Unbounded PML based forward modeling of a 2D GPR tomography problem

A thesis submitted

in partial fulfillment of the requirements

for the degree of

Master of Technology

By
Gaurav Sunilrao Nandode

(14104040)

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

December 2016

Certificate

It is certified that the work contained in this thesis entitled "**Unbounded PML based forward modeling of a 2D GPR tomography problem**" by **Gaurav Sunilrao Nandode** (Roll No. 14104040) has been carried out under my supervision and that it has not been submitted elsewhere for a degree.

[illegible]

13 *December 2016*

Dr. Naren Naik

Department of Electrical Engineering

Indian Institute of Technology Kanpur

Date

Abstract

Ground penetrating radar (GPR) is a prominent modality used in major applications such as subsurface imaging, geophysical prospecting, non-destructive evaluation, to name a few. In order to reduce the probability of false alarms, we require quantitative information about the electrical parameters of the sub surface to be retrieved; this leads to GPR tomography.

The present work is about the optimization and implementation of a finite-element-method (FEM) based formulation for a Helmholtz-equation modeled forward problem of GPR tomography using an unbounded function as a perfectly matched layer (PML); these unbounded PMLs have been reported in the literature for acoustic problems. The obtained fields were verified by comparing with those obtained by solving the problem using COMSOL. Also, the Frechet-derivatives for the measured fields with respect to the electrical parameters have been derived using the method of adjoints; and these have been verified by comparison with the finite difference method.

Acknowledgements

I am very much thankful to God and my family for giving me such an opportunity in this world. I especially want to thank my mother who has been my friend, teacher and guide throughout my life. I wish to express my profound gratitude and heartiest thanks to my thesis supervisor Dr. Naren Naik for his excellent guidance and supervision for this work. His insightful knowledge of the subject and enthusiastic encouragement has helped me immensely throughout my research work. It has been my privilege to work with Ms. Nishigandha Patil (Ph.D. scholar). As a student she is a best mentor I have ever seen my whole life. Her guidance and knowledge has helped me every time I was not able to solve the problems. Her contribution to my work cannot be expressed in words. I would like to express my gratitude to Mr. Omprakash Gottam (Ph.D. scholar) whose support was immensely helpful in this research work and Mr. J Bharani Bharadwaj (M.Tech. student) who was always around me to help me with coding part. I feel very lucky to get the colleagues like them.

Contents

List of Figures

Dedicated

To

my family, my thesis supervisor and my friends

Chapter 1

Introduction

1.1 GPR Tomography Problem

The need to image the subsurface of the earth arises across diverse requirements ranging from geophysical prospecting to landmine detection. One commonly used modality for these investigation is the ground penetrating radar (GPR) which is an advanced imaging technique used to estimate the electrical parameters of the subsurface obstacle irrespective of whether it is metallic or dielectric. In a typical GPR tomography problem, there is an air-soil medium. An obstacle with a different electric parameter than the surrounding exists in the soil medium. A transmitter antenna or a source placed either touching the ground or at a small distance above the ground excites an electromagnetic wave. We measure the scattered fields from the underground obstacle at points in the plane parallel to soil surface containing the source. This is known as the "Forward Problem" or solutions obtained are called "Forward Solutions". We also measure the derivative of the scattered field with respect to the electrical parameter which is known as the "Frechet Derivative". Further in the "Inverse problem", an optimal value of a cost function is obtained by minimizing the difference between the simulated forward solutions and the experimental results and thereby the unknown electrical parameter of the obstacle is estimated.

1.2 Literature Survey

Accuracy and efficiency of forward models is very important in a GPR tomography problem. Several works have been reported in the literature for forward modeling of GPR[1][2].

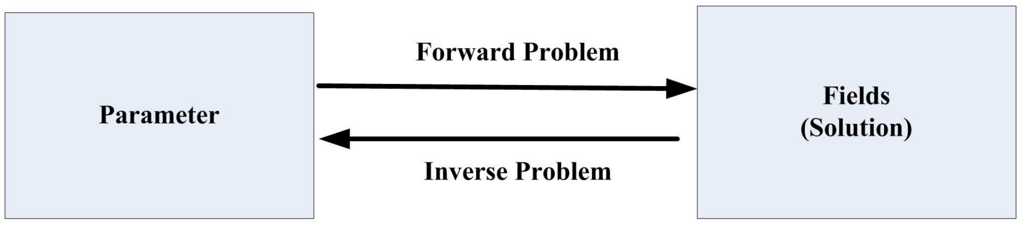


Figure 1.1: Typical GPR tomography problem

They mainly focus on GPR antenna design and use Finite Difference Time Domain (FDTD) method. FDTD has limitations to model irregular geometries and complex materials. In [3], semi-analytic mode matching (SAMM) technique is used for forward modeling. In [4] and [5], the forward problem is solved by the method of moments. In [6], time-domain adjoint method is used to calculate Jacobian matrix.

Truncating the computational domain to avoid the reflections from the boundary is a crucial part in any scattering problem. Absorbing boundary condition (ABC) and perfectly matched layer (PML) are the two methods to solve this problem. The implementation of the ABCs is very simple but the accuracy is poor. Higher order ABCs are required but those too come with computational cost and complex implementation.

The perfectly matched layer (PML) is a technique introduced by Berenger [7][8][9] in 1994. The absorbing boundary condition has a limitation of working at single frequency whereas PMLs work at different frequencies and at any angle of incidence. PML method incorporate an absorbing layer of anisotropic damping material around the physical domain. There are various challenges in PML design like choosing the PML thickness, choosing the absorption coefficient. Collino and Monk [10] have worked on the optimization problem for these challenges. In [11] pure imaginary stretching of co-ordinates is proposed for PML. Now complex co-ordinate stretching approach is widely used[12] for solving the acoustic scattering problem. We are using this approach to solve the electromagnetic problem.

In the present work, we use FEM to solve a GPR forward problem. Generally the forward solvers for such problems are slow as the calculations are done for number of source frequency configurations. So we try to optimize the forward solvers.

1.3 Complex Co-ordinate Stretching Approach

We consider the modified Maxwell's equations

$$\nabla_s \times \mathbf{E} = -j\omega\mu\mathbf{H} \tag{1.1}$$

$$\nabla_s \times \mathbf{H} = j\omega\epsilon\mathbf{E} + \mathbf{J} \tag{1.2}$$

$$\nabla_s.(\epsilon\mathbf{E}) = 0 \tag{1.3}$$

$$\nabla_s.(\mu\mathbf{H}) = 0 \tag{1.4}$$

where

$$\nabla_s = \hat{x}\frac{1}{\gamma_x}\frac{\partial}{\partial x} + \hat{y}\frac{1}{\gamma_y}\frac{\partial}{\partial y} + \hat{z}\frac{1}{\gamma_z}\frac{\partial}{\partial z}. \tag{1.5}$$

Here in ∇_s is a standard ∇ operator[20] in Cartesian coordinates whose x, y and z axes are stretched by some complex factors γ_x, γ_y and γ_z respectively. Hence it is called "Complex coordinate" stretching approach. Specifically $\gamma_x = \gamma_x(x)$,$\gamma_y = \gamma_y(y)$ and $\gamma_z = \gamma_z(z)$

So in place of derivatives we replace

$$\frac{\partial}{\partial x} \Rightarrow \frac{1}{\gamma_x(x)}\frac{\partial}{\partial x} \tag{1.6}$$

Also

$$x \Rightarrow \hat{x}(x) = \int_0^x \gamma_x(x^{'})dx^{'} \tag{1.7}$$

For example, for e^{jkx} wave

$$e^{jkx} \Rightarrow e^{jk\int_0^x \gamma_x(x^{'})dx^{'}} \tag{1.8}$$

We assume

$$\gamma_x(x) = 1 - \frac{j}{\omega}\sigma_x(|x|) \tag{1.9}$$

$$\gamma_y(y) = 1 - \frac{j}{\omega}\sigma_y(|y|) \tag{1.10}$$

$$\gamma_z(z) = 1 - \frac{j}{\omega}\sigma_z(|z|) \tag{1.11}$$

$\frac{1}{\omega}$ term takes care of the frequency independence of the PML. The entire process of PML can be conceptually summed up by a single transformation of our original equation:

$$\frac{\partial}{\partial x} \Rightarrow \frac{1}{1 - j\frac{\sigma_x(x)}{\omega}}\frac{\partial}{\partial x} \tag{1.12}$$

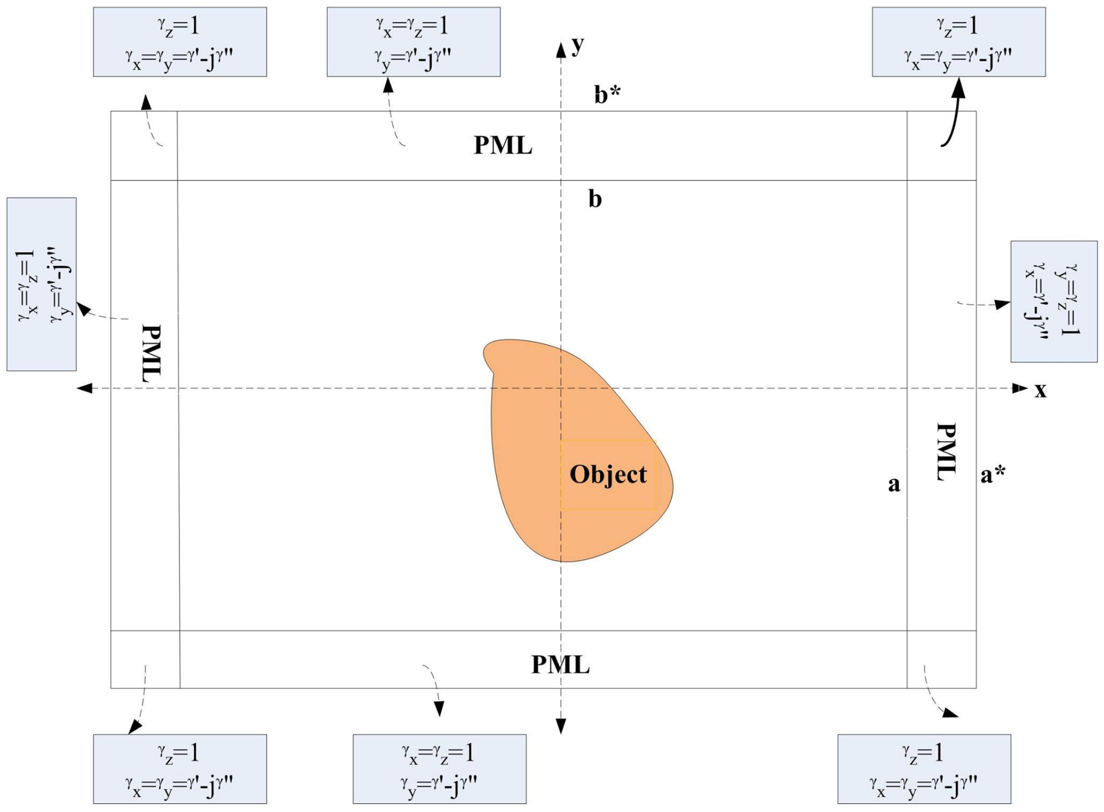


FIGURE 1.2: $\gamma^{'} = 1$ and $\gamma^{"} = \frac{j}{\omega}\sigma$

Also

$$x \Rightarrow \hat{x}(x) = \int_0^x \gamma_x(x^{'})dx^{'} = x - \frac{j}{\omega}\int_a^x \sigma_x(x^{'})dx^{'} \tag{1.13}$$

where a is the outermost x-coordinate of the physical region as PML starts from there. For example, for e^{jkx} wave

$$e^{jkx} \Rightarrow e^{jkx}e^{-\frac{k}{\omega}\int_a^x \sigma_x(x^{'})dx^{'}} \tag{1.14}$$

Here we can see the $\frac{k}{\omega}$ factor which is equal to $\frac{1}{c}$, inverse of the velocity. This shows that the attenuation in the PML is independent of the frequency.

1.4 Classical Absorbing Functions vs Unbounded Absorbing Functions for a PML

Absorbing functions σ's plays an important role in the perfectly matched layer : σ_x absorbs wave travelling in x-direction whereas σ_y absorbs that travelling in y-direction. On corners of the PML σ_x and σ_y act simultaneously. In classical PMLs, σ can be constant, linear or quadratic varying with x or y.

$$\sigma_x(x) = \sigma^*(1 - j) \tag{1.15}$$

$$\sigma_x(x) = \sigma^*(x - a) \tag{1.16}$$

$$\sigma_x(x) = \sigma^*(x - a)^2 \tag{1.17}$$

where σ^* is a parameter. As we go on increasing σ^* absorption rate increases but discretization errors too increase simultaneously. Though it has been reported in literature that quadratic or higher absorbing function perform better, choosing an optimal value of σ^* is a difficult task as it depends on the mesh size and thickness of the PML layer[10].

Bermudez[12] has used unbounded absorbing functions like

$$\sigma_x(x) = \frac{c}{(a^* - x)} \tag{1.18}$$

which do not have such disadvantages. He has solved an acoustic problem and shown the efficiency of the unbounded absorbing functions compared to the classical PML. Other than [12], [15] has compared the classical PMLs and unbounded PMLs. In [15] relative error as a function of PML thickness has been analyzed for various PML schemes and shown clearly that unbounded PML performs the best. Now this method is widely used in literature such as in aeroacoustics[21], for second order elliptic problems[23], for simulating the scattering of light in an open region[24], for solving a wave generalized propagation problem[22].

1.5 An Overview of Ground Penetrating Radar

Ground penetrating radar (GPR) uses microwaves to probe the subsurface of lossy dielectric materials. Two modes of measurement are common. In the first, reflected or scattered energy is detected. In the second, the effects on the energy transmitted through the material are observed[13]. Ground-penetrating radar uses a variety of technologies to generate the radar signal: these are impulse, stepped frequency and frequency-modulated continuous-wave (FMCW).

A GPR antenna is placed right above the air-ground interface. In humanitarian demining, the maximum depth to be surveyed is defined as 20 cm[13]; whereas we go deeper in soil exploration application to characterize the soil layers. Approximate penetration depth for 250 MHz antenna is 4m in wet soil and 12m in dry soil whereas for 500 MHz antenna it is 1.8m in wet soil and 4.4m in dry soil. For a 1GHz antenna the penetration depth is 1m

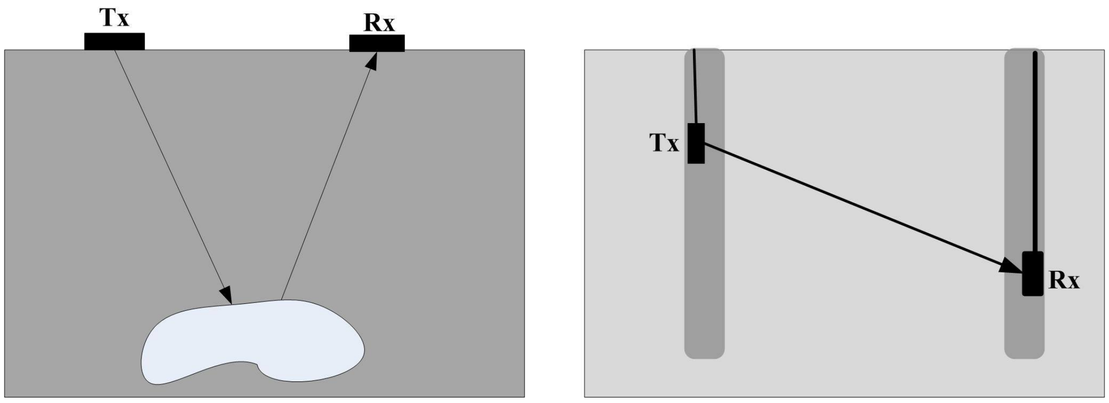


FIGURE 1.3: Types of GPR[13] In the first, reflected or scattered energy is detected. In the second, effects on energy transmitted through the material are observed.

to 2m and for a 2GHz antenna it is 15cm to 60cm. We present our models for frequency range of 250MHz to 850MHz.

1.6 Organization of Thesis

The outline of the thesis is as follows: **Chapter 2** explains the forward problem along with the implementation of the Perfectly Matched Layer required for truncating the computational domain and calculations associated with the PML. **Chapter 2** also describes verification with COMSOL. In **Chapter 3** Frechect derivative is computed using the adjoint method which is verified with finite difference method and optimization of the forward problem is explained. **Chapter 4** presents conclusion and future work.

Chapter 2

Forward Problem

2.1 Problem Setting

We choose an air-soil domain of size $3m \times 3m$, with the lower half describing a soil layer of dimensions $1.5m \times 3m$ and upper half as air with realtive permittivity 1 both surrounded by a PML of thickness 0.5m. So effectively the domain size is $4m \times 4m$. The relative dielectric permittivity of dry soil is around 2.5 with a loss tangent of 0.017. The relative dielectric permittivity of a plastic or wooden (when dry) casing varies from 2 to 4. The relative dielectric permittivity of explosive depends on its type and typically has a value around 3. We choose 3 as dielectric permittivity of scatterer and loss tangent as 0.005 considering it as plastic.

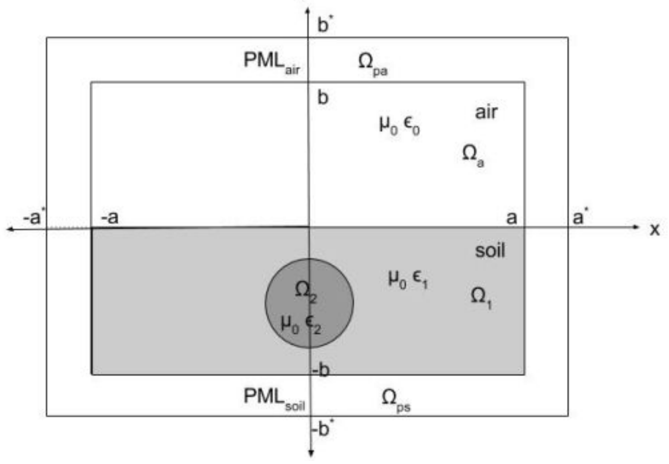


FIGURE 2.1: The domain

2.2 Governing Equation

Maxwell's equations for time harmonic fields assuming $e^{j\omega t}$ time dependence

$$\nabla \times \mathbf{E} = -j\omega\mathbf{B} \tag{2.1}$$

$$\nabla \times \mathbf{H} = j\omega\mathbf{D} + \mathbf{J} \tag{2.2}$$

$$\nabla.\mathbf{D} = \rho \tag{2.3}$$

$$\nabla.\mathbf{B} = 0 \tag{2.4}$$

Solving equations Eq. (2.1) and Eq. (2.2) gives vector wave equation as

$$\nabla \times (\frac{1}{\mu}\nabla \times \mathbf{E}) - \omega^2\epsilon\mathbf{E} = -j\omega\mathbf{J} \tag{2.5}$$

We apply Dirichlet boundary condition at the boundary for scattered fields.

2.3 Interpretation of PML as Anisotropic Absorber

The anisotropic absorber model of PML was first derived by Sacks et al.[16]. This can be done by first writing Maxwell's equations in an anisotropic medium as

$$\nabla \times \mathbf{E} = -j\omega\bar{\mu}\mathbf{H} \tag{2.6}$$

$$\nabla \times \mathbf{H} = j\omega\bar{\epsilon}\mathbf{E} \tag{2.7}$$

$$\nabla.(\bar{\epsilon}.\mathbf{E}) = 0 \tag{2.8}$$

$$\nabla.(\bar{\mu}.\mathbf{H}) = 0 \tag{2.9}$$

where $\bar{\epsilon}$ and $\bar{\mu}$ are the permittivity and permeability tensors give by

$$\bar{\epsilon} = \begin{bmatrix} \epsilon_{xx} & 0 & 0 \\ 0 & \epsilon_{yy} & 0 \\ 0 & 0 & \epsilon_{zz} \end{bmatrix} \tag{2.10}$$

$$\bar{\mu} = \begin{bmatrix} \mu_{xx} & 0 & 0 \\ 0 & \mu_{yy} & 0 \\ 0 & 0 & \mu_{zz} \end{bmatrix} \tag{2.11}$$

According to Sacks et al.[16], we may find $\bar{\epsilon}$ and $\bar{\mu}$ in terms of σ_x, σ_y and σ_z. So we choose

$$\bar{\epsilon} = \epsilon\bar{\Lambda}, \quad \bar{\mu} = \mu\bar{\Lambda} \tag{2.12}$$

with

$$\bar{\Lambda} = \hat{x}\hat{x}(\frac{\gamma_y\gamma_z}{\gamma_x}) + \hat{y}\hat{y}(\frac{\gamma_z\gamma_x}{\gamma_y}) + \hat{z}\hat{z}(\frac{\gamma_x\gamma_y}{\gamma_z}) \tag{2.13}$$

2.4 PML Calculations

The physical layer exists along x-direction between $(-a, a)$ and along y-direction between $(-b, b)$ whereas including the PML layer the dimensions are $(-a^*, a^*)$ and $(-b^*, b^*)$ The PML has been divided into 3 different types depending upon whether it absorbs waves travelling along x-direction, y-direction or towards corner as shown in Fig.2.3(a) .
We use a hybrid meshing scheme with triangular meshing for the physical medium and rectangular meshing for the PML regions.The reason why we use such hybrid meshes is that triangles are more adequate to fit the boundary of the scatterer, whereas rectangles will allow us to compute explicitly the integrals involving the absorbing function that will appear in the elements in the layer.

