universität
wien

BACHELORARBEIT / BACHELOR'S THESIS

Titel der Bachelorarbeit / Title of the Bachelor's Thesis

Finite elements methods for partial differential equations

verfasst von / submitted by

Tobias Heinzle

angestrebter akademischer Grad / in partial fulfilment of the requirements for the degree of

Bachelor of Science (BSc.)

Wien, Februar 2022 / Vienna, February 2022

Studienkennzahl lt. Studienblatt/ degree programme code as it appears on the student record sheet:	A 033621
Studienrichtung lt. Studienblatt / field of study as it appears on the student record sheet:	Mathematik / Mathematics
Betreuer / Supervisor:	Assoz. Prof. Sara Merino Aceituno, PhD

Acknowledgements

I wish to thank my supervisor for her support, guidance and detailed feedback. Many thanks also to my parents for supporting me at all times and to my friends Rafael Roman and Alexander Posch for their feedback and suggestions.

Abstract

This thesis presents an interesting approach for numerically solving elliptic boundary value problems, namely the finite element method (FEM). We first have a brief look at ordinary differential equations and how to solve them numerically. This shows by example how numerical solutions are inherently always approximations of the true solution and the importance of convergence of such methods. After introducing the basics in this setting, we move on to partial differential equations (PDEs). First we look at the definition of a PDE and some examples, as well as a small demonstration of explicitly solving a PDE. Then we move on to the main subject of the thesis, the numerical treatment of PDEs. As a preparation step, we introduce the idea of finite difference schemes by example of the 5-point formula for the Poisson equation. After this we begin our work towards the finite element method. Before we can talk about the core ideas of FEM, we first need to establish some mathematical tools that will later enable us to introduce the weak or integral form of elliptic boundary value problems (BVP). We make our way towards the Lax-Milgram theorem, which guarantees that the weak form of an elliptic BVP is a well posed problem. At this point we can now give the recipe for the finite element method and illustrate its key concepts with a 1D example. In the last chapter we model airflow around an aircraft wing via the FEM. Throughout the section the model is developed and a concrete finite element scheme is derived. After presenting the numerical results computed with the Python implementation of the scheme, a few error bounds and estimates are stated. In the appendix the full source code is provided.

Contents

1 Introduction

Before we discuss methods for partial differential equations, it is natural to to illustrate the basic concept of numerical solutions in the simplest setting. Namely we will discuss equations of a function $u : \mathbb{R} \to \mathbb{R}$. But first we will define in general an ordinary differential equation. The content of this first chapter is mainly sourced from [6].

Definition. (Ordinary differential equation) We choose $\|.\|$ to be the euclidean norm on $\mathbb{R}^p$ and $\Omega \subset \mathbb{R}^p$ an open bounded domain. Let $f : [t_0, \infty) \times \Omega \to \mathbb{R}$ be a continous function that satisfies the Lipschitz condition

$$|f(t, x) - f(t, y)| \leq \lambda \|x - y\| \quad \forall x, y \in \Omega$$

where λ is a positive real value and independent of x and y. Let $u^{(k)}$ denote the k-th order derivative of u. Then the problem

$$u^{(p)} = f(t, u, u', ..., u^{(p-1)}),\ t \geq t_0$$

is called an ordinary differential equation of order p written in the explicit form. If we specify initial conditions for u and its derivatives up to order $p - 1$, we obtain

$$u^{(p)} = f(t, u, u', ..., u^{(p-1)}),\ t \geq t_0, \text{ and } \forall k \in \{0, 1, ..., p-1\} \quad u^{(k)}(t_0) = u_k \in \mathbb{R}$$

We also require that $u_0 = (u(t_0), u'(t_0), ...u^{(k)}) \in \Omega$. This is called an initial value problem(IVP).

In the situation from the above definition, the theorem of Picard-Lindelöf (see [1]) guarantees the existance of a unique solution u at least in an intervall $[t_0, t_0+T]$ for some $T > 0$ if f obeys the Lipschitz condition in some neighborhood $U_{u_0} \subset \mathbb{R}^p$. The existence of a unique solution that continuously depends on the initial data u_0 implies that the problem is well posed. Despite the guarantee that a solution exists it is often impossible to find explicit solutions for more complicated problems. Fortunately however there exists a vast array of methods that make it possible to compute the solutions numerically. We will now introduce the most basic of these schemes.

1.1 Eulers method

Consider the ordinary differential equation of first order

$$y' = f(t, y), \quad t \geq t_0, \text{ and } y(t_0) = y_0 \in \mathbb{R} \tag{1.1}$$

To set the stage for Eulers method, we note that from 1.1 we can tell two bits of information. For one we know the value of y at time t_0, and by plugging it into $f(t, y)$ we get the slope of y at t_0. We can now try to guess the value of $y(t_0 + h)$ for some small time step h by using a linear approximation. By integrating 1.1 we get

$$y(t) = y(t_0) + \int_{t_0}^{t} f(\tau, y(\tau))d\tau \approx y_0 + (t - t_0)f(t_0, y_0)$$

if we assume that $f(t, y(t)) \approx f(t_0, y(t_0))$ for $t \in [t_0, t_0 + h]$. We now introduce the sequence $(t_n)_{n\in\mathbb{N}}$ where $t_n = nh$ for some small time step h and we let y_n be the numerically approximation of the exact solution $y(t_n)$. The Euler method is no given by

$$y_{n+1} = y_n + hf(t_n, y_n) \quad \text{for} \quad n \in \mathbb{N}$$

This is the most basic way to numerically compute solutions for ordinary differential equations, in Figure 1 the true solution of $u' = u;\ u(0) = 1$ is compared to the approximate solution for different step sizes h. We will now discuss in more detail why we can "trust" the results of this computation.

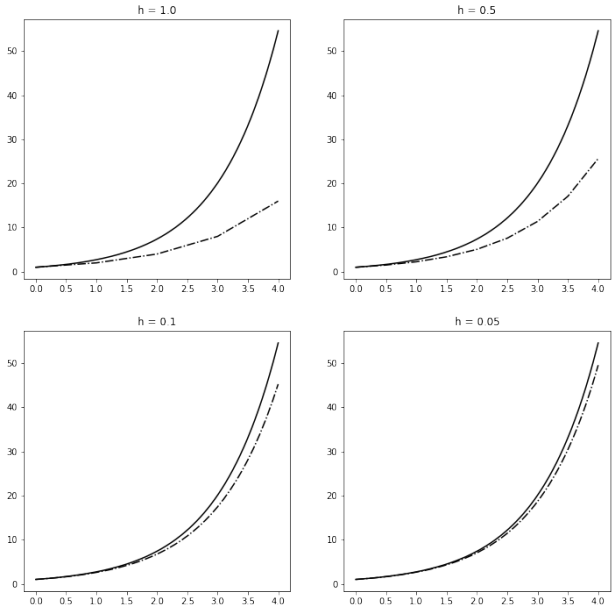

Figure 1: Comparison of the true solution u (solid line) and the approximate solution u_h (dotted line) on the interval $[0, 4]$ for different step sizes h

1.2 Convergence of Eulers method

Before we continue on to more advanced topics, it is important to state that for a method to be of any use we first and foremost need to make sure that the computed solutions do in fact converge to the true solution of the ODE. This needs to be veriefied for every method before employing it.

Definition. (Convergence) A numerical method is convergent if the error $\max_n \|y_n - y(t_n)\|$ in the Euclidian norm tends to zero for $h \to 0$, so if

$$\lim_{h\to 0} \max_n \|y_n - y(t_n)\| = 0$$

For the Euler method, we can prove the convergence directly in the right setting. We will assume $f(t, y)$ is analytic for simplicity. In general this is not the case, but the convergence of the Euler method can be proven for an f that is continously differentiable.

Definition. (Analytic, [7]) A function $f : \mathbb{R} \to \mathbb{C}$ is called *analytic in the point* $x_0 \in \mathbb{R}$ if there exists open disk $D_r(x_0) := \{x \in \mathbb{R} : |x - x_0| < r\}$ for some $r > 0$ and a power series with a radius of convergence $R \geq r$ such that

$$f(x) = \sum_{k=0}^{\infty} a_k (x - x_0)^k \quad \text{for} \quad x \in D_r(x_0)$$

Further, f is called *analytic on* $M \subseteq \mathbb{R}$ if f is analytic in every point of $M \subseteq \mathbb{R}$.

We are now ready to proof the convergence auf Eulers method for the assumption that f is analytic in every argument [6].

Theorem 1.1. *For the Euler method, the norm of the error tends to zero as* $h \to 0_+$.

Proof. Let $h > 0$ and $y_n = y_{n,h}$ be the numerically computed approximations of $y(t_n)$ for $n \in \{0, 1, ..., \lfloor T/h \rfloor\}$. By $e_{n,h} = y_{n,h} - y(t_n)$ we denote the error vector and thus we want to prove that $\lim_{h\to 0+} \max_n \|e_{n,h}\| = 0$. Now we use Taylor's theorem and $y' = f(t, y)$ to get

$$y(t_{n+1}) = y(t_n) + hy'(t_n) + O(h^2) = y(t_n) + hf(t_n, y(t_n)) + O(h^2) \qquad (1.2)$$

Note that y is continously differentiable by the theorem of Picard-Lindelöf and this enables us to bound the $O(h^2)$ term. In fact it holds that $O(h^2) < ch^2$ for some constant $c > 0$, see [6]. Subtracting 1.2 from 1.1 we obtain

$$e_{n+1,h} = e_{n,h} + h[f(t_n, y(t_n) + e_{n,h}) - f(t_n, y(t_n))] + O(h^2)$$

Taking the norm of the above and employing the triangle inequality as well as $O(h^2) < ch^2$ we get

$$\|e_{n+1,h}\| \leq \|e_{n,h}\| + \|h[f(t_n, y(t_n) + e_{n,h}) - f(t_n, y(t_n))]\| + ch^2$$

and since f satisfies the Lipschitz condition we can find the bound

$$\|e_{n+1,h}\| \leq (1 + h\lambda)\|e_{n,h}\| + ch^2 \tag{1.3}$$

for $n \in \{0, 1, ..., \lfloor T/h \rfloor - 1\}$. We now want to use the above result to proof

$$\|e_{n,h}\| \leq \frac{c}{\lambda} h[(1 + h\lambda)^n - 1] \quad \text{, for} \quad n \in \mathbb{N} \tag{1.4}$$

This will be done by induction:
$n = 0$)
It is trivial that $\|e_{n,h}\| \leq \frac{c}{\lambda} h[(1+h\lambda)^0 - 1] = 0$ is satisfied, as $y(t_0) = y_{t_0}$ always and thus $e_{0,h} = 0$ for all h.
$n \geq 1$)
Assume that 1.4 is true up to n. Plugging in 1.4 into 1.3 we get

$$\|e_{n+1,h}\| \leq (1 + h\lambda)\frac{c}{\lambda} h[(1 + h\lambda)^n - 1] + ch^2 = \frac{c}{\lambda} h[(1 + h\lambda)^{n+1} - 1]$$

This means 1.4 holds for all n and since $h\lambda > 0$ we have $1 + h\lambda < e^{h\lambda}$ and therefore $(1+h\lambda)^n < e^{nh\lambda}$. Since we are only interested in the $n \in \{0, 1, ..., \lfloor \frac{T}{h} \rfloor\}$ we can use the fact that $(1 + h\lambda)^n < e^{\lfloor \frac{T}{h} \rfloor h\lambda} \leq e^{T\lambda}$ and this leaves us with

$$|e_{n,h}\| \leq \frac{c}{\lambda}[e^{T\lambda} - 1]h, \text{ for } n \in \{0, 1, ..., \lfloor \frac{T}{h} \rfloor\}$$

Observe that $\frac{c}{\lambda}[e^{T\lambda} - 1]$ is independent of h and thus it follows that

$$\lim_{\substack{h \to 0 \\ n \in \{0,1,...,\lfloor \frac{T}{h} \rfloor\}}} \|e_{n,h}\| = 0$$

This concludes the proof. □

2 PDEs and finite difference methods for PDEs

We now move from the realm of ordinary differential equations to partial differential equations. However, before taking a dive into the numerical analysis part we first have to introduce some definitions. The definitions and notation are taken from [9]. In the following section we will define $\Omega \subset \mathbb{R}^n$ to be an open domain. First we introduce the gradient ∇.

Definition. (Gradient) The gradient of a function $u : \Omega \to \mathbb{R}$ where $x = (x_1, ..., x_n) \in \Omega \mapsto u(x)$ is defined as

$$\nabla u := (\frac{\partial u}{\partial x_1}, ..., \frac{\partial u}{\partial x_n}) \quad \text{and thus formally} \quad \nabla := (\frac{\partial}{\partial x_1}, ..., \frac{\partial}{\partial x_n})^T$$

Definition. (Multiindices k and ∇^k) A multiindex k is defined as $k := (k_1, ..., k_n)^T \in (\mathbb{N}_0)^n$ and its order $|k| := k_1 + ... + k_n$ and factorial $k! := k_1! \cdot ... \cdot k_n!$. For some vector $y \in \mathbb{R}^n$ we define

$$y^k := y_1^{k1} \cdot ... \cdot y_n^{k_n}$$

For the gradient ∇ we now get

$$\nabla^k u := \frac{\partial^{|k|} u}{\partial x_1^{k_1} \cdot ... \cdot \partial x_n^{k_n}}$$

Now we are ready to define a partial differential equation.

Definition. (Differential operator and partial differential equation) Let $F : \Omega \times \mathbb{R}^s \to \mathbb{R}$ and $k_1, ...k_s$ multiindices. Then

$$L(u) := F(x, \nabla^{k_1} u(x), ..., \nabla^{k_s} u(x))$$

defines a formal differential operator of order $p = \max_{1 \leq i \leq s} |k_i|$ on the open domain Ω. The equation

$$L(u) = 0$$

is called a partial differential equation of order p for the function u.

We will look at two examples:
Example: Consider the Laplace operator

$$\Delta := \nabla^2 = \frac{\partial^2}{\partial x_1^2} + ... + \frac{\partial^2}{\partial x_n^2}$$

Note that $L(u) := \Delta \cdot u$ defines a valid differential operator and

$$L_1(u) = \frac{\partial^2 u}{\partial x_1^2} + ... + \frac{\partial^2 u}{\partial x_n^2} = 0$$

is called the Laplace equation. L_1 is also a linear differential operator since for functions $f, g \in C^2$ and $\lambda, \mu \in \mathbb{R}$ we have $L_1(\lambda f + \mu g) = \lambda L_1(f) + \mu L_1(g)$.
Example: Not all differential operators are linear, consider

$$L_2(u) = \rho \frac{\partial u}{\partial t} + \rho u \cdot \nabla u + \nabla p - \mu \Delta u - f$$

where $u : [0, T] \times \mathbb{R}^3 \to \mathbb{R}^3$ is the flow velocity of some fluid at time t and point x, ρ is the density of the fluid, and p the pressure. Additionally we have some function of external force f as well as a viscosity coefficient $\mu \in \mathbb{R}_+$. This operator defines the incompressible Navier-Stokes equation[4] used for modelling flows in fluids. The terms $\rho \frac{\partial u}{\partial t}$ as well as $-\mu \Delta u$ are in fact linear, but $\rho u \cdot \nabla u$ is not. To see this we again plug in $\lambda f + \mu g$ like in the first example. We now get

$$\rho(\lambda f + \mu g) \cdot \nabla(\lambda f + \mu g) = \rho[\lambda^2 f \nabla f + \lambda \mu f \nabla g + \lambda \mu g \nabla f + \mu^2 g \nabla g]$$

and from here we can easily see that L_2 is in general not linear.

