ymin;
    cout<<"\nPlease enter the max y-coordinate\n";
    cin>>ymax;
    cout<<"\nPlease enter the min z-coordinate\n";
    cin>>zmin;
    cout<<"\nPlease enter the max z-coordinate\n";
    cin>>zmax;
    cout<<"\n"<<xmin<<","<<xmax<<","<<ymin<<","<<ymax<<","<<
```

```
        zmin<<","<<zmax;

    if(counter == 0) ///i.e. if on first pass of cell remova
        l
    {

    ///Filtering out unwanted points

        for(FILE_INT i=0; i<no_points; ++i)
        {
            if(!(point_coords[i].get_a()>xmin&&
            point_coords[i].get_a()<xmax&&
            point_coords[i].get_b()>ymin&&
            point_coords[i].get_b()<ymax&&
            point_coords[i].get_c()>=zmin&&
            point_coords[i].get_c()<=zmax)) selected_points.
                push_back(i);
        }

    ///Filtering out unwanted cells

        for(FILE_INT i=0; i<no_cells; ++i)
        {
            vector<int>::iterator viter(lower_bound(
                selected_points.begin(),selected_points.end()
                ,cell_vertices[i].get_a()));
            if(viter!=selected_points.end()&&(*viter==
                cell_vertices[i].get_a()))
            {
                //cell_vertices[i].a=distance(
```

```
                    selected_points.begin(),viter);
                viter=lower_bound(selected_points.begin(),
                    selected_points.end(),cell_vertices[i].
                    get_b());

                if(viter!=selected_points.end()&&(*viter==
                    cell_vertices[i].get_b()))
                {
                    //cell_vertices[i].b=distance(
                        selected_points.begin(),viter);
                    viter=lower_bound(selected_points.begin
                        (),selected_points.end(),
                        cell_vertices[i].get_c());

                    if(viter!=selected_points.end()&&(*viter
                        ==cell_vertices[i].get_c()))
                    {
                        //cell_vertices[i].c=distance(
                            selected_points.begin(),viter);

                        if(!(point_coords[cell_vertices[i].
                            get_a()].get_c() == zmin &&
                            point_coords[cell_vertices[i].
                                get_b()].get_c() == zmin &&
                            point_coords[cell_vertices[i].
                                get_c()].get_c() == zmin))
                        {
                            selected_cells.push_back(i);
                        }
                    }
```

```
                }
            }
        }
    }
    else ///i.e. if on subsequent pass of cell removal
    {
        cout<<"\nPass "<<counter;
        for(FILE_INT i=0; i<no_selected_points; ++i)
        {
            if(!(point_coords[selected_points_buffer[i]].
               get_a()>xmin&&
            point_coords[selected_points_buffer[i]].get_a()<
               xmax&&
            point_coords[selected_points_buffer[i]].get_b()>
               ymin&&
            point_coords[selected_points_buffer[i]].get_b()<
               ymax&&
            point_coords[selected_points_buffer[i]].get_c()
               >=zmin&&
            point_coords[selected_points_buffer[i]].get_c()
               <=zmax)) selected_points.push_back(
               selected_points_buffer[i]);
        }

        for(FILE_INT i=0; i<no_selected_cells; ++i)
        {
            vector<int>::iterator viter(lower_bound(
               selected_points.begin(),selected_points.end()
               ,cell_vertices[selected_cells_buffer[i]].get_
               a()));
```

```
                if(viter!=selected_points.end()&&(*viter==
                   cell_vertices[selected_cells_buffer[i]].get_
                   a()))
                {
                    cell_vertices[selected_cells_buffer[i]].a=
                       distance(selected_points.begin(),viter);
                    viter=lower_bound(selected_points.begin(),
                       selected_points.end(),cell_vertices[
                       selected_cells_buffer[i]].get_b());

300                 if(viter!=selected_points.end()&&(*viter==
                       cell_vertices[selected_cells_buffer[i]].
                       get_b()))
                    {
                        cell_vertices[selected_cells_buffer[i]].
                           b=distance(selected_points.begin(),
                           viter);
                        viter=lower_bound(selected_points.begin
                           (),selected_points.end(),
                           cell_vertices[selected_cells_buffer[i
                           ]].get_c());

                        if(viter!=selected_points.end()&&(*viter
                           ==cell_vertices[selected_cells_buffer
                           [i]].get_c()))
                        {
                            cell_vertices[selected_cells_buffer[
                               i]].c=distance(selected_points.
                               begin(),viter);
```

