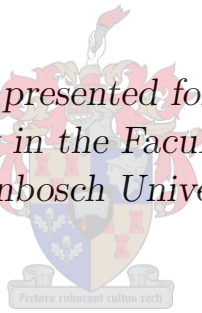
# The Method of Manufactured Solutions for the Verification of Computational Electromagnetic Codes

by

Renier Gustav Marchand



*Dissertation presented for the degree*
*Doctor of Philosophy in the Faculty of Engineering at*
*Stellenbosch University*

Promoter: Prof. David B. Davidson

Department of Electrical and Electronic Engineering

March 2013

# Declaration

By submitting this dissertation electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the sole author thereof (save to the extent explicitly otherwise stated), that reproduction and publication thereof by Stellenbosch University will not infringe any third party rights and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

Date: March 2013 .....

# Abstract


## The Method of Manufactured Solutions for the Verification of Computational Electromagnetic Codes

RG Marchand

*Department of Electrical and Electronic Engineering,*
*University of Stellenbosch,*
*Private Bag X1, 7602 Matieland, South Africa.*

Dissertation: PhD

March 2013

In this work the Method of Manufactured Solutions (MMS) is introduced for the code verification of full-wave frequency dependent electromagnetic computational software.

At first the method is sketched in the context of the verification and validation process and the need for proper code verification is highlighted.

Subsequently, the MMS is investigated in its natural context: the Finite Element Method, specifically for the E-field Vector Wave Equation. The usefulness of the method to detect error in a computational code is demonstrated. The selection of Manufactured Solutions is discussed and it is demonstrated how it can be used to find the probable cause of bugs. Mutation testing is introduced and used to show the ability to detect errors present in code.

The MMS is finally applied in a novel manner to a Method of Moments (MoM) code. The challenges of numerical integration associated with the application of the operator is discussed and correct integration is successfully demonstrated. Subsequently the MMS is demonstrated to be successfully applied to the MoM and mutation testing is used to demonstrate the practical efficacy of the method.

The application of the MMS to the MoM is the main contribution of this work.

# Uittreksel

## The Method of Manufactured Solutions for the Verification of Computational Electromagnetic Codes

RG Marchand

*Departement Elektriese en Elektroniese Ingenieurswese,*
*Universiteit van Stellenbosch,*
*Privaatsak X1, 7602 Matieland, Suid Afrika.*

Proefskrif: PhD

Maart 2013

Die Metode van Vervaardigde Oplossings (MVO) word hier bekend gestel vir die verifikasie van numeriese volgolf frekwensie-afhanklike elektromagnetise kode.

Die metode word eerstens in die breë konteks van algemene verifikasie en validasie geplaas en gevolglik word die noodsaaklikheid van kode verifikasie beklemtoon.

Daarna, word die toets-metode in die konteks van die Eindige Element Metode vir die E-veld vektorgolf vergelyking bestudeer. Die MVO is oorspronklik ontwikkel in die differentiaalvergelyking omgewing. Die bruikbaarheid van die metode vir elektromagnetiese simulasies word prakties gedemonstreer deur die opsporing van werklike foute. Die metode word ook verder ondersoek vir die oorsprong van foute. Mutasietoetsing word bekendgestel en word gebruik om die metode verder prakties te verifieer.

Die MVO word laastens in 'n nuwe manier gebruik om 'n Moment Metode kode te verifieer. Die praktiese probleme betrokke by numeriese integrasie word ondersoek en die korrekte toepassing van die integraal operator word prakties gedemonstreer. Daarna, word die MVO in hierdie konteks gedemonstreer deur verskeie voorbeelde te ondersoek. Mutasietoetsing word weereens gebruik om na die effektiewiteit van die MVO te kyk om 'n Moment Metode kode te toets.

Die toepassing van die MVO op 'n Moment Metode kode is die hoof bydrae van hierdie werk.

# Acknowledgements

I would like to thank Professor Davidson for the invaluable conversations and guidance during this project. I would also especially like to thank him for all the encouragement he gave me while working on this.

Then I would like to thank Matthys Botha for valuable conversations we had and insights he gave me over a casual cup of coffee. Then all the members of the CEMAGG team for their work in the form of SUCEM:FEM and conversations we had over numerous cups of tea and coffee, these were especially valuable breaks and helped me a great deal.

I would like to thank the CHPC for the Flagship project funding for "HPC electromagnetic simulation for the MeerKAT and SKA" and the NRF/SKA-SA for funding under the "MeerKAT High Performance Computing for Radio Astronomy Research Programme."

Thank you to all my friends, family and loved ones for their patience with the ups and downs that are involved in a study of this kind.

Lastly I would also like to thank the numerous unknown people that worked on the open source software that I used. These include Sympy [78], FEniCS [19] and the LaTeX developers and maintainers.

# Contents

# List of Figures

# List of Tables

# Notation and Symbols used

## General mathematical notation used in this work

- General unknown scalar quantities are indicated by italic text – $x$
- Known scalar quantities, typically constants are indicated by normal text – x
- Vector quantities are indicated using boldface text – $\boldsymbol{x}$
- Unit vectors are indicated using boldface and a hat over the symbol – $\hat{\boldsymbol{x}}$
- Matrices and linear algebra vectors are also indicated using boldface capital and lowercase characters respectively, but context will make the use clear – $\boldsymbol{A}$, $\boldsymbol{x}$
- In general the discretised quantity of a continuous value is indicated with a subscript $h$ – $E$ becomes $E_h$
- The letter $j$ is used to indicate the imaginary component of a complex number

# Operators

The following table lists general operators used in this work

**Table 1:** Table of notation used

| Notation | Meaning |
|---|---|
| $\langle A, B\rangle_X$ | Inner product of $A$ and $B$ for the space $X$ |
| $\langle A, B\rangle$ | Bilinear form of $A$ and $B$ |
| $L(A)$ | A linear functional<br>(The linear form defined as $\langle L, A\rangle$) |
| $(A, B)$ | The sesquilinear form defined as<br>$(A, B) = \langle A, \overline{B}\rangle$ |
| $\|A\|_X$ | The norm of $A$ in the space $X$,<br>defined as $\|A\| = \langle A, A\rangle_X$ for Hilbert-spaces |

# Symbols and quantities used

Table 2: Symbols used in this work

| Symbol | Meaning |
|---|---|
| $\boldsymbol{E}$, $\boldsymbol{E}^{sc}$, $\boldsymbol{E}^{inc}$ | Electric field intensity: total, scattered and incident<br>$\boldsymbol{E} = \boldsymbol{E}^{sc} + \boldsymbol{E}^{inc}$ |
| $\boldsymbol{D}$ | Electric flux density |
| $\boldsymbol{H}$ | Magnetic field intensity |
| $\boldsymbol{B}$ | Magnetic flux density |
| $\boldsymbol{J}$ | Electric current density |
| $\boldsymbol{J}_s$ | Electric surface current density |
| $\boldsymbol{A}$ | Vector potential |
| $\phi$ | Scalar potential |
| $\boldsymbol{P}$ | Dirichlet boundary condition |
| $\boldsymbol{U}$ | Neumann boundary condition |
| $\rho$ | Electric charge density |
| $\rho_s$ | Electric surface charge density |
| $\epsilon$, $\epsilon_0$, $\epsilon_r$ | Permittivity: total, free-space and relative respectively<br>$\epsilon = \epsilon_0\epsilon_r$<br>($\epsilon$ may also refer to a small arbitrary number but context should make its use clear) |
| $\mu$, $\mu_0$, $\mu_r$ | Permeability: total, free-space and relative respectively<br>$\mu = \mu_0\mu_r$ |

Continued on next page

| Symbol | Meaning |
|---|---|
| $\sigma$ | Conductivity |
| $\omega$ | Angular frequency: $\omega = 2\pi\omega$ |
| $k_0$ | Free-space wave number: $k_0 = \frac{2\pi}{\lambda_0}$ |
| $Z_0$ | Free-space wave impedance: $Z_0 = \sqrt{\frac{\mu_0}{\epsilon_0}}$ |
| $c$ | Speed of light $c = \frac{1}{\sqrt{\mu\epsilon}}$ |
| $t, x, y, z$ | Time and the usual Cartesian coordinates |
| | |
| $\Omega$ | Solution domain or free-space around solution domain. |
| $\Gamma$ | Solution boundary $\partial\Omega$, also screen/solution domain embedded in $\Omega$ |
| $\Gamma_D$ | Dirichlet boundary |
| $\Gamma_N$ | Neumann boundary |
| $\hat{\boldsymbol{n}}$ | Normal unit vector |
| | |
| $X, X^s$ | The general solution space of a numerical method, possibly dependent on the Sobolev-regularity |
| | |
| $u, u_h$ | A general continuous and discretised solution |
| $u_{h\to 0}$ | The limit of the discretised solution as $h \to 0$. |
| $u_{h,c}$ | A particular solution on a particular computer, $c$ |
| $\tau_h$ | A mesh set of size $h$ |
| $K_i$ | The mesh element $i$ |
| $h$ | An indication of mesh element size |
| $\boldsymbol{\phi}_i$ | Basis function $i$ |
| $p$ | Polynomial order, typically of a basis function |
| $\boldsymbol{W}$ | A general testing function |
| $C$ | A constant not dependent on mesh size $h$ |
| $E$ | An indication of the difference between the continuous and discretised solution |
| $s$ | Sobolev-regularity, discussed in Appendix C |
| $\mathcal{L}$ | A general operator |
| | |
| $P_0$ | A original unmutated code |
| $P_M, P_i$ | A set of mutated code versions with a particular denoted by $i$ |

Next a short list of mathematical spaces are mentioned, more information such as relevant norms may be found in Appendix C.

Table 3: Mathematical spaces

| Symbol | Meaning |
|---|---|
| $\mathbb{R}^n$ | The space of $n$-dimensional real numbers |
| $\mathbb{Z}, \mathbb{Z}^+$ | The space of integers, and integers such that $n \geq 0$ |
| $L^n$ | The space of functions to the $n$-th power of integrable distributions |
| $H^s$ | The Sobolev space $W^{2,s}$, a Hilbert space of differentiability $s$ |
| $H^s_{\text{curl}}$ | The *curl*-conforming Sobolev space |
| $H^s_{\text{div}}$ | The *div*-conforming Sobolev space |

# Chapter 1

# Introduction

In this work the Method of Manufactured Solutions (MMS) is presented to verify full-wave electromagnetic simulation code implementations. The method is presented for the verification of a full-wave, frequency domain finite element (FE) code implementation and a full-wave, frequency domain method of moments (MoM) code implementation. It is shown that this method is successful in the detection of coding error and coding conceptual error.

The MMS has been used for code verification in the fluid dynamics community [59, 69, 72, 77] but not, as far as the author could ascertain, for computational electromagnetic code. It is applied here to a differential method, the FEM, which is the original environment where the method was developed. Various practical aspects of the MMS are investigated for the FEM. The method is further extended in this work to the verification of a full-wave integral method, the MoM, which is the main contribution of this work.

The MMS is a method used to detect the presence of coding and coding conceptual errors [71]. The method is not aimed towards the verification of the mathematical accuracy of the original computation model, nor is it a method to validate the agreement between physical and simulated systems. However, the implementation of a computational code must be verified before it is used to calculate the aforementioned physical system. The MMS is conceptually simple to understand and can be explained in the following four steps:

1. Choose a solution that is desired and is in the solution space of the method being tested.

2. Compute the driving and boundary terms that should result in the desired solution.

3. Numerically solve the system and compare computed results to the chosen solution.

4. Compare the convergence rate to the expected theoretical rate.

The application of these four basic steps results in a very powerful tool to detect the presence of errors. Each of these four steps will be investigated as they are relevant to a particular solution method.

The FEM and the MoM are widely used throughout the electromagnetics community. As such, customised versions of these methods are frequently implemented for research purposes. Commercial versions are also available for these methods. There is a clear need to verify and validate that these methods are correctly implemented. There are validation standards available for these types of codes in literature [38, 39] that are aimed at the verification and validation of the physical implementation and simulation ability of a computational code. The aforementioned validation standards are typically aimed towards the validation of an implementation and not the code verification. The distinction between and the role that the MMS and validation standards play respectively is discussed in Chapter 2.

The need to verify the efficacy of a testing method becomes important when it is used to verify a code implementation. To this end, Mutation Testing is introduced in Chapter 3. Mutation Testing is widely used by the Computer Science community to investigate the efficacy of a testing procedure for a specific code implementation. Therefore, this method is used throughout this document to verify the applicability of the MMS. A mutation testing tool, MutantPie, has been developed for Python and released into the public domain. This tool was used throughout this work.

In Chapter 4 the MMS is specifically investigated for the FEM. It is shown that the method does indeed work as presented by Roache [69] and that it is applicable to computational electromagnetics for 2D problems as well as 3D problems. The effect of meshing on the calculated convergence rate is investigated and a preferred meshing strategy is discussed. It is shown how the MMS is used to detect the presence of a coding/conceptual error. The method is then used to show how the origin of an error might be detected. The use of the method for a simple regression test is also investigated. Finally the lack of interaction between a selected manufactured solution and the applicable geometry is discussed.

Subsequently, the application of the MMS to the MoM is discussed in Chapter 5. When applying the MMS to the MoM a number of difficulties are introduced. The selected MS is now intimately tied to the selected geometry because general volumetric source terms are not formulated for the MoM. As such, the solution must obey some realistic requirements imposed by physics on the current. Furthermore the expected convergence rates for the MoM is not widely available in accessible texts on the MoM from an electromagnetics perspective. An in-depth study was necessary to determine which convergence rates have been mathematically proven. The applicability of these convergence rates are also discussed in terms of selected manufactured solutions. The MoM is an intrinsically integral method and therefore numerical integration is necessary for the computation of the driving terms in Step 2 above. This integration

involves the handling of singularities in integrands of type $1/R$ and $1/R^2$ on triangular domains. Integration of these singularities proved to be quite difficult and is currently an active research topic in the electromagnetics community. The ability to use the MMS is practically shown in results presented in Chapter 5. This chapter is finally concluded with Mutation Testing results.

A key contribution of this work includes the investigation of the applicability of the MMS to full-wave electromagnetic numerical techniques. More importantly, this method has been extended to the MoM and it has been shown that the method is indeed usable in that regard. Key contributions are also discussed in each chapter as relevant.

This work should be relevant to researchers as well as commercial code developers. The author believes that the method will serve well as a regression testing technique due to the ability to represent various numerical features in a relatively small problem. This method should also be applicable to robustness testing because a problem of arbitrary size could be generated using this method.

# Chapter 2

# Validation, Verification and the Method of Manufactured Solutions

The role that the Method of Manufactured Solutions fills in verification and validation is discussed here. The definition and use of the words *verification* and *validation* must however first be clarified. The definition of these terms and the processes that are entailed by each are discussed in this section. The Method of Manufactured Solutions (MMS) is finally introduced here.

## 2.1 What is Validation and Verification: Language becomes a barrier

Different terminology has arisen in different engineering community dealing with essentially the same type of issues with validation and verification. Most engineering communities engage in the numerical approximation of some sort of PDE equation. These PDE equations arise from the study of physical phenomena such as electromagnetic radiation, fluid dynamic processes and structural behaviour [3]. All of these engineering communities are interested in the validity of results computed using codes and techniques approximating PDEs. In this work the term Verification and Validation shall refer to the overall set of activities that tries to answer the question of correctness of results and shall be abbreviated as V&V. When there are referred to *engineering fields* or *engineering* it should be interpreted as meaning engineering fields that are interested in the numerical approximation to systems that can be described using PDEs.

The community that first started addressing V&V is the operations research community as related by Oberkampf [59]. However the first community that addressed V&V from the engineering fields is the computational fluid dynamics (CFD) community as related in [59], [70] and [72]. The CFD com-

munities has a large body of work relating to V&V due to their early start in the field. To the author's knowledge the first book on the subject from an engineering standpoint comes from Roache [70] and a newer edition by the same author [69].

The electronic engineering community has also been interested in validation and verification and to this end has developed IEEE Standard 1597 [38, 39]. This standard has been initiated by the IEEE EMC and ACES societies. IEEE Standard 1597 [38, 39] has a strong influence from the EMC community and a large section thereof consists of methods used to compare measured and computed results. This standard will be discussed in more detail when model validation is discussed later on.

The terms validation and verification are often used in different contexts and with different meanings. The meaning of *verification* and *validation* defined by the IEEE [37] and paraphrased by Oberkampf [59] is as follows (emphasis added by the current author):

**Verification** the process of evaluating the products of a software development phase to provide assurance that they meet *the requirements defined for them by the previous phase.*

**Validation** the process of testing a computer program and evaluating the results to *ensure compliance with specific requirements.*

These definitions are fully general and use referential phrases emphasised in the above definitions. These general definitions are not very meaningful in themselves and the context of the current work require further clarification.

The definitions used in this work however are those used by the American Institute of Aeronautics and Astronautics (AIAA) and are as follows [60]:

**Verification** the process of determining that a model implementation accurately represents the developer's conceptual description of the model and solution to the model.

**Validation** the process of determining the degree to which a model is an accurate representation of the real world from the perspective of the intended uses of the model.

In short the term *verification* can be described as "are we solving the equations correctly for the model?" and *validation* can be described as "are the model and equations correct when comparing against physical reality?".

The different activities associated with V&V are discussed in what follows. The processes outlined by Roach [69, 71], Salari and Knupp [72], Oberkampf and Trucano [59] and the author's own will be discussed here. The processes and definitions used by IEEE Standard 1597 [38, 39] will be contrasted and compared with the processes and definitions used here.

The act of V&V is an overarching term for a set of distinct activities. The three activities as described by Roache [71] are verification of code, verification of calculations and validation of results to physical reality. The processes discussed in IEEE Standard 1597 adds to this the validation of technique but lumps verification of calculations and validation into one activity. In the present work the following list of activities are considered under the term V&V: Technique Validation, Mathematical Technique Verification, Code Verification, Calculations Verification, Model Validation and Validation of Governing Equations.

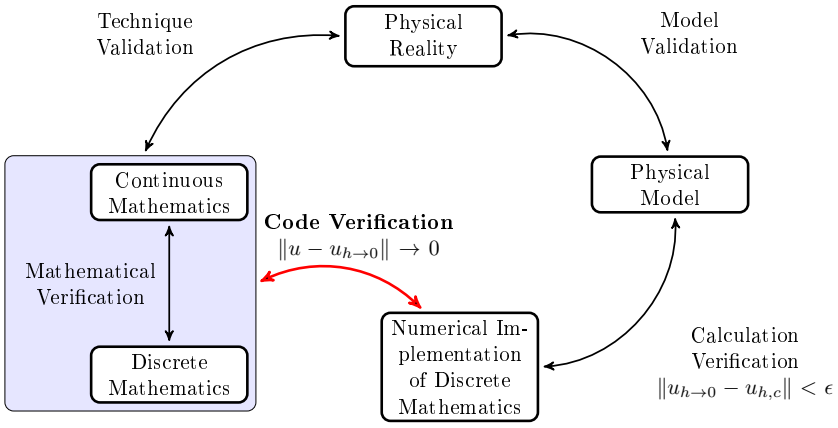



**Figure 2.1:** The relationship between verification and validation of mathematics, code, calculation and physics. Code Verification, the topic of this work, is written in bold. The equations shown in the diagram are explained in Section 2.2.

Each of these will be discussed in what follows and their relationship is shown graphically in Figure 2.1. There are various figures similar to Figure 2.1 in literature most notably [59], but this particular version is how the current author sees the interrelationship between these activities.

## Technique Verification and Validation

### Mathematical Technique Verification

This is the activity that is engaged in by mathematical analysts when trying to prove that the mathematical numerical techniques used does indeed solve the governing equations. In other words, analysts such as Buffa and Christiansen [13] try to answer the question of whether the MoM for the EFIE solves the scattering problem from Lipschitz screens in a consistent manner. Others include Bendali [7, 8], Christiansen [15], Bespalov and Heuer [9], Monk [54] and Hiptmair and Schwab [35].

### Technique Validation

This is an ongoing activity and is in effect done by all practitioners that use numerical methods and compare them to physical results every day. This is what the IEEE Standard 1597 calls technique validation but it is stated there as the "Mathematical level" of validation.

## Code Verification

This is the activity of showing that the code that was implemented is indeed following the mathematical process as stated by the code developers. This does not say anything about the validity of results generated by any specific model, but just states that the code implements the stated discretised methods. This is the main focus of the technique discussed in this work and will be expanded on subsequently.

## Calculation Verification

This is the activity that has to be engaged in by the user of the numerical code. This does not yet answer the question whether the actual model that is implemented does indeed agree with physical reality. This activity shows that the model as implemented is consistent and that numerical techniques such as domain truncation are used correctly. The following are examples of questions that have to be answered by the practitioner:

- Do the results change when refining the mesh? In other words is the problem being solved in the asymptotic region?
- Do the results change when the absorbing boundary used is moved further away from the structure? This is typical of FEM or FDTD type codes.

The aim of asking these questions is to ensure that the model applies numerical techniques as theoretically required by the techniques involved.

## Model Validation

With this activity the code user ideally compare simulated results with physical reality. This is the most difficult activity for a numerical code user. Measurements are compared to calculated values from simulation. This is also the main concern of IEEE Standard 1597 [38, 39].

### IEEE Standard 1597 [38, 39]

IEEE Standard 1597 (called *standard* in this section) will briefly be discussed here as model validation is the main concern of the standard. The standard

consists of a set of analytical, accurately measured and accurately computed benchmark problems. A substantial part of the standard is also devoted to a method called Feature Selective Validation (FSV). Briefly FSV is a technique that is used to compare complex measurement results and simulated results. The technique enables a non-expert to compare results that can normally only be compared by an expert.

It is not always possible to compare simulated results to measured results and the standard suggests that users compare results obtained by using other numerical methods that use different physical techniques. For example, compare results obtained using a MoM code with those obtained using a FEM code. The present author would also suggest that tools from different vendors be used as different modeling environments increase the modelling diversity.

### Validation of the Governing Equations

This is validation in the classical physics sense. We are trying to answer whether Maxwell's Equations we are solving do indeed describe physical reality. Physicists and engineers use the governing equations in everyday tasks and physical experiments agree quite well with reality.

The answer to this is the concern of physics and engineering. Results

## 2.2 Code and Calculation Error

Next the definition and sources of error will be discussed to clarify and introduce some notation.

Errors in code can be classified in two categories, namely *acknowledged error* and *unacknowledged error*. The term unacknowledged error is quite descriptive in specifying that the error is not known. The most notable contribution to this error is coding error that produce results that are seemingly correct but is actually very inaccurate or incorrect. Unacknowledged errors should be reduced to a minimum. Acknowledged errors on the other hand are errors present in the solution that are known but acceptable. Examples of this type of errors are numerical rounding errors introduced by finite precision and discretisation errors. Discretisation error is discussed as it leads to a practical method to investigate the presence of unacknowledged errors.

### Discretisation

The numerical solution to a CEM problem is the results of a continuous operator equation that has been discretised. The operators equations discussed here are the BVP shown in Appendix E.1 for the FEM and the integral equation shown in Appendix F.2 for the MoM. The difference between the discretised solution and the continuous solution is known as the *discretisation error*. Dis-

cretisation error is typically denoted as $\|u-u_h\|$ with $u$ the continuous solution and $u_h$ the discretised solution and $h$ an indication of mesh element size. If this discretisation error tends to zero as the meshing size is decreased, that is $\|u-u_h\| \to 0$ as $h \to 0$, the solution is *consistent*. If the rate of discretisation error improvement can be expressed as

$$\|u-u_h\| \leq Ch^s \tag{2.1}$$

for some C independent of $h$, then the order of accuracy is said to be of order $s$, e.g. 1st order, 2nd order. The exact error is however dependent on a specific problem.

### Expression of error

The expression of error in CEM is as mentioned in Section 2.2

$$E = \|u-u_h\|.$$

Oberkampf and Trucano [59] introduces a further concept in the representation of error. Let $u_{h\to 0}$ be the exact solution of the discretised operator equations in the limit where $h \to 0$. Using the triangle inequality, an estimation of $E$ can be written as

$$E \leq \|u-u_{h\to 0}\| + \|u_{h\to 0}-u_{h,c}\| < \epsilon, \tag{2.2}$$

where $u_{h,c}$ is the solution of a particular discretisation on a particular computer $c$. $\epsilon$ is an arbitrary small number that is typially determined by the choice of the accuracy bound for the numerical solution. Norm symbols in Eq. 2.2 are only suggestive of differences between values and does not refer to specific norms. It can be argued that the first term of Eq. 2.2 is equal to zero for strongly consistent methods. Ensuring that this term is zero is primarily the concern of numerical analysts, algorithm developers and mathematicians. Consistency might not be the case for strongly coupled multiphysics problems but is certainly the case for the methods and physical systems discussed here as proven by analysts amongst others [9, 13, 54]. The second part of Eq. 2.2 is quite another situation. It is this term that represents the discretisation error, the numerical rounding error and other algorithmic errors that are present for the particular problem under investigation. This error can often not be reduced using for instance a finer mesh due to resource constraints. It is the value of this error indicator that is the subject of Model Validation and Verification.

## 2.3 Code Verification

Code verification will now be discussed in more detail because it is the specific V&V activity that is the topic of this work. The aim of code verification as

mentioned above in Section 2.1 is to show that the code implemented does indeed solve the stated equations. Code verification like system validation can be seen as an ongoing process in that confidence is gained as a technique is more and more used successfully. Also, similar to validation, a piece of code cannot be *proved* correct but can only be shown to be incorrect [42]. However confidence is increasingly being built as experience is gained while using and testing the relevant code. Activities that increases a practitioner's confidence in the correctness of code [58,59,69,72] are (1) the convergence of relevant operators equations, (2) formalised tests used for verification, (3) benchmarks used for accuracy quantification and (4) Software Quality Assurance techniques.

There is a line of thinking that intends to show that a piece of code is verified and that there is a point at which the process is terminated [69, 72]. Most practitioners would appreciate that there are no codes that are perfectly correct. It is the view of the author that verification is an ongoing process but that there is a point at which there is a level of high confidence in the accuracy of implemented code.

## Software Quality Assurance

Concepts used in Software Quality Assurance (SQA), also called *Software Quality Engineering* or just *Software Engineering*, are useful when showing correctness of code and are recognised by various authors some of which are [72], [59] and [69]. Conventional software quality assurance is concerned with processes (such as management, planning, acquisition, supply, development, operation and maintenance) as well as administration, reporting and document requirements [42,58]. The aspects of SQA that is important to verification of code as discussed here are the concepts of testing. These concepts are *static testing*, *dynamic testing* and *formal testing*. These testing regimes agree with typical coding mistakes made during development of code.

### Static Testing

Static code analysis is especially important for compiled languages such as C and FORTRAN. These languages allow the coder to execute syntactically correct code but that are not necessarily correct. Examples of mistakes that can be detected by static testing is the use of uninitialised variables.

Interpreted languages such as Python are dynamic by nature. They will only start to execute code that have correct syntax but due to the dynamic nature of the language errors that would be detected by a static analysis tool are detected at run time by the interpreter. Compared to the example mentioned above, the interpreter will issue an error if an uninitialised variable is used and execution will be stopped.