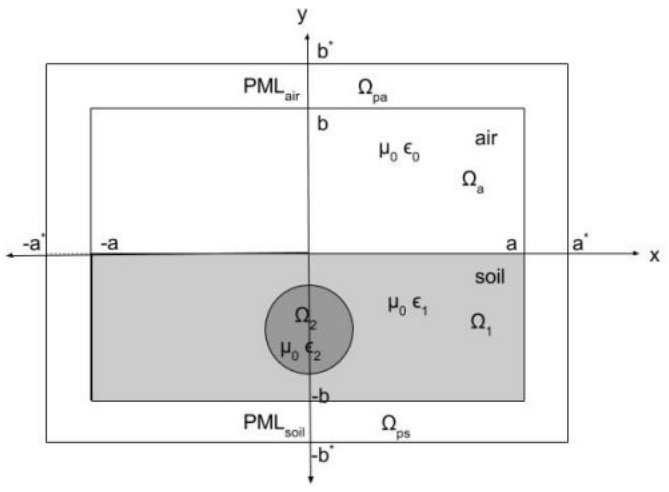
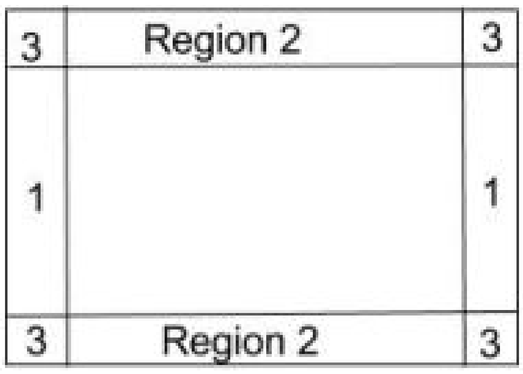
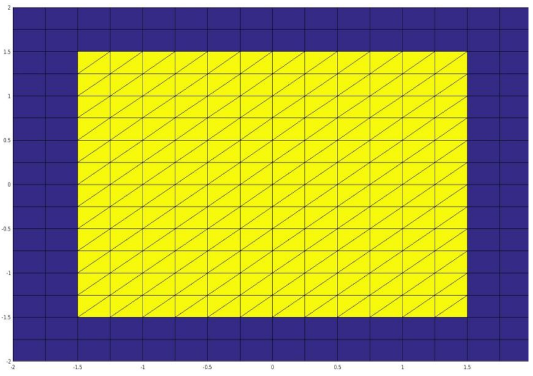


FIGURE 2.2: PML limits



(a) PML regions



(b) Meshing

FIGURE 2.3: Left : PML Regions, Right : Meshing

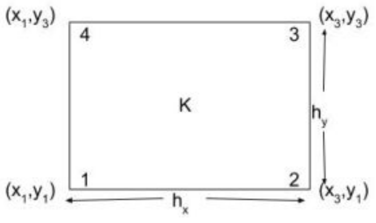


FIGURE 2.4: An rectangular element inside PML

The PML region 1 absorbs waves travelling along x-direction

$$\gamma_y(y) = 1 \tag{2.14}$$

$$\gamma_x(x) = \begin{cases} 1 & \text{if } |x| < a \\ 1 - \frac{j}{\omega}\sigma_x(|x|) & \text{if } a \leq |x| < a^* \end{cases} \tag{2.15}$$

For PML region 2

$$\gamma_x(x) = 1 \tag{2.16}$$

$$\gamma_y(y) = \begin{cases} 1 & \text{if } |y| < b \\ 1 - \frac{j}{\omega}\sigma_y(|y|) & \text{if } b \leq |y| < b^* \end{cases} \tag{2.17}$$

For region 3

$$\gamma_x(x) = \begin{cases} 1 & \text{if } |x| < a \\ 1 - \frac{j}{\omega}\sigma_x(|x|) & \text{if } a \le |x| < a^* \end{cases} \tag{2.18}$$

$$\gamma_y(y) = \begin{cases} 1 & \text{if } |y| < b \\ 1 - \frac{j}{\omega}\sigma_y(|y|) & \text{if } b \le |y| < b^* \end{cases} \tag{2.19}$$

$\gamma_x(x)$,$\gamma_y(y)$ and $\gamma_z(z)$ are unbounded absorbing functions. When we solve weak formulation for forward problem the following integrals involve the unbounded absorbing functions,

$$\int_K \frac{\gamma_y}{\gamma_x}\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x}dxdy, \qquad \int_K \frac{\gamma_x}{\gamma_y}\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dxdy \qquad \text{and} \qquad \int_K \gamma_x\gamma_y N_i N_j dxdy, \tag{2.20}$$

where n_i's and n_j's are the shape functions for the rectangular elements. As in Bermudez[12], we choose absorbing functions as for positive x and positive y

$$\sigma_x(x) = \frac{c}{a^* - x} \tag{2.21}$$

$$\sigma_y(y) = \frac{c}{b^* - y} \tag{2.22}$$

Shape functions for rectangular elements are chosen as

$$N_1 = \frac{(x - x_3)(y - y_3)}{h_x h_y} \qquad \frac{\partial N_1}{\partial x} = \frac{y - y_3}{h_x h_y} \qquad \frac{\partial N_1}{\partial y} = \frac{x - x_3}{h_x h_y} \tag{2.23}$$

$$N_2 = -\frac{(x - x_1)(y - y_3)}{h_x h_y} \qquad \frac{\partial N_2}{\partial x} = -\frac{y - y_3}{h_x h_y} \qquad \frac{\partial N_2}{\partial y} = -\frac{x - x_1}{h_x h_y} \tag{2.24}$$

$$N_3 = \frac{(x - x_1)(y - y_1)}{h_x h_y} \qquad \frac{\partial N_3}{\partial x} = \frac{y - y_1}{h_x h_y} \qquad \frac{\partial N_3}{\partial y} = -\frac{x - x_1}{h_x h_y} \tag{2.25}$$

$$N_4 = -\frac{(x - x_3)(y - y_1)}{h_x h_y} \qquad \frac{\partial N_4}{\partial x} = -\frac{y - y_1}{h_x h_y} \qquad \frac{\partial N_4}{\partial y} = -\frac{x - x_3}{h_x h_y} \tag{2.26}$$

To calculate integrals in (2.20), we first need to calculate some specific forms of integrals

$$s_{11} = \int_{x_1}^{x_3} \frac{(x - x_1)^2\, dx}{(a^* - x)} = h_x(\frac{3x_1}{2} - \frac{x_3}{2} - a^*) + (a^* - x_1)^2 \log(\frac{a^* - x_1}{a^* - x_3}) \tag{2.27}$$

$$s_{33} = \int_{x_1}^{x_3} \frac{(x - x_3)^2\, dx}{(a^* - x)} = h_x(\frac{3x_3}{2} - \frac{x_1}{2} - a^*) + (a^* - x_3)^2 \log(\frac{a^* - x_1}{a^* - x_3}) \tag{2.28}$$

$$s_{13} = \int_{x_1}^{x_3} \frac{(x-x_1)(x-x_3)\,dx}{(a^*-x)} = \frac{h_x(x_1+x_3-2a*)}{2} + (a^*-x_1)(a*-x_3)\log(\frac{a^*-x_1}{a^*-x_3}) \tag{2.29}$$

$$p_{11} = \int_{x_1}^{x_3} (x-x_1)^2\,dx = \frac{{h_x}^3}{3} \tag{2.30}$$

$$p_{33} = \int_{x_1}^{x_3} (x-x_3)^2\,dx = \frac{{h_x}^3}{3} \tag{2.31}$$

$$p_{13} = \int_{x_1}^{x_3} (x-x_1)(x-x_3)\,dx = -\frac{{h_x}^3}{6} \tag{2.32}$$

$$q_{11} = \int_{y_1}^{y_3} (y-y_1)^2\,dy = \frac{{h_y}^3}{3} \tag{2.33}$$

$$q_{33} = \int_{y_1}^{y_3} (y-y_3)^2\,dy = \frac{{h_y}^3}{3} \tag{2.34}$$

$$q_{13} = \int_{y_1}^{y_3} (y-y_1)(y-y_3)\,dy = -\frac{{h_y}^3}{6} \tag{2.35}$$

$$t_{33} = \int_{y_1}^{y_3} \frac{(y-y_3)^2\,dy}{(b^*-y)} = h_y(\frac{3y_3}{2} - \frac{y_1}{2} - b^*) + (b^*-y_3)^2\log(\frac{b^*-y_1}{b^*-y_3}) \tag{2.36}$$

$$t_{11} = \int_{y_1}^{y_3} \frac{(y-y_1)^2\,dy}{(b^*-y)} = h_y(\frac{3y_1}{2} - \frac{y_3}{2} - b^*) + (b^*-y_1)^2\log(\frac{b^*-y_1}{b^*-y_3}) \tag{2.37}$$

$$t_{13} = \int_{y_1}^{y_3} \frac{(y-y_1)(y-y_3)\,dy}{(b^*-y)} = \frac{h_y(y_1+y_3-2b*)}{2} + (b^*-y_1)(b*-y_3)\log(\frac{b^*-y_1}{b^*-y_3}) \tag{2.38}$$

$$m_{2x} = \int_{x_1}^{x_3} \frac{dx}{\gamma_x} = h_x + \frac{jc}{\omega} log(\frac{\omega(a^*-x_3)+jc}{\omega(a^*-x_1)+jc}) \tag{2.39}$$

$$m_{2y} = \int_{y_1}^{y_3} \frac{dy}{\gamma_y} = h_y + \frac{jc}{\omega} log(\frac{\omega(b^*-y_3)+jc}{\omega(b^*-y_1)+jc}) \tag{2.40}$$

Absorbing functions for negative x and negative y are chosen as

$$\sigma_x(x) = \frac{c}{a^*+x} \tag{2.41}$$

$$\sigma_y(y) = \frac{c}{b^*+y} \tag{2.42}$$

In such a case, following integrals change

$$t'_{11} = \int_{y_1}^{y_3} \frac{(y-y_1)^2\,dy}{(b^*+y)} = h_y(\frac{y_3}{2} - b^* - \frac{3y_1}{2}) + (b^*+y_1)^2\log(\frac{b^*+y_3}{b^*+y_1}) \tag{2.43}$$

$$t'_{33} = \int_{y_1}^{y_3} \frac{(y-y_3)^2\,dy}{(b^*+y)} = h_y(\frac{y_1}{2} - b^* - \frac{3y_3}{2}) + (b^*+y_3)^2\log(\frac{b^*+y_3}{b^*+y_1}) \tag{2.44}$$

$$t'_{13} = \int_{y_1}^{y_3} \frac{(y-y_1)(y-y_3)\,dy}{(b^*+y)} = -\frac{h_y(y_1+y_3+2b^*)}{2} + (b^*+y_1)(b^*+y_3)\log(\frac{b^*+y_3}{b^*+y_1}) \tag{2.45}$$

$$s'_{11} = \int_{x_1}^{x_3} \frac{(x-x_1)^2\,dx}{(a^*+x)} = h_x(\frac{x_3}{2} - \frac{3x_1}{2} - a^*) + (a^*+x_1)^2 \log(\frac{a^*+x_1}{a^*+x_3}) \tag{2.46}$$

$$s'_{33} = \int_{x_1}^{x_3} \frac{(x-x_3)^2\,dx}{(a^*+x)} = h_x(\frac{x_1}{2} - \frac{3x_3}{2} - a^*) + (a^*+x_3)^2 \log(\frac{a^*+x_3}{a^*+x_1}) \tag{2.47}$$

$$s'_{13} = \int_{x_1}^{x_3} \frac{(x-x_1)(x-x_3)\,dx}{(a^*+x)} = -\frac{h_x(x_1+x_3+2a*)}{2} + (a^*+x_1)(a*+x_3)\log(\frac{a^*+x_3}{a^*+x_1}) \tag{2.48}$$

$$m'_{2x} = \int_{x_1}^{x_3} \frac{dx}{\gamma_x} = h_x - \frac{jc}{\omega} log(\frac{\omega(a^*+x_3)+jc}{\omega(a^*+x_1)+jc}) \tag{2.49}$$

$$m'_{2y} = \int_{y_1}^{y_3} \frac{dy}{\gamma_y} = h_y - \frac{jc}{\omega} log(\frac{\omega(b^*+y_3)+jc}{\omega(b^*+y_1)+jc}) \tag{2.50}$$

We can solve integrals in (2.20) now

$$\int_K \frac{\gamma_y}{\gamma_x}\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} dxdy = \int_{x_1}^{x_3} \frac{dx}{\gamma_x} \int_{y_1}^{y_3} \gamma_y \frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} dy \tag{2.51}$$

In PML region 1 this evaluates to

$$\int_K \frac{\gamma_y}{\gamma_x}\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} dxdy = m_{2x} \times \frac{1}{h_x^2 h_y^2} \times \begin{bmatrix} q_{33} & -q_{33} & q_{13} & -q_{13} \\ -q_{33} & q_{33} & -q_{13} & q_{13} \\ q_{13} & -q_{13} & q_{11} & -q_{11} \\ -q_{13} & q_{13} & -q_{11} & q_{11} \end{bmatrix} \tag{2.52}$$

In PML region 2 this evaluates to

$$\int_K \frac{\gamma_y}{\gamma_x}\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} dxdy = h_x \times \frac{1}{h_x^2 h_y^2} \times \begin{bmatrix} q_{33}+\frac{jc}{\omega}t_{33} & -q_{33}-\frac{jc}{\omega}t_{33} & q_{13}+\frac{jc}{\omega}t_{13} & -q_{13}-\frac{jc}{\omega}t_{13} \\ -q_{33}-\frac{jc}{\omega}t_{33} & q_{33}+\frac{jc}{\omega}t_{33} & -q_{13}-\frac{jc}{\omega}t_{13} & q_{13}+\frac{jc}{\omega}t_{13} \\ q_{13}+\frac{jc}{\omega}t_{13} & -q_{13}-\frac{jc}{\omega}t_{13} & q_{11}+\frac{jc}{\omega}t_{11} & -q_{11}-\frac{jc}{\omega}t_{11} \\ -q_{13}-\frac{jc}{\omega}t_{13} & q_{13}+\frac{jc}{\omega}t_{13} & -q_{11}-\frac{jc}{\omega}t_{11} & q_{11}+\frac{jc}{\omega}t_{11} \end{bmatrix} \tag{2.53}$$

In PML region 3 this evaluates to

$$\int_K \frac{\gamma_y}{\gamma_x}\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} dxdy = m_{2x} \times \frac{1}{h_x^2 h_y^2} \times \begin{bmatrix} q_{33}+\frac{jc}{\omega}t_{33} & -q_{33}-\frac{jc}{\omega}t_{33} & q_{13}+\frac{jc}{\omega}t_{13} & -q_{13}-\frac{jc}{\omega}t_{13} \\ -q_{33}-\frac{jc}{\omega}t_{33} & q_{33}+\frac{jc}{\omega}t_{33} & -q_{13}-\frac{jc}{\omega}t_{13} & q_{13}+\frac{jc}{\omega}t_{13} \\ q_{13}+\frac{jc}{\omega}t_{13} & -q_{13}-\frac{jc}{\omega}t_{13} & q_{11}+\frac{jc}{\omega}t_{11} & -q_{11}-\frac{jc}{\omega}t_{11} \\ -q_{13}-\frac{jc}{\omega}t_{13} & q_{13}+\frac{jc}{\omega}t_{13} & -q_{11}-\frac{jc}{\omega}t_{11} & q_{11}+\frac{jc}{\omega}t_{11} \end{bmatrix} \tag{2.54}$$

The second integral in (2.20) is

$$\int_K \frac{\gamma_x}{\gamma_y}\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dxdy = \int_{y1}^{y3}\frac{dy}{\gamma_y}\int_{x1}^{x3}\gamma_x\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dx \tag{2.55}$$

In PML region 1 this evaluates to

$$\int_K \frac{\gamma_x}{\gamma_y}\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dxdy = h_y\times\frac{1}{h_x^2h_y^2}\times\begin{bmatrix} p_{33}+\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ -p_{13}-\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ p_{13}+\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ -p_{33}-\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \end{bmatrix} \tag{2.56}$$

In PML region 2 this evaluates to

$$\int_K \frac{\gamma_x}{\gamma_y}\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dxdy = m_{2y}\times\frac{1}{h_x^2h_y^2}\times\begin{bmatrix} p_{33} & -p_{13} & p_{13} & -p_{33} \\ -p_{13} & -p_{13} & p_{13} & -p_{33} \\ p_{13} & -p_{13} & p_{13} & -p_{33} \\ -p_{33} & -p_{13} & p_{13} & -p_{33} \end{bmatrix} \tag{2.57}$$

In PML regin 3 this evaluates to

$$\int_K \frac{\gamma_x}{\gamma_y}\frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}dxdy = m_{2y}\times\frac{1}{h_x^2h_y^2}\times\begin{bmatrix} p_{33}+\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ -p_{13}-\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ p_{13}+\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \\ -p_{33}-\frac{jc}{\omega}s_{33} & -p_{13}-\frac{jc}{\omega}s_{13} & p_{13}+\frac{jc}{\omega}s_{13} & -p_{33}-\frac{jc}{\omega}s_{33} \end{bmatrix} \tag{2.58}$$

Now let us solve the third integral (2.20)

$$\int_K \gamma_x\gamma_y N_iN_j dxdy \tag{2.59}$$

In PML region 1 this evaluates to

$$\int_K \gamma_x\gamma_y N_iN_j dxdy = \frac{1}{h_x^2h_y^2}([p_{ij}q_{ij}] + \frac{jc}{\omega}[s_{ij}q_{ij}]) \tag{2.60}$$

In PML region 2 this evaluates to

$$\int_K \gamma_x\gamma_y N_iN_j dxdy = \frac{1}{h_x^2h_y^2}([p_{ij}q_{ij}] + \frac{jc}{\omega}[p_{ij}t_{ij}]) \tag{2.61}$$

In PML region 3 this evaluates to

$$\int_K \gamma_x \gamma_y N_i N_j dxdy = \frac{1}{h_x^2 h_y^2}([p_{ij}q_{ij}] + \frac{jc}{\omega}[s_{ij}q_{ij}] + \frac{jc}{\omega}[p_{ij}t_{ij}] - \frac{c^2}{\omega^2}[s_{ij}t_{ij}]) \tag{2.62}$$

where

$$[p_{ij}q_{ij}] = \begin{bmatrix} p_{33}q_{33} & -p_{13}q_{33} & p_{13}q_{13} & -p_{33}q_{13} \\ -p_{13}q_{33} & p_{11}q_{33} & -p_{11}q_{13} & p_{13}q_{13} \\ p_{13}q_{13} & -p_{11}q_{13} & p_{11}q_{11} & -p_{13}q_{11} \\ -p_{33}q_{13} & p_{13}q_{13} & -p_{13}q_{11} & p_{33}q_{11} \end{bmatrix} \tag{2.63}$$

$$[s_{ij}q_{ij}] = \begin{bmatrix} s_{33}q_{33} & -s_{13}q_{33} & s_{13}q_{13} & -s_{33}q_{13} \\ -s_{13}q_{33} & s_{11}q_{33} & -s_{11}q_{13} & s_{13}q_{13} \\ s_{13}q_{13} & -s_{11}q_{13} & s_{11}q_{11} & -s_{13}q_{11} \\ -s_{33}q_{13} & s_{13}q_{13} & -s_{13}q_{11} & s_{33}q_{11} \end{bmatrix} \tag{2.64}$$

$$[p_{ij}t_{ij}] = \begin{bmatrix} p_{33}t_{33} & -p_{13}t_{33} & p_{13}t_{13} & -p_{33}t_{13} \\ -p_{13}t_{33} & p_{11}t_{33} & -p_{11}t_{13} & p_{13}t_{13} \\ p_{13}t_{13} & -p_{11}t_{13} & p_{11}t_{11} & -p_{13}t_{11} \\ -p_{33}t_{13} & p_{13}t_{13} & -p_{13}t_{11} & p_{33}t_{11} \end{bmatrix} \tag{2.65}$$

$$[s_{ij}t_{ij}] = \begin{bmatrix} s_{33}t_{33} & -s_{13}t_{33} & s_{13}t_{13} & -s_{33}t_{13} \\ -s_{13}t_{33} & s_{11}t_{33} & -s_{11}t_{13} & s_{13}t_{13} \\ s_{13}t_{13} & -s_{11}t_{13} & s_{11}t_{11} & -s_{13}t_{11} \\ -s_{33}t_{13} & s_{13}t_{13} & -s_{13}t_{11} & s_{33}t_{11} \end{bmatrix} \tag{2.66}$$

For the waves travelling in negative x just replace s values by $s^{'}$ and m_{2x} by $m_{2x}^{'}$. Similarly for the waves travelling in negative y replace t values by $t^{'}$ and m_{2y} by $m_{2y}^{'}$.

While writing codes, separate functions have been written for different PML types. Again in each function separate functions have been written for positive x or y and negative x or y.

We verified these calculations using MuPAD in matlab. These calculations are used when we solve the forward problem in the PML domain.

Now with the PML calculations we proceed for the forward model. As we are interested in a 2D problem, we use the scalar approximation of the vector wave equation. We get the incident field from the homogeneous medium. We use this incident field as a source in in-homogeneous medium and solve for the scattered field.

2.5 Scalar Approximation

$$\nabla \times (\bar{\mu}^{-1}.\nabla \times \mathbf{E}) - \omega^2 \bar{\epsilon}.\mathbf{E} = -j\omega \mathbf{J} \tag{2.67}$$

The PML scheme is based on using a layer of diagonally anisotropic material to absorb outgoing waves from the computational domain. As discussed in the previous sections, for PML, we choose $\bar{\epsilon}$ and $\bar{\mu}$ such that

$$\bar{\epsilon} = \epsilon \bar{\Lambda}, \quad \bar{\mu} = \mu \bar{\Lambda} \tag{2.68}$$

where

$$\bar{\Lambda} = \begin{bmatrix} \frac{\gamma_y \gamma_z}{\gamma_x} & 0 & 0 \\ 0 & \frac{\gamma_z \gamma_x}{\gamma_y} & 0 \\ 0 & 0 & \frac{\gamma_x \gamma_y}{\gamma_z} \end{bmatrix} \tag{2.69}$$

and ϵ and μ are taken corresponding to medium through which we want to absorb the outgoing waves. For two-dimensional problems assuming E_z-polarization and no variation along the z-axis, the governing equation is

$$\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}} \frac{\partial E_z}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}} \frac{\partial E_z}{\partial y}) + \omega^2 \epsilon_{zz} E_z = j\omega J_z{}^1 \tag{2.70}$$

The Incident field in a homogeneous medium is given by

$$\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}} \frac{\partial E_z^{inc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}} \frac{\partial E_z^{inc}}{\partial y}) + \omega^2 \epsilon_{zz} E_z^{inc} = j\omega J_z \tag{2.71}$$

The Total field with inhomogeneities is thus

$$E_z = E_z^{inc} + E_z^{sc} \tag{2.72}$$

Hence, from the governing equation we can write

$$\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}} \frac{\partial (E_z^{inc} + E_z^{sc})}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}} \frac{\partial (E_z^{inc} + E_z^{sc})}{\partial y}) + \omega^2 (\epsilon_{zz} + \Delta\epsilon)(E_z^{inc} + E_z^{sc}) = j\omega J_z \tag{2.73}$$

where $\Delta\epsilon$ is the difference between permittivities of the scatterer and the background soil. Solving Eq.(2.71) and Eq.(2.73) we get

$$\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}} \frac{\partial E_z^{sc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}} \frac{\partial E_z^{sc}}{\partial y}) + \omega^2 (\epsilon_{zz} + \Delta\epsilon) E_z^{sc} = -\omega^2 \Delta\epsilon E_z^{inc} \tag{2.74}$$

[1] It is given as $-j\omega J$ in Jin[17].That is a mistake

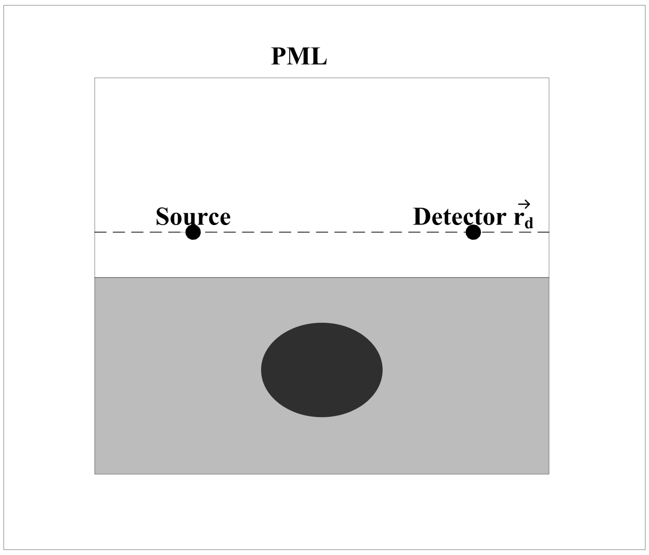


FIGURE 2.5: 2D view of a GPR set up

2.6 Weak Formulation

The weighted residual form using the Galerkin Weak Formulation with test function q is

$$\int_\Omega q[\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}) + \omega^2(\epsilon_{zz} + \Delta\epsilon)E_z^{sc} + \omega^2\Delta\epsilon E_z^{inc}] = 0 \tag{2.75}$$

where,

$$\int_\Omega q\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x})d\Omega = \int_{\Gamma_x} q(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x})\hat{x}d\Gamma - \int_\Omega(\frac{1}{\mu_{yy}}\frac{\partial q}{\partial x}\frac{\partial E_z^{sc}}{\partial x})d\Omega \tag{2.76}$$

$$\int_\Omega q\frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y})d\Omega = \int_{\Gamma_y} q(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y})\hat{y}d\Gamma - \int_\Omega(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y})d\Omega \tag{2.77}$$

Γ is the outermost boundary. Substituting Eq. (2.76) and Eq. (2.77) in Eq. (2.75)

$$\int_\Gamma q(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}\hat{x} + \frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}\hat{y})d\Gamma - \int_\Omega \frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_\Omega \frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_\Omega \omega^2 q(\epsilon_{zz} + \Delta\epsilon)E_z^{sc}d\Omega + \int_\Omega \omega^2\Delta\epsilon q E_z^{inc}d\Omega = 0 \tag{2.78}$$

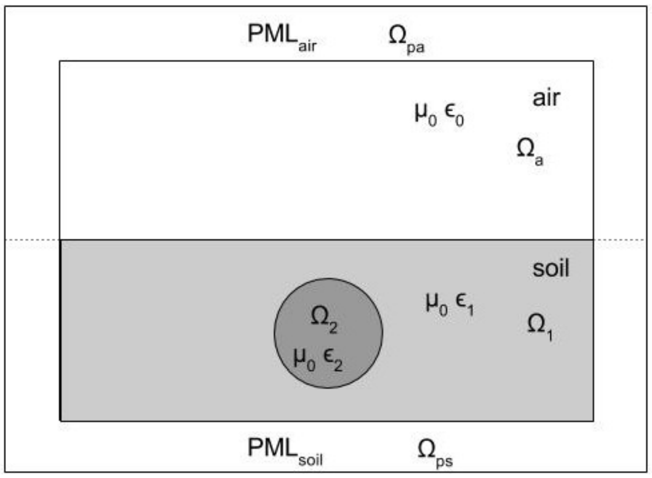


FIGURE 2.6: Domain with parameters

Applying Dirichlet boundary condition $E_{sc} = 0$ on the outermost boundary Γ

$$\int_\Omega \frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega + \int_\Omega \frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega - \int_\Omega \omega^2 q(\epsilon_{zz}+\Delta\epsilon)E_z^{sc}d\Omega = \int_\Omega \omega^2\Delta\epsilon q E_z^{inc}d\Omega \tag{2.79}$$

So first we need to find E_z^{inc} from Eq. (2.71) i.e. incident field in homogeneous or background medium. We obtain the Galerkin weak form as on Eq. (2.71)

$$\int_\Omega q[\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial E_z^{inc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial E_z^{inc}}{\partial y}) + \omega^2\epsilon_{zz}E_z^{inc}]d\Omega = \int_\Omega qj\omega J_z d\Omega \tag{2.80}$$

$$\int_\Gamma q(\frac{1}{\mu_{yy}}\frac{\partial E_z^{inc}}{\partial x}\hat{x} + \frac{1}{\mu_{xx}}\frac{\partial E_z^{inc}}{\partial y}\hat{y})d\Gamma - \int_\Omega \frac{1}{\mu_{yy}}\frac{\partial E_z^{inc}}{\partial x}\frac{\partial q}{\partial x}d\Omega - \int_\Omega \frac{1}{\mu_{xx}}\frac{\partial E_z^{inc}}{\partial y}\frac{\partial q}{\partial y}d\Omega + \int_\Omega \omega^2\epsilon_{zz}qE_z^{inc}d\Omega = \int_\Omega qj\omega J_z d\Omega \tag{2.81}$$

Applying Dirichlet boundary condition on the outermost boundary Γ and considering a delta point source, we get

$$\int_\Omega \frac{1}{\mu_{yy}}\frac{\partial E_z^{inc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_\Omega \frac{1}{\mu_{xx}}\frac{\partial E_z^{inc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_\Omega \omega^2\epsilon_{zz}qE_z^{inc}d\Omega = -j\omega q(x_0,y_0) \tag{2.82}$$

This E_z^{inc} value can be used in Eq. (2.79) to get E_z^{sc}.