2.1 Explicit solutions of PDEs

For some partial differential equations, it is possible to explicitly calculate a solution. Consider for example

$$\frac{\partial u}{\partial x} + t \frac{\partial u}{\partial t} = u$$

Now assume that u can be written as a product of two functions $X(x)$ and $T(t)$, we get the Ansatz $u = X(x)T(t)$. Plugging in yields

$$\begin{aligned} X'T + tX\dot{T} &= XT \\ \Rightarrow \quad \frac{X'}{X} + t\frac{\dot{T}}{T} &= 1 \\ \Rightarrow \quad \frac{X'}{X} = 1 - t\frac{\dot{T}}{T} &= \lambda \\ \Rightarrow \quad X = \lambda X' \text{ and } \frac{\dot{T}}{T} &= \frac{(1-\lambda)}{t} \end{aligned}$$

and solving these equations yields

$$\begin{aligned} &X(x) = Ae^{\lambda x} \quad T(t) = Be^{(1-\lambda)\log(t)} \\ &u(x,t) = ABte^{\lambda(x-\log(t))} \end{aligned}$$

for some $A, B, \lambda \in \mathbb{R}$. We call this method of solving PDEs separation of variables. Note that the solution we obtained is not the only solution of this

PDE, in fact for any differentiable function $f : \mathbb{R} \to \mathbb{R}$ we have

$$\begin{aligned} &\frac{\partial}{\partial x} t f(\lambda(x - \log(t))) + t \frac{\partial}{\partial t} t f(\lambda(x - \log(t))) \\ &= \lambda t f'(\lambda(x - \log(t))) + t \left(f(\lambda(x - \log(t))) - \lambda \frac{t}{t} f'(\lambda(x - \log(t))) \right) \\ &= t f(\lambda(x - \log(t)) \end{aligned}$$

and thus $u(x, y) = tf(\lambda(x - \log(t)))$ is also a solution.
In practice, solving PDEs explicitly is not a viable approach for more difficult problems. For this reason the equations are mainly solved numerically, using schemes that approximate the true solution u of the problem. There are many approaches how to achieve this and we will now introduce the simplest of such methods.

2.2 Motivating the finite difference method

The remainder of section 2 will present the concept of finite difference methods to numerically solve partial differential equations. As the main source of this part [6] was used. The general idea of finite difference methods is to approximate the partial derivatives occurring in the PDE by linear combinations of function values at discrete points. For this purpose we define a grid on the domain Ω and we are interested in the values of the solution at every grid point. To illustrate this we will start out with an example before discussing the details.
Example: Let us have a look at the 2-dimensional heat equation

$$\frac{\partial u}{\partial t} = \nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}$$

Assume we posses knowledge of the function values of u at some time t_0 at discrete points $x_{i,j}$ for $i, j \in \{0, 1, ..., n\}$ with $x_{i+1,j} = x_{i,j} + (h, 0)$ and $x_{i,j+1} = x_{i,j} + (0, h)$ respectively. We define $u_{i,j} = u(x_{i,j})$ in the following discussion. How can we approximate the second derivatives $\frac{\partial^2 u}{\partial x^2}$ and $\frac{\partial^2 u}{\partial y^2}$? One way to achieve this would be to approximate the first derivative by

$$\frac{\partial u_{i,j}}{\partial x} \approx \frac{u_{i+\frac{1}{2},j} - u_{i-\frac{1}{2},j}}{h}$$

Note that $u_{i+\frac{1}{2},j}$ does not make sense as there is no grid point $x_{i+\frac{1}{2},j}$. We only write this formally as an in-between step. It will disappear as we approximating the second derivative again by the same method

$$\frac{\partial^2 u_{i,j}}{\partial x^2} \approx \left(\frac{\partial u_{i+\frac{1}{2},j}}{\partial x} - \frac{\partial u_{i-\frac{1}{2},j}}{\partial x}\right)\frac{1}{h} \approx \left(\frac{u_{i+1,j} - u_{i,j}}{h} - \frac{u_{i,j} - u_{i-1,j}}{h}\right)\frac{1}{h}$$

So we arrive at the estimate

$$\frac{\partial^2 u_{i,j}}{\partial x^2} \approx \frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2}$$

and analogous for the derivative in respect to y

$$\frac{\partial^2 u_{i,j}}{\partial y^2} \approx \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}$$

We can now use a simple scheme resembling the Euler method to numerically approximate the solution u at the inner grid points $x_{i,j}$ with $i, j \in \{1, 2, ..., n-1\}$. Here we introduce the time difference $\Delta t = t_{k+1} - t_k$, not to be confused with the Laplace operator. The formula

$$\begin{aligned} u_{i,j}^{t_{k+1}} &\approx u_{i,j}^{t_k} + \Delta t \frac{\partial u_{i,j}^{t_k}}{\partial t} = u_{i,j}^{t_k} + \Delta t \left(\frac{\partial^2 u_{i,j}^{t_k}}{\partial x^2} + \frac{\partial^2 u_{i,j}^{t_k}}{\partial y^2} \right) \\ &\approx u_{i,j}^{t_k} + \Delta t \left(\frac{u_{i+1,j}^{t_k} - 2u_{i,j}^{t_k} + u_{i-1,j}^{t_k}}{h^2} + \frac{u_{i,j+1}^{t_k} - 2u_{i,j}^{t_k} + u_{i,j-1}^{t_k}}{h^2} \right) \end{aligned}$$

approximates the next time step $u_{i,j}^{t_{k+1}}$ by a linear combination of values from the last time step. Even though we have conveniently left out some details, this example captures the basic idea of finite differences. We have discretized the problem both in time and space, this means we consider only discrete points in time t_k and discrete points in space $x_{i,j}$.

2.3 Five point formula

We will now in more detail discuss the most basic method for numerically solving PDEs in two space dimensions via finite differences. The main source of this section is [6], please refer there for more details. Approximating the solution u on the discrete grid points $x_{i,j}$ will ultimately involve solving a large sparse system of linear equations. This is a common theme in numerical approximation of PDE solutions and we will later discover that the finite element method also boils down to solving a linear system. For the remainder of the thesis we will discuss Poisson type PDEs, so here we will examine how to solve the Poisson equation with finite differences. Contrary to our last problem, it does not involve any time derivatives, so there are only two independent variables x and y. Consider

$$\nabla^2 u = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = f \quad (x, y) \in \Omega$$

in the special setting where $\Omega \subset \mathbb{R}^2$ is a bounded open domain and its boundary $\partial\Omega$ is piecewise smooth. We take $f = f(x, y)$ as some continuous function on Ω

and assume Dirichlet boundary conditions

$$u(x,y) = \Phi(x,y), \qquad (x,y) \in \partial\Omega$$

This is a so-called boundary value problem (BVP) and the PDE we try to solve is called the Poisson equation. For implementing a simple finite difference method, we now embed a grid in Ω. A simple way to do this is to choose a grid parallel to the axes with equal spacing $\Delta x = h > 0$ in both x and y direction. We choose some starting point $(x_0, y_0) \in \Omega$ and then

$$G := \{(x_0 + kh, y_0 + lh) \in R^2 : (k,l) \in \mathbb{Z}^2\} \cap \overline{\Omega}$$

is the set of all grid points in the closure of Ω. In the following section $u_{k,l}$ will denote the approximation of the value of the exact solution $u(x_0 + kh, y_0 + lh)$. For grid points on $G \backslash \Omega$, we do not need to approximate $u_{k,l}$ as these points lie on the boundary $\partial\Omega$ and we know the values of u there according to our boundary conditions. Further we can distinguish three types of points in G:

- *Boundary points:* These points lie on $\partial\Omega$, the value of u at these points is known from the boundary conditions.
- *Internal points:* These points lie inside of Ω and all of their neighbours in both directions x and y lie inside Ω as well. This means (x_{in}, y_{in}) is an internal point if and only if $(x_{in} + kh, y_{in} + lh) \in \Omega$ for all $k, l \in \{-1, 1\}$.
- *Near boundary points:* These points are inside of Ω but not all of their neighbours.

Let $v = v(x,y), \quad (x,y) \in \Omega$ be at least C^2, then we have the following estimates for its derivatives along x and y directions at internal points $(x_{in}, y_{in}) = (x_0 + kh, y_0 + lh)$

$$\frac{\partial^2 v}{\partial x^2}(x_{in}, y_{in}) = \frac{u_{k-1,l} - 2u_{k,l} + u_{k+1,l}}{h^2} + \mathcal{O}(h^2)$$
$$\frac{\partial^2 v}{\partial y^2}(x_{in}, y_{in}) = \frac{u_{k,l-1} - 2u_{k,l} + u_{k,l+1}}{h^2} + \mathcal{O}(h^2)$$

$v_{k,l}$ in the above equation is the value of v at the internal grid point $(x_0 + kh, y_0 + lh)$, for details about the approximations refer to [6]. Thus we can estimate ∇^2 up to order $\mathcal{O}(h^2)$ via

$$\frac{(u_{k-1,l} - 2u_{k,l} + u_{k+1,l}) + (u_{k,l-1} - 2u_{k,l} + u_{k,l+1})}{h^2}$$

Plugging back into the Poisson equation leaves us with

$$\frac{u_{k-1,l} + u_{k+1,l} + u_{k,l-1} + u_{k,l+1} - 4u_{k,l}}{h^2} = f_{k,l}$$

where $f_{k,l} = f(x_0 + kh, y_0 + lh)$. Note that we are using a linear combination of values of u at 5 grid points to calculate the approximation for ∇^2 at the internal points, hence the name of the scheme. For each internal point we have obtained a linear equation

$$u_{k-1,l} + u_{k+1,l} + u_{k,l-1} + u_{k,l+1} - 4u_{k,l} = h^2 f_{k,l} \tag{2.1}$$

Let $N = |G \cap \Omega|$ be the total number of non boundary grid points and I_j a series that maps $1 \leq j \leq N$ onto our non boundary grid points $G \cap \Omega$ such that $I_j = (x_0 + k_j h, y_0 + l_j h)$. 2.1 now defines a system of linear equations and the solution vector $U = (u_{I_j})_{1 \leq j \leq N}$ is the desired approximation of the exact solution of the Poisson equation. In practice, I_j rearranges the the grid points into a one-dimensional column vector $U \in \mathbb{R}^N$. Thus we can write our system of equations as

$$AU = b$$

with $b = h^2(f_{I_j})_{1 \leq j \leq N} \in \mathbb{R}^N$ and A the matrix containing the coefficients of the 5 point scheme. The task of approximating the solution of the Poisson equation has now been reduced to solving this system.
It is possible to prove that the matrix A is nonsingular and thus the system $AU = b$ has a unique solution [6]. Furthermore, for square domains with equidistant grid points we also have the following theorem:

Theorem 2.1. *Let $e_{k,l}$ be the error of the five-point formula at the k,l-th grid point, thus $e_{k,l} = u_{k,l} - \tilde{U}_{k,l}$ where $u_{k,l}$ and $\tilde{U}_{k,l}$ denote the exact and calculated solution (mapped back to the original indices) at the k,l-th grid point respectively. If f and Φ are sufficiently smooth, then there exists a $c > 0$ independent of h such that*

$$\|\boldsymbol{e}\| \leq ch^2 \qquad h \to 0$$

The error $\boldsymbol{e}$ is measured in the Euclidean norm.

For a proof of this theorem please consult [6]. The above result means that the approximate solution does converge to the true solution as we refine the mesh, i.e. as $h \to 0$.

3 Finite element methods for PDEs

In the following section we will discuss the finite element method for approximating solutions of elliptic boundary value problems. The majority of this section is sourced from[10] unless stated otherwise. When discussing the finite difference method in the last section, we noted that the discretization of the problem occurs in the domain of the independent variables. This means we find a discrete set of grid points in our domain for which we want to find the values of our solution. In contrast, the finite element method discretizes the function space in which we search for our solution. This is considerably more involved mathematically, but provides a flexible set of tools for dealing with elliptic boundary value problems defined on complex shaped domains. In practice, discretizing the function space will most often also involve discretizing the domain in some way, but we have much more freedom in doing so. More precisely the domain will be decomposed into a finite set of elements (hence the name) over which we define simple basis functions through which we seek to approximate the true solution. This yields a method that breaks down the big problem of solving a BVP into solving many smaller problems over the individual elements. Since there are only finitely many of them, we call this the finite element method. Finite elements are most often employed when working with elliptic PDEs, thus we will focus the discussion exclusively on them. The first part of the section will deal with some general theoretical foundations, we will then further investigate how elliptic equations can be studied by transforming them into a weak integral formulation. From this we will yield the method of finite elements. Finally we will treat a 1D example and establish a connection back to finite difference schemes. This chapter will specifically showcase how we go about discretizing our solution space, and in the next chapter we will then focus on how the FEM "divides" up the work into small contributions of the individual elements.

3.1 Sobolev spaces

When discussing PDEs, it is important to precisely choose the spaces in which we seek the solutions. We sill work towards introducing Sobolev spaces, a class of function spaces that will serve as suitable solution spaces later on.
Before we can introduce them we need to motivate the notion of weak derivative. For a brief summary of the mathematical theory used here please refer to the section 5.1 in the appendix.

Definition. Let u be a locally integrable function on Ω, precisely $u \in L_1(\omega)$ for each bounded open set ω with $\overline{\omega} \subset \Omega$. If there exists a function ω_α that is also

locally integrable on Ω and satisfies

$$\int_\Omega \omega_\alpha(x)v(x)\ dx = (-1)^\alpha \int_\Omega u(x)D^\alpha v(x)\ dx \quad \forall v \in C_0^\infty(\Omega)$$

we call ω_α a weak derivative of u with order $|\alpha|$.

Let f and g both be weak derivatives of u and consider

$$\begin{aligned}\int_\Omega f(x)v(x)\ dx - \int_\Omega g(x)v(x)\ dx &= 0 \\ \rightarrow \int_\Omega (f(x) - g(x))v(x)\ dx &= 0\end{aligned}$$

As a consequence of the following lemma they must be equal almost everywhere and thus a weak derivative of u is unique.

Lemma 3.1. *(DuBois Reymond's lemma)[10] Suppose that ω is locally integrable on an open set $\Omega \subset \mathbb{R}^n$, then*

$$\int_\Omega \omega(x)v(x)\ dx = 0 \quad \forall v \in C_0^\infty(\Omega) \Rightarrow \omega(x) = 0 \quad \textit{almost everywhere}$$

If $u \in C^k(\Omega)$, its weak derivatives D^α of order $|\alpha| \leq k$ are thus the corresponding classical partial derivatives

$$\frac{\partial^{|\alpha|}u}{\partial x_1^{\alpha_1}...\partial x_n^{\alpha_n}}$$

From the above definition of the weak derivative it is not immediately obvious that the weak derivative is a very natural extension of the regular notion of derivatives. To illustrate this we will prove the following proposition about weak derivatives of continuous piecewise differentiable functions.