```
                        if(!(point_coords[cell_vertices[
                           selected_cells_buffer[i]].get_a()
                           ].get_c() == zmin &&
                        point_coords[cell_vertices[
                           selected_cells_buffer[i]].get_b()
                           ].get_c() == zmin &&
                        point_coords[cell_vertices[
                           selected_cells_buffer[i]].get_c()
                           ].get_c() == zmin))
                        {
                            selected_cells.push_back(
                               selected_cells_buffer[i]);
                        }
                    }
                }
            }
        }

        no_selected_points = selected_points.size();
        cout<<"\n\nNumber of selected points="<<
           no_selected_points;

        no_selected_cells = selected_cells.size();
        cout<<"\nNumber of selected cells="<<no_selected_cells<<
           "\n";

        cout<<"\nWould you like to remove another group of cells
           ? (yes = 1; no = 0)";
        cin>>answer;
```

```
330         if(answer==1)
            {
                selected_points_buffer.clear();
                if (selected_points_buffer.empty() == true) cout<<"\
                   nselected_points_buffer emptied";
                selected_cells_buffer.clear();
                if (selected_cells_buffer.empty() == true) cout<<"\
                   nselected_cells_buffer emptied";

                for(int i=0;i<no_selected_points;++i)
                {
                    selected_points_buffer.push_back(selected_points
                       [i]);
340             }

                for(int i=0;i<no_selected_cells;++i)
                {
                    selected_cells_buffer.push_back(selected_cells[i
                       ]);
                }

                selected_points.clear();
                if (selected_points.empty() == true) cout<<"\
                   nselected_points emptied";
                selected_cells.clear();
350             if (selected_cells.empty() == true) cout<<"\
                   nselected_cells emptied";

                counter++;
```

```
        }

    }while(answer==1);

/// Export as an Excel TXT file

    string opfile(string("from_")+string(argv[1])+string(".txt")
       ); // opfile = "from_filename.csv"

    cout<<"\n\nWriting the ASCII file (arrays starting at 1) {"
       <<opfile<<"}";
    cout.flush();

    ofstream fout (opfile.c_str());

    fout<<"POINTS\n";

    fout<<no_selected_points;

    for(int i=0; i<no_selected_points; ++i) fout<<"\n"<<i+1<<" "
       <<point_coords[selected_points[i]].get_a()<<" "<<
       point_coords[selected_points[i]].get_b()<<" "<<
       point_coords[selected_points[i]].get_c();

    fout<<"\nEND_POINTS";

    fout<<"\nTRIANGLES\n";

    fout<<no_selected_cells;
```

```
    for(int i=0; i<no_selected_cells; ++i) fout<<"\n"<<i+1<<" "
       <<cell_vertices[selected_cells[i]].get_a() + 1<<" "<<
       cell_vertices[selected_cells[i]].get_b() + 1<<" "<<
       cell_vertices[selected_cells[i]].get_c() + 1;

    fout<<"\nEND_TRIANGLES";

    fclose(pfile);

    fout.close();

    cout<<"\n\nOpen the output file in a txt viewer - it maybe
       very large! \n\nSearch for the words POINTS, TRIANGLES a
       nd RECORDS for the start of the three sections\n\n";

   return(0);
}
```

Appendix D

BEAST Implementation of Waveguide-Post Simulation

In the sections of this appendix, the contributions to the Boundary Element Analysis and Simulation Toolkit (BEAST) software package are listed.

For further information on the BEAST package please visit `https://github.com/krcools/BEAST.jl` or contact Dr Kristof Cools (`Kristof.Cools@nottingham.ac.uk`).

D.1 Example: Waveguide-Post Model

In this section, the example file in which the waveguide-post model was implemented is listed.

This example is located at `https://github.com/krcools/BEAST/examples/waveguide_with_post.jl`

```
using BEAST
using CompScienceMeshes
```

```
# Creation of a Euclidian space
o, x, y, z = euclidianbasis(3)

# Name of modified UTLM surface mesh file to be input to program
filename = "from_centred_post_in_waveguide.tri.txt"

# Calling function to parse mesh into program (function listed
   in Appendix D.2)
input_mesh = CompScienceMeshes.read_TRI_mesh(filename)

