TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

DIPLOMARBEIT

# Numerical Continuation for Periodic Pipe Flow with Finite Element Method

zur Erlangung des akademischen Grades

**Diplom-Ingenieur**

im Rahmen des Studiums

**Masterstudium technische Mathematik**

eingereicht von

**Dominik Worf**
Matrikelnummer 01025569
Gentzgasse 132/14
1180 Wien

ausgeführt am Institut für Analysis und Scientific Computing
der Fakultät für Mathematik und Geoinformation
der Technischen Universität Wien

Betreuer: Ass. Prof. Dr. Christian Kühn

Wien, 27.03.2018 _____ _____

Verfasser Betreuer

# Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig verfasst habe, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und die aus anderen Quellen entnommenen Stellen, einschließlich Tabellen und Abbildungen, als solche gekennzeichnet habe.

Wien, am 27.03.2018

Dominik Worf

# Abstract

This thesis is concerned with the continuation theory of incompressible periodic pipe flow. For describing the dynamics of incompressible fluids we use the incompressible Navier-Stokes equation. For a better understanding of it we'll look at its derivation.

For a long time now the consensus has been that the laminar solution is linearly stable for all Reynolds numbers. The original idea of this thesis was to adapt a numerical continuation procedure to see if it is possible to jump from the laminar solution branch onto a turbulent one, as it happens in practical experiments.

Therefore we inspect the different numerical methods that are used in this procedure. Especially we look at a preconditioner for the linearized problem as the matrix given by the finite element method, using Hood-Taylor elements, becomes less well conditioned as the Reynolds number increases. Prompted by this we look at the convection-diffusion equation and the streamline diffusion discretisation to be able to use it in a multigrid method.

To motivate the use of the continuation procedure we look at bifurcation theory, with Fredholm operators and Crandall-Rabinowitz' theorem. We also take a short look at the Allen-Cahn equation to test if the algorithm is correctly defined.

**Keywords:** Navier-Stokes equation, finite element method, Hood-Taylor elements, convection-diffusion equation, streamline-diffusion method, preconditioning, bifurcation theory, Allen-Cahn equation

# Kurzfassung

Diese Arbeit befasst sich mit der Pfadverfolgungstheorie inkompressibler periodischer Rohrströmung. Zur Beschreibung der Dynamik von inkompressiblen Fluiden nutzen wir die inkompressible Navier-Stokes Gleichung. Um diese besser zu verstehen betrachten wir auch ihre Herleitung.

Lange Zeit besteht schon der Konsens, dass die laminare Lösung für alle Reynolds-Zahlen linear stabil ist. Die ursprüngliche Idee dieser Arbeit war einen numerischen Pfadverfolgungsalgorithmus zu adaptieren, um zu sehen ob es möglich ist von der laminaren Lösung auf eine turbulente zu springen, so wie es auch in praktischen Experimenten passiert.

Deshalb betrachten wir die verschiedenen numerischen Methoden die in diesem Algorihmus verwendet werden. Insbesondere betrachten wir einen Vorkonditionierer für das linearisierte Problem, da die Matrix aus der Finiten Elemente Methode, mit Hood-Taylor Elementen, mit steigender Reynolds-Zahl immer schlechter konditioniert wird. Das führt uns zur Konvektions-Diffusionsgleichung und zur Streamline-Diffusion Diskretisierung um sie in einer Multigrid Methode zu verwenden.

Um die Pfadverfolgungsmethode zu motivieren betrachten wir auch die Bifurkationstheorie, mit Fredholm Operatoren und dem Satz von Crandall-Rabinowitz. Wir betrachten auch kurz die Allen-Cahn Gleichung um zu testen ob der Algorihmus richtig definiert ist.

**Schlagworte:** Navier-Stokes Gleichung, Finite Elementen Methode, Hood-Taylor Elemente, Konvektions-Diffusionsgleichung, Streamline-Diffusion Methode, Vorkonditionierung, Bifurkationstheorie, Allen-Cahn Gleichung

# Acknowledgements

First I want to thank my thesis advisor Prof. Christian Kühn for all the time he has invested in tutoring and giving his advice. I also want to thank him for his enormous patience after this thesis took way longer than expected.

I want to thank Prof. Joachim Schöberl and his team for answering all my questions and giving me advice regarding their program Netgen/NGSolve.