*Static mistakes* are those that can be detected using static testing techniques.

### Dynamic Testing

Code is dynamically tested when it is executed and computed results are compared to expected results. It is in this category that the MMS is most useful. A typical error that might be detected during dynamic testing is when the index of an array is out of bounds.

*Dynamic mistakes* are those mistakes that are detected during dynamic testing. More of interest are the dynamic mistakes that cause deviation from order-of-accuracy and invalidate the consistency of the method used. Oberkampf and Trucano [59] have argued that aspects that affect the order-of-accuracy amongst other things should be considered as a separate activity from SQA.

These are the mistakes that are the focus of the Method of Manufactured Solutions.

### Formal Testing

Formal testing is the execution of a set of tests that the customer and the developer agrees on to prove that the code does indeed fulfill the requirements. Typical tests that could be used for this purpose would be benchmarks as set by the IEEE Standard 1597 [38, 39]. The customer in this scenario is the engineer using computational codes.

*Formal mistakes* are those that are detected during formal testing, that is, the code does not do what the customer asks.

## Dynamic tests used for Code Verification

In this section, the methods used by engineering code developers during dynamic testing are discussed. Typical methods ranging from least accurate to arguably most accurate are: trend, symmetry, comparison of measurements and simulations, analytical solutions and the method of manufactured solutions. Our discussion of these methods is based on [69, 72]. Finally it should be noted that [80] adds to this the conservation of various physical quantities such as energy and reciprocity but, not all of these are guaranteed to be retained during simulation or are only approximated.

### Trend

With trend testing a code for which the result is not know is run. Quantities in the simulated model are changed between simulations and derived results are compared; if the derived results have the expected trend then the code is considered to be correct. An example of trend testing is to decrease the distance between two parallel plates and to compare the capacitance computed. If the capacitance decreases as the distance between the plates is increased then the code is said to pass the test. This method however relies on *expert judgement*. The problem with this method is that even faulty code might

show the expected trend and in the absence of known correct results cannot be decisive for correctness.

### Symmetry

With symmetry testing expected symmetries are exploited to make conclusions about the correctness of code. Symmetry is subdivided by Salari [72] into three cases. (a) Expected physical symmetry : if the physical layout has symmetries and should induce symmetrical simulated results then these should be apparent. (b) Translated and rotational symmetry : if the geometry is translated or rotated in the physical domain then the result should be the same. (c) A symmetrical simulation in 3D is constructed in such a way that the 2D version of the same problem can be compared to a sectional cut of the 3D results.

### Comparison

Comparison Testing compares results gained through other methods with the calculated result to. In order to do comparisons between different results reliably expert judgement is required. As mentioned in Section 2.1 the FSV helps with comparison between difficult to compare data sets. As will be discussed below, the improved ease and systematic comparison does not increase the reliability of comparison testing as a code verification technique.

**Measurements** Accurate comparison between measurement and simulation is often difficult. The difference between simulation and experimental setup is often not clear. Perfect conditions that are implementable in simulation can often not be realised in an experimental setup. Differences often occur between simulated and measured results. The differences between these two are then dealt with by one of the following methods: the computational model is tweaked such that measured and simulated results agree better, the differences are explained away by referring to uncertainties in measurements or the person doing the comparison accuses the person that did the measurements of not doing them properly. This situation is clearly not adequate to verify code as correct. Measurements do have an important place in V&V and that is to validate either the model or the method.

**Simulations** Simulations using different numerical techniques or the same technique of a trusted code base can be used to compare simulated results. Comparison between two different numerical techniques may introduce their own difficulties in comparison as results obtained for complex models often do not agree exactly. Techniques such as the FSV can be employed to aid comparison but discrepancies between results might also be difficult to explain.

The idea to compare results from two simulations is not unique to the field of computational engineering. The Computer Science community frequently compares different codes solving the same problem to assess accuracy. Experiments conducted by Knight and Leveson [47] used 27 different versions of code developed independently by coders at two universities. The 27 different code versions were subjected to 1 million tests. Each version of the code would participate in a voting process and the aggregated voting result would be used in each of the tests. It was found that there was a large number of tests where a large number of programs failed even though the codes were individually reliable. However Adams and Taha [1] performed a similar experiment using code developed using different programming methodologies. It was found that diverse methodologies did indeed increase reliability when comparing between codes. It is therefore stressed that different solution methods should be used when comparing between results. If it is known that a specific code using a specific technique is verified and accurate then the *different technique* requirement becomes less important for code verification.

The method of comparing results from more than one numerical technique is also endorsed by the IEEE Standard 1597. However this method of comparison is endorsed there as a method to validate a model and not to verify code.

### Analytical Solutions

Analytical Solutions is the method mostly used practically to verify code by code developers and theoretical analysts. The method of testing is also called *Method of Exact Solutions* by Salari and Knupp [72]. This work will show by using mutation testing, that analytical solutions do indeed provide a good way to verify code as a first iteration. There are however also problems with using analytical solutions for code comparison. One example of such problems is encountered when computing the scattering from a PEC sphere: there are infinite sums that have to be truncated and if care is not taken with implementation this may lead to inaccurate comparison. The biggest problem with analytical solutions however is that they are quite limited in scope. Studying a recently revised standard textbook on EM [6] reveals quite a limited number of solutions.

### The Method of Manufactured Solutions

The MMS is a method by which a very large set of solutions can be generated against which to test code. The details of the MMS will be discussed in Section 2.4 below. The MMS is considered as the best way to verify code used for numerical methods in approximating PDEs [24, 58, 59, 69, 71]. It is the application of this method to electromagnetics that will be discussed and expanded upon in this work.

## 2.4 The Method of Manufactured Solutions

A general overview of the Method of Manufactured Solutions will now be outlined. More details for the MMS in context of the FEM and MoM will be given in Chapters 4 and 5 respectively. As related by Oberkampf [59], the idea of a Manufactured Solution (MS) for code debugging was first introduced by [74] but the combination of a MS and a grid convergence study to verify code using an order of accuracy method was first used by Steinberg and Roache [69, 77]. As related in [69], the term *Method of Manufactured Solutions* was first used by Oberkampf *et al.* [2].

The basic idea of the MMS can be described succinctly as *solving the problem by starting at the end*. With this is meant that the solution that is sought by the code under test (CUT) is selected by the user and not determined by the boundary conditions and geometry of the model under test. In symbolic terms, the process can be described following [71]. Consider the problem of finding the unknown function $u$ for a given operator $\mathcal{L}$ such that

$$\mathcal{L}[u(x, y, z, t)] = 0, \tag{2.3}$$

with sufficient extra information such as boundary conditions. Subsequently, choose a manufactured solution written as

$$u' = M(x, y, z, t). \tag{2.4}$$

Now modify the original problem to produce the new operator $\mathcal{L}'$ such that

$$\mathcal{L}'[u'(x, y, z, t)] = 0, \tag{2.5}$$

where the solution is the known solution $u'$. The most obvious way to select $\mathcal{L}'$ is to subtract the manufactured solution from the original problem:

$$\mathcal{L}' = \mathcal{L} - \mathcal{L}[M]. \tag{2.6}$$

All boundary condition values can be computed by applying the relevant operators for the problem to the selected solution.

The CUT is now executed with the new operator equation, or as shall be shown later, with the original operator equations but with source terms added that renders the original operator $\mathcal{L}$ equivalent to the new operator $\mathcal{L}'$.

A convergence study is performed with the CUT using the selected MS and practical convergence rates are compared to theoretical convergence rates. If the convergence rate achieved using the MS is what is considered correct for the method and setup then it is concluded that this specific test has passed.

This section will conclude with an example of the MMS applied to the electric field FEM.

## An FEM example

The MMS will be demonstrated using the continuous operator equation solved when applying the E-field FEM.

The operator equation solved using the BVP Eqs. E.1 for the FEM is the Vector Wave Eq. B.13 and related boundary conditions; it is repeated here for convenience. Solve

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} - k_0^2 \epsilon_r \boldsymbol{E} + j k_0 Z_0 \boldsymbol{J} = 0. \tag{2.7}$$

for the impressed current ($\boldsymbol{J}$) on the problem domain $\Omega$ with the associated boundary conditions

$$\hat{n} \times \boldsymbol{E} = 0 \tag{2.8}$$

on $\Gamma_D$, the Dirichlet boundary, and

$$\hat{n} \times \nabla \times \boldsymbol{E} = \boldsymbol{N} \tag{2.9}$$

on $\Gamma_N$, the Neumann boundary.

The FEM method is described in detail in Appendix E.

The operator $\mathcal{L}(\boldsymbol{E})$ is then the first two terms of Eq. 2.7,

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} - k_0^2 \epsilon_r \boldsymbol{E}. \tag{2.10}$$

For this example the following MS is selected

$$\boldsymbol{E_U} = \tanh(k_0 x y) \hat{\boldsymbol{x}} \tag{2.11}$$

Applying the operator $\mathcal{L}$ to the selected MS [1] yields

$$\begin{aligned} \boldsymbol{J} = & \frac{j k_0 \tanh(k_0 x y)}{Z_0} \left[ \frac{2x^2}{\mu_r} (\tanh(k_0 x y)^2 - 1) + \epsilon_r \right] \hat{\boldsymbol{x}} \\ & \frac{-j(\tanh(k_0 x y)^2 - 1)}{Z_0 \mu_r} \left[ 2 k_0 x y \tanh(k_0 x y) - 1 \right] \hat{\boldsymbol{y}}. \end{aligned} \tag{2.12}$$

The result in Eq. 2.12 can be added to the Operator 2.7. This can also be achieved by setting

$$\mathcal{L}(\boldsymbol{E_U}) = j k_0 Z_0 \boldsymbol{J} \tag{2.13}$$

and computing $\boldsymbol{J}$. A general FEM code that implements arbitrary sources can be tested using MMS for FEM without further modification.

[1] Differentiation can easily be done using a symbolic tool such as the Python symbolic math tool, Sympy [78]

## Guidelines

The following set of guidelines for a MMS verification regime is as given by Salari and Knupp [72]. Here the viewpoint is taken that a specific MS can render a piece of code verified and it will be seen that full generality is required.

### Guidelines for selecting a MS

Any arbitrary MS may be selected, but some MS will be better than others. In this section a set of guidelines for selecting an MS is discussed. It should be noted that these are general guidelines; method specific guidelines will be discussed in the relevant chapters.

1. Use smooth analytic functions to build a smooth analytic MS. Easing the implementation and evaluation of the MS ensures that extra coding and rounding errors are not introduced when testing the CUT.

2. The MS should be general enough to exercise all the terms in the governing equations. In the FEM example discussed above, if the selected MS is curl-free then the section of the code that implements the generation of the stiffness matrix will not be thoroughly tested.

3. The MS should have a sufficient number of non-trivial derivatives. If some of the higher order derivatives are not exercised because lower order derivatives are zero or do not exist then the relevant part of the code that deals with implementing those features is not exercised.

4. The solutions should not prevent the code from finishing. Robustness is not the aim of code verification.

5. The MS should be defined on a connected subset of $\mathbb{R}^2$ or $\mathbb{R}^3$. This allows the tester to use an arbitrary geometry (as allowed by the CUT) when testing code.

It should also be noted that the guidelines presented here are for the strict task of code verification. Situations where these guidelines are deliberately disregarded to test certain aspects of the CUT will be discussed at a later stage.

It is *very important* to realise that physical reality is *not* a necessity when selecting a MS. The act of code verification is strictly to test the code's ability to solve the stated operator equation and in general it will employ solutions that are not physically realisable.

### Guidelines for selecting coefficients involved in the MS

Often constitutive properties are specified by the physics of the problem. Guidelines for selecting those constitutive properties will now be discussed.

These guidelines are applicable for non-constant constitutive values, if the CUT only deals with constant values then these guidelines does not apply.

1. Analytical functions should be used.
2. Non-trivial functions should be used and as general as required by the CUT.
3. The selected functions should allow nearly physical values for the properties they represent. The reason for this is that robustness of the code might be affected if for instance negative values for $\epsilon_r$ are chosen, if such values are not supported by the code.
4. If the properties may be differentiated in the code implementation then the functions used to represent them should be differentiable. If the code however deals efficiently with jump discontinuities as present in stratified media then these discontinuities may also be present.

### Guidelines for selecting a geometry

The most general geometry that can be implemented in a model by the CUT should be used to test the code. This means that it is *not* sufficient to verify code that implements a fully 3D MoM technique by only using a square plate in the x-y plane. Specific geometries however can be used to debug certain aspects of the code.

### Guidelines for selecting BCs.

The issue of BCs come into play once the geometry has been selected for a particular execution of the MS. The following should be kept in mind when generating an MS.

1. *BCs for a particular solution should be general.* If the code states that it implements a certain set of boundary conditions then these should be verified.
2. *The placement of BCs should be considered.* Different placements of BCs might influence the calculations of these. An example might be the calculation of specific Neumann boundaries values present because the calculation of the curl of the field is important in the implementation of these; as such, orientation of these Neumann boundaries might be important.

## Strengths and Limitations of the MMS

Strengths and limitations of the MMS listed in the literature [69, 72] are summarised here.

**Strengths**

1. *Most code capabilities can be verified with the MMS procedure.* The strength of the method comes from the fact that an arbitrary solution can be used and can be applied to an arbitrary geometry with great flexibility in the BCs used.

2. *The solution domain can be selected after the MS has been generated.* This allows flexibility for large amounts of geometries.

3. *Source terms can be computed using symbolic tools.* This is true for the FEM - but not for BEM methods.

4. The MS is composed of easily evaluable analytic functions.

5. The ability to choose the MS allows flexibility to debug code by a process of elimination.

6. The procedure is applicable to a large number of numerical techniques.

7. *The MMS is self-correcting.* If errors are introduced when implementing arbitrary source terms then these errors will be detected using the MMS.

**Limitations**

1. The method does not say anything about the accuracy of solutions generated by the CUT.

2. The MMS procedure is complicated when the implementation of a numerical technique lacks volumetric source terms. Also the MoM only implements source terms on the surface of the scatterer and not volumetric source terms in the surrounding free space and as such is significantly more restricted.

3. *Not all coding mistakes affect accuracy.* Efficiency mistakes for instance will not be detected by the method. This however does not prevent the method to be used as a method to show correctness.

4. Whether the solution procedure followed by the code is in effect the same as intended by the developer is not verified by the MMS.

5. The method does not say anything about individual terms of mixed-order methods.

## 2.5 Conclusion

In this section the concepts of and processes involved with Validation and Verification was discussed. V&V was subdivided into technique verification and validation, code verification, calculation verification, model validation and validation of governing equations. Code verification has been identified as the topic of this work. Various techniques used for code verification was discussed with the MMS being the technique that will be dealt with in the rest of this work. The basic concept of the MMS was discussed and further explained in context of a FEM example. Finally some general guidelines from literature was presented for the selection of a MS, coefficients, geometry and BCs. The general strengths and limitations of the MMS was also discussed.

Next the MMS will be discussed in context of the FEM and then in context of the MoM.

# Chapter 3

# Testing the test : Mutation testing

Any code verification or code test needs to be carefully selected to be an effective tool in terms of not just resources consumed but also in the ability to detect coding mistakes. The MMS is one of many techniques used to test code and its applicability relies on its ability to successfully detect coding mistakes. In this work, mutation testing will be used to investigate the efficacy of the testing method. The concept of mutation testing will be introduced here and used in each of the subsequent chapters to investigate the MMS.

Mutation testing was introduced respectively by DeMillo *et al.* [18] and Hamlet [31] in the late 1970s as a method to investigate the efficacy of software testing in detecting coding errors. The method is extensively used in the computer science field when investigating testing methods and their efficacy. The extensive use of the mutation testing method is evident from the recent (2011) review study done by Jia and Harman [40]. Mutation testing, its applicability, verification of its fundamental principles and optimisation have been done using various code examples, from simple laboratory programs to complex real world programs such SPACE, a European Space Agency program [4].

It is important to realise that mutation testing does not test a code but instead evaluates the ability of a test to test a piece of code. The code that is tested by the test set will be known as the code under test. Now mutation testing can be summarized as follows. A test set is tested by making small changes to the CUT. If the test set detects the small change and all small changes made during mutation testing then the test set is said to be sufficient. If the test set fails to detect a mutation of the CUT then the test set fails the mutation test.

In the rest of this chapter the principles, methods and issues involved in mutation testing will be discussed.

## 3.1 Principals of mutation testing

Mutation testing relies on two basic principles as outlined by DeMillo [18]. These two principles are:

**The principle of a competent coder (PCC)** Code is not just a random selection of statements but are coded by a competent programmer and is thus close to the correct code.

**Coupling effect** Simple faults are coupled to more complex faults and the detection of simple faults is indicative of detection of more complex faults.

Offutt [62] has defined simple faults as those that can be fixed with only one mutation and complex faults as those requiring more than one mutation to be fixed. The more contentious principle is the coupling effect and extensive practical and theoretical work have been done validating this principle, [61, 62] amongst others. Jia and Harman [40] can be consulted for an extensive bibliography.

## 3.2 Mutation operators

Mutation testing generates a large set of code versions, each known as a mutant. The CUT will be denoted by $P_0$ and the set of code mutations is identified by $P_M$. A specific mutant is then denoted by $P_i$ where $i = 1...m$ and $m$ is the number of mutants in the set $P_M$. Not all possible mutations of code is a valid compilable/executable version and these are excluded from the set of mutants.

Mutation operators have been introduced to ensure that valid mutants are created in an orderly and repeatable manner [5]. In this work, numerical computation is investigated and mutation tests are limited to the numerical computation section of the code. For the previously stated reason, the operator classes discussed here will be limited to the typical mistakes made in procedural coding as typical in the C-language [5]. Operators are classified by the general class of mutations that they introduce. The mutation operator classes used in this work will be listed next; an example of each operator class is given in Table 3.1. In the following the operator class will be represented by the character in parenthesis.

**Statement (S)** This affects general statements in a code. This includes deletion of statements, position of return statements in code and code groupings amongst others.

**Variable (V)** Variable mutants typically interchanges variables in code, multidimensional array structure might be interchanged and access to subkeys of object like variables are removed or interchanged.

**Operator (O)** Operator mutants interchange mathematical and logical operators.

**Constant (C)** Constants in code is typically replaced by other values. One example operator changes any constant in code by one in the set $\{0.0, 1.0, -1.0, \text{user defined real}\}$.

A full list of mutation operators is given for the C-language by [5].

Other classes of operators not used here, include *interface* operators. Interface operators typically mutate the parameters received and passed to called procedures.

The code tested here is coded in a dynamic object oriented language Python, but the code being investigated is coded in a typical procedural fashion. Mutation testing serves in this work as an indication of method usefulness, but not as an exhaustive test of the complete code coverage of the specific code used. There are mutation operators that are applicable to more dynamic language features but research is more limited for these.

**Table 3.1:** Mutation operator examples

| Class | Specific operator | Explanation | Original program $P_0$ | Mutated program $P_i$ |
|---|---|---|---|---|
| Statement | SSOM | Change the sequence in which statements are executed | ...<br>b = b + 1<br>a = 3 + b<br>... | ...<br>a = 3 + b<br>b = b + 1<br>... |
| Variable | VTWD | Change the value of variable by a small amount | ...<br>a = 3 + b<br>... | ...<br>b = b + 1<br>a = 3 + b<br>... |
| Operator | OAAN | Mutate binary arithmetic operators to another arithmetic operators | ...<br>a = 3 + b<br>... | ...<br>a = 3 - b<br>... |
| Constant | CRCR | Replace constants by 0, 1.0, -1.0 or by a defined real constant | ...<br>a = 3 + b<br>... | ...<br>a = 1 + b<br>... |

## 3.3 Computational issues of mutation testing

There are a few limitations associated with mutation testing and these are discussed in the sections that follow.

### A large set of mutants

It should be relatively intuitive that a small piece of code together with all defined mutation operators will generate a large number of mutants. A large number of mutants can run for a very long time and hence make mutation testing impractical for complex pieces of code. Extensive research [40], most recently by Namin *et al.* [55], has been done to reduce the number of mutants required to test a code test. Namin *et al.* [55] found a set of 28 operators that is as effective as a full set of mutation operators but that results in a 92% reduction in resultant mutants. These results are for mutation in C code but is here used in Python code and considered as effective due to the similar arithmetic nature of the tested code. As such, the reduced set of 28 operators is used here due to the impracticality of using the full set of mutation operators. Table 3.2 list the 28 essential operators plus an extra equivalent operator as applicable to Python, highlighted below.

Table 3.2: Essential mutation operators

| Operator | Description | Impl. | Notes |
|---|---|---|---|
| | **Statement operators** | | |
| STRI | Trap on *True*. Test coverage of if conditions | No | Serves as code coverage test |
| SGLR | Replace goto label | No | Goto statements not used in Python |
| SWDD | Replace while-do with do-while | No | do-while not applicable to Python only while-do |
| SMTC | Mutate loop execution number | No | Serves as code coverage test |
| SMVB | Move brace up or down | Yes | |
| SSWM | Switch the order of case statements | No | Implemented new equivalent operator SIEM that switches if/elif/else statements |
| | **Operator operators - binary** | | |
| OAAN | Interchange between arithmetic operators | Yes | |
| OABN | Change from arithmetic to bitwise operators | Yes | |
| OALN | Change from arithmetic to logic operators | Yes | |
| OBBN | Interchange between bitwise operators | Yes | |
| OBSN | Change from bitwise to shift operators | Yes | |
| OLAN | Change from logic to arithmetic operators | Yes | |
| OLBN | Change from logic to bitwise operators | Yes | |
| OLLN | Interchange between logic operators | Yes | |
| OLSN | Change from logic to shift operators | Yes | |
| ORSN | Change from assignment to shift operators | Yes | |
| | **Operator operators - unary and assignment** | | |
| OAEA | Interchange arithmetic assignment | Yes | |

Continued on next page

Table 3.2:

| **Operator** | **Description** | **Impl.** | **Notes** |
|---|---|---|---|
| | for plain assignment | | |
| Oido | Increase/Decrease value | No | Operators are not applicable to Python |
| OLNG | Logical negation of variable | Yes | |
| OCNG | Logical negation of entire test statement | Yes | |
| OBNG | Bitwise negation | No | Not present in any code implemented |
| OCOR | Cast variable type | Yes | Only applicable in Python to division |
| | **Interface operators** | | |
| IndVarAriNeg | Inserts arithmetic negation at non interface variables | Yes | |
| IndVarBitNeg | Inserts bit negation at non interface variables | Yes | |
| RetStaDel | Deletes return statement | No | No conditional returns in code and Python default return value (None) is not numerically confusable. |
| ArgDel | Deletes arguments in function call | Yes | |
| **KwargDel** | Deletes keyword arguments in function call | Yes | This was added for Python as keyword arguments are not present in the C-language, but plays a similar role to arguments for ArgDel. |
| ArgLogNeg | Inserts argument logical negation | No | Logical arguments not used in code |
| | **Variable operators** | | |
| VGPR | Mutate global pointer reference | No | Python code does not use pointers |

### Equivalent mutants

Some mutants, known as equivalent mutants, produce output that is indistinguishable to that of the unmutated code. The following illustrates this. If the original code,

```
i = 0
while i < 10:
    i += 1
    ...
```

is mutated to

```
i = 0
while i <> 10:
    i += 1
    ...
```

then the value of `i` in the while loop and the value at which the loop is terminated, will be the same for both code versions. It is clear that no test will detect the difference between these two equivalent code versions (unless `i` is changed elsewhere in the loop). There is a fairly large amount of work that has been done to detect equivalent mutants [40]. However in this work equivalent mutants were checked by hand as the amount produced was not prohibitive to successful testing.

## 3.4 MutantPie

Mutation testing is a general exercise used by various software practitioners. In the interest of further research the code produced for mutation testing was released as an open source package, namely MutantPie. This package does not rely on the availability of the actual source code that has to be tested and mutated. The code object as used by Python is used to generate a representative source listing and this is used to produce various mutants. The advantages of this is that the practitioner does not have to have access to the actual source code that is being tested, but just the code object. A further advantage is that code can be mutated dynamically as code objects in memory so that code as used by external programs is not affected by mutation testing.

MutantPie implements the four classes of operators, statement, variable, operator and constant, the essential interface operators are also implemented. MutantPie is written in such a fashion as to encourage further development and expansion.

MutantPie can be accessed from https://launchpad.net/mutantpie.

# Chapter 4

# The Method of Manufactured Solutions : The Finite Element Method

The MMS will now be investigated for EM problems solved by the FEM method. The FEM is the natural setting where the MMS has been developed by Roache [69, 71, 77]. The method has however not been used in the EM field to the knowledge of the author. The FEM will briefly be stated, then theoretical convergence rates will be discussed. This is followed by a discussion of the FEM software used and meshing techniques required. Subsequently the practical ability of the MMS to detect errors will be shown by way of two real errors detected by the author and by way of mutation testing.

## 4.1 The MMS applied to the FEM

The boundary value problem being solved by the E-field FEM is discussed here. The governing PDE is the vector wave equation given in Appendix E.1 and is repeated here for convenience,

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} - k_0^2 \epsilon_r \boldsymbol{E} + j k_0 Z_0 \boldsymbol{J} = 0 \text{ in } \Omega. \tag{4.1}$$

The appropriate boundary conditions are

$$\hat{\boldsymbol{n}} \times \boldsymbol{E} = \boldsymbol{P} \qquad \text{on } \textit{Dirichlet} \text{ boundaries } (\Gamma_D) \tag{4.2}$$

$$\hat{\boldsymbol{n}} \times \nabla \times \boldsymbol{E} = \boldsymbol{U} \qquad \text{on } \textit{Neumann} \text{ boundaries } (\Gamma_N), \tag{4.3}$$

with $\Gamma = \Gamma_D \cap \Gamma_N = \partial\Omega$ and $\Gamma_D \cap \Gamma_N = \emptyset$.

One possible MS has already been discussed in Section 2.4. The current driving terms were calculated and shown there. As mentioned in Section 2.4 the calculation of the current ($\boldsymbol{J}$), *Dirichlet*- ($\boldsymbol{P}$) and *Neumann*-

boundary conditions ($\boldsymbol{U}$) are calculated using numerical symbolic packages such as Sympy [78].

Selection of a MS for the FEM is not significantly constrained. It is desirable that the E-field is selected in the appropriate function space [36, 54],

$$X^s(\Omega) = \{\boldsymbol{u} \in \boldsymbol{H}^{\boldsymbol{s}}_{\text{curl}}(\Omega) | \hat{\boldsymbol{n}} \times \boldsymbol{u} \in (L^2(\Gamma_D))^3 \text{ on } \Gamma_D \quad \text{and} \quad \hat{\boldsymbol{n}} \times \hat{\boldsymbol{n}} \times \boldsymbol{u} \in (L^2(\Gamma_N))^3 \text{ on } \Gamma_N\},$$

for $s \geq 1/2$. Monk [54] defines this space with $\hat{\boldsymbol{n}} \times \boldsymbol{u} = 0$ because this is the essential condition imposed by perfectly conducting boundaries; here the selection of a MS enables solutions where this strict condition is not the case. The previously mentioned relaxation of the BC requirement does not influence the convergence results proven in [54] if this condition is kept consistent throughout. The requirement that $s \geq 1/2$ ensures the applicability of the appropriate convergence results.

