BACHELOR

Solving the Grad-Shafranov equation
a finite element method

Derks, M.J.W.

Award date:
2015

Link to publication
Solving the Grad-Shafranov equation
A finite element method

Author:
Milou Derks
0774040
m.j.w.derks@student.tue.nl

Supervisors:
Artur Palha
Federico Felici
Barry Koren

June 20, 2015
Due to the growing energy consumption and global warming new forms of clean energy are needed. Nuclear fusion could be a solution to this problem. Nuclear fusion is the process of hydrogen nuclei that collide and form helium nuclei. Energy is released in this process. Nuclear fusion can only take place under extremely high temperatures and no material on earth can contain a plasma of these high temperatures. This is why tokamaks are used. These torus shaped installations contain the plasma using magnetic fields so the plasma will not touch the boundaries. This plasma is unstable, this is why research is needed to better understand the behaviour of the plasma so we can control it in a more efficient way.

To better understand the behaviour of the plasma mathematical simulations are used. In these simulations partial differential equations play a large part. Especially the Grad-Shafranov Equation, which describes the equilibrium between the plasma pressure and the force of the magnetic field, is important. But this equation can not be solved analytically. That is why we approximate the Grad-Shafranov Equation numerically. We do this by using the tool FEniCS. FEniCS is a tool that helps solve PDEs using the Finite Element method.

To solve the Grad-Shafranov Equation using FEniCS we first need to change our problem from a continuous problem to a variational problem. We do this by multiplying the Grad-Shafranov Equation with a test function $v$. Then we integrate over a domain $\Omega$. We simplify the equation using integration by parts. Using the appropriate boundary conditions we obtain equations of the form:

$$\text{find } u \in H^1(\Omega): a(u, v) = F(v).$$  \hfill (1)

We continue by discretization of the space $V = H^1$ using finite elements. This results in a subspace $V_h \subset V$. This space consists of a mesh of triangles with nodes and in every node the value of the exact solution is approximated using polynomials of a fixed degree.

A program is made in python to approximate the exact solution. Increasing the mesh size or the degree of the polynomial results in a better approximation of the exact solution. Also increasing the degree of the polynomial has more effect then increasing the mesh size. To obtain the optimal effect both mesh size and degree of the polynomial should be increased. By making graphs and plots of the solution and error we concluded that our program gives the desired results. We therefore conclude that we made a correct solver for the Grad-Shafranov Equation.
Contents

1 Introduction 3

2 Differential Operators 5
   2.1 The Gradient of a scalar field .......................... 5
       2.1.1 Conditions for differentiability .................... 5
   2.2 The Divergence Operator .................................. 6
       2.2.1 Some properties of the Divergence Operator .......... 6
   2.3 The Laplacian .................................................. 6

3 Introduction to Functional Analysis 8
   3.1 Function Spaces .............................................. 8
   3.2 Weak Formulation of Poisson’s Equation ................... 10
       3.2.1 Dirichlet Boundary Conditions ....................... 11
       3.2.2 Neumann Boundary Conditions ........................ 13
       3.2.3 Mixed and Non-Homogeneous Boundary Conditions .... 14

4 The Application of the Finite Element Method to Elliptic Problems 16
   4.1 Introduction .................................................... 16
   4.2 Fundamental Properties ...................................... 17
       4.2.1 Existence, Uniqueness and Stability of the solution \( u_h \) .......... 17
       4.2.2 Convergence of \( u_h \) to the exact solution \( u \) .................. 17
       4.2.3 Convergence to the exact solution ...................... 18
   4.3 Application to Poisson’s Equations .......................... 18
       4.3.1 Dirichlet Boundary Conditions ........................ 18
       4.3.2 Neumann Boundary Conditions ........................ 21
       4.3.3 Mixed Boundary Conditions ............................. 22

5 Method of Manufactured Solution 24
   5.1 Example of MMS for the Poisson Equation ................... 24
   5.2 Example of MMS for the Grad-Shafranov Equation .......... 25

6 The Grad-Shafranov Equation 26
   6.1 Implementation of the Poisson Equation .................... 26
       6.1.1 Investigate the error .................................... 27
   6.2 Implementation of the Grad-Shafranov Equation .......... 30

7 Conclusion 34
Chapter 1

Introduction

The world population is expanding rapidly. Due to this and technological developments we are consuming more and more energy. To meet this demand we are using a lot of fossil fuels. But there are various cons about using fossil fuels. The amount of fossil fuels is decreasing rapidly, there will only be enough energy out of fossil fuels to live up to our high demands for a few decades. Also the use of fossil fuels is very bad for the environment and it is contributing a lot to global warming. This is why other sustainable energy methods are needed to meet our demand on the long term and also to reduce the problem of global warming. At this moment a lot of sustainable energy comes from wind, sun and water. But there is another method of gaining ‘clean’ energy: Nuclear Fusion.

Nuclear fusion is the process of atomic nuclei that collide with each other under very high temperatures. The atomic nuclei form a new atomic nucleus. If the atomic nuclei have a mass that is less than iron then there will be a release of energy. In nuclear fusion reactors hydrogen nuclei collide with each other, resulting in helium nuclei and a release of energy. This nuclear fusion can only take place under very high temperatures. These temperatures can be up to a few million degrees. The nuclear fusion takes place is a plasma. But there is no material on earth that can bare these high temperatures. To solve this problem the tokamak was invented.

A tokamak is a device in the shape of a torus that keeps the plasma from touching the boundaries using magnetic fields. But the plasma is very unstable and sometimes it hits the boundary of the device. Than there will be no more nuclear fusion in the tokamak until it is on the right temperature again and this consumes a lot of energy. So right now we are not gaining any energy by nuclear fusion because the plasma is so unstable. To solve this problem a better understanding of the plasma and how it interacts with the magnetic field in the tokamak is needed. That is why models are needed to gain more understanding about how the plasma changes in time.

To describe these changes partial differential equations are used. Especially the Grad Shafranov Equation plays a large role in these models. But solving the Grad-Shafranov Equation can be tricky. That is why we will investigate and try to numerically solve the Grad Shafranov Equation in this report. We will start by looking at the Poisson Equation, this because it is similar to the Grad Shafranov Equation but it is a bit simpler and there is much more documentation about it.
We will start by introducing some operators and function spaces we will need in the analysis of the Partial Differential Equations. Than we will give an introduction into PDE’s and their different types of boundary conditions. We will continue by introducing a method to solve PDE’s numerically. And eventually we will construct an equation solver for the Grad Shafranov Equation.
Chapter 2

Differential Operators

In this chapter we will introduce some necessary Differential Operators that we will use along this report. For a more complete understanding of Differential Operators one can have a look at Calculus by Tom M. [1]

2.1 The Gradient of a scalar field

The gradient of a differentiable and scalar-valued function \( f(x_1, \ldots, x_n) \), with \( (x_1, \ldots, x_n) \in \Omega \subset \mathbb{R}^n \) is the vector-valued function composed of the \( n \) partial derivatives of \( f \).

The gradient \( \nabla f \) is a vector field defined at each point \( a \in \Omega \) where the partial derivatives exist. The vectors of the gradient point in the direction of the greatest rate of increase of the function. The magnitude of the gradient is the rate of change of \( f \) in that direction. The derivative \( f'(a; y) \) is the variation of \( f \) in the direction \( y \) and can be represented using the gradient as follows:

\[
f'(a; y) = \sum_{k=1}^{n} \frac{\partial}{\partial x_k} f(a)y_k = (a) \cdot y
\]

with \( y = \sum_{i=1}^{m} y_i e_i \).