I thank my proofreader Isabella for reading the text with the unbiased eyes of someone who isn't already used to every notation there is in FEM and also for steadily pressuring me to work harder.

Also I want to thank all my friends and family who have supported me during my studies.

# Contents

# List of Figures

# Chapter 1

# Introduction

How and why does laminar flow become turbulent? This question is as old as fluid dynamics itself. It is very important since turbulent flow needs more energy than laminar flow. Therefore pipe flow is one of the classical problems of stability theory. We now know that the laminar flow in pipes is linearly stable. This means that transition can only be started by finite amplitude disturbances. Transitional flow thus is delicate but leads abruptly to a very disordered motion. It is mostly areas of turbulence seperated by regions of laminar flow.

Due to the cylindrical geometry of pipe flow this transition process starts at higher relative flow rates than in planar flow. This is the reason why it is the least studied type theoretically although it is the easiest to realise in the lab.

In the next two sections I'll give a short overview over the history of this subject summarised from [9].

I want to construct a bifurcation diagram for the problem of periodic pipe flow using the finite element method. This thesis is structured into two parts: at first we'll look at the theory of the problem, in the second part we'll look at the computational experiments. In chapter 2 we'll look at the numerical methods used, especially finite lement method for mixed problems. Afterwards we'll look at the Navier-Stokes equation itself and its numerical properties in chapter 3. In the last chapter of the theoretical part we'll look at bifurcation theory and the algorithm used to construct the bifurcation diagram. Finally in chapter 5 I will show the results of my computations.

## 1.1 Experimental history

The first serious experimental studies of pipe flow were carried out by Gotthilf Hagen in 1839 and Jean Poiseuille in 1840. Poiseuille concentrated on small capillaries of 0.015 to 0.6 mm and therefore only got laminar flow

while Hagen used pipes of larger diameter of about 2.5 to 6 mm. Thus he also got the unsteady three-dimensional flow.

Due to their work pipe flow is nowadays known as Hagen-Poiseuille flow (HPF).

In 1883 Reynolds realised that a non-dimensional number was enough to characterise flow. This lead to the Reynolds-number $Re$.

He also discovered that the transitional Reynolds-number $Re^t$ varied according to the level of disturbance. He found that the earliest onset of turbulence was at about $Re^t \approx 2000$, but was also able to push the transition to about $Re^t \approx 12000$ in more tightly controlled experiments. Since then the lower number was confirmed many times while the higher number was able to be pushed even further ($Re^t \approx 100000$).

The implication of this is that there exists a threshold of disturbances which decreases as $Re$ increases.

Reynolds also found that the flow becomes turbulent in patches seperated by laminar areas.

This was later studied by Wygnanski (1970s) and he found two different states, called puffs and slugs:

- Puffs are turbulent regions with sharp upstream and blurred downstream boundaries. These are found for $2000 \lesssim Re \lesssim 2700$ and have a length of about 20-30 pipe diameters. The speeds of the up- and downstream borders are roughly equal and a bit less than the mean flow velocity. Thus fluid passes through puffs and they can be seen as an incomplete relaminarisation process.

- Slugs are also turbulent regions with sharp upstream but also sharp downstream boundaries. Flow inside a slug looks equal to turbulent pipe flow. Slugs are found for $Re > 3200$ where the upstream front moves slower than the mean flow while the downstream front moves faster. This means that fluid in a slug is trapped and never relaminarises.

In 1995 Darbyshire and Mullin mapped the finite amplitude threshold curve. Afterwards Draad et al. (1998) found that this curve is sensitive to the frequency of perturbations. It also depends on their azimuthal structure. Hof et al. found that this can be lowered to a $Re^{-1}$ scaling for $2000 < Re < 20000$ if the perturbation is applied long enough.

## 1.2 Theoretical history

The first theoretical work put small perturbations on HPF. Rayleigh found in 1892 that inviscid infinitesimal disturbances don't grow, thus we know that HPF is inviscidly linearly stable. Sexl incorporated in 1927 the effects of viscosity and found that HPF is also stable to axisymmetric disturbances.

After these findings many studies concentrated an asymmetric disturbances. Now the consensus is that it is linearly stable, although no proof for this is found yet (as far as I have found).