Proposition 3.2. *(Weak derivative of a continuous, piecewise differentiable function on $\mathbb{R}$) Let $x_1, ..., x_n \in \mathbb{R}$ with $x_1 < ... < x_n$ be a finite sequence of points that partitions $\mathbb{R}$ into semi open intervals of the form $I_j := (x_j, x_{j+1}]$ where $x_0 = -\infty$ as well as $x_{n+1} = \infty$. The last interval is defined as the open interval $I_n = (x_n, \infty)$. Let now f be a continuous function on $\mathbb{R}$ with*

$$f := f_j(x) \quad \textit{for} \quad x \in I_j, \quad j \in \{0, ..., n\}$$

where $f_j \in C^1(I_j)$. Then the weak derivative $f^{(1)}$ of f is given by

$$f^{(1)} := (f_j)'(x) \quad \textit{for} \quad x \in I_j, \quad j \in \{0, ..., n\}$$

Proof. Let $v \in C_0^\infty(\mathbb{R})$, using this we can calculate

$$\begin{aligned}
\int_{\mathbb{R}} f(x)v'(x)\ dx &= \sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} f_i(x)v'(x)\ dx \\
&= \sum_{i=0}^{n} \left([f_i(x)v(x)]_{x_i}^{x_{i+1}} - \int_{x_i}^{x_{i+1}} f_i'(x)v(x)\ dx \right) \\
&= \sum_{i=0}^{n} [f_i(x)v(x)]_{x_i}^{x_{i+1}} - \sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} f_i'(x)v(x)\ dx \\
&= \sum_{i=0}^{n} (f_i(x_{i+1})v(x_{i+1}) - f_i(x_i)v(x_i)) - \sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} f_i'(x)v(x)\ dx \\
&= f_i(\infty)v(\infty) - f_i(-\infty)v(-\infty) - \sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} f_i'(x)v(x)\ dx \\
&= -\sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} f_i'(x)v(x)\ dx \\
&= -\int_{\mathbb{R}} f^{(1)}(x)v(x)\ dx
\end{aligned}$$

Note that the left sum is telescopic and thus only the first and last summand remain. □

We are now prepared to introduce Sobolev spaces.

Definition. (Sobolev space) Let $k \in \mathbb{N}$ and $p \in [1, \infty]$. By D^α we mean a weak derivative of order $|\alpha|$. Then

$$W_p^k(\Omega) := \{u \in L_p(\Omega) : D^\alpha u \in L_p(\Omega), \quad |\alpha| \leq k\}$$

defines a Sobolev space of order k on Ω. For $p \in [1, \infty)$

$$\|u\|_{W_p^k(\Omega)} := \left(\sum_{|\alpha| \leq k} \|D^\alpha u\|_{L_p(\Omega)}^p \right)^{1/p}$$

defines the Sobolev norm on $W_p^k(\Omega)$. In the special case of $p = \infty$ we define it by

$$\|u\|_{W_\infty^k(\Omega)} := \sum_{|\alpha| \leq k} \|D^\alpha u\|_{L_\infty(\Omega)}$$

Of special importance in the remainder of this section are the Sobolev spaces for $p = 2$. For $u, v \in W_2^k(\Omega)$ we can define the scalar product

$$\langle u, v \rangle_{W_2^k(\Omega)} := \sum_{|\alpha| < k} \langle D^\alpha u, D^\alpha v \rangle = \sum_{|\alpha| < k} \int_\Omega D^\alpha u(x) D^\alpha v(x)\ dx$$

In particular, this inner product makes $W_2^k(\Omega)$ into a Hilbert space, we thus denote it by $H^k(\Omega)$. Additionally, we define $H_0^1(\Omega)$ as the closure of $C_0^\infty(\Omega)$ in the $\|\cdot\|_{H^1(\Omega)}$ norm. This means H_0^1 is the set of all $u \in H^1(\Omega)$ such that u is the limit of a sequence of test functions $\{u_m\}_{m\in\mathbb{N}}, \quad u_m \in C_0^\infty(\Omega)$. For a sufficiently smooth boundary $\partial\Omega$ we have

$$H_0^1(\Omega) = \{u \in H^1(\Omega) : u = 0 \ \text{ on } \ \partial\Omega\}$$

Note that $H_0^1(\Omega)$ is again a Hilbert space. Finally, for a real Hilbert space $\mathbb{H}$ (or more generally an inner product space) we can consider its dual space $\mathbb{H}'$

$$\mathbb{H}' := \{f : \mathbb{H} \to \mathbb{R} : f \text{ is linear and } \ \exists\ M > 0 \ \text{ such that } f(u) \le M\|u\|_{\mathbb{H}} \ \forall\ u \in \mathbb{H}\}$$

The dual space $\mathbb{H}'$ can be equipped with a norm $\|\cdot\|'$ defined by

$$\|f\|' := \sup_{v\in\mathbb{H}} \frac{|f(v)|}{\|v\|_{\mathbb{H}}} \quad \text{for } f \in \mathbb{H}'$$

3.2 Poincaré-Friedrichs inequality

In this short section we will introduce and prove the Poincaré-Friedrichs inequality, which will later be used when we show that the weak formulation of an elliptic BVP is indeed well posed. In the proof of the Poincaré-Friedrichs inequality we will use the following result.

Lemma 3.3. *(Cauchy-Schwarz inequality) Let $\mathbb{H}$ be a real Hilbert space and thus*

$$\|u\|^2 = \langle u, u\rangle$$

Then for $u, v \in \mathbb{H}$

$$|\langle u, v\rangle| \le \|u\|\|v\|$$

Proof. Let $\lambda \in \mathbb{R}$, we have

$$\begin{aligned} 0 \le \|u + \lambda v\|^2 &= \langle u + \lambda v, u + \lambda v\rangle \\ = \langle u, u\rangle + \langle \lambda v, u\rangle + \langle u, \lambda v\rangle + \langle \lambda v, \lambda v\rangle \\ &= \|u\|^2 + 2\lambda\langle u, v\rangle + \lambda^2\|v\|^2 \end{aligned}$$

We have obtained a quadratic polynomial $p(\lambda) = a\lambda^2 + b\lambda + c$ with $a, b, c \in \mathbb{R}$. We also know that $p(\lambda) \ge 0 \quad \forall\lambda \in \mathbb{R}$. Thus it can either have one or zero real

solutions, this means

$$b^2 - 4ac \leq 0$$

and thus

$$\begin{aligned} |2\langle u, v\rangle|^2 - 4\|u\|^2\|v\|^2 &\leq 0 \\ \Rightarrow |2\langle u, v\rangle|^2 &\leq 4\|u\|^2\|v\|^2 \\ \Rightarrow |\langle u, v\rangle| &\leq \|u\|\|v\| \end{aligned}$$

□

Lemma 3.4. *(Poincaré-Friedrichs inequality)[10] Let $\Omega \subset \mathbb{R}^n$ be open and bounded with sufficiently smooth boundary $\partial\Omega$. There exists a constant $\hat{c}$ such that for every $u \in H_0^1(\Omega)$ the inequality*

$$\int_\Omega |u(x)|^2 \, dx \leq \hat{c} \sum_{i=1}^{n} \int_\Omega \left| \frac{\partial u}{\partial x_i}(x) \right|^2 \, dx$$

holds.

Proof. For the sake of simplicity, we will only prove the above lemma for the special case of $\Omega = (a, b) \times (c, d) \subset \mathbb{R}^2$. Further, since every $u \in H_0^1(\Omega)$ is the limit of a sequence $\{u_m\}_{m\in\mathbb{N}}$ for $u_m \in C_0^\infty(\Omega)$, it is sufficient to prove it for $u \in C_0^\infty(\Omega)$. Let $c < y < d$, we first observe that

$$u(x, y) = u(a, y) + \int_a^x \frac{\partial u}{\partial x}(t, y) \, dt = \int_a^x \frac{\partial u}{\partial x}(t, y) \, dt$$

since u vanishes on the boundary of Ω. It follows that

$$\int_\Omega |u(x, y)|^2 \, dxdy = \int_a^b \int_c^d \left| \int_a^x \frac{\partial u}{\partial x}(t, y) \, dt \right|^2 \, dydx$$

and applying the Cauchy-Schwarz inequality onto $\left|\int_a^x \frac{\partial u}{\partial x}(t, y) \, dt\right|^2 = \left|\langle 1, \frac{\partial u}{\partial x}(t, y)\rangle\right|^2$ yields

$$\left| \int_a^x \frac{\partial u}{\partial x}(t, y) \, dt \right|^2 \leq \|1\|^2_{L_1((a,x))} \|\frac{\partial u}{\partial x}(t, y)\|^2_{L_1((x,a))}$$

We can now plug this back into the formula and

$$\begin{aligned}
&\int_a^b \int_c^d \left| \int_a^x \frac{\partial u}{\partial x}(t,y)\ dt \right|^2\ dydx \\
&\leq \int_a^b \int_c^d (x-a) \left(\int_a^x \left| \frac{\partial u}{\partial x}(t,y) \right|^2\ dt \right)\ dydx \\
&\leq \left(\int_a^b (x-a)\ dx \right) \cdot \left(\int_c^d \int_a^b \left| \frac{\partial u}{\partial x}(t,y) \right|^2\ dtdy \right)
\end{aligned}$$

Since $|\frac{\partial u}{\partial x}(t,y)|^2 \geq 0$ we can bound the integral from a to x from above by integrating from a to b, which enables the last inequality. After evaluating the first integral, we obtain

$$\int_\Omega |u(x,y)|^2\ dxdy \leq \frac{1}{2}(b-a)^2 \int_\Omega |\frac{\partial u}{\partial x}(x,y)|^2\ dxdy$$

The same way we can achieve the bound

$$\int_\Omega |u(x,y)|^2\ dxdy \leq \frac{1}{2}(d-c)^2 \int_\Omega |\frac{\partial u}{\partial y}(x,y)|^2\ dxdy$$

Now both inequalities can be summed together, and this finally results in

$$\int_\Omega |u(x,y)|^2\ dxdy \leq \hat{c} \int_\Omega |\frac{\partial u}{\partial y}(x,y)|^2 |\frac{\partial u}{\partial x}(x,y)|^2\ dxdy$$

for $\hat{c} = \left(\frac{2}{(b-a)^2} + \frac{2}{(d-c)^2} \right)^{-1}$ □

In particular, for the domain $\Omega = (0,1)^2 \subset \mathbb{R}^2$ it follows that $\hat{c} = \frac{1}{4}$ and for $\Omega = (0,1) \subset \mathbb{R}$ we get $\hat{c} = \frac{1}{2}$

3.3 Lax-Milgram theorem

At this point we are almost ready to introduce the Lax-Milgram theorem. This theorem will prove useful throughout the following section as we investigating elliptic boundary value problems. The proof of the Lax-Milgram theorem will make use of the Riesz representation theorem, so we will state it without proof before moving on.

Theorem 3.5. *(Riesz representation theorem)[11] Let $\mathbb{H}$ a Hilbert space and l a bounded linear fuctional on $\mathbb{H}$. Then there exists a uniquely determined element $u_l \in \mathbb{H}$ such that*

$$l(v) = \langle v, u_l \rangle \quad \forall\ v \in \mathbb{H} \quad \textit{and} \quad \|l\| = \|u_l\|$$

Further, every $u \in \mathbb{H}$ defines a bounded linear functional l_u on $\mathbb{H}$ via

$$l_u(v) = \langle v, u \rangle \quad \forall\ v \in \mathbb{H} \quad \textit{and} \quad \|l_u\| = \|u\|$$

Theorem 3.6. *(Lax-Milgram theorem)[2] Suppose $\mathbb{H}$ a real Hilbert space with norm $\|\cdot\|_{\mathbb{H}}$ and let $a(\cdot,\cdot)$ be a bilinear functional on $\mathbb{H} \times \mathbb{H}$ satisfying*

- $\exists\ c_0 > 0 \quad \forall\ u \in \mathbb{H} \quad a(u,u) \geq c_0 \|u\|_{\mathbb{H}}^2$
- $\exists\ c_1 > 0 \quad \forall\ u, v \in \mathbb{H} \quad |a(u,v)| \leq c_1 \|u\|_{\mathbb{H}} \|v\|_{\mathbb{H}}$

and let $l(\cdot)$ be a linear functional on $\mathbb{H}$ such that

- $\exists\ c_2 > 0 \quad \forall\ u \in \mathbb{H} \quad |l(u)| \leq c_2 \|u\|_{\mathbb{H}}$

Then there exits one and only one $u \in \mathbb{H}$ satisfying

$$a(u,v) = l(v) \quad \forall v \in \mathbb{H}$$

Proof. First note that $|a(u,v)| \leq c_1 \|u\|_{\mathbb{H}} \|v\|_{\mathbb{H}}$. We have that for each $u \in \mathbb{H}$, the linear form $v \mapsto a(u,v)$ is continuous and thus there exists a unique element $Au \in \mathbb{H}'$ such that

$$a(u,v) = Au(v) \quad \forall v \in \mathbb{H}$$

Considering Au under the norm $\|\cdot\|'$ of the dual space, we can see that

$$\|Au(v)\|' = \sup_{v \in \mathbb{H}} \frac{|Au(v)|}{\|v\|_{\mathbb{H}}} \leq c_1 \|u\|$$

We have now shown that the linear map $A \in L(\mathbb{H};\ \mathbb{H}')\ ;\quad u \mapsto Au$ is bounded and thus continuous, in particular

$$\|A\|_{L(\mathbb{H};\ \mathbb{H}')} = \sup_{u \in \mathbb{H}} \frac{\|Au\|'}{\|u\|_{\mathbb{H}}} \leq c_1$$

where $\|\cdot\|_{L(\mathbb{H};\ \mathbb{H}')}$ denotes the operator norm on $L(\mathbb{H};\ \mathbb{H}')$. From the Riesz representation theorem we know that for each $f \in \mathbb{H}'$ there is a unique $u_f \in \mathbb{H}$ such that $f(v) = \langle v, u_f \rangle$. Since this also holds in the reverse direction, there exists a bijective map $\tau : \mathbb{H}' \to \mathbb{H};\ f \mapsto \tau u$ such that

$$\forall f \in \mathbb{H}', \quad \forall v \in \mathbb{H} : \quad f(v) = \langle v, \tau f \rangle$$

This means finding $u \in \mathbb{H}$ such that

$$a(v,u) = l(v)$$

is equivalent to finding u such that

$$\tau A u = \tau l$$

We will show that this equation has a unique solution by establishing that, for an appropriate value $p > 0$, the map

$$U : \mathbb{H} \to \mathbb{H}; \quad v \mapsto v - p(\tau A v - \tau l)$$

is a contraction. First observe that

$$\begin{aligned} \|v - p\tau A v\|_{\mathbb{H}}^2 &= \langle v - p\tau A v, v - p\tau A v\rangle^2 \\ &= \|v\|_{\mathbb{H}}^2 - 2p\langle \tau A v, v\rangle + p^2 \|\tau A v\|_{\mathbb{H}}^2 \end{aligned}$$

Now note that

$$\langle \tau A v, v\rangle = A v(v) = a(v, v) \geq c_0 \|v\|_{\mathbb{H}}^2$$

and by virtue of the Riesz theorem

$$\|\tau A v\|_{\mathbb{H}} = \|A v\|' \leq \|A\|_{L(\mathbb{H};\ \mathbb{H}')} \|v\|_{\mathbb{H}} \leq c_1 \|v\|_{\mathbb{H}}$$

Using both of the above estimates, we can bound from above

$$\|v - p\tau A v\|_{\mathbb{H}}^2 \leq (1 - 2pc_0 + p^2 c_1^2) \|v\|_{\mathbb{H}}^2$$

Since U is a linear map, $U(x) - U(y) = U(x - y)$ and we have thus shown that

$$\|U(x) - U(y)\|_{\mathbb{H}} \leq C \|x - y\|_{\mathbb{H}}$$

for $C = \sqrt{1 - 2pc_0 + p^2 c_1^2}$. This means for $p \in (0, \frac{2c_0}{c_1^2})$ it holds that $0 < C < 1$ and thus U is a contraction. The Banach fixed point theorem guarantees us the existence and uniqueness of a fixed point $u \in \mathbb{H}$ such that

$$\begin{aligned} u = U(u) &= u - p(\tau A u - \tau l) \\ &\Rightarrow \tau A u - \tau l = 0 \end{aligned}$$

and this shows that u is indeed a solution. □

3.4 Elliptic PDEs and boundary value problems

Now we can accurately define the class of differential equations we wish to investigate. The equations are usually accompanied by boundary conditions and the complete problem consisting of both is called a boundary value problem or BVP. To employ the FEM method later on, we want to formulate these BVPs in the so-called weak formulation. This section will explain why and how this is possible and what the advantages are. The main source for this section is [10].