2.1.1 Conditions for differentiability

If a function \( f \) at a point \( a \in \Omega \) is differentiable then the partial derivatives exist. However the existence of the partial derivatives of \( f \) at a point \( a \in \Omega \) does not necessarily imply that \( f \) is differentiable at the point \( a \). Nevertheless, a sufficient condition for differentiability exists:

**Differentiability.** Let \( a \) be a given point in \( \mathbb{R}^n \) and \( r \) be a given positive number. Then \( B(a) \) is the set of all points \( x \) in \( \mathbb{R}^n \) such that \( \|x - a\| < r \). \( B(a) \) is then called the open \( n \)-ball of center \( a \) and radius \( r \). If all the partial derivatives exist in some \( n \)-ball \( B(a) \) and are continuous at \( a \). Then \( f \) is differentiable at \( a \).

**Proof.** The proof can be found in Calculus by Tom M. Apostol, section 8.13, page 261, proof of theorem 8.7 [1]
2.2 The Divergence Operator

Let \( X(\Omega) \) be the space of differentiable vector fields in \( \Omega \subset \mathbb{R}^n \) and \( \mathbf{F}(x_1,\ldots,x_n) = \sum_{i=1}^{n} F_i(x_1,\ldots,x_n) \mathbf{e}_n \in X(\Omega) \), with \((x_1,\ldots,x_n) \in \Omega\). The divergence operator, \( \nabla \cdot \), is a mapping:

\[
\nabla \cdot : X(\Omega) \rightarrow V(\Omega),
\]

(2.2)

where \( V(\Omega) \) is the space of scalar functions in \( \Omega \), defined by:

\[
\nabla \cdot \mathbf{F} := \sum_{i=1}^{n} \frac{\partial F_i}{\partial x_i}.
\]

(2.3)

2.2.1 Some properties of the Divergence Operator

**Linearity**

The divergence operator is a linear operator. Given \( \mathbf{F}, \mathbf{G} \in X(\Omega) \)

\[
\nabla \cdot (a \mathbf{F} + b \mathbf{G}) = a \nabla \cdot \mathbf{F} + b \nabla \cdot \mathbf{G},
\]

(2.4)

(2.5)

with \( a, b \in \mathbb{R} \) and \( \mathbf{F}, \mathbf{G} \) differentiable vector fields.

**Leibniz Rule**

When \( \varphi \in V(\Omega) \) is a differentiable scalar field then:

\[
\nabla \cdot (\varphi \mathbf{F}) = \varphi \nabla \cdot \mathbf{F} + \nabla \varphi \cdot \mathbf{F}.
\]

(2.6)

**Gauss’ Theorem**

The Gauss’ Theorem expresses a relationship between the flow (flux) of a vector field through a closed surface and the behavior of the vector field inside the volume enclosed by the surface. It states that the outward flux of a vector field through a closed surface is equal to the volume integral of the divergence over the region enclosed by the surface.

**Gauss’ Theorem.** Let \( W \subset \mathbb{R}^n \) be a volume which is compact and has a piecewise smooth boundary \( \partial W \) and let \( \mathbf{n} \) be the unit outer normal to \( \partial W \). If \( \mathbf{F} \) is a continuously differentiable vector field defined in \( W \), we have

\[
\iiint_{W} (\nabla \cdot \mathbf{F}) dW = \iint_{\partial W} \mathbf{F} \cdot \mathbf{n} d\partial W.
\]

(2.7)

2.3 The Laplacian

Given a twice differentiable scalar function \( f \) defined in \( \Omega \subset \mathbb{R}^n \), the laplacian of \( f \), \( \Delta \), is a mapping:

\[
\Delta : C^k(\Omega) \rightarrow C^{k-2}(\Omega),
\]

(2.8)
defined as:

\[
\Delta f := \sum_{i=1}^{n} \frac{\partial^2 f}{\partial x^i}.
\]

(2.9)

The Laplacian can be written as the divergence of the gradient of \( f \):

\[
\Delta f = \nabla \cdot (\nabla f)
\]

(2.10)
Introduction to Functional Analysis

The role of functional analysis in applied sciences and mathematics is increasing rapidly. Also in our problem functional analysis will play a big role. This because it forms the mathematical basis for the Finite Element Method which we will use later. For this reason we will now look at some important function spaces, norms and theorems that will play an important role in the numerical solution of PDEs with the Finite Element Method.

3.1 Function Spaces

In this section we will discuss some important function spaces that will play a role when using the Finite Element Method. A function space $V$ is a set of function from $X$ to $Y$, equipped with additional structure that makes it a space. For a more complete understanding of function spaces and Functional Analysis one can use Kreyszig [2] as a guideline.

**Definition 1. Linear Vector Space**

A linear vector space, \{V_i\}, is a set of elements, which may be added and multiplied by scalars \{a_i\} in such a way that the followint axioms hold:

\begin{align*}
    u + (v + w) &= (u + v) + w \\
    u + v &= v + u \\
    \exists 0 \in V, \text{ such that: } v + 0 &= v, \forall v \in V \\
    \forall v \in V, \exists -v, \text{ such that: } v + (-v) &= 0 \\
    a(bv) &= (ab)v \\
    1v &= v \\
    a(u + v) &= au + av \\
    (a + b)v &= av + bv.
\end{align*}

**Definition 1. Normed Space**

A normed space $X$ is a vector space on which a norm is defined. In a normed space the alge-
algebraic operations of a vector space are applicable. A correct norm has the following properties:

\[ \|x\| \geq 0 \quad (3.9) \]
\[ \|x\| = 0 \iff x = 0 \quad (3.10) \]
\[ \|\alpha x\| = |\alpha| \|x\| \quad (3.11) \]
\[ \|x + y\| \leq \|x\| + \|y\| \quad (3.12) \]

The norm also expresses the length of a vector.

**Definition 2. Inner Product Space**

An inner product space is a vector space \( X \) with an inner product defined on \( X \). This means that for every pair of vectors \( x, y \) there is an associated scalar that can be written as:

\[ \langle x, y \rangle \quad (3.13) \]

This is called the inner product of the Inner Product Space \( X \). The inner product is well-defined if it has the following properties:

\[ \langle x + y, z \rangle = \langle x, z \rangle + \langle y, z \rangle \quad (3.14) \]
\[ \langle \alpha x, y \rangle = \alpha \langle x, y \rangle \quad (3.15) \]
\[ \langle x, y \rangle = \langle y, x \rangle \quad (3.16) \]
\[ \langle x, x \rangle \geq 0 \quad (3.17) \]
\[ \langle x, x \rangle = 0 \iff x = 0 \quad (3.18) \]

An example of an inner product space that will play a large role in analyzing PDE’s is the \( L^2 \) space.

**Definition 3. Space \( L^2[a, b] \)**
The \( L^2 \) space is the space of square integrable functions with:

\[ L^2(\Omega) := \{ f : \|f\|_{L^2}^2 < \infty \} \quad (3.19) \]
\[ \text{with } \|f\|_{L^2}^2 := \langle f, f \rangle = \int_{\Omega} f \cdot f \, d\Omega \quad (3.20) \]

The inner product defined on the \( L^2 \) space is:

\[ \langle f, g \rangle = \int_{\Omega} f(x)g(x) \, dx \quad (3.21) \]

If the inner product space is complete with a norm induced by the inner product then it is called a Hilbert space.

**Definition 4. Hilbert Space**

A Hilbert Space is a complete inner product space with a norm induced by the inner product. A complete space is a space wherein every Cauchy sequence of points in \( M \) has a limit which also lies in \( M \).
**Definition 5. Sobolev Space**

A sobolev space, \( W^{k,p} \), is a complete normed space wherein the functions as well as their derivatives up to a given order \( k \) belong to the \( L^p \) space.

