Skolkovo Institute of Science and Technology

In collaboration with Massachusetts Institute of Technology

Giannakopoulos Ilias
Sudakov Oleg
Korenev Artem
Email

Project:
"*Direct and Iterative Solvers for the Electric Field Integral Equations on perfect electric conductors*"

Professor: Oseldets Ivan

Skoltech

Skolkovo Institute of Science and Technology

Moscow 2016

CONTENTS

CHAPTER 1

PROJECT PROPOSAL

§ 1.1. Description

In the following project we choose to test some direct and iterative solvers of linear systems on electromagnetic problems. We will write the electric field integral equation for perfect electric conductors for some geometries and we will solve it with various solver, in order to test, which is of them is more accurate, more faster, and how much faster it can be if we minimize the accuracy.

We will start by giving a solid proof of the electric field integral equation and we will implement the method of moments on it, in order to create a linear system. This linear system, will be solved with a variety of solvers, both direct and iterative. We want to see which solver is the faster, which is more accurate, and if we can sacrifice some accuracy, for faster solutions.

This project is useful if someone is interested in solving big systems fast and accurate because it will show how good some basic implementations (for example SVD) work. Also it is the first step for the creation of a good, fast and accurate solver for electromagnetic problems, which have many applications nowadays such as MRI machines and X rays.

It is clear why this project is connected with Numerical Linear Algebra. One needs to know very well LU decomposition, QR factorization, SVD and other techniques in order to solve huge linear system, which always exist in computational electromagnetic problems. As we will see in the project, even the command \ of Matlab, uses some of the above decompositions, and we need to know why. Furthermore, this project will help us to understand the basics of numerical decompositions and the properties of matrices which are an essential first step for someone who wants to study computational sciences, like computational electromagnetism.

§ 1.2. Team

Our team consists of two Master students and one PhD student. The research of the PhD student is around surface integral equation for the moment, which is the idea for this project.

The PhD student will work on the formulation of the EFIE and will give the rest of the team members a simple exaplanation of the computational electromagnetic, Integral Equations and Method of moments. After that, all of the students will focus on solving a linear system of the form $\mathbf{A}\vec{x} = \vec{b}$ with many different solvers. Next we will check the time that the solvers need to solve the system. We will do that for both iterative and direct solvers.

To conclude, all of the team members will work together, so they can understand the solving methods for linear systems that will be used in this project. We as a team choose this way, instead for example, each student to do a singe solver.

CHAPTER 2

ELECTRIC FIELD INTEGRAL EQUATION FOR PEC

§ 2.1. Formulation

Equivalence Principle and Extinction Theorem

The vector wave equation, for a homogeneous medium is given by the following equation.

$$\nabla \times \nabla \times \vec{E}(\vec{r}) - k^2 \vec{E}(\vec{r}) = j\omega\mu \vec{J}(\vec{r}) \tag{2.1}$$

where $k^2 = \omega^2 \mu\epsilon$ and it is a constant.
We let the electric field to be the following.

$$\vec{E}(\vec{r}) = j\omega\mu \int_V \vec{J}(\vec{r'}) \bar{G}(\vec{r'}, \vec{r}) d\vec{r'} \tag{2.2}$$

Now if we assume that $\vec{J}(\vec{r'})$ is arbitary, which means it is equal to $\delta(\vec{r} - \vec{r'})$ Then we can easily prove the following expression about the dyadic Green's function.

$$\nabla \times \nabla \times \bar{G}(\vec{r}, \vec{r'}) - k^2 \bar{G}(\vec{r}, \vec{r'}) = \mathbf{I}\delta(\vec{r} - \vec{r'}) \tag{2.3}$$

Where I is the identity matrix. The solution of equation 2.3 is the following:

$$\bar{G}(\vec{r}, \vec{r'}) = (\mathbf{I} + \frac{\nabla\nabla}{k^2}) g(\vec{r}, \vec{r'}) \tag{2.4}$$

where

$$g(\vec{r}, \vec{r'}) = \frac{e^{-jk|\vec{r} - \vec{r'}|}}{4\pi|\vec{r} - \vec{r'}|} \tag{2.5}$$

From the above it is easy to see that $\bar{G}^t(\vec{r'}, \vec{r}) = \bar{G}(\vec{r}, \vec{r'})$. Therefore the equation 2.2 becomes the following.

$$\vec{E}(\vec{r}) = j\omega\mu \int_V \bar{G}(\vec{r}, \vec{r'}) \vec{J}(\vec{r'}) d\vec{r'} \tag{2.6}$$

Where r is an observation point (the point that we want to calculate the field) and r' is a source point.

Now we will derive the equivalence principle and extinction theorem. We consider the figure 2.1.

According to the figure 2.1 and the equation 2.1 the current $\vec{J}(\vec{r})$ is the currents $\vec{J}_1(\vec{r})$ in the volume V_1 and $\vec{J}_2(\vec{r})$ in the volume V_2.

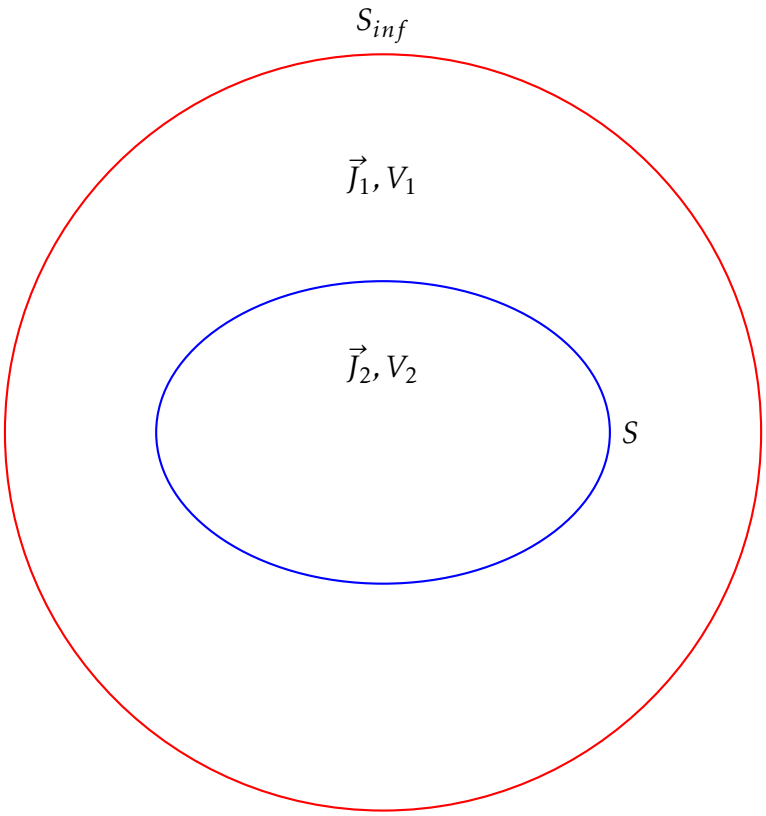


Figure 2.1: Basic Geometry for the derivation of the equivalence principle and extinction theorem

We will derive the extinction theorem. We multiply 2.1 by $\bar{G}(\vec{r},\vec{r'})$ from the left and also we multiply 2.3 by $\vec{E}(\vec{r})$ from the right. We will substract the two new equations and we will have the following equation:

$$\vec{E}(\vec{r})\nabla\times\nabla\times\bar{G}(\vec{r},\vec{r'})-\nabla\times\nabla\times\vec{E}(\vec{r})\bar{G}(\vec{r},\vec{r'})=\delta(\vec{r}-\vec{r'})\vec{E}(\vec{r})-j\omega\mu\vec{J}(\vec{r})\bar{G}(\vec{r},\vec{r'}) \tag{2.7}$$

We will intergrate the above equation over the volume V_1 which lies between S and the infinite sphere. First we consider the following equations.

$$\vec{E}(\vec{r'})=\int_{V_1}\delta(\vec{r}-\vec{r'})\vec{E}(\vec{r})d\vec{r} \tag{2.8}$$

$$\vec{E}_1(\vec{r'})=j\omega\mu\int_{V_1}\vec{J}_1(\vec{r})\bar{G}(\vec{r},\vec{r'})d\vec{r} \tag{2.9}$$

Therefore the integration is:

$$\vec{E}(\vec{r'})-\vec{E}_1(\vec{r'})=\int_{V_1}[\vec{E}(\vec{r})\nabla\times\nabla\times\bar{G}(\vec{r},\vec{r'})-\nabla\times\nabla\times\vec{E}(\vec{r})\bar{G}(\vec{r},\vec{r'})]dV \tag{2.10}$$

In the above integrals the current is only the $\vec{J}_1(\vec{r})$ because the integral takes place only in the volume V_1, therefore the current $\vec{J}_2(\vec{r})$ does not contriute to the integration.
In the equation 2.9 we will aply the following identity.

$$\nabla \cdot (\vec{A} \times \vec{B}) = (\nabla \times \vec{A}) \cdot \vec{B} - \vec{A} \cdot (\nabla \times B) \tag{2.11}$$

And after that the Gauss divergence theorem.

$$\iiint_V (\nabla \cdot \vec{F}) dV = \oiint (\vec{F}\hat{n}) dS \tag{2.12}$$

where $\hat{n}$ is the normal vector of the surface S and points outward of her.
Therefore according to the above the equation 2.10 becomes

$$\vec{E}(\vec{r'}) - \vec{E}_1(\vec{r'}) = \oiint_{S+S_{inf}} [\hat{n} \times \vec{E}(\vec{r}) \nabla \times \bar{G}(\vec{r}, \vec{r'}) + j\omega\mu\hat{n} \times \bar{H}(\bar{r}) \bar{G}(\bar{r}, \bar{r'})] dS \tag{2.13}$$

where we also have applied the Maxwell's equation (Faraday's Law) $\nabla \times \vec{E}(\vec{r}) = -j\omega\mu\vec{H}(\vec{r})$. In the equation 2.12 $\vec{E}(\vec{r'})$ vanishes for $\in V_2$ because the integral in equation 2.8 will be zero. Therefore the equation 2.13 can be written in a more appropriate way as follows:

$$\begin{cases} r \in V_1, & \vec{E}(\vec{r}) \\ r \in V_2, & 0 \end{cases} = \vec{E}_1(\vec{r}) + \oiint_S [\hat{n} \times \vec{E}(\vec{r}) \nabla \times \bar{G}(\vec{r}, \vec{r'}) + j\omega\mu\hat{n} \times \bar{H}(\bar{r}) \bar{G}(\bar{r}, \bar{r'})] dS' \tag{2.14}$$

Notice that we removed the integral over S_{inf}. That is because a finite source generates an electromagnetic field that decays as $\frac{1}{r}$. This is explained in chapter 3.3 of [1]. In 2.14 we swap $\vec{r}, \vec{r'}$ and we use the following two identities

$$\nabla \times \bar{G}(\vec{r}, \vec{r'}) = [\nabla \times \bar{G}(\vec{r'}, \vec{r})]^t \tag{2.15}$$

$$\bar{G}(\vec{r}, \vec{r'}) = [\bar{G}(\vec{r'}, \vec{r})]^t \tag{2.16}$$

Therefore we have:

$$\begin{cases} r \in V_1, & \vec{E}(\vec{r}) \\ r \in V_2, & 0 \end{cases} = \vec{E}_1(\vec{r}) + \oiint_S [\nabla \times \bar{G}(\vec{r}, \vec{r'}) \hat{n'} \times \vec{E}(\vec{r'}) + j\omega\mu \bar{G}(\bar{r}, \bar{r'}) \hat{n'} \times \bar{H}(\bar{r'})] dS' \tag{2.17}$$

We define the equivalent surface electric and magnetic currents impressed on a surface S as follows:

$$\vec{M}_s(\vec{r}) = -\hat{n} \times \vec{E}(\vec{r}) \tag{2.18}$$

$$\vec{J}_s(\vec{r}) = \hat{n} \times \vec{H}(\vec{r}) \tag{2.19}$$

Therefore the equation 2.17 takes the following final form.

$$\begin{cases} r \in V_1, & \vec{E}(\vec{r}) \\ r \in V_2, & 0 \end{cases} = \vec{E}_1(\vec{r}) - \iint_S [\nabla \times \bar{G}(\vec{r},\vec{r'})\vec{M}_s(\vec{r'}) - j\omega\mu\bar{G}(\bar{r},\bar{r'})\vec{J}_s(\vec{r'})]dS' \quad (2.20)$$

The explanation of the above equation is as follows. The electric field in the volume V_1 is generated by the sources inside V_1 (in our case $\vec{J}_1$) and the sources outside V_1 (in our case $\vec{J}_2$). This electric field is equal to the field $\vec{E}_1$ which is a product of $\vec{J}_1$, and of the field that it is generated by the equivalence surface currents.
Here we can see the *equivalence principle*: The field generated by the surface currents $\vec{M}_s$,$\vec{J}_s$ that are impressed on the surface S is equal to the field that it is generated by the all the sources inside V_2.
Also, the field is zero inside V_2. This means that the equivalnce currents cancel the electric field $\vec{E}_1$ inside V_2. This is the *extinction theorem*.

Electric Field Integral Equation

We consider an incident electromagnetic field $\vec{E}_{inc}$ that impinges on a perfect electric conductor (PEC). Therefore there will be (only) electric current on the PEC object, which will give a scattered field. To create a formulation for an electric field integral equation we will follow the formulation of chapter 3.4 of [1]. We consider the figure 2.2 for a scattering on PEC object.