The fact that volumetric currents can be represented numerically, albeit not physically, eases the implementation of the MMS. Currents that are limited to only conductive structures would make the arbitrary selection of an E-field MS intimately tied to the structure that is used for the MMS. This will be seen when the MMS for the MoM is discussed in the next chapter.

The structure of the vector wave equation (4.1) is such that the selection of a MS can be used as an aid to debug the CUT. If a MS is selected such that,

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} = 0, \tag{4.4}$$

then the origin of a certain type of error can at least be eliminated from the calculation of the appropriate code components. A MS for which Eq. 4.4 is true shall be referred to as a *static solution* and solutions for which Eq. 4.4 does not hold shall be known as a *dynamic solution*. The ability to discern the location of code error will be investigated further on in this chapter.

## Convergence Rates

A knowledge of theoretical convergence rates is required in order to use the MMS effectively. Convergence rates that are available in fundamental texts (and their references) on the FEM such as [41] and [75] generally consider dispersion analysis of specific elements or the convergence of derived quantities such as the calculation of resonant frequencies. However, for the present work a general theory for convergence is necessary. Here only the cavity problem will be considered. Results are available in [54, Chapter 7.2] for a cavity with simple material parameters ($\epsilon_r$ and $\mu_r$ are constant throughout) and only one boundary condition throughout the geometry. In this case, the boundary condition used is $\hat{\boldsymbol{n}} \times \boldsymbol{E} = 0$ on $\Gamma = \partial\Omega$. Then for $h$ small enough and $\boldsymbol{E} \in H^s_{\text{curl}}$,

$$\|\boldsymbol{E} - \boldsymbol{E}_h\|_{H^0_{\text{curl}}} \leq Ch^s,$$

with $\frac{1}{2} < s \leq p$ and $p$ is the polynomial order of the vector finite elements used. General polyhedral boundaries will cause singularities that prevent high global regularity. As stated in [54] a convergence rate better than $O(h^{1/2})$ can be expected if the mesh can be refined strongly near the singularity. Fortunately, with the MMS, any solution can be selected and as such singularities can be avoided. If the handling of singularities is investigated then the appropriate measures must be taken if they are present in the MS. Convergence is also proven in [54, Chapter 7.3] for more general cavity structures and more complex material properties, is optimal and similar to the rates discussed here. Numerical results presented in [54, p.189] are however only measured in the $L^2$-norm and not in the $H^0_{\text{curl}}$-norm.

The graph norm used to calculate the error is defined as follows,

$$\|\boldsymbol{E}\|_{H^0_{\text{curl}}} = \sqrt{\|\boldsymbol{E}\|^2_{L^2} + \|\nabla \times \boldsymbol{E}\|^2_{L^2}}.$$

It is obvious that both the field and the curl of the field should converge at least at the prescribed rate of convergence. The $H^0_{\text{curl}}$-norm is therefore dominated by the worst performing constituent component. In results presented, the $L^2$-norm will be shown for the field *and* separately for the curl of the field. It should be remembered that the worst performing constituent error norm will dominate results.

The same convergence results are applicable for general scattering problems but with restrictions placed on material properties. These are discussed in [54, Chapter 10].

## Dolfin: An Open Source FEM code

In the current investigation, a stable FEM code base (namely Dolfin [19, 50]), is used as a CUT. This is a collection of highly optimised C-libraries with a Python interface that supplies the necessary machinery for a typical FEM implementation. The interface to Python enables the rapid development and investigation of FEM concepts. This code base is general to the extent that various computational fluid dynamic (CFD) problems and some EM problems are typically solved using it [49]. The Dolfin code base is unfortunately not implemented for general complex problems such as electromagnetic problems. The Stellenbosch University computational group (CEMAGG) have implemented an open source extension SUCEM-FEM [20, 63] to Dolfin that implements the complex math required to solve EM problems. This relatively stable code base is used as a starting point for the current MMS investigation. The stable code base enables the author to introduce known errors into the code and investigate the effect thereof on the convergence rate calculated.

## A note about meshing

In this section, a few brief comments will be made about meshing and refinement requirements. It is a well known fact [17, 41] that a regular mesh consisting of triangles or tetrahedra stacked in a specific repeating pattern will lead to varying anisotropic dispersion dependent on pattern and direction. This is mostly due to phase error that is incurred by the elements in the mesh [41]. A random mesh generated and optimised by well known tools such as `gmsh` [28] often leads to more accurate results using less elements due to cancellation of phase error.

The topic of this work however deals with error convergence rate and not absolute error. To study convergence rates it is preferable that the mesh set used is a sequence of refined meshes and that the quality of the mesh is not degraded as mesh elements are refined. Each subsequent mesh must introduce proportionally the same amount of phase error.

The above mentioned requirements are easily archived for 2D problems by simply subdividing a triangle element by halving each of its edges. For 3D meshes this becomes more difficult. By simply halving the edges of tetrahedra four sub-tetrahedra are formed and four tetrahedra in the center. The central tetrahedra however becomes more elongated and hence of lower quality. To overcome the quality issue of the central tetrahedra, a direction has to be chosen so as to keep the quality of the mesh high and as such sub cells of the refined mesh exhibit different error behaviour than their parent cells. A remedy to this is to restrict oneself to geometries that only consist of rectangular elements and using the meshing cell shown in Figure 4.1. This cell can be subdivided by simple subdividing of edges.

This requirement is not absolute to the functioning of the MMS but eases evaluation of convergence rates considerably as will be seen in the examples shown below. Overall convergence rates can still be ascertained from a general mesh set of good quality.

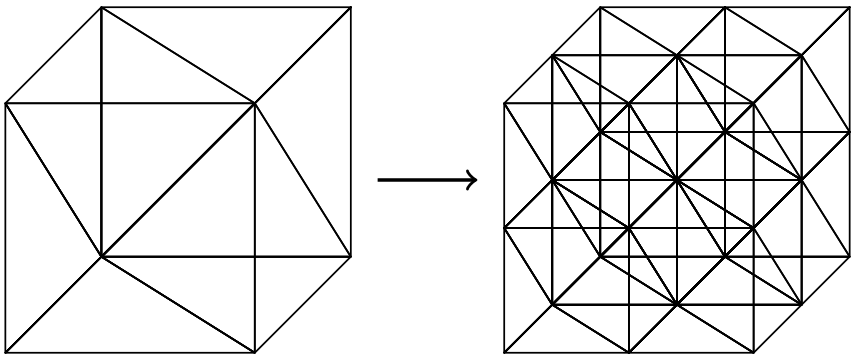


**Figure 4.1:** Unit cell used to mesh rectangular 3D structures. The refinement of this cell is highly regular and convergence rates achieved are close to theoretical rates.

## Numerical examples and proof of concept

### A 2D example

A basic flat 1m x 1m square PEC plate is used to look at the functioning of the MMS in 2D. The previously mentioned plate is similar to the plate presented in Figure 5.6 except that it is defined between 0.0m and 1.0m in both directions. Two MS are investigated. The first is a *static* solution,

$$\boldsymbol{E} = \sin(k_0(x+y))\hat{\boldsymbol{x}} + \sin(k_0(x+y))\hat{\boldsymbol{y}}, \tag{4.5}$$

with a Dirichlet boundary condition on two of the edges and a Neumann boundary condition on the other two edges.

The other is a *dynamic* solution,

$$\boldsymbol{E} = \tanh(k_0(xy))\hat{\boldsymbol{x}} + \tanh(k_0(xy))\hat{\boldsymbol{y}}, \tag{4.6}$$

with Dirichlet boundary conditions on all the edges. The reason for the boundary difference will become apparent in the next section. Figure 4.2 show the convergence results for the above mentioned MS. Note that correct convergence is obtained for all the orders of basis functions.

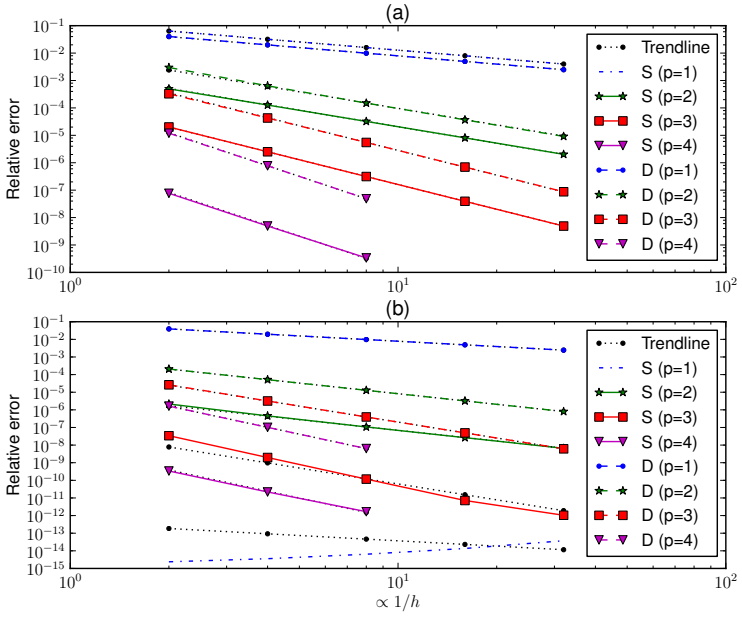



**Figure 4.2:** Convergence results using various orders of basis functions (represented by p) for the MS Eq. 4.5 (S) and Eq. 4.6 (D). (a) $L^2$-norm of $\boldsymbol{E}$ and (b) the $L^2$ norm of $\nabla \times \boldsymbol{E}$.

### A 3D example

A basic 1m x 1m x 1m cube will be used to look at the functioning of the MMS. Three of the sides of the cube are chosen as Dirichlet boundaries and the other three are chosen as Neumann boundaries. Two MS are investigated that are the natural extensions of Eq. 4.5 and Eq. 4.6. The first is,

$$\boldsymbol{E} = \sin(k_0(x+y+z))\hat{\boldsymbol{x}} + \sin(k_0(x+y+z))\hat{\boldsymbol{y}} + \sin(k_0(x+y+z))\hat{\boldsymbol{z}} \quad (4.7)$$

and the other is,

$$\boldsymbol{E} = \tanh(k_0(xyz))\hat{\boldsymbol{x}} + \tanh(k_0(xyz))\hat{\boldsymbol{y}} + \tanh(k_0(xyz))\hat{\boldsymbol{z}}. \quad (4.8)$$

Convergence results for the static example, Eq. 4.7, is given in Figures 4.3 and 4.4. The effect of meshing is clearly visible on the smoothness of the convergence rates observed. This effect is nothing new but is shown here to highlight the effect that needs to be taken into consideration. Results marked as Optimal Split (Opt. Split) are those generated using the cell discussed above. Results marked as Regenerated Mesh (Regen. Mesh) use a optimised regenerated mesh for each discretisation step. The results between the previous two mesh sets are the most favourable. Results marked as Mesh Split and Mesh Split Optimal (Mesh Split Opt.) are generated by simply subdividing a starting mesh and then optionally optimising the elements using algorithms available in `gmsh`. The choices of mesh refinement strategies are investigated because they represent the refinement strategies that are typically available to a practitioner when using standard tools. It is interesting to note the failure to converge of the mesh that was simply split. This failure to converge is because tetrahedrons generated by splitting other tetrahedrons become elongated and as such have low quality. This is in contrast to the 2D case where splitting a triangle results in triangles with similar shape to the original triangle. Note that the second order $\nabla \times \boldsymbol{E}$ results appear to converge sub-optimally. In fact, the observed convergence rate is 1.5 for complete second order elements and 2.5 for complete third order elements (not shown here).

The convergence rate obtained for the dynamic example Eq. 4.8 is shown in Figure 4.5. In this instance only the preferred meshing scheme is used. The results shown however are for a completely Dirichlet box and the split boundary box used in the static example. Correct convergence can be observed for the Dirichlet box but incorrect convergence is observed for the split boundary box. The reason for this incorrect convergence is due to the underlying FEM solution software (Dolfin) being used and represents the detection of a coding/conceptual error. This specific error is due to the incorrect integration along edges of Neumann boundaries for Nédélec's 1st elements of arbitrary order. This bug has been reported at `https://launchpad.net/dolfin` by the present author as Bug #1083092 and subsequently marked as not valid due to a conceptual error. However this highlighted the fact that it is not yet possible

to define the needed boundary condition as explained in the followup Question #205455 on the same website. Another important aspect to consider is the fact that the incorrect results presented in Figure 4.5 still converges albeit at a slower than expected rate and that one should be vigilant as to the exact computed rate.

What is particularly useful about using the MMS to debug code is that specific examples can be created where only one small aspect is changed between problems. A change might include the kind of boundary condition employed or the numerical values expected at the boundary.

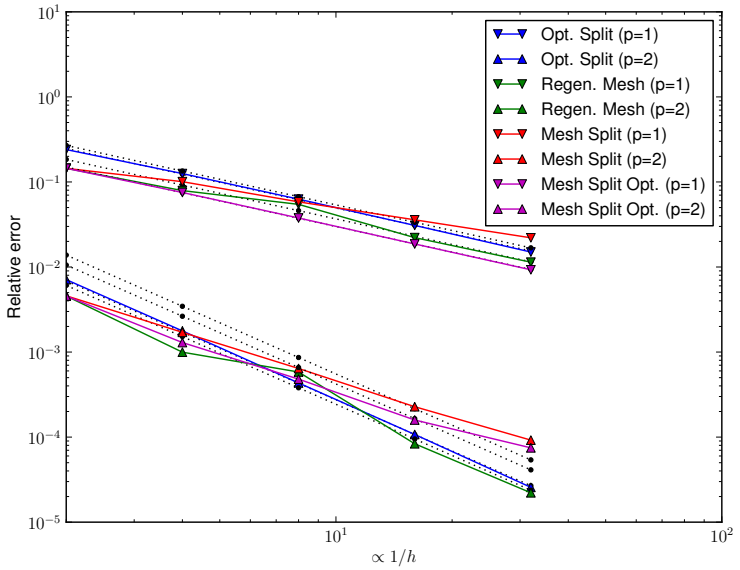



**Figure 4.3:** $L^2$-norm convergence results for the static MS Eq. 4.7. The effects of different meshes are clearly visible. The order of the basis functions used on the elements is represented by $p$. The black dotted line shows the expected convergence rate. In this figure $k_0 h$ ranges from 0.16 to 0.01.

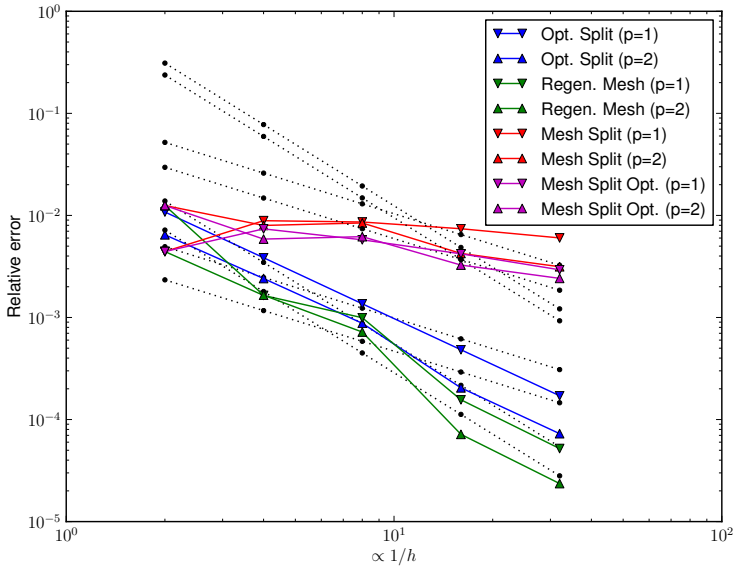



**Figure 4.4:** $L^2$-norm of $\nabla \times \boldsymbol{E}$ convergence results for the static MS Eq. 4.7. The order of the basis functions used on the elements is represented by $p$. The effects of different meshes are clearly visible. The black dotted line shows the expected convergence rate. In this figure $k_0 h$ ranges from 0.16 to 0.01.

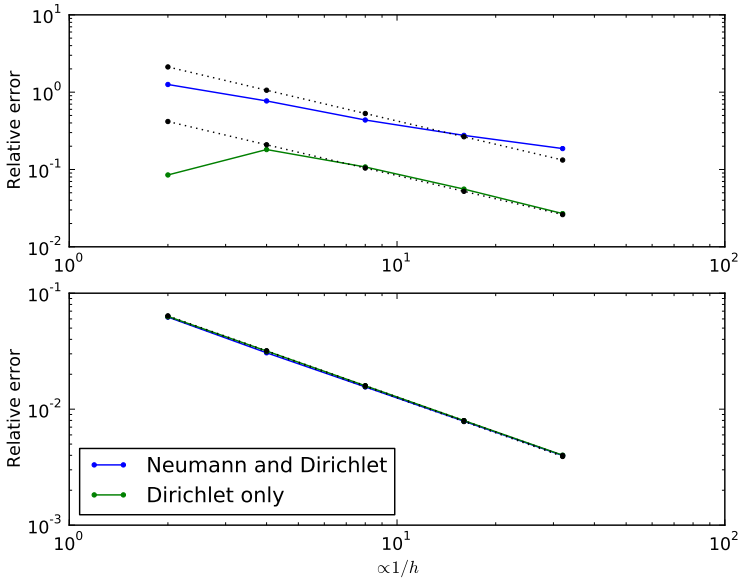



**Figure 4.5:** Convergence results for the dynamic MS Eq. 4.8. The error introduced by the faulty Neumann boundary condition is clearly visible. (a) $L^2$-norm of $\boldsymbol{E}$ and (b) the $L^2$ norm of $\nabla \times \boldsymbol{E}$. The black dotted line shows the expected convergence rate.

## Towards finding the origin of error

Here the use of a static and a dynamic equation to aid in the investigation of the origin of error will be investigated. It has already been seen in the previous section that the different equations can aid in the detection of error concerning different boundary conditions. We will demonstrate how these two different types of equations can be used to help find the origin of error in matrix assembly.

The two MS used in this instance are those used in Section 4.1.

In this section only first order elements were used and as such a convergence rate of $O(h)$ is expected.

An error was artificially introduced in the section of the code that is responsible for the assembling the mass matrix. In Figure 4.6 it is clearly visible that both the static and the dynamic solutions detect this error. Furthermore, an error was subsequently introduced into the code that assembles the stiffness matrix. It is also clear from Figure 4.6 that the static MS does not detect the error present in this case. This clearly demonstrates that it is possible to use specifically designed MS to detect the origin of error. It should however be kept in mind that it is theoretically possible that an error introduced in the assembling of the stiffness matrix could mimic the error introduced in the mass matrix and as such the origin of error cannot be determined absolutely.

## The MMS as a simple regression test

Software development often involves iterative changes to code. A different submodule might also be introduced in a larger code structure. This requires the constant retesting of a code base and the set of tests that is used in this process is known as regression tests. The following anecdote shows the usefulness of a simple MS on a simple geometry as a regression test. Previous results reported by the author [53] included the correct convergence results for the following MS,

$$\boldsymbol{E} = \tanh(k_0(xy))\hat{\boldsymbol{x}} + \tanh(k_0(xy))\hat{\boldsymbol{y}}. \tag{4.9}$$

These results were computed on a simple square geometry with a Dirichlet boundary condition on all four edges. The results obtained in [53] are reproduced here in Figure 4.7 for convenience.

The results reported in [53] are computed using Dolfin with an EM implementation developed by the author. Subsequently, the CEMAGG group at the University of Stellenbosch implemented SUCEM:FEM based on Dolfin as noted above, and the author switched to this implementation. However, a calculation of similar results reported above shown in Figure 4.8 demonstrates that results diverge for 1st order basis functions. This non-convergence represented the detection of an error in the implementation of Dirichlet boundary conditions by the MMS, and shows the usefulness of it as a simple regression test. It should also be noted that the higher order basis functions still

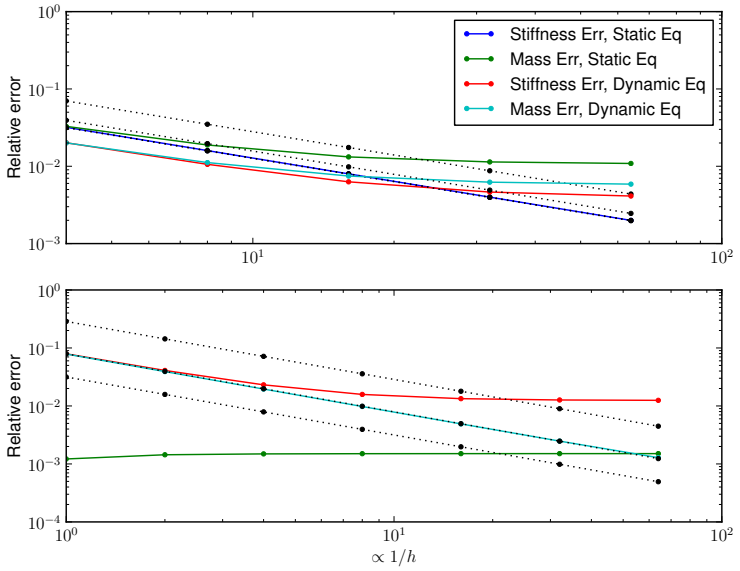



**Figure 4.6:** The convergence results obtained when using a static solution and a dynamic solution to determine the origin of code error. The static solution detects bugs that are present in the assembling of the mass matrix while the dynamic solution detects bugs that are present in the assembling of the mass and stiffness matrices. Axes shifted for result clarity. (a) $L^2$-norm of $\boldsymbol{E}$ and (b) the $L^2$ norm of $\nabla \times \boldsymbol{E}$, Static Eq. in the order of $10^{-15}$ and not shown.

converge for the incorrect implementation of the Dirichlet BCs. The author speculates that this is due to the current sources that are present throughout the geometry space and the added degrees of freedom available on the basis function. The fact that higher order basis functions do not show a deviation from the expected results should also serve as a caution of the limitations of the method.

## Geometric interactions and the MMS

The geometry present when considering electromagnetic problems have a profound impact on the characteristic fields present. This is equally true for driven problems and eigenvalue problems. If re-entrant corners are present in the geometry, as exemplified in Figure 4.9 then it is expected that a singular field be present at this corner [54, p. 76]. Consider, however, the results presented in Figure 4.10. These results show the convergence of a selected MS, Eq. 4.6 for first-order and second-order elements. It can clearly be seen that the results converge to the selected solution and that the expected singular effect of the re-entrant corner is not detected. The author speculates that this is due to the fact that the MMS imposes a volumetric current in the space surrounding

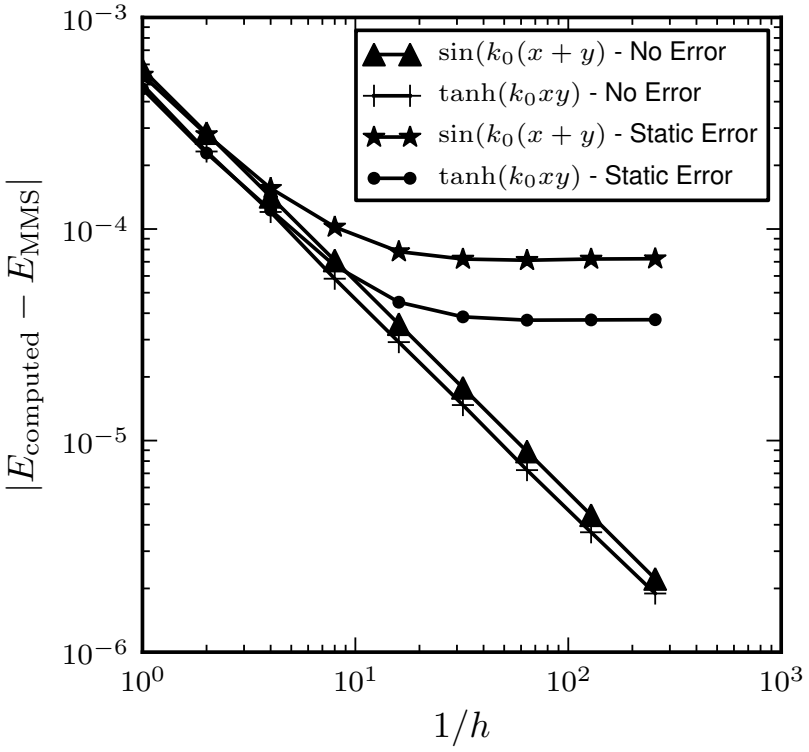



**Figure 4.7:** Previously correct $L^2$ convergence results of the E-field for Eq. 4.9. These results also show the effect of an error present in the calculation of the mass matrix for the FEM. Figure from [53]. © 2011 IEEE.

the re-entrant corner. This serves to demonstrate that the MMS tests the mathematical machinery of the implementation and not the physics that is simulated.

If however the CUT implements a strategy to deal with singular fields generated by re-entrant corners, then it is essential that the CUT be tested with a singular MS of the type expected.

From the above reasoning, the following should be evident: A well known, but different, source of solution error is often the modelling of curvilinear surfaces, such as the surface of a PEC sphere. However the MMS only measures the error incurred between the selected MS and the computed field *within* the prescribed boundary. If a sphere is modeled inaccurately the MMS will still show good convergence to the selected MS – because the *physics* of the problem is not considered.

This serves as a warning against the overconfident use of the MMS. It can be used to investigate the correctness of the mathematical implementation but not the correct physical representation of a problem. The well known phrase, "garbage in, garbage out" is often applicable to careless use of simulation code.

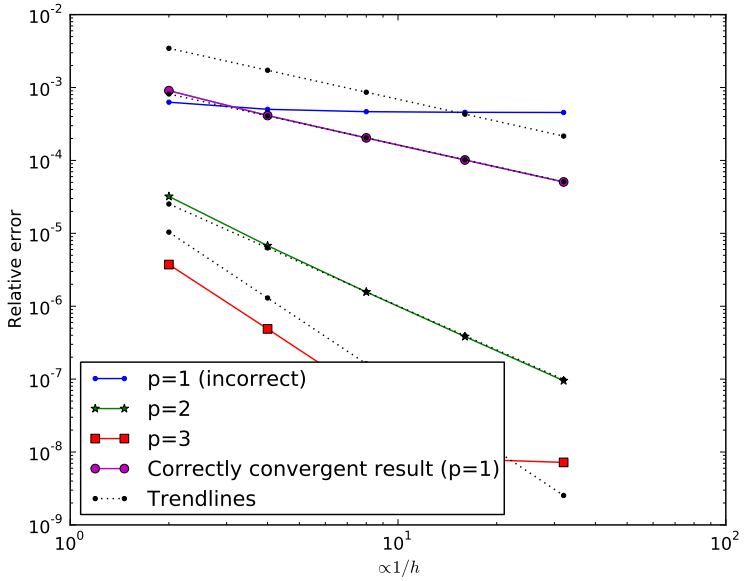



**Figure 4.8:** New incorrect (for p=1) convergence results for Eq. 4.9. Higher order correct convergence results are not shown as they are very close to the incorrect results. The final divergence of the p=3 results is due to numerical conditioning of the system (largely hidden by legend).