Let us now introduce the Sobolev space \( W^{1,2} \):

\[
W^{1,2} := H^1(\Omega) = \{ v \in \Omega \rightarrow R : v \in L^2(\Omega), \nabla \in L^2 \} \tag{3.22}
\]

**Theorem 1. Cauchy-Schwarz Inequality**

The Cauchy-Schwarz Inequality is a useful inequality that states that:

\[
\left| \int_{\mathbb{R}^n} f(x)g(x)dx \right|^2 \leq \int_{\mathbb{R}^n} |f(x)|^2dx \cdot \int_{\mathbb{R}^n} |g(x)|^2dx \tag{3.23}
\]

**Proof.** For the proof the book of Kreyszig can be consulted. \( \square \)

### 3.2 Weak Formulation of Poisson’s Equation

In this chapter we will focus on two-dimensional problems. However the generalization to three dimensions follows the same structure. A partial differential equation is an equation of the form:

\[
Au_{xx} + 2Bu_{xy} + Cu_{yy} + Du_{x} + Eu_{y} + F = 0. \tag{3.24}
\]

The partial differential equation is called elliptic if it satisfies the condition \( B^2 - AC < 0 \). In other words, the discriminant is smaller than zero.

- The PDE is called hyperbolic if \( B^2 - AC > 0 \).
- The PDE is called parabolic if \( B^2 - AC = 0 \).

In this chapter we will consider the Poisson equation, which is an elliptic partial differential equation of the form:

\[
\begin{align*}
  u_{xx} + u_{yy} &= f, \tag{3.25} \\
  \text{which can also be written as:} \quad \Delta u &= f \text{ in } \Omega, \tag{3.26} \\
  &\text{in which } \Omega \text{ is an open and bounded set, and } \partial\Omega \text{ is its boundary.} \tag{3.27}
\end{align*}
\]

To find a \( u \) that satisfies this equation we need to have some information on the boundary conditions. This can be the exact value of the function \( u \) at the boundary or the value of the normal derivative at the boundary, or a mix of these two. In general it is difficult to find a closed form solution \( u \) for these problems. Therefore we formulate an alternative formulation for our problem, the so called weak formulation, which will allow us to construct a numerical approximation of our problem using the finite element method. The weak formulation is considered with solving integral functions instead of solving a partial differential equation. The integral function contains the differential equations implicitly, but it is easier to solve numerically. Also in the strong formulation the boundary conditions need to be satisfied by the solution. In the weak formulation they need to be satisfied on an average sense. The so called weak boundary conditions.
In this chapter we will formulate the weak formulation of the Poisson equation for these various types of boundary conditions and provide a way to numerically solve the differential problem by solving the associated weak integral formulation. For a more complete understanding of this, one can consult chapter 3 of Numerical Models for Differential Problems.

### 3.2.1 Dirichlet Boundary Conditions

Dirichlet boundary conditions are conditions in which the value of the displacement of the solution $u$ at the boundary $\partial \Omega$ is known.

$$u = g \text{ on } \partial \Omega. \quad (3.28)$$

If $g=0$ the conditions are said to be homogeneous.

#### Homogeneous boundary conditions

In this section we look at the Poisson equation in which the values of the solution at the boundary are known. Consider a domain $\Omega \subset \mathbb{R}^n$ with boundary $\partial \Omega$. The problem is then:

$$-\Delta u = f \text{ in } \Omega, \quad (3.29)$$

$$u = 0 \text{ on } \partial \Omega. \quad (3.30)$$

To reduce the order of the strong equation from a second-order differential problem to a first-order one-integral problem (the weak formulation) we multiply the equation with a test function $v$ and integrate on the interval $\Omega$. This gives us:

$$-\int_{\Omega} \Delta u v \, d\Omega = \int_{\Omega} f v \, d\Omega. \quad (3.31)$$

For the integrals to make sense $-\int_{\Omega} \Delta u v \, d\Omega < \infty$ and $\int_{\Omega} f v \, d\Omega < \infty$.

Out of the Cauchy Schwarz Inequality stated in Theorem 1 follows that if $\int_{\Omega} \Delta u \, d\Omega < \infty$, $\int_{\Omega} vv \, d\Omega < \infty$ and also $\int_{\Omega} ff \, d\Omega < \infty$, then the integral $-\int_{\Omega} \Delta u v \, d\Omega = \int_{\Omega} f v \, d\Omega$ is well defined.

So from this it follows that $\forall v \in L^2(\Omega), f \in L^2$ and $\Delta u \in L^2(\Omega)$.

In order to eliminate the second order derivative we use integration by parts. Integration by parts yields:

$$\nabla(qv) = (\nabla \cdot q)v + q \cdot \nabla v \Leftrightarrow (\nabla \cdot q)v = \nabla \cdot (qv) - q \cdot \nabla v \quad (3.32)$$

$$\Rightarrow \int_{\Omega} (\nabla \cdot q)v \, dv = \int_{\Omega} \nabla \cdot (qv) \, dv - \int_{\Omega} q \cdot \nabla v \, dv. \quad (3.33)$$

$$\nabla(qv) = (\nabla \cdot q)v + q \cdot \nabla v \Leftrightarrow (\nabla \cdot q)v = \nabla \cdot (qv) - q \cdot \nabla v \quad (3.32)$$

$$\Rightarrow \int_{\Omega} (\nabla \cdot q)v \, dv = \int_{\Omega} \nabla \cdot (qv) \, dv - \int_{\Omega} q \cdot \nabla v \, dv. \quad (3.33)$$

$$\nabla(qv) = (\nabla \cdot q)v + q \cdot \nabla v \Leftrightarrow (\nabla \cdot q)v = \nabla \cdot (qv) - q \cdot \nabla v \quad (3.32)$$

$$\Rightarrow \int_{\Omega} (\nabla \cdot q)v \, dv = \int_{\Omega} \nabla \cdot (qv) \, dv - \int_{\Omega} q \cdot \nabla v \, dv. \quad (3.33)$$

$$\nabla(qv) = (\nabla \cdot q)v + q \cdot \nabla v \Leftrightarrow (\nabla \cdot q)v = \nabla \cdot (qv) - q \cdot \nabla v \quad (3.32)$$

$$\Rightarrow \int_{\Omega} (\nabla \cdot q)v \, dv = \int_{\Omega} \nabla \cdot (qv) \, dv - \int_{\Omega} q \cdot \nabla v \, dv. \quad (3.33)$$
Using Gauss Theorem, we obtain:

\[ \int_{\Omega} (\nabla \cdot \mathbf{q}) v \, dv = \int_{\partial \Omega} (\mathbf{q} v) \cdot \mathbf{n} \, dS - \int_{\Omega} \mathbf{q} \cdot \nabla v \, dv. \]  

(3.35)

If we apply this to the left hand side of our equation we get:

\[ - \int_{\Omega} \Delta u v \, d\Omega = \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega - \int_{\partial \Omega} v \nabla u \cdot \mathbf{n} \, d\gamma. \]  

(3.36)

Because we have a homogeneous Dirichlet problem we need to choose a test function \( v \) that is zero at the boundary. Therefore the last term of equation (3.36) vanishes. If we substitute the result we get at (3.36) in (3.31) we obtain the following equation:

\[ \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega = \int_{\Omega} f v \, d\Omega. \]  

(3.37)

For these integrals to make sense, the functions, but also their derivatives must be square integrable. Also the integrals must be finite. Therefore, \( \nabla u \in L^2(\Omega) \) and \( \nabla v \in L^2(\Omega) \). This means that \( v \in H^1(\Omega) \) (Definition 5).