We define a surface S^+ that it is just large enough to contain S, and we apply the *equivalence principle* and the *extinction theorem* to it. After the application of the above it is obvious that we will have an electric field (total field) outside of S^+ and zero field inside S^- (S^- is just small enough to be contained by S). Hence we remove the PEC object and we don't disturbed the field outside. The equivalent electric current is $\vec{J}_{eq} = \hat{n} \times \vec{H}$ and the magnetic current is zero, due to the PEC object. According to the figure 2.2 The equivalence principle and the extinction theorem becomes

$$\begin{cases} r \in V_1, & \vec{E}(\vec{r}) \\ r \in V_2, & 0 \end{cases} = \vec{E}_{inc}(\vec{r}) + \iint_S^+ j\omega\mu\bar{G}(\bar{r},\bar{r'})\vec{J}_{eq}(\vec{r'})dS' \quad (2.21)$$

Therefore for $r \in S$ we have the following equation

$$0 = \vec{E}_{inc}(\vec{r}) + \iint_S^+ j\omega\mu\bar{G}(\bar{r},\bar{r'})\vec{J}_s(\vec{r'})dS' \quad (2.22)$$

Where $\vec{J}_s = \vec{J}_{eq}$ because S^+ is infinitesimally close to S.
For a resonant modes whose surface is S, the requirement that $\hat{n} \times \vec{E} = 0$ on the cavity wall will ensure null solution inside the resonant modes except at the cavity resonance. Therefore away from the resonance it is sufficient to include the tangential component of 2.22 and we will get the following equation:

$$0 = \hat{n} \times \vec{E}_{inc}(\vec{r}) + j\omega\mu\hat{n} \times \iint_S \bar{G}(\bar{r},\bar{r'})\vec{J}_{eq}(\vec{r'})dS' \quad (2.23)$$

Equations 2.23 is the **Electric Field Integral Equation for Perfect Electric Conductors** (EFIE).

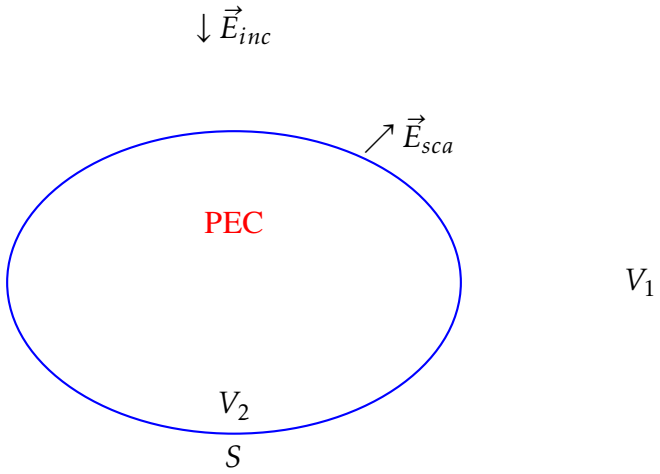


Figure 2.2: Scattering from a perfect electric conductor

§ 2.2. Solution

We will solve the EFIE for PEC sphere and we will see how close they are with the analytical solution for the Electric and Magnetic field.
Firt of all the analytical solution was given by Gustav Mie in 1908 and one can see it's formulation in the chapther 5 of [2].
To solve the EFIE we will use the method of moments. It would be better if we give a simple explanation of the method below.

Method of Moments

We consider the following problem

$$L(f) = g \tag{2.24}$$

Where L is an operator on a function f, in our case L is the operator of EFIE and f is the equivalent surface current $\vec{J}_{eq}(\vec{r})$ .Also g is the incident electric field for EFIE. We consider basis functions f_n for f and it's expression becomes:

$$f = \sum_{n=1}^{+\infty} a_n f_n \tag{2.25}$$

Where a_n are some coefficients that we need to solve for. Obviously we can't solve for an infinite number of basis function, so we choose a number N and we consider as an approximation of f the following function.

$$\tilde{f} = \sum_{n=1}^{N} a_n f_n \tag{2.26}$$

We consider another set of function w_n in a different space than the space of the f_n functions. The method of moments is summarized to the following expression:

$$R = L(f) - L(\tilde{f}) = min \tag{2.27}$$

The above can be achieved if we impose a zeroing in an averange sense as follows:

$$\langle w_m, R \rangle = 0 \Leftrightarrow \langle w_m, L(f) \rangle = \langle w_m, L(\tilde{f}) \rangle \tag{2.28}$$

From the equations 2.25 and 2.26 we derive the following matrix equation:

$$[\mathbf{X}_{nm}] a_n = g_m \tag{2.29}$$

where

$$[\mathbf{X}_{nm}] = \begin{bmatrix} < w_1, L(f_1) > & < w_1, L(f_2) > & \cdots & < w_1, L(f_N) > \\ < w_2, L(f_1) > & < w_2, L(f_2) > & \cdots & < w_2, L(f_N) > \\ \vdots & \vdots & \ddots & \vdots \\ < w_N, L(f_1) > & < w_N, L(f_2) > & \cdots & < w_N, L(f_N) > \end{bmatrix} \tag{2.30}$$

$$\mathbf{a}_n = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_N \end{bmatrix} \tag{2.31}$$

and

$$\mathbf{g}_m = \begin{bmatrix} < w_1, g > \\ < w_2, g > \\ \vdots \\ < w_N, g > \end{bmatrix} \tag{2.32}$$

We have to choose three things in order to create a linear system for solution. The first thing is the dimension N. We will solve many systems with bigger N every time and we will see which one is closer to the analytical Mie's solution. The second one is the basis functions. We will use the RWG functions introduced in [3]. For the testing function w_m we will use the Galerkin's method in which the spaces of the basis functions and testing functions are the same and therefore the functions themselves are the same. In the case of the Integral Equations we will have a matrix of the following form

$$[\mathbf{Z}] = \sum_{n=1}^{N} \sum_{m=1}^{N} [\mathbf{Z}^{mn}] \tag{2.33}$$

where,

$$[\mathbf{Z}^{mn}] = \int_{S_m} \vec{J}_m(\vec{r}) \vec{E}(\vec{J}_n(\vec{r'}))dS \tag{2.34}$$

The matrix $[\mathbf{Z}^{mn}]$ is a matrix that relates the surface equivalent current of an element n to the electric field of an element m. In the case of the RWG functions those elements are triangles like in the figure 2.3

We will write the EFIE as follows, in order to derive the $[\mathbf{Z}^{(mn)}]$ easily. The equation 2.23 is:

$$\vec{E}(\vec{r'}) = -j\omega\mu \iint_{S_n} g(\vec{r}, \vec{r'})\vec{J}(\vec{r'})dS' - \frac{1}{j\omega\epsilon}\nabla \int_{S_n} g(\vec{r}, \vec{r'})\nabla'\vec{J}(\vec{r'})dS' \tag{2.35}$$

We apply the MoM to EFIE and we have the following equation.

$$[\mathbf{Z}^{mn}] = -j\omega\mu \iint_{S_m} \vec{J}_m(\vec{r}) \iint_{S_n} g(\vec{r}, \vec{r'})\vec{J}_n(\vec{r'})dS'dS - \frac{1}{j\omega\epsilon} \iint_{S_m} \nabla \cdot \vec{J}_m(\vec{r}) \iint_{S_n} g(\vec{r}, \vec{r'})\nabla' \cdot \vec{J}_n(\vec{r'})dS'dS \tag{2.36}$$

From now on the things are easy. We only need to apply the RWG basis function in the above equations. It is very useful to do use simplex coordinates. Also since we can't calculate analytically the above integrals we will use a Gauss quadrature rule for triangles. All of the above for EFIE can be found in the chapter 3.4 of [4].

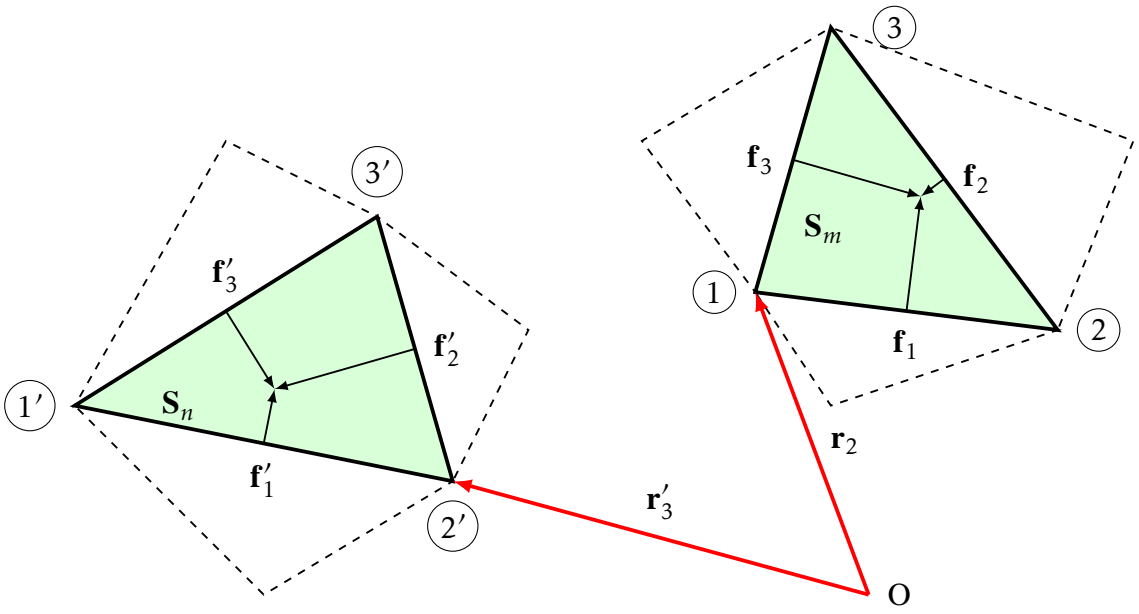


Figure 2.3: Relation of $[\mathbf{Z}^{(mn)}]$ and RWG functions

Singular Integrals

Someone could see that there are some singular integrals when we calculate the matrix **Z**. In the EFIE the singular integrals are weakly singular because we deal with just the Green's function (g). There are three kinds of weakly singular integrals for the EFIE.

- When we have coincident triangles
- When we have edge adjacent triangles
- When we have vertex adjacent triangles

Thankfully, we can use the open source code (DEMCEM) [5], provided in .m , .cpp and .mex files which calculates these kind of integrals.

CHAPTER 3

DIRECT SOLVERS

From this moment we can try and solve the linear system that is created.

$$[\mathbf{Z}]\vec{I} = \vec{B} \tag{3.1}$$

Where the matrix $[\mathbf{Z}]$ is the matrix of the equation 2.34, $\vec{B}$ is the incident field, and $\vec{I}$ are the coefficients of the surface equivalent electric current

$$\vec{J}_{eq} = \sum_{n=0}^{N} I_n \cdot \vec{f}_n \tag{3.2}$$

Where $\vec{f}_n$ is a RWG function and N is the number of edges of the discretization of our geometry (a RWG function exists on one edge).

We will use the following direct solvers for our system:

1. LU Decomposition.
2. QR Factorization.
3. Singular Value Decomposition.
4. Matlab's direct solver: `I = Z\B`.
5. Solving with the inverse of $[\mathbf{Z}]$.

We will check the solutions with two different discretizations.

1. *Case* 1: 230 Elements.
2. *Case* 2: 802 Elements.

Before we give the results it will be better if we gave a small explanation about all the above methods, how it is possible to solve linear systems with them, and how Matlab can handle them.

§ 3.1. LU decomposition

In the LU decomposition of a square matrix A we can write the matrix **A** as a multiplication of a lower triangular **L** and an upper triangular **U** matrix, as follows:

$$\mathbf{A} = \mathbf{LU} \tag{3.3}$$

Therefore, if we want to solve a linear system $\mathbf{A}\vec{x} = \vec{b}$, we write the system in the following form:

$$\mathbf{LU}\vec{x} = \vec{b} \tag{3.4}$$

And we break the task to two parts:

$$\mathbf{L}\vec{y} = \vec{b} \tag{3.5}$$

$$\mathbf{U}\vec{x} = \vec{y} \tag{3.6}$$

And we solve the two systems. This is very good, because since we do the LU decomposition, the two systems can be solved with only one pass, because of the lower and upper triangular forms of the matrices **L, U**. The solution is equivalent to the Gaussian elimination after the formulation of the triangular matrices.

§ 3.2. QR factorization

In the QR factorization we decompose a matrix **A** in to two matrices **Q** and **R**, where **Q** is orthogonal and **R** is an upper triangular matrix. Therefore the matrix **A** is:

$$\mathbf{A} = \mathbf{QR} \tag{3.7}$$

This factorization is also very useful because **Q** is unitary, therefore $\mathbf{Q}^T\mathbf{Q} = I$, where I is the identity matrix. To solve the linear system $\mathbf{A}\vec{x} = \vec{b}$ one doesn't need to compute the inverse of **Q**, but only it's transpose. The system then becomes:

$$\mathbf{R}\vec{x} = \mathbf{Q}^T\vec{b} \tag{3.8}$$

Now **R** is upper triangular and we have to solve a sytem like the two last system of LU decomposition, which as we said before can be solved with just one pass.

§ 3.3. SVD

In the singular vector decomposition, one decomposes a matrix into the following multiplication:

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^* \tag{3.9}$$

If **A** has dimensions $m \times n$, then **U** is a $m \times m$ unitary matrix, Σ is a $m \times n$ rectangular diagonal matrix with non-negative real numbers on the diagonal and **V** is a $n \times n$ unitary matrix.

The diagonal entries of Σ are the singular values of **A**, the columns of **U** are the left singular vectors of **A** and the columns of **V** are the right singular vectors of **A**.