On the other side Joseph and Carmi found in 1969 that HPF is a global attractor for $Re < 81.49$. This leaves a large gap between 81.49 and 2000 which isn't really explained. The gap could be closed by rotating the pipe as the transitional Reynolds-number $Re^t$ falls to 82.88 in the limit as the rotatioin rate goes to infinity but that completely changes the problem.

In the early nineties people started to focus on the ability of certain disturbances to temporarily grow algebraicly in shear flows. Physically this happens if a small initial disturbance with some wall-normal velocities is introduced. The wall-normal velocities only decay weakly, this leads to a slow advection of the mean shear which can produce large local anomalies in the streamwise velocity called streaks. Mathematically this is due to the non-normality of the linear operator.

This non-normality means that the eigenfunctions of this operator are not orthogonal (in the energy norm) which means that some initial conditions are poorly spanned. This means that some coefficients in the eigenfunction expansion need to be very large due to some eigenfunctions canceling. When the eigenfunctions then decay with different rates it looses this initial cancellation and the large coefficients become significant. This leads to a temporary period of algebraic growth. (In [9] a simple example for this process is given).

In pipe flow these inital conditions are two-dimensional streamwise independent vortices (rolls) with an azimuthal wavenumber of one. These can lead to 'streaks' or azimuthal (spanwise) variations in the mean flow.

In this linearized setting the inital disturbance of streamwise rolls can be amplified by $O(Re^2)$ in energy (while changing into streaks) before decaying in $O(Re)$ time.

Then people looked at how energy could be fed back from streaks into rolls. Zikanov (1996) studied three-dimensional linear instability of pipe flow when adding two-dimensional streamwise rolls. He found that if rolls induce large enough streaks inflection points appear which are unstable to three-dimensional disturbances.

During this time the question shifted from how turbulence is initiated to how it is maintained.

Waleffe (1995) found a spatially and temporally organised cycle of events during turbulence. It has three phases:

- formation of streaks by streamwise vortices,
- breakdown of streaks,
- regeneration of streamwise vortices.

The last phase was least understood since it is fundamentally nonlinear but it looked like the streaks directly regenerated the rolls. He explored this by confirming the feasibility of each phase in isolation. With this he showed that the streak instability could feed energy back into the rolls. Thus he found a potentially self-sustaining process (SSP).

He converted this into a smooth numerical continuation procedure. With this he got nonlinear steady state solutions and travelling wave solutions for plane Couette and Poiseuille flow to arbitrary accuracy.

These consist of the three flow structures discussed such that they maintain each other against viscous decay.

People thought that one of these solutions might be an organising centre for the found quasi-cycle.

In 2001 Kawahara and Kida confirmed that there exists a periodic orbit which uses a version of Waleffe's SSP.

The appearance of a streak instability is now considered enough to signify transition.

Ma et al (1999) confirmed numerically this sequence:

$$\text{rolls} \rightarrow \text{streaks} \rightarrow \text{streak instability} \rightarrow \text{turbulence}.$$

The biggest problem for numerical simulation is to ensure that the pipe is long enough while still having a usable resolution that transitional structures can evolve. This is a problem only in transitional flow, in fully developed turbulent flow this is no problem (see [9]).

# Chapter 2

# Numerical Methods

In this chapter we'll look at the different numerical methods that are used in the continuation algorithm. We'll start with the Galerkin method and the finite element method (FEM) in particular. FEM is a method for approximating the solution of the weak form of a partial differential equation (PDE) in a finite dimensional subspace.

The main sources for the background are [1], [3] and [12].

## 2.1 Galerkin method

A linear PDE in its weak form is given by a bilinearform $a(u, v)$ and a linearform $f(v)$ on a space $V$:

$$\text{find } u \in V: \qquad a(u, v) = f(v) \qquad \forall v \in V.$$

The idea of the Galerkin method is to solve this problem in a finite dimensional subspace $V_h \subset V$:

$$\text{find } u_h \in V_h: \qquad a(u_h, v_h) = f(v_h) \qquad \forall v_h \in V_h.$$

## 2.2 Coercive problems

To gain existence and uniqueness of a solution we need the following definition.