Now on the boundary \( \partial \Omega \) the test function \( v \) must be 0. This gives us the following space:

\[ H^1_0(\Omega) = \{ v \in H^1(\Omega) : v = 0 \text{ on } \partial \Omega \}. \]  

(3.38)

And therefore the weak formulation of our original problem will be:

\[ \text{find } u \in H^1_0(\Omega): \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega = \int_{\Omega} f v \, d\Omega, \forall v \in H^1_0(\Omega). \]  

(3.39)

Now we can rewrite this weak formulation in a more compact way. Therefore we introduce the bilinear form \( a \):

\[ a : H^1_0 \times H^1_0 \rightarrow \mathbb{R}, \quad a(u,v) = \int_{\Omega} \nabla u \nabla v \, d\Omega, \]

(3.40)

\[ F : V \rightarrow \mathbb{R}, \quad F(v) \forall v \in V. \]  

(3.41)

With \( a(\cdot, \cdot) \) a continuous and coercive bilinear form and \( F(\cdot) \) a continuous linear functional. The new formulation of the problem is:

\[ \text{find } u \in H^1_0(\Omega): a(u,v) = F(v). \]  

(3.42)
Non-Homogeneous boundary conditions

We will now consider the same Poisson equation, but now with non-homogeneous boundary conditions. This gives us the following problem:

\[-\Delta u = f \text{ in } \Omega, \tag{3.43}\]
\[u = g \text{ on } \partial \Omega. \tag{3.44}\]

We can relate the case of non-homogeneous boundary conditions to the case of of homogeneous boundary conditions by noticing that if \(u\) is a solution to the non-homogeneous problem, then we suppose to know a function \(R_g\) which is called the lifting of the boundary, such that:

\[R_g \in H^1_g(\Omega) \text{ and } R_g = g \text{ on } \partial \Omega \tag{3.45}\]

in which \(H^1_g(\Omega) = \{v \in H^1(\Omega) : v = g \text{ on } \partial \Omega\}\).

We introduce \(\tilde{u} = u - R_g\), thus \(\nabla u = \nabla \tilde{u} + \nabla R_g\). Now the problem becomes the following:

\[
\text{find } \tilde{u} \in H^1_0(\Omega) : \int_{\Omega} \nabla \tilde{u} \cdot \nabla v \, d\Omega = \int_{\Omega} f v \, d\Omega - \int_{\Omega} \nabla R_g \cdot \nabla v \, d\Omega, \forall v \in H^1_0(\Omega). \tag{3.46}
\]

Let us now rewrite the problem. The new formulation of the problem is:

\[
\text{find } \tilde{u} \in H^1_0(\Omega): a(\tilde{u},v) = F(v). \tag{3.47}
\]

with \(F(v) = \int_{\Omega} f v \, d\Omega - \int_{\Omega} \nabla R_g \cdot \nabla v \, d\Omega\)

3.2.2 Neumann Boundary Conditions

Let us consider the Poisson equation with Neumann boundary conditions. This means the normal derivative on the boundary \(\partial \Omega\) is known:

\[
\frac{\partial u}{\partial n} = \phi \text{ on } \partial \Omega. \tag{3.48}
\]

If \(\phi = 0\) this is called a homogeneous Neumann boundary problem.

Non-Homogeneous boundary conditions

The problem with non-homogeneous boundary conditions becomes the following:

\[-\Delta u = f \text{ in } \Omega, \tag{3.49}\]
\[\frac{\partial u}{\partial n} = \phi \text{ on } \partial \Omega. \tag{3.50}\]

By following the same procedure as before with the Dirichlet boundary conditions we obtain the following equation:

\[
\int_{\Omega} \nabla u \nabla v \, d\Omega - \int_{\partial \Omega} v \nabla u \cdot n \, d\gamma = \int_{\Omega} f v \, d\Omega, \forall v \in H^1(\Omega) \tag{3.51}
\]
Recall that $\frac{\partial u}{\partial n} = \nabla u \cdot n = \phi$, this results in:

$$\int_{\Omega} \nabla u \nabla v \, d\Omega - \int_{\partial\Omega} \phi v \, d\gamma = \int_{\Omega} fv \, d\Omega,$$

(3.52)
in which $v \in H^1(\Omega)$, $u \in H^1(\Omega)$ and $f \in L^2(\Omega)$.

The weak formulation of our problem now becomes:

$$\text{find } u \in H^1(\Omega) : \int_{\Omega} \nabla u \nabla v \, d\Omega = \int_{\Omega} fv \, d\Omega + \int_{\partial\Omega} \phi v \, d\gamma, \; \forall v \in H^1(\Omega).$$

(3.53)

Let us rewrite the problem, the new formulation of the problem is:

$$\text{find } u \in H^1(\Omega) : a(u,v) = F(v).$$

(3.54)

with $F(v) = \int_{\Omega} fv \, d\Omega + \int_{\partial\Omega} \phi v \, d\gamma$

**Homogeneous boundary conditions**

Now if the boundary conditions are homogeneous the Poisson problem now is:

$$-\Delta u = f \text{ in } \Omega,$$

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega.$$

(3.55)

(3.56)

It is easy to see that the weak formulation of our Poisson problem now becomes:

$$\text{find } u \in H^1(\Omega) : \int_{\Omega} \nabla u \nabla v \, d\Omega = \int_{\Omega} fv \, d\Omega \; \forall v \in H^1(\Omega).$$

(3.57)

Let us rewrite the problem, the new formulation of the problem is:

$$\text{find } u \in H^1(\Omega) : a(u,v) = F(v)$$

**3.2.3 Mixed and Non-Homogeneous Boundary Conditions**

We now consider an elliptic problem with mixed. This means that on a part of the boundary $\partial\Omega$, lets say $\Gamma_D$, the value of the solution is known. On another part of the boundary, $\Gamma_N$, we know the value of the normal derivative of our solution $u$. Note that $\partial\Omega = \Gamma_D \cup \Gamma_N$ and $\Gamma_D \cap \Gamma_N = \emptyset$. We can formulate a weak formulation for such problems. Let us consider the following elliptic problem:

$$-\Delta u = f \text{ in } \Omega,$$

$$u = g_D \text{ on } \Gamma_D,$$

$$\frac{\partial u}{\partial n} = g_N \text{ on } \Gamma_N.$$

(3.59)

(3.60)

(3.61)
We start in the same way as before by multiplying the equation with a test function \( v \) and integrating over the domain \( \Omega \). This gives us:

\[
\int_\Omega \Delta u v \, d\Omega = \int_\Omega f v \, d\Omega,
\]

in which \( \Delta u, v, f \in L^2 \). Integration by parts together with Gauss’ theorem gives us:

\[
\int_\Omega \nabla u \cdot \nabla v \, d\Omega - \int_{\partial \Omega} \frac{\partial u}{\partial n} v \, d\gamma = \int_\Omega f v \, d\Omega,
\]

in which \( u \in H^1(\Omega), \forall v \in H^1(\Omega) \) and \( \frac{\partial u}{\partial n} \in L^2(\partial \Omega) \).

Now we split \( \int_{\partial \Omega} \frac{\partial u}{\partial n} v \, d\gamma \) in two integrals. One that integrates over \( \Gamma_D \) and the other over \( \Gamma_N \). The equation can now be rewritten as:

\[
\int_\Omega \nabla u \cdot \nabla v \, d\Omega - \int_{\Gamma_D} \frac{\partial u}{\partial n} v \, d\gamma = \int_\Omega f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\gamma.
\]

On \( \Gamma_D \) the test function \( v \) needs to be zero. And on \( \Gamma_N \) we know that the normal derivative is equal to \( g_N \). This gives us the following result:

\[
\int_\Omega \nabla u \cdot \nabla v \, d\Omega = \int_\Omega f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\gamma.
\]

Now we use a lifting function \( R_{gD} \) to relate the non-homogeneous solution \( u = g_D \) on \( \Gamma_D \) to the homogeneous solution in the same way as we saw in section (3.2.1) about non-homogeneous Dirichlet boundary conditions. We use a lifting function \( R_{gD} \in H^1_{gD}(\Omega) = \{ v \in H^1(\Omega) : v = g_D \text{ on } \Gamma_D \} \). The problem now becomes:

\[
\text{find } \tilde{u} \in H^1_{gD}(\Omega) \text{ such that } \int_\Omega \nabla \tilde{u} \cdot \nabla v \, d\Omega = \int_\Omega f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\gamma - \int_\Omega \nabla R_{gD} \cdot \nabla v \, d\Omega, \forall v \in H^1_{gD}(\Omega).
\]

Let us rewrite the problem, the new formulation of the problem is:

\[
\text{find } \tilde{u} \in H^1_{gD}(\Omega) : a(\tilde{u}, v) = F(v),
\]

with \( F(v) = \int_\Omega f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\gamma - \int_\Omega \nabla R_{gD} \cdot \nabla v \, d\Omega \).
Chapter 4

The Application of the Finite Element Method to Elliptic Problems

In the following sections Numerical Models for Differential Problems was consulted. [3]

4.1 Introduction

In section 3.2 we derived a weak formulation for different types of boundary conditions for the poisson equation. We showed that the Poisson equation can be written in the following form, for all the different boundary conditions:

$$\text{find } u \in V : a(u,v) = F(v) \quad \forall v \in V,$$  \hspace{1cm} (4.0)

with $V$ the appropriate subspace of $H^1(\Omega)$.