As shown on the fig. 2.6, Ω_a, Ω_1 and Ω_2 are the regions representing air, soil and scatterer mediums. ϵ_0 is the absolute permittivity in air medium, ϵ_1 and ϵ_2 are the relative permittivities in soil and scatterer regions. μ_0 is same through out the domain. PML is

divided in two regions Ω_{Pa} and Ω_{Ps} as the parameters in the PML are defined as :
PML surrounding air has parameters

$$\bar{\epsilon} = \epsilon_0[\Lambda] \tag{2.83}$$

$$\bar{\mu} = \mu_0[\Lambda] \tag{2.84}$$

whereas PML surrounding soil has parameters

$$\bar{\epsilon} = \epsilon_0\epsilon_1[\Lambda] \tag{2.85}$$

$$\bar{\mu} = \mu_0[\Lambda] \tag{2.86}$$

$$\bar{\Lambda} = \begin{bmatrix} \frac{\gamma_y\gamma_z}{\gamma_x} & 0 & 0 \\ 0 & \frac{\gamma_z\gamma_x}{\gamma_y} & 0 \\ 0 & 0 & \frac{\gamma_x\gamma_y}{\gamma_z} \end{bmatrix} \tag{2.87}$$

So for Eq. (2.79)

$$\int_\Omega \frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_\Omega \frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_\Omega \omega^2 q(\epsilon_{zz} + \Delta\epsilon)E_z^{sc}d\Omega = \int_\Omega \omega^2\Delta\epsilon q E_z^{inc}d\Omega \tag{2.88}$$

we have in Ω_{Pa} i.e. PML surrounding air

$$\epsilon_{zz} = \epsilon_0\frac{\gamma_x\gamma_y}{\gamma_z} \tag{2.89}$$

$$\mu_{xx} = \mu_0\frac{\gamma_y\gamma_z}{\gamma_x} \tag{2.90}$$

$$\mu_{yy} = \mu_0\frac{\gamma_z\gamma_x}{\gamma_y} \tag{2.91}$$

In Ω_{Ps} i.e. PML surrounding soil

$$\epsilon_{zz} = \epsilon_0\epsilon_1\frac{\gamma_x\gamma_y}{\gamma_z} \tag{2.92}$$

$$\mu_{xx} = \mu_0\frac{\gamma_y\gamma_z}{\gamma_x} \tag{2.93}$$

$$\mu_{yy} = \mu_0\frac{\gamma_z\gamma_x}{\gamma_y} \tag{2.94}$$

ϵ_1 and ϵ_2 are considered with losses

$$\epsilon_1 = \epsilon_{r1}(1 - jtan\delta_1) \tag{2.95}$$

$$\epsilon_2 = \epsilon_{r2}(1 - jtan\delta_2) \tag{2.96}$$

Separating each integral into domains of Eq. (2.88)

$$\begin{aligned}
&\int_{\Omega_a} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_a} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_{\Omega_a} \omega^2 q\epsilon_0 E_z^{sc}d\Omega + \int_{\Omega_1} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega \\
&+ \int_{\Omega_1} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_{\Omega_1} \omega^2 q\epsilon_0\epsilon_1 E_z^{sc}d\Omega + \int_{\Omega_2} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_2} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega \\
&- \int_{\Omega_2} \omega^2 q\epsilon_0\epsilon_2 E_z^{sc}d\Omega + \int_{\Omega_{Pa}} \frac{\gamma_y}{\mu_0\gamma_z\gamma_x}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_{Pa}} \frac{\gamma_x}{\mu_0\gamma_y\gamma_z}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega \\
&- \int_{\Omega_{Pa}} \omega^2 q\epsilon_0\frac{\gamma_x\gamma_y}{\gamma_z}d\Omega + \int_{\Omega_{Ps}} \frac{\gamma_y}{\mu_0\gamma_z\gamma_x}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_{Ps}} \frac{\gamma_x}{\mu_0\gamma_y\gamma_z}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega E_z^{sc}d\Omega \\
&- \int_{\Omega_{Ps}} \omega^2 q\epsilon_0\epsilon_1\frac{\gamma_x\gamma_y}{\gamma_z}d\Omega = \int_{\Omega_2} \epsilon_0(\epsilon_2 - \epsilon_1)qE_z^{inc}d\Omega
\end{aligned} \tag{2.97}$$

γ-values for PML are calculated in Appendix. So finally Eq. (2.97) can be written as

$$\begin{aligned}
&\int_{\Omega_a+\Omega_1+\Omega_2} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_a+\Omega_1+\Omega_2} \frac{1}{\mu_0}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega - \int_{\Omega_a} \omega^2 q\epsilon_0 E_z^{sc}d\Omega - \int_{\Omega_1} \omega^2 q\epsilon_0\epsilon_1 E_z^{sc}d\Omega \\
&- \int_{\Omega_2} \omega^2 q\epsilon_0\epsilon_2 E_z^{sc}d\Omega + \int_{\Omega_{Pa}+\Omega_{Ps}} \frac{\gamma_y}{\mu_0\gamma_z\gamma_x}\frac{\partial E_z^{sc}}{\partial x}\frac{\partial q}{\partial x}d\Omega + \int_{\Omega_{Pa}+\Omega_{Ps}} \frac{\gamma_x}{\mu_0\gamma_y\gamma_z}\frac{\partial E_z^{sc}}{\partial y}\frac{\partial q}{\partial y}d\Omega E_z^{sc}d\Omega \\
&- \int_{\Omega_{Pa}} \omega^2 q\epsilon_0\frac{\gamma_x\gamma_y}{\gamma_z}d\Omega - \int_{\Omega_{Ps}} \omega^2 q\epsilon_0\epsilon_1\frac{\gamma_x\gamma_y}{\gamma_z}d\Omega = \int_{\Omega_2} \epsilon_0(\epsilon_2 - \epsilon_1)qE_z^{inc}d\Omega
\end{aligned} \tag{2.98}$$

Similarly incorporating PML parameters in Eq. (2.82), E_z^{inc} can be calculated with ϵ for homogeneous medium.

2.7 Numerical Results

The boundary between obstacle and soil is separated by comparing the distances of centroids of elemental triangles with radius of the scatterer. Maximum wavelength in a medium with complex permittivity is approximately given by $\lambda = \frac{c}{f\sqrt{\epsilon'_{max}}}$. At 250 MHz, its value is $\frac{3\times10^8}{250\times10^6\sqrt{3}} = 0.6928m = 69.28cm$. Hence, the maximum mesh size can be $\lambda/10 = 6.928cm$. We have taken number of nodes along x and y direction as 161. Hence, mesh size is $dx = dy = \frac{400}{160} = 2.5cm$.

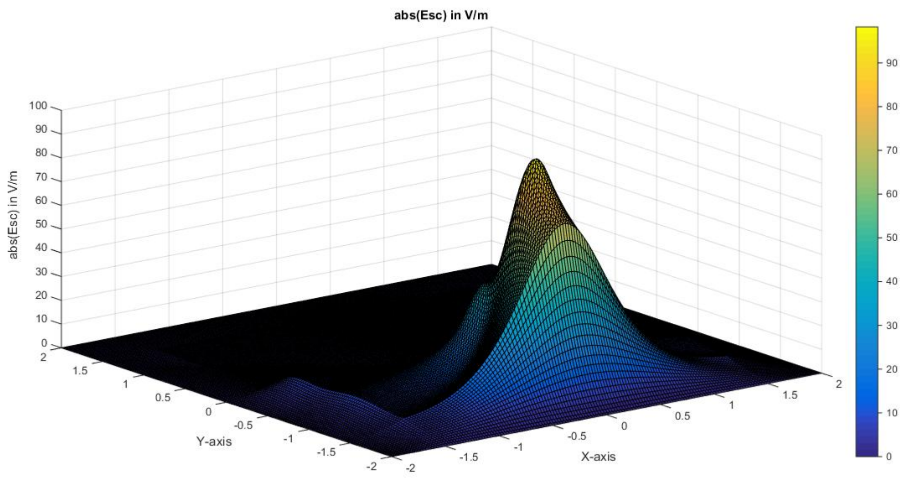


Figure 2.7: abs(Esc) 3D view

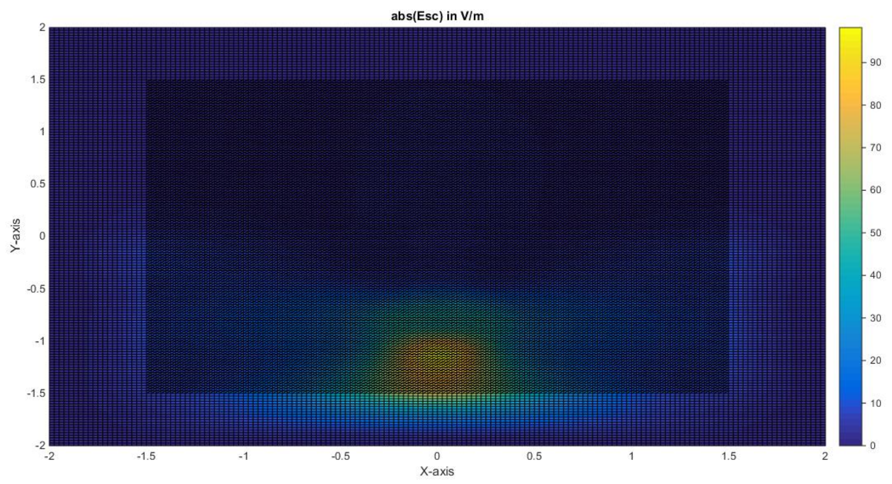


Figure 2.8: abs(Esc) 2D view

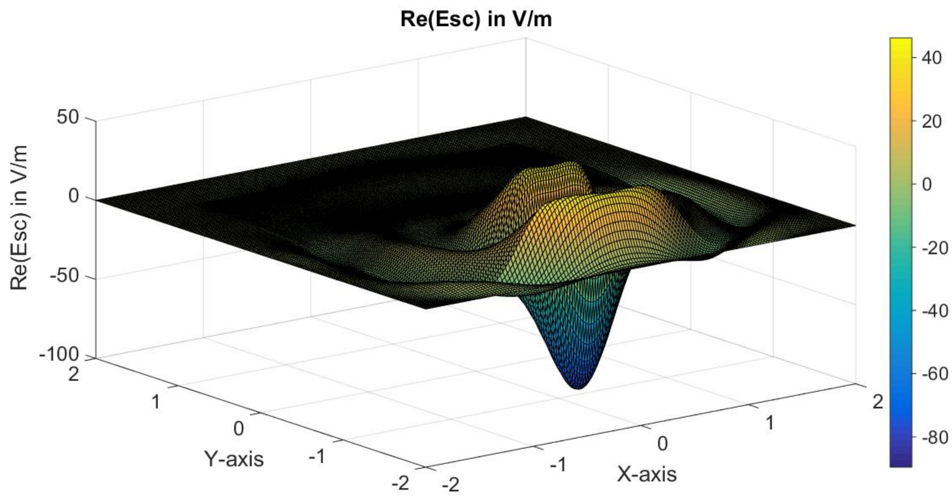


FIGURE 2.9: real(Esc) 3D view

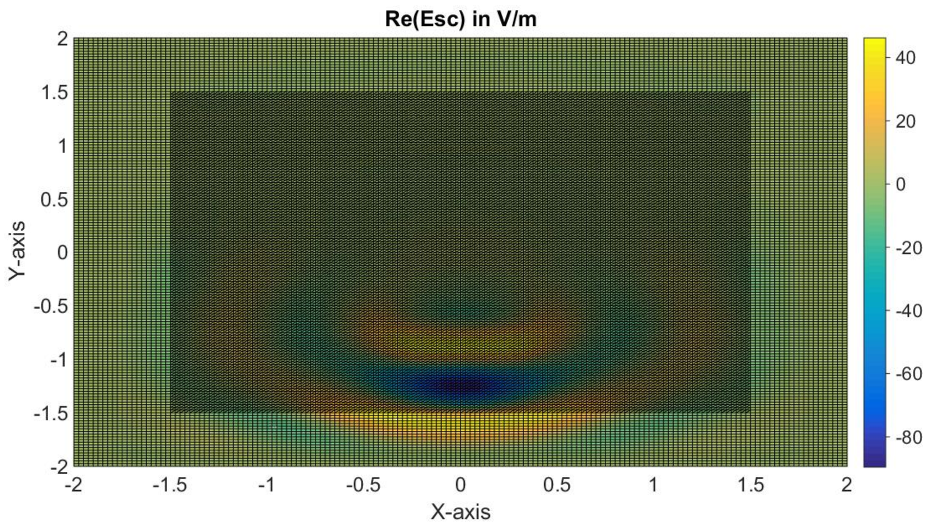


FIGURE 2.10: real(Esc) 2D view

2.8 Verification with COMSOL

We compared the results of forward model with COMSOL implementation of the same problem. The slight differences in the field values of MATLAB and COMSOL implementation is due to the way we have incorporated PML in our problem. Following are the required steps to implement this model in COMSOL: After launching a new instance of COMSOL on your system do:

- Go to File > New > Model Wizard
- Select Space Dimension > 2D
- Select Physics > Radio Frequency > Electromagnetic Waves, Frequency Domain(emw) then click on "Add"
- Click "Study"
- Select Study > Custom Studies > Stationary Studies then click on "Done"

Build geometry of the model :

- Add a rectangle with corner at (-1.5,0), width = 1.5m and height = 3m for air medium
- Add another rectangle with center at (0,-0.75), width = 1.5m and height = 3m for soil medium
- Add a circle of radius 0.4 with center at (0,-0.8).
- Add a rectangle with center at (0,0), width = 4m and height = 4m and insert a layer of 0.5m thickness from inside on each side of this rectangle in layers window. Select the check boxes "Layers to the left","Layers to the right","Layers on top","Layers on bottom"
- Right click "Definitions" and add perfectly matched layer. Select the PML region on the geometry.
- Add a Bezier Polygon. In "Polygon Segments" select "Add Linear". Enter control points (-2,0) and (2,0).
- Add a point at (-1,1) for source.

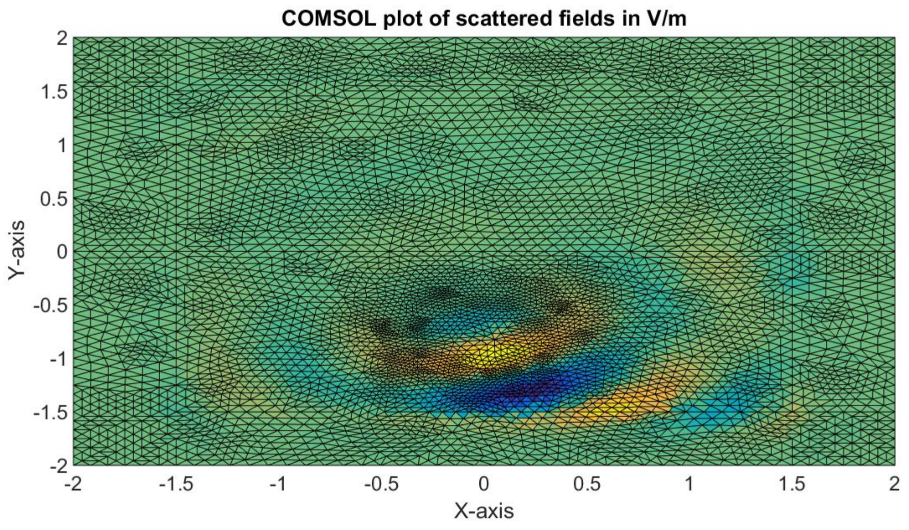


FIGURE 2.11: Scattered fields using COMSOL : 2D view

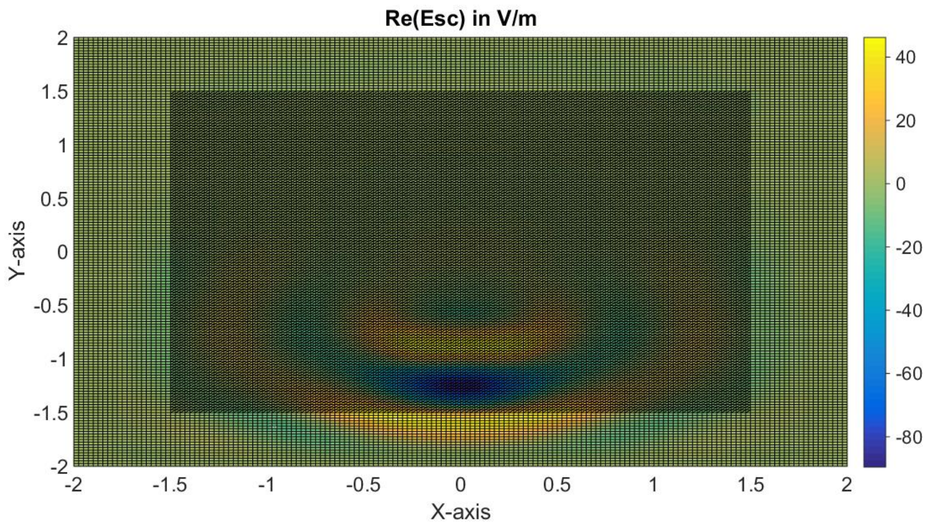


FIGURE 2.12: Scattered fields using MATLAB : 2D view

Add 3 blank materials. Rename them as air, soil and scatter.For scatterer select the circle in domain selection. For air and soil in domain selection, select PML surrounding that medium too. Assign values for relative permittivity, relative permeability and conductivity.

Under "Electromagnetic Waves, Frequency Domain(emw)" add a Line Current(Out-of-Plane) with a unit magnitude and in the Domain Selection select the point at (-1,1). Right click study and click on compute. This will give total fields. To get the background fields, we removed the scatter and simulated the model again. This gave just the background fields. We imported the total fields and background fields in a file and exported it to matlab. Algebraic difference between total and background field gave the scattered fields.

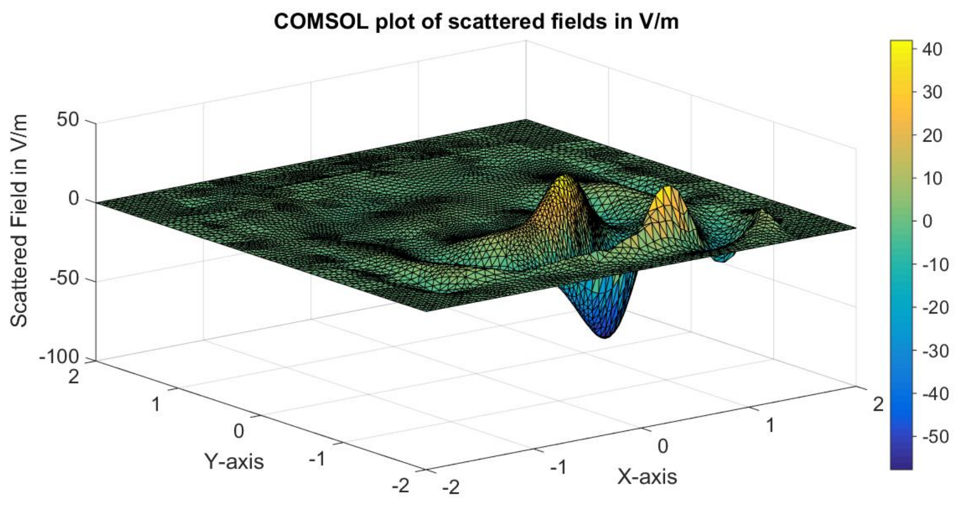


FIGURE 2.13: Scattered fields using COMSOL : 3D view

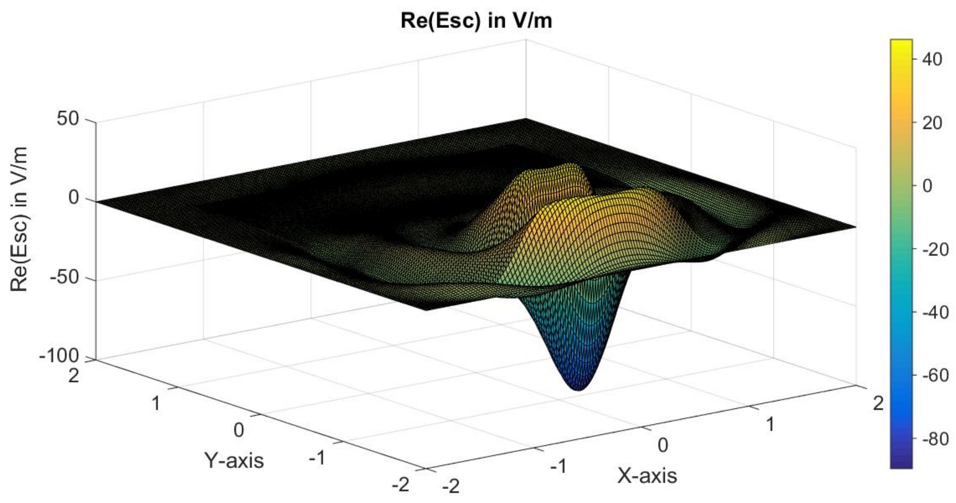


FIGURE 2.14: Scattered fields using matlab : 3D view

TABLE 2.1: Comparison with COMSOL results

Method	Co-ordinate	Scattered Field
MATLAB	$(0.125, -1.25)$	$-80.78V/m$
COMSOL	$(0.1991, -1.316)$	$-57.73V/m$

In COMSOL plot we found the bottom peak at (0.1991, -1.316) with a magnitude of -57.73V/m where in MATLAB plot we found the bottom peak at (0.125, -1.25) with a magnitude of -80.78V/m.