**Definition 1.** *A bilinearform $a(u, v)$ on a Hilbertspace $V$ is called*

1. *continuous if there exists $C > 0$ such that*

$$a(u, v) \leq C\|u\|_V\|v\|_V \qquad \forall u, v \in V.$$

2. *coercive if there exists $\alpha > 0$ such that*

$$a(u, u) \geq \alpha\|u\|^2 \qquad \forall u \in V.$$

With these properties we can prove existence and uniqueness with Lax-Milgram:

**Theorem 1.** *Given a Hilbertspace $V$, a coercive and continuous bilinearform $a(\cdot,\cdot)$ and a continuous linearform $f(\cdot)$. Then there exists a unique solution $u \in V$ of*

$$a(u,v) = f(v) \qquad \forall v \in V,$$

*There holds*

$$\|u\|_V \leq \alpha^{-1}\|f\|_{V^*}.$$

*Proof.* The proof of this theorem can be found in [12]. □

For the approximating problem the solvability is inherited from the original problem. The approximation error is quasi-optimal:

**Theorem 2** (Cea)**.** *The approximating error of the Galerkin method is quasi-optimal*

$$\|u - u_h\|_V \leq \frac{C}{\alpha} \inf_{v_h \in V_h} \|u - v_h\|_V.$$

*Proof.* A very important property is the Galerkin orthogonality: $u \in V$ also solves the discrete problem

$$a(u, v_h) = f(v_h) \qquad \forall v_h \in V_h.$$

Therefore

$$a(u - u_h, v_h) = a(u, v_h) - a(u_h, v_h) = f(v_h) - f(v_h) = 0 \qquad \forall v_h \in V_h.$$

Now with an arbitrary $v_h \in V_h$

$$\begin{aligned}
\|u - u_h\|_V^2 &\leq \frac{1}{\alpha} a(u - u_h, u - u_h) \\
&= \frac{1}{\alpha} a(u - u_h, u - v_h) + \underbrace{\frac{1}{\alpha} a(u - u_h, \underbrace{v_h - u_h}_{\in V_h})}_{=0} \\
&\leq \frac{C}{\alpha} \|u - u_h\|_V \|u - v_h\|_V.
\end{aligned}$$

By dividing $\|u - v_h\|_V$ we get an estimate which also holds for the infimum since $v_h$ was arbitrary. □

## 2.3 Inf-sup stable problems

It is not always possible to show coercivity, therefore we need a weaker property to show solvability.

**Definition 2.** *Let $V$ and $W$ be Hilbertspaces. A continuous bilinearform $b(\cdot,\cdot) : V \times W \to \mathbb{R}$, fulfills the inf-sup condition if there exists $\beta > 0$ such that*

$$\inf_{\substack{u\in V,\\ u\neq 0}} \sup_{\substack{v\in W,\\ v\neq 0}} \frac{b(u,v)}{\|u\|_V \|v\|_W} \geq \beta.$$

We also need another condition in the other direction

$$\exists \beta_1 > 0 : \qquad \inf_{\substack{v\in W,\\ v\neq 0}} \sup_{\substack{u\in V,\\ u\neq 0}} \frac{b(u,v)}{\|u\|_V \|v\|_W} \geq \beta_1.$$

Here it suffices to use the weaker condition

$$\sup_{\substack{u\in V,\\ u\neq 0}} \frac{b(u,v)}{\|u\|_V \|v\|_W} > 0 \qquad \forall v \in W. \tag{2.1}$$

**Theorem 3.** *Let $b(\cdot,\cdot)$ be a continuous bilinearform which fulfills the inf-sup condition (Def. 2) and condition* (2.1)*. Then*

$$b(u,v) = f(v) \qquad \forall v \in W.$$

*has a unique solution. This problem is then called inf-sup-stable. The solution depends continuously on the righthandside*

$$\|u\|_V \leq \beta_1^{-1} \|f\|_{W^*}.$$

*Proof.* The proof of this theorem can be found in [12]. □

For the approximating problem of inf-sup-stable problems the solvability isn't inherited, so we need an extra inf-sup condition for the discrete problem

$$\inf_{\substack{u_h\in V_h,\\ u_h\neq 0}} \sup_{\substack{v_h\in W_h,\\ v_h\neq 0}} \frac{b(u_h,v_h)}{\|u_h\|_V \|v_h\|_W} \geq \beta_h. \tag{2.2}$$

On a finite dimensional space we don't need the second condition since injectivity is equivalent to surjectivity.