#-------------------VIEWING MESH----------------------

# Assigning random numbers to each mesh cell
A = rand(numcells(input_mesh)) #

# Plotting mesh using PlotlyJS package, by assigning random
   colours to each cell
include(Pkg.dir("CompScienceMeshes","examples","plotlyjs_patches
   .jl"))
p = patch(input_mesh, A)
PlotlyJS.plot([p])

# Plotting mesh using MATLAB package
include(Pkg.dir("CompScienceMeshes","examples","matlab_patches.
   jl"))
p = patch(input_mesh, A)

#------------RAVIART-THOMAS BASIS GENERATION------------
```

```
RT = raviartthomas(input_mesh)

#---------------------CONSTANTS-------------------------

permeability = pi * 4 * exp10(-7)
permittivity = 8.85 * exp10(-12)

#-------------MODAL EXCITATION SOLVING---------------

# Constant arguments necessary for computing TE_10 modal field
amplitude_value = 1.0
waveguide_width_value = 22.86*exp10(-3)
waveguide_height_value = 10.16*exp10(-3)
waveguide_length_value = 100*exp10(-3)
mode_number_m_value = 1
mode_number_n_value = 0

angular_frequency_modal = 2 * pi * 10 * exp10(9)
wavenumber_modal = angular_frequency_modal * sqrt(permeability *
    permittivity)

t_modal = Maxwell3D.singlelayer(gamma=im*wavenumber_modal)
E_MODAL = Maxwell3D.modal(direction=z, polarization=y, wavenum
   ber=wavenumber_modal, amplitude=amplitude_value, frequency=a
   ngular_frequency_modal,
                          waveguide_width=waveguide_width_value,
                              waveguide_height=wa
                             veguide_height_value,
                          mode_number_m=mode_number_m_value, m
                             ode_number_n=mode_number_n_value)
```

```
e_modal = (n x E_MODAL) x n

@hilbertspace j
@hilbertspace k

# Discretisation of electric field integral equation (EFIE)
efie_modal = @discretise t_modal[k,j] == e_modal[k] jâĹĹRT
    kâĹĹRT

# Solving of EFIE
u_modal = gmres(efie_modal)
fcr_modal, geo_modal = facecurrents(u_modal, RT)

#-----DEFINITION OF GRIDS OF POINTS AT WAVEGUIDE INPUT & OUTPUT
    PORTS------

grid_input = [SVector(x, y, 0) for x in linspace(-5*exp10(-3),
    30*exp10(-3), 100) for y in linspace(-5*exp10(-3), 15*exp10
    (-3), 100)]
grid_output = [SVector(x, y, 100*exp10(-3)) for x in linspace
    (-5*exp10(-3), 30*exp10(-3), 100) for y in linspace(-5*exp10
    (-3), 15*exp10(-3), 100)]

#------------------MODAL FIELD COMPUTATION
    ------------------------

fundamental_mode_field_input = E_MODAL.(grid_input)
fundamental_mode_field_input = reshape(fundamental_m
    ode_field_input, (100, 100))
```

```
fundamental_mode_field_output = E_MODAL.(grid_output)
fundamental_mode_field_output = reshape(fundamental_m
   ode_field_output, (100, 100))

#---------------ITERATION THROUGH FREQUENCIES
   ----------------------

frequencies = collect(0 : 0.5*exp10(9) : 40*exp10(9))
S11_results = zeros(length(frequencies),1)
S21_results = zeros(length(frequencies),1)

for i in eachindex(frequencies)

  #-------------PLANE WAVE EXCITATION SOLVING------------

  angular_frequency = 2 * pi * frequencies[i]
   wavenumber = angular_frequency * sqrt(permeability * perm
      ittivity)

  t_pw = Maxwell3D.singlelayer(gamma=im*wavenumber)
  E_PW = Maxwell3D.planewave(direction=z, polarization=y, wavenu
     mber=wavenumber)
  e_pw = (n x E_PW) x n

  efie_pw = @discretise t_pw[k,j] == e_pw[k] jâĹĹRT kâĹĹRT

  u_pw = gmres(efie_pw)
  println("Solution computed")
```

```
fcr_pw, geo_pw = facecurrents(u_pw, RT)

println("Frequency ", i, " of ", length(frequencies), ":")

 #----------------NEARFIELD CALCULATION----------------------

 nearfield_input = potential(MWSingleLayerField3D(wavenumber),
     grid_input, u_pw, RT)
 nearfield_input = reshape(nearfield_input, (100, 100))

 nearfield_output = potential(MWSingleLayerField3D(wavenumber)
    , grid_output, u_pw, RT)
 nearfield_output = reshape(nearfield_output, (100, 100))