## 4.2 Mutation Testing

Mutation Testing is used to further look at the efficacy of testing FEM code with the MMS. The reduced set of mutation operators as discussed in Chapter 3 is used here. The convergence results generated for the 2D problem presented in Chapter 4.1 for all the mutated versions of the code is shown in Figure 4.11. A total of 527 mutants was produced for this specific instance. 106 of these mutants were detected by the MMS, 279 ended in catastrophic failure while 142 equivalent/undetected mutants were not detected.

Figure 4.12 shows the convergence results for the static 3D problem presented in Chapter 4.1. The same 527 mutants were produced for this case. Only 85 instances were successfully detected by the MMS, 138 ended in catastrophic failure while 304 equivalent/undetected mutants were not detected. The large number of equivalent mutants in this case is due to specific code mutated during testing being only applicable to 2D results.

It is important to realise that a large number of equivalent mutants will be produced when using mutation testing. The important result here is that the MMS does indeed detect errors as expected.

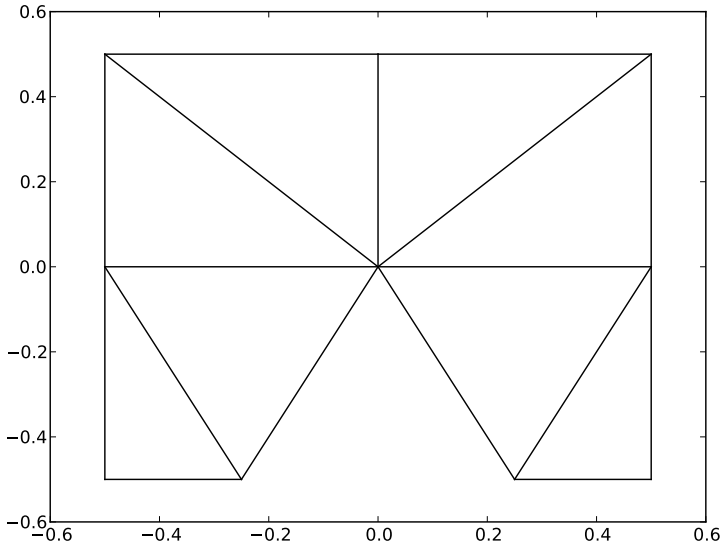



**Figure 4.9:** Geometry used to simulate the effect of re-entrant corners.

## 4.3 Cumulative effect of various MS

Mutation testing has also been used to look at the cumulative effect of testing a piece of code with various manufactured solutions. This will give an indication of the efficacy of a particular method and if multiple methods should be used to obtain better code coverage. Figure 4.13 shows the result from this analysis for the static and dynamic MS given in Eqs. 4.5 and 4.6. It is clearly noticeable that there is a set of mutants that are not detected by both MS. This is largely due to the static and dynamic nature of the MS and also due to the fact that the dynamic MS have classic PEC boundary conditions at $x = 0$ and $y = 0$. Note the effect that frequency has on the percentage errors detected and also note that the amount of errors becomes more (but unstable) towards higher frequencies. The author speculates that this is either due to the change in number of elements used to represent the solution or due to the change in boundary values present (results presented in [53] show less stable convergence rates for higher frequencies even when the solution is highly refined).

## 4.4 Conclusion

In this chapter, the MMS has been investigated in its natural setting, namely the FEM. Convergence rates in the literature for the FEM have been discussed. Numerical examples and a demonstration of concept have been given. The effective ability of the MMS to detect errors in a widely used open source

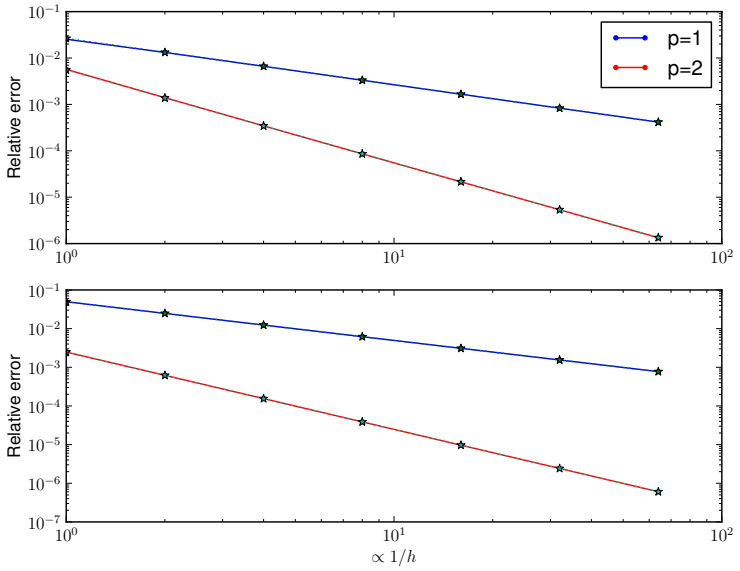



**Figure 4.10:** Convergence results with a geometry with a re-entrant corner using 1st and 2nd order elements. (a) $L^2$-norm of $\boldsymbol{E}$ and (b) the $L^2$ norm of $\nabla \times \boldsymbol{E}$.

package have been demonstrated. The effectiveness of the MMS as a regression test have also been demonstrated. A possible bug/conceptual error has been identified in a widely used open-source package, namely Dolfin. Finally mutation testing have been used to investigate the efficacy of the MMS to detect errors. The MMS in a differential setting is not new but this is the first time that the author have seen its use with respect to the E-field FEM. This is also the first instance where mutation testing have been used on the MMS to prove its efficacy from a pure software implementation perspective.

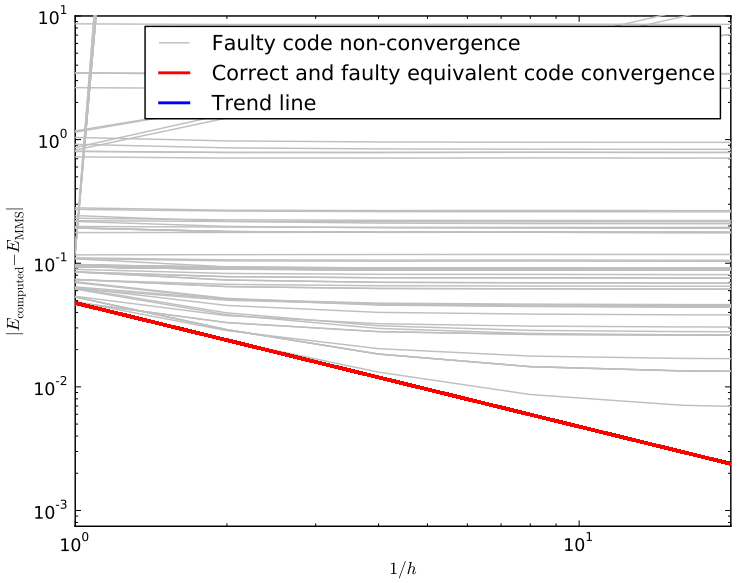



**Figure 4.11:** Mutation results computed using the MS of Eq. 4.6 discussed in Section 4.1 for a 2D geometry. The expected trend line is covered by the correct/equivalent convergence results line.

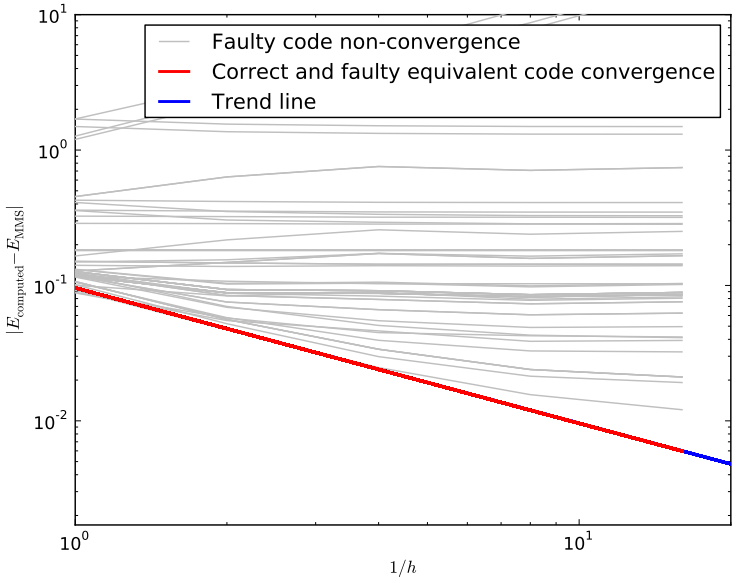



**Figure 4.12:** Mutation results computed using the MS of Eq. 4.8 discussed in Chapter 4.1. The expected trend line is covered by the correct/equivalent convergence results line.

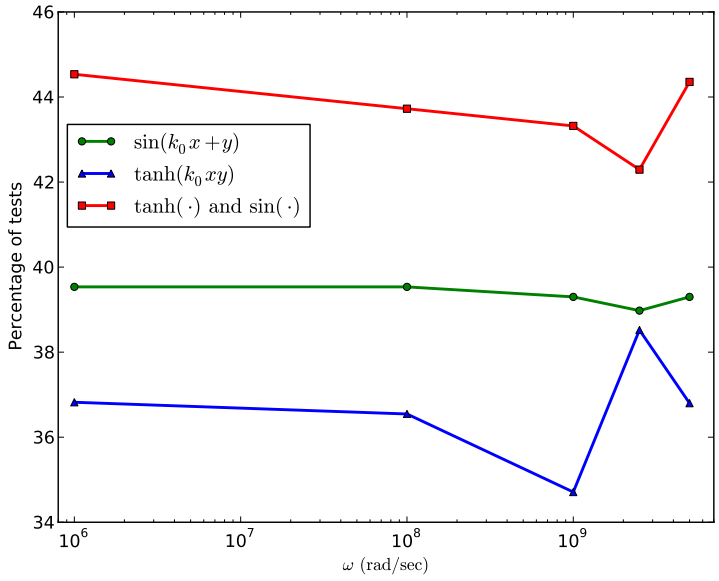



**Figure 4.13:** The effect of frequency and MS on the number of mutations detected.

# Chapter 5

# The Method of Manufactured Solutions : The Method of Moments

In this chapter the MMS process will be discussed for the Method of Moments (MoM). First, the MoM will be briefly introduced by stating the BVP problem that it solves and then discussing the specific integral equation (the EFIE) that is studied here. The issue of selecting a MS is investigated in this chapter. When discussing the MoM in the rest of this chapter, it should be assumed that the EFIE is being solved and not other formulations such as the MFIE. The process of the MMS and the driving terms computation for other formulations such as the MFIE is essentially the same as for the EFIE, but restricted to geometry requirements for the particular methods (such as closed surfaces for the MFIE.)

The operator equation being solved using the MoM uses an integral operator on a specific domain, and it is therefore not a simple exercise to apply the operator as is the case for the FEM. Integrals are not always analytically computable and therefore numerical integration is used. Numerical integration does however introduce its own set of accuracy issues which will be discussed in more detail in this chapter.

The incident E-field obtained after applying the EFIE operator to the MS is used to drive the MoM process, and the computed current is compared to the selected MS. The convergence rate for the MoM is subsequently used to verify the CUT. Theoretical convergence rates are once again required to compare with the computed rate and they will therefore be discussed in this chapter. This chapter will conclude with a mutation test investigation of the method.

## 5.1 The Method of Moments

The MoM is briefly discussed here. More information regarding the method is discussed in Appendix F and in well known reference texts [14, 25, 32, 73].

The MoM as discussed in this work solves the scattering BVP

$$\begin{aligned} \nabla \times \nabla \times \boldsymbol{E}_{\text{sc}} - k_0^2 \boldsymbol{E}_{\text{sc}} &= 0 \qquad \text{in } \Omega, \\ \hat{\boldsymbol{n}} \times \boldsymbol{E}_{\text{sc}} &= -\hat{\boldsymbol{n}} \times \boldsymbol{E}_{\text{inc}} \quad \text{on } \Gamma \end{aligned} \tag{5.1}$$

together with the Silver-Müller radiation condition (B.30) using a boundary integral approach. $\Omega$ is the open solution domain [1] and $\Gamma$ is a Lipschitz PEC scatterer in $\Omega$. The scatterer may be a screen (thin plate like) structure or a polyhedral object [9, 13]

The MoM however is a boundary element method that solves the well-known EFIE,

$$\hat{\boldsymbol{n}} \times \boldsymbol{E}_{\text{inc}} = \hat{\boldsymbol{n}} \times \int_{\Gamma} \left[ jk\eta \boldsymbol{J}_s(\boldsymbol{r'}) G(\boldsymbol{r}, \boldsymbol{r'}) + \frac{\eta}{jk} \left\{ \nabla' \cdot \boldsymbol{J}_s(\boldsymbol{r'}) \nabla' G(\boldsymbol{r}, \boldsymbol{r'}) \right\} \right] dS', \tag{5.2}$$

derived and discussed in Appendix F.2.

The following discussion of the solution spaces for the BVP (Eq. 5.1) and boundary values thereof as solved by the EFIE (Eq. 5.2) is discussed next, and is based on [13, 36]. The solution to the E-field, solved by the BVP (Eq. 5.1) is in the Hilbert space $H_{\text{div,curl},\Omega} = H_{\text{div}}(\Omega) \cap H_{\text{curl}}(\Omega)$ [2]. The fact that the result of the *curl* and *div* operators acting on a function are square integrable does not imply that the components of the function are square integrable and hence the boundary values of the E- and H-field are at least only in $H_{\text{div}}^{-1/2}(\Gamma)$ [36]. It has been shown however that the boundary values for $H_{\text{div,curl}}(\Omega)$ are in $H_{\text{div,curl}}^{-1/2}(\Gamma) = H_{\text{div}}^{-1/2}(\Gamma) \cap H_{\text{curl}}^{-1/2}(\Gamma)$ [36]. The solution to the EFIE is related to the boundary values of the E- and H-fields by the relation

$$\boldsymbol{J} = \hat{\boldsymbol{n}} \times \boldsymbol{H}|_{\Gamma} = \frac{j}{\omega\mu} \hat{\boldsymbol{n}} \times \nabla \times \boldsymbol{E}|_{\Gamma},$$

which leads to the well known fact that $\boldsymbol{J} \in H_{\text{div}}^{-1/2}(\Gamma)$.

As noted by various practitioners, the solution $\boldsymbol{J}$ to the EFIE is defined in $H_{\text{div}}^{s}(\Gamma)$, with $s \geq -1/2$. More on this can be found in Appendix C and in [13, 36].

It should be stressed that the current $\boldsymbol{J}$ that solves the EFIE is the vector sum of the current on the two sides of the thin metal sheet represented by $\Gamma$.

All solutions to the EFIE (Eq. 5.2) that are in $H_{\text{div}}^{s}(\Gamma)$ do not however induce E-fields that are also solutions to the BVP (Eq. 5.1). Only solutions

[1] as used by Buffa

[2] These spaces and subsequently used spaces are defined and discussed in Appendix C

that are in the space

$$X^s = \{u \in H^s_{\text{div}}(\Gamma) | \langle u, \nabla v \rangle + \langle \nabla \cdot u, v \rangle = 0, \forall v \text{ on } \Gamma\}$$

[13, Definition 1, Theorem 3.2] will actually induce valid solutions to the BVP (Eq. 5.1).

Solutions calculated by the EFIE (Eq. 5.2) that induce solutions to the BVP (Eq. 5.1) are forced by physically suitable boundary conditions, that is $\hat{\boldsymbol{n}} \times \boldsymbol{E}_{\text{inc}}$ is such that correct results are produced.

### Resonant frequencies

It is a well known fact [17, 64] that physically irrelevant interior resonances are present in the solutions to the EFIE at certain frequencies. These interior resonances should not contribute to radiation of the external field but do indeed radiate due to numerical accuracy issues. There are methods such as the combination of the EFIE and MFIE into the CFIE that do not exhibit this behaviour. In this work, it is assumed that either of the following are valid: screens are used that do not exhibit interior resonances, the frequency chosen is such that interior resonances do not occur, or that appropriate steps have been taken to mitigate the effect of interior resonance [64].

## 5.2 MS selection for the application of the MMS to MoM

When selecting an MS for the MoM, there are a few important restrictions and practicalities that should be kept in mind. The MS selected does not have to induce physically realisable solutions that are solutions to the MoM BVP (Eq. 5.1), as discussed in Section 5.1. Practical issues that arise when selecting a MS include:

1. Solutions to the EFIE are in the space $H^s_{\text{div}}(\Gamma)$ ($s \geq -1/2$), and as such, an MS should be selected in this space.

2. The weak requirement that $\nabla \cdot \boldsymbol{J} \in H^s(\Gamma)$ for $s \geq -1/2$ implies that $\nabla \cdot \boldsymbol{J}$ does not have to be square integrable. On the other hand, for practical numerical integration, it is preferable that $\nabla \cdot \boldsymbol{J}$ not have any singularities present.

3. A selected solution should not have any current that flows normally over the edge of screens. This is due to the fact that the solution current to the EFIE is the vector sum of the current that flows on both sides of thin plate like structures. The current flowing normally over an edge is equal in magnitude but opposite in sign and is therefore zero. There are three ways to ensure that this requirement is fulfilled:

a) Select an MS such that the component normal to the edge of the scatterer is zero,

b) Use a weighting function that is determined by how close the evaluation point is to the edge of the structure, and

c) Project the selected MS onto the space used to solve the EFIE, that is, use a fine mesh with the same type of basis functions used for the final solution and project the selected MS onto the mesh. The basis functions (such as RWG [65]) used on the selected mesh are designed not to admit any solutions that have normal components to the boundary of the selected mesh.

## 5.3 Numerical Integration

In contrast to the MMS for the differential FEM discussed in Chapter 4, the MMS for the MoM involves an integral operator. The calculation of the driving terms ($\hat{\boldsymbol{n}} \times \boldsymbol{E}_{\text{inc}}$) using the integral operator is unfortunately not always analytically possible, and therefore numerical integration is necessary. Accurate numerical integration of the EFIE is discussed in this section.

The following should be noted about the EFIE (Eq. 5.2) to accurately determine the driving terms. The term of the EFIE that corresponds to the calculation of the vector potential Eq. B.18.

$$\int_{\Gamma} jk\eta \boldsymbol{J}_s(\boldsymbol{r'}) \frac{e^{-jkR}}{4\pi R} dS',$$

contains the Green's function singularity $\frac{1}{R}$ with $R = |\boldsymbol{r} - \boldsymbol{r'}|$. This will be referred to as the *singularity*. The term of the EFIE that corresponds to the calculation of the scalar potential Eq. B.19

$$\text{PV} \int_{\Gamma} \frac{\eta}{jk} \left\{ \nabla' \cdot \boldsymbol{J}_s(\boldsymbol{r'}) \nabla' G(\boldsymbol{r}, \boldsymbol{r'}) \right\} dS', \tag{5.3}$$

contains the gradient of the Green's function $\nabla G(\boldsymbol{r}, \boldsymbol{r'}) = -jke^{-jRk}/R - e^{-jRk}/R^2$. This clearly has a $\frac{1}{R^2}$ singularity and will be referred to here as the *strong-singularity*.

The presence of these two singularities complicates accurate numerical integration. These complications are ameliorated in the standard MoM implementation due to the following. The EFIE can be numerically solved using a Galerkin testing procedure (testing and weighting functions are the same) and as such the gradient operator acting on the Green's function is moved onto the testing function. Hence integration in the standard MoM procedure does not involve the *strong-singularity* but only a *singularity* [80, p. 196]. In this work however Equation 5.2 is computed directly and hence direct integration techniques are required. In the following integration techniques will be discussed for this purpose.

Triangles that are not close to the source point causing the singular behaviour can be integrated by various methods. One of the well known methods for integration on a triangle is the method by Dunavant [23]. This method calculates weights and integration points in terms of simplex coordinates using a Moment Method. Tables are supplied for integration points and weights. Drawbacks to this method are that there are negative weights, for certain orders, involved which have implications for accuracy. Also in instances where a large number of points are required, some of the sampling points might be outside the triangle as is apparent when scrutinising the tables in [23]. This method is often used in MoM code but was decided against in this instance. Here a simple Gauss product rule, with appropriate Duffy transformation [22], was implemented which does not suffer from negative weights. All integration points are also in the interior of the triangle. This method is less efficient and produces non-symmetrical integration points but ease of implementation and the ability to easily vary accuracy was considered advantageous for integration accuracy investigations.

## Methods for numerical integration of singularities and strong-singularities

Accurate integration of *singularities*, *strong-singularities* and their near equivalent singularities is important for the correct determination of the incident E-field. Efficient evaluation of these singularities is however not the main aim of the current investigation, but methods that are not tailored to the mentioned singularities often either fail to obtain accurate results [10] or take a very long time; often this is orders of magnitude longer than for the non-singular case. The efficient and accurate numerical integration of these types of singularities is presently an active research topic in the CEM community.

The methods available in CEM literature and used in this work are discussed next. The first method widely used is the method by Duffy [22]. This method weakens the singularity, but as mentioned by Botha [10], does not remove the singularity completely and is thus limited in accuracy. Khayat and Wilton [44] introduced a singularity cancellation method in 2005 which was superseded in 2008 by the more efficient method [45] used here. Graglia and Lombardi [29] introduced a method that is able to integrate to machine precision. This method, however, relies on an external FORTRAN routine, GQRAT by Gautschi [27], that integrates rational functions. Interaction between the implementation environment and the external routine incurred large administrative computational overhead. The current author was also not able to achieve the advertised machine accuracy for the integral studied. Tong and Chew [79] introduced a singularity subtraction method in 2010 that used Stokes theorem to integrate the subtracted highly singular kernels. The method by Tong and Chew was not used because the singular kernel extracted could potentially be

dependent on the MS selected, and independence from this selection is deemed desirable.

The method used in this work to evaluate singularities and near-singularities is the method introduced by Khayat *et al.* [45]. This method considers a singularity on or close above the vertex of a triangle. The investigation of a singularity only above the vertex of a triangle does not limit the method's applicability for general singularities on or close to arbitrary triangles. This is because any singularity above a triangle can be decomposed into three sub-triangles with the singularity on or above the point used to subdivide the triangle. This integration method becomes more inefficient (more points are required for accuracy) when the sub-triangle being integrated becomes more obtuse as evident from results presented in [45] and results that will be presented later in this work. This method is presented in [45] for flat rectilinear triangles but as noted by amongst others [12] can be implemented for curvilinear elements by a simple tangential surface.

The strong-singularity has to be evaluated using a different method. Literature in the CEM field is more sparse when it comes to strong-singularities. The methods that are encountered include the above mentioned method by Tong and Chew [79]. Fink, Wilton and Khayat [26] introduced a method to evaluate *near* strong-singularities. This method uses a circular section that is beneath the singularity present and introduces the following two issues. First, the method is not valid for source points that are above edges or vertexes of triangles, which might be the case when using a finer mesh to calculate the driving terms than the mesh used for the MoM CUT. Secondly, the integration in the circular disc requires an integration rule that involves exponential functions that have a large dynamic range, which limits the accuracy of the method. New methods have recently (2012) been introduced by Botha [10], based on a generalised Duffy transformation.

The method used here is the one by Weile and Wang [81], which is based on the method by Guiggiani and Gigante [30]. This method is specifically suited to the principal value integral that contains the gradient of the Green's function. This method is also valid for curvilinear elements without resorting to equivalent tangential triangles as is the case for the above methods.

## EFIE integration verification

The implementation of the EFIE integration is next verified. The EFIE verification consists of two steps. First, the integration method is compared to a highly accurate reference value calculated on a single triangle. Secondly, the practical ability to directly apply the EFIE to a reference current is investigated.

### Integration accuracy for a single triangle

To verify the accurate integration of the EFIE on a single triangle, the two singular parts of the EFIE are investigated separately. First, a reference value for integration is computed by applying a low order Gaussian quadrature scheme (n=10) to a highly refined triangulation of a reference triangle. The mesh used to generate the reference values is not uniform but is more refined at the source point. This grading ensures that an accurate answer can be computed using less computational resources. Two reference meshes were used, both with an outer mesh density of 0.1 and a inner mesh density with either $10^{-6}$ or $10^{-7}$. The finest limits were chosen because the mesh generating tool, `gmsh` [28], only generated meshes with accuracy up to this limit.

**The $\frac{1}{R}$ singularity** A comparison between the second method by Khayat *et al.* (2008) [45], the reference result and the first method by Khayat *et al.* (2005) [44] is shown in the top half of Figure 5.1. The comparison in Figure 5.1 is done for points as the perpendicular distance $z$ approaches 0, that is $z \to 0$. The following can be noted: The meshing density does have an influence on the accuracy of the reference value calculated as can be seen by the sudden increase in error at $z = 10^{-6}$ and $z = 10^{-7}$. The value computed using the first method by Khayat *et al.* (2005) does indeed also agree well with the value calculated using Khayat *et al.* (2008). It can be seen that the reference value approaches the value calculated by Khayat *et al.* [45] as $n$ is increased. A comparison between the value calculated using Khayat *et al.* (2008) for $z = 0$ and as $z \to 0$ for the same method also confirms that the expected asymptotic behaviour is achieved.

**The $\frac{1}{R^2}$ singularity** The Weile and Wang integration method [81] (WWI) is compared here to the method by Fink *et al.* [26] (FWKI). FWKI is not valid for a source point on the surface of the integration triangle [26]. The principal value integral Eq. 5.3 has a normal component that is discontinuous as the source point approaches the surface of the triangle [26, 43, 81]. Fink *et al.* [26] note that the discontinuous behaviour of the integral is normally dealt with when testing the function. WWI deals with this normal component explicitly.

A comparison between FWKI, reference results computed as above and WWI for ($z = 0$) is shown in the top half of Figure 5.2. The effect of meshing density and number of integration points ($n$) per subtriangle is again visible for the reference results. The results obtained by WWI for $z = 0$ is compared to the results obtained using FWKI as $z \to 0$. It can be seen that FWKI approaches the correct value until $z = 10^{-11}$ after which results starts to diverge. The comparison of FWKI to reference results for $z > 0$ also starts to worsen as $z \to 0$ with accuracy that is close to that of the comparison to Weile and Wang for $z = 10^{-11}$. The results presented persists even if the number of integration points for FWKI are increased significantly.

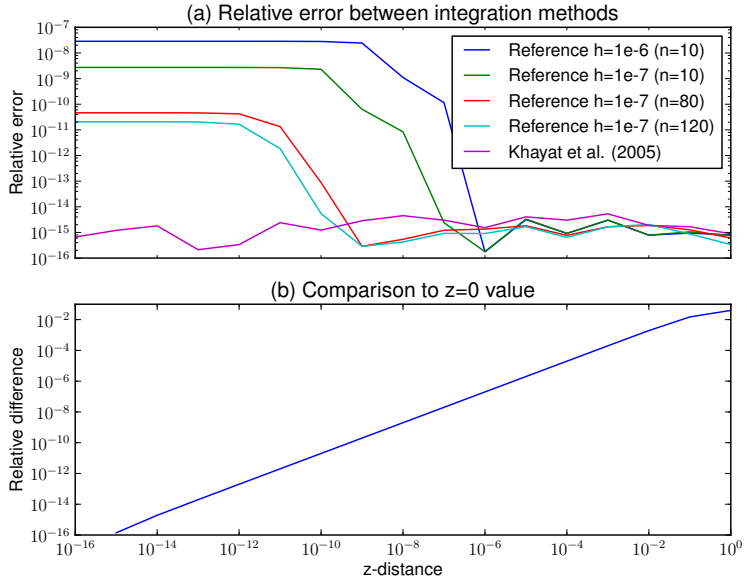



**Figure 5.1:** (a) Comparison of the integration method by Khayat *et al.* [45] to calculated reference values and the 2005 method by Khayat *et al.* [44]. (b) Comparison between the calculated value for $z = 0$ and $z \geq 0$ using the method by Khayat *et al.* [45].