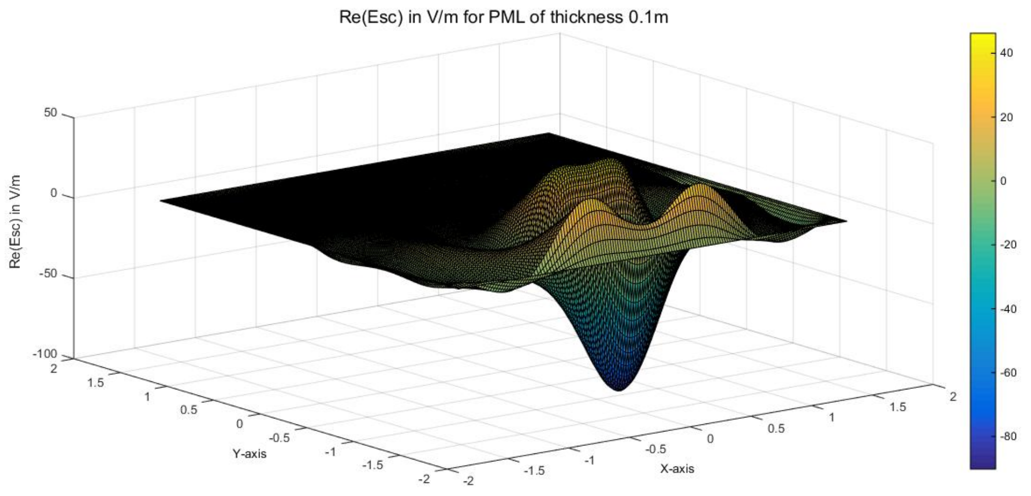


FIGURE 2.15: Re(Esc) with PML thickness=0.1m

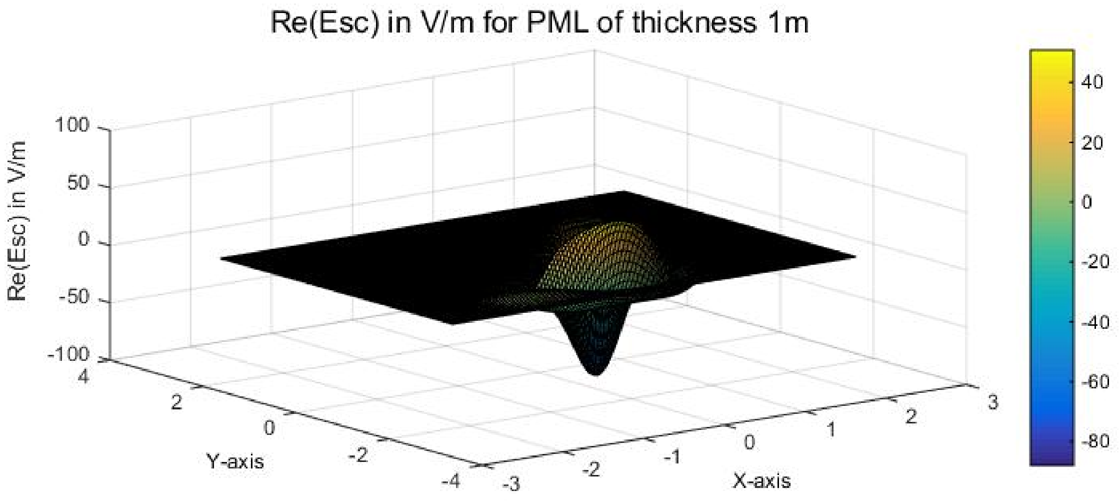


FIGURE 2.16: Re(Esc) with PML thickness=1m

2.9 Verification of the Absorbing Property of the PML

We verified the absorbing nature of the PML with varying PML thickness. We varied it from 0.1m to 2m. We found that even if there exist some ripples near the inner boundary of the PML, they get decayed while traveling outwards to the outer boundary of the PML.

We have also checked whether there are exists any scattered fields without inhomogeneities i.e. without scatterer. We found no scattering field in this case. So PML does not contribute to any scattered fields.

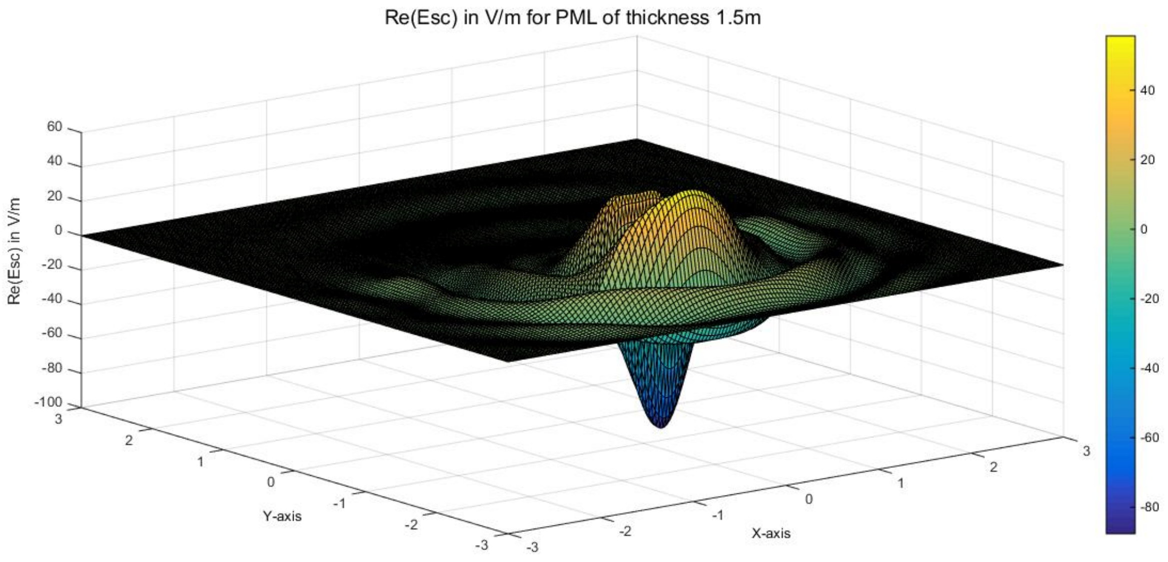


FIGURE 2.17: Re(Esc) with PML thickness=1.5m

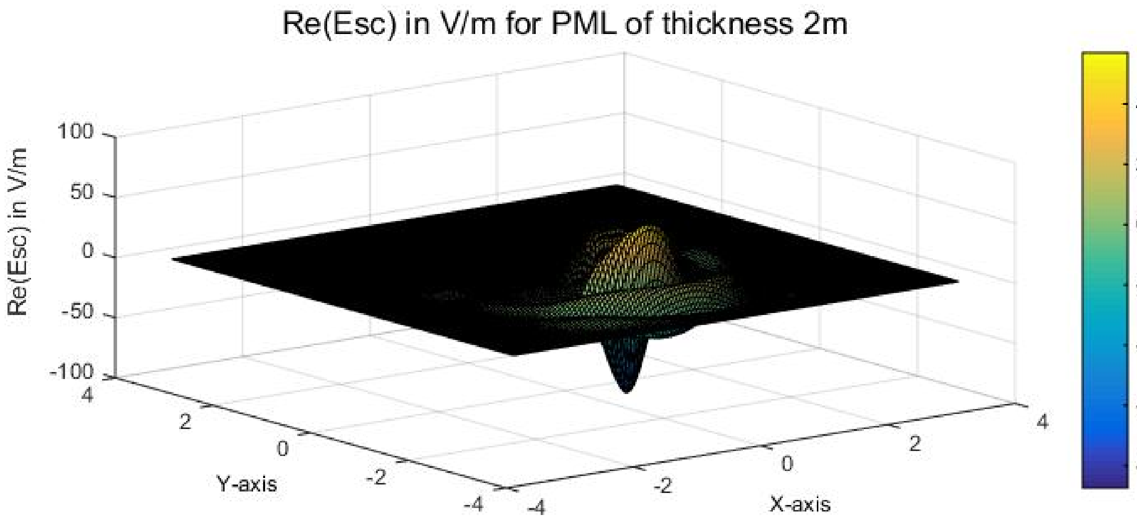


FIGURE 2.18: Re(Esc) with PML thickness=2m

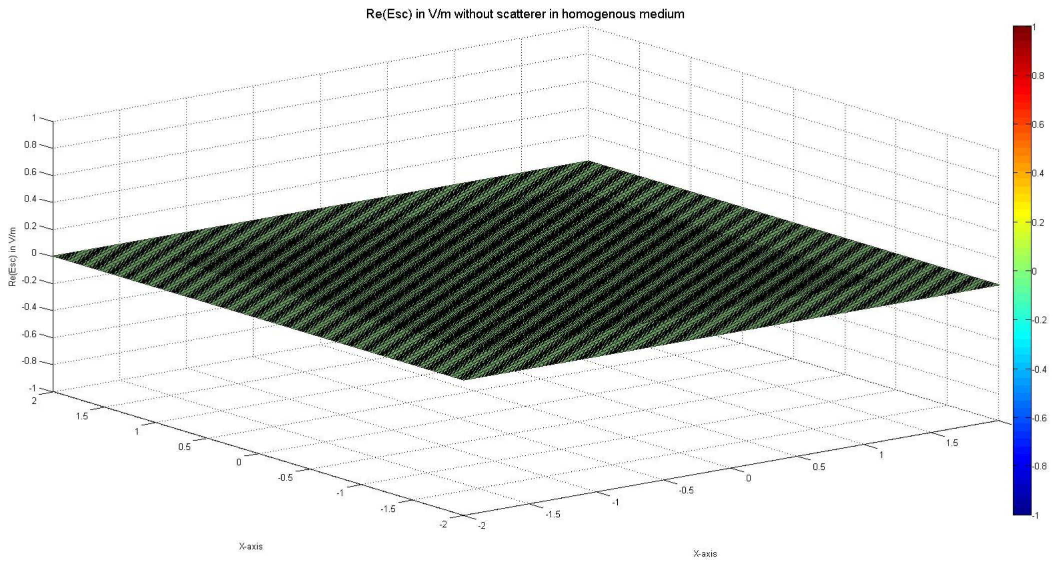


FIGURE 2.19: No scattered field due to PML

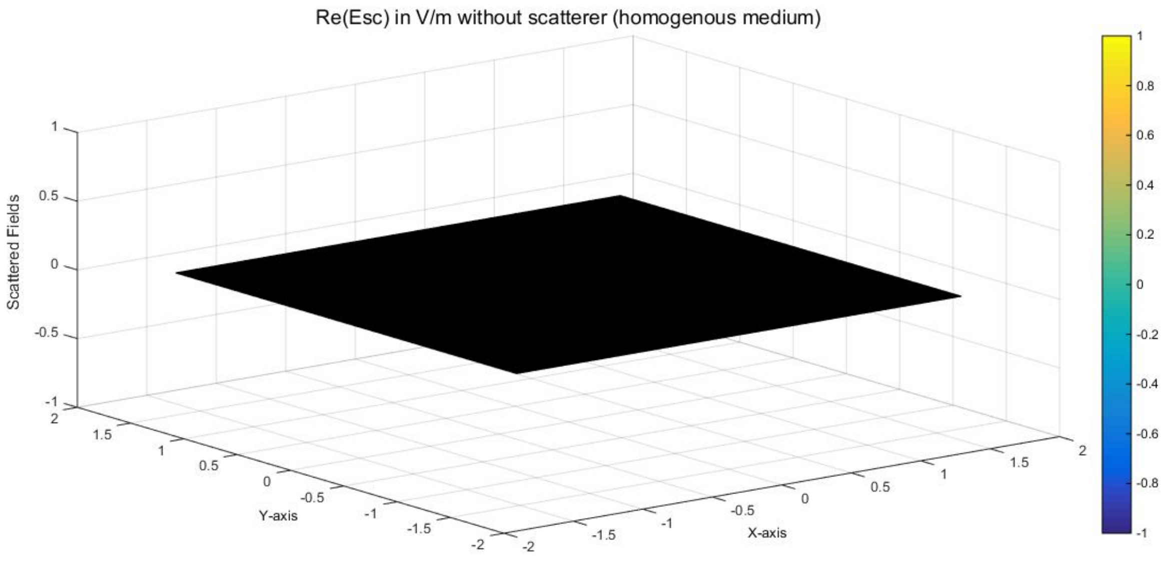


FIGURE 2.20: No scattered field due to PML with PML of thickness 0.1m

Chapter 3

Frechet Derivative Calculation and its Optimization

3.1 Introduction

The adjoint method is used to calculate the sensitivity matrix. In conventional FDM method, to calculate Frechet derivative we have to perturb each node which is computationally heavy. While in adjoint method we have to solve an adjoint problem only once which has similar computational requirements as solving the forward problem. We generally perturb the governing equation to formulate the adjoint problem. We relate the change in the forward response to a small perturbation in the model.

3.2 Perturbed Governing Equation

The basic governing equation is

$$\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}) + \omega^2(\epsilon_{zz} + \Delta\epsilon)E_z^{sc} = -\omega^2\Delta\epsilon E_z^{inc} \tag{3.1}$$

Let

$$L = \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial E_z^{sc}}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial E_z^{sc}}{\partial y}) \tag{3.2}$$

Let $\Delta\epsilon = \epsilon_s$,$E_z^{inc} = u_{inc}$,$E_z^{sc} = u_{sc}$ and $\delta E_{sc} = u_{sc}^{\delta}$. So the equation becomes

$$(L + \omega^2\epsilon + \omega^2\epsilon_s)u_{sc} = -\omega^2\epsilon_s u_{inc} \tag{3.3}$$

Perturbing the parameter ϵ_s to $\delta\epsilon_s$, perturbs u from u to u^δ. We can thus write the perturbed equation as

$$(L + \omega^2\epsilon + \omega^2(\epsilon_s + \delta\epsilon_s))(u_{sc} + u_{sc}^\delta) = -\omega^2(\epsilon_s + \delta\epsilon_s)u_{inc} \tag{3.4}$$

Solving Eq. (3.3) and Eq. (3.4) and igonring $\omega^2\delta\epsilon_s u_{sc}^\delta$ term, we get

$$(L + \omega^2\epsilon + \omega^2\epsilon_s)u_{sc}^\delta = -\omega^2\delta\epsilon_s u_{sc} - \omega^2\delta\epsilon_s u_{inc} \tag{3.5}$$

Let u be the total field, $u = u_{sc} + u_{inc}$.

So the equation becomes

$$(L + \omega^2\epsilon + \omega^2\epsilon_s)u_{sc}^\delta = -\omega^2\delta\epsilon_s u \tag{3.6}$$

Let

$$\mathcal{L} = L + \omega^2\epsilon + \omega^2\epsilon_s \tag{3.7}$$

3.3 Derivation of Adjoint System of Equations

Let Ψ^* be an adjoint field

$$< \Psi^*, \mathcal{L}u_{sc}^\delta >= \int_\Omega \Psi^*(\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial}{\partial y}) + \omega^2\epsilon + \omega^2\epsilon_s)u_{sc}^\delta d\Omega = -\int_\Omega \Psi^*\omega^2\delta\epsilon_s u d\Omega \tag{3.8}$$

Integration by parts

$$\int_\Omega \frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}\frac{\partial \psi^*}{\partial x}d\Omega = \int_\Gamma \frac{\Psi^*}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}d\Gamma - \int_\Omega \Psi^*\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x})d\Omega \tag{3.9}$$

This implies

$$\int_\Omega \Psi^*\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x})d\Omega = \int_\Gamma \frac{\Psi^*}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}d\Gamma - \int_\Omega \frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}\frac{\partial \psi^*}{\partial x}d\Omega \tag{3.10}$$

Let

$$\Psi^*|_\Gamma = 0 \tag{3.11}$$

which is a boundary condition for adjoint system.
So the first integral vanishes and we get

$$\int_\Omega \Psi^* \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x})d\Omega = -\int_\Omega \frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}\frac{\partial \psi^*}{\partial x}d\Omega \tag{3.12}$$

Similarly we can have

$$\int_\Omega \Psi^* \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial u_{sc}^\delta}{\partial y})d\Omega = -\int_\Omega \frac{1}{\mu_{xx}}\frac{\partial u_{sc}^\delta}{\partial y}\frac{\partial \psi^*}{\partial y}d\Omega \tag{3.13}$$

Again applying integration by parts on RHS of Eq. (3.12)

$$\int_\Omega u_{sc}^\delta \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x})d\Omega = \int_\Gamma \frac{u_{sc}^\delta}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x}d\Gamma - \int_\Omega \frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}\frac{\partial \psi^*}{\partial x}d\Omega \tag{3.14}$$

We know from forward problem that $u_{sc}^\delta|_\Gamma = 0$. So we get

$$-\int_\Omega \frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x}\frac{\partial \psi^*}{\partial x}d\Omega = \int_\Omega u_{sc}^\delta \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x})d\Omega \tag{3.15}$$

Similarly for Eq. (3.13)we can have

$$-\int_\Omega \frac{1}{\mu_{xx}}\frac{\partial u_{sc}^{delta}}{\partial y}\frac{\partial \psi^*}{\partial y}d\Omega = \int_\Omega u_{sc}^\delta \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial \Psi^*}{\partial y})d\Omega \tag{3.16}$$

So from Eq. (3.12),(3.13),(3.15) and (3.16) we can write

$$\int_\Omega \Psi^* \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial u_{sc}^\delta}{\partial x})d\Omega = \int_\Omega u_{sc}^\delta \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x})d\Omega \tag{3.17}$$

and

$$\int_\Omega \Psi^* \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial u_{sc}^\delta}{\partial y})d\Omega = \int_\Omega u_{sc}^\delta \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial \Psi^*}{\partial y})d\Omega \tag{3.18}$$

Using Eq. (3.17) and (3.18) in Eq. (3.8) we get

$$< \Psi^*, \mathcal{L}u_{sc}^\delta >= \int_\Omega u_{sc}^\delta(\frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial \Psi^*}{\partial y}) + \omega^2\epsilon\Psi^* + \omega^2\epsilon_s\Psi^*)d\Omega = -\int_\Omega \Psi^*\omega^2\delta\epsilon_s u d\Omega \tag{3.19}$$

Using Lagrangian identity and above equation

$$< \Psi^*, \mathcal{L}u_{sc}^\delta >=< u_{sc}^\delta, \mathcal{L}^*\Psi >= -\int_\Omega \Psi^*\omega^2\delta\epsilon_s u d\Omega \tag{3.20}$$

So

$$\mathcal{L}^*\Psi = \frac{\partial}{\partial x}(\frac{1}{\mu_{yy}}\frac{\partial \Psi^*}{\partial x}) + \frac{\partial}{\partial y}(\frac{1}{\mu_{xx}}\frac{\partial \Psi^*}{\partial y}) + \omega^2\epsilon\Psi^* + \omega^2\epsilon_s\Psi^* \tag{3.21}$$

$$\mathcal{L}^*\Psi = (L + \omega^2\epsilon + \omega^2\epsilon_s)\Psi^* \tag{3.22}$$

If we choose $\mathcal{L}^*\Psi = \delta(\vec{r} - \vec{r}_d)$, where $\vec{r}_d$ is detector location, we get

$$< u^\delta_{sc}, \mathcal{L}^*\Psi >= \int_\Omega u^\delta_{sc}\delta(\vec{r} - \vec{r}_d)d\Omega = u^\delta_{sc}(\vec{r}_d) \tag{3.23}$$

So from Eq. (3.20) we get

$$u^\delta_{sc}(\vec{r}_d) = -\int_\Omega \Psi^*\omega^2\delta\epsilon_s u d\Omega \tag{3.24}$$

This is a sensitivity relation which gives the change in the scattered field at detector locations with respect to change in the perturbed permittivity of the scatterer given in terms of adjoint field. We solve sensitivity relation using the method given in Fedele[18] So to get the adjoint field we solve Eq. (3.22) with $\mathcal{L}^*\Psi = \delta(\vec{r} - \vec{r}_d)$. So the adjoint field equation is

$$(L + \omega^2\epsilon + \omega^2\epsilon_s)\Psi^* = \delta(\vec{r} - \vec{r}_d) \tag{3.25}$$

along with the boundary condition from Eq.(3.11)

$$\Psi^*|_\Gamma = 0 \tag{3.26}$$

3.4 Numerical Results

We perturbed one element and compared the Jacobian values with finite difference method. The graphs shows comparison and error.

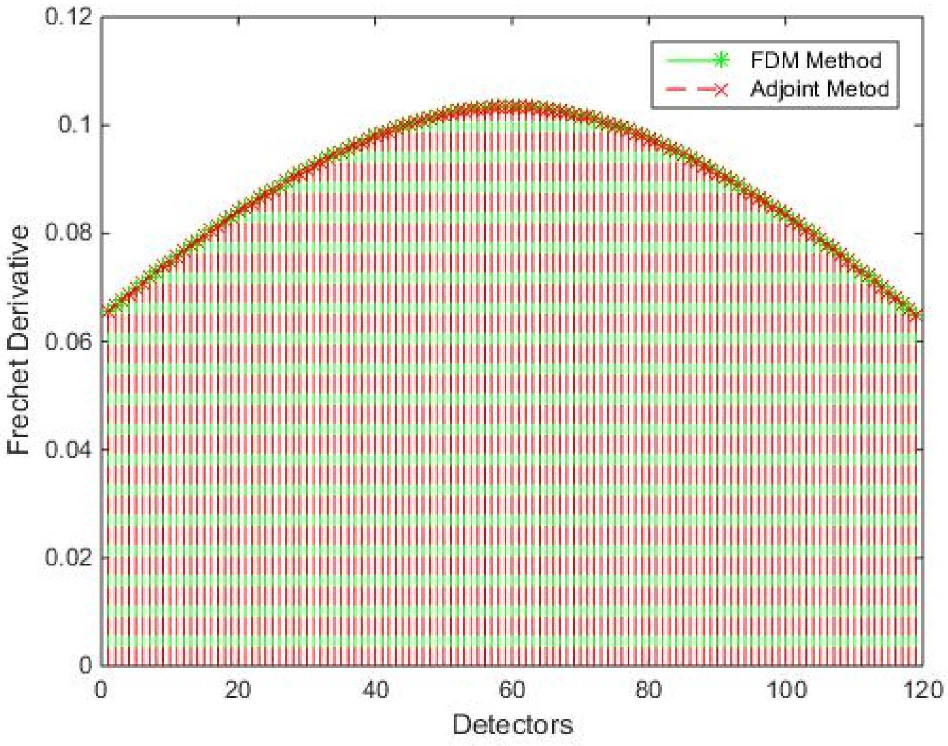


Figure 3.1: Comparison between Jacobian calculated with Adjoint Method and Jacobian with FDM (perturbed element inside scatterer)

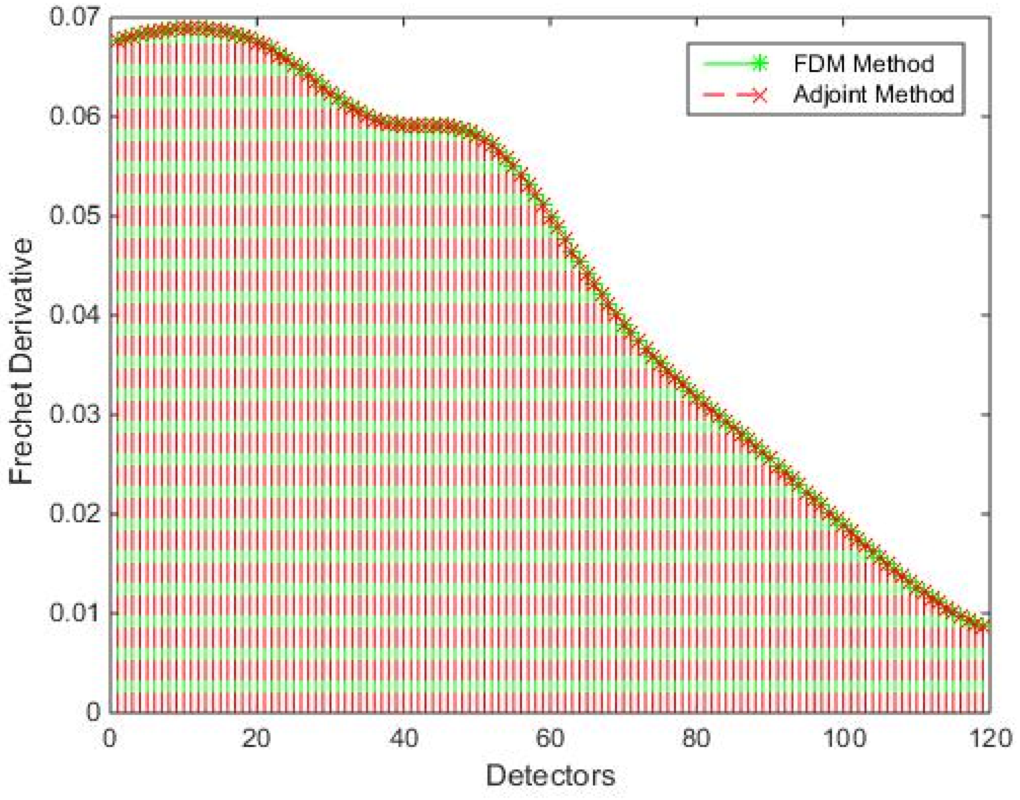


FIGURE 3.2: Comparison between Jacobian calculated with Adjoint Method and Jacobian with FDM (perturbed element in soil region)