 #-------------CALCULATING S-PARAMETERS
    ------------------------

# Calling function to compute scattering parameters (function
   listed in Appendix D.3)
 S11, S21 = get_scatter_parameters(waveguide_width, wa
    veguide_height, waveguide_length, fundamental_m
    ode_field_input, fundamental_mode_field_output,
                                          nearfield_input, nea
                                             rfield_output)

# Array storage of computed scattering parameter values at
   current frequency
 S11_results[i] = 20 * log10(abs(S11))
 S21_results[i] = 20 * log10(abs(S21))
```

```
end

#------------PLOTTING SCATTERING PARAMETERS---------------

include(Pkg.dir("CompScienceMeshes","examples","plotlyjs_patches
    .jl"))
using Plots
plot(frequencies, S11_results)

#-------------VIEWING E-FIELD------------------

A = [real(f[1]) for f in fundamental_mode_field_input]
A = real.(norm.(fundamental_mode_field_input))
A = clamp(A,0.0,2.5)
heatmap(A)

#-------------VIEWING FACE CURRENTS------------------

include(Pkg.dir("CompScienceMeshes","examples","plotlyjs_patches
    .jl"))
A = [real(norm(f)) for f in fcr]
p = patch(input_mesh, A)
p = patch(geo, real.(norm.(fcr)))
PlotlyJS.plot([p])

include(Pkg.dir("CompScienceMeshes","examples","matlab_patches.
    jl"))
p = patch(input_mesh, A)

nothing;
```

D.2 Function: Parsing UTLM Surface Mesh into BEM

In this section, the function which was written to parse the modified UTLM mesh file (generated by the code in Appendix C) is listed; this function is called on line 12 of the example in section D.1.

This function is located in the file stored at: `https://github.com/krcools/CompScienceMeshes/src/fileio/TRI_mesh.jl`

```
"""
    read_TRI_mesh(filename) -> mesh::Mesh
"""
function read_TRI_mesh(fn::AbstractString)
    #Opens input file
    open(fn, "r") do io
        read_TRI_mesh(io)
    end
end

"""
    read_TRI_mesh(mesh_filename) -> mesh::Mesh

Imports a surface mesh (stored in an ASCII file named ‘m
   esh_filename‘) into a
    ‘Mesh‘ object (i.e. node list and element list).

NOTE: The contents of ‘mesh_filename‘ must include the file
   extension, and the
    file must be stored in the current directory.

"""
function read_TRI_mesh(io)
```

```
#Reads in vertices from file
thisLine = io |> readline |> strip
while thisLine != "POINTS"
    thisLine = io |> readline |> strip
end

#Parses vertex ID into integer variable
thisLine = io |> readline |> strip
s = split(thisLine)
NV = parse(Int, s[1])

#Parses vertex x, y and z coordinates into vector variable
P = SVector{3,Float64}
v = Vector{P}(NV)
for i in 1:NV
    thisLine = io |> readline |>  strip
    d = readdlm(IOBuffer(thisLine), Float64)
    v[i] = P(d[2], d[3], d[4])
end

#Reads in faces from file
while thisLine != "TRIANGLES"
    thisLine = io |> readline |> strip
end

#Parses face ID into integer variable
thisLine = io |> readline |> strip
s = split(thisLine)
NF = parse(Int, s[1])
```

```
    #Parses 3 vertex IDs of face into vector variable
    thisLine = io |> readline |> strip
    P = SVector{3,Int64}
    f = Vector{P}(NF); i = 0
    while thisLine != "END_TRIANGLES"
        d = readdlm(IOBuffer(thisLine), Int64)
        thisLine = io |> readline |>  strip
60      f[i += 1] = P(d[2], d[3], d[4])
    end

    #Returns lists of vertices and faces to program
    return Mesh(v, f)

end
```

D.3 Function: Scattering Parameter Computation

In this section, the function which was written to compute the scattering parameters of the BEM waveguide-post simulation is listed; this function is called on line 116 of the example in section D.1. It is important to note that this function does not currently work correctly, but is included for completeness.

This function is located in the file stored at: `https://github.com/krcools/BEAST/src/postproc.jl`

```
"""
   get_scatter_parameters(waveguide_width, waveguide_height, wa
      veguide_length,
                                        fundamental_mode_input,
                                           fundamental_mode_output,
```