In the bottom half of Figure 5.2 the normal component calculated by WWI is compared to the normal component computed using FWKI. This component is not used in the code presented but it is used as a proxy for calculation accuracy comparison. This confirms the numerical inaccuracy of the method of FWKI for $z \to 0$. The values calculated for the normal component by WWI are trusted because its calculation is a simple analytical expression $f(\boldsymbol{r})/2\hat{\boldsymbol{n}}$, where $f$ is $\nabla \cdot \boldsymbol{J}$ from the selected MS and $\hat{\boldsymbol{n}}$ is the normal at the singular source point $\boldsymbol{r}$.

It should be evident that the method by Weile and Wang is preferred.

### Practical application of the EFIE

The practical ability to apply the EFIE operator (Eq. 5.2) is investigated here by using the current induced on a PEC sphere when it is illuminated by an incident plane wave [6]. The Mie-series solution of the field scattered from a PEC sphere gives the field close to and on the surface of a sphere. This requires the accurate implementation of the Mie-series and a proper curvilinear representation of the sphere surface. The surface of a sphere can be represented very accurately using a mesh consisting of triangles and applying the curvilinear transform presented in [34].

The H-field and divergence of the H-field obtained through the Mie-series

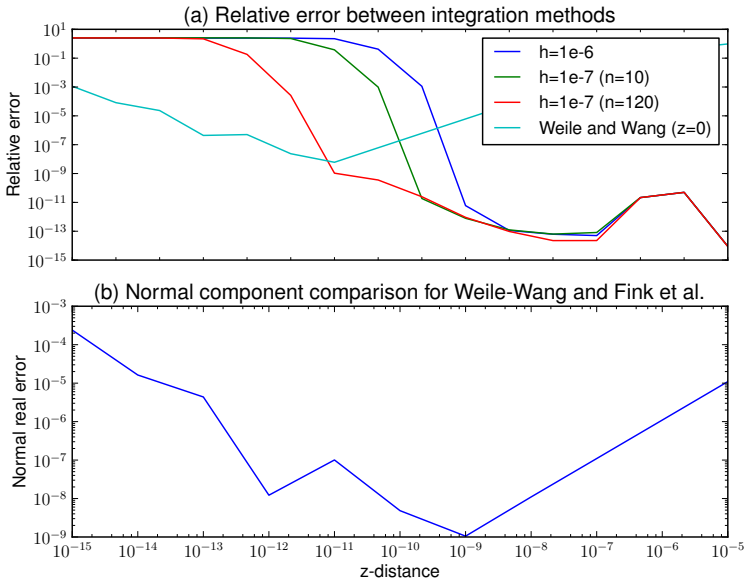



**Figure 5.2:** (a) Integration using the method by Fink *et al.* [26] is compared to reference results and the result obtained by the method of Weile and Wang [81] for $z = 0$. (b) The behaviour of the normal component of the method by Fink *et al.* [26] when compared to the trusted value computed using Weile and Wang [81].

can be used to calculate the current and the divergence of the current on the surface of the sphere using the boundary condition $\boldsymbol{J}_s = \hat{\boldsymbol{n}} \times \boldsymbol{H}$. The computed currents and divergence of the current can be used to calculate the tangential component of the incident E-field on the surface of the PEC sphere using the EFIE (Eq. 5.2). The computed tangential incident E-field is compared to the incident E-field used to compute the original current on the PEC sphere.

The three points on a triangle used for verification of the integration method are shown in Fig. 5.3. The central point, P1, is expected to yield accurate results. The two points P2 and P3, located close to a vertex and an edge respectively, are known to give less accurate results [45]. It is possible to integrate the EFIE to an arbitrary precision as shown in Fig. 5.4. It should be noted that for points 2 and 3 it appears as if the accuracy is limited. This is due to the fact that the number of points used in the singular methods mentioned above was chosen such that a certain level of accuracy is achieved ($r = t = 60$). Also, near-singular integration was not used on adjacent triangles which decreases accuracy/efficiency of the overall integration.

It is possible to integrate the EFIE to high precision as shown in Figure 5.4.

It should be noted that for points 2 and 3 it appears as if the accuracy is limited. This is again due to the fact that the number of points used in the singular methods mentioned above was chosen such that a certain level

of accuracy is achieved. It should also be noted that singular integration was not used on adjacent triangles which also decreases accuracy/efficiency of the integration method used.

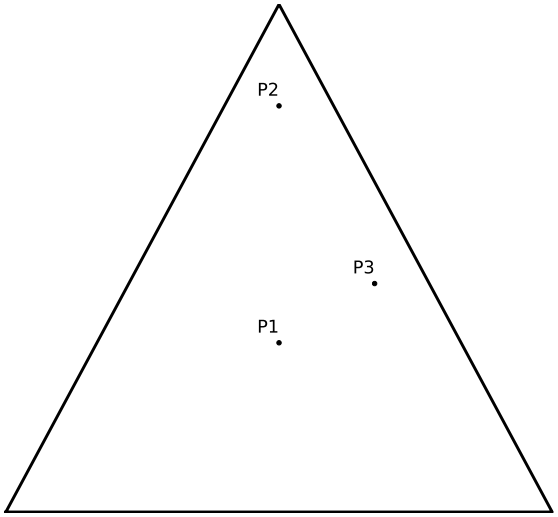



**Figure 5.3:** Three points used for verifying the accuracy of EFIE integration. P1 is not expected to give accuracy issues, but the attainable accuracy at P2 and P3 are known to be limited [45].

## Effect of inaccurate integration on convergence results

Next the effect of integration accuracy on the ability to detect coding error will be discussed. Scattering from a PEC sphere [6] is used once again to look at the effect of integration accuracy. In this instance, the incident E-field that illuminates the PEC sphere is rounded off to a specified level. This rounded off incident E-field is then used to drive the assumed-to-be-correct CUT. The resultant computed current and divergence of the current are compared to the expected current and divergence of the current computed using the Mie-series expansion. The results from a deliberately inaccurate integration are shown in Fig. 5.5. It can be seen that the effect of integration does have an influence on the convergence rate as expected. It is important to integrate to a sufficient level of accuracy. From the author's experience, it is necessary to use integration that is at least 3 significant digits more accurate than the difference between the exact result and the desired computed result.

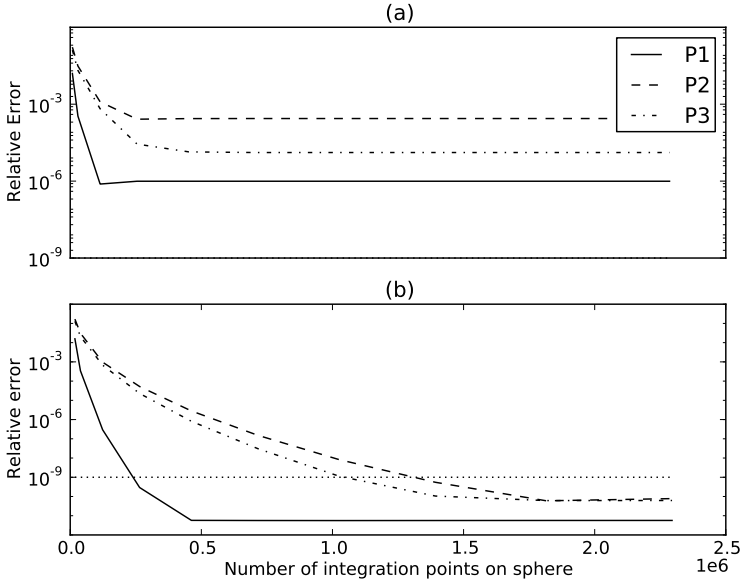



**Figure 5.4:** The accuracy achieved at the three specified points shown in Fig. 5.3. (a) shows the effect an inaccurate singularity handling scheme with only 1200 integration points in the singular triangle and (b) shows the improved results with 10800 integration points in the singular triangle.

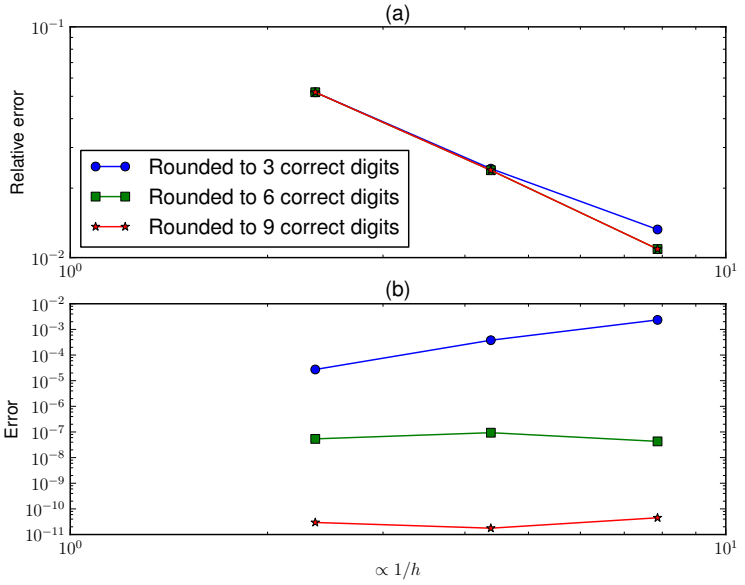



**Figure 5.5:** The detrimental effect of low integration accuracy is visible here. The incident E-field impinging on a PEC sphere is rounded to simulate integration accuracy. (a) shows the observed convergence rates. (b) shows the difference between the observed convergence rate with imperfect integration and perfect integration.

## 5.4 Expected convergence rates

The final step in the MMS is to compare the computed convergence rate to the expected theoretical convergence rate. The expected rate of convergence is known for the RWG [65] (lowest order Raviart-Thomas (RT0) [66]) basis functions and is discussed in the rest of this section. The rate of convergence is governed by the Sobolev regularity ($s$) of the solution as is the case for the FEM and as will be seen in the rest of this section. However it is assumed that the polynomial order of the solution space is high enough not to be a limiting factor. Thus, if the regularity of the solution is higher than the polynomial order, the convergence rate will be governed by the polynomial order of the basis functions.

The issue of convergence rates has been investigated by various practitioners and here a short summary of their work will be given. In the discussion that follows $\boldsymbol{u}$ will be a quantity that is proportional to the current $\boldsymbol{J}$ with $\boldsymbol{u}_h$ the discrete representation of $\boldsymbol{u}$. The quantity $\lambda$ is proportional to the charge and $\lambda_h$ denotes its discrete representation. The quantity $\boldsymbol{c}$ will be proportional to the tangential projection of the incident E-field on the scatterer $\Gamma$. The size of the largest discrete element of the mesh is denoted by $h$. The polynomial order of the basis function representation used is $p$, and for the RWG (RT0) basis functions $p = 1$. All of the discussed convergence rates are for quasi-uniform meshes.

The first convergence rate result was proved by Bendali [7, 8] for *smooth closed* scatterers. The rate proved by Bendali is

$$\begin{aligned}\|\boldsymbol{u} - \boldsymbol{u}_h\|_{\boldsymbol{H}_T^{-1/2}(\Gamma)} + \|\lambda - \lambda_h\|_{H^{-1/2}(\Gamma)} \leq C[&(h^{p-\sigma} + h^l)\|\boldsymbol{u}\|_{p,\Gamma} \\ &+ (h^{p+1/2} + h^l)\|\lambda\|_{p,\Gamma} \\ &+ h^{l+1/2}\|\boldsymbol{c}\|_{C^1(\Gamma_\delta)}],\end{aligned}$$

with $l$ representing the polynomial order for the map between a reference triangulation and the actual triangulation; this map is important for curvilinear elements but is not used here. For RWG basis functions $l = 1$ [8]. The real number $\sigma$ with $0 < \sigma \leq 1/2$ indicates the regularity of $\lambda$ such that $\lambda \in H^{-1/2+\sigma}(\Gamma)$. The constant $C$ is dependent on $\sigma$. The most notable limitation of this error norm is that it is only valid for *closed smooth* scatterers and that it is not optimal [15].

Hiptmair and Schwab [35] proved in 2002 a convergence rate for closed polyhedra. This rate can be stated as

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} \leq C h^{\min\{3/2-\epsilon, s+1/2-\epsilon, 1+s^*, s+s^*\}} \|\boldsymbol{u}\|_{H_{\text{div}}^s(\Gamma)},$$

with $s$ the regularity of the correct solution, $s^*$ a geometry parameter defined in [35], and $\epsilon$ an arbitrary positive constant. The constant $C$ depends only on $\Gamma$, the wave number and $\epsilon$.

For closed smooth surfaces Christiansen [15] proved the following optimal convergence rate,

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-s}(\Gamma)} \leq Ch^{s+s'}\|\boldsymbol{u}\|_{H_{\text{div}}^{s'}(\Gamma)}. \tag{5.4}$$

It is evident that the rate of convergence in this case is dependent not only on the solution regularity, but also on the norm used to measure the error. It is important to note that the proof given by Christiansen requires that the polynomial order of the basis functions used is of sufficient high degree. For smooth closed surfaces, a highly regular solution $s' \geq 1$ and using RWG basis functions this implies

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} \leq Ch^{3/2}\|\boldsymbol{u}\|_{H_{\text{div}}^{s'}(\Gamma)}.$$

Buffa and Christiansen [13] proved for Lipschitz screens

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} \leq Ch^{1/2+s}\|\boldsymbol{u}\|_{H_{\text{div}}^{s}(\Gamma)}, \tag{5.5}$$

with $s$ once again the Sobolev regularity of the solution but only for $s \in (-1/2, 0)$. This convergence rate is also valid for polyhedral structures as noted by Bespalov and Heuer [9]. In contrast to the convergence rates given for smooth closed surfaces (Eq. 5.4), the convergence rates for Lipschitz screens (Eq. 5.5) are given only in the $H_{\text{div}}^{-1/2}(\Gamma)$ norm. The requirement for the specified norm are primarily due to the nature of the singularities in the solution for scattering from screens as shown by [16]. A description of the appropriate spaces and norms can be found in Appendix C and in [36, 52]. Besplov and Heuer stated that the estimate in Eq. 5.5 is optimal.

The convergence rates proved by Buffa and Christiansen are however valid for solutions that have a higher order of regularity as explained in the following. The final proof for convergence [13, Proposition 4.10] is based on the quasi-optimal convergence rate [13, Theorem 4.5] of the MoM and the convergence behaviour of the projection operator [13, Lemma 4.9]. The quasi-optimal convergence rate however is stated in terms of $\|\cdot\|_{H_{\text{div}}^{s}(\Gamma)}$ which is valid for solutions of higher regularity because more regular spaces are subsets of less regular spaces [52]. The convergence behaviour of the projection operator [13, Lemma 4.9] is already stated for higher regular functions. It follows that if the regularity of singularities is lifted, [13, Theorem 4.10] is valid for more regular functions.

Bespalov and Heuer [9] proved a convergence rate that is valid for Lipschitz screens and closed polyhedra using basis functions other than the RWG (RT0) basis functions. This convergence rate is listed here to highlight the effect that the polynomial order $p$ of the basis functions has on the convergence rate:

$$\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} \leq Ch^{1/2+\min(s,p)}p^{-(s+1/2)}\|\boldsymbol{u}\|_{H_{\text{div}}^{s}(\Gamma)},$$

with the symbols having the same definitions as given above. For $p = 1$ this is the same convergence rate proven by Buffa and Christiansen.

The following comment by Bespalov and Heuer, given in the introduction of [9] should be noted. The expected convergence rate of the MoM is dependent on the Sobolev regularity of the solution minus the Sobolev order of the energy norm -1/2. This is also in essence expressed in the convergence rates given by Christiansen (Eq. 5.4) where the rate is dependent on the norm used.

Calculation of the $H_{\text{div}}^{-1/2}(\Gamma)$-norm is not entirely obvious to CEM practitioners and is discussed in Appendix C.1.

## 5.5 Case studies

MS examples will be discussed next. First simple plate-like geometries will be investigated to asses the basic applicability of the method. After this more complex geometries will be investigated. All examples are computed with a basic MoM code named GMoM developed by Ludick while working on his M.Sc degree [51]. This code uses RWB(RT0) basis functions and as such the polynomial order $p$ is equal to 1. The norm used throughout this work is the graph norm for $H_{\text{div}}^{-1/2}(\Gamma)$ given in Eq. C.2.

### Manufactured solutions on rectangular plates

The Manufactured Solutions discussed in this section are all based on a simple rectangular geometry. A PEC plate is used that is in the xy-plane with vertexes at (-1,-1,0), (1,-1,0), (1,1,0) and (-1,1,0) as shown in Figure 5.6. Also shown on the plate is the base mesh used. Subsequent refinements are made by halving the triangle edges.

Various MS on the rectangular plate will be used to show different aspects of the MMS and basic issues involved.

#### A simple proof of concept

This is a simple proof of concept using a very simple manufactured solution. The MS chosen is

$$\boldsymbol{J}_s = \cos(u_s x)\hat{\boldsymbol{x}} \quad \text{with } u_s = \frac{\pi}{2}, \tag{5.6}$$

The value of $u_s$ is chosen such that the MS has zero normal components at the edges of the geometry. A graphical representation of the MS can be seen in Figure 5.7.

It is easy to see that the Sobolev regularity of the solution is $s > 1$ because all orders of derivatives of $\cos(x)$ are square integrable. As such, the convergence rate expected is $\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} < Ch^{3/2}$. The computed convergence rate is shown in Figure 5.8 together with the expected trendline.

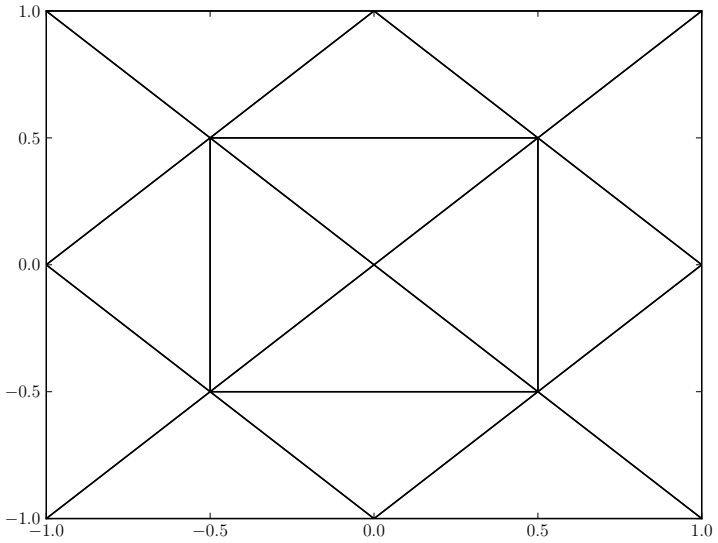



**Figure 5.6:** The simple rectangular plate used to show various aspects of the MMS. The meshing of the plate is the base mesh used and is subsequently refined by splitting triangles by halving their edges.

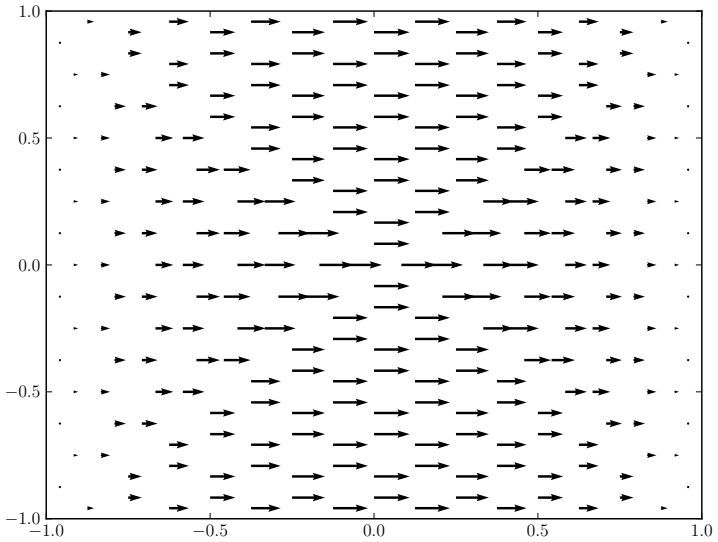



**Figure 5.7:** A graphical representation of the expected MS $\boldsymbol{J}_s = \cos(u_s x)\hat{\boldsymbol{x}}$ with $u_s = \frac{\pi}{2}$ on a simple rectangular plate.

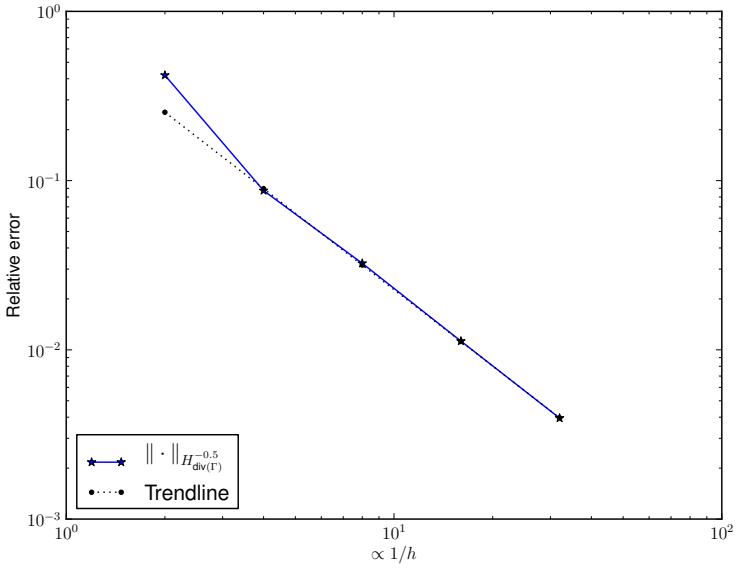



**Figure 5.8:** The computed convergence rate for the MS Eq. 5.6 compared to the expected trend line with slope 3/2. In this figure $k_0 h$ ranges from 0.16 to 0.01.

### An incorrect solution

In this example, an MS is used that is not in the solution space. Consequently it is obvious that the desired convergence rate is not achieved. The MS used is

$$\boldsymbol{J}_s = \cos(u_s y)\hat{\boldsymbol{x}} \quad \text{with } u_s = \frac{\pi}{2}, \tag{5.7}$$

with $u_s$ chosen such that the tangential components are 0 on the edges of the plate. The MS is displayed graphically in Figure 5.9.

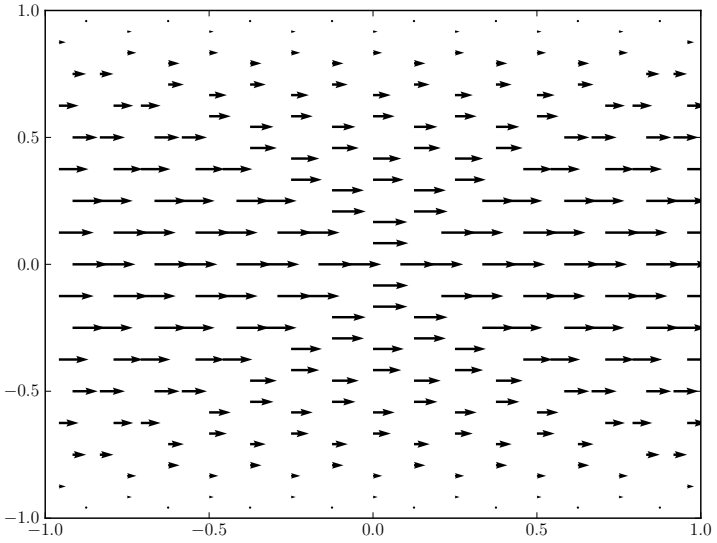



**Figure 5.9:** A graphical representation of the expected MS $\boldsymbol{J}_s = \cos(u_s y)\hat{\boldsymbol{x}}$ with $u_s = \frac{\pi}{2}$ on a rectangular plate.

It can be easily seen that the Sobolev regularity is once again $s > 1$ and as such the wrongly expected convergence rate is $\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} < Ch^{3/2}$. The failure to converge is obvious in Figure 5.10.

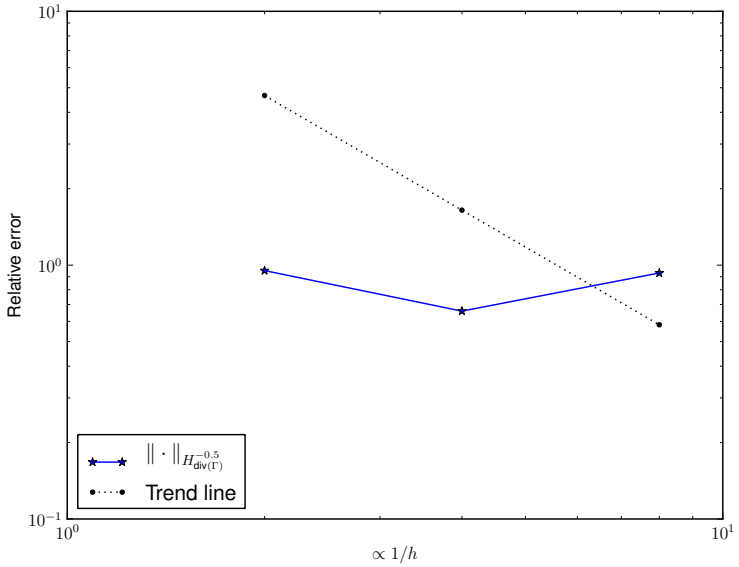



**Figure 5.10:** The computed convergence rate for the MS Eq. 5.7 compared to the expected trend line with slope 3/2. It is obvious that convergence is not achieved due to the normal components of current at the edge of the square.

### The effects of different norms used to compute error

In this example, the effect of an error artificially introduced in the code is investigated. The behaviour of different norms in the presence of an error is also investigated. The chosen MS is

$$\boldsymbol{J}_s = \cos(u_s y)\cos(u_s x)\hat{\boldsymbol{x}} \quad \text{with } u_s = \frac{\pi}{2}. \tag{5.8}$$

Two error scenarios were artificially introduced by perturbing the excitation vector by 1% and 10% respectively. The norms used to calculate the convergence rate is the familiar $L^2$ graph norm for $H^0_{\mathrm{div}}$-space given by Eq C.2 and the $H^{-1/2}_{\mathrm{div}}$ graph norm in this chapter. The following observations can be made about the respective convergence rates: The correct code version converges at the rate as expected from results from Buffa and Christiansen [13] for the $H^{-1/2}_{\mathrm{div}}$-norm ($h^{3/2}$) and as expected for the $H^0_{\mathrm{div}}$-norm ($h^1$) given by Christiansen [15] but stated there for a closed smooth surface. The $H^0_{\mathrm{div}}$ norm used to calculate the error is valid in this screen case because the solution has been chosen such that the Sobolev regularity is larger than 0. It is also intuitive from the comments made in [9] that the expected rate is equal to the Sobolev regularity minus the regularity of the norm. It should be emphasised that it is not suggested here that the $H^0_{\mathrm{div}}$-norm is the correct norm to use or that convergence rates are specified correctly for open surfaces but that the

effect is merely observed. The more useful observation is that the $H_{\text{div}}^{-1/2}$-norm is more sensitive to errors as can be seen especially for the 1% error case.