To solve the linear system $\mathbf{A}\vec{x} = \vec{b}$ with SVD with consider the following equations:

$$\tilde{b} = \mathbf{U}^* \vec{b} \tag{3.10}$$

$$\tilde{x} = \mathbf{V}^* \vec{b} \tag{3.11}$$

Now we need to solve the following system

$$\mathbf{\Sigma}\tilde{x} = \tilde{b} \tag{3.12}$$

But here Σ is a diagonal matrix with positive numbers in the diagonal. Therefore the solution is the following

$$\tilde{x}_i = \frac{\tilde{b}_i}{\sigma_i} \tag{3.13}$$

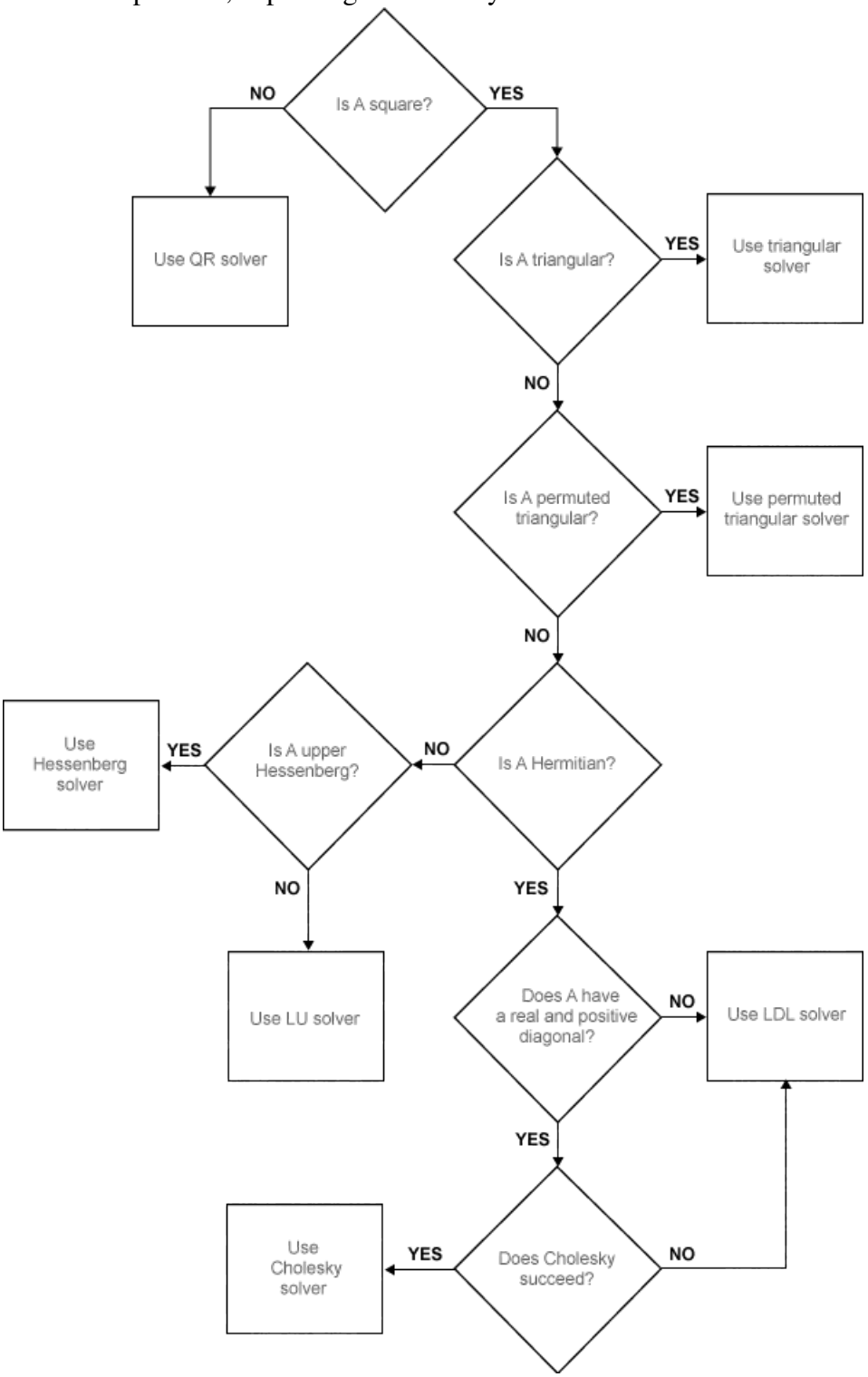
The system is solved again.

§ 3.4. Inverse Matrix Solution

For the inverse matrix, we will only mention how Matlab finds it. The Matlab's command `inv(A)` performs an LU decomposition of the input matrix. It then uses the results to form a linear system whose solution is the matrix inverse A^{-1}.

§ 3.5. Matlab's Backslash

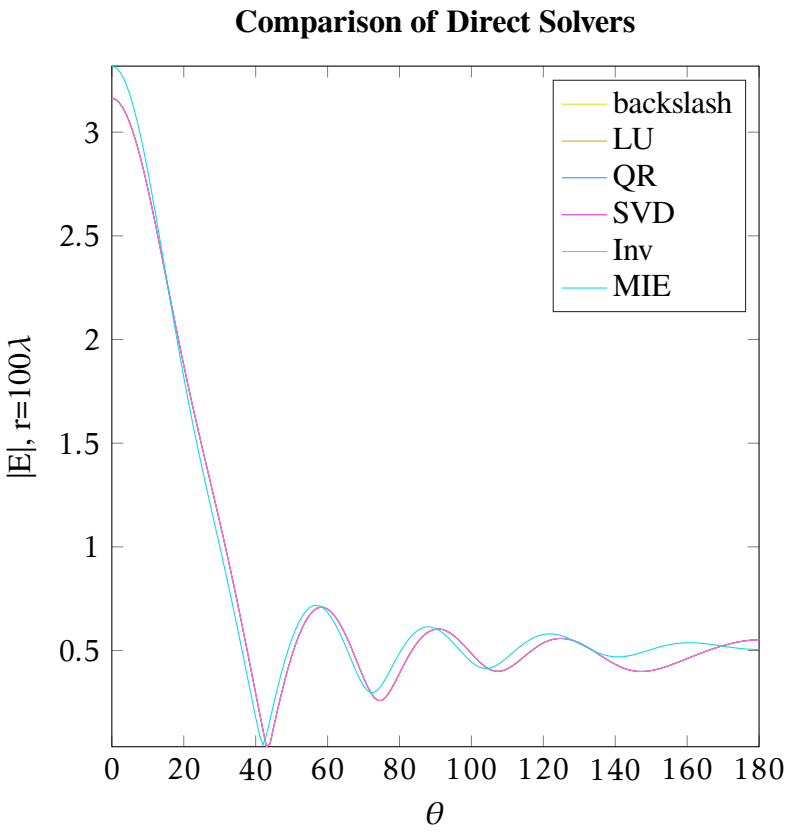
The Matlab's well known direct solver `I = Z\B` uses the following diagramm ([6]) and does a different computation, depending the linear system we want to solve.



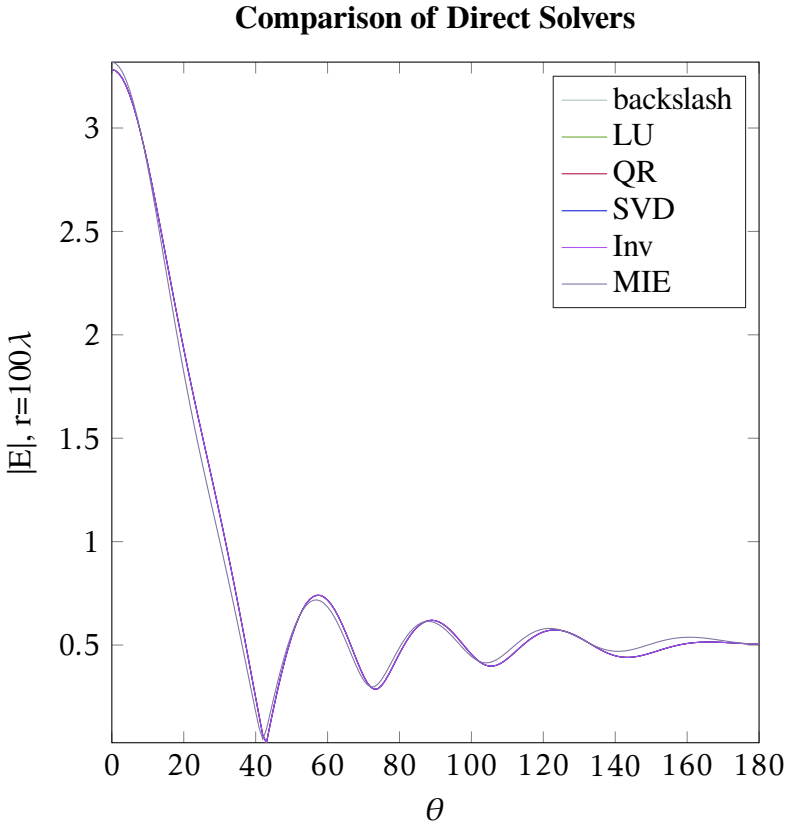
§ 3.6. Results

In this chapter we will show the results that occured from the solution of the EFIE with all the solvers, and we will compare them with the analytical solution of Gustav Mie.

As expected all of the solutions give the same results. This happens because our matrix [**Z**] is not ill-conditioned.