The idea behind the Finite Element Method is to use a finite dimensional subspace of $V$, $V_h^p$, instead of $V$ itself. This because in this way we can divide our domain $\Omega$ into a set of non-overlapping subdomains $\Omega_i$, such that $\bigcup_i \Omega_i = \Omega$, and define $V_h^p$ as the space of polynomials of degree up to $p$.

Now let $V_h^p \subset V$, and let $u_h$ be the solution of the problem at the subspace $V_h$. The approximate problem then becomes:

$$\text{find } u_h \in V_h : a(u_h,v_h) = F(v_h) \quad \forall v_h \in V_h,$$  \hspace{1cm} (4.0)

Now let $\{\varphi_j, j = 1, ..., N_h\}$ be a basis for the vector space $V_h$. Now all possible functions in $V_h$ are linear combinations of the $\varphi_j$. Therefore we will show in the following sections that the problem can be reduced to a simpler problem:

$$a(u_h, \varphi_i) = F(\varphi_i), \quad i = 1, ..., N_h.$$  \hspace{1cm} (4.0)

Now because $u_h \in V_h$ all possible solutions are also linear combinations of the basis vectors $\varphi_j$. This means we can write $u_h$ as:
\[ u_h(x) = \sum_{j=1}^{N_h} u_j \varphi_j(x). \] (4.0)

Using this result in equation (4.1), the result then becomes:

\[ a(\sum_{j=1}^{N_h} u_j \varphi_j, \varphi_i) = \sum_{j=1}^{N_h} u_j a(\varphi_j, \varphi_i) = F(\varphi_i), \quad i = 1, \ldots, N_h. \] (4.0)

We introduce a matrix \( A \) called the stiffness matrix. The stiffness matrix \( A \) has elements \( a_{ij} = a(\varphi_j, \varphi_i) \). Furthermore \( f \) is the vector with components \( f_i = F(\varphi_i) \). And \( u \) is the vector with components the unknown coefficients \( u_j \). If we use this notation we can write the problem into a linear system of equations:

\[ Au = f. \] (4.0)

### 4.2 Fundamental Properties

#### 4.2.1 Existence, Uniqueness and Stability of the solution \( u_h \)

To prove the existence, uniqueness and stability of the solution of a PDE, we use Lax-Milgram theorem.

**The Lax-Milgram Theorem 1.** Let \( V \) be a Hilbert space and \( a(\cdot, \cdot) \) a bilinear form on \( V \), which is

1. bounded if \( \exists C < \infty \text{ such that } |a(u,v)| \leq C \|u\|_V \|v\|_V \) and
2. coercive on \( W \in V \) if \( \exists c > 0 \text{ such that } a(u,u) \geq c \|u\|_V^2 \).

Then, for any \( F \in V \), there is a unique solution \( u \in V \) to the equation:

\[ a(u,v) = F(v), \quad \forall v \in V. \] (4.0)

Now our bilinear form \( a(\cdot, \cdot) \) is bounded. This follows directly out of the Cauchy-Schwarz inequality:

\[ |a(u,v)| \leq \|\nabla u\|_{L^2} \|\nabla v\|_{L^2}. \] (4.0)

The bilinear form is also coercive. So the theorem holds and therefore there exists a unique and stable solution to the problem.

#### 4.2.2 Convergence of \( u_h \) to the exact solution \( u \)

We want to prove that the approximate solution \( u_h \) of the problem converges to the real solution \( u \) of the problem, in the limit \( h \) goes to zero. To do this a lot of work is needed. We can easily prove though that the error we make by using space \( V_h \) is the smallest by using FEM. We will do this by showing that the approximate solution is orthogonal to the exact solution. Therefore we need to prove that:

\[ a(u - u_h, v_h) = 0 \quad \forall v_h \in V_h. \] (4.0)
Proof. Note that $V_h \in V$ and therefore $u$ satisfies the weak problem 4.1 for each element $v_h \in V - h$. And there for the problem takes the form:

$$\text{find } u \in V : \ a(u, v_h) = F(v_h) \ \forall v_h \in V_h.$$  \hspace{1cm} (4.0)

Now we subtract equation 4.2 from the result in 4.2.2, and we obtain:

$$a(u, v_h) - a(u_h, v_h) = F(v_h) - F(v_h) \ \forall v_h \in V_h,$$ \hspace{1cm} (4.1)

$$a(u - u_h, v_h) = 0 \ \forall v_h \in V_h.$$ \hspace{1cm} (4.2)

And thus the solution $u_h$ is orthogonal to the real solution $u$. \hfill \Box

4.2.3 Convergence to the exact solution

The numerical solution $u_h$ does indeed converge to the exact solution $u$. The error we make when numerically approximating $u$ using the finite element method is:

$$\|u - u_h\| \leq c \cdot h^{p+1} = c \cdot \frac{1}{N^{p+1}}$$ \hspace{1cm} (4.2)

with $p$ is the order of the polynomial used to approximate the exact solution and $h$ is the number of refinements of the mesh.

Proof. To prove that $u_h$ does indeed converge to the real solution $u$ a lot more is needed. This prove can be found in Numerical Models for Differential Problems by Alfio Quarteroni in chapter 4.5 [3]. \hfill \Box

4.3 Application to Poisson’s Equations

In this section we will apply the Finite Element Method to the weak formulation of the Poisson equation for different types of boundary conditions. Therefore we introduce the subspace $V_h$ of the space $V$ that approximates the space $H^1(\Omega)$ and depends on parameter $h$. We now construct a partition of $\Omega$ and define a collection of spaces:

$$X^r_h = \{ v_h \in C^0(\Omega) : v_h \big|_K \in P^r, \forall K \in V_h(\Omega) \}, \ r = 1, 2, ...$$ \hspace{1cm} (4.2)

These spaces are all subspaces of $H^1(\Omega)$ and are the spaces of continuous functions that are polynomials of degree lower than or equal to $r$ on the triangle elements of the triangulation $V_h(\Omega)$.

4.3.1 Dirichlet Boundary Conditions

Homogeneous Boundary Conditions

Recall the form of the Poisson problem with homogeneous Dirichlet boundary conditions.

$$-\Delta u = f \text{ in } \Omega,$$ \hspace{1cm} (4.3)

$$u = 0 \text{ in } \partial \Omega.$$ \hspace{1cm} (4.4)
In section 3.2.1 we saw that the problem with Dirichlet boundary conditions can be written as:

\[
\text{find } u \in H^1_0(\Omega): \int \nabla u \cdot \nabla v \, d\Omega = \int f v \, d\Omega, \forall v \in H^1_0(\Omega). \quad \text{(4.4)}
\]

Now let us add the appropriate boundary conditions to the space \( \tilde{X}_h^r \):

\[
\tilde{X}_h^r = \{ v_h \in X_h : v_h |_{\partial \Omega} = 0 \}. \quad \text{(4.4)}
\]

The weak formulation of the problem now becomes:

\[
\text{find } u_h \in \tilde{X}_h^r \int \nabla u \cdot \nabla v \, d\Omega = \int f v \, d\Omega, \forall v_h \in \tilde{X}_h^r. \quad \text{(4.4)}
\]

Now the functions \( v_h \in V_h \) are defined by the values they take in the nodes of the triangle \( N_i \) with \( i = 1, ..., N_h \) of the grid \( \tau_h \).