Definition. (Elliptic differential equation) Let Ω be an open bounded subset of $\mathbb{R}^n$, consider the linear second-order partial differential equation

$$-\sum_{i,j=1}^{n} \frac{\partial}{\partial x_j}\left(a_{i,j}(x)\frac{\partial u}{\partial x_i}\right) + \sum_{i=1}^{n} b_i(x)\frac{\partial u}{\partial x_i} + c(x)u = f(x), \quad x \in \Omega$$

with coefficients $a_{i,j} \in C^1(\bar{\Omega})$ for $i, j = 1, ...n$ as well as $b_i \in C(\bar{\Omega})$ for $i = 1, ...n$ and $c \in C(\bar{\Omega})$. Additionally there is a forcing term $f \in C(\bar{\Omega})$. Further we require the uniform ellipticity condition

$$\sum_{i,j=1}^{n} a_{i,j}(x)v_i v_j \geq \hat{c}\sum_{i=1}^{n} v_i^2 \quad \forall v = (v_1, ...v_n) \in \mathbb{R}^n \tag{3.1}$$

for some positive constant $\tilde{c}$ that does not depend on x nor v. The differential equation defined above is called elliptic.

Example: The simplest elliptic equation is the Laplace equation discussed earlier. It is an example for a homogeneous elliptic equation. The Poisson equation given by

$$-\Delta u = f$$
$$\Delta = \sum_{i=1}^{n} \frac{\partial^2}{\partial x_i^2}$$

is the simplest non-homogeneous elliptic PDE.
In practice we are interested in solving a given differential equation with regards to some predefined boundary conditions. These boundary conditions are limitations that are imposed on the solution u at on the boundary $\partial\Omega$. Let g denote some function defined on $\partial\Omega$, the most important types of boundary conditions are the following

- $u = g$ on $\partial\Omega$ (Dirichlet)
- $\frac{\partial u}{\partial \nu} = g$ on $\partial\Omega$, ν denotes the outward normal vector on $\partial\Omega$ (Neumann)
- $\frac{\partial u}{\partial \nu} + \sigma u = g$ on $\partial\Omega$, $\sigma(x) > 0$ on $\partial\Omega$ (Robin)

Consider a general second order elliptic PDE as defined above with the Dirichlet boundary conditions. Precisely

$$-\sum_{i,j=1}^{n} \frac{\partial}{\partial x_j}\left(a_{i,j}(x)\frac{\partial u}{\partial x_i}\right) + \sum_{i=1}^{n} b_i(x)\frac{\partial u}{\partial x_i} + c(x)u = f(x), \quad x \in \Omega \tag{3.2}$$

$$u = 0 \quad \text{on} \quad \partial\Omega \tag{3.3}$$

A function $u \in C^2(\Omega) \cap C(\bar{\Omega})$ that satisfies the above equation as well as the boundary condition is called a classical solution of the problem. We can assume such a solution to exist for sufficiently smooth coeffiecients $a_{i,j}, b_i, c, f$ and boundary $\partial\Omega$. Additionally for the coefficients $a_{i,j}$ the uniform ellipticity condition 3.1 must hold[10]. After establishing the general setting of the problem, let us examine a more concrete case. Consider the Poisson equation with zero Dirichlet boundary conditions

$$\begin{aligned} -\Delta u = \operatorname{sgn}(\frac{1}{2} - |x|), \quad x \in \Omega \\ u = 0, \quad x \in \partial\Omega \end{aligned}$$

for $\Omega = B_1(0) \subset \mathbb{R}^n$ the unit ball. This can in fact not have a classical solution in $C^2(\Omega) \cap C(\partial\Omega)$. If this was the case, Δu would be continuous on Ω, but this cannot be true since $\operatorname{sgn}(\frac{1}{2} - |x|)$ is not continuous. Classical theory for partial differential equations does not cover this complication, but in practice this type of non-smooth data does occur. To overcome these limitations we will now generalize the concept of a solution. Remember that in section two we have defined a partial differential equation via a differential operator L. In the case of our general second order differential equation, we can write the corresponding differential for the left side as

$$L(u) := -\sum_{i,j=1}^{n} \frac{\partial}{\partial x_j}\left(a_{i,j}\frac{\partial u}{\partial x_i}\right) + \sum_{i=1}^{n} b_i\frac{\partial u}{\partial x_i} + cu$$

and thus solving the problem means finding u such that

$$L(u) = f$$

We will now introduce the integral form of a PDE. Suppose u is a classical solution of the above BVP, then for all test functions $v \in C_0^\infty(\Omega)$ we have

$$\int_\Omega L(u)(x)v(x)\ dx = \int_\Omega f(x)v(x)\ dx$$

and thus

$$-\sum_{i,j=1}^{n}\int_\Omega \frac{\partial}{\partial x_j}\left(a_{i,j}\frac{\partial u}{\partial x_i}\right)v\ dx+\sum_{i=1}^{n}\int_\Omega b_i\frac{\partial u}{\partial x_i}v\ dx+\int_\Omega cuv\ dx=\int_\Omega fv\ dx$$

This is called the integral form of our differential equation. We can now do integration by parts in the integrals contained in the first sum, noting that $v=0$ on $\partial\Omega$, which leads us to

$$\sum_{i,j=1}^{n}\int_\Omega a_{i,j}\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_j}\ dx+\sum_{i=1}^{n}\int_\Omega b_i\frac{\partial u}{\partial x_i}v\ dx+\int_\Omega cuv\ dx=\int_\Omega fv\ dx$$

With the last step we have achieved lower differentiability requirements for u, in fact we will no longer have to insist on the solution being in $C^2(\Omega)$. Considering the condition $u=0$ on $\partial\Omega$, it is now natural to seek u in $H_0^1(\Omega)$. Formalizing this notion leads to the following definition.

Definition. (Weak solution of a BVP) Let $a_{i,j},b_i,c\in L_\infty(\Omega),\ i,j=1,...,n$ and $f\in L_2(\Omega)$. We call $u\in H_0^1(\Omega)$ a weak solution of the BVP 3.2 if u satisfies

$$\sum_{i,j=1}^{n}\int_\Omega a_{i,j}\frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_j}\ dx+\sum_{i=1}^{n}\int_\Omega b_i\frac{\partial u}{\partial x_i}v\ dx+\int_\Omega cuv\ dx=\int_\Omega fv\ dx \tag{3.4}$$

$$\forall v\in H_0^1(\Omega) \tag{3.5}$$

further more we call the above formulation a weak form of the BVP 3.2.

Let us observe a few remarks about weak formulations of boundary value problems. It is indeed possible to lower the differentiability requirements even further for our solution u in the special circumstance that L is a linear continuous operator. Consider a differential equation given by such an L and a inhomogeneous term $f\in L_2(\Omega)$. With the scalar product on $L_2(\Omega)$ given by $\langle u,v\rangle:=\int_\Omega u(x)v(x)\ dx$ we can write the integral form of the equation as

$$\langle Lu,v\rangle=\langle f,v\rangle$$

and thus by calculating the adjoint operator L^* for L.

$$\langle Lu,v\rangle=\langle u,L^*v\rangle=\langle f,v\rangle$$

We can then remove any differentiability constraints on u and allow general distributional solutions. This of course has the effect that we now assume the test functions v to be at least twice differentiable. Aside from this remark we will from now on only consider the weak form as defined in 3.4. It is also worth noting

that a classical solution of a BVP is always a solution of the weak form, but the converse does not hold. Before moving on to the discussion of approximating these weak solutions via the finite element method, we have to make sure that the problem indeed posseses a unique weak solution.

Theorem 3.7. *(Existence and uniqueness of solutions) Let the coefficients* $a_{i,j} \in L_\infty(\Omega)$ *for* $i,j = 1,...n$, $b_i \in W^1_\infty(\Omega)$ *for* $i = 1,...,n$, $c \in L_\infty(\Omega)$ *and* $f \in L_2(\Omega)$. *If the second-order elliptic BVP with* $u = 0$ *on* $\partial\Omega$ *satisfies the uniform ellipticity condition 3.1 and the following inequality*

$$c(x) - \frac{1}{2}\sum_{i=1}^{n} \frac{\partial b_i}{\partial x_i} \geq 0 \quad x \in \bar{\Omega}$$

holds, then there exists a unique weak solution $u \in H^1_0(\Omega)$. *Additionally there exists a* $C > 0$ *such that*

$$\|u\|_{H^1(\Omega)} \leq \frac{1}{C}\|f\|_{L_2(\Omega)}$$

Proof. To show this we will employ the Lax-Milgram theorem from the last section. As preparation we introduce some notation

$$\begin{aligned} a(u,v) &:= \sum_{i,j=1}^{n} \int_\Omega a_{i,j} \frac{\partial u}{\partial x_i}\frac{\partial v}{\partial x_j}\, dx + \sum_{i=1}^{n} \int_\Omega b_i \frac{\partial u}{\partial x_i} v\, dx + \int_\Omega cuv\, dx \\ l(v) &:= \int_\Omega fv\, dx \end{aligned}$$

For us to apply the Lax-Milgram theorem with $\mathbb{H} = H^1_0(\Omega)$, we need to make sure that $a(\cdot,\cdot)$ and $l(\cdot)$ satisfie the prerequesites required by the theorem. We will start by showing that $l(\cdot)$ is a bounded linear functional on $H^1_0(\Omega)$, let $\alpha, \beta \in \mathbb{R}$, then

$$\begin{aligned} l(\alpha v + \beta w) &= \int_\Omega f(x)(\alpha v(x) + \beta w(x))\, dx \\ &= \alpha \int_\Omega f(x)v(x)\, dx + \beta \int_\Omega f(x)w(x)\, dx \\ &= \alpha l(v) + \beta l(w) \end{aligned}$$

$$\forall v, w \in H^1_0(\Omega)$$

and by using the Cauchy-Schwarz inequality

$$\begin{aligned}
|l(v)| &= |\int_\Omega f(x)v(x)\ dx| \\
&\leq \left(\int_\Omega |f(x)|^2\ dx\right)^{1/2} \left(\int_\Omega |v(x)|^2\ dx\right)^{1/2} \\
&= \|f\|_{L_2(\Omega)}\|v\|_{L_2(\Omega)} \\
&\leq \|f\|_{L_2(\Omega)}\|v\|_{H^1(\Omega)}
\end{aligned}$$

we have shown that $l(\cdot)$ satisfies the conditions. We will next prove show that $a(v,w)$ is bound from above by the product $\|v\|\|w\|$ as required. It is easily seen that

$$\begin{aligned}
|a(v,w)| \leq &\sum_{i,j=1}^{n} \max_{x\in\bar{\Omega}} |a_{i,j}(x)| \left|\int_\Omega \frac{\partial w}{\partial x_i}\frac{\partial v}{\partial x_j}\ dx\right| \\
&+ \sum_{i=1}^{n} \max_{x\in\bar{\Omega}} |b_i(x)| \left|\int_\Omega \frac{\partial w}{\partial x_i} v\ dx\right| \\
&+ \max_{x\in\bar{\Omega}} |c(x)| \left|\int_\Omega wv\ dx\right|
\end{aligned}$$

and applying the Cauchy-Schwarz inequality yields

$$\begin{aligned}
|a(v,w)| \leq \tilde{c} \Bigg[&\sum_{i,j=1}^{n} \left(\int_\Omega |\frac{\partial w}{\partial x_i}|^2\ dx\right)^{1/2} \left(\int_\Omega |\frac{\partial v}{\partial x_j}|^2\ dx\right)^{1/2} \\
&+ \sum_{i=1}^{n} \left(\int_\Omega |\frac{\partial w}{\partial x_i}|^2\ dx\right)^{1/2} \left(\int_\Omega |v|^2\ dx\right)^{1/2} \\
&+ \left(\int_\Omega |w|^2\ dx\right)^{1/2} \left(\int_\Omega |v|^2\ dx\right)^{1/2} \\
\leq \tilde{c} \Bigg[&\left(\int_\Omega |w|^2\ dx\right)^{1/2} + \sum_{i=1}^{n} \left(\int_\Omega \left|\frac{\partial w}{\partial x_i}\right|^2\ dx\right)^{1/2}\Bigg] \\
&\times \left[\left(\int_\Omega |v|^2\ dx\right)^{1/2} + \sum_{j=1}^{n} \left(\int_\Omega \left|\frac{\partial v}{\partial x_j}\right|^2\ dx\right)^{1/2}\right]
\end{aligned}$$

for $\tilde{c} = \max\left\{\max_{1\leq i,j\leq n} \max_{x\in\bar{\Omega}} |a_{i,j}(x)|, \max_{1\leq i\leq n} \max_{x\in\bar{\Omega}} |b_i(x)|, \max_{x\in\bar{\Omega}} |c(x)|\right\}$. Recall that for positive real values $a_1, ..., a_n$ the follwing inequality holds

$$(a_1^{1/2} + ...a_m^{1/2}) \leq 2n(a_1 + ... + a_n)^{1/2}$$

this can be proven by induction and thus we have

$$|a(v,w)| \leq 2n\tilde{c}\left[\int_\Omega |w|^2\ dx + \sum_{i=1}^n \int_\Omega \left|\frac{\partial w}{\partial x_i}\right|^2\ dx\right]^{1/2} \times \left[\int_\Omega |v|^2\ dx + \sum_{j=1}^n \int_\Omega \left|\frac{\partial v}{\partial x_j}\right|^2\ dx\right]^{1/2}$$

and for $c_1 = 2n\tilde{c}$ we arrive at the desired bound

$$|a(v,w)| \leq c_1 \|w\|_{H_1(\Omega)} \|v\|_{H_1(\Omega)}$$

Our last task is to bound $a(v,v)$ from below by $\|v\|^2_{H^1(\Omega)}$. To achieve this we will leverage the uniform ellipticity condition 3.1 and rewrite $\frac{\partial v}{\partial x_i} v$ as $\frac{1}{2}\frac{\partial}{\partial x_i}(v^2)$, which yields us

$$a(v,v) \geq \hat{c}\sum_{i=1}^n \int_\Omega \left|\frac{\partial v}{\partial x_i}\right|^2\ dx + \sum_{i=1} n \int_\Omega b_i(x)\frac{1}{2}\frac{\partial}{\partial x_i}(v^2)\ dx\text{“} \int_\Omega c(x)|v|^2\ dx$$

The next step is to integrate by parts in the second sum, we can move over the derivative onto b_i and we get

$$a(v,v) \geq \hat{c}\sum_{i=0}^n \int_\Omega \left|\frac{\partial v}{\partial x_i}\right|^2\ dx + \int_\Omega \left(c(x) - \frac{1}{2}\sum_{i=1}^n \frac{\partial b_i}{\partial x_i}\right)|v|^2\ dx$$

Since we have

$$c(x) - \frac{1}{2}\sum_{i=1}^n \frac{\partial b_i}{\partial x_i} \geq 0 \quad x \in \bar{\Omega}$$

we have achieved

$$a(v,v) \geq \hat{c}\sum_{i=0}^n \int_\Omega \left|\frac{\partial v}{\partial x_i}\right|^2\ dx \tag{3.6}$$

Now we are able to employ the Poincaré-Friedrichs inequality, which leaves us with

$$a(v,v) \geq \frac{\hat{c}}{c_*}\int_\Omega |v|^2\ dx \tag{3.7}$$

Adding together 3.6 and 3.7 leads to

$$a(v,v) \geq c_0\left(\int_\Omega |v|^2\ dx + \sum_{i=1}^n \int_\Omega \left|\frac{\partial v}{\partial x_i}\right|^2\ dx\right) = c_0\|v\|^2_{H^1(\Omega)}$$

where $c_0 = \frac{\hat{c}}{1+c^*}$. Now we can guarantee the existence and uniqueness of a weak solution $u \in H_0^1(\Omega)$ via the Lax-Milgram theorem and the only thing that remains to be shown is the bound for u. We already have that

$$\begin{aligned} c_0\|u\|^2_{H^1(\Omega)} \leq a(u,u) = l(u) &= \langle f, u\rangle \\ \leq |\langle f, u\rangle| &\leq \|f\|_{L_2(\Omega)}\|u\|_{L_2(\Omega)} \\ &\leq \|f\|_{H^1(\Omega)}\|u\|_{H^1(\Omega)} \end{aligned}$$

and thus

$$\|u\|_{H^1(\Omega)} \leq \frac{1}{c_0}\|f\|_{H^1(\Omega)}$$

□

3.5 Finite element method for elliptic BVP

Building upon the theory presented first part of section 3, we will now develop the finite element method. We have shown that the weak formulation of elliptic BVPs is indeed a well posed problem and we will now use this to formulate a procedure for numerically approximating solutions of elliptic BVPs. This section is again based on the material presented in [10]. The finite element method involves the following steps:

(1). Convert the problem into its weak form

$$\text{find} \quad u \in V \quad \text{such that} \quad a(u,v) = l(v) \quad \forall v \in V$$

with V being the solution space.