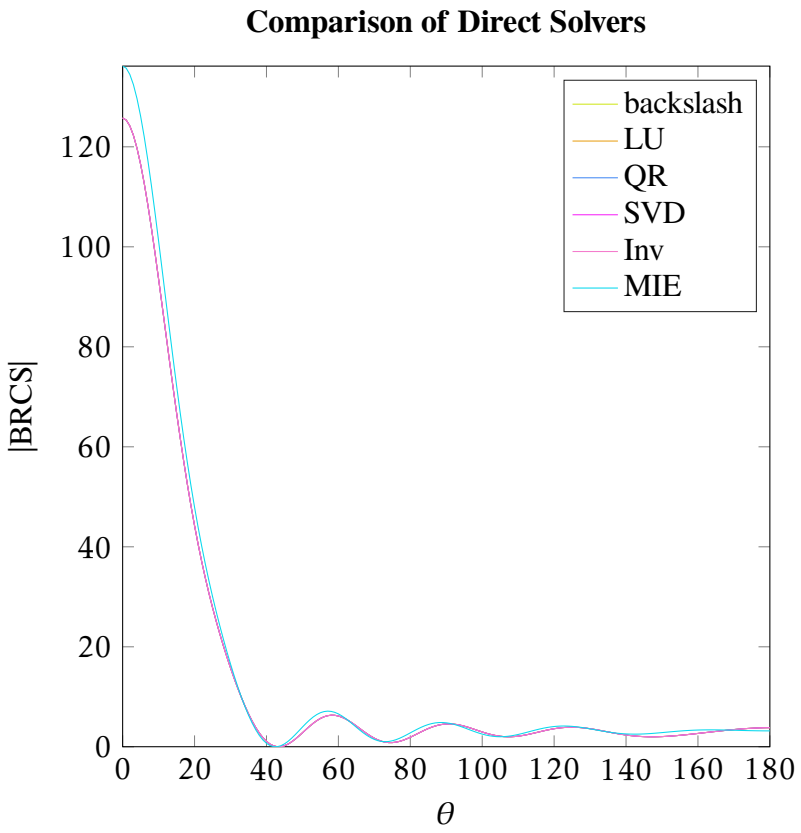


(α′) 230 Elements

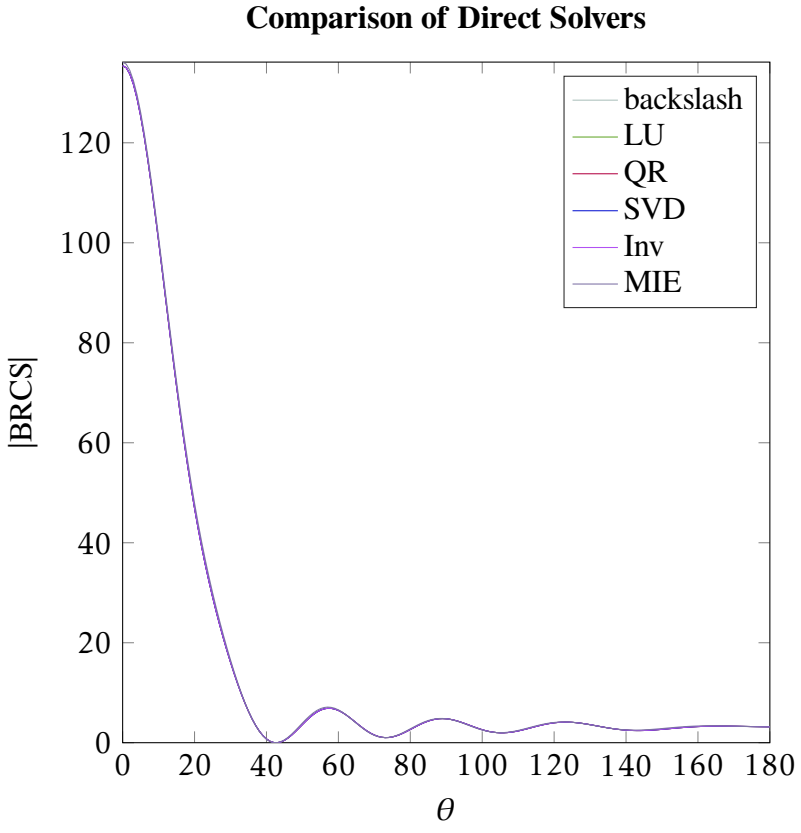


(β′) 802 Elements

Figure 3.1: Electric Field



(α′) 230 Elements

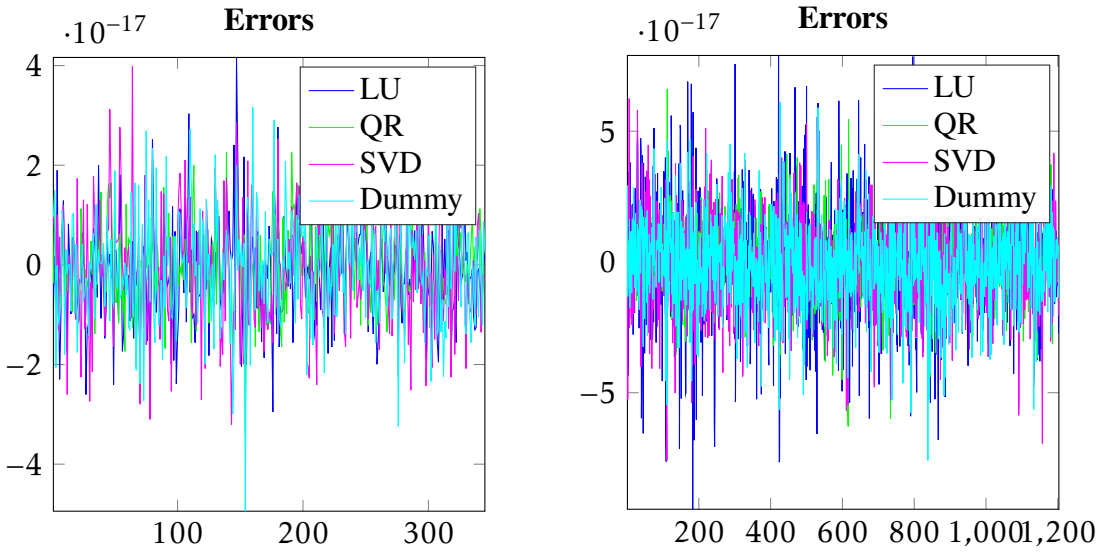


(β′) 802 Elements

Figure 3.2: BRCS

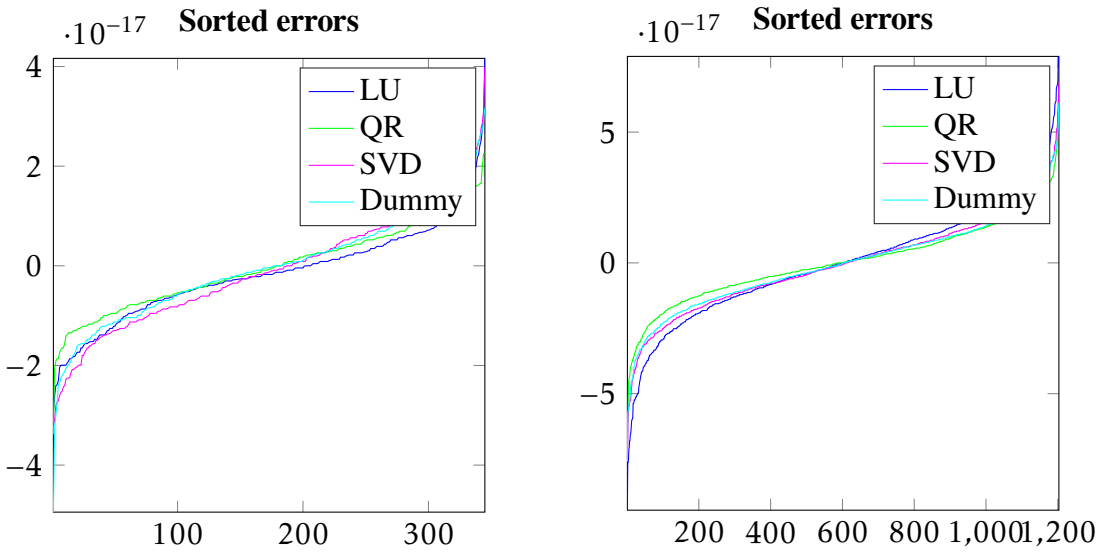
§ 3.7. Errors

In this chapter, we will just show in the figures below what is the difference between the four linear solvers and Matlab's backslash. We expect tiny differences.



(α′) 230 Elements (β′) 802 Elements

Figure 3.3: Error



(α′) 230 Elements (β′) 802 Elements

Figure 3.4: Sorted Error

The differences in all the cases are tiny, as expected. From the sorted plots in 3.4 it appears that QR factorization has the samllest errors, and LU the biggest for 230 and 802 elements.

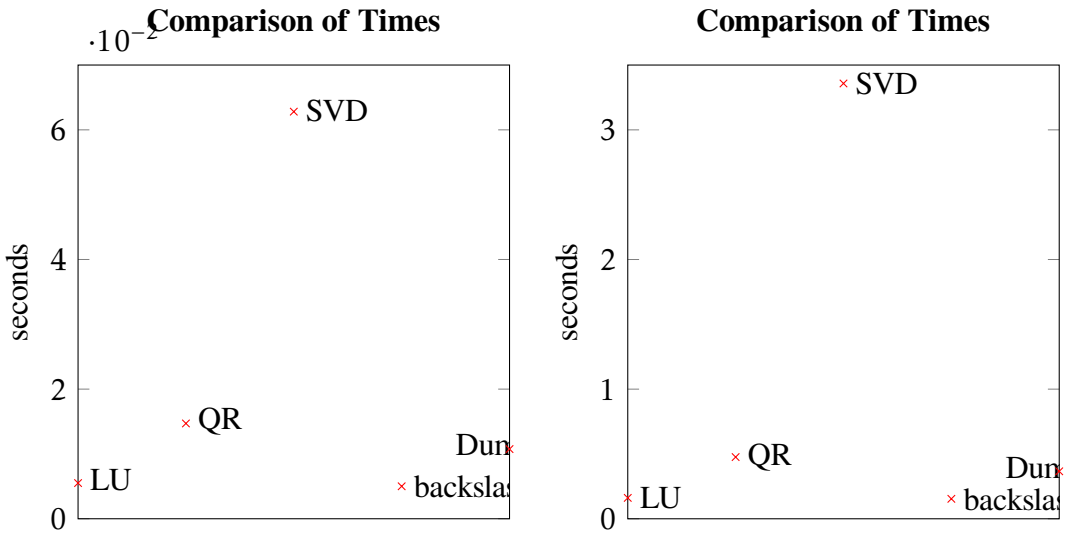
§ 3.8. Time Comparisson

With the above we only proved that there are extremely tiny differences from the result of the three solvers. Therefore we can use, for this specific matrix **Z**, any of them that we want if we only want to see the result. Although if we care about the time that the linear system needs to be solved, the case is different. Firt we can check in the figure 3.4 the time that all of the solvers need for the three discretizations.

We can see that SVD in every case takes the most of the time, and the matlab's method is the fastest. Therefore, we can't overcome matlab's speed. Although, let's take a closer look, with Matlab's profiler (run & time) to see more details about the time that every solver needs. This is a very important thing to check, because we need to see for example in QR factorization, how much the factorization itself needs, and how much time do we need to solve the system after the factorization.

Solver	230 elements		802 elements	
	Decomposition (secs)	Solver (secs)	Decomposition (secs)	Solver (secs)
LU	0.081	0.021	2.129	0.241
QR	0.217	0.013	6.778	0.132
SVD	0.930	0.011	27.053	0.074
Inverse	NA	0.161	NA	5.410
Backslash	NA	0.077	NA	2.328

Table 3.1: Time Comparisson for Direct Methods



(α′) 230 Elements (β′) 802 Elements

Figure 3.5: Times

So what happened. Are LU,QR and SVD slower from the Matlab's method. If we are going to solve one system definetely yes. But what if we want to solve 100 systems with the same matrix $[\mathbf{Z}]$ but different source matrices $\vec{B}$? Then the SVD is the faster way to do it, since we only need to compute the decomposition for $[\mathbf{Z}]$ once. This is common in computation electromagnetics. For example one may want to find the bistatic radar cross section for 180 different incident planes

waves in every integer pokar angle. If we discretize or object with 802 elements then, Matlab's method would need a total time of $180 \cdot 5.410 = 973.8$ secs but the SVD method would need a total time of $180 \cdot 0.074 + 27.053 = 40.372999$ secs. Therefore Matlab's method is the fastest direct solver from the five compared here, but if someone finds the decomposition of the matrix by himself, he doesn't have to do it again when he needs to solve another system with different sources. Therefore the solver that we will pick depends every time. Note that the inverse solver is the slower than the backslash solver and doesn't give any decomposition, therefore we will never use it.

CHAPTER 4

ITERATIVE SOLVERS

In the previous chapter we saw some direct solvers that give an excellent result in any case, but they might be a bit slow. In many cases someone doesn't care for the result to be exact, but he just wants to visualize it as fast as possible. For example, someone might want to measure the bistatic radar cross section from an antenna to see if the scattered field is within the legal framework. A microwave antenna for cell phones can't radiate an electromagnetic field over a legal limit. There are many similar examples, therefore we will study some iterative solvers, in order to achieve about half of the time that the backslash Matlab's solver needs.

We will study the following iterative solvers for our system for the same discretizations as above.

1. Richardson iteration
2. Richardson with different time steps
3. Preconditioned conjugate gradient method
4. Stabilized biconjugate gradient method
5. Generalized minimal residual method (GMRES)

As we did before, first we are going to explain the above iterative methods, then we will check the results that these methods provide, and also the time that they need for a standard set of tolerance and iteration. Then for the best of the above methods (By best we mean the faster and more accurate) we will sacrifice some accuracy until we can make them twice faster than the direct solver in the case with 802 elements.

§ 4.1. Richardson Iteration

The Richardson iterative method is the simplest iteration method (that is why it is also called "simple iteration method"). We have the following system to solve

$$[\mathbf{Z}]\vec{x} = \vec{b} \tag{4.1}$$

or

$$\tau([\mathbf{Z}]\vec{x} - \vec{b}) = 0 \tag{4.2}$$

where τ is the iteration parameter, which can always be chosen such that the method converges. The equation 4.2 can be written as follows:

$$\vec{x} - \tau([\mathbf{Z}]\vec{x} - \vec{b}) = \vec{x} \tag{4.3}$$

and for the final step

$$\vec{x}_{k+1} = \vec{x}_k - \tau([\mathbf{Z}]\vec{x}_k - \vec{b}) \tag{4.4}$$

An optimal choice for the iteration parameter τ for a diagonalizable matrix is:

$$\tau = \frac{2}{\lambda_{min} + \lambda_{max}} \tag{4.5}$$

The above is a result of the following thought

If $\vec{x}_e$ is the exact solution of out system, then the error in the k iteration will be $\vec{e}_k = \vec{x}_k - \vec{x}_e$. Therefore from the equation 4.4 it is:

$$\vec{e}_{k+1} = ([\mathbf{I}] - \tau[\mathbf{Z}])\vec{e}_k. \tag{4.6}$$

Therefore the solution converges if and only if $\|[\mathbf{I}] - \tau[\mathbf{Z}]\| < 1$ in every norm. The error will converge to zero if for all eigenvalues $|1 - \tau\lambda| < 1$, therefore the optimal choice is the equation 4.5 because we don't know the sign of all the eigenvalues. If we knew that every eigenvalue is positive we would have picked $\tau = \frac{2}{\lambda_{max}}$.

§ 4.2. Richardson Iteration With Different Time Steps

In this case we make a list of optimal parameters of τ and we calculate the vector $\vec{x}_{k+1}$ for this list of parameters with the exactly the same way as above. Then we will have a list of vector as candidates to be used in the next iteration. We will use this vector $\vec{x}_k$ that comes from this iteration parameter τ that minimizes the residual $\|[\mathbf{I}] - \tau[\mathbf{Z}]\|$.

§ 4.3. Preconditioned Conjugate Gradient Method

First we will explain the simple conjugate gradient method and then we will talk about the preconditioned method.

We choose an initial solution $\vec{x_0} = \vec{0}$. We need a metric which will tell us when whether we are closer to the solution, starting from $\vec{x_0}$. It is proven, that the solution of the system $[\mathbf{A}]\vec{x} = \vec{b}$ is also a unique minimizer of the following quadratic function

$$f(\vec{x}) = \frac{1}{2}\vec{x}^T[\mathbf{A}]\vec{x} - \vec{x}^T b \tag{4.7}$$

Where $x \in \mathbb{R}^n$. For the implementation of the above we will follow the algorithm below. First of all we want to minimize the residual $\vec{r} = \vec{b} - [\mathbf{A}]\vec{x}$. For the the first step we will have:

$$\vec{r_0} := \vec{b} - [\mathbf{A}]\vec{x_0} \tag{4.8}$$

$$\vec{p_0} := \vec{r_0} \tag{4.9}$$

Then we will do some repeats, in which we will have:

$$\alpha_k := \frac{\vec{r_k}^T \vec{r_k}}{\vec{p_k}^T [\mathbf{A}]\vec{p_k}} \tag{4.10}$$

$$\vec{x}_{k+1} := \vec{x_k} + \alpha_k \vec{p_k} \tag{4.11}$$

$$\vec{r}_{k+1} := \vec{r_k} - \alpha_k [\mathbf{A}]\vec{p_k} \tag{4.12}$$

Now, If $\vec{r}_{k+1}$ is small enough for our case, we will exit the loop and the solution will be the $\vec{x}_{k+1}$, otherwise we continue and

$$\beta_k := \frac{\vec{r}_{k+1}^T \vec{r}_{k+1}}{\vec{r_k}^T \vec{r_k}} \tag{4.13}$$

$$\vec{p}_{k+1} := \vec{r}_{k+1} - \beta_k \vec{p_k} \tag{4.14}$$

And we start another loop, until $\vec{r}_{k+1}$ is small enough.