We can now set a basis for the space \( V_h \) consisting of basis function \( \varphi_j \in V_h, j = 1, ..., N_h \) such that:

\[
\varphi_j(N_i) = \delta^i_j \text{ denoting the Kronecker delta.} \quad \text{(4.4)}
\]

Now a more generic function \( v_h \in V_h \) can be written as a linear combination of the basis functions of \( V_h \):

\[
v_h(x) = \sum_{i=1}^{N_h} v_i \varphi_i(x) \forall x \in \Omega, \quad \text{(4.4)}
\]

with \( v_i \) the value of \( v_h \) at \( N_i \).

Now we can express the solution \( u_h \) in terms of the basis functions \( \varphi_j \):

\[
u_h(x) = \sum_{i=1}^{N_h} u_j \varphi_j(x), \quad \text{(4.4)}
\]

with \( u_j \) is the value of the solution at point \( N_j \).

Now we can write equation 4.3.1 as a system of \( N_h \) linear equations.

\[
\sum_{j=1}^{N_h} u_j \int \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega = \int f \varphi_i \, d\Omega, \quad i = 1, ..., N_h, \quad \text{(4.5)}
\]

\[
\sum_{j=1}^{N_h} a_{ij} u_j = f_i, \quad \text{(4.6)}
\]

with:

\[
A = [a_{ij}] \text{ with } a_{ij} = \int \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega, \quad \text{(4.6)}
\]

\[
u = [u_j], \quad \text{(4.6)}
\]
\( f = [f_i] \) with \( f_i = \int_{\Omega} f \varphi_i \, d\Omega \). (4.6)

We can write the linear system of equations as:

\[ A \mathbf{u} = \mathbf{f}. \] (4.6)

**Non-Homogeneous Boundary Conditions**

Recall the form of the Poisson problem with non-homogeneous Dirichlet boundary conditions as we saw in section 3.2.2.

\[
-\Delta u = f \text{ in } \Omega, \\
u = g \text{ in } \partial \Omega.
\] (4.7) (4.8)

We proceed in the same way as with homogeneous boundary conditions, but we add the lifting function \( g_h \) which is the lifting of the boundary data from the function \( g \). To represent the values of \( g_h \), we approximate the value of the true function \( g \) on the boundary nodes. Therefore we add a set of boundary nodes. We state that there are \( N_h \) internal nodes of the grid \( \tau_h \) and that there are \( N^b_h \) nodes in total. This will result in the following function for \( g_h \):

\[
R_{g_h}(x) = \sum_{i=N_h+1}^{N^b_h} g(N_i) \varphi_i(x) \quad \forall x \in \Omega. \] (4.8)

Now we proceed the same way as in section 3.2.2 by introducing an approximate solution \( \tilde{u}_h = \bar{u}_h + R_{g_h} \). The finite element formulation of the Poisson problem now is:

\[
\text{find } \bar{u}_h \in V_h : \int_{\Omega} \nabla \bar{u}_h \cdot \nabla v_h \, d\Omega = \int_{\Omega} f v_h \, d\Omega - \int_{\Omega} \nabla R_{g_h} \cdot \nabla v_h \, d\Omega \quad \forall v_h \in V_h.
\] (4.8)

If we proceed in the same way as in the case of homogeneous Dirichlet boundary conditions we obtain:

\[
\sum_{j=1}^{N^b_h} u_j \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega = \int_{\Omega} f \varphi_i \, d\Omega - \int_{\Omega} \nabla \varphi_{j+N_h} \cdot \nabla \varphi_i \, d\Omega. \] (4.8)

Let us define the following vectors and matrix:

\[
\mathbf{u} = [u_j] \text{ with } u_j \text{ is the value of } \tilde{u}_h \text{ at the node } N_j, \] (4.8)

\[
A = [a_{ij}] \text{ with } a_{ij} = \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega, \] (4.8)

\[
f = [f_i] \text{ with } f_i = \int_{\Omega} f \varphi_i \, d\Omega, \] (4.8)

\[
g = [g_i] \text{ with } g_i \text{ is the value of } g \text{ at the node } N_i+N_h, \] (4.8)

\[
B = [b_{ij}] \text{ with } b_{ij} = \int_{\Omega} \nabla \varphi_{j+N_h} \cdot \nabla \varphi_i \, d\Omega \text{ with } i = 1, \ldots, N_h \text{ and } j = 1, \ldots, N^b_h. \] (4.8)

And thus we can write the system of linear equations as:

\[ A \mathbf{u} = \mathbf{f} - B \mathbf{g}. \] (4.8)
4.3.2 Neumann Boundary Conditions

Non-Homogeneous Boundary Conditions

Recall that the Poisson problem with non-homogeneous boundary conditions as we saw in section 3.2.2 has the following form:

\[-\Delta u = f \text{ in } \Omega,\]
\[\frac{\partial u}{\partial n} = \phi \text{ on } \partial \Omega.\]  

(4.9)  
(4.10)

The weak formulation of this Poisson problem is:

\[\int_{\Omega} \nabla u \nabla v \, d\Omega = \int_{\Omega} fv \, d\Omega + \int_{\partial \Omega} \phi v \, d\gamma.\]  

(4.10)

We proceed the same way as in the case of Dirichlet boundary conditions. Now we obtain:

\[N_h \sum_{j=1}^{N_h} u_j \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega = \int_{\Omega} f \varphi_i \, d\Omega + \int_{\partial \Omega} \phi \varphi_i \, d\gamma, \quad i = 1, \ldots, N_h,\]  

(4.11)

\[N_h \sum_{j=1}^{N_h} a_{ij} u_j = f_i + k_i.\]  

(4.12)

Let us define the following vectors and matrix:

\[\mathbf{u} = [u_j] \text{ with } u_j \text{ is the value of } \tilde{u}_h \text{ at the node } N_j,\]  

(4.12)

\[A = [a_{ij}] \text{ with } a_{ij} = \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega,\]  

(4.12)

\[\mathbf{f} = [f_i] \text{ with } f_i = \int_{\Omega} f \varphi_i \, d\Omega,\]  

(4.12)

\[\mathbf{k} = [k_i] \text{ with } k_i = \int_{\Omega} \phi \varphi_i \, d\gamma.\]  

(4.12)

And thus we can write the system of linear equations as:

\[A \mathbf{u} = \mathbf{f} + \mathbf{k}.\]  

(4.12)

Homogeneous Boundary Conditions

Recall that the Poisson problem with homogeneous Neumann boundary conditions has the following form:

\[-\Delta u = f \text{ in } \Omega,\]
\[\frac{\partial u}{\partial n} = 0 \text{ on } \partial \Omega.\]  

(4.13)  
(4.14)

21
The weak formulation of this Poisson problem is:

\[ \int_{\Omega} \nabla u \cdot \nabla v \, d\Omega = \int_{\Omega} fv \, d\Omega + \int_{\partial \Omega} \phi v \, d\gamma. \]  

(4.14)

We proceed the same way as in the case of Neumann non-homogeneous boundary conditions. Now we obtain:

\[ \sum_{j=1}^{N_h} u_j \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega = \int_{\Omega} f \varphi_i \, d\Omega \quad i = 1, ..., N_h, \]  