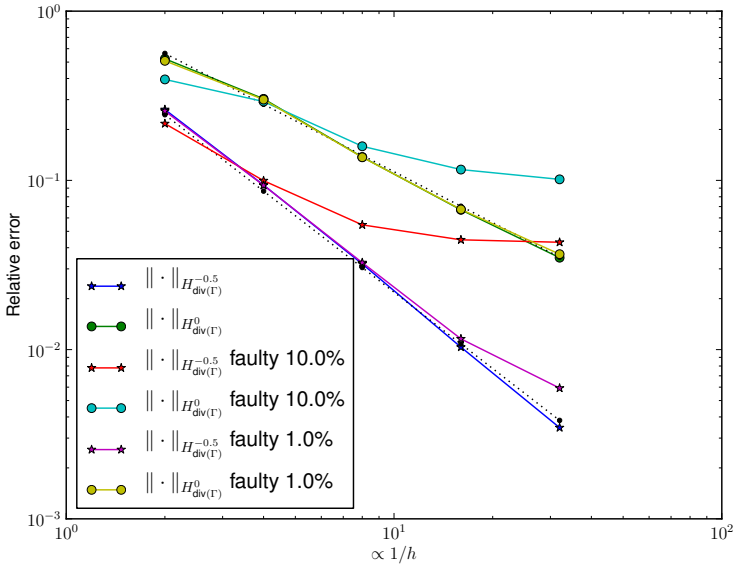



**Figure 5.11:** The effect of an error on the observed convergence rate using the $H_{\text{div}}^{0}$- and $H_{\text{div}}^{-1/2}$-norms. In this figure $k_0 h$ ranges from 0.16 to 0.01.

### The effects of regularity on convergence rate

To demonstrate the effect of regularity on convergence rate the following hat function is used as a MS,

$$\boldsymbol{J} = \begin{cases} 1 + x & \text{if} \quad x < 0 \\ 1 - x & \text{if} \quad x \geq 0. \end{cases} \tag{5.9}$$

A sectional cut on the x-axis of the expected solution is shown in Figure 5.12. The derivative in the x-direction is also shown in Figure 5.12. The step in the derivative implies that there will be a Dirac delta in the first derivative of the divergence of the current.

It is known that the Sobolev regularity of this function is $s = 1/2$. This implies that the expected convergence rate is $\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} < C\, h^1$. The computed convergence rate compares favourably to the expected convergence rate as shown in Figure 5.13.

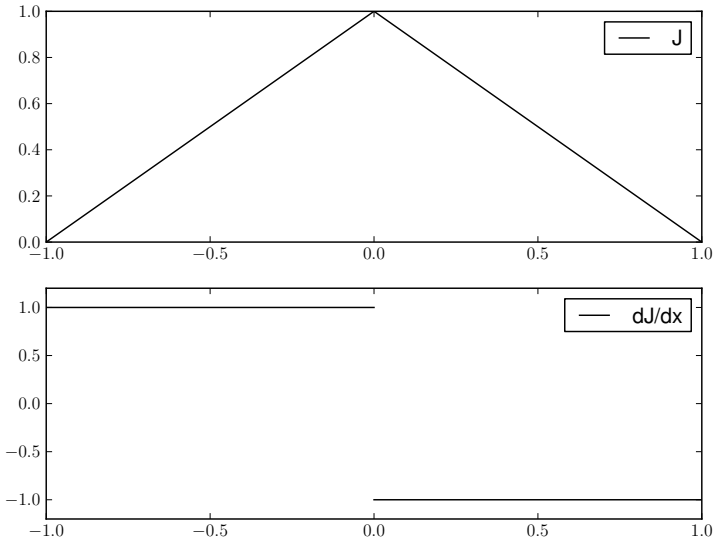



**Figure 5.12:** (a) A sectional cut through the MS Eq. 5.9 in the x-direction showing the hat shape. (b) The derivative of the MS Eq. 5.9 in the x-direction showing the step function that is present in the divergence of the MS.

## Geometry with edges

An MoM code is rarely only used to solve problems on flat objects and it is therefore important to investigate the method when used on structures that exhibit more realistic features. In this section, an example will be shown for a structure that has multiple edges present.

### A ribbon like structure

This structure consists of a long piece of PEC metal that is bent at 90 degree angles. A side profile is shown in Figure 5.14 and a 3D projection with the MS is shown in Figure 5.15. The ribbon has a width of 1m that ranges from $y = -0.5$ to $y = 0.5$. The MS selected is defined as,

$$J = \hat{\boldsymbol{n}} \times \boldsymbol{f_h}, \tag{5.10}$$

with

$$\boldsymbol{f_h} = (\cos(x\pi) + \cos((z+1)\pi))\hat{\boldsymbol{y}}$$

and is also shown in Figure 5.15. The Sobolev regularity of this function is $s > 1$ and therefore the expected convergence rate is $\|\boldsymbol{u} - \boldsymbol{u}_h\|_{H_{\text{div}}^{-1/2}(\Gamma)} < C\, h^{3/2}$. The obtained convergence can be seen in Figure 5.16.

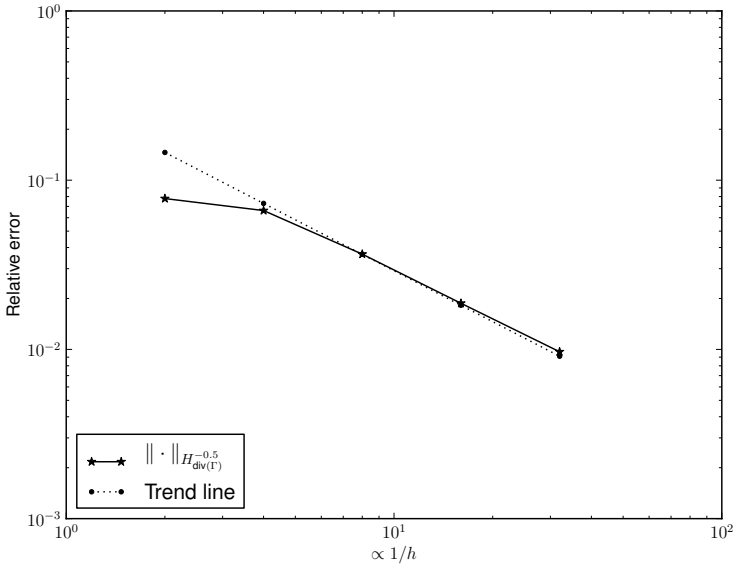



**Figure 5.13:** The convergence obtained for the hat function Eq. 5.9, the rate of convergence is close to the expected rate of $h^1$. In this figure $k_0 h$ ranges from 0.16 to 0.01.

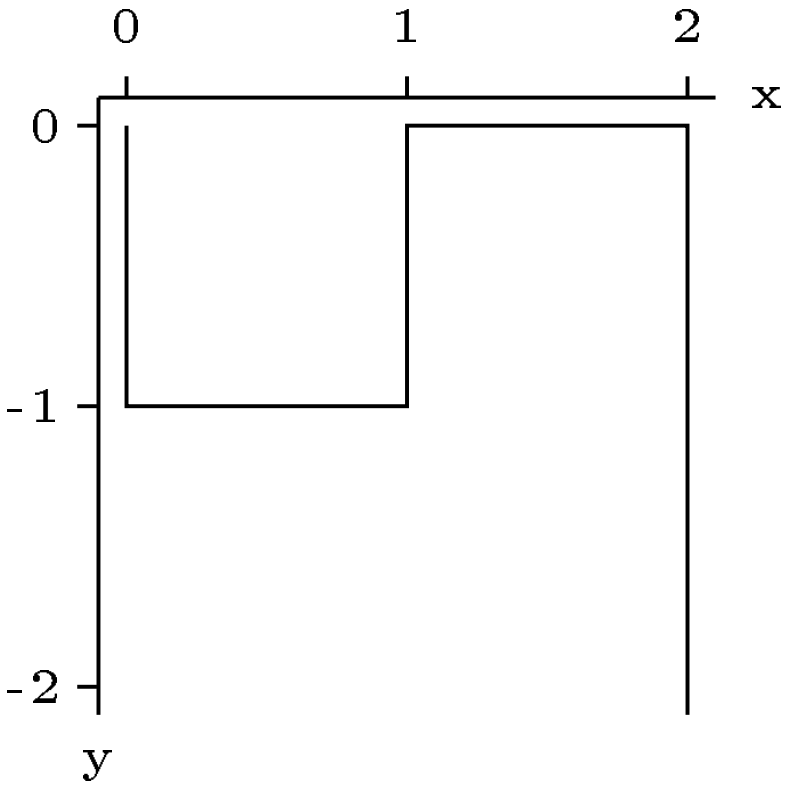



**Figure 5.14:** Side profile of the ribbon like geometry used for Eq. 5.10.

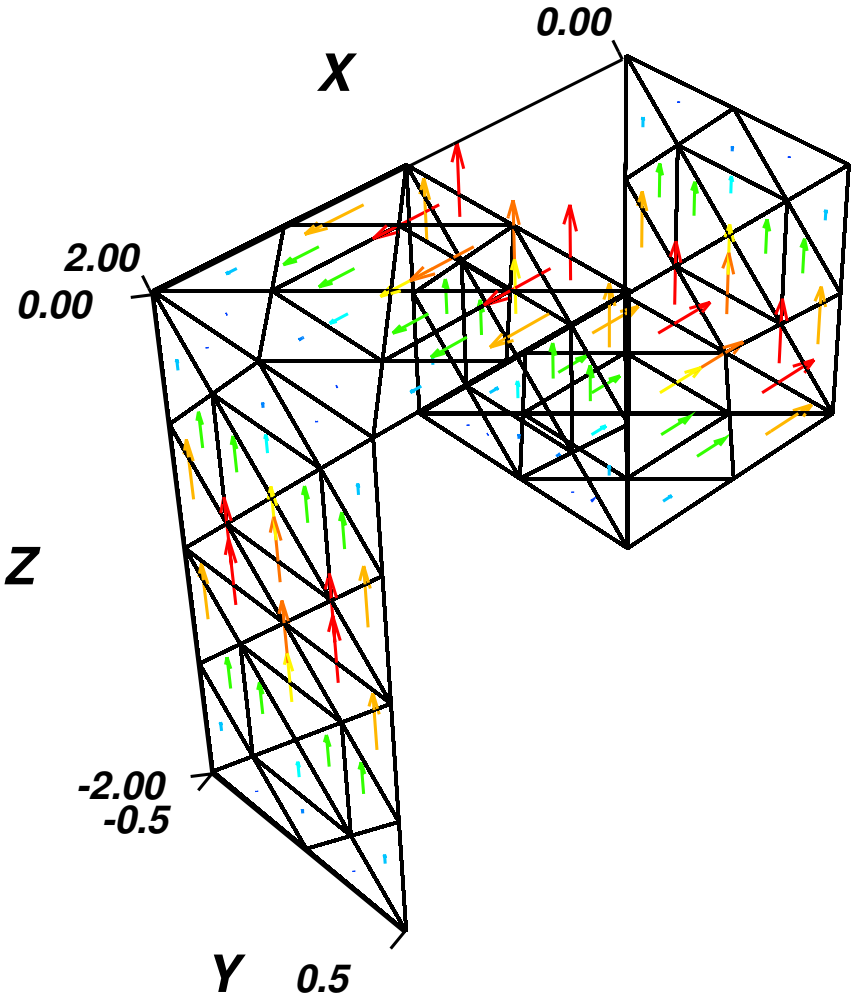



**Figure 5.15:** A ribbon like geometry and visual plot of the MS solution used for Eq. 5.10.

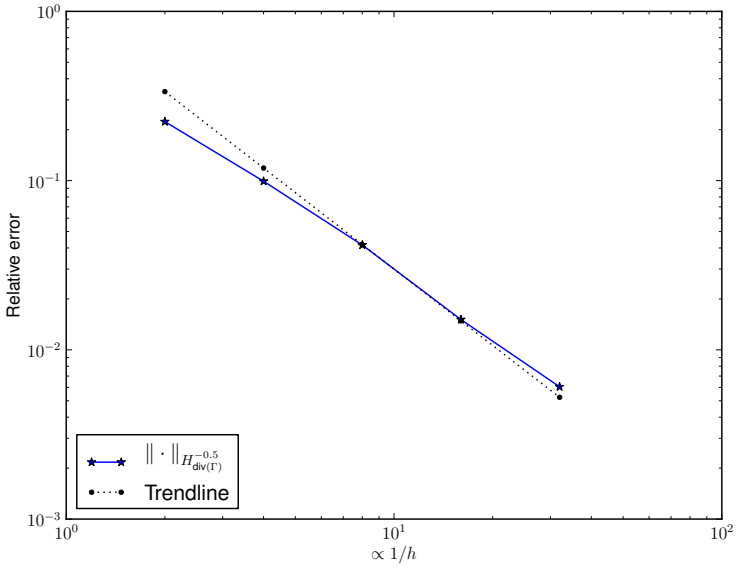



**Figure 5.16:** The computed convergence rate for the MS Eq. 5.10 compared to the expected trend line with slope 3/2. In this figure $k_0 h$ ranges from 0.16 to 0.01.

### A counter example/possible bug

The following examples was encountered while investigating the ability to apply the MMS for MoM to more general geometries. The MS used was,

$$\begin{aligned} \rho &= \sqrt{x^2 + z^2}, \\ \psi &= \cos(y)\sin(\rho), \\ \boldsymbol{J} &= \hat{\boldsymbol{n}} \times \nabla^t \psi, \end{aligned} \tag{5.11}$$

and is divergence free because

$$\nabla^t \cdot \hat{\boldsymbol{n}} \times \nabla^t \psi = -\hat{\boldsymbol{n}} \cdot \nabla \times \nabla^t \psi = 0,$$

as shown in [36] [3].

The geometry used is a PEC plate that is bent at a 45 degree angle. A side profile of the structure is shown in Figure 5.17. This MS, and the 45 degree geometry used, is shown in Figure 5.18.

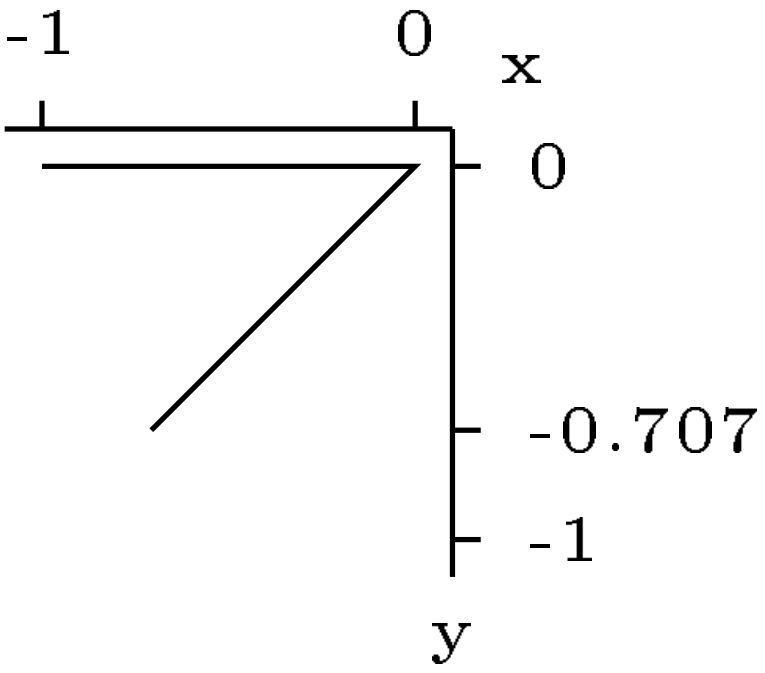



**Figure 5.17:** A side profile of the 45 degree edge geometry used for Eq. 5.11.

The convergence results obtained are shown in Figure 5.19. This clearly indicate that the solution is not converging at the expected rate of 3/2. The reason for this is currently an open question. It is either the MoM code used

[3] Note that $\nabla^t$ can be replaced by $\nabla$ because $\psi$ is a scalar function. [36]

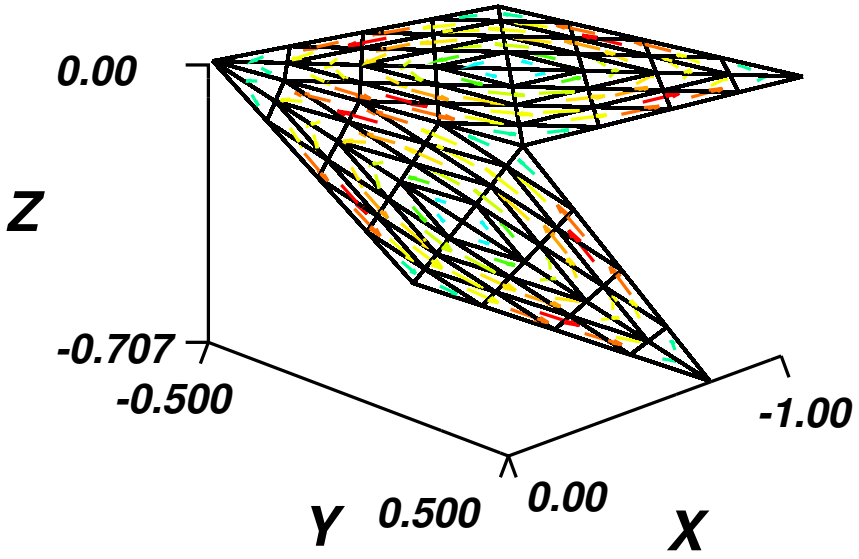



**Figure 5.18:** A 45 degree edge geometry and visual plot of the MS solution used for Eq. 5.11.

that is at fault (the code was originally only tested by its author for flat planar geometries, and near integration *might* be an issue in the matrix assembly), or there is an unexpected reason why the method does not work for these specific geometries. The sharp change in convergence at the third point presented in Figure 5.18 is due to small excitation features that is not properly resolved for rough meshes as observed by the author. It should be noted that similar behaviour is observed for other manufactured solutions.

## 5.6 Mutation Testing

As with the FEM, Mutation Testing will be used to look at the efficacy of testing MoM code with the MMS. The reduced set of mutation operators is used for the MoM as is discussed in Chapter 3. The convergence results generated for all the mutated versions of the code are shown in Figure 5.20. In this particular instance, there were 221 mutations that produced correct executing code. 135 of these mutations are detected by catastrophic failure and 48 are detected by convergence rate error. 38 of the mutations were not detected, but it was found that they are all equivalent mutants. It should once again be remembered that mutation testing naturally produce a large set or mutants that will end catastrophically or result in equivalent mutants. The important result is that all non-equivalent mutants were detected using a rather simple MS.

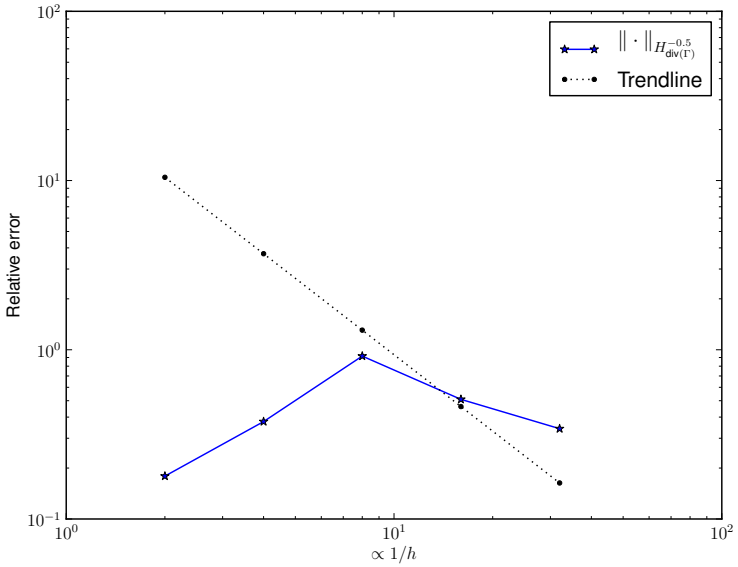



**Figure 5.19:** The computed convergence rate for the MS Eq. 5.11 compared to the expected trend line with slope 3/2. Optimal convergence is not obtained as discussed in the text.

## 5.7 Conclusion

In this chapter the MMS was investigated for the MoM. The MoM was briefly introduced and the solution space was discussed. Selection criteria of the MS was discussed that is specific to the MoM. A brief overview of the theoretical convergence rates that are available in literature for the MoM was discussed and the dependence on the Sobolev regularity was observed. The applicability of the stated convergence rates was discussed in terms of solutions that are smoother than required by the original proofs was noted. The more Sobolev-regular solutions are the result of the freedom to choose a MS that is not necessarily also a valid solution to the BVP solved by the MoM. The necessity of numerical integration was discussed and various methods were compared to one another. The practicality of applying the EFIE operator to a MS was shown using an accurate Mie-series solution. The effect of inaccurate integration was also discussed on the ability to use the MMS as a testing procedure.

Practical examples for the MS were discussed and it was shown that the MMS is a practical method. Mutation testing was also used to look at the applicability of the MMS as a way to test a MoM code. A counter example/possible bug was shown and discussed that indicates the need for further research.

Future research should include an investigation into the ability of the MMS

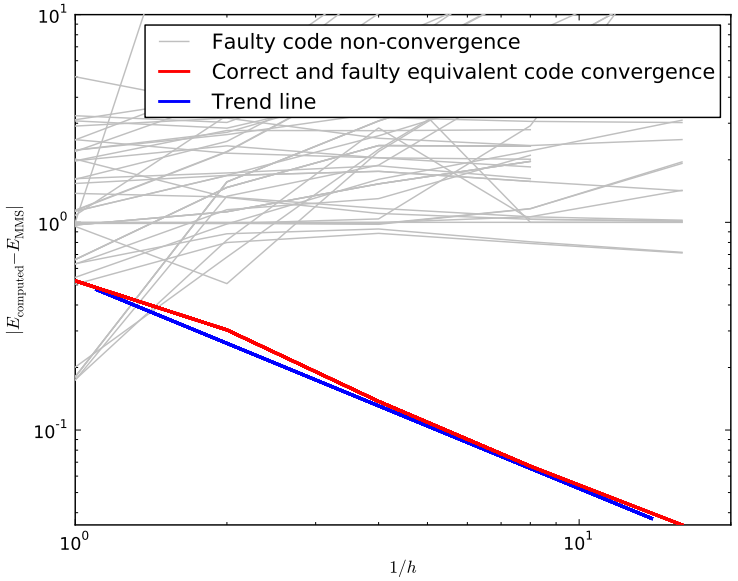



**Figure 5.20:** Convergence results obtained during mutation testing. The $H^0_{\text{div}}(\Gamma)$-norm was used here because it's calculation is significantly faster than the $H^{-1/2}_{\text{div}}(\Gamma)$ norm. Equivalent and correct code convergence are coincident.

to test for the correct implementation of the low frequency breakdown problem. It is known that the system matrix of the MoM is illconditioned at low frequencies and as such slow or no convergence are observed [48]. This would make low frequency problems an excellent testing case for the MMS. The selection of the MS for these problems might be an interesting question. It is known that solutions at DC frequencies are divergence free [48] and the selection of an MS might have to take this fact into consideration. A particular useful geometry that could be used at low frequencies is an annulus because a divergence free solution on this geometry is particularly easy to construct. The low frequency regime is once again an excellent example of a systematic testing approach that should be taken when testing a MoM code: that is, decompose the testing problem to test specific advertised abilities of the CUT.

# Chapter 6

# Conclusion

In this work, the MMS was introduced as a method to test computational code. This method has previously been used for the verification of differential methods in other fields such as the computational fluid dynamics field. Here, the method was investigated in the differential setting for full-wave electromagnetic numerical code. The method was further implemented and investigated for the method of moments.

The key contributions made by this work are the following:

- Various uses of the MMS method were investigated for the FEM.
- The method was implemented for the first time for a BEM and it was shown that the method has merit in verification for BEM code.
- This is also the first time as far as the author could ascertain that the method has been tested by way of mutation testing. A mutation testing system have been introduced for the Python language.
- An automated system was developed that computes the driving and boundary terms that are required for both the FEM and the MoM. This system is very useful in the investigation of various MS.

This is an initial investigation in the implementation of the MMS for electromagnetics. Key areas for future investigation would consists of the following:

- The technique could be used to verify a code with large realistic geometries
- This technique could possibly be used with asymptotic methods and an investigation into the trade-offs made by these methods.
- There are still a few outstanding issues related to the MoM implementation that need to be investigated especially with regard to the application to sharp edges and corners.

# Appendix A

# Vector Identities

The following vector identities should be noted.

$$\nabla \times (\nabla u) = 0 \tag{A.1}$$

$$\nabla \cdot (\nabla \times \boldsymbol{u}) = 0 \tag{A.2}$$

The relationship and the spaces defined for their domains and ranges are interrelated by the following de Rham diagram, as discussed in [54].

$$H^1(\Omega) \xrightarrow{\nabla} H_{\text{curl}}(\Omega) \xrightarrow{\nabla\times} H_{\text{div}}(\Omega) \xrightarrow{\nabla\cdot} L^2(\Omega) \tag{A.3}$$

The definition of these spaces are given in Appendix C.

# Appendix B

# Electromagnetic theory

## B.1 Maxwell's Equations

Maxwell's equations in differential form are summarised here for reference purposes. For more information about the general theory of electromagnetics, any of the well known texts can be consulted [6, 33, 76]. Maxwell's equations are,

$$\nabla \times \boldsymbol{E} = -\frac{\partial \boldsymbol{B}}{\partial t} \qquad \text{(Faraday's law)} \tag{B.1}$$

$$\nabla \times \boldsymbol{H} = \frac{\partial \boldsymbol{D}}{\partial t} + \boldsymbol{J} \qquad \text{(Maxwell-Ampère's law)} \tag{B.2}$$

$$\nabla \cdot \boldsymbol{D} = \rho \qquad \text{(Gauss's law)} \tag{B.3}$$

$$\nabla \cdot \boldsymbol{B} = 0 \qquad \text{(Gauss's law - magnetic)} \tag{B.4}$$

(B.5)

with the meaning of symbols defined in Table 2. Another important equation is the *equation of continuity*,

$$\nabla \cdot \boldsymbol{J} = -\frac{\partial \rho}{\partial t}, \tag{B.6}$$

which is the statement of conservation of charge.

### Time harmonic waves

All electromagnetic waves can be expressed as the sum of harmonic electromagnetic waves due to the Fourier expansion. It is therefore convenient to express Maxwell's equations in a harmonic form.