(2). Replace V by a finite-dimensional subspace $V_h \subset V$ of piecewise polynomial functions of fixed degree. We can then consider the approximation

$$\text{find} \quad u_h \in V_h \quad \text{such that} \quad a(u_h, v_h) = l(v_h) \quad \forall v_h \in V_h$$

of the original problem. This step is called discretization of the problem.

(3). Transform the discrete problem into a system of linear equations to find an approximate solution of our equation expressed as a linear combination of elements in V_h

To illustrate this idea and explain what "discretizing" a function space actually means, we will examine the 1 dimensional equation

$$-\frac{d}{dx}\left[p(x)\frac{du}{dx}\right] + q(x)u = f(x) \quad \text{for} \quad u \in C^2((0,1))$$

where $p(x) > c > 0$ and $q(x) \geq 0$, with the Dirichlet boundary conditions

$$u(0) = 0, \qquad u(1) = 0$$

In the first step we will convert the problem into the weak form, just like we have done in the previous section.

$$-\int_0^1 \frac{d}{dx}\left[p(x)\frac{du}{dx}(x)\right] v(x)\ dx + \int_0^1 q(x)u(x)v(x)\ dx = \int_0^1 f(x)v(x)\ dx$$
$$\text{for} \quad v \in H_0^1((0,1))$$

And again we integrate by parts in the first term to arrive at the weak formulation

$$\int_0^1 p(x)\frac{du}{dx}(x)\frac{dv}{dx}(x)\ dx + \int_0^1 q(x)u(x)v(x)\ dx = \int_0^1 f(x)v(x)\ dx$$
$$\text{for} \quad v \in C_0^\infty(0,1)$$

We are now interested in solutions $u \in H_0^1((0,1))$. As the next step we discretize the function space $V = H_0^1((0,1))$. We choose V_h to be the linear span of a set of basis functions φ_i such that each basis function vanishes along most of $(0,1)$. As a consequence most products $\varphi_i\varphi_j$ of basis functions will equal zero and this is a desirable property as we will discover later on. For this particular example we will use hat functions of the form

$$\Psi : \mathbb{R} \to \mathbb{R}, \quad \Psi(x) := \begin{cases} (1+x) & \text{if} \quad -1 \leq x \leq 0 \\ (1-x) & \text{if} \quad 0 \leq x \leq 1 \\ 0 & \text{else} \end{cases}$$

to achieve a discretization. We can now choose a mesh $x_1 = h, ..., x_n = nh$ of equidistant points in $(0,1)$, where $x_i - x_{i+1} = h$ and $h = \frac{1}{n+1}$. We can now associate a hat function to each mesh point x_i via

$$\varphi_i = \Psi(x/h - i)$$

If we let $V_h = \text{span}(\varphi_1, ...\varphi_n)$ we get the space of piecewise linear interpolators on the mesh $x_i, ..., x_n$. In Figure 2 the shape of such a hat function as well as

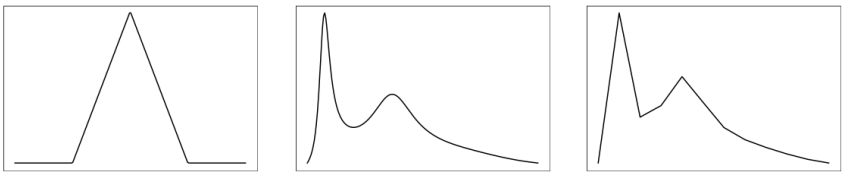

Figure 2: The image on the left shows a basis function φ_i, the middle picture shows a function $f \in H_0^1((0,1))$ and the rightmost image depicts a piecewise linear approximation of f in V_h. Note that the two images on the right have a different scaling than the left one.

an element of V_h is shown.
Our problem has now transformed into finding a $u_h \in \text{span}(\varphi_1, ...\varphi_n)$ such that

$$\int_0^1 p(x)\frac{du_h}{dx}(x)\frac{dv_h}{dx}(x)\ dx + \int_0^1 q(x)u_h(x)v_h(x)\ dx = \int_0^1 f(x)v_h(x)\ dx \qquad \forall v_h \in \text{span}(\varphi_1, ...\varphi_n)$$

Since every v_h is a linear combination of the hat functions $\varphi_1, ..., \varphi_n$, it is equivalent to solve

$$\int_0^1 p(x)\frac{du_h}{dx}(x)\frac{d\varphi_i}{dx}(x)\ dx + \int_0^1 q(x)u_h(x)\varphi_i(x)\ dx = \int_0^1 f(x)\varphi_i(x)\ dx \qquad \forall \varphi_i \in \{\varphi_1, ...\varphi_n\}$$

Also, since $u_h \in \text{span}(\varphi_1, ...\varphi_n)$ as well, we can replace it with a linear combination as well

$$u_h = \sum_{i=1}^{n} U_i\varphi_i$$

for some coefficients $U_i \in \mathbb{R}$. Thus we arrive at

$$\sum_{i=1}^{n} U_i \left[\int_0^1 p(x)\frac{d\varphi_i}{dx}(x)\frac{d\varphi_j}{dx}(x)\ dx + \int_0^1 q(x)\varphi_i(x)\varphi_j(x)\ dx\right] = \int_0^1 f(x)\varphi_j(x)\ dx \qquad \forall \varphi_j \in \{\varphi_1, ...\varphi_n\}$$

The problem has been reduced to finding the coefficients $U_i \in \mathbb{R}$ such that the

above equation is satisfied, in particular if we let

$$a_{i,j} = \int_0^1 p(x)\frac{d\varphi_i}{dx}(x)\frac{d\varphi_j}{dx}(x)\ dx + \int_0^1 q(x)\varphi_i(x)\varphi_j(x)\ dx \qquad \text{for } i,j = 1,...,n$$
$$F_j = \int_0^1 f(x)\varphi_j(x)\ dx \qquad \text{for } j = 1,...,n$$

we can define the symmetric matrix $A = (a_{i,j})$ and the vector $F = (F_1, ..., F_n)$ to formulate the problem as a system of linear equations given by

$$AU = F$$

where $U = (U_1, ..., U_n)$ contains the coefficients for our approximate solution u_h in respect to the basis $\{\varphi_1, ..., \varphi_n\}$ of V_h. The matrix A is usually called the "stiffness matrix" of the BVP. Since the basis functions φ_i are equal to 0 outside of $[x_{i-1}, x_{i+i}]$ for $1 \leq i \leq n-1$, the entries $a_{i,j}$ with $|i-j| > 1$ are 0. Thus A is a tridiagonal matrix. We could now calculate the entries of A and F numerically. Assuming a simple case where the functions $p(x)$ and $q(x)$ are constant, we can calculate the the entries of A exactly:

$$\begin{aligned}
a_{i,j} &= p\int_0^1 \frac{d\varphi_i}{dx}(x)\frac{d\varphi_j}{dx}(x)dx, \quad +q\int_0^1 \varphi_i(x)\varphi_j(x)dx \\
&= p\begin{cases} 2/h & \text{if } \ i=j \\ -1/h & \text{if } \ |i-j|=1 \\ 0 & \text{else} \end{cases} \quad + q\begin{cases} 4h/6 & \text{if } \ i=j \\ h/6 & \text{if } \ |i-j|=1 \\ 0 & \text{else} \end{cases} \\
&= \begin{cases} 2p/h + 4hq/6 & \text{if } \ i=j \\ -p/h + hq/6 & \text{if } \ |i-j|=1 \\ 0 & \text{else} \end{cases}
\end{aligned}$$

Now it is possible to see that we have obtained a set of linear equations

$$-p\frac{U_{i-1} - 2U_i + Ui + 1}{h^2} + q\frac{U_{i-1} + 4U_i + U_{i+1}}{6} = \frac{1}{h}\int_{x_{i-1}}^{x_{i+1}} f(x)\varphi_i(x)dx$$

for $1 \leq i \leq n-1$. Since we imposed the boundary conditions $u(0) = u(1) = 0$, the coefficients U_0 and U_n are already determined as 0. Solving this system leads to the approximate solution $u_h = \sum_{i=0}^{n} U_i\varphi_i(x)$. To see what we have actually accomplished with the FEM procedure, we note that the above system corresponds to a three-point finite difference scheme and our approximate solution is in fact a linear interpolation of the values U_i at grid points x_i. As with finite differences, the solution process boils down to solving a system of linear equations.

4 Solving the 2D Poisson equation with FEM

In this chapter we will use the finite element method to numerically solve a physical problem. We have seen a basic example of the FEM with 1D equations, but in now we are going to deal with the 2D Poisson equation. In the last section, the idea of discretizing a function space was the central theme, now we will also see how the problem can be split up into the contributions of the individual elements. This will then enable us to significantly simplify the calculations. Section 4 is mostly sourced from [8] unless stated otherwise.

4.1 Potential flow over a wing

As a model problem we will consider the airflow around an aircraft wing. In a very simple case we can model this situation via the Poisson equation. More precisely we want to find a vector field V of air velocity around the wing. We make a few assumptions a priori

- Restrict the discussion to the 2D situation of a cross section of the wing. This is possible if we assume the wing is much longer than it is wide.
- Assume that the airflow is frictionless and irrotational, this means $\nabla \times V = 0$. This is of course not the case in reality, but we will neglect these effects for the sake of simplicity.
- Assume the flow to be incompressible, this means div $V = 0$.

Thus let $\Omega \subset \mathbb{R}^2$. Our second assumption implies that $V = \nabla\Phi$ for some potential $\Phi : \Omega \to \mathbb{R}$, in other words

$$V(x, y) = (u, v) \quad \text{where} \quad u = \frac{\partial \Phi}{\partial x}, \quad v = \frac{\partial \Phi}{\partial y}$$

Together with div $V = 0$ we obtain

$$\begin{aligned} \text{div } V = \frac{\partial u}{\partial y} + \frac{\partial v}{\partial y} &= 0 \\ \Rightarrow \frac{\partial^2 \Phi}{\partial y^2} + \frac{\partial^2 \Phi}{\partial y^2} &= 0 \end{aligned}$$

This means we can indeed use the Poisson equation to model the situation. As a domain Ω we will choose a rectangular box with a wing shaped contour in the center. Figure 3 depicts the configuration. On the boundary $\partial\Omega$ we want to simulate an inflow of air on the left side and a low pressure outlet on the right. The top and bottom boundary of the box, as well as the contour of the wing will obey no-flux boundary conditions, meaning we allow no flow of air through them at all. This leads us to the boundary conditions

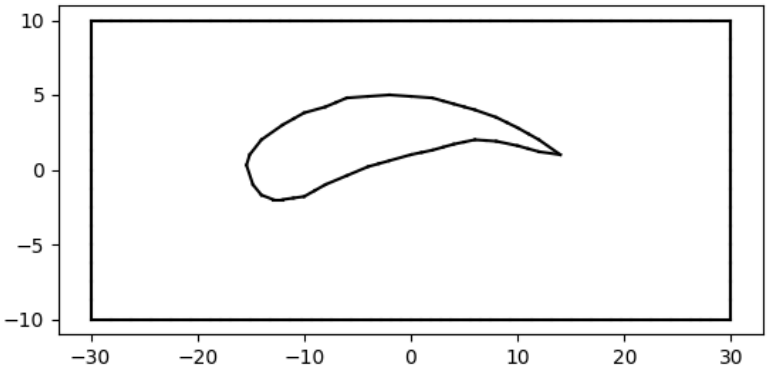

Figure 3: The outlines in the plot are the boundary $\partial\Omega$ and the area between the box and the wing contour is Ω

- $\frac{\partial\Phi}{\partial\nu} = 1$ on the left boundary of the box
- $\Phi = 0$ on the right boundary of the box
- $\frac{\partial\Phi}{\partial\nu} = 0$ on the top and bottom boundary, as well as the wing surface

We thus finally obtain the BVP

$$-\Delta\Phi = 0 \qquad \text{on} \quad \Omega \tag{4.1}$$

$$-\nu\nabla\Phi = \kappa\Phi - g \qquad \text{on} \quad \partial\Omega \tag{4.2}$$

where ν is the inside normal vector on the boundary $\partial\Omega$ and $\kappa \geq 0$ as well as g are in piecewise constant functions on $\partial\Omega$. In particular κ and g will be constant on every straight line segment of our boundary. We will seek solutions in the space

$$V = \{v : \|v\|_{L^2(\Omega)} + \|\nabla v\|_{L^2(\Omega)} < \infty\}$$

Note that we can not achieve the second boundary condition directly, since $\nabla\Phi$ does not have a coefficient. Thus we will approximate it by choosing κ to be very large on the right boundary, this has the effect that $u = \frac{-\nu\nabla\Phi}{\kappa} \approx 0$ on that part of $\partial\Omega$.