The problem with the method above is that it is slower if the condition number of the matrix $[\mathbf{A}]$ is big. Therefore we will use precondition in order to replace the original system $[\mathbf{A}]\vec{x} = \vec{b}$ with $[\mathbf{M}^{-1}]([\mathbf{A}]\vec{x} - \vec{b}) = 0$ such that the condition number of the matrix $[\mathbf{M}^{-1}][\mathbf{A}]$ will be smaller than the condition number of $[\mathbf{A}]$. The algorithm of the preconditioned conjuate gradient method is similar with the one of the simple conjugate gradient method. We will provide it below:

Initial step:

$$\vec{r_0} := \vec{b} - [\mathbf{A}]\vec{x_0} \tag{4.15}$$

$$\vec{z_0} := [\mathbf{M}^{-1}]\vec{r_0} \tag{4.16}$$

$$\vec{p_0} := \vec{z_0} \tag{4.17}$$

Repeats:

$$\alpha_k := \frac{\vec{r}_k^T \vec{z}_k}{\vec{p}_k^T [\mathbf{A}] \vec{p}_k} \tag{4.18}$$

$$\vec{x}_{k+1} := \vec{x}_k + \alpha_k \vec{p}_k \tag{4.19}$$

$$\vec{r}_{k+1} := \vec{r}_k - \alpha_k [\mathbf{A}] \vec{p}_k \tag{4.20}$$

Stop if $\vec{r}_{k+1}$ is small enough.

$$\vec{z}_{k+1} := [\mathbf{M}^{-1}] \vec{r}_{k+1} \tag{4.21}$$

$$\beta_k := \frac{\vec{z}_{k+1}^T \vec{r}_{k+1}}{\vec{z}_k^T \vec{r}_k} \tag{4.22}$$

$$\vec{p}_{k+1} := \vec{z}_{k+1} - \beta_k \vec{p}_k \tag{4.23}$$

End of repeats

Solution: $\vec{x}_{k+1}$.

§ 4.4. Stabilized Biconjugate Gradient Method

We will show the algorithm for the unpreconditioned biconjugate gradient method and then for the preconditioned one (which we use). First of all, this method has faster and smoother convergence than other conjugate gradient methods. We note that it is a Krylov subspace method. The unpreconditioned algorithm for the following system $[\mathbf{A}]\vec{x} = \vec{b}$ is the following:

1. Initial guess $\vec{x}_0$
2. $\vec{r}_0 = \vec{b} - [\mathbf{A}]\vec{x}_0$
3. Choose $\hat{r}_0$, such as $< \hat{r}_0, \vec{r}_0 > neq 0$, e.g. $\hat{r}_0 = \vec{r}_0$
4. $\rho_0 = \alpha = \omega_0 = 1$
5. $\vec{v}_0 = \vec{p}_0 = \vec{0}$
6. Repeats (for $i = \dots$)
   - 6.1. $\rho_i = < \hat{r}_0, \vec{r}_{i-1} >>$
   - 6.2. $\beta = \frac{\rho_i}{\rho_{i-1}} \frac{\alpha}{\omega_{i-1}}$
   - 6.3. $\vec{p}_i = \vec{r}_{i-1} + \beta(\vec{p}_{i-1} - \omega_{i-1}\vec{v}_{i-1})$
   - 6.4. $\vec{v}_i = [\mathbf{A}]\vec{p}_i$
   - 6.5. $\alpha = \frac{\rho_i}{<\hat{r}_0, \vec{v}_i>}$
   - 6.6. $\vec{h} = \vec{x}_{i-1} + \alpha\vec{p}_i$
   - 6.7. If $\vec{h}$ is accurate enough, then set $\vec{x}_i = \vec{h}$ and quit.
   - 6.8. $\vec{s} = \vec{r}_{i-1} - \alpha\vec{v}_i$
   - 6.9. $\vec{t} = [\mathbf{A}]\vec{s}$
   - 6.10. $\omega_i = \frac{<\vec{t}, \vec{s}>}{<\vec{t}, \vec{t}>}$
   - 6.11. $\vec{x}_i = \vec{h} + \omega_i\vec{s}$
   - 6.12. If $\vec{x}_i$ is accurate enough, quit.
   - 6.13. $\vec{r}_i = \vec{s} - \omega_i\vec{t}$

The preconditioned algorithm, which we will use, for the following system $[\mathbf{A}]\vec{x} = \vec{b}$ with a preconditioner $[\mathbf{K}] = [\mathbf{K}_1][\mathbf{K}_2] \approx [\mathbf{A}]$ is the following:

1. Initial guess $\vec{x}_0$
2. $\vec{r}_0 = \vec{b} - [\mathbf{A}]\vec{x}_0$
3. Choose $\hat{r}_0$, such as $< \hat{r}_0, \vec{r}_0 > neq 0$, e.g. $\hat{r}_0 = \vec{r}_0$
4. $\rho_0 = \alpha = \omega_0 = 1$
5. $\vec{v}_0 = \vec{p}_0 = \vec{0}$

6. Repeats (for $i = \ldots$)

   6.1. $\rho_i = < \hat{r}_0, \vec{r}_{i-1} >>$

   6.2. $\beta = \frac{\rho_i}{\rho_{i-1}} \frac{\alpha}{\omega_{i-1}}$

   6.3. $\vec{p}_i = \vec{r}_{i-1} + \beta(\vec{p}_{i-1} - \omega_{i-1}\vec{v}_{i-1})$

   6.4. $\vec{y} = [\mathbf{K}^{-1}]\vec{p}_i$

   6.5. $\vec{v}_i = [\mathbf{A}]\vec{y}$

   6.6. $\alpha = \frac{\rho_i}{<\hat{r}_0, \vec{v}_i>}$

   6.7. $\vec{h} = \vec{x}_{i-1} + \alpha\vec{y}$

   6.8. If $\vec{h}$ is accurate enough, then set $\vec{x}_i = \vec{h}$ and quit.

   6.9. $\vec{s} = \vec{r}_{i-1} - \alpha\vec{v}_i$

   6.10. $\vec{z} = [\mathbf{K}^{-1}]\vec{s}$

   6.11. $\vec{t} = [\mathbf{A}]\vec{z}$

   6.12. $\omega_i = \frac{<[\mathbf{K}_1^{-1}]\vec{t}, [\mathbf{K}_1^{-1}]\vec{s}>}{<[\mathbf{K}_1^{-1}]\vec{t}, [\mathbf{K}_1^{-1}]\vec{t}>}$

   6.13. $\vec{x}_i = \vec{h} + \omega_i\vec{z}$

   6.14. If $\vec{x}_i$ is accurate enough, quit.

   6.15. $\vec{r}_i = \vec{s} - \omega_i\vec{t}$

§ 4.5. Generalized Minimal Residual Method (GMRES)

The generalized minimal residual method is used to solve nonsymmetric systems of linear equations (it can solve also symmetric systems like ours). The method approximates the solution by the vector in a Krylov subspace with minimal residual. The Arnoldi iteration is used to find this vector. We will see the implementation for the solution of the system $[\mathbf{A}]\vec{x} = \vec{b}$. We assume that $\vec{b}$ is normalized. Even if it is not we can make it normal and just change the matrix $[\mathbf{A}]$ a bit. The $n-th$ dimension Krylov subspace is

$$K_n = K_n([\mathbf{A}], \vec{b}) = span\{\vec{b}, [\mathbf{A}]\vec{b}, [\mathbf{A}^2]\vec{b}, \ldots, [\mathbf{A}^{n-1}]\vec{b}\} \tag{4.24}$$

The GMRES approximates the exact solution of the system, by the vector $\vec{x}_n \in K_n$ that minimizes the Euclidean norm of the residual $\vec{r}_n = [\mathbf{A}]\vec{x}_n - \vec{b}$.

Because of the almost linearly dependency of the vectors of the Krylov subspace, we will use the Arnoldi iteration to find an orthonormal basis $\{\hat{q}_1, \hat{q}_2, \hat{q}_3, \ldots, \hat{q}_n,\}$ for K_n. Therefore we can express the vector $\vec{x}_n$ in the basis $[\mathbf{Q}_n]$ such as:

$$\vec{x}_n = [\mathbf{Q}_n]\vec{y}_n \tag{4.25}$$

In the equation 4.25 $\vec{y}_n \in \mathbb{R}^n$ and $[\mathbf{Q}_n]$ is the $m \times n$ matrix formed by the basis vectors.

Furthermore the Arnoldi iteration produces an $(n+1)\times n$ upper Hessenberg matrix $[\tilde{\mathbf{H}}_n]$ such as:

$$[\mathbf{A}][\mathbf{Q}_n] = [\mathbf{Q}_{n+1}][\tilde{\mathbf{H}}_n] \tag{4.26}$$

Due to the orhtogonality of $[\mathbf{Q}_n]$ we have the following equations

$$\|[\mathbf{A}]\vec{x}_n - \vec{b}\| = \|[\tilde{\mathbf{H}}_n]\vec{y}_n - [\mathbf{Q}_{n+1}^T]\vec{b}\| = \|[\tilde{\mathbf{H}}_n]\vec{y}_n - \beta\hat{e}_1\| \tag{4.27}$$

Where $\beta = \|\vec{b} - [\mathbf{A}]\vec{x}_0\|$, with $\vec{x}_0 = \vec{0}$ being the first trial vector, and $\vec{e}_1 = (1, 0, 0, \ldots, 0)^T$ is the first vector of the standard basis of $\mathbb{R}^{n+1}$.

The solution $\vec{x}_n$ can be found by minimizing the Euclidean norm of the the following residual

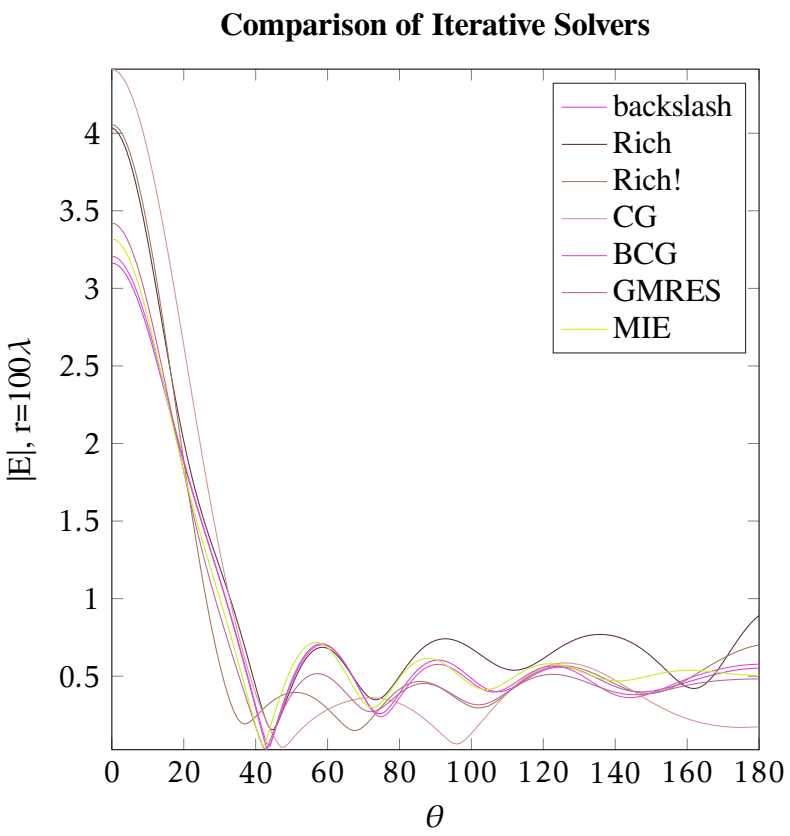
$$\vec{r}_n = [\tilde{\mathbf{H}}_n]\vec{y}_n - \beta\hat{e}_1 \tag{4.28}$$

We repeat the above procedure until the residual is small enough.

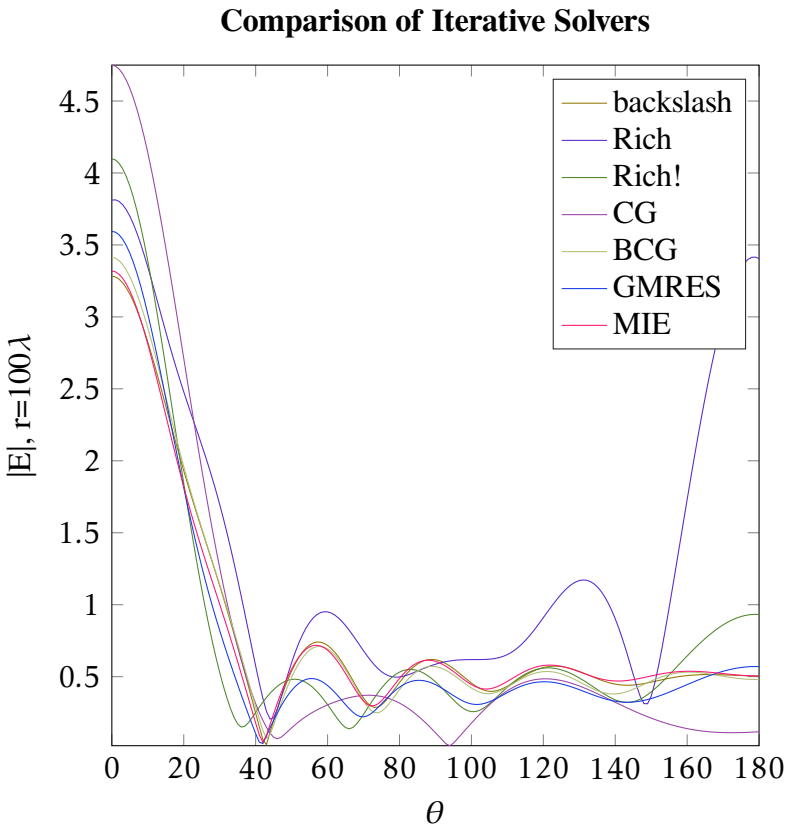
§ 4.6. Results

In this section we will show the electric field in some distance from the sphere and also the bistatic radar cross section. We expect different results compared with the direct solvers. Remember that we are using iterative solvers, which means that there will be some erros. We provide some figures that compare all the above solvers with the backslash solver and the analytical solutuion. The iterations for the Richardson's solvers will be 5 ,for GMRES will be 10 and for every other solver will be 20 and the tolerance will be 10^{-6}.