```
                                        nearfield_input, nea
                                       rfield_output)

Computes the S11 (reflection) and S21 (transmission) scattering
   parameters of a two port network
"""
function get_scatter_parameters(waveguide_width, wa
   veguide_height, waveguide_length, fundamental_mode_input,
   fundamental_mode_output,
                                                nearfield_input,
                                                    nea
                                                   rfield_output
                                                   )

    #Definition of quadrature points for input waveguide port
    origin_input = point(0,0,0)
    x_input = point(waveguide_width,0,0)
    y_input = point(0,waveguide_height,0)
    rect_input = CompScienceMeshes.RectangleChart(origin_input
       , SVector((x_input,y_input)))
    QP_input = quadpoints(rect_input, (5, 6))

    #Definition of quadrature points for output waveguide port
    origin_output = point(0,0,waveguide_length)
    x_output = point(waveguide_width,0,waveguide_length)
    y_output = point(0,waveguide_height,waveguide_length)
    rect_output = CompScienceMeshes.RectangleChart(
       origin_output, SVector((x_output,y_output)))
    QP_output = quadpoints(rect_output, (5, 6))
```

```
    S11_fields = collect(zip(nearfield_input,fundamental_m
        ode_input))
    S21_fields = collect(zip(nearfield_output,fundamental_m
        ode_output))

    #Initialisation of variables
    S11_numerator, S11_denominator, S21_numerator, S21_denomin
        ator = 0, 0, 0, 0

    #Iterative loop summing all terms of the discretised
        integral
    for i in eachindex(QP_input)
        S11_numerator += QP_input[i][2] * dot((S11_fields[i
            ][1] - S11_fields[i][2]), conj(S11_fields[i][2]))
        S11_denominator += QP_input[i][2] * dot(S11_fields[i
            ][2], conj(S11_fields[i][2]))

        S21_numerator += QP_output[i][2] * dot(S21_fields[i
            ][1], conj(S21_fields[i][2]))
        S21_denominator += QP_output[i][2] * dot(S21_fields[
            i][2], conj(S21_fields[i][2]))
    end

    #Computation of final S11 and S21 values
    S11 = S11_numerator / S11_denominator
    S21 = S21_numerator / S21_denominator