4.2 Discretization of the problem

To employ FEM we first have to state the problem in its weak form and then discretize the space V. For this we will need Greens formula:

Definition. (Greens formula)[3] Let $\Omega \subset \mathbb{R}^2$ be a domain with boundary $\partial\Omega$ and ν the external unit normal vector of $\partial\Omega$. Then, for sufficiently smooth

functions $f, g : \Omega \to \mathbb{R}^2$ it holds that

$$\int_\Omega -\Delta f g \; dx = \int_\Omega \nabla f \nabla g \; dx - \int_{\partial\Omega} \nu \nabla f g \; ds$$

This is a consequence of the Gauss integral thm, for a proof of this result consult R.Courant, it is also given in most Analysis 2 textbooks. Writing u instead of Φ, we can multiply equation 4.1 with test functions $v \in V$, integrate and apply Greens formula

$$\begin{aligned} 0 &= \int_\Omega -\Delta u v \; dx \\ &= \int_\Omega \nabla u \nabla v \; dx - \int_{\partial\Omega} \nu \nabla u v \; ds \\ &= \int_\Omega \nabla u \nabla v \; dx - \int_{\partial\Omega} (\kappa u - g) v \; ds \end{aligned}$$

where the last equality is just the result of plugging in the boundary conditions. At this point we can pull all terms containing u to the left side and we arrive at

$$\int_\Omega \nabla u \nabla v \; dx + \int_{\partial\Omega} \kappa u v \; ds = \int_{\partial\Omega} g v \; ds$$

The first step of the FEM is now completed, we have arrived at a weak formulation. Now we want to replace V with the discretized space V_h and find $u_h \in V_h$ such that

$$\int_\Omega \nabla u_h \nabla v_h \; dx + \int_{\partial\Omega} \kappa u_h v_h \; ds = \int_{\partial\Omega} g v_h \; ds \forall v_h \in V_h$$

To approximate our solution space $V = \{v : \|v\|_{L^2(\Omega)} + \|\nabla v\|_{L^2(\Omega)} < \infty\}$ we choose a triangular grid on $\bar{\Omega}$: Let $G = \{K_1, K_2, ..., K_n\}$ be the a set of triangles $K_i \subset \bar{\Omega}$ that cover $\bar{\Omega}$. In particular, if we have two triangles they can only intersect each other on their boundary if at all, so $K_i \cap K_j = \partial K_i \cap \partial K_j$ if $i \neq j$ and $\sum_{i=1}^n K_i = \bar{\Omega}$. Each triangle K_i consists of three vertices $N_{i_1}, N_{i_2}, N_{i_3} \in \bar{\Omega} \subset \mathbb{R}^2$ and three edges $E_{i_1}, E_{i_2}, E_{i_3} \subset \bar{\Omega}$ connecting them to form the triangle. Figure 4 shows a possible triangulation of Ω.
Now let m be the total number of vertices and $M = \{N_1, N_2, ..., N_m\}$ be the set of all vertices in our triangulation. For every triangle K_i we impose that it must contain exactly three vertices and they must be its corner points. We will not discuss the existence of such a triangulation here, but it is indeed possible for our particular case. Now we choose V_h to be the space of all continuous functions v_h such that v_h is linear on every K_i. This means that each element of V_h is a linear combination of the piecewise linear basis functions $\varphi_i : \bar{\Omega} \to \mathbb{R}$ with $\varphi_i(N_j) = \delta_{i,j}$. These basis functions are the two dimensional counterpart to the

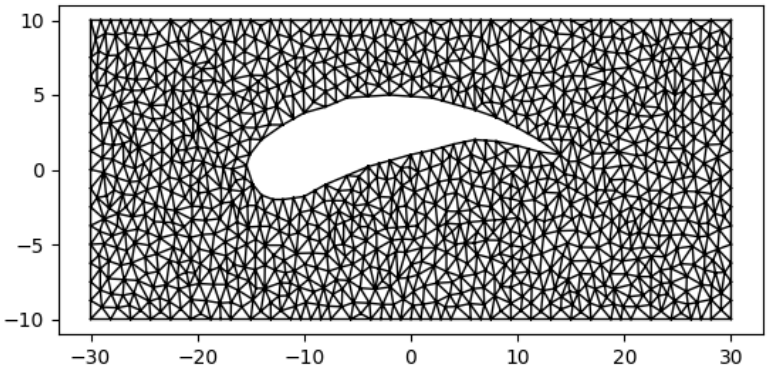

Figure 4: Possible triangulation of Ω. There are of course many ways to triangulate a domain. If some parts of the domain are of particular interest, the grid can be locally refined in the surrounding area [8].

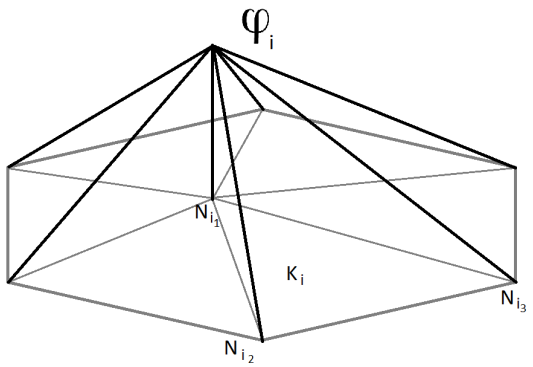

Figure 5: A 2D hat function φ_i associated to the node N_{i_1}

hat functions in the previous section. In Figure 5 the graph of one such basis function is depicted, Figure 6 shows an element of V_h. The triangles $K_i \in G$ are called elements and on each element there are only three basis functions φ_i that are not zero on K_i. In particular these functions are $\varphi_{i_1}, \varphi_{i_2}, \varphi_{i_3}$ corresponding to the nodes $N_{i_1}, N_{i_2}, N_{i_3}$.

Now that we have chosen a discetization, we can construct a system of linear equations to find the coefficients U_i such that our approximate solution u_h is given by

$$u_h = \sum_{i=1}^{m} U_i \varphi_i$$

where m is the number of vertices of our triangulation and φ_i is the basis function corresponding to the i-th vertex. Plugging in the formula for u_h into the

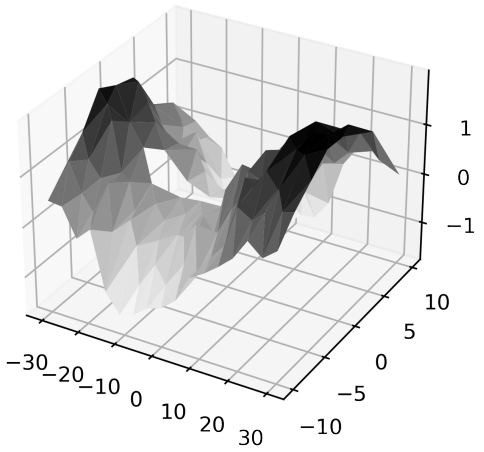

Figure 6: The elements of V_h are piecewise linear functions on Ω, in the above picture the graph of one such function is depicted. The hole in the middle corresponds to the inside of the wing, which is not part of Ω

weak form of our problem yields

$$\sum_{i=1}^{m} U_i \left[\int_\Omega \nabla\varphi_i \nabla v_h \; dx + \int_{\partial\Omega} \kappa\varphi_i v_h \; ds \right] = \int_{\partial\Omega} g v_h \; ds \forall v_h \in V_h$$

and this is equivalent to requiring

$$\sum_{i=1}^{m} U_i \left[\int_\Omega \nabla\varphi_i \nabla\varphi_j \; dx + \int_{\partial\Omega} \kappa\varphi_i\varphi_j \; ds \right] = \int_{\partial\Omega} g\varphi_j \; ds \forall j \in \{1, ..., m\}$$

Thus we have again reduced our problem to solving a system of linear equations

$$(A + R)U = b$$

with A, R and b given by

$$A_{i,j} = \int_\Omega \nabla\varphi_i \nabla\varphi_j \; dx$$
$$R_{i,j} = \int_{\partial\Omega} \kappa\varphi_i\varphi_j \; ds$$
$$b_j = \int_{\partial\Omega} g\varphi_j \; ds$$

for $i, j \in \{1, ..., m\}$. As before in the 1D case, the solution vector will contain the coefficients U_i to represent u_h in the basis $\{\varphi_i\}_{i\in\{1,...,m\}}$. Our solution will thus be a piecewise linear function, in other words its graph will be a surface consisting of many triangles.

4.3 Assembling the stiffness matrix

In this section, the final important advantage of the finite element method will become apparent. Since we need to calculate many integrals, we want to do it in the most efficient way possible. The first idea is to split up the integration domain Ω into the individual triangle components $K_i \in G$, this means

$$A_{i,j} = \int_\Omega \nabla\varphi_i \nabla\varphi_j \, dx = \sum_{K\in G} \int_K \nabla\varphi_i \nabla\varphi_j \, dx$$

We can observe that it is actually not important in what order we calculate the contributions of all triangles K, as long as we add them together correctly we still get back the original integral. The contribution of a triangle K to $A_{i,j}$ is given by

$$A^K_{i,j} = \int_K \nabla\varphi_i \nabla\varphi_j \, dx$$

Thus we can iterate over the set of all triangles G and for each K we add $\int_K \nabla\varphi_i \nabla\varphi_j \, dx$ to $A_{i,j}$. Since on each K there are only three φ_i that are nonzero on K, we have to calculate a total of 9 integrals per element. The contribution of the integrals are then added back to the global stiffness matrix A and we move on to the next element. For example consider a triangle K with vertices N_1, N_2, N_3, then $\varphi_1, \varphi_2, \varphi_3$ are nonzero on K. Thus K only contributes to the the components $A_{1,1}, A_{1,2}, A_{1,3}, A_{2,1}, A_{2,2}, A_{2,3}, A_{3,1}, A_{3,2}, A_{3,3}$ of the global stiffness matrix, since the integral of all other combinations of φ_i on K is zero. This means we essentially first calculate a 3×3 local stiffness matrix A^K for each element K and then add the entries of all local matrices to the global stiffness matrix A.

After dividing up the work over all the elements, we can further exploit the simple structure of our basis functions φ_i. First look at a general triangle with vertices A, B, C. Assume that for some basis function φ we have $\varphi(A) = 1$ and $\varphi(B) = \varphi(C) = 0$. Since φ is linear and zero on the the line connecting B and C, its gradient will be orthogonal to this line. We can construct a unit vector in this direction easily in $\mathbb{R}^2$ via $(x, y)^\perp = (-y, x)$. The length of this new orthogonal vector $(C - B)^\perp$ will be the same as the length of $C - B$, thus the unit vector in the direction of $\nabla\varphi$ is given by $\frac{(C-B)^\perp}{|(C-B)|}$. Let H be the normal distance of A to

the line $C - B$, then we know that $|\nabla\varphi| = \frac{1}{|H|}$ since φ rises from 0 to 1 linearly along H and there is no change in the normal direction. Thus our gradient vector is given by

$$\nabla\varphi = \frac{(C-B)^{\perp}}{|H||C-B|} = \frac{(C-B)^{\perp}}{2|K|}$$

where $|K|$ denotes the area of the triangle. Applying what we just calculated, we again consider a triangle K with nodes $N_1 = (x_1, y_1), N_2 = (x_2, y_2), N_3 = (x_3, y_3)$ for the basis function φ_i and thus

$$\nabla\varphi_i = \frac{1}{2|K|}\begin{pmatrix} y_j - y_k \\ x_k - x_j \end{pmatrix}$$

Cyclic permutation of i, j, k over $\{1, 2, 3\}$ yields the three gradients. Choosing $b_i = \frac{y_j - y_k}{2|K|}$ and $c_i = \frac{x_k - x_j}{2|K|}$, we can calculate the entries of our 3×3 local stiffness matrix A^K like so

$$\begin{aligned} A^K_{i,j} &= \int_K \nabla\varphi_i \nabla\varphi_j \, dx \\ &= (b_i b_j + c_i c_j) \int_K dx \\ &= (b_i b_j + c_i c_j)|K| \end{aligned}$$

where again i, j, k permutate over $\{1, 2, 3\}$. We have obtained a simple, easy to evaluate expression for $A^K_{i,j}$. Note that after this step we still have to add the entries of our local stiffness matrix A^K to the global stiffness matrix A and this will involve translating the local i, j indices to global indices. In the next step we can also simplify the integrals over the boundary $\partial\Omega$. By G_e we will denote the set of all edges contained in our triangulation. Note that $\partial\Omega$ is the union of a finite set of triangle edges $E \in G_e$ such that $\partial\Omega = \bigcup_{\substack{E \in G_e \\ E \subset \partial\Omega}} E$ and the intersection of two such edges is at most one point. As a consequence we can divide the boundary integrals into the parts contributed by individual edge segments E of our triangulation

$$R_{i,j} = \int_{\partial\Omega} \kappa\varphi_i\varphi_j \, ds = \sum_{\substack{E \in G_e \\ E \subset \partial\Omega}} \int_E \kappa\varphi_i\varphi_j \, ds$$

Also, we observe that only two basis functions can contribute per line segment E. Consider E given by the line connecting vertices N_1 and N_2. Then only φ_1 and φ_2 are nonzero on E. Knowing this, we can again just iterate over the boundary

edges and for each edge we get the contribution

$$\begin{aligned} R_{i,j}^E &= \int_E \kappa \varphi_i \varphi_j \, ds \\ &= \kappa \begin{cases} \int_0^{|E|} \frac{t^2}{|E|^2} \, dt = \frac{|E|}{3} & \text{if } \; i = j \\ \int_0^{|E|} \frac{t}{|E|} \frac{|E|-t}{|E|} = \frac{|E|}{6} \, dt & \text{if } \; i \neq j \end{cases} \\ &= \frac{\kappa}{6}(1 + \delta_{i,j})|E| \end{aligned}$$

where $|E|$ denotes the length of E and we only consider the φ_i, φ_j that are both nonzero on E. For calculating the entries of b we also use the same approach

$$b_j^E = \int_E \kappa g \varphi_j \, ds = \kappa g \int_0^{|E|} t \, dt = \frac{1}{2} \kappa g |E|$$

Note that we have used the fact that κ and g are piecewise constant on E. Using the above formulas for calculating A, R and b makes it possible to iterate over the set of all triangles and boundary edges. The big problem of approximating a solution of the BVP boils down to a large number of small problems involving simple calculations. Also note that we have avoided the need to approximate integrals completely, but this was only possible because we assumed our boundary functions to be piecewise constant. As a final result we have obtained a system of linear equations

$$(A + R)U = b$$

and after solving it for $U = (U_1, ..., U_m)$ the approximate solution is given by

$$\Phi_h(x) = \sum_{i=1}^{m} U_i \varphi_i(x)$$

In Figure 7 we can see the equipotential lines of our approximate solution Φ_h. Recall that we originally we wanted to calculate the flow field around an aircraft wing, thus we can numerically differentiate Φ_h in x and y direction to obtain an approximation of $\nabla\Phi$, the vector field describing the flow. Figure 8 shows a visualization of the field. Furthermore, we can also calculate the Bernoulli pressure around the wing, defined by $p = -|\nabla\Phi|^2$. This is shown in Figure 9. If you are interested in the source code of this program take a look at section 5.2 of the appendix, the full Python source is provided there.

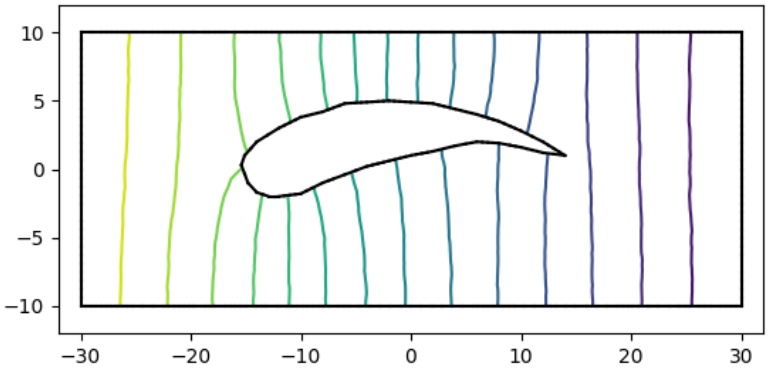

Figure 7: Equipotential lines of Φ_h around the wing

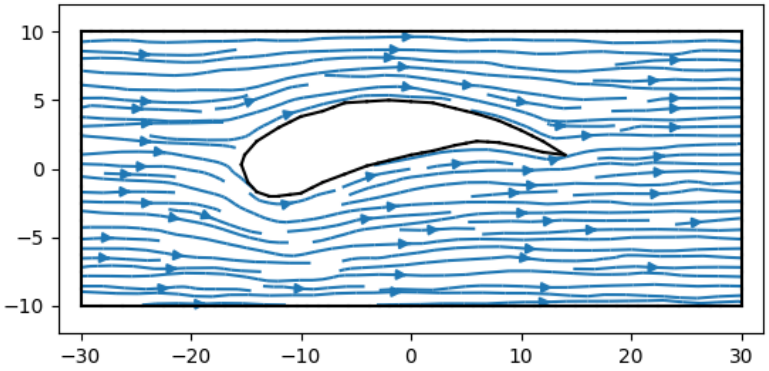

Figure 8: Flow lines showing the direction of the vector field $\nabla\Phi_h$ around the wing

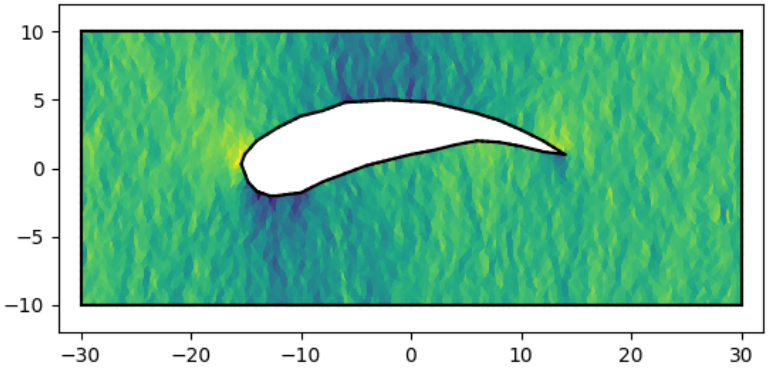

Figure 9: Bernoulli pressure in Ω. Because the gradient of our approximation is constant on each element, the plot resembles a mosaic

4.4 Theoretical remarks about the FEM

To provide some information on the validity of the finite element method, we will now examine a few properties of FEM solutions for problems of the form

$$-\Delta u = f \qquad \text{on } \Omega \tag{4.3}$$

$$u = 0 \qquad \text{on } \partial\Omega \tag{4.4}$$

where we search for weak solutions in $H_0^1(\Omega)$. Note that for the following discussion we will limit ourself to the basic case of Dirichlet boundary conditions where $u = 0$ on $\partial\Omega$. We will assume V_h to be a space of piecewise linear functions that vanish on $\partial\Omega$ and $\{\varphi_i\}_{1\leq i\leq m}$ is a basis of V_h as defined in the previous section. The following statments along with more detailed estimates can be found in [8].
First we establish that the stiffness matrix A is symmetric and positive definite.