In the figures 4.1 and 4.2 we can see that not all solvers work well. Especially for the second discretization with 802 elements. But among them there are two very good cases, The preconditioned stabilized biconjugate gradient method and the generalized minimal residual method. We can see the electric field only with these two cases for both our discretizations in the figure 4.3. We will focus on these two later.

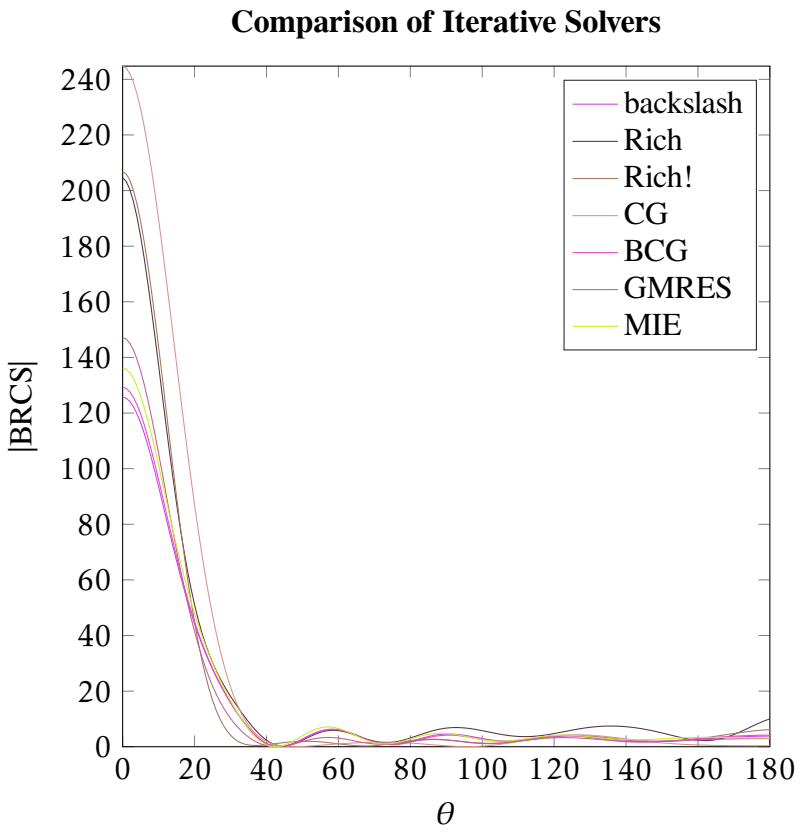


(α′) 230 Elements

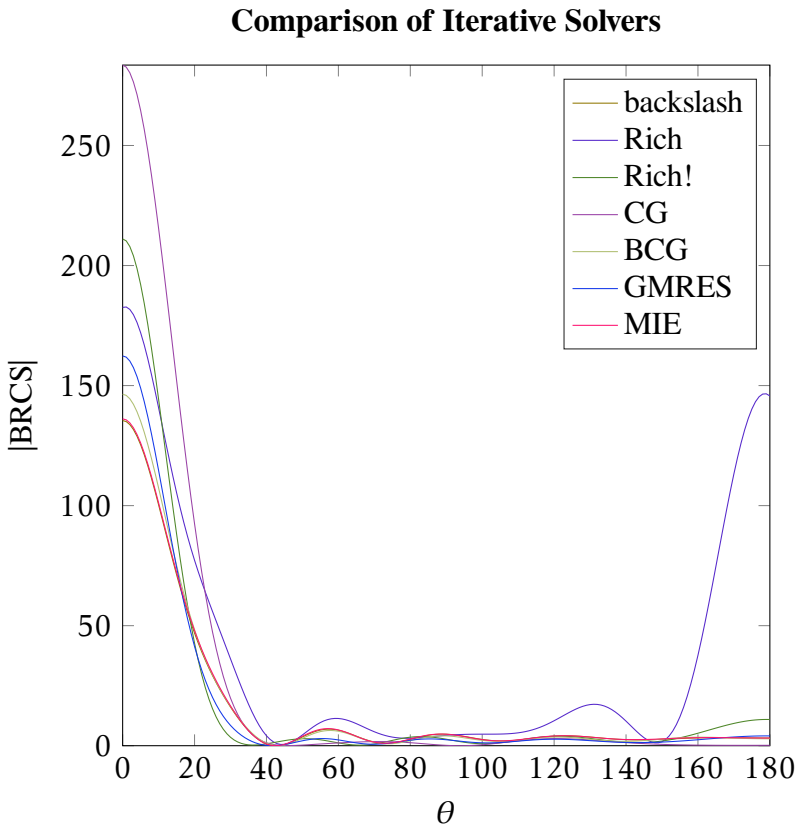


(β′) 802 Elements

Figure 4.1: Electric Field

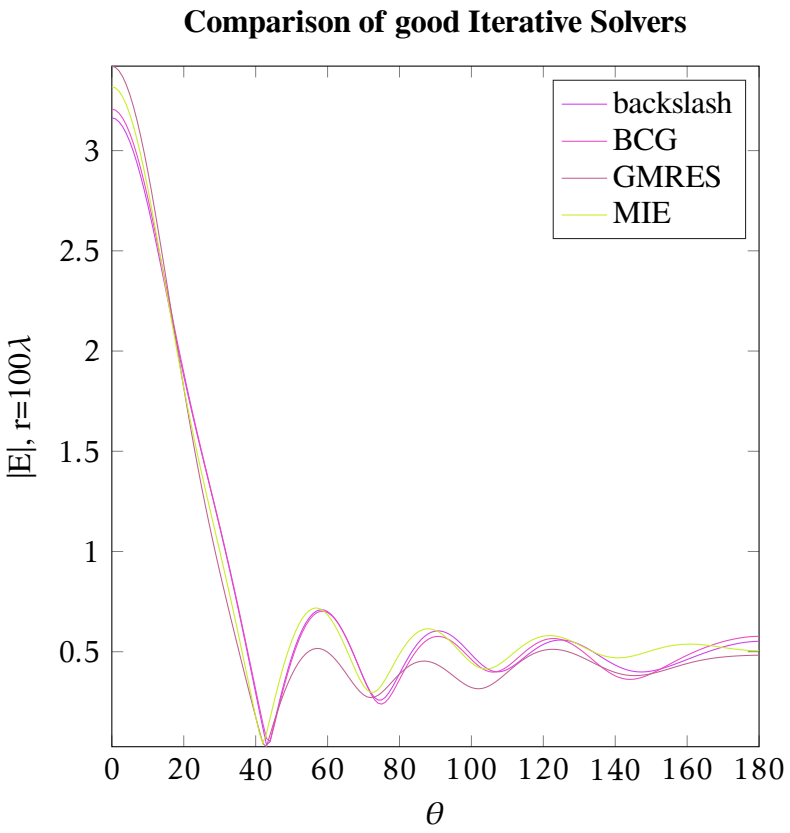


(α′) 230 Elements

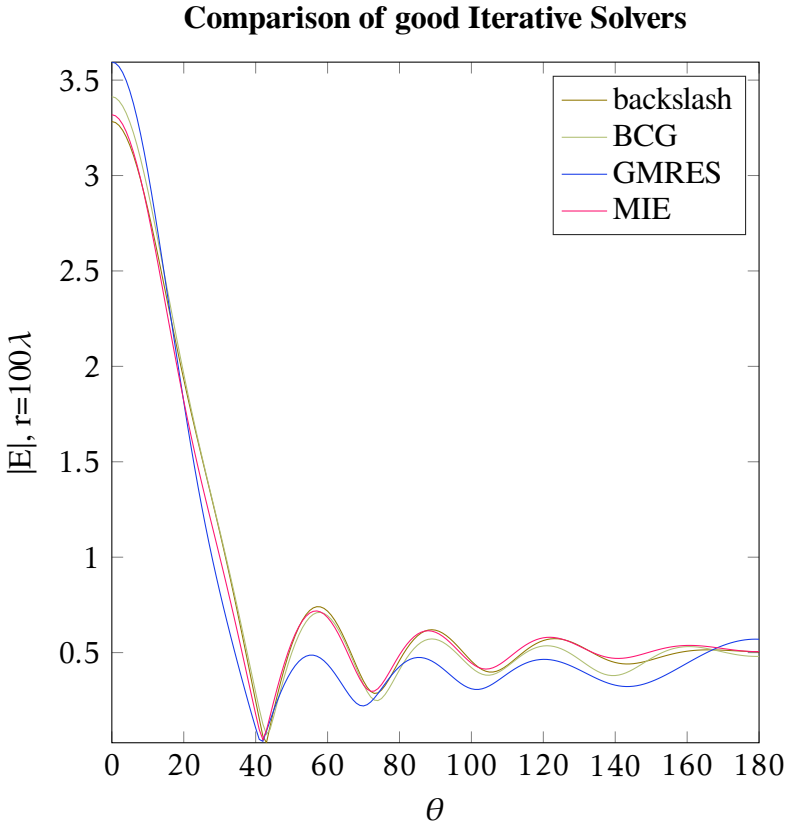


(β′) 802 Elements

Figure 4.2: BRCS



(α′) 230 Elements

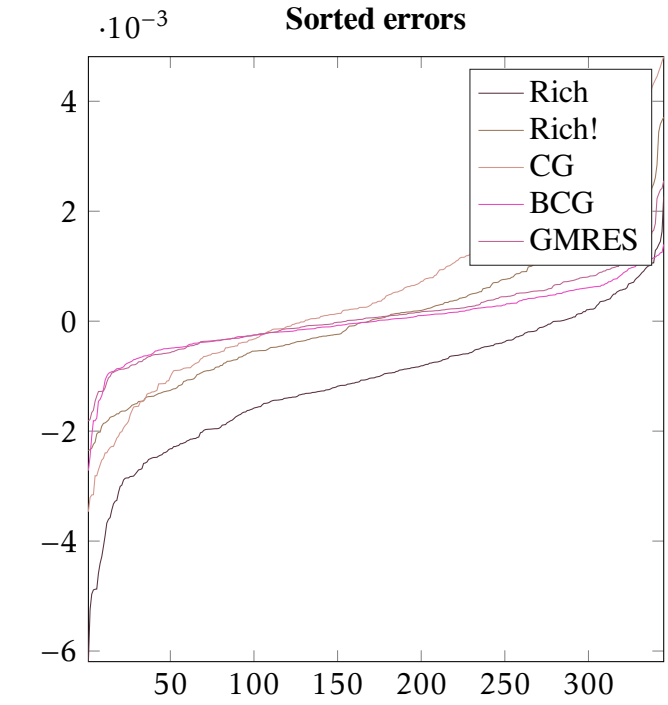


(β′) 802 Elements

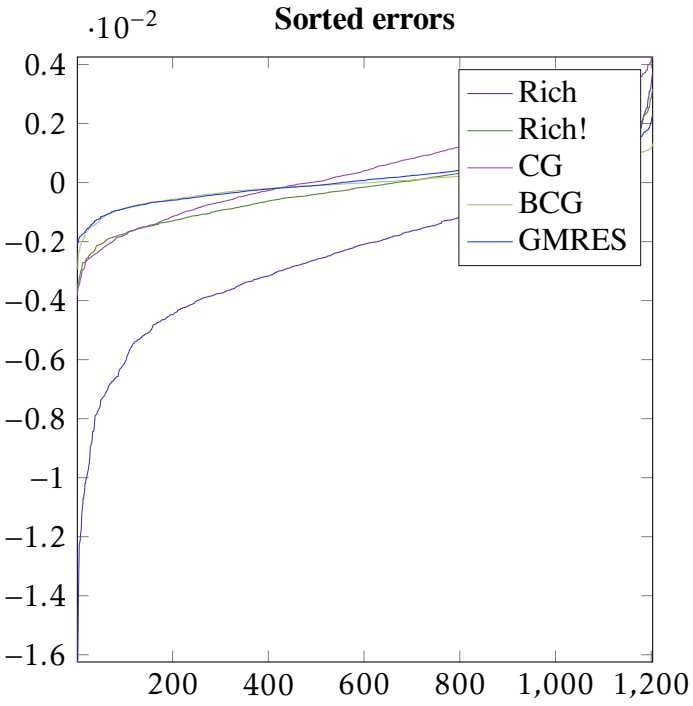
Figure 4.3: Electric Field

§ 4.7. Errors

In this section we will see the errors of the above solvers. We expect big differences. In the direct solvers the absolute error was technically zero. But here things change. Let's have a look at the sorted errors (difference with backslash) in the figure 4.4.