Equations B.1, B.2 and B.6 for fields that oscillate harmonically with one frequency ($f = 2\pi\omega$) can be written as follows,

$$\nabla \times \boldsymbol{E} = -j\omega \boldsymbol{B} \tag{B.7}$$
$$\nabla \times \boldsymbol{H} = j\omega \boldsymbol{D} + \boldsymbol{J} \tag{B.8}$$
$$\nabla \cdot \boldsymbol{J} = -j\omega \rho \tag{B.9}$$

where the time convention $e^{j\omega t}$ is used and suppressed.

## B.2 Material properties

Electromagnetic energy propagates through a variety of materials. These materials consists of microscopic elements that interacts with this energy. The interaction between electromagnetic energy and physical material is described through macroscopic relations, known as constitutive relations, and is valid for most, but not all, materials encountered.

These constitutive relations are,

$$\boldsymbol{D} = \epsilon \boldsymbol{E} \tag{B.10}$$
$$\boldsymbol{B} = \mu \boldsymbol{H} \tag{B.11}$$
$$\boldsymbol{J} = \sigma \boldsymbol{E}. \tag{B.12}$$

The quantities $\epsilon$, $\mu$ and $\sigma$ are tensors for anisotropic media and scalars for isotropic media. For inhomogeneous materials these quantities are functions of position. Here only simple isotropic media is used.

## B.3 The Vector Wave Equation

Time harmonic fields involve both *electric* ($\boldsymbol{E}$) as well as *magnetic* ($\boldsymbol{H}$) quantities. It is possible to derive a governing partial differential equation involving only one of these field quantities. The vector wave equation for the E-field can be generated by eliminating the magnetic flux from Eq. B.1 using Eq. B.2 and the Constitutive Relations B.10 and B.11. The Vector Wave Equation,

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} - k_0^2 \epsilon_r \boldsymbol{E} + j k_0 Z_0 \boldsymbol{J} = 0, \tag{B.13}$$

also satisfies Gauss's law (B.3). A similar equation can be derived in terms of the H-field.

## B.4 Scalar and Vector potentials

Electric and magnetic time harmonic fields can be described using the scalar and vector potentials [6, 32, 33, 76]. These potentials are described here.

## Vector potential

The magnetic flux $\boldsymbol{B}$ can be written as

$$\boldsymbol{B} = \nabla \times \boldsymbol{A} \tag{B.14}$$

where $\boldsymbol{A}$ is called the *vector potential*. The magnetic field can always be written in the above form due to Eq. B.4 and the fact that the curl of a vector field is in the kernel of the *divergence* operator, see Figure A.3. The definition of the vector potential Eq. B.14 does not uniquely specify $\boldsymbol{A}$ because for every $\boldsymbol{A}$,

$$\boldsymbol{A'} = \nabla \times \boldsymbol{A} + \nabla f, \tag{B.15}$$

for any $f$. This is due to the fact that is that $\nabla f$ is in the kernel of the *curl* operator as can be seen in Eqs. A.1 and A.3. The non-uniqueness of the vector potential forces the need to select a value for $\nabla \cdot \boldsymbol{A}$ and is called the *gauge condition*. A typical gauge to pick is the Lorenz gauge:

$$\nabla \cdot \boldsymbol{A} + \frac{1}{c^2}\frac{\partial \phi}{\partial t} = 0, \tag{B.16}$$

with $\phi$ the scalar potential defined below.

## Scalar potential

The time-harmonic E-field can be written as

$$\boldsymbol{E} = -\nabla\phi - \frac{\partial \boldsymbol{A}}{\partial t}, \tag{B.17}$$

where $\phi$ is the *scalar potential*. The scalar potential is dependant on the gauge choice made for the vector potential.

## Representation in terms of source distributions

The vector and scalar potentials can be written as [33],

$$\boldsymbol{A}(\boldsymbol{r}) = \frac{\mu_0}{4\pi}\int_\Omega \boldsymbol{J}(r')\frac{e^{-jkR}}{R}\,\mathrm{d}r' \tag{B.18}$$

and

$$\phi(\boldsymbol{r}) = \frac{1}{4\pi\epsilon_0}\int_\Omega \rho(r')\frac{e^{-jkR}}{R}\,\mathrm{d}r', \tag{B.19}$$

respectively for the selection of the *Lorenz* gauge. The quantity, $\frac{e^{-jkR}}{R}$, is often written as $G(\boldsymbol{r}, \boldsymbol{r'})$, with $R = |\boldsymbol{r} - \boldsymbol{r'}|$ and is known as the *Green's function*.

## B.5 Boundary conditions

The vector wave equation Eq. B.13 is a partial differential equation that have many solutions. A specific solution to Eq. B.13 is defined with extra information known as boundary conditions. The following is a set of boundary conditions derived using Maxwell's equations.

### At the interface between two media

The following *field continuity conditions* are satisfied at any source-free interface between two media named 1 and 2,

$$\hat{\boldsymbol{n}} \times (\boldsymbol{E}_1 - \boldsymbol{E}_2) = 0 \tag{B.20}$$
$$\hat{\boldsymbol{n}} \cdot (\boldsymbol{D}_1 - \boldsymbol{D}_2) = 0 \tag{B.21}$$
$$\hat{\boldsymbol{n}} \times (\boldsymbol{H}_1 - \boldsymbol{H}_2) = 0 \tag{B.22}$$
$$\hat{\boldsymbol{n}} \cdot (\boldsymbol{B}_1 - \boldsymbol{B}_2) = 0, \tag{B.23}$$

where $\hat{\boldsymbol{n}}$ is the unit vector pointing from medium 2 into medium 1.

If there are sources present on the surface between medium 1 and 2 then Eqs B.21 and B.22 becomes

$$\hat{\boldsymbol{n}} \cdot (\boldsymbol{D}_1 - \boldsymbol{D}_2) = \rho_s \tag{B.24}$$
$$\hat{\boldsymbol{n}} \times (\boldsymbol{H}_1 - \boldsymbol{H}_2) = \boldsymbol{J}_s. \tag{B.25}$$

### At a Perfectly Electric Conducting Surface

When one of the mediums above is a perfectly conducting electric surface the Continuity Conditions Eq. B.20 and Eq. B.22 become

$$\hat{\boldsymbol{n}} \times \boldsymbol{E} = 0 \tag{B.26}$$
$$\hat{\boldsymbol{n}} \cdot \boldsymbol{B} = 0. \tag{B.27}$$

It should be noted that the boundary can always support a surface current and surface charge and therefore Eqs B.24 and B.25 become,

$$\boldsymbol{J}_s = \hat{\boldsymbol{n}} \times \boldsymbol{H} \tag{B.28}$$
$$\rho_s = \hat{\boldsymbol{n}} \cdot \boldsymbol{D} \tag{B.29}$$

respectively.

## B.6 Radiation conditions

Many practical situations contain outer boundaries that recede into infinity. However to obtain a unique solution the *radiation condition* must be satisfied.

For Maxwell's equations, the radiation condition is known as the *Silver-Müller* radiation condition,

$$\lim_{\rho\to\infty} \rho((\nabla \times \boldsymbol{E}^s) \times \hat{\boldsymbol{x}} - ik\boldsymbol{E}^s) = 0, \tag{B.30}$$

is used here with $\rho = |x|$ and the limit being uniform in $\hat{\boldsymbol{x}} = \boldsymbol{x}/|\boldsymbol{x}|$ [54].

# Appendix C

# Sobolev Theory and Energy Spaces

In order to discuss properties of solutions to PDEs, the spaces in which these solutions exists should be discussed. The most suitable spaces for solutions to PDEs that arise from the study of physical phenomena is *Sobolev spaces*. These *Sobolev spaces* embody the concept of finite energy that is critical to all physical processes. In this Appendix the general spaces and theory as applicable to Maxwell's equations will be reviewed but anyone interested in the field should consult fundamental texts dealing with the subject [52]. Of particular interest to engineers would be the book by B.D. Reddy [67] and J.N. Reddy [68] which introduces the subject from their perspective. Also for a readily accessible text on fractional order Sobolev spaces [21] can be consulted. Here basic definitions of the spaces used will be given. General mathematical notation will also be stated.

## C.1 Some standard spaces and notation

### General notation

The following general notation and definitions are used:

$D^\alpha$ All *weak* partial derivatives up to order $\alpha$. The standard multi-index notation for derivatives is used:

$$\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3, ..., \alpha_N)^T \in \mathbb{Z}_+^N$$

where $\mathbb{Z}_+$ is the set of non-negative integers. Let $|\boldsymbol{\alpha}|_1 = \sum_{i=1}^N |\alpha_i|$ and for $f \in \mathcal{C}^{|\boldsymbol{\alpha}|_1}(\Omega)$

$$D^\alpha = \frac{\partial^{\boldsymbol{\alpha}} f}{\partial \boldsymbol{x}^{\boldsymbol{\alpha}}} = \frac{\partial^{|\alpha|_1} f}{\partial x_1^{\alpha_1}, ..., \partial x_N^{\alpha_N}}$$

$\langle u, v \rangle_X$ The inner product appropriate for the space $X$

$\langle u, v\rangle_X$ The bilinear form of the space $X$

$(u, v)_X$ The sesquilinear form for the space $X$, defined as $(u, v)_X = \langle u, \overline{v}\rangle_X$

$L(u)_X$ The linear form defined for the space $X$

## Solution Domains

The following basic domains are defined:

$\mathbb{E}$ The Euclidean three dimensional space ($\mathbb{E} \approx \mathbb{R}^3$)

$\Omega$ A subspace of $\mathbb{R}^n$

$\Gamma$ Either the boundary $\partial\Omega$ of $\Omega$ or a Lipschitz screen embedded in $\Omega$, context will make clear which is used

$\mathcal{D}(\Omega)$ General distributions defined on $\Omega$.

## Function Spaces and Appropriate norms

Functions spaces defined below are all defined on $\Omega$. Similar definitions are available for $\Gamma$ but the appropriate differential operators should be used in these spaces and definitions. Buffa and Christiansen [13] define operators such as *grad*, *div* and *curl* on Lipschitz screens $\Gamma$. Spaces defined on $\Gamma$ is discussed in the next section.

$L^n(\Omega)$ All distributions to the power of $n$ that are integrable on $\Omega$. That is $\int_\Omega |f|^n \,\mathrm{d}x < \infty$.

$L^2(\Omega)$ An important special case of $L^n$. All distributions that are square integrable on $\Omega$.

$C^k(\Omega)$ The set of $k$ times differentiable functions on $\Omega$.

$C_0^k(\Omega)$ The set of functions $f \in C^k(\Omega)$ that have compact support in $\Omega$.

$H^n(\Omega)$ The Hilbert space of all distributions such that $f \in H^n(\Omega) = \{f | f \in L^2, D^\alpha f \in L^2(\Omega), \}$, $n \in \mathbb{Z}^+$, endowed with the inner product [67],

$$\langle u, v\rangle_{H^n} = \int_\Omega \sum_{\alpha \le n} (D^\alpha u)(D^\alpha v) \,\mathrm{d}x \quad \text{for} \quad u, v \in H^n.$$

$H^0(\Omega)$ For $n = 0$ this is equal to $L^2(\Omega)$.

$H^s(\Omega)$ For $s \in \mathbb{R}^+$ is the extension of $H^n$ for derivatives of fractional order. The definition is through intermediary derivatives, the Fourier transform or the interpolation between spaces for various values of $s$. See for instance [52, 68]. Here only $s \in (0, 1)$ will be considered, and from [21],

$$H^s(\Omega) = \{f \in L^2(\Omega) : \frac{|f(x) - f(y)|}{|x - y|^{\frac{n}{2}+s}} \in L^2(\Omega \times \Omega)\}.$$

The endowed inner product is [21],

$$\langle u, v\rangle_{H^s} = \int_\Omega u(x)v(x)\,\mathrm{d}\Omega + \int_\Omega\int_\Omega \frac{[u(x) - u(y)][v(x) - v(y)]}{|x - y|^{\frac{n}{2}+s}}\,\mathrm{d}\Omega_x\,\mathrm{d}\Omega_y$$

$H_0^s(\Omega)$ The completion of $C_0^s(\Omega)$ in the norm $\|\cdot\|_{H_s}$. This represents distributions that have support only on $\Omega$ and have boundary values that are equal to 0 on $\partial\Omega$.

$H^{s'}(\Omega)$ The dual space of $H^s(\Omega)$, that is, the space of bounded linear functionals defined on $H^s(\Omega)$ [67]. The norm to this space is given by,

$$\|u\|_{H^{s'}} = \sup \frac{|\langle u, v\rangle_{H^s}|}{\|v\|_{H^s}}, \quad \forall v \in H^s, v \neq 0.$$

$H^{-s}(\Omega) = H_0^{s'}(\Omega)$, the dual of the completion $H_0^s(\Omega)$. The norm associated with this space is the norm

$$\|u\|_{H^{-s}} = \sup \frac{|\langle u, v\rangle_{H_0^s}|}{\|v\|_{H_0^s}}, \quad \forall v \in H_0^S, v \neq 0.$$

$H^s_{\mathbf{div}}(\Omega)$ The space of *div*-conforming distributions.

$$H^s_{\mathrm{div}} = \{f \in H^s(\Omega)|\nabla \cdot f \in H^s(\Omega)\}, \tag{C.1}$$

equipped with the graph norm

$$\|f\|_{\mathrm{div},H_s} = \sqrt{\|f\|^2_{H^s} + \|\nabla \cdot f\|^2_{H^s}}. \tag{C.2}$$

$H^s_{\mathbf{curl}}(\Omega)$ The space of *curl*-conforming distributions.

$$H^s_{\mathrm{curl}} = \{f \in H^s(\Omega)|\nabla \times f \in H^s(\Omega)\}, \tag{C.3}$$

equipped with the graph norm

$$\|f\|_{\mathrm{curl},H_s} = \sqrt{\|f\|^2_{H^s} + \|\nabla \times f\|^2_{H^s}}.$$

$H^s_{\mathbf{div,curl}}(\Omega)$ The intersection between $H^s_{\mathrm{div}}(\Omega)$ and $H^s_{\mathrm{curl}}(\Omega)$, that is, $H^s_{\mathrm{div,curl}}(\Omega) = H_{\mathrm{div}}(\Omega) \cap H_{\mathrm{curl}}\Omega$.

## Solution Space to the EFIE

Buffa and Christiansen [13] defines the concept of Sobolev spaces on $\Gamma$ a Lipschitz screen in $\Omega$. Spaces and trace operators defined by them are described here for convenience.

### Preliminary spaces

$\gamma$ The trace of a continuous function defined on $\overline{\Omega} = \Omega \cup \partial\Omega$ is the value of the function on the boundary, $\partial\Omega$. Distributions defined in $H^s$ is not necessary continuous and the trace might not be defined.

$\gamma_T$ The *tangential* trace operator projects the vector field that it operates on, onto the tangential field on the space defined by $\Gamma$.

$H^s(\Gamma)$ For $s > 1$, this is the space of traces of $H^{1/2+s}(\mathbb{E})$. The norm $|\cdot|_s$ induced by $\gamma$ makes this a Hilbert space.

$H_T^s(\Gamma)$ For $s > 0$, this is the space of tangential traces of $H_T^{1/2+s}(\mathbb{E})$. The subscript T refers to the fact that vector spaces are involved and that more specifically on $\Gamma$ these are *tangential*. The norm induced by $\gamma_T$, $|\cdot|_s$, makes this a Hilbert space.

### EFIE solution spaces

The following spaces are as defined by Buffa and Christiansen [13] for Lipschitz screens and by Hsiao and Kleinman [36] for smooth scatterers. These are solution spaces for the EFIE (Eq. F.3) and solutions to the EFIE that also induce solutions to the scattering BVP (Eq. F.1) respectively.

$H_{\mathbf{div}}^s(\Gamma)$ For Lipschitz screens and closed polyhedral scatterers this denotes the space of elements $u \in H_T^{-s}(\Gamma)'$ such that $\nabla \cdot u \in H^{-s}(\Gamma)'$. This space is equipped with the graph norm,

$$\|f\|_{H_{\mathrm{div}}^s(\Gamma)} = \sqrt{\|f\|^2_{H_T^{-s}(\Gamma)'} + \|\nabla \cdot f\|^2_{H^{-s}(\Gamma)'}}$$

which makes it a Hilbert space. This is the *div*-conforming space that the current on a metal object will occupy. For smooth closed scatterers Hsiao and Kleinman [36] defined this space for $s = 1/2$ as

$$H_{\mathrm{div}}^{1/2}(\Gamma) = \{\boldsymbol{u} | \hat{\boldsymbol{n}} \cdot \boldsymbol{u} = 0, \boldsymbol{u} \in H^{1/2}(\Gamma), \nabla^t \cdot \boldsymbol{u} \in H^{1/2}(\Gamma)\},$$

with the usual graph norm.

$X^s$ Let $X^s$ with elements $u$ be the subspace of $H_{\mathrm{div}}^s(\Gamma)$ such that for all $v \in \mathcal{D}(\overline{\Gamma})$:

$$\langle u, \nabla v \rangle + \langle \nabla \cdot u, v \rangle = 0$$

## Norm calculation

The norm

$$\|u\|_{H^{-s}} = \sup \frac{|\langle u, v\rangle_{H_0^s}|}{\|v\|_{H_0^s}}, \quad \forall v \in H_0^S, v \neq 0,$$

is clearly not computable and is estimated using an equivalent norm. The equivalent norm used here is

$$\|u\|_{H^{-s}} \approx \sqrt{\langle u, V(u)\rangle},$$

with V the Laplace single layer operator $V(u) = \int_\Gamma \frac{u}{4\pi|\boldsymbol{x}-\boldsymbol{y}|} d\sigma_s$ as shown in [73]. Theoretical convergence rates are valid because convergence carries over to equivalent norms [67, p. 107]. This norm is however computationally intensive because a double integral have to be performed over the computational domain, that is, one to compute the Laplace single layer potential and the other to compute the inner product of the equivalent norm.

# Appendix D

# The Galerkin Process and discretisation

Exact solutions to boundary value problems and integral equations are not always available and as such approximate solutions are sought. Approximate solutions are often written as variational problems or as weighted residual problems. Both of these problems can be solved using the Galerkin process. The variational problem might be a variational boundary value problem [67,68] derived from a boundary value problem and the weighted residual problem may be derived from an integral problem for instance [17]. From here on, *the problem* refers to either one of the above mentioned problems.

The problem is written as: find the solution $\boldsymbol{u} \in V$ such that

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle_V = L(\boldsymbol{v})_V \tag{D.1}$$

for all $\boldsymbol{v}$ in the appropriate infinite dimensional search space $(V)$. $\langle \cdot, \cdot \rangle$ is the appropriate bilinear form and $L(\cdot)$ is the appropriate linear form.

The Galerkin method states that the solution and the weighting function should be from the same finite dimensional function space used to solve/approximate the solution [67, 68] – above still infinite dimensional.

## D.1 Domain discretisation

The domain $\Omega$ is discretised into a *mesh* set $\tau_h$. This mesh consists of $N_h$ cells $K_i$, such that the union of all cells span the solution domain but only intersects on the boundaries of these elements. That is,

$$\Omega = \bigcup_{i=1}^{N_h} K_i,$$

provided that

$$K_m \cap K_n = \partial K_m \cap \partial K_n, m \neq n.$$

The subscript $h$ refers to a spatial measure of the largest element in the set.

Following [11], if the constraint

$$K_m \cap K_n \in \{\emptyset; n_{mn}; e_{mn}; f_{mn}\} \quad m \neq n,$$

with $n_{mn}$ a mutual element node, $e_{mn}$ a mutual element edge and $f_{mn}$ a mutual element face is further placed on the mesh then the mesh is known as *regular*. It is preferable that a mesh is regular especially when considering mesh refinement where singularities are present [54, p. 169].

A basis set is defined on each of the elements of this mesh. Basis elements may span more than one cell and is usually chosen such that the desired continuity between elements hold. This basis is also chosen such that the space spanned by the basis approximates the space in which the solution is sought.

The appropriate basis functions and finite element spaces spanned will be discussed where the relevant method is discussed.

### Triangles and Tetrahedra

Elements used to discretise the problem domain depends on the dimensionality of the problem considered. In 2D the geometry can be discretised into polygonal elements. Typically triangles of quadrilaterals are used. These are convenient because any polygonal surface can be subdivided into a regular set of these elements [75]. For curvilinear geometry curvilinear elements are typically used. Curvilinear elements are often defined from a base rectilinear element and transformed onto applicable curvilinear elements. These curvilinear elements are not considered here.

Three dimensional geometries can be discretised using either 2D elements, those mentioned above, or typically 3D elements such as tetrahedra, triangular prisms of rectangular bricks [41]. Two dimensional elements are only used when a thin surface shell of the geometry needs to be meshed as is often the case for the MoM. Three dimensional elements are used when the volume of the geometry needs to be meshed. Tetrahedral elements are used here because any polyhedral shape can be divided into a set of tetrahedra [75]. Three dimensional elements may also be curvilinear and are derived an a manner similar to those for 2D elements. In this work 3D curvilinear elements are not used.

## D.2 Matrix System Development

Let $\boldsymbol{\phi}_N$ be the set of basis functions chosen for the problem and associated mesh. The sought unknown function $\boldsymbol{u}$ and weighting function $\boldsymbol{v}$ is discretised

as follows,

$$\boldsymbol{u_h} = \sum_{i=1}^{N_h} x_i \boldsymbol{\phi}_i \quad \text{and} \quad \boldsymbol{v_h} = \sum_{j=1}^{N_h} d_j \boldsymbol{\phi}_j.$$

These properly constrained approximations to $\boldsymbol{u}$ and $\boldsymbol{v}$ is substituted into Eq. D.1, that is

$$\langle \boldsymbol{u_h}, \boldsymbol{v_h} \rangle = L(\boldsymbol{v_h}).$$

Expressing this in terms of the chosen basis functions results in,

$$\sum_{i=1}^{N_h} \sum_{j=1}^{N_h} \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle_V x_i d_j = \sum_{j=1}^{N_h} L(\boldsymbol{\phi}_j)_V d_j.$$

The coefficients $d_j$ are arbitrary and can be dropped. The resulting equation to solve is

$$\sum_{i=1}^{N_h} A_{ij} x_i = b_j, \quad j = 1, ..., N_h$$

with $A_{ij} = \langle \boldsymbol{\phi}_i, \boldsymbol{\phi}_j \rangle_V$ and $b_j = L(\boldsymbol{\phi}_j)_V$. This is clearly a matrix equation

$$\boldsymbol{Ax} = \boldsymbol{b} \tag{D.2}$$

that have to be solved numerically.

### Continuity enforcement

A note about continuity enforcement is appropriate. The continuity between elements $K_i$ and $K_j$ is enforced by equating degrees of freedom that is shared by adjacent elements.

## D.3 Basis functions

Basis functions are typically chosen to solve a particular kind of problem and aid in the numerical stability of the solution. It is also advantageous to model specific features present in the expected solution. These issues will be discussed next and are based on [11, 17, 41, 54].

### Vector and nodal elements

The quantities typically solved using electromagnetic full-wave solvers are E-fields and H-fields. These fields are vector quantities and therefore the vector nature needs to be modelled by the approximate solution space.

There are two distinct ways to represent a vector field using basis functions, namely a collection of scalar elements or the use of vector elements. For nodal

elements N sets of basis functions are defined for a N-dimensional problem. Each basis set represents a vector component. The degrees of freedom is typically located on the corners of the mesh elements so as to enforce continuity between elements. There are however well known problems when modelling the E-field using nodal elements, most notably spurious modes [17]. There are stabilisation methods [41, p.199] that can be applied to correct for this but these are not used here.

Vector elements are mostly used in full-wave simulations. Vector elements consists of basis functions that explicitly have specific vector defined components throughout the element basis space. Degrees of freedom are typically defined on edges, faces and volumes of the mesh elements. Specific kinds of continuity defined by the basis elements are therefore enforced between edges and faces of adjoining elements. The curl- and div-conforming elements by Nédélec [56, 57] are widely used. The most notable reason for the wide spread use of these elements is that it is possible to explicitly enforce the curl- or div-conforming nature of the fields and quantities modelled. More about the curl- and div-conforming properties will be discussed next.

## Curl-conforming

When modelling the E-field in a full-wave code it is important to model the curl-conforming nature of the E-field, $\boldsymbol{E} \in H^0_{\text{curl}}(\Omega)$ [36]. The basis functions ($\boldsymbol{\phi}$) chosen should therefore be such that the curl-conforming nature of the computed E-field be represented accurately. The classical basis functions typically used are the first curl conforming elements by Nédélec [56] and are used here. Basis functions defined on a mesh are said to be curl-conforming if tangential continuity is enforced on the faces/edges of elements [56].

The proper enforcement of the tangential-continuity and hence curl-conforming nature of these types of elements eliminates the spurious mode problem that is encountered when using nodal basis functions [11, 17, 41].

Explicit equations for these elements are not given here as they are readily available in literature [41,54] and because they were not manually implemented by the author but used through the FIAT package [46].

## Div-conforming

When modelling the current in a full-wave code it is important to model the div-conforming nature of the current, ($\boldsymbol{J} \in H^{-1/2}_{\text{div}}(\Gamma)$) [13]. The basis functions($\boldsymbol{\phi}$) chosen are therefore div-conforming when modelling the current as is done with MoM-code. Here only the first order Raviart-Thomas (RT) [66] basis functions are used, they are also known as Rao-Wilton-Glisson (RWG) [65] elements. Nédélec have extended the RT elements to three dimensions [54, 56] but only the two dimensional case is used here. Basis functions

defined on a mesh are said to be div-conforming if normal continuity is enforced on the faces/edges of elements [56].

The explicit expression for RWG elements defined on two adjacent triangles namely $T_i^+$ and $T_i^-$ sharing edge $i$ between them is [65],

$$\boldsymbol{f_i} = \begin{cases} \frac{l_i}{2A_i^+}\boldsymbol{\rho}_i^+ & \text{for } \boldsymbol{r} \text{ on } T_i^+ \\ \frac{l_i}{2A_i^-}\boldsymbol{\rho}_i^- & \text{for } \boldsymbol{r} \text{ on } T_i^- \\ 0 & \text{otherwise} \end{cases} \tag{D.3}$$

where $\rho_i^+$ is the vector pointing from the vertex of $T_i^+$ that is not shared by the triangles to a point on $T_i^+$ and $\rho_i^-$ is the vector pointing to the vertex that is not shared by the triangles of $T_i^-$ from a point on $T_i^-$. The area of $T_i^+$ and $T_i^-$ is defined by $A_i^+$ and $A_i^-$ respectively.

## Higher order elements

Both the div- and curl-conforming elements are specified using arbitrary order polynomial spaces. The elements specifically introduced by Nédélec [56] makes a trade off between approximation of the field and the curl/div of the field and give equal polynomial approximation to both [54, 56]. There is an extensive study into implementations of these higher order basis functions and they are classified by the approximation ability of both the field and the div/curl of the field. For a review of this the interested reader may consult [17]. This issue is not pursued further here except to note that here the original mixed-order elements of Nédélec is used.

# Appendix E

# The Finite Element Method

The Finite Element Method (FEM) is a well known technique used to solve boundary value problems (BVP). This method will be discussed here for reference. Readers interested in more detail about the method is advised to consult [17, 41, 54, 75]. Davidson [17] gives a good introduction to the basic methods of numerical electromagnetics, Jin [41] has a good practical perspective while Monk [54] has a strong mathematical perspective.