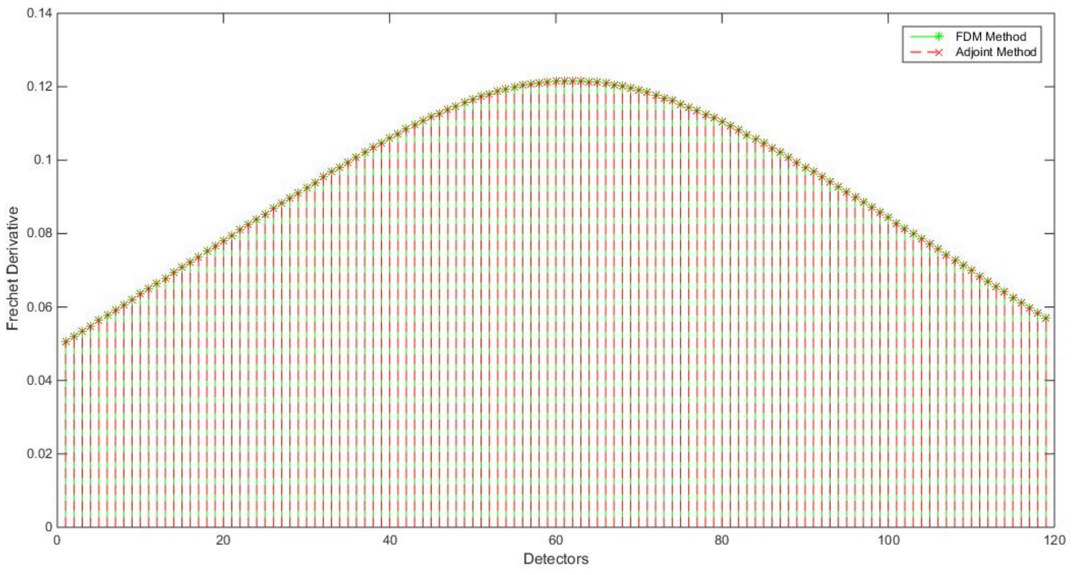


FIGURE 3.3: Comparison between Jacobian calculated with Adjoint Method and Jacobian with FDM (perturbed element at source location)

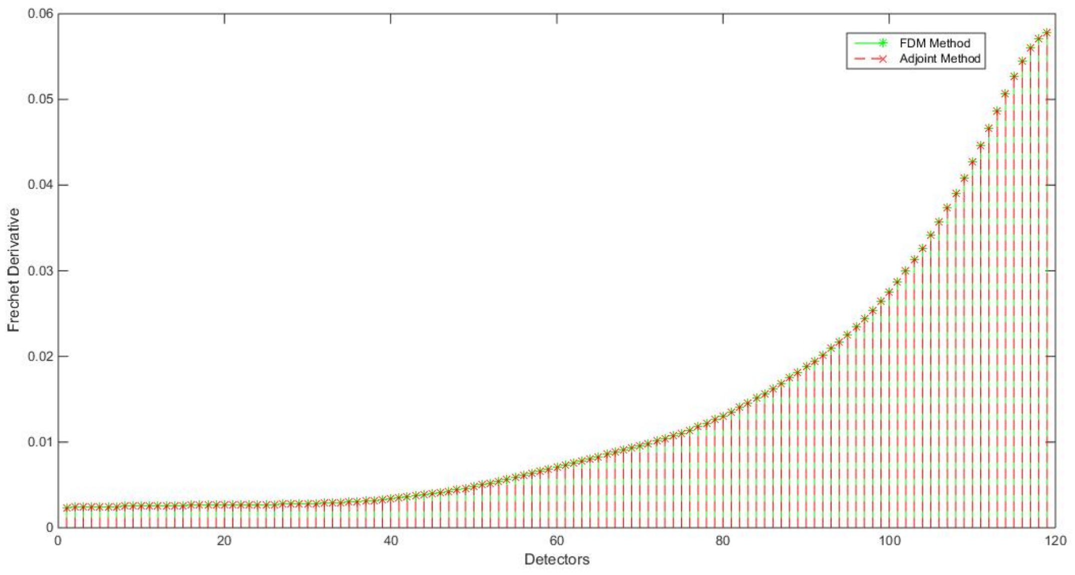


FIGURE 3.4: Comparison between Jacobian calculated with Adjoint Method and Jacobian with FDM (perturbed element in air)

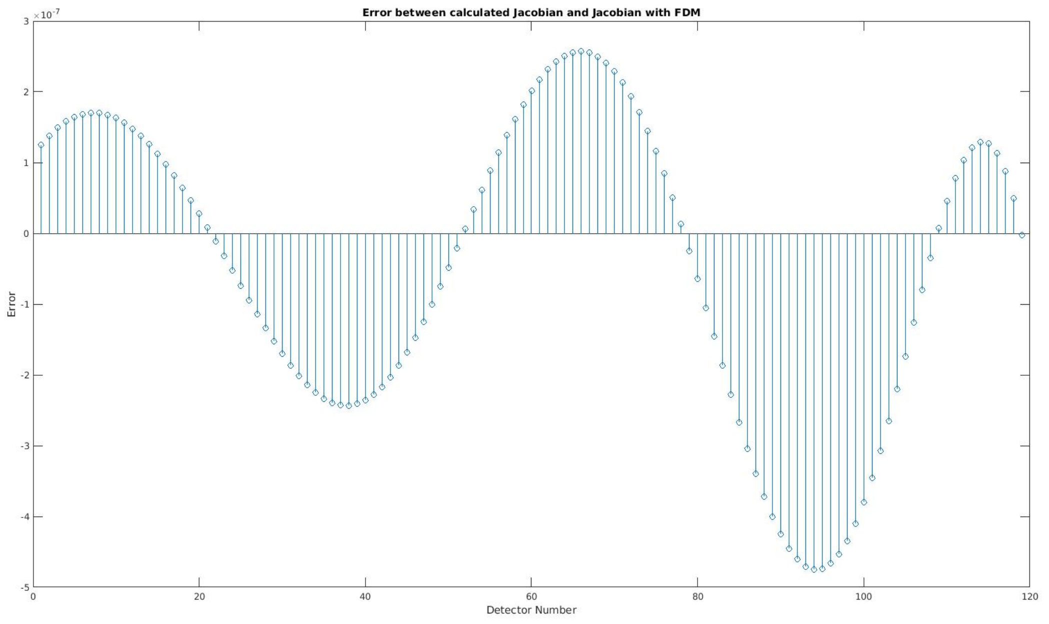


FIGURE 3.5: Error between Jacobian calculated using adjoint scheme and FDM method

3.5 Optimization of Different GPR configurations

We solved the forward problem for different configurations of source frequency and source locations. We tried to optimize the speed for each of these configurations.

We meshed the domain only once in the main function and called the routine which gives the scattered fields and Jacobian for each iteration. The problem with this approach was that we were looping over number of frequencies into number of sources times. Even if we ran the algorithm in a system with higher RAM, it was taking around 30 minutes for 12 frequencies and 12 sources configuration. And we were not able to decide which part of algorithm is taking more time.

We did profiling analysis on one iteration and found that system solving $Ax = b$ takes more time for adjoint field calculations.So we stacked different b columns of adjoint fields one after the other detector number of times and solved $Ax = b$ system with multiple right hand side.Further we divided the routine into different functions which separately calculates system matrix, scattered fields, adjoint fields and jacobian. We tried to speed these various functions within for loop using *parfor*. It speeded up algorithm to some extent for less number of source - frequency configurations. But still the system was going out of memory with increase in number of source configurations.

We solved this problem by storing all variables for each iteration in structures. The time got reduced to 8 minutes for 12 frequencies and 12 sources which was 30 minutes initially.

After further analysis we found that we were doing repetitive calculations for system matrix K. K changes only with frequency and hence can be computed only once for each frequency. And like adjoint field calculation we appended horizontally different b columns one after the other for one frequency and solved $Ax = b$ system for forward problem with multiple right hand side. So now we were looping only frequency number of times. The algorithm times got reduced to 174 seconds for 12 frequency and 12 source configuration.

So we started with our algorithm taking 30 minutes which got reduced to 8 minutes which further got reduced to 174 seconds.

3.6 Numerical Results

We used a system with 12GB RAM. The time taken by each function is shown below.

Table 3.1: Time taken by different parts of program

Variable	1 frequency 1 source	4 frequencies 4 sources	12 frequencies 12 sources
K	0.404549 *sec*	1.709784 *sec*	3.080221 *sec*
Incident Field	0.246786 *sec*	1.344582 *sec*	7.723653 *sec*
Scattered Field	0.229212 *sec*	1.333234 *sec*	7.689345 *sec*
Adjoint Fields	4.403231 *sec*	17.990988 *sec*	56.413914 *sec*
Jacobian	0.951136 *sec*	7.682130 *sec*	101.223031 *sec*

Chapter 4

Conclusion and Future Work

Unbounded PMLs [21], [23], [24], [22] are being used of late to truncate the computational domain in numerical solvers of various wave propagation models. Unbounded PMLs come with an advantage over classical polynomial PMLs such as no discretization error, no tuning of decay parameter [15].

In our work, with the objective of modelling a GPR tomography problem, we have implemented and optimized a PML based FEM solver for a 2D Helmoltz equation based setting, along with the adjoin-based Frechet-derivatives/sensitivities needed to solve the inverse problem.

We verified scattered fields using COMSOL. The Jacobian calculations are verified with finite difference scheme.

1. Further optimization can be achieved using various techniques of parallelization or using GPUs.
2. We can now proceed to a 3D solution of the vector scattering problem using these unbounded PMLs.

Appendix A

FEM

A.1 FEM Derivation for Forward Solution

Ω_a, Ω_1 and Ω_2 are the regions representing air, soil and scatterer mediums, as shown in the fig. (2.6)

$$\int_{\Omega_a+\Omega_1+\Omega_2} \frac{1}{\mu_0} \frac{\partial E_z^{sc}}{\partial x} \frac{\partial q}{\partial x} d\Omega + \int_{\Omega_a+\Omega_1+\Omega_2} \frac{1}{\mu_0} \frac{\partial E_z^{sc}}{\partial y} \frac{\partial q}{\partial y} d\Omega - \int_{\Omega_a} \omega^2 q \epsilon_0 E_z^{sc} d\Omega - \int_{\Omega_1} \omega^2 q \epsilon_0 \epsilon_1 E_z^{sc} d\Omega$$
$$- \int_{\Omega_2} \omega^2 q \epsilon_0 \epsilon_2 E_z^{sc} d\Omega + \int_{\Omega_{Pa}+\Omega_{Ps}} \frac{\sigma_y}{\mu_0 \sigma_z \sigma_x} \frac{\partial E_z^{sc}}{\partial x} \frac{\partial q}{\partial x} d\Omega + \int_{\Omega_{Pa}+\Omega_{Ps}} \frac{\sigma_x}{\mu_0 \sigma_y \sigma_z} \frac{\partial E_z^{sc}}{\partial y} \frac{\partial q}{\partial y} d\Omega E_z^{sc} d\Omega$$
$$- \int_{\Omega_{Pa}} \omega^2 q \epsilon_0 \frac{\sigma_x \sigma_y}{\sigma_z} d\Omega - \int_{\Omega_{Ps}} \omega^2 q \epsilon_0 \epsilon_1 \frac{\sigma_x \sigma_y}{\sigma_z} d\Omega = \int_{\Omega_2} \epsilon_0 (\epsilon_2 - \epsilon_1) q E_z^{inc} d\Omega \tag{A.1}$$

Non-derivative terms

$$\int_{\Omega} \frac{\partial E_z^{sc}}{\partial x} \frac{\partial q}{\partial x} d\Omega \tag{A.2}$$

Using nodal expansion in terms of its basis functions

$$E_z^{sc}(x,y) = \sum_{i=1}^{N} \phi_i N_i(x,y) \tag{A.3}$$

Substituting in Eq. (A.2)

$$\int_{\Omega} (\sum_{i=1}^{N} \phi_i \frac{\partial N_i}{\partial x}) \frac{\partial q}{\partial x} d\Omega \tag{A.4}$$

In Galerkin Formulation we take test function $q = \frac{\partial \phi}{\partial \phi_i} = N_j$, so we can write

$$\int_\Omega (\sum_{i=1}^{N} \phi_i \frac{\partial N_i}{\partial x}) \frac{\partial N_j}{\partial x} d\Omega \tag{A.5}$$

where j=1 to N. We solve this term over an element first and them assemble in global assembly later. Similarly the derivative term becomes

$$\int_\Omega q E_z^{sc} d\Omega = \int_\Omega (\sum_{i=1}^{N} \phi_i N_i) N_j \tag{A.6}$$

A.2 Elemental Interpolation

The unknown function ϕ within each triangular element is approximated as

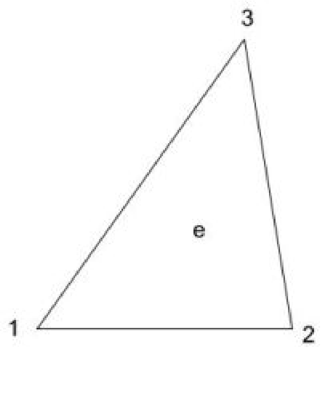
$$\phi^e(x, y) = a^e + b^e x + c^e y \tag{A.7}$$

At the three nodes, we obtain

$$\begin{aligned}
\phi_1^e &= a^e + b^e x_1 + c^e y_1^e \\
\phi_2^e &= a^e + b^e x_2 + c^e y_2^e \\
\phi_3^e &= a^e + b^e x_3 + c^e y_3^e
\end{aligned} \tag{A.8}$$

Solving for the constant coefficients a^e, b^e and c^e in terms of ϕ_j^e and substituting

$$\phi^e(x, y) = \sum_{j=1}^{3} N_j^e(x, y) \phi_j^e \tag{A.9}$$



where $N_j^e(x,y)$ are the interpolation or expansion functions given by

$$N_j^e(x,y) = \frac{1}{2\Delta^e}(a_j^e + b_j^e x + c_j^e y) \qquad j = 1,2,3 \tag{A.10}$$

in which

$$\begin{aligned} a_1^e &= x_2^e y_3^e - y_2^e x_3^e; & b_1^e &= y_2^e - y_3^e; & c_1^e &= x_3^e - x_2^e \\ a_2^e &= x_3^e y_1^e - y_3^e x_1^e; & b_2^e &= y_3^e - y_1^e; & c_2^e &= x_1^e - x_3^e \\ a_3^e &= x_1^e y_2^e - y_1^e x_2^e; & b_3^e &= y_1^e - y_2^e; & c_3^e &= x_2^e - x_1^e \end{aligned} \tag{A.11}$$

and

$$\Delta^e = \frac{1}{2}\begin{vmatrix} 1 & x_1^e & y_1^e \\ 1 & x_2^e & y_2^e \\ 1 & x_3^e & y_3^e \end{vmatrix} = \frac{1}{2}(b_1^e c_2^e - b_2^e c_1^e) \tag{A.12}$$

$$= \text{area of the eth element} \tag{A.13}$$

Interpolation function have the property

$$N_i^e(x_j^e, y_j^e) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \tag{A.14}$$

$$K_{stiffness} = \int_{\Omega_e} (\frac{\partial N_i}{\partial x}\frac{\partial N_j}{\partial x} + \frac{\partial N_i}{\partial y}\frac{\partial N_j}{\partial y}) d\Omega = [\int_{\Omega_e} (\nabla[N])^T \nabla[N]] \tag{A.15}$$

$$K_{mass} = \int_{\Omega_e} N_i N_j d\Omega = [\int_{\Omega_e} ([N])^T [N]] \tag{A.16}$$

Closed form solution

$$\int_{\Omega_e} (N_1^e)^l (N_2^e)^m (N_3^e)^n d\Omega = \frac{l!m!n!}{(l+m+n+2)!} 2\Delta^e \tag{A.17}$$

Hence we get

$$K_{stiffness} = \begin{bmatrix} \frac{b_1^2+c_1^2}{4A_e} & \frac{b_1b_2+c_1c_2}{4A_e} & \frac{b_1b_3+c_1c_3}{4A_e} \\ \frac{b_1b_2+c_1c_2}{4A_e} & \frac{b_2^2+c_2^2}{4A_e} & \frac{b_2b_3+c_2c_3}{4A_e} \\ \frac{b_3b_1+c_3c_1}{4A_e} & \frac{b_3b_2+c_3c_2}{4A_e} & \frac{b_3^2+c_3^2}{4A_e} \end{bmatrix} \tag{A.18}$$

Similarly

$$K_{mass} = \omega^2 \epsilon \begin{bmatrix} \frac{\Delta^e}{6} & \frac{\Delta^e}{12} & \frac{\Delta^e}{12} \\ \frac{\Delta^e}{12} & \frac{\Delta^e}{6} & \frac{\Delta^e}{12} \\ \frac{\Delta^e}{12} & \frac{\Delta^e}{12} & \frac{\Delta^e}{6} \end{bmatrix} \tag{A.19}$$

A.3 FEM implementation of Sensitivity Equation

Sensitivity equation is given by

$$u^{\delta}(\vec{r}_d) = -\int_{\Omega} \Psi^{*}\omega^{2}\delta\epsilon_s u_T d\Omega \tag{A.20}$$

In a particular element Ω_e

$$u^{\delta}(\vec{r}_d) = -\int_{\Omega_e} \Psi^{*e}\omega^{2}\delta\epsilon_s u_T^{e} d\Omega \tag{A.21}$$

We assume a dummy node for all the triangular elements with node number one more than the total number of nodes and assign zero field value at dummy node. So now all the elements will have 4 nodes. We expand Ψ^{*} and u_T within the element Ω_e using the same basis function $N_i(x, y)$.

$$u^{\delta}(\vec{r}_d) = \delta\epsilon_s[-\int_{\Omega_e} \sum_{i=1}^{4}\sum_{j=1}^{4} N_i(x,y)N_j(x,y)\Psi_i^{*e}u_{Tj}^{e} d\Omega] \tag{A.22}$$

We can write this in matrix form as

$$u^{\delta}(\vec{r}_d) = \delta\epsilon_s[-\begin{bmatrix}\Psi_1^{*e} & \Psi_2^{*e} & \Psi_3^{*e} & \Psi_4^{*e}\end{bmatrix}\begin{bmatrix}\int_{\Omega_e} N_i(x,y)N_j(x,y)d\Omega\end{bmatrix}\begin{bmatrix}u_{T1}^{e}\\ u_{T2}^{e}\\ u_{T3}^{e}\\ u_{T4}^{e}\end{bmatrix}] \tag{A.23}$$

Further we use Fedele's [18] approach to solve this equation.

Appendix B

Codes

B.1 Main Routine

```
% clear all;
% close all;
% clc;
n=17; % No. of nodes along either x or y in the chosen square geometry

nx=n-mod(n,16)+1; %No. of nodes along x-direction
%As per Bermudez, there is an even number of rectangles in the PML region
% To make (nx-1) a perfect multiple of 16
ny=n-mod(n,16)+1;

% as=input('enter outer boundary along x (a*) : ');
% bs=input('enter outer boundary along y (b*) : ');
as=2; %The domain-rectangle: (1)PML : -as to as and -bs to bs,
%(2) Physical region: -a to a and -b to b
bs=2;
a=1.5;
b=1.5;
dx=2*as/(nx-1);
dy=2*bs/(ny-1);
N=round((as-a)/dx); %No. of rectangles along a direction in the PML
At=0.5*dx*dy; %At=area of a triangle
Ar=dx*dy; %Ar=area of a rectangle

p=zeros(nx*ny,2); % storing co-ordinates of nodes
tot_nodes=nx*ny;  % total number of nodes
tri_elements=2*(nx-1-2*N)*(ny-1-2*N);  % total number of triangular elements
rec_elements=2*N*(nx-1)+2*N*(ny-1-2*N);      % total number of  rectangular elements
t_tri=zeros(tri_elements,3);% for local node number to global node numbers of triangles
t_rec=zeros(rec_elements,4); % for local node number to global node numbers of rectangles

% storing x and y coordinates of nodes :Remove for-loop below
```

```
vecx=-as:dx:as;
vecy=-bs:dy:bs;
vecx_new=repmat(vecx',1,ny);
vecx_new=vecx_new(:);
vecy_new=repmat(vecy,nx,1);
vecy_new=vecy_new(:);
p=[vecx_new vecy_new];

%[x,y]=coordinates_from_mesh(p);
x=p(:,1);
y=p(:,2);
%Mapping for local node number to global node numbers of triangles
% Local node numbering of an element is anticlockwise for both triangle and rectangle
% and begins from bottom left node
% Global node numbering of the meshgrid begins from bottom row left most node
% each row contains lower triangles fitted with upside down triangles
for i=1:(nx-1)-2*N % lower triangles of last row of physical region
    t_tri(i,1)=N*nx+N+i;
    t_tri(i,2)=N*nx+N+i+1;
    t_tri(i,3)=(N+1)*nx+N+i+1;
end
for i=(nx-1)-2*N+1:2*(nx-1)-2*2*N % upper upside down triangles
    t_tri(i,1)=N*nx+N+i-((nx-1)-2*N);
    t_tri(i,2)=(N+1)*nx+N+i+1-((nx-1)-2*N);
    t_tri(i,3)=(N+1)*nx+N+i-((nx-1)-2*N);
end
for i=2*(nx-1)-2*2*N+1:tri_elements % replicating periodically for the rows above
    for j=1:3
        t_tri(i,j)=t_tri(i-2*(nx-1-2*N),j)+nx;
    end
end

% mapping for local node number to global node numbers of rectangles

% bottom horizontal rows in PML
for i=1:nx-1 % last row
    t_rec(i,1)=i;
    t_rec(i,2)=i+1;
    t_rec(i,3)=nx+i+1;
    t_rec(i,4)=nx+i;
end
for i=nx:N*(nx-1) %replicating periodically for above rows
    for j=1:4
        t_rec(i,j)=t_rec(i-(nx-1),j)+nx;
    end
end

%sides : left and right side of physical region in PML
for i=N*(nx-1)+1:N*(nx-1)+N
    for j=1:4
        t_rec(i,j)=t_rec(i-(nx-1),j)+nx;
    end
end
for i=N*(nx-1)+N+1:rec_elements-N*(nx-1)
    for j=1:4
        t_rec(i,j)=t_rec(i-2*N,j)+nx;
    end
end
%top
for i=rec_elements-N*(nx-1)+1:rec_elements
    for j=1:4
        t_rec(i,j)=t_rec(i-(rec_elements-N*(nx-1)),j)+nx*(ny-1-N);
    end
end
% toc;

% Forward Problem
parameters_material;

[air_tri,soil_tri]=element_classifier(t_tri,nx,ny,N);
% [air_pml,soil_pml]=pml_classifier2(t_rec,nx,ny);

% [xc,yc]=centroid_triangle(x,y,soil_tri);
[xc,yc]=centroid_triangle(x,y,t_tri);

y0=-0.80;
x0=0;
rad=0.4;

[sgn_distance]=element_signed_distance(xc,yc,x0,y0,rad);
[eps]=eps_calculator(er1,tand_1,erd,tand_d,eps0,t_tri,sgn_distance);
[eps_homo]=eps_homo_calculator(er1,tand_1,eps0,t_tri,sgn_distance);

% [eps_pml]=eps_pml_calculator(eps0,er1,tand_1,t_rec,nx,ny);
% tic;

freq=2; %number of frequencies
loc=2;  %number of source locations

for i=1:loc
    source_location_from_left(i)=round((nx-2*N)*((1/(loc+1))*i));
end

for i=1:freq
    omegas(i)=parameters_source(250+(i-1)*50);
end

tic;
parfor i=1:freq
    [homo(i).K,inhomo(i).K]=cal_k(omegas(i),...
        as,bs,x,y,dx,dy,nx,ny,N,...
        t_tri,t_rec,...
        eps,eps_homo);
```

```
end
toc;

tic;
parfor i=1:freq
    inc(i).Einc=cal_Einc(loc,source_location_from_left,omegas(i),homo(i).K,...
        nx,ny,dx,x,y,a);
end
toc;


scat_tri=soil_tri(sgn_distance>0,:);
del_eps=eps-eps_homo;
[row_inc,col_inc,val_inc]=incident(x,y,t_tri,del_eps);
Krhs=sparse(row_inc,col_inc,val_inc,nx*ny,nx*ny);


tic;
parfor i=1:freq
    sc(i).Esc=cal_Esc(inhomo(i).K,omegas(i),del_eps,Krhs,inc(i).Einc);
end
toc;


tic;
line=nonzeros((y==dx*round(0.25/dx) & (abs(x)<a)).*(1:1:nx*ny)');
measure_line=[line,x(line)];
detectors=line;

b_adjoint=zeros(nx*ny,size(detectors,1));
for i=1:size(detectors,1)
    b_adjoint(detectors(i),i)=-1;
end
toc;

tic;
parfor i=1:freq
     adjoint(i).adj=inhomo(i).K\b_adjoint;
end
toc;


[k_ninj_mass] = ninj_mass(x,y,t_tri,t_rec);

t_tri_app0=[t_tri (nx*ny+1)*ones(size(t_tri,1),1)];
t_tri_rec=[t_tri_app0; t_rec];


tic;
parfor i=1:freq
    JT(i).jaco=jacobian_new(loc,sc(i).Esc,adjoint(i).adj,omegas(i),eps,eps0,...
        detectors,k_ninj_mass,nx,ny,...
        t_tri_rec,tri_elements,rec_elements);
end
toc;
```