(α′) 230 Elements



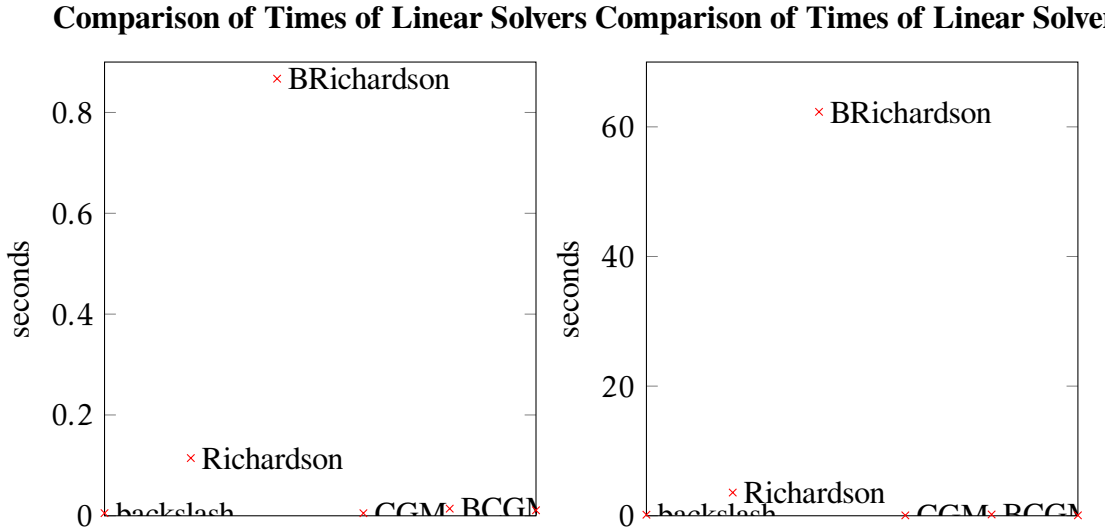
(β′) 802 Elements

Figure 4.4: Sorted Error

As we see the errors are $\approx \frac{10^{-2}}{10^{-17}} = 10^{15}$ times bigger than the direct solver errors. Although, all of the error curves are close to each other, and if we see the results we see that the differences between them are small (not acceptable in some cases though). Only the simple iteration method of Richardson is pretty different than the others. We were expecting that since the electric field and the bistatic radar cross section produced from this solver are very differrent compared to the other solvers.

§ 4.8. Time Comparisson

We will provided a figure with the comparisson of the time in seconds that all the solvers need, in order to compute the result.



(α′) 230 Elements (β′) 802 Elements

Figure 4.5: Times

As we see the Richardson Iteration with different time steps, takes ridiculosly more time the other methods, This is due to the fact that we need to compute the eigenvalues of a big matrix. Also we remember from the figures 4.1 and 4.2 that the simple Richardson iteration gave wrong results. Therefore we will focus only in the other three solvers.

We will use the Matlab's profiler as before to check the exact time that the solvers need in order to achieve the results in 4.1 and 4.2.

Solver	230 elements seconds	802 elements seconds	
CGM	0.110	0.953	
BicCGstab	0.259	3.081	
GMRES	0.236	1.223	
Backslash	0.075	2.315	

Table 4.1: Time Comparisson for fast iterative methods

The times are pretty much similar. Lets focus on the second case with the bigger matrices, because iterative methods work better in bigger matrices. Now the GMRES method and the CGM method are about twice as fast as the backslash solver, therefore we don't need to change something in them. The preconditioned stabilized biconjugate gradient method takes more time than the direct solver and we don't want that. There is no logic in using an iterative method which is slower from the direct method. Therefore we will change the number of the iterations in order to achieve faster results.

With 7 iterations and a tolerance 10^{-12} we can achieve a time of 1.296 seconds which is about twice faster than the backslash solver.

Now let's see the result for the electric field with the backslash solver, the analytical solution, the preconditioned conjugate gradient method (20 iterations), the generalized minimal residual method (10 iterations) and the faster preconditioned stabilized biconjugate gradient method (7 iterations).

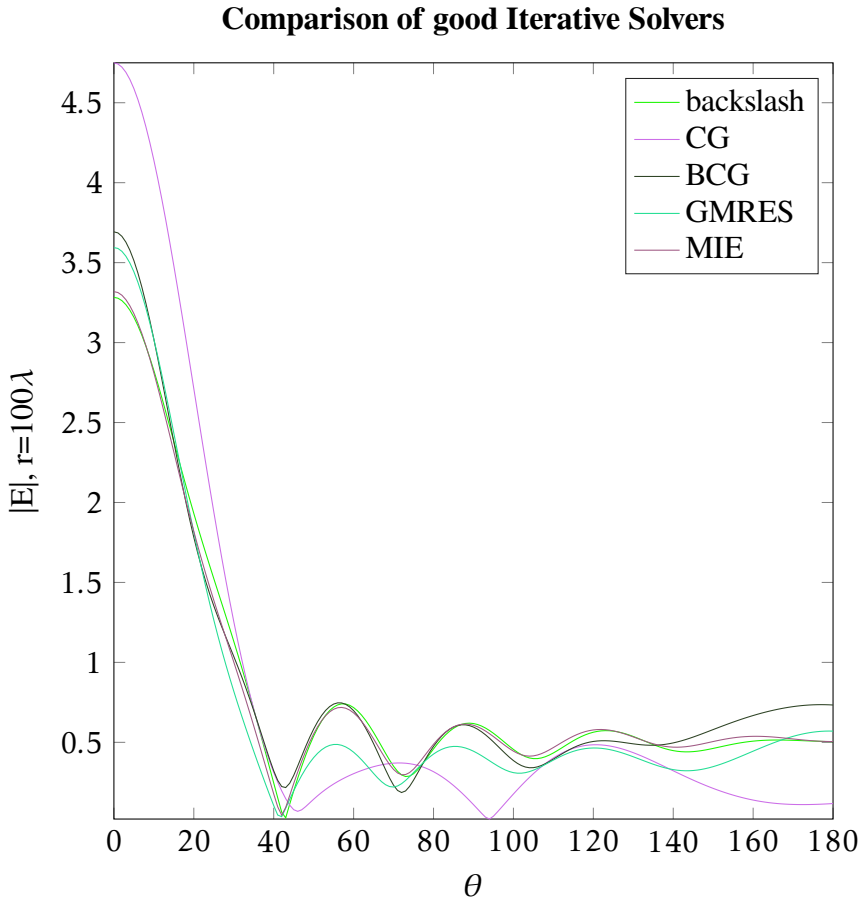


Figure 4.6: 803 elements

From the figure 4.6 we can see that the simple CG method doesn't give good results, therefore we will choose either the GMRES or the BCG method.

§ 4.9. Big System

For the last part of this chapter we will check what happens when we have a bigger system than before (3680 elements) elements. We want to do that because 1 or 2 seconds is not too much time. Therefore we want to check the results, and the differences in time, from the GRMES and BCG solvers when the time that the systems needs to be solved is actually big (around one minute).

We set again a tolerance of 10^{-6} ,10 iterations for the GMRES and 20 iterations for the BCG. In the figure 4.7 you can see the results and in the table 4.2 we see the time the three solvers need.

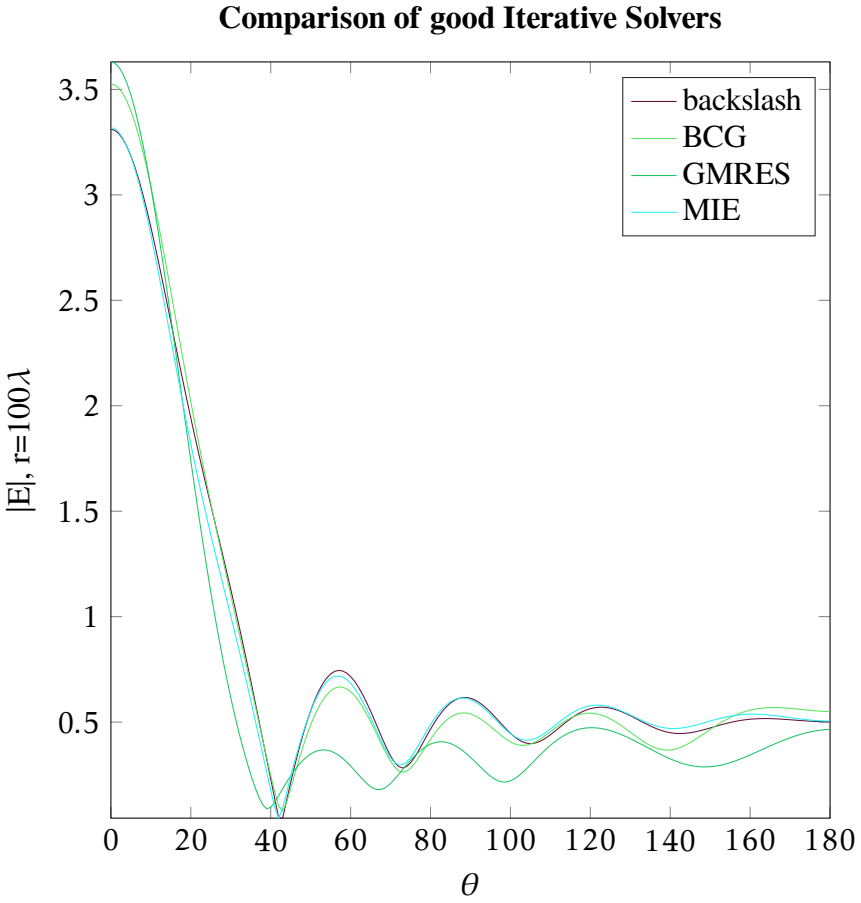


Figure 4.7: 3680 elements

Solver	Seconds	
BicCGstab	29.185	
GMRES	16.619	
Backslash	51.322	

Table 4.2: Times for big system

From the table 4.3 we conclude that the iterative method BCG is faster (almost twice as fast) than the backslash method. Also the GMRES is about 3 to 4 times faster than the backslash method. Although the results show that the BicCGstab is much closer to the exact solution. Therefore we conclude that the best solution is the BCG method. And it is better to use it when your system is big. Although since GMRES method is so fast we can increase its iterations an see what happens both in time and in the accuracy.

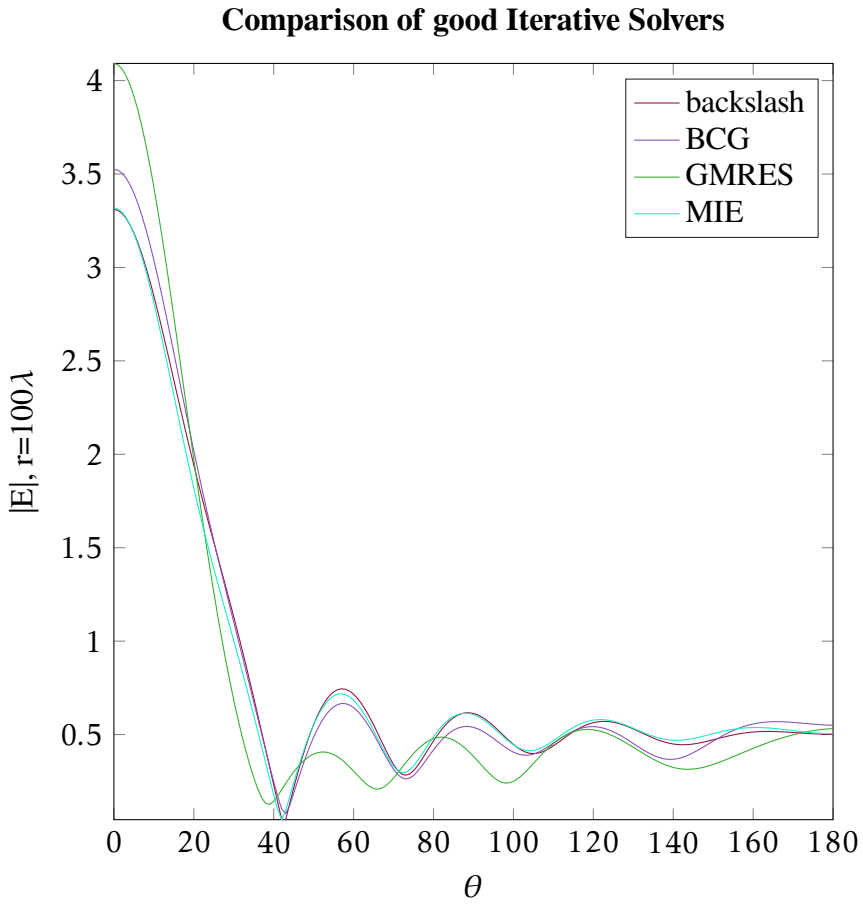


Figure 4.8: 3680 elements

Solver	Seconds	
BicCGstab	29.185	
GMRES	30.517	
Backslash	51.322	

Table 4.3: 20 Iterations

Unfortunately the result doesn't get any better when we increase the repeats. One could say that it is worse for small angles. The time that GMRES needs for 20 iterations is bigger than the one that BCG needs. Therefore our conclusion, based on the above results is final. BCG method is the best iteration method of the above. It is pretty fast for big systems and it is far more accurate to the exact result than any other method.

§ 4.10. Restarted GMRES

We are not yet convinced that our result has such differences (small but noticeable) with the exact result. Therefore we will study another method, the restarted generalized minimal residual method. Because the cost of the iteration grows, we choose to restart the method after a number of iterations. The problem here is that the restarted subspace is close to the earlier subspace.

For starters, let us try to see what happens if we choose, 5 restarts, tolerance equal to 10^{-12} and 10 iterations. We will work for the big matrix, because as we saw before iterative solvers work better for big systems. The electric field, in some distance R can be seen in the figure 4.9.