Theorem 4.1. *The stiffness matrix A is symmetric and positive definite.*

Proof. Recalling that $a_{i,j} = \int_\Omega \nabla\varphi_i \cdot \nabla\varphi_j \, dx$, A is symmetric by definition, so we only have to show that

$$v^T A v > 0 \quad \forall v \in \mathbb{R}^m$$

We calculate

$$\begin{aligned} v^T A v &= \sum_{i,j=1}^{m} v_i A_{i,j} v_j \\ &= \sum_{i,j=1}^{m} v_i v_j \int_\Omega \nabla\varphi_i \cdot \nabla\varphi_j \, dx \\ &= \int_\Omega \nabla \left(\sum_{i=1}^{m} v_i \varphi_i \right) \cdot \left(\sum_{j=1}^{m} v_j \varphi_j \right) \, dx \\ &= \| \nabla \left(\sum_{i=1}^{m} v_i \varphi_i \right) \|^2_{L_2(\Omega)} \end{aligned}$$

The above norm is larger than zero as long as $f = \sum_{i=1}^{m} v_i \varphi_i$ is constant. Since all functions in V_h are 0 on $\partial\Omega$, the only constant function in V_h is the zero function and thus $f > 0$ if $v \neq 0$. □

Theorem 4.2. *(Existence and uniqueness of FEM solutions) For the above problem there exits a unique finite element solution in $z_h \in V_h$.*

Proof. Since u_h is determined by a linear system, it follows from the positive definiteness of A that there always exists a solution. Thus we only have to show

uniqueness of u_h. Let u_h and $\tilde{u_h}$ be two solutions satisfying 4.3. Then

$$\int_\Omega \nabla u_h \cdot \nabla v_h \; dx = \int_\Omega f v_h \; dx \qquad \forall v_h \in V_h$$
$$\int_\Omega \nabla \tilde{u_h} \cdot \nabla v_h \; dx = \int_\Omega f v_h \; dx \qquad \forall v_h \in V_h$$

Subtracting these equations yields

$$\int_\Omega \nabla (u_h - \tilde{u_h}) \cdot \nabla v_h \; dx = 0 \; dx \qquad \forall v_h \in V_h$$

and now we can choose $v_h = u_h - \tilde{u_h} \in V_h$ and we get

$$\int_\Omega |\nabla (u_h - \tilde{u_h})|^2 \; dx = 0 \; dx \qquad \forall v_h \in V_h$$

Thus $u_h - \tilde{u_h}$ must be constant. Considering the boundary conditions we arrive at $u_h - \tilde{u_h} = 0$ since both solutions must be 0 on $\partial\Omega$. □

Theorem 4.3. *(Galerkin orthogonality) The FEM solution u_h satisfies the orthogonality*

$$\int_\Omega \nabla (u - u_h) \nabla v \; dx = 0 \qquad \forall v \in V_h$$

Proof. The weak formulation of 4.3 is

$$\int_\Omega \nabla u \cdot \nabla v \; dx = \int_\Omega f v \;\; dx \qquad \forall v \in H_0^1(\Omega)$$

and thus

$$\int_\Omega \nabla u_h \cdot \nabla v_h \; dx = \int_\Omega f v_h \; dx \qquad \forall v_h \in V_h$$

Since the top equation also holds for all $v_h \in V_h$, we can substract the two and arrive at the orthogonality statement. □

Using Galerkin orthogonality we can show that u_h is the best possible approximation of u in V_h.

Theorem 4.4. *The FEM solution u_h satisfies*

$$\|u - u_h\|_{L_2(\Omega)} \leq \|u - v_h\|_{L_2(\Omega)} \qquad \forall v_h \in V_h$$

Proof. Using $u - u_h = u - v_h + v_h - u_h$ for any $v_h \in V_h$ we have

$$\begin{aligned}
\|u - u_h\|^2_{L_2(\Omega)} &= \int_\Omega \nabla(u - u_h) \cdot \nabla(u - u_h)\ dx \\
&= \int_\Omega \nabla(u - u_h) \cdot \nabla(u - v_h)\ dx \\
&\quad + \int_\Omega \nabla(u - u_h) \cdot \nabla(v_h - u_h)\ dx \\
&= \int_\Omega \nabla(u - u_h) \cdot \nabla(u - v_h)\ dx \\
&\leq \|u - u_h\|_{L_2(\Omega)} \|u - v\|_{L_2(\Omega)}
\end{aligned}$$

where we have used that $\int_\Omega \nabla(u - u_h) \cdot \nabla(v_h - u_h)\ dx = 0$ because of Galerkin orthogonality. We now divide by $\|u - u_h\|_{L_2(\Omega)}$ and the proof is done. □

To conclude the discussion, the following error bound for u_h holds:

Theorem 4.5. *The FEM solution u_h satisfies*

$$\begin{aligned}
\|u - u_h\|_{H^1(\Omega)} &\leq Ch\|u\|_{H^2(\Omega)} \\
\|u - u_h\|_{L_2(\Omega)} &\leq Ch^2\|u\|_{H^2(\Omega)}
\end{aligned}$$

for some $C > 0$, where u is the true solution of 4.3 and $h = \max\{h_K | K \in G\}$ is the mesh size parameter (h_K is the diameter of the circumcircle of the triangle K).

For a proof of this statement consult [10].

4.5 Further references

The finite element schemes outlined here are the most basic configurations possible. In real world applications, the concrete techniques employed are far more sophisticated than what has been outlined here. Besides from basic triangular elements, one can also use e.g. quadrilateral elements [6]. Additionally we have limited the discussion to only piecewise linear basis functions, it is also possible to employ higher order polynomial interpolation [10]. Techniques such as adaptive FEM use results of prior calculations to refine or adapt the mesh in order to achieve better approximations of the solution [8]. It is also possible to alter the continuity requirements for the FEM solutions, examples of this are Morley and Crouzeix-Raviart elements [8]. The concept of finite elements can also be used to tackle problems outside the realm of differential equations, for example to approximate minimal surfaces [5].

5 Appendix

5.1 Important function spaces and lemmas

The statements here can be found in [9] unless stated otherwise.

Definition. (Spaces of continuously differentiable functions) Let $\Omega \subset \mathbb{R}^n$ be open and bounded domain, $k \in \mathbb{N}$. Then $C^k(\Omega)$ is the set of all k-times continuously differentiable real-valued functions defined on the domain Ω. More precisely the derivative $D^\alpha u$ is continuous for all multiindices α with $|\alpha| \leq k$. For a continuous function u the set $\text{supp}_u := \{x \in \Omega : u(x) \neq 0\}$ is called the support of u. By $C_0^k(\Omega)$ we denote the set of all $u \in C^k(\Omega)$ with compact support.

Consider the following example
Example: We define the function f on $\mathbb{R}^n$ by

$$f(x) := \begin{cases} e^{-\frac{1}{1-|x|^2}} & \text{if} \quad |x| < 1 \\ 0 & \text{else} \end{cases}$$

where $|x| = (x_1^2 + x_2^2 + ... + x_n^2)^{1/2}$. The support of f is the set $\{x \in \mathbb{R}^n : |x| \leq 1\}$ and f is continuous since $\lim_{h \to 0} e^{-1/h} = 0$. Thus $f \in C_0^k(\mathbb{R}^n)$ for all $k \in \mathbb{N}$.

Let

$$C_0^\infty(\Omega) = \bigcap_{k \geq 0} C_0^k(\Omega)$$

$C_0^\infty(\Omega)$ is called the set of test functions on Ω. Note that the f from the above example is an element of $C_0^\infty(\mathbb{R}^n)$. We will now define spaces of integrable functions, in particular the $L_p(\Omega)$ function spaces.

Definition. (Spaces of Lebesgue-integrable functions) Let Ω be the same as above, $p \in \mathbb{R}$ with $p \geq 1$. The set $L_p(\Omega)$ contains all real-valued functions defined on Ω that satisfy

$$\int_\Omega |u(x)|^p dx < \infty$$

Two functions $f, g \in L_p(\Omega)$ are called equal "almost everywhere" if $f(x) - g(x) \neq 0$ only on a set with measure zero. If two functions are equal almost everywhere, we consider them identical. We can define a norm on $L_p(\Omega)$ via

$$\|u\|_{L_p(\Omega)} := \left(\int_\Omega |u(x)|^p dx \right)^{1/p}$$

The space $L_\infty(\Omega)$ contains all functions u defined on Ω that satisfy

$$\exists M > 0 : |u(x)| \leq M \quad \forall x \in \Omega \setminus \Gamma$$

where Γ is a set of measure zero. The smallest such number M is called the essential supremum of $|u|$ and we can define a norm on $L_\infty(\Omega)$ by

$$\|u\|_{L_\infty(\Omega)} := \operatorname{ess.sup}_{x\in\Omega} |u(x)|$$

In the particular case of $p = 2$ we can equip $L_2(\Omega)$ with an inner product

$$\langle u, v\rangle := \int_\Omega u(x)v(x)dx$$

As a result,

$$\|u\|_{L_2(\Omega)} = \langle u, u\rangle^{1/2} := \left(\int_\Omega |u(x)|^2 \, dx\right)^{1/2}$$

Lemma 5.1. *(Partial integration in higher dimensions) Let Ω be an open subset of $\mathbb{R}^n$ and $u \in C^k(\Omega)$, then for $v \in C_0^\infty(\Omega)$ the following formula holds*

$$\int_\Omega D^\alpha u(x)v(x) \, dx = (-1)^{|\alpha|} \int_\Omega u(x)D^\alpha v(x) \, dx$$
$$|\alpha| \le k, \quad \forall v \in \mathbb{C}_0^\infty$$

Proof. Since $v = 0 \quad \forall x \in \mathbb{R}^n \setminus \Omega$ we can integrate over the entire space $\mathbb{R}^n$ instead of Ω. Note that u must not be defined on the whole space, but this does not pose any significant difficulties since for u integrable on Ω we can expand it to an integrable function on the entire space by setting it to zero everywhere except Ω. We now have

$$\int_\Omega D^\alpha u(x)v(x) \, dx = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} D^\alpha u(x_1, ..., x_n)v(x_1, ..., x_n) \, dx_1...dx_n$$

and if we integrate by parts in respect to x_i we obtain

$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} D^\alpha u(x_1, ..., x_n)v(x_1, ..., x_n) \, dx_1...dx_n =$$
$$\left[D^{\alpha-\beta}u(x_1, ..., x_n)v(x_1, ...x_n)\right]_{x_i=-\infty}^{x_i=\infty} -$$
$$\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} D^{\alpha\beta}u(x_1, ..., x_n)D^\beta v(x_1, ..., x_n) \, dx_1...dx_n$$

where β is a multiindex which is 0 everywhere except at the i-th entry. Since

v is a test function, it and its derivatives vanish outside of Ω and thus

$$\left[D^{\alpha-\beta}u(x_1,...,x_n)v(x_1,...x_n)\right]_{x_i=-\infty}^{x_i=\infty}=0$$

and we are left with

$$(-1)\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty}D^{\alpha\beta}u(x_1,...,x_n)D^{\beta}v(x_1,...,x_n)\ dx_1...dx_n = (-1)\int_{\Omega}D^{\alpha-\beta}u(x)D^{\beta}v(x)\ dx$$

We now observe that $D^{\beta}v(x)$ is again a test function and we can thus repeat this procedure and move the derivatives over to v step by step via integration by parts. Repeating the above step until $\alpha-\beta=(0,0,...,0)$ and thus $\beta=\alpha$ yields

$$\int_{\Omega}D^{\alpha}u(x)v(x)\ dx=(-1)^{|\alpha|}\int_{\Omega}u(x)D^{\alpha}v(x)\ dx$$

□

Lemma 5.2. *(Triangle inequality) For $u,v\in\mathbb{H}$ it holds that*

$$\|u+v\|\leq\|u\|+\|v\|$$

Proof. We have

$$\begin{aligned}\|u+v\|^2&=\langle u+v,u+v\rangle\\&=\|u\|^2+2\langle u,v\rangle+\|v\|^2\end{aligned}$$

We can now apply the Cauchy-Schwarz inequality at $\langle u,v\rangle\leq|\langle u,v\rangle|$ and obtain

$$\begin{aligned}\|u+v\|^2&\leq(\|u\|+\|v\|)^2\\\Rightarrow\|u+v\|&\leq\|u\|+\|v\|\end{aligned}$$

□

Lemma 5.3. *(Hölder inequality) Let $p,q\geq1$ such that $1/p+1/q=1$. For $u\in L_p(\Omega)$ and $v\in L_q(\Omega)$ we have*

$$\left|\int_{\Omega}u(x)v(x)\ dx\right|\leq\|u\|_{L_p(\Omega)}\|v\|_{L_q(\Omega)}$$

Proof. Let $a, b > 0$, we will first proof that

$$a^{1/p} b^{1/q} \leq \frac{a}{p} + \frac{b}{q}$$

and then use this to proof the original inequality. Consider the function

$$f(t) = \frac{t^p}{p} + \frac{1}{q} - t \quad \text{for} \quad t > 0$$

Without loss of generality we have assumed that $p > 1$. We have $f'(t) = t^{p-1} - 1$ and $f''(t) = (p-1)t^{p-2}$.

$$\begin{aligned} \Rightarrow f'(1) = 0 \quad \text{and} \quad f'(t) \neq 0 \ \ \forall t \in (0, \infty) \setminus \{1\} \\ \text{and} \quad f''(1) = (p-1) > 0 \end{aligned}$$

hence f assumes its minimum at $t = 1$ and thus $f(t) \geq \frac{1}{p} + \frac{1}{q} - 1 = 0$. In particular

$$f((a/b)^{1/p}) = \frac{\frac{a}{b}}{p} + \frac{1}{q} - \left(\frac{a}{b}\right)^{1/p} \geq 0$$

and this yields

$$\frac{a}{p} + \frac{b}{q} \geq b \left(\frac{a}{b}\right)^{1/p} = a^{1/p} b^{1/q}$$