B.2 Calculation of K

```
function [ K_homo,K_inhomo ] = cal_k(omegas,...
                           as,bs,x,y,dx,dy,nx,ny,N,...
                           t_tri,t_rec,...
                           eps,eps_homo)


parameters_material;


% % mass matrix
[row_mass,col_mass,val_mass] = mass_mat(x,y,t_tri,eps,omegas);


% soil & air stiffness matrix
[row_stiff,col_stiff,val_stiff]=stiffness_mat(x,y,t_tri,mu0);

[t_rec1,t_rec2,t_rec3]=pml_types(x,y,nx,ny,N,dx,dy,t_rec);
[row_pml1,col_pml1,val_pml1]=pml1(x,y,t_rec1,as,omegas,c,eps0,er1,tand_1,mu0);
[row_pml2,col_pml2,val_pml2]=pml2(x,y,t_rec2,bs,omegas,c,eps0,er1,tand_1,mu0);
[row_pml3,col_pml3,val_pml3]=pml3(x,y,t_rec3,as,bs,omegas,c,eps0,er1,tand_1,mu0);

t_recnodes=t_rec(:);
t_bc=unique(t_recnodes(x(t_recnodes)==as | x(t_recnodes)==-as...
              | y(t_recnodes)==bs | y(t_recnodes)==-bs));

[row_bc,col_bc,val_bc,pd]=dirichlet_boundary_conditions(t_bc);

row=[row_mass;row_stiff;row_pml1;row_pml2;row_pml3;row_bc];
col=[col_mass;col_stiff;col_pml1;col_pml2;col_pml3;col_bc];
val=[val_mass;val_stiff;val_pml1;val_pml2;val_pml3;val_bc];

K_inhomo=sparse(row,col,val,ny*nx,ny*nx);

[row_mass_homo,col_mass_homo,val_mass_homo] = mass_homo(x,y,t_tri,eps_homo,omegas);

row_homo=[row_mass_homo;row_stiff;row_pml1;row_pml2;row_pml3;row_bc];
col_homo=[col_mass_homo;col_stiff;col_pml1;col_pml2;col_pml3;col_bc];
val_homo=[val_mass_homo;val_stiff;val_pml1;val_pml2;val_pml3;val_bc];
K_homo=sparse(row_homo,col_homo,val_homo,ny*nx,ny*nx);

end
```

B.3 Calculation of E_{inc}

```
function Einc=cal_Einc(loc,source_location_from_left,omegas,K_homo,...
    nx,ny,dx,x,y,a)

source=zeros(nx*ny,loc);

line=nonzeros((y==dx*round(0.25/dx) & (abs(x)<a)).*(1:1:nx*ny)');
measure_line=[line,x(line)];
source_node=measure_line(source_location_from_left,1);

for i=1:loc
    source(source_node(i),i)=-1j*omegas;
end

Einc=K_homo\source;

end
```

B.4 Calculation of E_{sc}

```
function Esc=cal_Esc(K_inhomo,omegas,del_eps,Krhs,Einc)

rhs_vector=omegas^2*del_eps*Krhs*Einc;

Esc=K_inhomo\rhs_vector;

end
```

B.5 Calculation of Jacobian

```
function [J_new]=jacobian_new(loc,Esc,adjoint,omega,eps,eps0,detectors,...
    k_ninj_mass,nx,ny,...
    t_tri_rec,tri_elements,rec_elements)

Esc(nx*ny+1,:)=0;
Einc(nx*ny+1,:)=0;

Esc_new=Esc(:);
Einc_new=Einc(:);

sz=size(t_tri_rec,1);
E_app=zeros(loc*sz,4);
for i=1:loc
    E_app((i-1)*sz+1:i*sz,:)=Esc_new(t_tri_rec + (i-1)*(nx*ny+1))
    +Einc(t_tri_rec + (i-1)*(nx*ny+1));

end

E_new=zeros(1,4,loc*(tri_elements+rec_elements));

E_new(1,:,:)=transpose(E_app);
k_ninj_mass_new=repmat(k_ninj_mass,[1 1 loc]);

for i=1:4
    L(i,:,:)=-k_ninj_mass_new(i,:,:).*omega^2.*eps0;
    T(i,1,:)=sum(L(i,:,:).*E_new,2);
end

T_new=repmat(T,1,size(detectors,1),1);

adjoint(nx*ny+1,:)=0;
adjoint_app=zeros(tri_elements+rec_elements,4,size(detectors,1));
for i=1:size(detectors,1)
    temp=adjoint(:,i);
    adjoint_app(:,:,i)=temp(t_tri_rec);
end

adjoint_app=permute(adjoint_app,[2 3 1]);
sz_adj=size(adjoint_app,3);
for i=1:loc
    temp2=adjoint_app.*T_new(:,:,1+(i-1)*sz_adj:i*sz_adj);
end

J=sum(temp2,1);
J_new(:,:)=J(1,:,:);

end
```

B.6 Parameters

```
c=3*10^8; % in meter per sec
mu0=4*pi*10^(-7);
eps0=8.854187817*10^(-12); %correct

er0=1; tand_0=0; %Air

er1=2.5; tand_1=0.017;   % sand
erd=3; tand_d=0.005;     %scatter
mur0=1; mur1=1; murd=1;
```

B.7 Mass Matrix

```
function [row_mass,col_mass,val_mass] = mass_mat(x,y,t_tri,eps,omega)

[a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae]=elemental_basis(x,y,t_tri);

k_mass=elemental_assembly(a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae);

for i=1:3
    for j=1:3
        k_mass(i,j,:)=(-k_mass(i,j,:).*eps)*omega*omega;
    end
end

% Filler
[row_mass,col_mass,val_mass]=tri_filler(t_tri,k_mass);
end

function [a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae]=elemental_basis(x,y,elemental_tri)
x1a=x(elemental_tri(:,1)); y1a=y(elemental_tri(:,1));
x2a=x(elemental_tri(:,2)); y2a=y(elemental_tri(:,2));
x3a=x(elemental_tri(:,3)); y3a=y(elemental_tri(:,3));

b1e=y2a-y3a;
b2e=y3a-y1a;
b3e=y1a-y2a;

c1e=x3a-x2a;
c2e=x1a-x3a;
c3e=x2a-x1a;

a1e=x2a.*y3a-y2a.*x3a;
a2e=x3a.*y1a-y3a.*x1a;
a3e=x1a.*y2a-y1a.*x2a;

areae=0.5*(b1e.*c2e-b2e.*c1e);
end

function k_elemental=elemental_assembly(a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae)
```

```
k_elemental(1,1,:)=areae./6.;
k_elemental(1,2,:)=areae./12.;
k_elemental(1,3,:)=areae./12.;
k_elemental(2,1,:)=areae./12.;
k_elemental(2,2,:)=areae./6.;
k_elemental(2,3,:)=areae./12.;
k_elemental(3,1,:)=areae./12.;
k_elemental(3,2,:)=areae./12.;
k_elemental(3,3,:)=areae./6.;

end
```

B.8 Stffness Matrix

```
function [row_stiff,col_stiff,val_stiff]=stiffness_mat(x,y,tri,mu0)

[a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae]=stiffness_basis(x,y,tri);

k_stiff=stiffness_assembly(a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae);

for i=1:3
    for j=1:3
        k_stiff(i,j,:)=k_stiff(i,j,:).*(1/mu0);
    end
end

[row_stiff,col_stiff,val_stiff]=tri_filler(tri,k_stiff);

end

function [a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae]=stiffness_basis(x,y,tri)
x1a=x(tri(:,1)); y1a=y(tri(:,1));
x2a=x(tri(:,2)); y2a=y(tri(:,2));
x3a=x(tri(:,3)); y3a=y(tri(:,3));


b1e=y2a-y3a;
b2e=y3a-y1a;
b3e=y1a-y2a;
```

```
c1e=x3a-x2a;
c2e=x1a-x3a;
c3e=x2a-x1a;

a1e=x2a.*y3a-y2a.*x3a;
a2e=x3a.*y1a-y3a.*x1a;
a3e=x1a.*y2a-y1a.*x2a;

areae=0.5*(b1e.*c2e-b2e.*c1e);
end

function k_stiff=stiffness_assembly(a1e,b1e,c1e,...
    a2e,b2e,c2e,...
    a3e,b3e,c3e,...
    areae)
k_stiff(1,1,:)=areae.*(b1e.^2./(4.*areae.^2) + c1e.^2./(4.*areae.^2));
k_stiff(1,2,:)=areae.*((b1e.*b2e)./(4.*areae.^2) + (c1e.*c2e)./(4.*areae.^2));
k_stiff(1,3,:)=areae.*((b1e.*b3e)./(4.*areae.^2) + (c1e.*c3e)./(4.*areae.^2));
k_stiff(2,1,:)=areae.*((b1e.*b2e)./(4.*areae.^2) + (c1e.*c2e)./(4.*areae.^2));
k_stiff(2,2,:)=areae.*(b2e.^2./(4.*areae.^2) + c2e.^2./(4.*areae.^2));
k_stiff(2,3,:)=areae.*((b2e.*b3e)./(4.*areae.^2) + (c2e.*c3e)./(4.*areae.^2));
k_stiff(3,1,:)=areae.*((b1e.*b3e)./(4.*areae.^2) + (c1e.*c3e)./(4.*areae.^2));
k_stiff(3,2,:)=areae.*((b2e.*b3e)./(4.*areae.^2) + (c2e.*c3e)./(4.*areae.^2));
k_stiff(3,3,:)=areae.*(b3e.^2./(4.*areae.^2) + c3e.^2./(4.*areae.^2));
end
```

B.9 trifiller

```
function [Iv,Jv,k]=tri_filler(tri,kk)
    % construct IV, JV
    Iv=repmat(tri,1,3);
    Jv=tri';
    Iv=Iv';
    Jv=Jv(:);
    Jv=Jv';
    Iv=Iv(:);
    Jv=repmat(Jv,3,1);
    Jv=Jv(:);
    k=kk(:);
end
```

B.10 Boundary Condition

```
function [row_bc,col_bc,val_bc,pd]=dirichlet_boundary_conditions(t_bc)
nd=unique(t_bc);
```

```
pd=zeros(size(nd,1),1);

row_bc=nd;
col_bc=nd;
val_bc=ones(size(nd,1),1);
end
```

B.11 PML functions

PML types

```
function [t_rec1,t_rec2,t_rec3]=pml_types(x,y,nx,ny,N,dx,dy,t_rec)
t_rec1=t_rec(y(t_rec(:,1))>= -(0.5*(ny-1)-N)*dy & y(t_rec(:,4))<=(0.5*(ny-1)-N)*dy,:);
t_rec2=t_rec(x(t_rec(:,1))>= -(0.5*(nx-1)-N)*dx & x(t_rec(:,2))<=(0.5*(nx-1)-N)*dx,:);
t_rec3=t_rec((y(t_rec(:,1))>= -(0.5*(ny-1)-N)*dy & y(t_rec(:,4))<=(0.5*(ny-1)-N)*dy)==0 ...
           & (x(t_rec(:,1))>= -(0.5*(nx-1)-N)*dx & x(t_rec(:,2))<=(0.5*(nx-1)-N)*dx)==0,:);
end
```

PML1

```
function [row_pml1,col_pml1,val_pml1]=pml1(x,y,t_rec1,as,omega,c,eps0,er1,tand_1,mu0)

t_rec1r=t_rec1(x(t_rec1(:,1))>0,:);
x1r=x(t_rec1r(:,1)); y1r=y(t_rec1r(:,1));
x3r=x(t_rec1r(:,3)); y3r=y(t_rec1r(:,3));
[row_r_i,col_r_i,val_r_i] = xp(t_rec1r,x1r,x3r,y1r,y3r,as,omega,c,eps0,er1,tand_1,mu0);

t_rec1l=t_rec1(x(t_rec1(:,1))<0,:);
x1l=x(t_rec1l(:,1)); y1l=y(t_rec1l(:,1));
x3l=x(t_rec1l(:,3)); y3l=y(t_rec1l(:,3));
[row_l_i,col_l_i,val_l_i] = xn(t_rec1l,x1l,x3l,y1l,y3l,as,omega,c,eps0,er1,tand_1,mu0);

row_pml1=[row_l_i;row_r_i];
col_pml1=[col_l_i;col_r_i];
val_pml1=[val_l_i;val_r_i];

end
```

PML2

```
function [row_pml2,col_pml2,val_pml2]=pml2(x,y,t_rec2,bs,omega,c,eps0,er1,tand_1,mu0)

t_rec2t=t_rec2(y(t_rec2(:,1))>0,:);
x1t=x(t_rec2t(:,1)); y1t=y(t_rec2t(:,1));
x3t=x(t_rec2t(:,3)); y3t=y(t_rec2t(:,3));
[row_t_i,col_t_i,val_t_i] = yp(t_rec2t,x1t,x3t,y1t,y3t,bs,omega,c,eps0,mu0);

t_rec2b=t_rec2(y(t_rec2(:,1))<0,:);
```

```
x1b=x(t_rec2b(:,1)); y1b=y(t_rec2b(:,1));
x3b=x(t_rec2b(:,3)); y3b=y(t_rec2b(:,3));
[row_b_i,col_b_i,val_b_i] = yn(t_rec2b,x1b,x3b,y1b,y3b,bs,omega,c,eps0,er1,tand_1,mu0);

row_pml2=[row_b_i;row_t_i];
col_pml2=[col_b_i;col_t_i];
val_pml2=[val_b_i;val_t_i];

end
```

PML3

```
function [row_pml3,col_pml3,val_pml3]=pml3(x,y,t_rec3,as,bs,omega,c,eps0,er1,tand_1,mu0)

t_c1=t_rec3(x(t_rec3(:,1))<0 & y(t_rec3(:,1))<0,:);
x1_c1=x(t_c1(:,1)); x3_c1=x(t_c1(:,3));
y1_c1=y(t_c1(:,1)); y3_c1=y(t_c1(:,3));
[row_c1,col_c1,val_c1]=c1(t_c1,x1_c1,x3_c1,y1_c1,y3_c1,as,bs,omega,c,eps0,er1,tand_1,mu0);

t_c2=t_rec3(x(t_rec3(:,1))>0 & y(t_rec3(:,1))<0,:);
x1_c2=x(t_c2(:,1)); x3_c2=x(t_c2(:,3));
y1_c2=y(t_c2(:,1)); y3_c2=y(t_c2(:,3));
[row_c2,col_c2,val_c2]=c2(t_c2,x1_c2,x3_c2,y1_c2,y3_c2,as,bs,omega,c,eps0,er1,tand_1,mu0);

t_c3=t_rec3(x(t_rec3(:,1))>0 & y(t_rec3(:,1))>0,:);
x1_c3=x(t_c3(:,1)); x3_c3=x(t_c3(:,3));
y1_c3=y(t_c3(:,1)); y3_c3=y(t_c3(:,3));
[row_c3,col_c3,val_c3]=c3(t_c3,x1_c3,x3_c3,y1_c3,y3_c3,as,bs,omega,c,eps0,mu0);

t_c4=t_rec3(x(t_rec3(:,1))<0 & y(t_rec3(:,1))>0,:);
x1_c4=x(t_c4(:,1)); x3_c4=x(t_c4(:,3));
y1_c4=y(t_c4(:,1)); y3_c4=y(t_c4(:,3));
[row_c4,col_c4,val_c4]=c4(t_c4,x1_c4,x3_c4,y1_c4,y3_c4,as,bs,omega,c,eps0,mu0);

row_pml3=[row_c1;row_c2;row_c3;row_c4];
col_pml3=[col_c1;col_c2;col_c3;col_c4];
val_pml3=[val_c1;val_c2;val_c3;val_c4];
end
```

xp

```
function [row,col,val] = xp(t_rec1r,x1r,x3r,y1r,y3r,as,omega,c,eps0,er1,tand_1,mu0)

%internal elements
t_rec1r_int=t_rec1r(x3r~=as,:);
x1r_int=x1r(x3r~=as); x3r_int=x3r(x3r~=as);
y1r_int=y1r(x3r~=as); y3r_int=y3r(x3r~=as);

% calling function
[row_int,col_int,val_int]=xp_non_boundary(t_rec1r_int,...
    x1r_int,x3r_int,...
    y1r_int,y3r_int,...
    as,...
 omega,c,eps0,er1,tand_1,mu0);

%boundary elements
t_rec1r_bnd=t_rec1r(x3r==as,:);
x1r_bnd=x1r(x3r==as); x3r_bnd=x3r(x3r==as);
y1r_bnd=y1r(x3r==as); y3r_bnd=y3r(x3r==as);
% calling function
[row_bnd,col_bnd,val_bnd]=xp_boundary(t_rec1r_bnd,...
    x1r_bnd,x3r_bnd,...
    y1r_bnd,y3r_bnd,...
    as,...
 omega,c,eps0,er1,tand_1,mu0);

row=[row_int;row_bnd];
col=[col_int;col_bnd];
val=[val_int;val_bnd];
end

function[row,col,val]=xp_non_boundary(t_rec1r_int,...
    x1r_int,x3r_int,...
    y1r_int,y3r_int,...
    as,...
 omega,c,eps0,er1,tand_1,mu0)

hx=x3r_int-x1r_int;
hy=y3r_int-y1r_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(1.5.*x1r_int-0.5.*x3r_int-as)+(as-x1r_int).^2.*log((as-x1r_int)./(as-x3r_int));
s33=hx.*(1.5.*x3r_int-0.5.*x1r_int-as)+(as-x3r_int).^2.*log((as-x1r_int)./(as-x3r_int));
s13=0.5.*hx.*(x1r_int+x3r_int-2.*as)
        +(as-x1r_int).*(as-x3r_int).*log((as-x1r_int)./(as-x3r_int));

m1=1./(hx.^2.*hy.^2);

m2=hx+-1i*c/omega.*log((omega.*(as-x3r_int)+-1i*c)./(omega.*(as-x1r_int)+-1i*c));

eps_xp_pml=zeros(size(t_rec1r_int,1),1);
for i=1:0.5*size(t_rec1r_int,1)
    eps_xp_pml(i)=eps0*er1*(1-1j*tand_1);
end
for i=0.5*size(t_rec1r_int,1)+1:size(t_rec1r_int,1)
    eps_xp_pml(i)=eps0;
```

```
end

kk(1,1,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*(p33.*q33+-1i*c/omega.*s33.*q33));
kk(1,2,:)=m1.*(1/mu0.*m2.*-q33...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*-(p13.*q33+-1i*c/omega.*s13.*q33));
kk(1,3,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(1,4,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));

kk(2,1,:)=m1.*(1/mu0.*m2.*-q33...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*-(p13.*q33+-1i*c/omega.*s13.*q33));
kk(2,2,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xp_pml.*(p11.*q33+-1i*c/omega.*s11.*q33));
kk(2,3,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xp_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(2,4,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));

kk(3,1,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(3,2,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xp_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(3,3,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xp_pml.*(p11.*q11+-1i*c/omega.*s11.*q11));
kk(3,4,:)=m1.*(1/mu0.*m2.*-q11...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*-(p13.*q11+-1i*c/omega.*s13.*q11));

kk(4,1,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));
kk(4,2,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(4,3,:)=m1.*(1/mu0.*m2.*-q11...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xp_pml.*-(p13.*q11+-1i*c/omega.*s13.*q11));
kk(4,4,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*(p33.*q11+-1i*c/omega.*s33.*q11));

[row,col,val]=filler(t_rec1r_int,kk);

end

function[row,col,val]=xp_boundary(t_rec1r_bnd,...
    x1r_bnd,x3r_bnd,...
    y1r_bnd,y3r_bnd,...
    as,...
 omega,c,eps0,er1,tand_1,mu0) % n1 and n4 will exist

hx=x3r_bnd-x1r_bnd;
hy=y3r_bnd-y1r_bnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s33=0.5.*hx.^2;
s13=0.5.*hx.*(x1r_bnd+x3r_bnd-2.*as);

m1=1./(hx.^2.*hy.^2);
m2=hx+-1i*c/omega.*log((omega.*(as-x3r_bnd)+-1i*c)./(omega.*(as-x1r_bnd)+-1i*c));

eps_xp_pml=zeros(size(t_rec1r_bnd,1),1);
for i=1:0.5*size(t_rec1r_bnd,1)
    eps_xp_pml(i)=eps0*er1*(1-1j*tand_1);
end
for i=0.5*size(t_rec1r_bnd,1)+1:size(t_rec1r_bnd,1)
    eps_xp_pml(i)=eps0;
end

kk(1,1,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*(p33.*q33+-1i*c/omega.*s33.*q33));
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=m1.*(1/mu0.*m2.*-q33...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xp_pml.*(p33.*q11+-1i*c/omega.*s33.*q11));

[row,col,val]=filler(t_rec1r_bnd,kk);

end
```

xn

```
function [row,col,val] = xn(t_rec1l,x1l,x3l,y1l,y3l,as,omega,c,eps0,er1,tand_1,mu0)
%internal elements
t_rec1l_int=t_rec1l(x1l~=-as,:);
x1l_int=x1l(x1l~=-as); x3l_int=x3l(x1l~=-as);
y1l_int=y1l(x1l~=-as); y3l_int=y3l(x1l~=-as);

% calling function
[row_int,col_int,val_int]=xn_non_boundary(t_rec1l_int,...
    x1l_int,x3l_int,...
    y1l_int,y3l_int,...
    as,...
 omega,c,eps0,er1,tand_1,mu0);

%boundary elements
t_rec1l_bnd=t_rec1l(x1l==-as,:);
x1l_bnd=x1l(x1l==-as); x3l_bnd=x3l(x1l==-as);
y1l_bnd=y1l(x1l==-as); y3l_bnd=y3l(x1l==-as);
% calling function
[row_bnd,col_bnd,val_bnd]=xn_boundary(t_rec1l_bnd,...
    x1l_bnd,x3l_bnd,...
    y1l_bnd,y3l_bnd,...
    as,...
 omega,c,eps0,er1,tand_1,mu0);

row=[row_int;row_bnd];
col=[col_int;col_bnd];
val=[val_int;val_bnd];
end

function[row,col,val]=xn_non_boundary(t_rec1l_int,...
    x1l_int,x3l_int,...
    y1l_int,y3l_int,...
    as,...
 omega,c,eps0,er1,tand_1,mu0)

hx=x3l_int-x1l_int;
hy=y3l_int-y1l_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(-1.5.*x1l_int+0.5.*x3l_int-as)+(as+x1l_int).^2.*log((as+x3l_int)./(as+x1l_int));
s33=hx.*(-1.5.*x3l_int+0.5.*x1l_int-as)+(as+x3l_int).^2.*log((as+x3l_int)./(as+x1l_int));
s13=-0.5.*hx.*(x1l_int+x3l_int+2.*as)
        +(as+x1l_int).*(as+x3l_int).*log((as+x3l_int)./(as+x1l_int));

m1=1./(hx.^2.*hy.^2);
m2=hx--1i*c/omega.*log((omega.*(as+x3l_int)+-1i*c)./(omega.*(as+x1l_int)+-1i*c));

eps_xn_pml=zeros(size(t_rec1l_int,1),1);
for i=1:0.5*size(t_rec1l_int,1)
    eps_xn_pml(i)=eps0*er1*(1-1j*tand_1);
end
for i=0.5*size(t_rec1l_int,1)+1:size(t_rec1l_int,1)
    eps_xn_pml(i)=eps0;
end

kk(1,1,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xn_pml.*(p33.*q33+-1i*c/omega.*s33.*q33));
kk(1,2,:)=m1.*(1/mu0.*m2.*-q33...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*-(p13.*q33+-1i*c/omega.*s13.*q33));
kk(1,3,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(1,4,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xn_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));

kk(2,1,:)=m1.*(1/mu0.*m2.*-q33...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*-(p13.*q33+-1i*c/omega.*s13.*q33));
kk(2,2,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*(p11.*q33+-1i*c/omega.*s11.*q33));
kk(2,3,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(2,4,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));

kk(3,1,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(3,2,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(3,3,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*(p11.*q11+-1i*c/omega.*s11.*q11));
kk(3,4,:)=m1.*(1/mu0.*m2.*-q11...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*-(p13.*q11+-1i*c/omega.*s13.*q11));

kk(4,1,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xn_pml.*-(p33.*q13+-1i*c/omega.*s33.*q13));
kk(4,2,:)=m1.*(1/mu0.*m2.*q13...
              +1/mu0.*hy.*(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*(p13.*q13+-1i*c/omega.*s13.*q13));
kk(4,3,:)=m1.*(1/mu0.*m2.*-q11...
              +1/mu0.*hy.*-(p13+-1i*c/omega.*s13)...
              -omega^2.*eps_xn_pml.*-(p13.*q11+-1i*c/omega.*s13.*q11));
kk(4,4,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p33+-1i*c/omega.*s33)...
              -omega^2.*eps_xn_pml.*(p33.*q11+-1i*c/omega.*s33.*q11));

[row,col,val]=filler(t_rec1l_int,kk);

end

function[row,col,val]=xn_boundary(t_rec1l_bnd,...
    x1l_bnd,x3l_bnd,...
    y1l_bnd,y3l_bnd,...
    as,...
 omega,c,eps0,er1,tand_1,mu0) % only n2 and n3 exists

hx=x3l_bnd-x1l_bnd;
hy=y3l_bnd-y1l_bnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=0.5.*hx.^2;
s13=-0.5.*hx.*(x1l_bnd+x3l_bnd+2.*as);
```

```
m1=1./(hx.^2.*hy.^2);
m2=hx--1i*c/omega.*log((omega.*(as+x3l_bnd)+-1i*c)./(omega.*(as+x1l_bnd)+-1i*c));

eps_xn_pml=zeros(size(t_rec1l_bnd,1),1);
for i=1:0.5*size(t_rec1l_bnd,1)
    eps_xn_pml(i)=eps0*er1*(1-1j*tand_1);
end
for i=0.5*size(t_rec1l_bnd,1)+1:size(t_rec1l_bnd,1)
    eps_xn_pml(i)=eps0;
end

kk(2,2,:)=m1.*(1/mu0.*m2.*q33...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*(p11.*q33+-1i*c/omega.*s11.*q33));



kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;

kk(2,3,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=m1.*(1/mu0.*m2.*-q13...
              +1/mu0.*hy.*-(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*-(p11.*q13+-1i*c/omega.*s11.*q13));
kk(3,3,:)=m1.*(1/mu0.*m2.*q11...
              +1/mu0.*hy.*(p11+-1i*c/omega.*s11)...
              -omega^2.*eps_xn_pml.*(p11.*q11+-1i*c/omega.*s11.*q11));
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row,col,val]=filler(t_rec1l_bnd,kk);
end
```