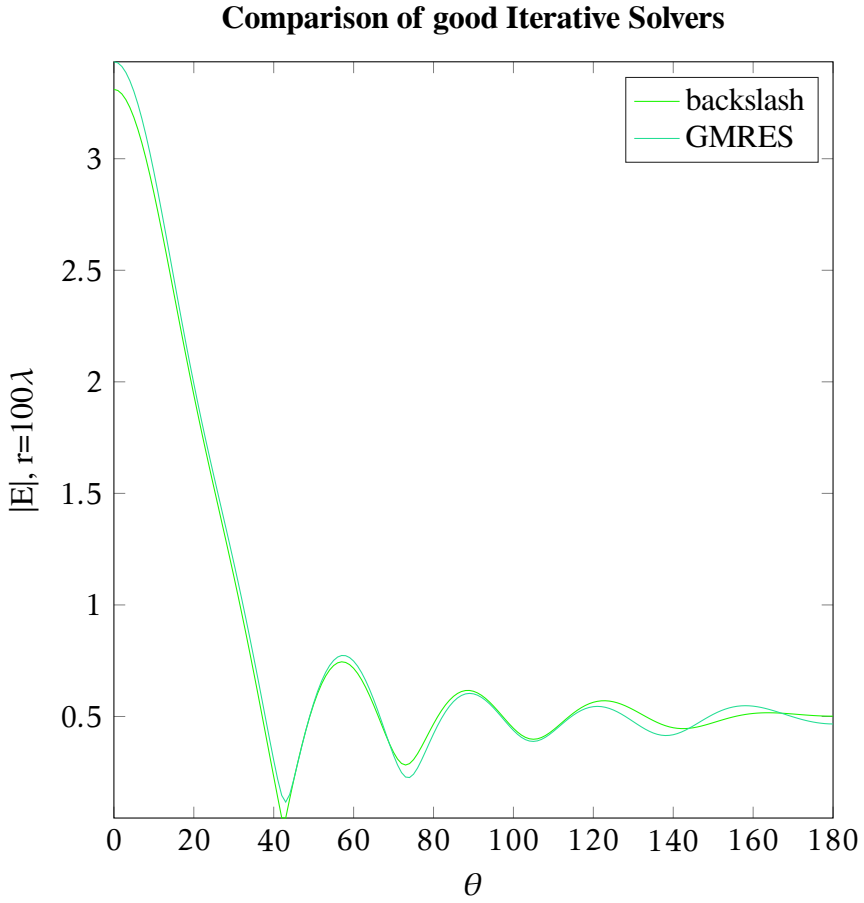


Figure 4.9: 3680 elements

As we see the results are extremely close to each other. Now let's check the times as we did before.

Solver	Seconds	
GMRES	34.680	
Backslash	51.569	

Table 4.4: 5 Restarts, 10 Iterations

So as we see the restarted GMRES is even better than the BCG. Let's try one more implementation.
We have the following results for 10 restarts and 10 iterations
The result is even better than before, but we should check the times again.

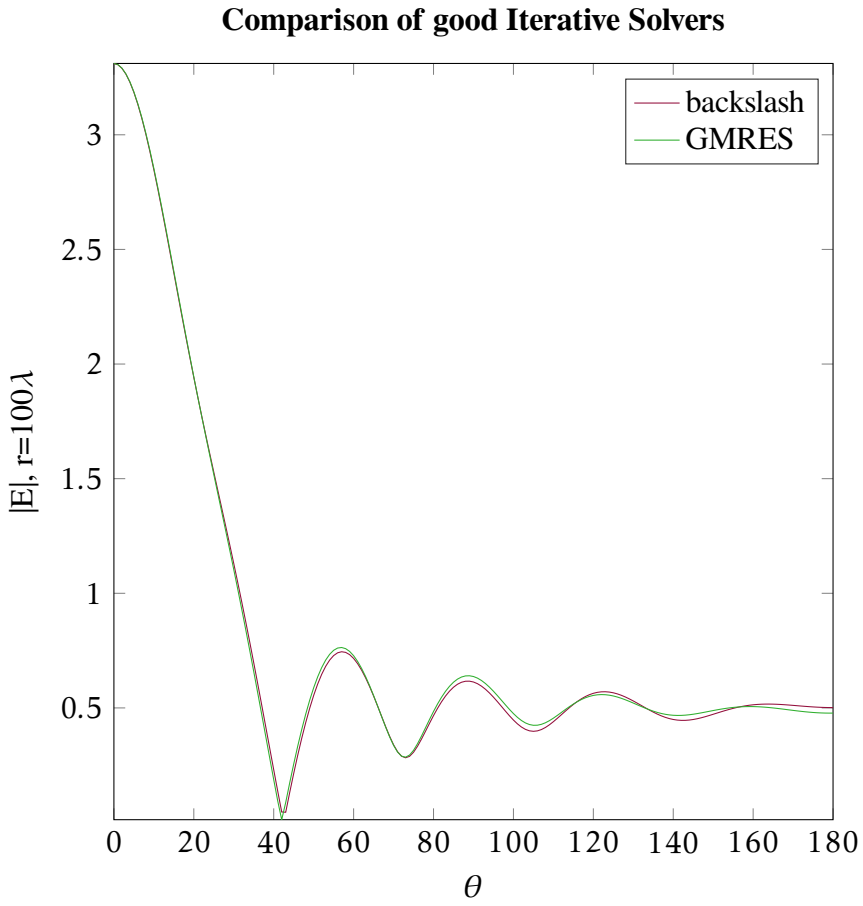


Figure 4.10: 3680 elements

Solver	Seconds	
GMRES	62.908	
Backslash	51.569	

Table 4.5: 10 Restarts, 10 Iterations

As we see, the GMRES is slower now than before. Although if we had an even bigger system (for example 30000 elements) it would be, probably, faster than the direct solver.

§ 4.11. Convergence Plots

When we are dealing with Iterative solvers, a useful figure is the one about the convergence. It would be good if we checked the residuals of in every iterations of our two good solvers, GMRES and PBicGMstab. We will check for the following case:

Solver	Seconds
GMRES	5 Restarts,10 Iterations,10^{-12} Tolerance
Backslash	20 Iterations,10^{-12} Tolerance

Table 4.6: Convergence Table

For the above the convergence plot is:

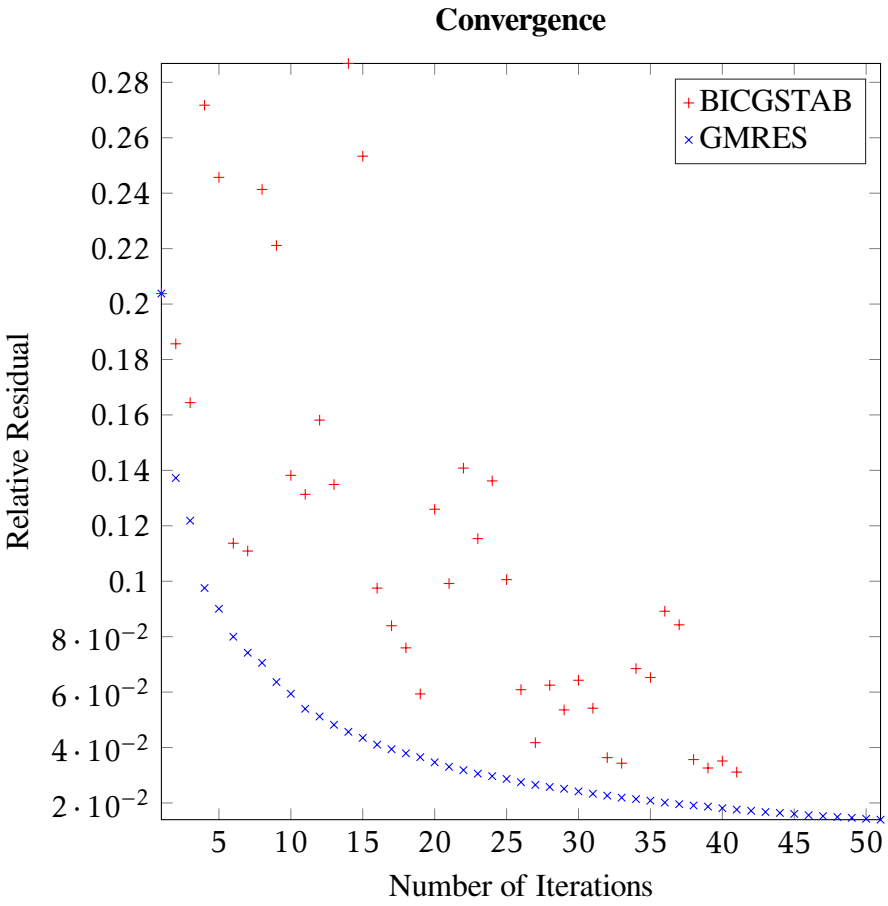


Figure 4.11: 3680 elements

Let's check for more iterations, according to the table **??** just to see the convergence. Obviously if we try to implement the solvers for more iterations, they will need more time, probablly more than the direct solver, but as we said before, they will be faster for huge matrices.

As we see both solvers converge. Although GMRES converge in a more smoother way. Of course they can't converge in 10^{-12}, but as we see in 4.12, they decay to 10^{-2}.

Solver	Seconds
GMRES	10 Restarts,10 Iterations,10^{-12} Tolerance
Backslash	50 Iterations,10^{-12} Tolerance

Table 4.7: Convergence Table

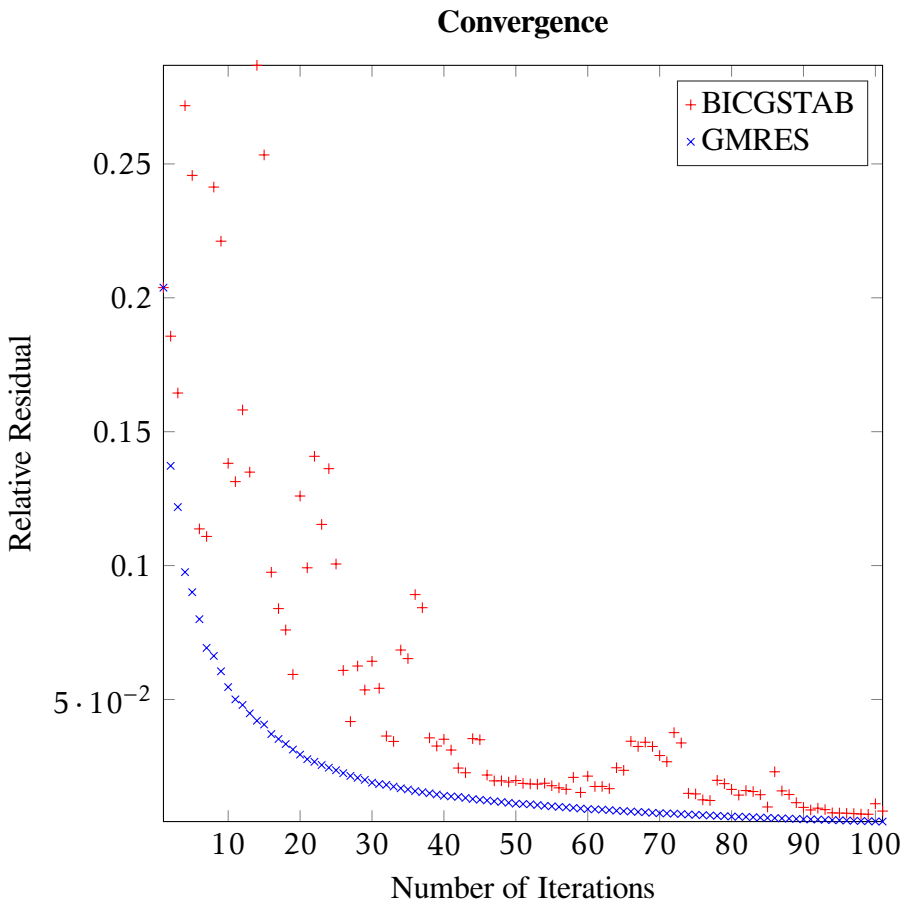


Figure 4.12: 3680 elements

CHAPTER 5

CONCLUSION

In this project we studied a variety of direct and iterative methods of solving linear symmetric systems that are produced from the electric field integral equation for perfect electric conductors. We conclude to the following list.

- For small systems always use backslash. It is faster than all of the direct methods and more accurate than any iterative method.
- If you want to solve the same system many times, for different incident fields (vector $\vec{b}$) then use SVD if those systems are many (over 100) or LU decomposition (between 10 and 100). This is because you will decompose the matrix only once. This, also if the matrix is small.
- If you have a big system and you don't care so much about the accuracy, but you want a solution as fast as possible use the restarted generalized minimal residual method. It's much faster than the backslash solver for big systems, and far more accurate than any other iterative solvers. Depending of the size of the matrix, increase the number of iterations and the number of restarts.
- For all of the above, note that the matrix is not ill conditioned. We don't know what will happen if the matrix is ill-conditioned, so don't follow this way of thinking for an ill-conditioned matrix. For example if we tried to solve the magnetic field integral equation, which has a bigger condition number than the electric field integral equation, we don't know if we would have received the same results. Especially for the iterative methods.

BIBLIOGRAPHY

[1] Integral Equation Methods for Electromagnetic and Elastic Waves – Chew,Tong,Hu – Morgan & Claypool publishers.

[2] Introduction to Scattering of Electromagnetic Waves – Chrisoulids – Tziolas.

[3] Electromagnetic Scattering by Surfaces of Arbitary Shape – Rao Wilton Glisson – IEEE Transactions of Antennas and Propagation, Vol. AP-30, NO.3, MAY 1982.

[4] Solution of scattering and radiation problems in flat layered media with the method of Integral Equations – Polimeridis – PhD thesis, AUTH.

[5] DEMCEM – Polimeridis.

[6] Mathworks.