Again we can show quasi-optimality

**Theorem 4.** *Let $b(\cdot,\cdot)$ be a continuous bilinearform which fulfills the discrete inf-sup condition* (2.2)*. Then there holds*

$$\|u-u_h\|_V \leq (1+\frac{C}{\beta_h}) \inf_{v_h \in V_h} \|u-v_h\|_V.$$

*Proof.* Again Galerkin orthogonality $b(u,w_h) = b(u_h,w_h)$ holds for all $w_h \in W_h$. Again choose an arbitrary $v_h \in V_h$:

$$\begin{aligned}
\|u-u_h\|_V &\leq \|u-v_h\|_V + \|v_h-u_h\|_V \\
&\leq \|u-v_h\|_V + \frac{1}{\beta_h} \sup_{\substack{w_h \in W_h, \\ w_h \neq 0}} \frac{b(v_h-u_h,w_h)}{\|w_h\|_W} \\
&= \|u-v_h\|_V + \frac{1}{\beta_h} \sup_{\substack{w_h \in W_h, \\ w_h \neq 0}} \frac{b(v_h-u,w_h)}{\|w_h\|_W} \\
&\leq \|u-v_h\|_V + \frac{1}{\beta_h} \sup_{\substack{w_h \in W_h, \\ w_h \neq 0}} \frac{C\|v_h-u\|_V\|w_h\|_W}{\|w_h\|_W} \\
&= (1+\frac{C}{\beta_h})\|u-v_h\|_V.
\end{aligned}$$

□

## 2.4 Mixed problems

The linearised Navier-Stokes equation looks a bit different than the general problem we looked at before. It uses two Hilbert spaces $H^1$ and $L^2$ and two bilinearforms. Therefore we need the theory of mixed problems. A mixed problem uses two Hilbert spaces $V$ and $Q$, bilinearforms

$$\begin{aligned}
a(u,v) : V \times V &\to \mathbb{R}, \\
b(u,q) : V \times Q &\to \mathbb{R}
\end{aligned}$$

and continuous linearforms

$$\begin{aligned}
f(v) : V &\to \mathbb{R}, \\
g(q) : Q &\to \mathbb{R}.
\end{aligned}$$

The problem then is to find $u \in V$, $p \in Q$ such that

$$\begin{aligned}
a(u,v) + b(v,p) &= f(v) \quad \forall v \in V, \\
b(u,q) &= g(q) \quad \forall q \in Q.
\end{aligned}$$

One can see this as one big equation with one big bilinearform in the space $V \times Q$:

$$B((u,p),(v,q)) = a(u,v)+b(v,p)+b(u,q) = f(v)+g(q) \qquad (v,q) \in V \times Q.$$

If this bilinearform fulfills the conditions of theorem 3 there exists a unique solution. To get this, we can just look at the mixed problem with Brezzi's theorem.

**Theorem 5** (Brezzi)**.** *Let $a(\cdot,\cdot)$ and $b(\cdot,\cdot)$ be continuous bilinearforms*

$$\begin{aligned} a(u,v) &\leq \alpha_2 \|u\|_V \|v\|_V \qquad \forall u, v \in V, \\ b(u,q) &\leq \beta_2 \|u\|_V \|q\|_Q \qquad \forall u \in V,\ \forall q \in Q. \end{aligned}$$

*Assume $a(\cdot,\cdot)$ is coercive on the kernel,*

$$a(u,u) \geq \alpha_1 \|u\|_V^2 \qquad \forall u \in V_0 = \{v \in V :\ b(v,q) = 0,\ \forall q \in Q\},$$

*and there holds the LBB (Ladyshenskaja-Babuška-Brezzi) condition*

$$\sup_{\substack{u \in V, \\ u \neq 0}} \frac{b(u,q)}{\|u\|_V} \geq \beta_1 \|q\|_Q \qquad \forall q \in Q.$$

*Then, the mixed problem is uniquely solvable. The solution fulfills*

$$\|u\|_V + \|p\|_Q \leq c(\|f\|_{V^*} + \|g\|_{Q^*})$$

*with the constant $c$ depending on $\alpha_1,\ \alpha_2,\ \beta_1,\ \beta_2$.*

*Proof.* The proof of this theorem can be found in [12]. □

Again to show the solvability of the discrete problem we have to show the discrete LBB condition

$$\sup_{\substack{u_h \in V_h, \\ u_h \neq 0}} \frac{b(u_h,q_h)}{\|u_h\|_V} \geq \beta_1 \|q_h\|_Q \qquad \forall q_h \in Q_h.$$