The specific FEM E-field formulation used here is most readily encountered in literature and is used quite widely. Inhomogeneous boundary conditions for Dirichlet as well as Neumann boundary conditions will be considered as these are the cases needed to implement the MMS. Inhomogeneous Dirichlet boundary conditions are not typically considered for practical problems but the difference between homogeneous and inhomogeneous Dirichlet boundary conditions is trivial. The cavity problem is exclusively considered here for the FEM method. Next the Maxwell BVP for cavities will be considered after which the appropriate variational boundary value problem will be discussed. This Appendix will conclude with the discretisation and basis functions required for the FEM.

## E.1 Boundary Value Problem

The cavity problem considered in this work is as shown in Figure E.1. The domain $\Omega$ is a subset of free space ($\Omega \subset \mathbb{R}^3$). The boundary of this space is denoted by $\Gamma = \partial\Omega$ and can be subdivided between *Dirichlet* boundaries ($\Gamma_D$) and *Neumann* boundaries ($\Gamma_N$). The boundaries $\Gamma_D$ and $\Gamma_N$ do not intersect and span the entire boundary $\partial\Omega$. In symbols the boundary requirements are,

$$\Gamma_D \cap \Gamma_N = \emptyset \tag{E.1}$$

$$\Gamma_N \cup \Gamma_N = \Gamma. \tag{E.2}$$

The Maxwell BVP for time-harmonic E-field solutions defined on the above

described domain is then stated as: Solve the vector wave equation Eq. B.13,

$$\nabla \times \frac{1}{\mu_r} \nabla \times \boldsymbol{E} - k_0^2 \epsilon_r \boldsymbol{E} + j k_0 Z_0 \boldsymbol{J} = 0 \text{ on } \Omega, \tag{E.3}$$

subject to the boundary conditions

$$\hat{\boldsymbol{n}} \times \boldsymbol{E} = \boldsymbol{P} \qquad \text{on } \textit{Dirichlet} \text{ boundaries } (\Gamma_D) \tag{E.4}$$

$$\hat{\boldsymbol{n}} \times \nabla \times \boldsymbol{E} = \boldsymbol{U} \qquad \text{on } \textit{Neumann} \text{ boundaries } (\Gamma_N). \tag{E.5}$$

These stated boundary conditions are appropriate as they allow a unique solution to Maxwell's equations as stated by the uniqueness theorem [6, p. 313]. The uniqueness theorem states that for time harmonic electromagnetic problems either the tangential electric or tangential magnetic or a combination of the two must be specified at all boundary locations. These quantities are stated by the Dirichlet and Neumann boundary conditions respectively.

This formulation is know as a *strong* form.

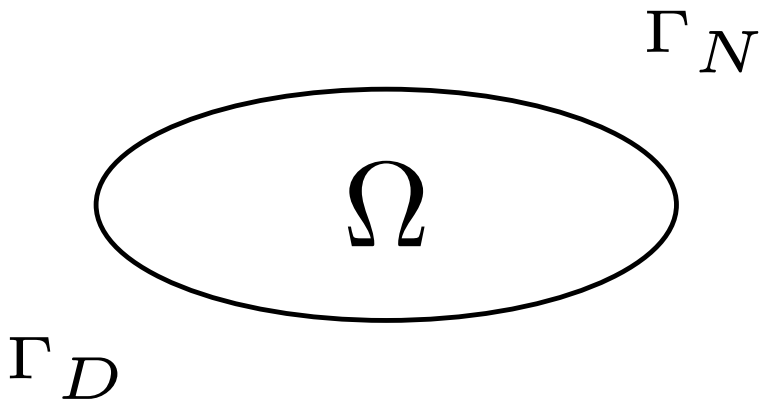



**Figure E.1:** The BVP domain, with $\Omega$ the interior domain and the boundary $\Gamma$ divided between the Dirichlet ($\Gamma_D$) and Neumann ($\Gamma_N$) non-intersecting boundaries.

## E.2 Variational Boundary Value Problem

Finding an exact analytical solution to the strong form BVP discussed above is not always possible. A numerical solution to the problem is required and towards this end the problem is stated as an equivalent *weak* form known as the variational boundary value problem (VBVP). The VBVP is known as a weak formulation because the solution space is larger and hence it is easier to find approximate solutions. For more general background regarding the weak form [67, 68] can be consulted.

In this section the VBVP for the Maxwell BVP will merely be stated and anyone interested in it's derivation can consult [11, 17, 41, 54].

The VBVP derived by [11] is stated as follows. Find a function $\boldsymbol{E}$ such that

$$\begin{cases} B(\boldsymbol{E}, \boldsymbol{W})_\Omega = L(\boldsymbol{W})_\Omega \\ \boldsymbol{E} \in V \end{cases} \tag{E.6}$$

for all $\boldsymbol{W} \in V$. Where

$$B(\boldsymbol{E}, \boldsymbol{W})_\Omega = (\frac{1}{\mu_r} \nabla \times \boldsymbol{E}, \nabla \times \boldsymbol{W})_\Omega - k_0^2 (\epsilon_r \boldsymbol{E}, \boldsymbol{W})_\Omega$$

$$L(\boldsymbol{W})_\Omega = -\int_{\Gamma_n} \frac{1}{\mu_r} \boldsymbol{U} \cdot \boldsymbol{W} \, \mathrm{d}S - jk_0 Z_0 (\boldsymbol{J}, \boldsymbol{W})_\Omega,$$

with $(\cdot, \cdot)_\Omega$ the *sesquilinear form* ($(u, v)_\Omega = \langle u, \overline{v} \rangle_\Omega$).

For the FEM discussed here $V = \{\boldsymbol{u} \in H_{\text{curl}}(\Omega) | \hat{\boldsymbol{n}} \times \boldsymbol{u} = \boldsymbol{P}\}$. The Dirichlet boundary conditions is known as the *essential* boundary condition because it must specifically be enforced by restriction of the solution space and the Neumann boundary condition is known as the *natural* boundary condition because it is included in the variational form [11, 41].

The Rayleigh-Ritz method can be used to cast this VBVP into a stationary functional [41]. The solution to this functional, when it is rendered stationary, is equivalent to the solution of the above variational form. This formulation will not be pursued here.

## E.3 Discretised Variational Boundary Value Problem

The VBVP, Eq. E.2, is clearly in the form that is required by Eq. D.1 to apply the Galerkin process. What remains is to select an appropriate basis functions such that $\boldsymbol{E}_h$ is in the correct function space. For this work 2D geometries are discretised using triangles and 3D geometries using tetrahedra as discussed in Appendix D.1.

# Appendix F

# The Method of Moments

The Method of Moments (MoM) is a boundary element method (BEM) that is typically used to solve scattering and antenna methods. In this section an overview of the MoM will be given for scattering problems. For more information regarding the MoM, well known texts such as [17, 73, 80] may be consulted amongst others. First the scattering BVP will be discussed. After that the integral equation used to solve this problem will be introduced. This will be concluded with a discussion of the discretisation of the integral equation.

## F.1 Scattering problem

The scattering problem discussed here is one that consists of a simply connected PEC metal scatterer in free-space with no dielectric or magnetic objects present. The object can be polyhedral or be a screen and can be as general as a non-orientable Lipschitz screen. This allows a large class of problems to be solved using this method. The exterior free space will in this instance be represented by $\Omega$ and the scatterer will be represented by $\Gamma$.

Scattering from a PEC object can be expressed as the following BVP problem

$$\begin{aligned} \nabla \times \nabla \times \boldsymbol{E}^{sc} - k_0^2 \boldsymbol{E}^{sc} &= 0 && \text{in } \Omega, \\ \hat{\boldsymbol{n}} \times \boldsymbol{E}^{sc} &= -\hat{\boldsymbol{n}} \times \boldsymbol{E}^{inc} && \text{on } \Gamma \end{aligned} \tag{F.1}$$

and the radiation condition Eq. B.30. $\boldsymbol{E}^{sc}$ is the scattered E-field and $\boldsymbol{E}^{inc}$ is the incident E-field.

## F.2 Integral Equation

The solution to Eq. F.1 can be found by using the integral equation derived in this section. The derivation of the EFIE given here is based on the method introduced by [65]. The E-field due to sources on an object can be expressed

as the sum of the scalar and vector potentials using Eq. B.17. That is, the scattered E-field ($\boldsymbol{E}^s$) can be written as

$$\boldsymbol{E}^s = -\nabla\phi - j\omega\boldsymbol{A} = -\boldsymbol{E}^{inc}. \tag{F.2}$$

The well known EFIE equation results by substituting the potentials written as functions of source terms, Eq. B.18 and Eq. B.19, into the above equation and noting that the equation of continuity Eq. B.9 states charge in terms of current. The EFIE is

$$\hat{\boldsymbol{n}}\times\boldsymbol{E}^{inc} = \hat{\boldsymbol{n}}\times\mathrm{PV}\int_\Gamma \left[jk_0 Z_0 \boldsymbol{J}_s(\boldsymbol{r'})G(\boldsymbol{r},\boldsymbol{r'}) + \frac{Z_0}{jk_0}\left\{\nabla'\cdot\boldsymbol{J}_s(\boldsymbol{r'})\nabla' G(\boldsymbol{r},\boldsymbol{r'})\right\}\right]\mathrm{d}S'. \tag{F.3}$$

The letters PV is there to remind us that the integral is a principle value integral. The operator on the right side is often expressed as $\hat{\boldsymbol{n}}\times\mathcal{T}(\boldsymbol{J})$. This integral equation is a *Fredholm integral equation of the first kind* because the unknowns are all in the kernel of the integral [17]. Solutions to this integral equation is in $H^{1/2}_{\mathrm{div}}(\Gamma)$ as reasoned by [36] for closed smooth scatterers and for general Lipschitz screens in $H^{-1/2}_{\mathrm{div}}(\Gamma)$ by [13]. Not all solutions that are in $H^{-1/2}_{\mathrm{div}}(\Gamma)$ are however solutions to the Scattering BVP (Eq. F.1). Buffa and Christiansen [13] have shown that only solutions of the integral representation Eq. F.3 restricted to the subspace

$$X^s = \{u \in H^s_{\mathrm{div}}(\Gamma) | \langle u, \nabla v\rangle + \langle \nabla\cdot u, v\rangle = 0, \forall v \text{ on } \Gamma\}$$

does indeed solve the BVP (Eq. F.1).

## F.3 Developing the weighted residual form

To numerically solve the EFIE, a weighted residual approach can be taken [65, 80]. The EFIE (Eq. F.3) will be represented using the vector and scalar potentials for ease of presentation in what follows. Weighing Eq. F.2 with a testing function $\boldsymbol{W}$ using the appropriate inner product yields

$$\langle \boldsymbol{E}^{inc}, \boldsymbol{W}\rangle = \langle \nabla\phi, \boldsymbol{W}\rangle + j\omega\langle \boldsymbol{A}, \boldsymbol{W}\rangle.$$

The application of the gradient operator on the scalar potential $\phi$ results in singularities that complicates the evaluation of the inner products considerably. Therefore the following identity is used [17],

$$\langle \nabla\phi, \boldsymbol{W}\rangle = \int_S \nabla\phi\cdot\boldsymbol{W}\,\mathrm{d}S = -\int_S \phi\nabla^t\cdot\boldsymbol{W}\,\mathrm{d}S, \tag{F.4}$$

to shift the differential operator to the testing function.

With

$$\begin{aligned} L(\boldsymbol{W})_\Gamma &= \langle \boldsymbol{E}^{inc}, \boldsymbol{W} \rangle_\Gamma \\ B(\boldsymbol{J}, \boldsymbol{W})_\Gamma &= \langle \nabla\phi, \boldsymbol{W} \rangle_\Gamma + j\omega \langle \boldsymbol{A}, \boldsymbol{W} \rangle_\Gamma, \end{aligned} \tag{F.5}$$

in the form required to apply the Galerkin process, with the exception that the testing of the scalar potential in Eq. F.5 is interpreted as defined in Eq. F.4. It should be emphasised that the testing function space consists of vector functions that are *tangential* to the geometry surface $\Gamma$ and therefore $\langle \boldsymbol{E}^{inc}, \boldsymbol{W} \rangle$ is equivalent to $\langle \hat{\boldsymbol{n}} \times \boldsymbol{E}^{inc}, \boldsymbol{W} \rangle$.

By using the Galerkin process and the RWG [65] basis functions defined in D.3 the well known Method of Moments result.

# Bibliography

[1] J.M. Adams and A. Taha. An experiment in software redundancy with diverse methodologies. In *System Sciences, 1992. Proceedings of the Twenty-Fifth Hawaii International Conference on*, volume II, pages 83–90 vol.2, Jan 1992.

[2] D.P. Aeschliman, W.L. Oberkampf, and F.G. Blottner. Proposed methodology for computational fluid dynamics code verification, calibration, and validation. In *ICIASF Record, International Congress on Instrumentation in Aerospace Simulation Facilities*, pages 27.1–27.13, 1995.

[3] John Anderson. *Computational Fluid Dynamics*. McGraw-Hill Science/Engineering/Math, 1st edition, Feb. 1995.

[4] J.H. Andrews, L.C. Briand, and Y. Labiche. Is mutation an appropriate tool for testing experiments? In *Software Engineering, 2005. ICSE 2005. Proceedings. 27th International Conference on*, pages 402–411, May 2005.

[5] H. Argrawal, R.A. DeMillo, B. Hathaway, W. Hsu, W. Hsu, E.W. Krauser, R.J. Martin, A.P. Mathur, and E. Spafford. Design of mutant operators for the C programming language. Technical report, Purdue, 1989.

[6] Constantine A. Balanis. *Advanced Engineering Electromagnetics*. Wiley, 2nd edition, Jan 2012.

[7] A. Bendali. Numerical analysis of the exterior boundary value problem for the time-harmonic Maxwell equations by a boundary finite element method part 1: The continuous problem. *Mathematics of Computation*, 43(167):pp. 29–46, 1984.

[8] A. Bendali. Numerical analysis of the exterior boundary value problem for the time-harmonic Maxwell equations by a boundary finite element method part 2: The discrete problem. *Mathematics of Computation*, 43(167):pp. 47–68, 1984.

[9] A. Bespalov and N. Heuer. The hp-BEM with quasi-uniform meshes for the electric field integral equation on polyhedral surfaces: A priori error analysis. *Applied Numerical Mathematics*, 60(7):705–718, 2010.

[10] Matthys M. Botha. Analysis and augmentation of the Duffy transformation for near-singular integrals. In *IEEE International Symposium on Antennas and Propagation*, July 2012. Chicago, USA.

[11] M.M. Botha. *Efficient finite element electromagnetic analysis of antennas and microwave devices: The FE-BI-FMM formulation and a posteriori error estimation for p adaptive analysis.* PhD thesis, University of Stellenbosch, 2002.

[12] M.M. Botha and T. Rylander. Error analysis of singularity cancellation quadrature on curvilinear triangles. In *Electromagnetics in Advanced Applications, 2007. ICEAA 2007. International Conference on*, pages 810–813, Sept. 2007.

[13] A. Buffa and S.H. Christiansen. The electric field integral equation on Lipschitz screens: Definitions and numerical approximation. *Numerische Mathematik*, 94(2):229–267, 2003.

[14] Weng Cho Chew, J. M. Jin, and E. Michielssen, editors. *Fast and Efficient Algorithms in Computational Electromagnetics (Artech House Antennas and Propagation Library).* Artech House Publishers, first edition edition, Jan 2001.

[15] S.H. Christiansen. Discrete Fredholm properties and convergence estimates for the electric field integral equation. *Mathematics of Computation*, 73(245):143–167, 2004.

[16] M. Costabel and M. Dauge. Singularities of electromagnetic fields in polyhedral domains. *Archive for Rational Mechanics and Analysis*, 151(3):221–276, 2000.

[17] D.B. Davidson. *Computational Electromagnetics for RF and Microwave Engineering.* Cambridge University Press, 2010.

[18] R.A. DeMillo, R.J. Lipton, and F.G. Sayward. Hints on test data selection: Help for the practicing programmer. *Computer*, 11(4):34–41, April 1978.

[19] FEniCS Developers. *The FEniCS Project*, November 2012. http://fenicsproject.org/.

[20] Sucem-FEM Developers. *SUCEM-FEM*, November 2012. https://github.com/cemagg/sucem-fem.

[21] E. Di Nezza, G. Palatucci, and E. Valdinoci. Hitchhiker's guide to the fractional Sobolev spaces. *Bulletin des Sciences Mathematiques*, 136(5):521–573, 2012.

[22] Michael G. Duffy. Quadrature over a pyramid or cube of integrands with a singularity at a vertex. *SIAM Journal on Numerical Analysis*, 19(6):pp. 1260–1262, 1982.

[23] D.A. Dunavant. High degree efficient symmetrical Gaussian quadrature rules for the triangle. *International Journal for Numerical Methods in Engineering*, 21(6):1129–1148, 1985.

[24] L. Eça and M. Hoekstra. Evaluation of numerical error estimation based on grid refinement studies with the method of the manufactured solutions. *Computers and Fluids*, 38(8):1580–1591, 2009.

[25] Edward H. Newman Edmund K. Miller, Louis Medgyesi-Mitschang, editor. *Computational Electromagnetics (IEEE Press Selected Reprint Series)*. Institute of Electrical & Electronics Engineering, 1 1992.

[26] P.W. Fink, D.R. Wilton, and M.A. Khayat. Simple and efficient numerical evaluation of near-hypersingular integrals. *Antennas and Wireless Propagation Letters, IEEE*, 7:469–472, 2008.

[27] Walter Gautschi. Algorithm 793: GQRAT - Gauss quadrature for rational functions. *ACM Trans. Math. Softw.*, 25(2):213–239, June 1999.

[28] C. Geuzaine and J.-F. Remacle. Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11):1309–1331, 2009.

[29] R.D. Graglia and G. Lombardi. Machine precision evaluation of singular and nearly singular potential integrals by use of Gauss quadrature formulas for rational functions. *Antennas and Propagation, IEEE Transactions on*, 56(4):981–998, April 2008.

[30] M. Guiggiani and A. Gigante. A general algorithm for multidimensional cauchy principal value integrals in the boundary element method. *Journal of Applied Mechanics, Transactions ASME*, 57(4):906–915, 1990.

[31] R.G. Hamlet. Testing programs with the aid of a compiler. *Software Engineering, IEEE Transactions on*, SE-3(4):279–290, July 1977.

[32] Roger F. Harrington. *Field Computation by Moment Methods (IEEE Press Series on Electromagnetic Waves)*. Wiley-IEEE Press, 4 1993. reprint of the original 1963 version.

[33] Roger F. Harrington. *Time-Harmonic Electromagnetic Fields (IEEE Press Series on Electromagnetic Wave Theory)*. Wiley-IEEE Press, 2nd edition, Aug. 2001. reprint of the original 1975 version.

[34] K. Hesse, I. H. Sloan, and R. S. Womersley. *Handbook of Geomathematics*. Springer-Verlag, Sept. 2010. Chapter *Numerical Integration on the Sphere*.

[35] R. Hiptmair and C. Schwab. Natural boundary element methods for the electric field integral equation on polyhedra. *SIAM Journal on Numerical Analysis*, 40(1):66–86, 2002.

[36] G.C. Hsiao and R.E. Kleinman. Mathematical foundations for error estimation in numerical solutions of integral equations in electromagnetics. *Antennas and Propagation, IEEE Transactions on*, 45(3):316–328, Mar 1997.

[37] IEEE. IEEE standard glossary of software engineering terminology. *IEEE Std 610.12-1990*, page 1, 1990.

[38] IEEE. IEEE standard for validation of computational electromagnetics computer modeling and simulations. *IEEE Std 1597.1-2008*, pages 1–41, 2008.

[39] IEEE. IEEE recommended practice for validation of computational electromagnetics computer modeling and simulations. *IEEE STD 1597.2-2010*, pages 1–124, 2011.

[40] Yue Jia and M. Harman. An analysis and survey of the development of mutation testing. *Software Engineering, IEEE Transactions on*, 37(5):649–678, Sept.-Oct. 2011.

[41] Jianming Jin. *The Finite Element Method in Electromagnetics*. John Wiley & Sons, 2nd edition, 2002.

[42] Cem Kaner, Jack Falk, and Hung Q. Nguyen. *Testing Computer Software*. Wiley, 2nd edition, April 1999.

[43] Oliver Dimon Kellogg. *Foundations Of Potential Theory*. Ungar, New York, 1929.

[44] M.A. Khayat and D.R. Wilton. Numerical evaluation of singular and near-singular potential integrals. *Antennas and Propagation, IEEE Transactions on*, 53(10):3180–3190, Oct. 2005.

[45] M.A. Khayat, D.R. Wilton, and P.W. Fink. An improved transformation and optimized sampling scheme for the numerical evaluation of singular and near-singular potentials. *Antennas and Wireless Propagation Letters, IEEE*, 7:377–380, 2008.

[46] Robert C. Kirby. Algorithm 839: FIAT, a new paradigm for computing finite element basis functions. *ACM Transactions on Mathematical Software*, 30(4):502–516, 2004.

[47] John C. Knight and Nancy G. Leveson. An experimental evaluation of the assumption of independence in multiversion programming. *Software Engineering, IEEE Transactions on*, SE-12(1):96–109, Jan. 1986.

[48] Jin-Fa Lee, Robert Lee, and Robert J. Burkholder. Loop star basis functions and a robust preconditioner for EFIE scattering problems. *Antennas and Propagation, IEEE Transactions on*, 51(8):1855–1863, August 2003.

[49] Anders Logg, Kent-Andre Mardal, Garth N. Wells, et al. *Automated Solution of Differential Equations by the Finite Element Method*. Springer, 2012.

[50] Anders Logg and Garth N. Wells. Dolfin: Automated finite element computing. *ACM Trans. Math. Softw.*, 37:20:1–20:28, April 2010.

[51] Daniel Jacobus Ludick. Efficient numerical analysis of focal plane antennas for the ska and the meerkat. Master's thesis, University of Stellenbosch, 2010.

[52] Enrico Magenes and J.-L. Lions. *Non-homogeneous boundary value problems and applications*. Springer-Verlag, 1972.

[53] R. Marchand and D.B. Davidson. The method of manufactured solutions for the verification of computational electromagnetics. In *Electromagnetics in Advanced Applications (ICEAA), 2011 International Conference on*, pages 487–490, September 2011.

[54] Peter Monk. *Finite Element Methods for Maxwell's Equations (Numerical Analysis and Scientific Computation Series)*. Oxford University Press, USA, June 2003.

[55] A.S. Namin, J. Andrews, and D. Murdoch. Sufficient mutation operators for measuring test effectiveness. In *Software Engineering, 2008. ICSE '08. ACM/IEEE 30th International Conference on*, pages 351–360, May 2008.

[56] J.C. Nédélec. Mixed finite elements in $\mathbb{R}^3$. *Numerische Mathematik*, 35(3):315–341, September 1980.

[57] J.C. Nédélec. A new family of mixed finite elements in ir3. *Numer. Math.*, 50(1):57–81, Nov 1986.

[58] William L Oberkampf and Timothy G Trucano. Verification and validation benchmarks. *Nuclear engineering and Design*, 238:716–743, 2008.

[59] W.L. Oberkampf and T.G. Trucano. Verification and validation in computational fluid dynamics. *Progress in Aerospace Sciences*, 38(3):209–272, 2002.

[60] American Institute of Aeronautics and Astronautics Staff. *AIAA Guide for the Verification and Validation of Computational Fluid Dynamics Simulations*. American Institute of Aeronautics & Astronautics, 1998.

[61] A. Offutt. The coupling effect: fact or fiction. *SIGSOFT Softw. Eng. Notes*, 14(8):131–140, November 1989.

[62] A. Jefferson Offutt. Investigations of the software testing coupling effect. *ACM Trans. Softw. Eng. Methodol.*, 1(1):5–20, January 1992.

[63] A.J. Otto, N. Marais, E. Lezar, and D.B. Davidson. Using the FEniCS package for FEM solutions in electromagnetics. *Antennas and Propagation Magazine, IEEE*, 54(4):206 –223, Aug. 2012.

[64] A.F. Peterson, S.L. Ray, and R Mittra. *Computational Methods for Electromagnetics*. Oxford University Press and IEEE Press, Oxford & New York, 1998.

[65] S. Rao, D. Wilton, and A. Glisson. Electromagnetic scattering by surfaces of arbitrary shape. *Antennas and Propagation, IEEE Transactions on*, 30(3):409–418, May 1982.

[66] P. Raviart and J. Thomas. A mixed finite element method for 2-nd order elliptic problems. In Ilio Galligani and Enrico Magenes, editors, *Mathematical Aspects of Finite Element Methods*, volume 606 of *Lecture Notes in Mathematics*, pages 292–315. Springer Berlin / Heidelberg, 1977.

[67] B. Daya Reddy. *Introductory Functional Analysis : With Applications to Boundary Value Problems and Finite Elements (Texts in Applied Mathematics, Vol. 27)*. Springer, Nov. 1997.

[68] J. N. Reddy. *Applied Functional Analysis and Variational Methods in Engineering*. Mcgraw-Hill College, 1 1986.

[69] Patrick J. Roache. *Fundamentals of Verification and Validation*. Hermosa Publishers, Sep. 2009.

[70] P.J. Roache. *Verification and Validation in Computational Science and Engineering*. Hermosa Publishers, 1998.

[71] P.J. Roache. Building PDE codes to be verifiable and validatable. *Computing in Science Engineering*, 6(5):30–38, Sept.-Oct. 2004.

[72] Kumbiz Salari and Patrick Knupp. Code Verification by the Method of Manufactured Solutions. Technical report, Sandia National Laboratories, 2000.

[73] Stefan A. Sauter and Christoph Schwab. *Boundary Element Methods*. Springer, 2011.

[74] T.M. Shih. Procedure to debug computer programs. *International Journal for Numerical Methods in Engineering*, 21(6):1027–1037, 1985.

[75] P.P. Silvester and R.L. Ferrari. *Finite elements for electrical engineers*. Cambridge University Press, 3rd edition, 1996.

[76] Glenn S. Smith. *An Introduction to Classical Electromagnetic Radiation*. Cambridge University Press, August 1997.

[77] Stanly Steinberg and Patrick J Roache. Symbolic manipulation and computational fluid dynamics. *Journal of Computational Physics*, 57(2):251–284, 1985.

[78] SymPy Developers. *SymPy: Python library for symbolic mathematics*, 2012.

[79] Mei Song Tong and Weng Cho Chew. A novel approach for evaluating hypersingular and strongly singular surface integrals in electromagnetics. *Antennas and Propagation, IEEE Transactions on*, 58(11):3593–3601, Nov. 2010.

[80] J. J. H. Wang. *Generalized Moment Methods in Electromagnetics*. John Wiley & Sons, Georgia Institue of Technology, Atlanta, Georgia, 1991.

[81] D.S. Weile and Xiaobo Wang. Strong singularity reduction for curved patches for the integral equations of electromagnetics. *Antennas and Wireless Propagation Letters, IEEE*, 8:1370–1373, 2009.