(4.15)

\[ \sum_{j=1}^{N_h} a_{ij} u_j = f_i. \]  

(4.16)

Let us define the following vectors and matrix:

\[ \mathbf{u} = [u_j] \text{ with } u_j \text{ is the value of } \tilde{u}_h \text{ at the node } \mathbf{N}_j, \]  

(4.16)

\[ A = [a_{ij}] \text{ with } a_{ij} = \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega, \]  

(4.16)

\[ \mathbf{f} = [f_i] \text{ with } f_i = \int_{\Omega} f \varphi_i \, d\Omega. \]  

(4.16)

And thus we can write the system of linear equations as:

\[ Au = f. \]  

(4.16)

4.3.3 Mixed Boundary Conditions

Recall the form of the Poisson equation for mixed boundary conditions as we saw in section 3.2.3:

\[ -\Delta u = f \text{ in } \Omega, \]  

(4.17)

\[ u = g_D \text{ on } \Gamma_D, \]  

(4.18)

\[ \frac{\partial u}{\partial n} = \phi \text{ on } \Gamma_N. \]  

(4.19)

In chapter 3 we saw that we can rewrite this problem in the following form:

\[ \text{find } \tilde{u} \in H^1_{\text{gd}}(\Omega) \text{ such that } \int_{\Omega} \nabla \tilde{u} \cdot \nabla v \, d\Omega = \int_{\Omega} f v \, d\Omega + \int_{\Gamma_N} g_N v \, d\gamma - \int_{\Omega} \nabla R_{gd} \cdot \nabla v \, d\Omega \quad \forall v \in H^1_{\text{gd}}(\Omega). \]  

(4.19)

We will use the nodes \( N_{h+1} - N_h^t \) to approximate the boundary \( \Gamma_D \) and the nodes \( N_{h+t+1} - N_{h+t}^s \) to approximate the boundary \( \Gamma_N \). We proceed in the same way as in section 3.2.3. We obtain:

\[ \sum_{j=1}^{N_h^t} u_j \int_{\Omega} \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega = \int_{\Omega} f \varphi_i \, d\Omega + \int_{\partial \Omega} \phi \varphi_i + N_{h+t} \, d\gamma - \int_{\Omega} \nabla \varphi_j + N_h \cdot \nabla \varphi_i \, d\Omega. \]  

(4.19)
Let us define the following vectors and matrix:

\( \mathbf{u} = [u_j] \) with \( u_j \) is the value of \( \bar{u}_h \) at the node \( N_j \), \hspace{1cm} (4.19)

\( A = [a_{ij}] \) with \( a_{ij} = \int_\Omega \nabla \varphi_j \cdot \nabla \varphi_i \, d\Omega \), \hspace{1cm} (4.19)

\( \mathbf{f} = [f_i] \) with \( f_i = \int_\Omega f \varphi_i \, d\Omega \), \hspace{1cm} (4.19)

\( \mathbf{k} = [k_i] \) with \( k_i = \int_\Omega \phi \varphi_i \, d\gamma \), \hspace{1cm} (4.19)

\( \mathbf{g} = [g_i] \) with \( g_i \) is the value of \( g \) at the node \( N_{i+N_h+t} \), \hspace{1cm} (4.19)

\( B = [b_{ij}] \) with \( b_{ij} = \int_\Omega \nabla \varphi_{j+N_h} \cdot \nabla \varphi_i \, d\Omega \) with \( i = 1, \ldots, N_h \) and \( j = 1, \ldots, N_t \). \hspace{1cm} (4.19)

We can now write this system of linear equations as:

\( A\mathbf{u} = \mathbf{f} + \mathbf{k} - B\mathbf{g} \). \hspace{1cm} (4.19)
Chapter 5

Method of Manufactured Solution

Partial Differential Equations, PDE’s, are equations that contain an unknown multi-variable function and its partial derivatives. Some PDE’s can be solved analytically but most cannot be solved analytically. However the analytic solution can be approximated using different numerical methods. But how can we ensure that the numerical method we developed to approximate the solution is a good one? Therefore we can use the method of manufactured solution. The idea of this method is to choose a solution for a particular PDE and then calculate its partial derivatives and boundary conditions. After this we can approximate the solution using the chosen numerical method. Because the real solution is already known, the error of the numerical solution can be calculated.

5.1 Example of MMS for the Poisson Equation

We want to test if our numerical method to solve Poisson’s equation is accurate. Recall that a Poisson pde is of the form: \( u_{xx} + u_{yy} = f(x, y) \). First let us choose a domain in which our calculations will take place. For this example we will choose a rectangle as a domain with \( x \in [0,1] \) and \( y \in [0,\delta] \). Now we choose a solution: \( u(x,y) = \sin(xy) \). Note that we could have chosen any two variable function we wanted, but to explain the MMS we keep it simple. Now let us calculate the partial derivatives:

\[
\begin{align*}
  u(x,y) &= \sin(xy), \\
  \nabla_x u &= y \cos(xy), \\
  \nabla_{xx} u &= -y^2 \sin(xy), \\
  \nabla_y u &= x \cos(xy), \\
  \nabla_{yy} u &= -x^2 \sin(xy).
\end{align*}
\]

Now we substitute \( u_{xx} \) and \( u_{yy} \) into \( u_{xx} + u_{yy} = f(x,y) \). This gives us the following problem with its associated boundary conditions:

\[
\begin{align*}
  \Delta u &= -(x^2 + y^2) \sin(xy), \\
  u(0,y) &= 0, \\
  u(1,y) &= \sin(y), \\
  u(x,0) &= 0, \\
  u(x,\delta) &= \sin(\delta x).
\end{align*}
\]
Now we can approximate the solution using a numerical method of our choice and then check if it is accurate because we know the real solution.

5.2 Example of MMS for the Grad-Shafranov Equation

The Grad Shafranov Equation is of the form:

\[ \nabla \cdot \left( \frac{1}{\mu_0 r} \nabla \psi \right) = j \]  

(5.0)

Cylindrical coordinates \((r,z)\) are used. \(j\) is a function of \((r,z)\).

Following the same structure as for the Poisson Equation we begin by choosing a domain in which our calculations will take place. We choose a rectangle as a domain with \(r \in [1,3]\) and \(z \in [-1,1]\). We choose a solution: \(\psi(r,z) = \sin(rz)\). Now we will calculate the partial derivatives:

\[ \psi(r, z) = \sin(rz), \]  

(5.0)

\[ \nabla_r \psi = z \cos(rz), \]  

(5.0)

\[ \nabla_r \left( \frac{1}{\mu_0 r} \nabla_r \psi \right) = -\frac{z}{\mu_0 r^2} \cos(rz) - \frac{z^2}{r} \sin(rz), \]  

(5.0)

\[ \nabla_z \psi = r \cos(rz), \]  

(5.0)

\[ \nabla_z \left( \frac{1}{\mu_0 r} \nabla_z \psi \right) = -\frac{r}{\mu_0} \sin(rz). \]  

(5.0)

This gives us the following problem with its associated boundary conditions:

\[ \nabla \cdot \left( \frac{1}{\mu_0 r} \nabla \psi \right) = -\frac{z}{\mu_0 r^2} \cos(rz) - \frac{z^2}{r} \sin(rz) + \frac{r}{\mu_0} \sin(rz), \]  

(5.0)

\[ \psi(1, -1) = \sin(-1), \]  

(5.0)

\[ \psi(1, 1) = \sin(1), \]  

(5.0)

\[ \psi(3, -1) = \sin(-3), \]  

(5.0)

\[ \psi(3, 1) = \sin(3). \]  

(5.0)
Chapter 6

The Grad-Shafranov Equation

Now that we analyzed the Poisson Equation we will start looking at the Grad-Shafranov Equation. As mentioned in the introduction, the Grad-Shafranov Equation is used to simulate the evolution of the plasma in a tokamak. It is an equilibrium equation for the plasma in a tokamak. The balance in a tokamak can be seen as the balance between the plasma pressure and the forces of the magnetic field. There are many different forms of the Grad-Shafranov Equation, we will use the following form

\[ \nabla \cdot \left( \frac{1}{\mu_0 r} \nabla \psi \right) = j \]

Unfortunately there aren’t many exact solutions to the Grad Shafranov Equation. Therefore it is necessary to approximate the exact solution as accurate as possible.