## 2.5 Finite element method

In the sections before we discussed the Galerkin approximation. The finite element method is a special case of the Galerkin approximation. The idea is to choose a special finite dimensional subspace $V_h$ of $V$ to work with simple basis functions. FEM uses a space of piecewise polynomial functions, since it leads to sparse matrices. This is done by triangulating the domain $\Omega$ on which the PDE is defined. In case of 2D-problems this is done by partitioning into triangles, in the case of 3D-problems this is done by partitioning into tetrahedrons. (It can also be done by using convex quadrilaterals (2D) or bricks (3D).) The triangulation of $\Omega$ will be denoted by $\mathcal{T}$.

There are certain properties that are important descriptors of the quality of the triangulation. The first one describes the coarseness of the mesh, by defining on each triangle (or tetrahedron) $T$

$$h_T := \operatorname{diam}(T)$$

and taking the maximum of this $h := \max_{T\in\mathcal{T}} h_T$. The shape of the triangles is also very important and can be described by

$$\rho_T := \sup\{\text{diam}(B) : B \text{ is a ball contained in } T\}.$$

Together these quantities give the regularity of $T$ by

$$\sigma_T := \frac{h_T}{\rho_T}.$$

**Definition 3** (regular triangulation)**.** *A family of triangulations $\mathcal{T}_h$ of $\overline{\Omega}$ is called regular as $h \to 0$ if there exists a constant $\sigma > 0$, independent of $h$ and $T$, such that*

$$\sigma_T \leq \sigma \quad \forall T \in \mathcal{T}_h.$$

*$\mathcal{T}_h$ is called uniformly regular for $h \to 0$ if there exists another constant $\tau > 0$ such that*

$$\tau \cdot h \leq h_T \leq \sigma \cdot \rho_T \quad \forall T \in \mathcal{T}_h.$$

Altough the problem is 3D we will look at 2D here because it is easier to describe but it is the same principle. The subspace for $V$ can be described with

$$V_h = \{v \in V : \forall T \in \mathcal{T} : v|_T \in \mathcal{P}_k(T)\}.$$

Where $\mathcal{P}_k(\Omega)$ is the space of all polynomials of order$\leq k$ on $\Omega$. This is done by using basis functions on the seperate triangles or patches. This leads to Ciarlet's definition of finite elements.

**Definition 4** (Finite element)**.** *A finite element is a triple $(T, V_T, \Psi_T)$ where*

1. *$T$ is a bounded set,*
2. *$V_T$ is a function space on $T$ of finite dimension $N_T$,*
3. *$\Psi_T = \{\psi_T^1, \ldots, \psi_T^{N_T}\}$ is a set of linearly independent functionals on $V_T$.*

Often the finite element is simply denoted by its bounded set $T$. The basis $\{\phi_T^1, \ldots, \phi_T^{N_T}\}$ for $V_T$ dual to $\Psi_T$, i.e.

$$\psi_T^i(\phi_T^j) = \delta_{ij},$$

is called the nodal basis.

The most famous version are the hat functions which are defined on the vertex patch, which is the set of triangles which contain the vertex ($w_v = \bigcup_{v\in T} T$), by constructing a function which is 1 on the vertex, 0 on the other vertices and linear on the triangles (fig. 2.1). The dual basis $\psi_T$ then are the point evaluation functionals on the vertices $\psi_i(u_h) = u_h(v_i)$.

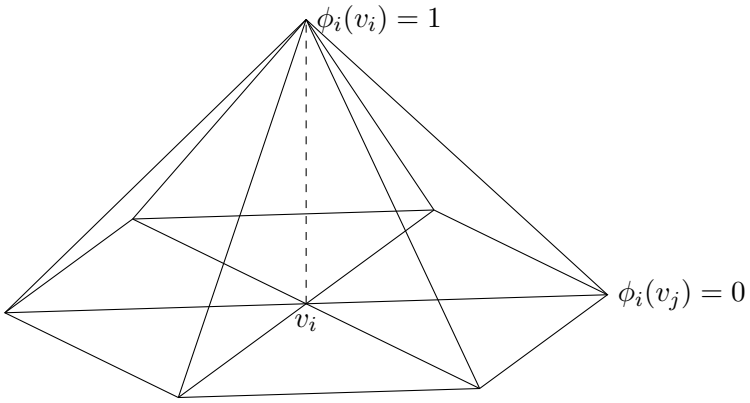


Figure 2.1: A hatfunction visualized.