We can now proceed to proving the Hölder inequality, let $u \in L_p(\Omega)$ and $v \in L_q(\Omega)$, then the product uv is in $L_1(\Omega)$ and

$$\begin{aligned} \frac{\|uv\|_{L_1(\Omega)}}{\|u\|_{L_p(\Omega)} \|v\|_{L_q(\Omega)}} &= \int_\Omega \frac{|u(x)|}{(\int_\Omega |u(y)|^p \ dy)^{1/p}} \frac{|v(x)|}{(\int_\Omega |v(y)|^q dy)^{1/q}} \ dx \\ &= \int_\Omega \left(\frac{|u(x)|^p}{\int_\Omega |u(y)|^p \ dy}\right)^{1/p} \left(\frac{|v(x)|^q}{\int_\Omega |v(y)|^q \ dy}\right)^{1/q} \ dx \end{aligned}$$

Applying the above result we now obtain

$$\frac{\|uv\|_{L_1(\Omega)}}{\|u\|_{L_p(\Omega)} \|v\|_{L_q(\Omega)}} \leq \int_\Omega \left(\frac{|u(x)|^p}{p \int_\Omega |u(y)|^p \ dy} + \frac{|v(x)|^q}{q \int_\Omega |v(y)|^q \ dy}\right) \ dx$$

Since we can seperate the integral and the integrals over y are constant, we obtain

$$\frac{\|uv\|_{L_1(\Omega)}}{\|u\|_{L_p(\Omega)} \|v\|_{L_q(\Omega)}} \leq \frac{1}{p} + \frac{1}{q} = 1 \quad \Rightarrow \quad \|uv\|_{L_1(\Omega)} \leq \|u\|_{L_p(\Omega)} \|v\|_{L_q(\Omega)}$$

□

5.2 Source code of 2D Poisson solver

Here is a complete Python source code of the FEM described in section 4. This code is based on some Matlab code snippets from [8]. Before running the code make sure you have all required packages installed. The triangle package can be downloaded on https://rufat.be/. The lines up until 244 contain the actual calculations, the remaining part of the code is visualization.

```
1
2  import numpy as np
3  import matplotlib.pyplot as plt
4  from triangle import *
5  from matplotlib.tri import Triangulation, LinearTriInterpolator
6
7  #set up triangulation mesh
8  #wing and boundary
9  region = {
10 "vertices": [[-30,-10],
11             [-30,10],
12             [30,10],
13             [30,-10],
14             [-19,-2],
15             [-17,0],
16             [-10,3],
17             [-3,3],
18             [4,3],
19             [19,0],
20             [10,0],
21             [0,-1],
22             [-9,-2],
23             [-15,-3]],
24 "segments":[[0,1],
25            [1,2],
26            [2,3],
27            [3,0],
28            [4,5],
29            [5,6],
30            [6,7],
31            [7,8],
32            [8,9],
33            [9,10],
34            [10,11],
35            [11,12],
36            [12,13],
37            [13,4]],
38 "holes":    [[0,0]]
39 }
40
41 #generate a mesh that will cover a rectangular region around the
       wing
```

```
mesh = triangulate(region, opts = 'qpa0.3')

#Plot the triangular mesh
plt.triplot([mesh['vertices'][i][0] for i in range(len(mesh['
    vertices']))],
                [mesh['vertices'][i][1] for i in range(len(mesh['
    vertices']))],
                mesh['triangles'], color = 'k',linewidth = 1)
plt.xlim(-32, 32)
plt.ylim(-12,12)
plt.show()

#calculate area of a triangular element with the coordinates of the
     nodes
#INPUT: node coordinates A, B and C
#OUTPUT: area of the triangular element
def triArea(A,B,C):
    return (1.0/2)*abs((A[0] - C[0])*(B[1] - A[1]) - (A[0] - B[0])
    *(C[1] - A[1]))


#using piecewise linear hat function to interpolate the solution,
    we need their
#gradients as well as the area of the triangle to calculate the
    stiffness matrix
#INPUT: index of the element, triangulation data
#OUTPUT: area of the triangle, gradient vectors (b,c) of the three
    hat functions corresponging to the nodes of the element
def hatGradients(elementID,tri):
    element = tri['triangles'][elementID]

    A = tri['vertices'][element[0]]
    B = tri['vertices'][element[1]]
    C = tri['vertices'][element[2]]

    #calculate area of the element
    area = triArea(A,B,C)

    #calculate the gradients of the three hat functions
    b = np.array([B[1] - C[1], C[1] - A[1], A[1] - B[1]])*(1/(2*
    area))
    c = np.array([C[0] - B[0], A[0] - C[0], B[0] - A[0]])*(1/(2*
    area))

    return [area, b, c]


#calculate the stiffness matrix by iterating over the individual
    elements
#using the formulas derived in section 4
```

```
#INPUT: triangulation tri
#OUTPUT: stiffness matrix A
def assembleStiffnessMatrix(tri):
    #number of nodes
    n = len(tri['vertices'])
    #number of elements
    m = len(tri['triangles'])

    #allocate stiffness matrix A
    A = np.zeros((n,n))

    for k in range(m):
        #compute gradients of the three hat functions corresponding to
        #the nodes of the element k
        area, b, c = hatGradients(k, tri)
        #compute local element mass matrix Ak
        Ak = np.array([[b[0]**2 + c[0]**2, b[0]*b[1] + c[0]*c[1], b[0]*b[2] + c[0]*c[2]],
                       [b[1]*b[0] + c[1]*c[0], b[1]**2 + c[1]**2, b[1]*b[2] + c[1]*c[2]],
                       [b[2]*b[0] + c[2]*c[0], b[2]*b[1] + c[2]*c[1], b[2]**2 + c[2]**2]])
        #add Ak to the global stiffness matrix
        for i in [0,1,2]:
            for j in [0,1,2]:
                #transform local indices to global indices
                loc2glb_i = tri['triangles'][k][i]
                loc2glb_j = tri['triangles'][k][j]
                A[loc2glb_i][loc2glb_j] += Ak[i][j]
    return A


#to determine if an edge is a boundary edge, check if it lies on one of the original boundaries
#to do this we need a function to check if a point lies on the boundary
#INPUT: coordinate vector P0
#OUTPUT: True if P0 is on the boundary of the region, False if not
def isOnBoundary(P0):
    #for every boundary segment we check if it contains P0
    for seg in region['segments']:
        P1 = region['vertices'][seg[0]]
        P2 = region['vertices'][seg[1]]

        #calculate the distance of P0 to the line segment
        dist = abs((P2[0] - P1[0])*(P1[1] - P0[1]) - (P1[0] - P0[0])*(P2[1] - P1[1]))/(np.sqrt((P2[0] - P1[0])**2 + (P2[1] -P1[1])**2))

```

```
124         #if P0 lies on the segment, it is a boundary point
125         if dist < 0.001:
126             return True
127     return False
128 
129 
130 #find the edges in the mesh that form the region boundary
131 #INPUT: none
132 #OUTPUT: array of boundary edges in respect to the mesh
133 def getBoundary():
134     boundaryEdges = []
135     for edge in mesh['segments']:
136         #get start end end point from triangulation
137         S = np.array(mesh['vertices'][edge[0]])
138         E = np.array(mesh['vertices'][edge[1]])
139 
140         #we also check if the midpoint is on the boundary,
    otherwise we cannot truly be sure if
141         #an edge contained in the boundary
142         M = (S + E)/2
143         if isOnBoundary(S) and isOnBoundary(E) and isOnBoundary(M):
144             boundaryEdges.append(edge)
145     return boundaryEdges
146 
147 boundary = getBoundary()
148 
149 
150 #assuming robin boundary conditions, calculate the boundary matrix
    R
151 #the function Kappa is assumed to be piecewise constant on a given
    edge
152 #INPUT: kappa, array of boundary edges, triangulation of the region
153 #OUTPUT: boundary matrix R
154 def assembleBoundaryMatrix(kappa, boundary, tri):
155     #number of nodes
156     n = len(tri['vertices'])
157 
158     #allocate global boundary matrix R
159     R = np.zeros((n,n))
160 
161     for edge in boundary:
162         #get start end end point from triangulation
163         S = np.array(mesh['vertices'][edge[0]])
164         E = np.array(mesh['vertices'][edge[1]])
165 
166         #calculate midpoint
167         M = (S + E)/2
168 
169         #calculate edge length
170         edgeLen = np.linalg.norm(S - E)
```

```
171
172         #calculate the value of kappa on this line segment
173         k = kappa(M)
174
175         #calculate the local boundary matrix
176         Re = (k/6)*np.array([[2,1],[1,2]])*edgeLen
177
178         for i in [0,1]:
179             for j in [0,1]:
180                 #transform local indices to global indices
181                 loc2glb_i = edge[i]
182                 loc2glb_j = edge[j]
183                 R[loc2glb_i][loc2glb_j] += Re[i][j]
184     return R
185
186
187 #assuming robin boundary conditons, calculate boundary vector r
188 #INPUT: kappa, array of boundary edges, triangulation of the region
        , boundary condition parameters gD and gN
189 #OUTPUT: boundary matrix R
190 def assembleBoundaryVector(kappa, boundary, tri, gD, gN):
191     #number of nodes
192     n = len(tri['vertices'])
193
194     #allocate global boundary vector r
195     r = np.zeros(n)
196
197     for edge in boundary:
198         #get start end end point from triangulation
199         S = np.array(mesh['vertices'][edge[0]])
200         E = np.array(mesh['vertices'][edge[1]])
201
202         #calculate midpoint
203         M = (S + E)/2
204
205         #calculate edge length
206         edgeLen = np.linalg.norm(S - E)
207
208         #calculate the coefficient on this line segment
209         coeff = kappa(M)*gD(M) + gN(M)
210
211         #calculate the local boundary vector
212         re = coeff*np.array([1,1])*(edgeLen/2)
213
214         for i in [0,1]:
215
216             #transform local indices to global indices
217             loc2glb_i = edge[i]
218             r[loc2glb_i] += re[i]
219     return r
```

```
220
221
222 #assemble boundary conditions for flow potential simulation
223 kappa = lambda P: 10**6 if P[0] > 29.99 else 0
224 gD1 = lambda P: 0
225 gN1 = lambda P: 1 if P[0] < -29.99 else 0
226
227
228 #compute the system and solve for u
229 #u contains the coefficients of the hat functions phi
230 A = assembleStiffnessMatrix(mesh)
231 R = assembleBoundaryMatrix(kappa, boundary, mesh)
232 r = assembleBoundaryVector(kappa, boundary, mesh, gD1, gN1)
233 u = np.linalg.solve(A+R,r)
234
235
236 #visualize the solution via triangular interpolation
237 x = [vertex[0] for vertex in mesh['vertices']]
238 y = [vertex[1] for vertex in mesh['vertices']]
239
240 tri = Triangulation(x,y,mesh['triangles'])
241
242 #U is the final approximate solution and a function of x and y
243 U = LinearTriInterpolator(tri, u)
244
245 #generate an array of values of U(x,y)
246 width =1200
247 height = 400
248 factor = 20
249
250 plot_x = np.zeros((height,width))
251 plot_y = np.zeros((height,width))
252 image = np.zeros((height,width))
253
254 #populate arrays with values for plotting
255 for i in range(height):
256     for j in range(width):
257         x = j/factor - 30
258         y = i/factor - 10
259         plot_x[i][j] = x
260         plot_y[i][j] = y
261         image[i][j] = U(x,y)
262
263
264 #display resulting contour plot
265 #first plot domain border
266 for edge in boundary:
267     x = [mesh['vertices'][edge[0]][0],
268          mesh['vertices'][edge[1]][0]]
269     y = [mesh['vertices'][edge[0]][1],
```

```
            mesh['vertices'][edge[1]][1]]
    plt.plot(x,y,color = 'k')

#plot contour lines over domain border
plt.contour(plot_x, plot_y,image,levels = 12)
plt.xlim(-32, 32)
plt.ylim(-12,12)
plt.show()


#display resulting stream plot
#first plot domain border
for edge in boundary:
    x = [mesh['vertices'][edge[0]][0],
         mesh['vertices'][edge[1]][0]]
    y = [mesh['vertices'][edge[0]][1],
         mesh['vertices'][edge[1]][1]]
    plt.plot(x,y,color = 'k')

#calculate gradient of the potential U(x,y)
v = -np.diff(image[1:, :], axis=1)
u = -np.diff(image[:, 1:], axis=0)

#plot stream lines over domain border
plt.streamplot(plot_x[1:, 1:], plot_y[1:, 1:],v, u)
plt.xlim(-32, 32)
plt.ylim(-12,12)
plt.show()


#plot bernoulli pressure -|(u,v)**2|
pressure = -np.sqrt((np.multiply(u,u) + np.multiply(v,v)))

#first plot domain border
for edge in boundary:
    x = [mesh['vertices'][edge[0]][0],
         mesh['vertices'][edge[1]][0]]
    y = [mesh['vertices'][edge[0]][1],
         mesh['vertices'][edge[1]][1]]
    plt.plot(x,y,color = 'k')

#plot contour lines over domain border
plt.pcolormesh(plot_x[1:,1:], plot_y[1:,1:],pressure)
plt.xlim(-32, 32)
plt.ylim(-12,12)
plt.show()
```

References

[1] Bernd Aulbach. *Gewöhnliche Differenzialgleichungen*. Spektrum Verlag, 2004. ISBN: 978-3-8274-1492-2.

[2] Philippe G. Ciarlet. *The Finite Element Method for Elliptic Problems*. Society for Industrial and Applied Mathematics, 2002. ISBN: 0-89871-514-8.

[3] Richard Courant. *Vorlesungen über Differential und Integralrechnung 2*. Springer Verlag, 1972. ISBN: 0-387-02956-7.

[4] Jörn Dunkel. *Lecture notes on Nonlinear Dynamics: Continuum Systems*. 2014. URL: `https://math.mit.edu/~dunkel/Teach/18.354_2014S/lecture_notes/L08_navier_stokes.pdf` (visited on 15/02/2022).

[5] Aymeric Grodet and Takuya Tsuchiya. 'Finite element approximations of minimal surfaces: algorithms and mesh refinement'. In: *Japan Journal of Industrial and Applied Mathematics* 35.2 (May 2018), pp. 707–725. ISSN: 1868-937X. DOI: `10.1007/s13160-018-0303-2`. URL: `http://dx.doi.org/10.1007/s13160-018-0303-2`.

[6] Arieh Iserles. *A First Course in the Numerical Analysis of Differential Equations*. Cambridge University Press, 2009. ISBN: 9780521734905.

[7] Konrad Königsberger. *Analysis 1*. Springer Verlag, 2004. ISBN: 9783540403715.

[8] M.G. Larson and F. Bengzon. *The Finite Element Method: Theory, Implementation and Applications*. Springer Verlag, 2013. ISBN: 978-3-642-33286-9. DOI: `10.1007/978-3-642-33287-6`.

[9] Christian Schmeiser. *Skriptum zur Vorlesung Partielle Differentialgleichungen*. 2022. URL: `https://homepage.univie.ac.at/christian.schmeiser/allpdg.pdf` (visited on 15/02/2022).

[10] Endre Süli. *Lecture notes on finite element methods for partial differential equations*. 2020. URL: `https://people.maths.ox.ac.uk/suli/fem.pdf` (visited on 15/02/2022).

[11] Kosaku Yosida. *Functional Analysis*. Springer Verlag, 1980. ISBN: 978-3-540-58654-8. DOI: `10.1007/978-3-642-61859-8`.

All images in this thesis were created by the author. Except for Figure 5, they have all been made using the Python library matplotlib, for Figure 5 Microsoft Paint was used.