To numerically solve the Grad-Shafranov Equation we will use the Finite Element Method. FEniCS is a tool that helps solving PDEs using the Finite Element Method. This tool demands that you first write your Partial Differential Equation into a variational problem. In chapters 2 and 3 we have seen how to set up a variational problem from a PDE, in our case the Poisson Equation. The next step in solving the PDE numerically is to construct a program in which the equation, boundary conditions and appropriate discrete function spaces are defined. In this chapter we will start by developing a program for solving the Poisson Equation. This because the Poisson Equation is very similar to the Grad Shafranov Equation, but it is simpler and there is much documentation about the implementation of a solver for the Poisson Equation. After this we will change our program so it will numerically solve the Grad-Shafranov Equation.

6.1 Implementation of the Poisson Equation

In this chapter we will solve the Poisson Equation numerically. To solve the Poisson Equation using FEniCS we need to transform our problem from a continuous variational problem to a discrete variational problem. To achieve this we use a finite dimensional space $V_h$. We then create a mesh and an element type and let $V$ correspond with our discrete space $V_h$. If we choose polynomials of degree 1 and a triangular mesh, then this space $V_h$ is the space of all
linear functions over a mesh of triangles. We will do this for various functions of which we already know the exact answer. This so we can investigate the error. It is very important to know how large the error is and how it depends on different parameters. These parameters will, of course, affect the accuracy and therefore the error. That is why we will investigate the error for different choices of mesh sizes and polynomial degree.

6.1.1 Investigate the error

In this chapter we will investigate how the error made in solving the Poisson Equation numerically changes for different choices of mesh sizes and polynomial degree. Therefore we start by choosing a function that satisfies the Poisson Equation. We will do this by the method of manufacturing solutions as we saw in chapter 4. We will investigate the error for the function \( u(x, y) = \sin(xy) \). By using the method of manufactured solution in section 5.1 we obtained the following right hand side:

\[
\Delta u = -(x^2 + y^2) \sin(xy)
\]  

(6.0)

To gain some insight in how the solution looks like we will now compute and plot the solution for different choices of refinements and degree of the polynomials.

Let us start with a mesh of 5 refinements in the horizontal and vertical direction. This means that there are 5 nodes in the horizontal and 5 nodes in the vertical direction. And let us use a polynomial of degree 1 to approximate the solution at the nodes. This gives us the following plot for our solution \( u \) (left) and visualization of the error we make in every node (right):

Now we want to investigate how the error changes if we increase the number of refinements and polynomial degree. First we will only change the mesh size to 80 refinements and keep on using a polynomial of degree 1. This results in the following solution and error:

As expected the error gets smaller as we increase the number of refinements. Now let us increase the degree of the polynomial to 5, but keep the mesh size at 5. This results in the
following plot of the solution and the error:

Also in this case the error gets significantly larger and we can see that increasing the order of the polynomial has significantly more effect then increasing the mesh size.

Now let us use a mesh of 80 refinements and a polynomial of order 5. This gives the following plots for the solution and the error:

As you can see the error gets significantly less when we increase the number of refinements and the degree of the polynomials. Also the solution itself gets smoother. To investigate how these factors influence the error we will make several graphs to visualize the decrease of the error when we increase the accuracy.

First we will investigate how the error changes if we increase the number of refinements. We will keep the polynomial at degree 1. Then we make a log plot with the $\log(L^2_{\text{error}})$ on the y-axis and the $\log(\text{refinements}N_h)$ on the x-axis. The expectation is that the graph is a straight line with slope 2. This because when analysing the difference between the exact solution and the approximate solution the error decreases exponentially with a power of $p+1$, as we saw in section 4.2.3. Wherein $p$ is the order of the polynomial. We use a polynomial of degree 1 so we expect the slope to be 2.
As you can see the error gets smaller when the number of refinements increase and the graph becomes a straight line with slope 2. Now we will continue by increasing the order of the polynomials. We will keep the mesh size at 5 refinements in the horizontal and vertical direction. We expect this to be a straight line that decreases when the order of the polynomial increases. This results in the following graph:

We see that when the degree of the polynomial increases the error decreases rapidly. As we can see in the plots the error is almost zero if we use 80 or more refinements and the error also goes to zero when we use a polynomial over order at least 4.

Increasing the degree of the polynomial has more effect than increasing mesh size.
6.2 Implementation of the Grad-Shafranov Equation

Now let us investigate the Grad-Shafranov Equation. We proceed in exactly the same way as in section 6.1. We addept the program so that it solves the Grad-Shafranov equation numerically. We will investigate the error for the function \( \psi(r,z) = \sin(rz) \). In section 5.2 we obtained the following right hand side for the Grad-Shafranov equation:

\[
\nabla \cdot \left( \frac{1}{\mu_0 r} \nabla \psi \right) = -\frac{z}{\mu_0 * r^2} \cos(rz) - \frac{z^2}{\mu_0 * r} \sin(rz) + \frac{-r}{\mu_0} \sin(rz),
\]

(6.0)

We will start by computing the solution (left) and the error (right) for a mesh size of 5 refinements in horizontal and vertical direction and a polynomial of degree 1. This results in the following plots:

Now we increase the mesh size to 80 refinements and we keep the polynomial of degree 1. This results in the following plots for the solution and error:

As you can see the error is smaller than in the case of a mesh of 5 refinements in horizontal and vertical direction. And the solution is smoother. Now let us increase the degree of the polynomial to 4. We will use a mesh of 5 refinements. This gives us the following plots for the solution and error:

We see that increasing the degree of the polynomials results in a smaller error. If we compare the decrease in the error with the decrease in the error of the case where we increased the refinements we see that increasing the degree of the polynomial results in a larger decrease of
the error. If we use a polynomial of order 4 and a mesh size of 80 refinements we obtain the following plots for the solution and the error:

In this plot the error is very small and the solution is smooth. But how fast does the error decrease when increasing the number of refinements or the degree of the polynomial? To investigate this we will make some graphs wherein we fix either the number of refinements or the degree of the polynomial and increase the other parameter. Let us start with investigating the number of refinements for a fixed degree. This results in the following graphs:
All the graphs are straight lines with slopes $p+1$. This is exactly what we expected. This means that our numerical solution converges to the exact solution.

Now let us look at how the error decreases when we fix the number of refinements and increase the degree of the polynomial. The error should decrease exponentially when increasing the degree. This means that in a log plot this should result in a straight line.
What happens here is also the same as with the Poisson Equation, which is what we expected.
Chapter 7

Conclusion

The numerical solution converges to the exact solution. This because our numerical solution satisfies the equation:

$$\|u - u_h\| \leq c \cdot h^{p+1} = c \cdot \frac{1}{N^{p+1}}$$  \hspace{1cm} (7.0)

Also when increasing the degree of the polynomial and fixing the mesh size the error decreases exponentially.

From these results we can conclude that we made a correct solver for the Grad-Shafranov equation. What we can also conclude is that increasing the degree of the polynomial results in a better approximation of the exact solution than increasing the mesh size. If a choice between increasing the mesh size or the degree of the polynomial has to be made, then the last option should be chosen.
Bibliography

[1] Tom M. Apostol, Calculus Volume II: Multi Variable Calculus and Linear Algebra, with applications to Differential Equations and Probability


[5] Blum, J., Le Foll, J., Plasma equilibrium evolution at the resistive time scale