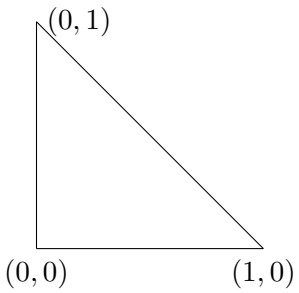


Figure 2.2: The most common reference triangle $\hat{T}$.

A function $u_h \in V_h$ then can be described by

$$u_h = \sum_{i=1}^{N} u_i \phi_i \quad , u_i = \psi_i(u_h).$$

The $u_i$ then are called degrees of freedom.

When implementing these elements one usually defines a finite element on a reference triangle (fig. 2.2) and then uses equivalence to define it on more general triangles.

**Definition 5.** *Two finite elements $(T, V_T, \Psi_T)$ and $(\hat{T}, V_{\hat{T}}, \Psi_{\hat{T}})$ are called equivalent if there exists an invertible function $F$ such that*

- $T = F(\hat{T})$,
- $V_T = \{\hat{v} \circ F^{-1} : \hat{v} \in V_{\hat{T}}\}$,
- $\Psi_T = \{\psi_i^T : V_T \to \mathbb{R} : v \mapsto \psi_i^{\hat{T}}(v \circ F)\}$.

*They are called interpolation equivalent if*

$$I_T(v) \circ F = I_{\hat{T}}(v \circ F) \quad \forall v \in V_T,$$

*where $I_T$, $I_{\hat{T}}$ are the so-called local nodal interpolation operators*

$$I_T v := \sum_{\alpha=1}^{N_T} \psi_T^\alpha(v)\phi_T^\alpha.$$

For triangles the function $F_T$ then can be the affine mapping of the reference triangle onto the general triangle

$$F_T(\hat{\mathbf{x}}) = B_T\hat{\mathbf{x}} + \mathbf{b}_T,$$

where $B_T$ is a suitable matrix in $\mathbb{R}^{2\times 2}$ and $\mathbf{b}_T$ a suitable vector in $\mathbb{R}^2$. There are certain properties of this mapping that can be important for showing the necessary conditions on the finite element space (e.g. the discrete LBB-condition). When $\hat{v} = v \circ F_T$ there holds (see [6])

$$\begin{aligned}
\|B_T\| &\leq \frac{h_T}{\rho_{\hat{T}}}, \\
\|B_T^{-1}\| &\leq \frac{h_{\hat{T}}}{\rho_T}, \\
|v|_{m,p,T} &\leq C_1\|B_T^{-1}\|^m|\det(B_T)|^{\frac{1}{p}}|\hat{v}|_{m,p,\hat{T}} \quad \forall \hat{v} \in W^{m,p}(\hat{T}), \\
|\hat{v}|_{m,p,\hat{T}} &\leq C_2\|B_T\|^m|\det(B_T)|^{-\frac{1}{p}}|v|_{m,p,T} \quad \forall v \in W^{m,p}(T).
\end{aligned} \tag{2.3}$$

### 2.5.1 Finite element system assembling

The finite element problem is

$$\text{Find } u_h \in V_h \text{ such that } a(u_h, v_h) = f(v_h) \quad \forall v_h \in V_h.$$

We describe $u_h$ as its basis expansion

$$u_h = \sum_{i=1}^{N} u_i\phi_i,$$

also we can reduce the number of testfunctions $v_h$ to the basis functions

$$a(\sum_{i=1}^{N} u_i\phi_i, \phi_j) = f(\phi_j) \quad \forall j = 1 \ldots N.$$

With

$$A_{ij} := a(\phi_i, \phi_j) \text{ and } f_j := f(\phi_j)$$

we get a linear equation system

$$A\mathbf{u} = \mathbf{f}.$$

If we then solve this equation we get the coefficients $u_i$ of the solution $u_h$ and therefore the solution of the finite element problem.