yp

```
function [row,col,val] = yp(t_rec2t,x1t,x3t,y1t,y3t,bs,omega,c,eps0,mu0)

%internal elements
t_rec2t_int=t_rec2t(y3t~=bs,:);
```

```
x1t_int=x1t(y3t~=bs); x3t_int=x3t(y3t~=bs);
y1t_int=y1t(y3t~=bs); y3t_int=y3t(y3t~=bs);

% calling function
[row_int,col_int,val_int]=yp_non_boundary(t_rec2t_int,...
    x1t_int,x3t_int,...
    y1t_int,y3t_int,...
    bs,...
 omega,c,eps0,mu0);

%boundary elements
t_rec2t_bnd=t_rec2t(y3t==bs,:);
x1t_bnd=x1t(y3t==bs); x3t_bnd=x3t(y3t==bs);
y1t_bnd=y1t(y3t==bs); y3t_bnd=y3t(y3t==bs);
% calling function
[row_bnd,col_bnd,val_bnd]=yp_boundary(t_rec2t_bnd,...
    x1t_bnd,x3t_bnd,...
    y1t_bnd,y3t_bnd,...
    bs,...
 omega,c,eps0,mu0);

row=[row_int;row_bnd];
col=[col_int;col_bnd];
val=[val_int;val_bnd];

end

function[row,col,val]=yp_non_boundary(t_rec2t_int,...
    x1t_int,x3t_int,...
    y1t_int,y3t_int,...
    bs,...
 omega,c,eps0,mu0)

hx=x3t_int-x1t_int;
hy=y3t_int-y1t_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(1.5.*y1t_int-0.5.*y3t_int-bs)+(bs-y1t_int).^2.*log((bs-y1t_int)./(bs-y3t_int));
t33=hy.*(1.5.*y3t_int-0.5.*y1t_int-bs)+(bs-y3t_int).^2.*log((bs-y1t_int)./(bs-y3t_int));
t13=0.5.*hy.*(y1t_int+y3t_int-2.*bs)
        +(bs-y1t_int).*(bs-y3t_int).*log((bs-y1t_int)./(bs-y3t_int));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

m1=1./(hx.^2.*hy.^2);
m2=hy+-1i*c/omega.*log((omega.*(bs-y3t_int)+-1i*c)./(omega.*(bs-y1t_int)+-1i*c));
```

```
kk(1,1,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p33...
              -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*p33.*t33));
kk(1,2,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(1,3,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(1,4,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p33...
              -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*p33.*t13));

kk(2,1,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(2,2,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p11...
              -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*p11.*t33));
kk(2,3,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p11...
              -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*p11.*t13));
kk(2,4,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*p13.*t13));

kk(3,1,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(3,2,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p11...
              -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*p11.*t13));
kk(3,3,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p11...
              -omega.^2.*eps0.*(p11.*q11+-1i*c/omega.*p11.*t11));
kk(3,4,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*p13.*t11));

kk(4,1,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p33...
              -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*p33.*t13));
kk(4,2,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(4,3,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*p13.*t11));
kk(4,4,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p33...
              -omega.^2.*eps0.*(p33.*q11+-1i*c/omega.*p33.*t11));
```

```
[row,col,val]=filler(t_rec2t_int,kk);

end

function[row,col,val]=yp_boundary(t_rec2t_bnd,...
    x1t_bnd,x3t_bnd,...
    y1t_bnd,y3t_bnd,...
    bs,...
 omega,c,eps0,mu0)% n1 and n2 will exists

hx=x3t_bnd-x1t_bnd;
hy=y3t_bnd-y1t_bnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t33=0.5.*hy.^2;
t13=0.5.*hy.*(y1t_bnd+y3t_bnd-2.*bs);

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

m1=1./(hx.^2.*hy.^2);
m2=hy+-1i*c/omega.*log((omega.*(bs-y3t_bnd)+-1i*c)./(omega.*(bs-y1t_bnd)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p33...
              -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*p33.*t33));
kk(1,2,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(2,2,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p11...
              -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*p11.*t33));
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;


[row,col,val]=filler(t_rec2t_bnd,kk);

end
```

yn

```
function [row,col,val] = yn(t_rec2b,x1b,x3b,y1b,y3b,bs,omega,c,eps0,er1,tand_1,mu0)

%internal elements
t_rec2b_int=t_rec2b(y1b~=-bs,:);
x1b_int=x1b(y1b~=-bs); x3b_int=x3b(y1b~=-bs);
y1b_int=y1b(y1b~=-bs); y3b_int=y3b(y1b~=-bs);


% calling function
[row_int,col_int,val_int]=yn_non_boundary(t_rec2b_int,...
    x1b_int,x3b_int,...
    y1b_int,y3b_int,...
    bs,...
 omega,c,eps0,er1,tand_1,mu0);

%boundary elements
t_rec2b_bnd=t_rec2b(y1b==-bs,:);
x1b_bnd=x1b(y1b==-bs); x3b_bnd=x3b(y1b==-bs);
y1b_bnd=y1b(y1b==-bs); y3b_bnd=y3b(y1b==-bs);
% calling function
[row_bnd,col_bnd,val_bnd]=yn_boundary(t_rec2b_bnd,...
    x1b_bnd,x3b_bnd,...
    y1b_bnd,y3b_bnd,...
    bs,...
 omega,c,eps0,er1,tand_1,mu0);

row=[row_int;row_bnd];
col=[col_int;col_bnd];
val=[val_int;val_bnd];

end

function[row,col,val]=yn_non_boundary(t_rec2b_int,...
    x1b_int,x3b_int,...
    y1b_int,y3b_int,...
    bs,...
 omega,c,eps0,er1,tand_1,mu0)

hx=x3b_int-x1b_int;
hy=y3b_int-y1b_int;

q11=hy.^3./3;
```

```
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(-1.5.*y1b_int+0.5.*y3b_int-bs)+(bs+y1b_int).^2.*log((bs+y3b_int)./(bs+y1b_int));
t33=hy.*(-1.5.*y3b_int+0.5.*y1b_int-bs)+(bs+y3b_int).^2.*log((bs+y3b_int)./(bs+y1b_int));
t13=-0.5.*hy.*(y1b_int+y3b_int+2.*bs)
        +(bs+y1b_int).*(bs+y3b_int).*log((bs+y3b_int)./(bs+y1b_int));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;


m1=1./(hx.^2.*hy.^2);
m2=hy--1i*c/omega.*log((omega.*(bs+y3b_int)+-1i*c)./(omega.*(bs+y1b_int)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p33...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p33.*q33+-1i*c/omega.*p33.*t33));
kk(1,2,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(1,3,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(1,4,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p33...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p33.*q13+-1i*c/omega.*p33.*t13));

kk(2,1,:)=m1.*(1/mu0.*hx.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q33+-1i*c/omega.*p13.*t33));
kk(2,2,:)=m1.*(1/mu0.*hx.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2.*p11...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p11.*q33+-1i*c/omega.*p11.*t33));
kk(2,3,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p11...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p11.*q13+-1i*c/omega.*p11.*t13));
kk(2,4,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p13.*q13+-1i*c/omega.*p13.*t13));

kk(3,1,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(3,2,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p11...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p11.*q13+-1i*c/omega.*p11.*t13));
kk(3,3,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p11...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p11.*q11+-1i*c/omega.*p11.*t11));
kk(3,4,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q11+-1i*c/omega.*p13.*t11));

kk(4,1,:)=m1.*(1/mu0.*hx.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*-p33...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p33.*q13+-1i*c/omega.*p33.*t13));
kk(4,2,:)=m1.*(1/mu0.*hx.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2.*p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p13.*q13+-1i*c/omega.*p13.*t13));
kk(4,3,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q11+-1i*c/omega.*p13.*t11));
kk(4,4,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p33...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p33.*q11+-1i*c/omega.*p33.*t11));

[row,col,val]=filler(t_rec2b_int,kk);

end

function[row,col,val]=yn_boundary(t_rec2b_bnd,...
    x1b_bnd,x3b_bnd,...
    y1b_bnd,y3b_bnd,...
    bs,...
 omega,c,eps0,er1,tand_1,mu0)% n3 and n4 will exists

hx=x3b_bnd-x1b_bnd;
hy=y3b_bnd-y1b_bnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=0.5.*hy.^2;
t13=-0.5.*hy.*(y1b_bnd+y3b_bnd+2.*bs);

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

m1=1./(hx.^2.*hy.^2);
m2=hy--1i*c/omega.*log((omega.*(bs+y3b_bnd)+-1i*c)./(omega.*(bs+y1b_bnd)+-1i*c));

kk(3,3,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p11...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p11.*q11+-1i*c/omega.*p11.*t11));
kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;

kk(3,4,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q11+-1i*c/omega.*p13.*t11));

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=m1.*(1/mu0.*hx.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*-p13...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*-(p13.*q11+-1i*c/omega.*p13.*t11));
kk(4,4,:)=m1.*(1/mu0.*hx.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2.*p33...
              -omega.^2.*(eps0*er1*(1-1j*tand_1)).*(p33.*q11+-1i*c/omega.*p33.*t11));

[row,col,val]=filler(t_rec2b_bnd,kk);

end
```

c1

```
function [row_c1,col_c1,val_c1]=c1(t_c1,x1_c1,x3_c1,y1_c1,y3_c1,...
        as,bs,omega,c,eps0,er1,tand_1,mu0)

%internal
c1_int=t_c1(x1_c1~=-as & y1_c1~=-bs,:);
x1_int=x1_c1(x1_c1~=-as & y1_c1~=-bs); x3_int=x3_c1(x1_c1~=-as & y1_c1~=-bs);
y1_int=y1_c1(x1_c1~=-as & y1_c1~=-bs); y3_int=y3_c1(x1_c1~=-as & y1_c1~=-bs);
[row_int,col_int,val_int]=c1_non_boundary(c1_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%x boundary
c1_xbnd=t_c1(x1_c1==-as & y1_c1~=-bs,:);
x1_xbnd=x1_c1(x1_c1==-as & y1_c1~=-bs); x3_xbnd=x3_c1(x1_c1==-as & y1_c1~=-bs);
y1_xbnd=y1_c1(x1_c1==-as & y1_c1~=-bs); y3_xbnd=y3_c1(x1_c1==-as & y1_c1~=-bs);
[row_xbnd,col_xbnd,val_xbnd]=c1_x_boundary(c1_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%y boundary
c1_ybnd=t_c1(x1_c1~=-as & y1_c1==-bs,:);
x1_ybnd=x1_c1(x1_c1~=-as & y1_c1==-bs); x3_ybnd=x3_c1(x1_c1~=-as & y1_c1==-bs);
```

```
y1_ybnd=y1_c1(x1_c1~=-as & y1_c1==-bs); y3_ybnd=y3_c1(x1_c1~=-as & y1_c1==-bs);
[row_ybnd,col_ybnd,val_ybnd]=c1_y_boundary(c1_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%xy boundary
c1_xybnd=t_c1(x1_c1==-as & y1_c1==-bs,:);
x1_xybnd=x1_c1(x1_c1==-as & y1_c1==-bs); x3_xybnd=x3_c1(x1_c1==-as & y1_c1==-bs);
y1_xybnd=y1_c1(x1_c1==-as & y1_c1==-bs); y3_xybnd=y3_c1(x1_c1==-as & y1_c1==-bs);
[row_xybnd,col_xybnd,val_xybnd]=c1_xy_boundary(c1_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

row_c1=[row_int;row_xbnd;row_ybnd;row_xybnd];
col_c1=[col_int;col_xbnd;col_ybnd;col_xybnd];
val_c1=[val_int;val_xbnd;val_ybnd;val_xybnd];

end

function [row_int,col_int,val_int]=c1_non_boundary(c1_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0)
hx=x3_int-x1_int;
hy=y3_int-y1_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(-1.5.*y1_int+0.5.*y3_int-bs)+(bs+y1_int).^2.*log((bs+y3_int)./(bs+y1_int));
t33=hy.*(-1.5.*y3_int+0.5.*y1_int-bs)+(bs+y3_int).^2.*log((bs+y3_int)./(bs+y1_int));
t13=-0.5.*hy.*(y1_int+y3_int+2.*bs)
        +(bs+y1_int).*(bs+y3_int).*log((bs+y3_int)./(bs+y1_int));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(-1.5.*x1_int+0.5.*x3_int-as)+(as+x1_int).^2.*log((as+x3_int)./(as+x1_int));
s33=hx.*(-1.5.*x3_int+0.5.*x1_int-as)+(as+x3_int).^2.*log((as+x3_int)./(as+x1_int));
s13=-0.5.*hx.*(x1_int+x3_int+2.*as)
        +(as+x1_int).*(as+x3_int).*log((as+x3_int)./(as+x1_int));

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_int)+-1i*c)./(omega.*(as+x1_int)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_int)+-1i*c)./(omega.*(bs+y1_int)+-1i*c));
```

```
eps=eps0.*er1*(1-1j.*tand_1);

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q33+-1i*c/omega.*s33.*q33...
                                +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q33+-1i*c/omega.*s13.*q33
                                +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(1,3,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                                +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                                +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));

kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q33+-1i*c/omega.*s13.*q33
                                +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q33+-1i*c/omega.*s11.*q33
                                +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                                +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(2,4,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                                +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));

kk(3,1,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                                +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                                +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                                +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                                +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                                +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,2,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                                +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                                +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                                +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

[row_int,col_int,val_int]=filler(c1_int,kk);


end

function [row_xbnd,col_xbnd,val_xbnd]=c1_x_boundary(c1_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) % only n2 and n3 exists

hx=x3_xbnd-x1_xbnd;
hy=y3_xbnd-y1_xbnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(-1.5.*y1_xbnd+0.5.*y3_xbnd-bs)+(bs+y1_xbnd).^2.*log((bs+y3_xbnd)./(bs+y1_xbnd));
t33=hy.*(-1.5.*y3_xbnd+0.5.*y1_xbnd-bs)+(bs+y3_xbnd).^2.*log((bs+y3_xbnd)./(bs+y1_xbnd));
t13=-0.5.*hy.*(y1_xbnd+y3_xbnd+2.*bs)
                +(bs+y1_xbnd).*(bs+y3_xbnd).*log((bs+y3_xbnd)./(bs+y1_xbnd));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=0.5.*hx.^2;
s13=-0.5.*hx.*(x1_xbnd+x3_xbnd+2.*as);

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_xbnd)+-1i*c)./(omega.*(as+x1_xbnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_xbnd)+-1i*c)./(omega.*(bs+y1_xbnd)+-1i*c));
```

```
eps=eps0*er1*(1-1j*tand_1);

kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q33+-1i*c/omega.*s11.*q33
                                +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                                +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));

kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row_xbnd,col_xbnd,val_xbnd]=filler(c1_xbnd,kk);
end

function [row_ybnd,col_ybnd,val_ybnd]=c1_y_boundary(c1_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) %only n3 and n4 exist
hx=x3_ybnd-x1_ybnd;
hy=y3_ybnd-y1_ybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=0.5.*hy.^2;
```

```
t13=-0.5.*hy.*(y1_ybnd+y3_ybnd+2.*bs);

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(-1.5.*x1_ybnd+0.5.*x3_ybnd-as)+(as+x1_ybnd).^2.*log((as+x3_ybnd)./(as+x1_ybnd));
s33=hx.*(-1.5.*x3_ybnd+0.5.*x1_ybnd-as)+(as+x3_ybnd).^2.*log((as+x3_ybnd)./(as+x1_ybnd));
s13=-0.5.*hx.*(x1_ybnd+x3_ybnd+2.*as)
                +(as+x1_ybnd).*(as+x3_ybnd).*log((as+x3_ybnd)./(as+x1_ybnd));

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_ybnd)+-1i*c)./(omega.*(as+x1_ybnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_ybnd)+-1i*c)./(omega.*(bs+y1_ybnd)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
                -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                                +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
                -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                                +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
                -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                                +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
                -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                                +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;

[row_ybnd,col_ybnd,val_ybnd]=filler(c1_ybnd,kk);
end
```

```
function [row_xybnd,col_xybnd,val_xybnd]=c1_xy_boundary(c1_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) %only n3 exists
hx=x3_xybnd-x1_xybnd;
hy=y3_xybnd-y1_xybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=0.5.*hx.^2;

t13=-0.5.*hy.*(y1_xybnd+y3_xybnd+2.*bs);

s13=-0.5.*hx.*(x1_xybnd+x3_xybnd+2.*as);

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_xybnd)+-1i*c)./(omega.*(as+x1_xybnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_xybnd)+-1i*c)./(omega.*(bs+y1_xybnd)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);

kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row_xybnd,col_xybnd,val_xybnd]=filler(c1_xybnd,kk);
end
```

c2

```
function [row_c2,col_c2,val_c2]=c2(t_c2,x1_c2,x3_c2,y1_c2,y3_c2,...
                              as,bs,omega,c,eps0,er1,tand_1,mu0)

%internal
c2_int=t_c2(x3_c2~=as & y1_c2~=-bs,:);
x1_int=x1_c2(x3_c2~=as & y1_c2~=-bs); x3_int=x3_c2(x3_c2~=as & y1_c2~=-bs);
y1_int=y1_c2(x3_c2~=as & y1_c2~=-bs); y3_int=y3_c2(x3_c2~=as & y1_c2~=-bs);
[row_int,col_int,val_int]=c2_non_boundary(c2_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%x boundary
c2_xbnd=t_c2(x3_c2==as & y1_c2~=-bs,:);
x1_xbnd=x1_c2(x3_c2==as & y1_c2~=-bs); x3_xbnd=x3_c2(x3_c2==as & y1_c2~=-bs);
y1_xbnd=y1_c2(x3_c2==as & y1_c2~=-bs); y3_xbnd=y3_c2(x3_c2==as & y1_c2~=-bs);
[row_xbnd,col_xbnd,val_xbnd]=c2_x_boundary(c2_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%y boundary
c2_ybnd=t_c2(x3_c2~=as & y1_c2==-bs,:);
x1_ybnd=x1_c2(x3_c2~=as & y1_c2==-bs); x3_ybnd=x3_c2(x3_c2~=as & y1_c2==-bs);
y1_ybnd=y1_c2(x3_c2~=as & y1_c2==-bs); y3_ybnd=y3_c2(x3_c2~=as & y1_c2==-bs);
[row_ybnd,col_ybnd,val_ybnd]=c2_y_boundary(c2_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

%xy boundary
c2_xybnd=t_c2(x3_c2==as & y1_c2==-bs,:);
x1_xybnd=x1_c2(x3_c2==as & y1_c2==-bs); x3_xybnd=x3_c2(x3_c2==as & y1_c2==-bs);
y1_xybnd=y1_c2(x3_c2==as & y1_c2==-bs); y3_xybnd=y3_c2(x3_c2==as & y1_c2==-bs);
[row_xybnd,col_xybnd,val_xybnd]=c2_xy_boundary(c2_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0);

row_c2=[row_int;row_xbnd;row_ybnd;row_xybnd];
```

```
col_c2=[col_int;col_xbnd;col_ybnd;col_xybnd];
val_c2=[val_int;val_xbnd;val_ybnd;val_xybnd];

end

function [row_int,col_int,val_int]=c2_non_boundary(c2_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0)
hx=x3_int-x1_int;
hy=y3_int-y1_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(-1.5.*y1_int+0.5.*y3_int-bs)+(bs+y1_int).^2.*log((bs+y3_int)./(bs+y1_int));
t33=hy.*(-1.5.*y3_int+0.5.*y1_int-bs)+(bs+y3_int).^2.*log((bs+y3_int)./(bs+y1_int));
t13=-0.5.*hy.*(y1_int+y3_int+2.*bs)
                +(bs+y1_int).*(bs+y3_int).*log((bs+y3_int)./(bs+y1_int));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(1.5.*x1_int-0.5.*x3_int-as)+(as-x1_int).^2.*log((as-x1_int)./(as-x3_int));
s33=hx.*(1.5.*x3_int-0.5.*x1_int-as)+(as-x3_int).^2.*log((as-x1_int)./(as-x3_int));
s13=0.5.*hx.*(x1_int+x3_int-2.*as)
                +(as-x1_int).*(as-x3_int).*log((as-x1_int)./(as-x3_int));

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_int)+-1i*c)./(omega.*(as-x1_int)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_int)+-1i*c)./(omega.*(bs+y1_int)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(1,3,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));

kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q33+-1i*c/omega.*s11.*q33
                        +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(2,4,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));

kk(3,1,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,2,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                        +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));
```

```
[row_int,col_int,val_int]=filler(c2_int,kk);
end

function [row_xbnd,col_xbnd,val_xbnd]=c2_x_boundary(c2_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) % only n1 and n4 exists

hx=x3_xbnd-x1_xbnd;
hy=y3_xbnd-y1_xbnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(-1.5.*y1_xbnd+0.5.*y3_xbnd-bs)+(bs+y1_xbnd).^2.*log((bs+y3_xbnd)./(bs+y1_xbnd));
t33=hy.*(-1.5.*y3_xbnd+0.5.*y1_xbnd-bs)+(bs+y3_xbnd).^2.*log((bs+y3_xbnd)./(bs+y1_xbnd));
t13=-0.5.*hy.*(y1_xbnd+y3_xbnd+2.*bs)
        +(bs+y1_xbnd).*(bs+y3_xbnd).*log((bs+y3_xbnd)./(bs+y1_xbnd));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s33=0.5.*hx.^2;
s13=0.5.*hx.*(x1_xbnd+x3_xbnd-2.*as);

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_xbnd)+-1i*c)./(omega.*(as-x1_xbnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_xbnd)+-1i*c)./(omega.*(bs+y1_xbnd)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                        +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));
kk(1,2,:)=0;
kk(1,3,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,2,:)=0;
kk(4,3,:)=0;

[row_xbnd,col_xbnd,val_xbnd]=filler(c2_xbnd,kk);
end

function [row_ybnd,col_ybnd,val_ybnd]=c2_y_boundary(c2_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) %only n3 and n4 exist
hx=x3_ybnd-x1_ybnd;
hy=y3_ybnd-y1_ybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=0.5.*hy.^2;
t13=-0.5.*hy.*(y1_ybnd+y3_ybnd+2.*bs);

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(1.5.*x1_ybnd-0.5.*x3_ybnd-as)+(as-x1_ybnd).^2.*log((as-x1_ybnd)./(as-x3_ybnd));
s33=hx.*(1.5.*x3_ybnd-0.5.*x1_ybnd-as)+(as-x3_ybnd).^2.*log((as-x1_ybnd)./(as-x3_ybnd));
s13=0.5.*hx.*(x1_ybnd+x3_ybnd-2.*as)
        +(as-x1_ybnd).*(as-x3_ybnd).*log((as-x1_ybnd)./(as-x3_ybnd));

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_ybnd)+-1i*c)./(omega.*(as-x1_ybnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_ybnd)+-1i*c)./(omega.*(bs+y1_ybnd)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                                  +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                                  +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps.*(p11.*q11+-1i*c/omega.*s11.*q11
                                  +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps.*-(p13.*q11+-1i*c/omega.*s13.*q11
                          +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;

[row_ybnd,col_ybnd,val_ybnd]=filler(c2_ybnd,kk);
end

function [row_xybnd,col_xybnd,val_xybnd]=c2_xy_boundary(c2_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,er1,tand_1,mu0) %only n4 exists
hx=x3_xybnd-x1_xybnd;
hy=y3_xybnd-y1_xybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s33=0.5.*hx.^2;
```

```
t13=-0.5.*hy.*(y1_xybnd+y3_xybnd+2.*bs);

s13=0.5.*hx.*(x1_xybnd+x3_xybnd-2.*as);

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_xybnd)+-1i*c)./(omega.*(as+x1_xybnd)+-1i*c));
m2y=hy--1i*c/omega.*log((omega.*(bs+y3_xybnd)+-1i*c)./(omega.*(bs+y1_xybnd)+-1i*c));

eps=eps0*er1*(1-1j*tand_1);

kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps.*(p33.*q11+-1i*c/omega.*s33.*q11
                                  +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;

[row_xybnd,col_xybnd,val_xybnd]=filler(c2_xybnd,kk);
end
```