    return S11, S21

end
```

References

[1] G. Stimson and K. Newsome, "Discussion of existing ssrp modelling and research motivation with representatives of teledyne e2v ltd," 07/11/2016 2016.

[2] B. Coaker, "Radar receiver protection technology," *Microwave Journal*, 2007.

[3] D. Leenov, "The silicon pin diode as a microwave radar protector at megawatt levels," *IEEE Transactions on Electron Devices*, vol. 11, no. 2, pp. 53–61, 1964.

[4] CPI Wireless Solutions (Beverly Microwave Division), "Receiver protector technology," CPI Wireless Solutions (Beverly Microwave Division), Report.

[5] Skyworks Solutions, Inc., "Pin limiter diodes in receiver protectors," Skyworks Solutions, Inc., Report, 2006.

[6] J. Allison, *Electronic Engineering: Semiconductors and Devices*. McGraw-Hill College, 1990.

[7] "Types of inductors in electronics," 2017. [Online]. Available: https://www.lifewire.com/types-of-inductors-818826

[8] B. M. Coaker, D. M. Dowthwaite, and N. E. Priestley, "High-power multi-function radar receiver protection," Report.

[9] N. Farrington, M. Dowthwaite, G. Fletcher, K. Newsome, and B. M. Coaker, "Development and verification of an accurate fem electromagnetic model for a complex multi-stage radar limiter," in *Radar Conference (EuRAD), 2011 European*, Conference Proceedings, pp. 37–40.

[10] D. K. Cheng, *Fundamentals of Engineering Electromagnetics*, ser. Addison Wesley Series in Electrical Engineering. Addison-Wesley Publishing Company, 1994.

[11] D. M. Pozar, *Microwave Engineering*, 4th ed. John Wiley & Sons, Inc., 2012.

[12] J.-M. Jin, *Theory and Computation of Electromagnetic Fields*. Hoboken, New Jersey, U.S.A.: John Wiley & Sons, Inc., 2010.

[13] C. Christopoulos, *The Transmission Modeling Method: TLM*, ser. IEEE/OUP Series on Electromagnetic Wave Theory. IEEE Press, 1995.

[14] N. Marcuvitz, *Waveguide Handbook*, ser. Electromagnetic Waves Series 21. London, U.K.: The Institution of Engineering and Technology, 2009, vol. 21.

[15] J. Schwinger and D. S. Saxon, *Discontinuities in Waveguides: Notes on Lectures by Julian Schwinger*, ser. Documents on Modern Physics. New York: Gordon and Breach Science Publishers, 1968.

[16] EverythingRF, "Wr90 | wg16 | r100 - rectangular waveguide size," 2017. [Online]. Available: https://www.everythingrf.com/tech-resources/waveguides-sizes/wr90

[17] R. E. Colin, *Field Theory of Guided Waves*. Piscataway, New Jersey: IEEE Press, 1991.

[18] D. Simmons, "Hybrid methods for modelling advanced electromagnetic systems using unstructured meshes," Thesis, 2016.

[19] E. K. Miller, "A selective survey of computational electromagnetics," *IEEE Transactions on Antennas and Propagation*, vol. 36, no. 9, pp. 1281–1305, 1988.

[20] "Cst microwave studio - tlm solver," 2015. [Online]. Available: https://www.cst.com/products/cstmws/solvers/tlm-solver

[21] P. Sewell, J. G. Wykes, T. M. Benson, C. Christopoulos, D. W. P. Thomas, and A. Vukovic, "Transmission-line modeling using unstructured triangular meshes," *IEEE Transactions on Microwave Theory and Techniques*, vol. 52, no. 5, pp. 1490–1497, 2004.

[22] H. Nasser, "3d to 2d surface mesh parameterisation for the purpose of unstructured transmission line modelling method simulations," Thesis, 2015.

[23] X. Meng, "Modelling multi-scale problems in the transmission line modelling method," Thesis, 2014.

[24] D. Simmons, K. Cools, and P. Sewell, "Coupling of unstructured tlm and bem for accurate 2d electromagnetic simulation," in *2015 International Conference on Electromagnetics in Advanced Applications (ICEAA)*, Conference Proceedings, pp. 1076–1079.

[25] P. Sewell, A. Vukovic, X. Meng, and T. M. Benson, "A note on material losses in unstructured transmission line modeling," *Microwave and Optical Technology Letters*, vol. 57, no. 9, pp. 2218–2222, 2015.

[26] P. Sewell, T. M. Benson, C. Christopoulos, D. W. P. Thomas, A. Vukovic, and J. G. Wykes, "Implicit element clustering for tetrahedral transmission-line modeling (tlm)," *IEEE Transactions on Microwave Theory and Techniques*, vol. 57, no. 8, pp. 2005–2014, 2009.

[27] "Ansys hfss capabilities - high-frequency electromagnetic solvers," 2017. [Online]. Available: http://www.ansys.com/en-GB/products/electronics/ansys-hfss/hfss-capabilities

[28] R. Mittra, "A look at some challenging problems in computational electromagnetics," *IEEE Antennas and Propagation Magazine*, vol. 46, no. 5, pp. 18–32, 2004.

[29] Y. Beghein, K. Cools, H. Bagci, and D. De Zutter, "A space-time mixed galerkin marching-on-in-time scheme for the time-domain combined field integral equation," *IEEE Transactions on Antennas and Propagation*, vol. 61, no. 3, pp. 1228 – 1238, 2013.

[30] W. C. Gibson, *The Method Of Moments in Electromagnetics*. Boca Raton, Florida: Chapman & Hall/CRC, 2008.

[31] L. Khashan, A. Vukovic, P. Sewell, and T. M. Benson, "Dispersion in the 2d unstructured transmission line modelling (utlm) method," in *2015 IEEE International Conference on Computational Electromagnetics (ICCEM)*, Conference Proceedings.

[32] P. Sewell, "Utlm solver manual," 2013.

[33] A. Vukovic, "Guidlines for utlm solver, version 1.0," 2014.

[34] F. T. Ulaby, E. Michielssen, and U. Ravaioli, *Fundamentals of Applied Electromagnetics*, 6th ed. Pearson, 2010.

[35] K. Cools, "Boundary element analysis and simulation toolkit," 2017. [Online]. Available: https://github.com/krcools/BEAST.jl

[36] C. Geuzaine and J. F. Remacle, "Gmsh: a three-dimensional finite element mesh generator with built-in pre- and post-processing facilities," *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.