c3

```
function [row_c3,col_c3,val_c3]=c3(t_c3,x1_c3,x3_c3,y1_c3,y3_c3,as,bs,...
                        omega,c,eps0,mu0)
%internal
c3_int=t_c3(x3_c3~=as & y3_c3~=bs,:);
x1_int=x1_c3(x3_c3~=as & y3_c3~=bs); x3_int=x3_c3(x3_c3~=as & y3_c3~=bs);
y1_int=y1_c3(x3_c3~=as & y3_c3~=bs); y3_int=y3_c3(x3_c3~=as & y3_c3~=bs);
[row_int,col_int,val_int]=c3_non_boundary(c3_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,mu0);

%x boundary
```

```
c3_xbnd=t_c3(x3_c3==as & y3_c3~=bs,:);
x1_xbnd=x1_c3(x3_c3==as & y3_c3~=bs); x3_xbnd=x3_c3(x3_c3==as & y3_c3~=bs);
y1_xbnd=y1_c3(x3_c3==as & y3_c3~=bs); y3_xbnd=y3_c3(x3_c3==as & y3_c3~=bs);
[row_xbnd,col_xbnd,val_xbnd]=c3_x_boundary(c3_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,mu0);

%y boundary
c3_ybnd=t_c3(x3_c3~=as & y3_c3==bs,:);
x1_ybnd=x1_c3(x3_c3~=as & y3_c3==bs); x3_ybnd=x3_c3(x3_c3~=as & y3_c3==bs);
y1_ybnd=y1_c3(x3_c3~=as & y3_c3==bs); y3_ybnd=y3_c3(x3_c3~=as & y3_c3==bs);
[row_ybnd,col_ybnd,val_ybnd]=c3_y_boundary(c3_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,mu0);

%xy boundary
c3_xybnd=t_c3(x3_c3==as & y3_c3==bs,:);
x1_xybnd=x1_c3(x3_c3==as & y3_c3==bs); x3_xybnd=x3_c3(x3_c3==as & y3_c3==bs);
y1_xybnd=y1_c3(x3_c3==as & y3_c3==bs); y3_xybnd=y3_c3(x3_c3==as & y3_c3==bs);
[row_xybnd,col_xybnd,val_xybnd]=c3_xy_boundary(c3_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,mu0);

row_c3=[row_int;row_xbnd;row_ybnd;row_xybnd];
col_c3=[col_int;col_xbnd;col_ybnd;col_xybnd];
val_c3=[val_int;val_xbnd;val_ybnd;val_xybnd];
end

function [row_int,col_int,val_int]=c3_non_boundary(c3_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,mu0)
hx=x3_int-x1_int;
hy=y3_int-y1_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(1.5.*y1_int-0.5.*y3_int-bs)+(bs-y1_int).^2.*log((bs-y1_int)./(bs-y3_int));
t33=hy.*(1.5.*y3_int-0.5.*y1_int-bs)+(bs-y3_int).^2.*log((bs-y1_int)./(bs-y3_int));
t13=0.5.*hy.*(y1_int+y3_int-2.*bs)
                +(bs-y1_int).*(bs-y3_int).*log((bs-y1_int)./(bs-y3_int));

p11=hx.^3./3;
```

```
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(1.5.*x1_int-0.5.*x3_int-as)+(as-x1_int).^2.*log((as-x1_int)./(as-x3_int));
s33=hx.*(1.5.*x3_int-0.5.*x1_int-as)+(as-x3_int).^2.*log((as-x1_int)./(as-x3_int));
s13=0.5.*hx.*(x1_int+x3_int-2.*as)
               +(as-x1_int).*(as-x3_int).*log((as-x1_int)./(as-x3_int));

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_int)+-1i*c)./(omega.*(as-x1_int)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_int)+-1i*c)./(omega.*(bs-y1_int)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(1,3,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));

kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                        +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(2,4,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));

kk(3,1,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q11+-1i*c/omega.*s11.*q11
                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,2,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q11+-1i*c/omega.*s33.*q11
                        +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

[row_int,col_int,val_int]=filler(c3_int,kk);
end

function [row_xbnd,col_xbnd,val_xbnd]=c3_x_boundary(c3_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,mu0)% n1 and n4 exist
hx=x3_xbnd-x1_xbnd;
hy=y3_xbnd-y1_xbnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(1.5.*y1_xbnd-0.5.*y3_xbnd-bs)+(bs-y1_xbnd).^2.*log((bs-y1_xbnd)./(bs-y3_xbnd));
t33=hy.*(1.5.*y3_xbnd-0.5.*y1_xbnd-bs)+(bs-y3_xbnd).^2.*log((bs-y1_xbnd)./(bs-y3_xbnd));
t13=0.5.*hy.*(y1_xbnd+y3_xbnd-2.*bs)
        +(bs-y1_xbnd).*(bs-y3_xbnd).*log((bs-y1_xbnd)./(bs-y3_xbnd));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;
```

```
s33=0.5.*hx.^2;
s13=0.5.*hx.*(x1_xbnd+x3_xbnd-2.*as);

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_xbnd)+-1i*c)./(omega.*(as-x1_xbnd)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_xbnd)+-1i*c)./(omega.*(bs-y1_xbnd)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));

kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));

kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q11+-1i*c/omega.*s33.*q11
                        +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));

kk(1,2,:)=0;
kk(1,3,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,2,:)=0;
kk(4,3,:)=0;

[row_xbnd,col_xbnd,val_xbnd]=filler(c3_xbnd,kk);
end

function [row_ybnd,col_ybnd,val_ybnd]=c3_y_boundary(c3_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,mu0)%n1 and n2 exist
hx=x3_ybnd-x1_ybnd;
hy=y3_ybnd-y1_ybnd;
```

```
q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t13=0.5.*hy.*(y1_ybnd+y3_ybnd-2.*bs);
t33=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(1.5.*x1_ybnd-0.5.*x3_ybnd-as)+(as-x1_ybnd).^2.*log((as-x1_ybnd)./(as-x3_ybnd));
s33=hx.*(1.5.*x3_ybnd-0.5.*x1_ybnd-as)+(as-x3_ybnd).^2.*log((as-x1_ybnd)./(as-x3_ybnd));
s13=0.5.*hx.*(x1_ybnd+x3_ybnd-2.*as)
               +(as-x1_ybnd).*(as-x3_ybnd).*log((as-x1_ybnd)./(as-x3_ybnd));

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_ybnd)+-1i*c)./(omega.*(as-x1_ybnd)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_ybnd)+-1i*c)./(omega.*(bs-y1_ybnd)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));

kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                        +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));

kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;


[row_ybnd,col_ybnd,val_ybnd]=filler(c3_ybnd,kk);
end

function [row_xybnd,col_xybnd,val_xybnd]=c3_xy_boundary(c3_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,mu0)% only n1 exists
hx=x3_xybnd-x1_xybnd;
hy=y3_xybnd-y1_xybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t13=0.5.*hy.*(y1_xybnd+y3_xybnd-2.*bs);
t33=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s13=0.5.*hx.*(x1_xybnd+x3_xybnd-2.*as);
s33=0.5.*hx.^2;

m1=1./(hx.^2.*hy.^2);
m2x=hx+-1i*c/omega.*log((omega.*(as-x3_xybnd)+-1i*c)./(omega.*(as-x1_xybnd)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_xybnd)+-1i*c)./(omega.*(bs-y1_xybnd)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
              -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=0;

kk(2,1,:)=0;
kk(2,2,:)=0;

kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;
kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row_xybnd,col_xybnd,val_xybnd]=filler(c3_xybnd,kk);
end
```

c4

```
\enfunction [row_c4,col_c4,val_c4]=c4(t_c4,x1_c4,x3_c4,y1_c4,y3_c4,as,bs,omega,c,eps0,mu0)
%internal
c4_int=t_c4(x1_c4~=-as & y3_c4~=bs,:);
x1_int=x1_c4(x1_c4~=-as & y3_c4~=bs); x3_int=x3_c4(x1_c4~=-as & y3_c4~=bs);
y1_int=y1_c4(x1_c4~=-as & y3_c4~=bs); y3_int=y3_c4(x1_c4~=-as & y3_c4~=bs);
[row_int,col_int,val_int]=c4_non_boundary(c4_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,mu0);

%x boundary
c4_xbnd=t_c4(x1_c4==-as & y3_c4~=bs,:);
x1_xbnd=x1_c4(x1_c4==-as & y3_c4~=bs); x3_xbnd=x3_c4(x1_c4==-as & y3_c4~=bs);
y1_xbnd=y1_c4(x1_c4==-as & y3_c4~=bs); y3_xbnd=y3_c4(x1_c4==-as & y3_c4~=bs);
[row_xbnd,col_xbnd,val_xbnd]=c4_x_boundary(c4_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,mu0);

%y boundary
c4_ybnd=t_c4(x1_c4~=-as & y3_c4==bs,:);
x1_ybnd=x1_c4(x1_c4~=-as & y3_c4==bs); x3_ybnd=x3_c4(x1_c4~=-as & y3_c4==bs);
y1_ybnd=y1_c4(x1_c4~=-as & y3_c4==bs); y3_ybnd=y3_c4(x1_c4~=-as & y3_c4==bs);
[row_ybnd,col_ybnd,val_ybnd]=c4_y_boundary(c4_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,mu0);

%xy boundary
c4_xybnd=t_c4(x1_c4==-as & y3_c4==bs,:);
x1_xybnd=x1_c4(x1_c4==-as & y3_c4==bs); x3_xybnd=x3_c4(x1_c4==-as & y3_c4==bs);
y1_xybnd=y1_c4(x1_c4==-as & y3_c4==bs); y3_xybnd=y3_c4(x1_c4==-as & y3_c4==bs);
[row_xybnd,col_xybnd,val_xybnd]=c4_xy_boundary(c4_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,mu0);
```

```
row_c4=[row_int;row_xbnd;row_ybnd;row_xybnd];
col_c4=[col_int;col_xbnd;col_ybnd;col_xybnd];
val_c4=[val_int;val_xbnd;val_ybnd;val_xybnd];
end

function [row_int,col_int,val_int]=c4_non_boundary(c4_int,...
    x1_int,x3_int,...
    y1_int,y3_int,...
    as,bs,...
 omega,c,eps0,mu0)
hx=x3_int-x1_int;
hy=y3_int-y1_int;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(1.5.*y1_int-0.5.*y3_int-bs)+(bs-y1_int).^2.*log((bs-y1_int)./(bs-y3_int));
t33=hy.*(1.5.*y3_int-0.5.*y1_int-bs)+(bs-y3_int).^2.*log((bs-y1_int)./(bs-y3_int));
t13=0.5.*hy.*(y1_int+y3_int-2.*bs)
                +(bs-y1_int).*(bs-y3_int).*log((bs-y1_int)./(bs-y3_int));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(-1.5.*x1_int+0.5.*x3_int-as)+(as+x1_int).^2.*log((as+x3_int)./(as+x1_int));
s33=hx.*(-1.5.*x3_int+0.5.*x1_int-as)+(as+x3_int).^2.*log((as+x3_int)./(as+x1_int));
s13=-0.5.*hx.*(x1_int+x3_int+2.*as)
                +(as+x1_int).*(as+x3_int).*log((as+x3_int)./(as+x1_int));

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_int)+-1i*c)./(omega.*(as+x1_int)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_int)+-1i*c)./(omega.*(bs-y1_int)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                                +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(1,3,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(1,4,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
```

```
kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                        +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(2,4,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));

kk(3,1,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q11+-1i*c/omega.*s11.*q11
                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));

kk(4,1,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*-(p33.*q13+-1i*c/omega.*s33.*q13
                        +-1i*c/omega.*p33.*t13-c^2/omega^2.*s33.*t13));
kk(4,2,:)=m1.*(1/mu0.*m2x.*(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*(p13.*q13+-1i*c/omega.*s13.*q13
                        +-1i*c/omega.*p13.*t13-c^2/omega^2.*s13.*t13));
kk(4,3,:)=m1.*(1/mu0.*m2x.*-(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q11+-1i*c/omega.*s13.*q11
                        +-1i*c/omega.*p13.*t11-c^2/omega^2.*s13.*t11));
kk(4,4,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q11+-1i*c/omega.*s33.*q11
                        +-1i*c/omega.*p33.*t11-c^2/omega^2.*s33.*t11));
```

```
[row_int,col_int,val_int]=filler(c4_int,kk);
end

function [row_xbnd,col_xbnd,val_xbnd]=c4_x_boundary(c4_xbnd,...
    x1_xbnd,x3_xbnd,...
    y1_xbnd,y3_xbnd,...
    as,bs,...
 omega,c,eps0,mu0) %only n2 and n3 exist
hx=x3_xbnd-x1_xbnd;
hy=y3_xbnd-y1_xbnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t11=hy.*(1.5.*y1_xbnd-0.5.*y3_xbnd-bs)+(bs-y1_xbnd).^2.*log((bs-y1_xbnd)./(bs-y3_xbnd));
t33=hy.*(1.5.*y3_xbnd-0.5.*y1_xbnd-bs)+(bs-y3_xbnd).^2.*log((bs-y1_xbnd)./(bs-y3_xbnd));
t13=0.5.*hy.*(y1_xbnd+y3_xbnd-2.*bs)
                +(bs-y1_xbnd).*(bs-y3_xbnd).*log((bs-y1_xbnd)./(bs-y3_xbnd));

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s13=-0.5.*hx.*(x1_xbnd+x3_xbnd+2.*as);
s11=0.5.*hx.^2;
m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_xbnd)+-1i*c)./(omega.*(as+x1_xbnd)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_xbnd)+-1i*c)./(omega.*(bs-y1_xbnd)+-1i*c));

kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                        +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));
kk(2,3,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=m1.*(1/mu0.*m2x.*-(q13+-1i*c/omega.*t13)...
              +1/mu0.*m2y.*-(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*-(p11.*q13+-1i*c/omega.*s11.*q13
                        +-1i*c/omega.*p11.*t13-c^2/omega^2.*s11.*t13));
```

```
kk(3,3,:)=m1.*(1/mu0.*m2x.*(q11+-1i*c/omega.*t11)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q11+-1i*c/omega.*s11.*q11
                        +-1i*c/omega.*p11.*t11-c^2/omega^2.*s11.*t11));
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;


[row_xbnd,col_xbnd,val_xbnd]=filler(c4_xbnd,kk);
end

function [row_ybnd,col_ybnd,val_ybnd]=c4_y_boundary(c4_ybnd,...
    x1_ybnd,x3_ybnd,...
    y1_ybnd,y3_ybnd,...
    as,bs,...
 omega,c,eps0,mu0)%n1 and n2 exist
hx=x3_ybnd-x1_ybnd;
hy=y3_ybnd-y1_ybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t13=0.5.*hy.*(y1_ybnd+y3_ybnd-2.*bs);
t33=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s11=hx.*(-1.5.*x1_ybnd+0.5.*x3_ybnd-as)+(as+x1_ybnd).^2.*log((as+x3_ybnd)./(as+x1_ybnd));
s33=hx.*(-1.5.*x3_ybnd+0.5.*x1_ybnd-as)+(as+x3_ybnd).^2.*log((as+x3_ybnd)./(as+x1_ybnd));
s13=-0.5.*hx.*(x1_ybnd+x3_ybnd+2.*as)
        +(as+x1_ybnd).*(as+x3_ybnd).*log((as+x3_ybnd)./(as+x1_ybnd));

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_ybnd)+-1i*c)./(omega.*(as+x1_ybnd)+-1i*c));
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_ybnd)+-1i*c)./(omega.*(bs-y1_ybnd)+-1i*c));

kk(1,1,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p33+-1i*c/omega.*s33)...
        -omega.^2.*eps0.*(p33.*q33+-1i*c/omega.*s33.*q33
                        +-1i*c/omega.*p33.*t33-c^2/omega^2.*s33.*t33));
kk(1,2,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                        +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
```

```
kk(2,1,:)=m1.*(1/mu0.*m2x.*-(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*-(p13+-1i*c/omega.*s13)...
        -omega.^2.*eps0.*-(p13.*q33+-1i*c/omega.*s13.*q33
                                +-1i*c/omega.*p13.*t33-c^2/omega^2.*s13.*t33));
kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
              +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                                +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));

kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row_ybnd,col_ybnd,val_ybnd]=filler(c4_ybnd,kk);
end

function [row_xybnd,col_xybnd,val_xybnd]=c4_xy_boundary(c4_xybnd,...
    x1_xybnd,x3_xybnd,...
    y1_xybnd,y3_xybnd,...
    as,bs,...
 omega,c,eps0,mu0) % only n2 exists
hx=x3_xybnd-x1_xybnd;
hy=y3_xybnd-y1_xybnd;

q11=hy.^3./3;
q33=hy.^3./3;
q13=-hy.^3./6;

t13=0.5.*hy.*(y1_xybnd+y3_xybnd-2.*bs);
t33=0.5.*hy.^2;

p11=hx.^3./3;
p33=hx.^3./3;
p13=-hx.^3./6;

s13=-0.5.*hx.*(x1_xybnd+x3_xybnd+2.*as);
s11=0.5.*hx.^2;

m1=1./(hx.^2.*hy.^2);
m2x=hx--1i*c/omega.*log((omega.*(as+x3_xybnd)+-1i*c)./(omega.*(as+x1_xybnd)+-1i*c));
```

```
m2y=hy+-1i*c/omega.*log((omega.*(bs-y3_xybnd)+-1i*c)./(omega.*(bs-y1_xybnd)+-1i*c));

kk(2,2,:)=m1.*(1/mu0.*m2x.*(q33+-1i*c/omega.*t33)...
             +1/mu0.*m2y.*(p11+-1i*c/omega.*s11)...
        -omega.^2.*eps0.*(p11.*q33+-1i*c/omega.*s11.*q33
                       +-1i*c/omega.*p11.*t33-c^2/omega^2.*s11.*t33));

kk(1,1,:)=0;
kk(1,2,:)=0;
kk(1,3,:)=0;
kk(1,4,:)=0;

kk(2,1,:)=0;
kk(2,3,:)=0;
kk(2,4,:)=0;

kk(3,1,:)=0;
kk(3,2,:)=0;
kk(3,3,:)=0;
kk(3,4,:)=0;

kk(4,1,:)=0;
kk(4,2,:)=0;
kk(4,3,:)=0;
kk(4,4,:)=0;

[row_xybnd,col_xybnd,val_xybnd]=filler(c4_xybnd,kk);
```

Bibliography

[1] Uduwawala U., Norgren M. and Fuks P., "A complete fdtd simulation of a real gpr antenna system operating above lossy and dispersive grounds.," *Progress In Electromagnetics Research*, vol. 50, pp. 209–229, 2005.

[2] Warren C. and Giannopoulos A., "Optimising models of commercial gpr antennas.," *Proceedings of the 5th International Workshop of Advanced Ground Penetrating Radar*, 2009.

[3] A. W. Morgenthaler and C. M. Rappaport, "Scattering from dielectric objects buried beneath randomly rough ground: Validating the semianalytic mode matching algorithm with 2-d fdfd," *IEEE Trans. Geosci. Remote Sens.*, vol. 39, pp. 2421–2428, Nov. 2001.

[4] Naren Naik, Rick Beatson, Jerry Eriksson and Elijah van Houten, "An implicit radial basis function based reconstruction approach to electromagnetic shape-tomography," *Inverse Problems, 25(2):025004*, February 2009.

[5] Naren Naik, Jerry Eriksson, Pieter de Groen and Hichem Sahli, "A nonlinear iterative reconstruction and analysis approach to shape-based approximate electromagnetic tomography," *IEEE Trans. Geosc. and Rem. Sens., 46(5)*, pp. 1558–1574, May 2008.

[6] Giovanni Angelo Meles, Stewart A. Greenhalgh, et. al., Stewart A. Greenhalgh and J. V. der Kruk, "Gpr full-waveform sensitivity and resolution analysis using an fdtd adjoint method," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, MAY 2012.

[7] J.-P. Bérenger, ""a perfectly matched layer for the absorption of electromagnetic waves," *J. Comput. Phys.*, vol. 114, p. 185–200, 1994.

[8] J.-P. Bérenger, "Perfectly matched layer for the fdtd solution of wave–structure interaction problems," *IEEE Trans. Antennas Propag.*, vol. 44, pp. 110–117, 1996.

[9] J.-P. Bérenger, "Three-dimensional perfectly matched layer for the absorption of electromagnetic waves," *Journal of Computational Physics*, vol. 114, pp. 363–379, 1996.

[10] F. Collino and P. Monk , "Optimizing the perfectly matched layer," *Comput. Methods Appl. Mech. Eng.*, vol. 164, pp. 157–171, 1998.

[11] S. Asvadurov, V. Druskin, et. al. , "On optimal finite-difference approximation of pml," *SIAM J. Numer. Anal.*, vol. 41, pp. 287–305, 2003.

[12] A. Bermúdez, "An optimal perfectly matched layer with unbounded absorbing function for time-harmonic acoustic scattering problems," *Journal of Computational Physics*, vol. 223, pp. 469–488, May 2007.

[13] H. M. Jol, *Ground Penetrating Radar Theory and Applications.* 2 ed., 2002.

[14] Johnson. S. G., "Notes on perfectly matched layers," *MIT lecture notes*, 2010.

[15] D. G. Daniel Rabinovich and E. Becache, "Comparison of high-order absorbing boundary conditions and perfectly matched layers in the frequency domain," *Int. J. Numer. Meth. Biomed. Engng.*, vol. 26, p. 1351–1369, 2010.

[16] Z. S. Sacks, "A perfectly matched anisotropic absorber for use as an absorbing boundary condition," *IEEE Trans. Antennas Propagat.*, vol. AP-43, pp. 1460–1463, Dec 1995.

[17] J. M. Jin, *The Finite Element Method in Electromagnetics.* 2 ed., 2002.

[18] F. Fedele, J.P. Laible and M.J. Eppstein, "Coupled complex adjoint sensitivities for frequency-domain fluorescence tomography: theory and vectorized implementation," *Journal of Computational Physics*, vol. 187, p. 597–619, 2003.

[19] S. R. Arridge, "Optical tomography in medical imaging," *Inverse Problems*, vol. 15, pp. R41–R93, 1999.

[20] W. C. Chew and W. H. Weedon, "A 3d perfectly matched medium from modified maxwell's equations with stretched coordinates," *Microwave Opt.Technol. Lett.*, vol. 7, pp. 599–604, 1994.

[21] Barbara Kaltenbachera, Manfred Kaltenbacherb and Imbo Sim, "A modified and stable version of a perfectly matched layer technique for the 3-d second order wave equation in time domain with an application to aeroacoustics," *Journal of Computational Physics*, vol. 235, p. 407–422, 15 February 2013.

[22] Seounghyun Ham and Klaus-Jürgen Bathe, “A finite element method enriched for wave propagation problems,” *Computers and Structures*, vol. 94–95, p. 1–12, March 2012.

[23] A. Bermúdez, Hervella Nieto, A. Prieto and R. Rodríguez, “Perfectly matched layers for time-harmonic second order elliptic problems,” *Archives of Computational Methods in Engineering*, vol. 17, pp. 77–107, March 2010.

[24] Garuda Fujii, Hayato Watanabe, et. al., “Level set based topology optimization for optical cloaks,” *Applied Pysics Letters*, vol. 102, 2013.