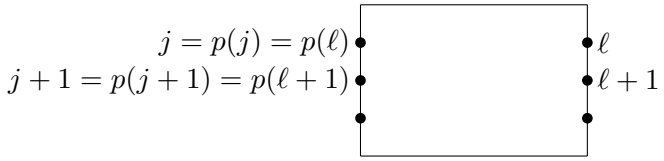


Figure 2.3: The mapping $p$ visualized

### 2.5.2 Boundary conditions

We still need to look at how to implement boundary conditions. We will look at Dirichlet boundary conditions and periodic boundary conditions (Neumann boundary conditions are already in the weak formulation and therefore taken care of).

In the case of Dirichlet boundary conditions we split the degrees of freedom $\{1, \ldots, N\}$ into those that correspond to points on the Dirichlet boundary $\gamma_D \subset \{1, \ldots, N\}$ and those that don't $\gamma_f = \{1, \ldots, N\} \backslash \gamma_D$. For the hatfunctions these represent the vertices on the boundary, for different elements this can be the edges or similar shapes (e.g. faces in 3D).

In the case of periodic boundary conditions we already have to keep track of the elements when constructing the mesh because for every degree of freedom on one side there has to be one on the other side so that they can be mapped onto each other. This means every surface element on one side looks like the corresponding one on the other side. We introduce a mapping $p$ which maps all degrees of freedom of one side onto the corresponding degrees of freedom on the other side while keeping mapped onto degrees of freedom the same and keeping degrees of freedom which are not in the periodic boundary the same (see figure 2.5.2.

When we then assemble the linear equation system we get a reduced system with the reduced matrix $A_{per}$ and the reduced vector $f_{per}$

$$A_{per} u_{per} = f_{per}.$$

Where for every entry in $A_{per}$ of a degree of freedom which gets mapped onto we also have to add the contribution of the mapped degree of freedom

$$a_{per,p(l)=p(i),p(k)=p(j)} = a_{i,j} + a_{l,k},$$

similarly for $f_{per}$

$$f_{per,p(l)=p(i)} = f_l + f_i.$$

This is often already done during the assembly by simply adding the element-contributions at the right spot.

After we have solved the linear equation system we get the full solution out of our reduced solution by simply setting

$$u_i = u_{per,p(i)}.$$

## 2.6 Eigenvalue calculation

We'll need to calculate eigenvalues to gain statements about stability. I used the function **eigs** from scipy [7] which in turn uses ARPACK [11]. The functions of ARPACK are based on the implicitely restarted Arnoldi algorithm (IRA).

I'll now describe the Arnoldi algorithm and how it is used for the calculation of eigenvalues. For an in depth view on large scale eigenvalue problems see [1].

### 2.6.1 Arnoldi algorithm and approximate eigenvalue calculation

The Arnoldi algorithm is an algorithm to calculate an orthonormal basis of the Krylov-subspace

$$\mathcal{K}^j(\mathbf{x}) := \mathcal{K}^j(\mathbf{x}, A) := \operatorname{span}\{\mathbf{x}, A\mathbf{x}, \ldots, A^{j-1}\mathbf{x}\}$$

by using the Gram-Schmidt orthogonalization process. Let $\{\mathbf{q}_1, \ldots, \mathbf{q}_i\}$ be an orthonormal basis of $\mathcal{K}^i(\mathbf{x})$ for $i \leq j$ define

$$\begin{aligned} \mathbf{y}_j &:= A^j\mathbf{x} - \sum_{i=1}^{j} \mathbf{q}_i\mathbf{q}_i^* A^j\mathbf{x}, \\ \mathbf{q}_{j+1} &:= \frac{\mathbf{y}_j}{\|\mathbf{y}_j\|}. \end{aligned}$$

The orthonormal basis $\{\mathbf{q}_1, \ldots, \mathbf{q}_{j+1}\}$ of $\mathcal{K}^{j+1}(\mathbf{x})$ reached this way is called Arnoldi basis and the $\mathbf{q}_i$ are called Arnoldi vectors. Since

$$\mathcal{K}^{j+1}(\mathbf{x}) = \operatorname{span}\{\mathbf{q}_1, \ldots, \mathbf{q}_j, A\mathbf{q}_j\},$$

it's more economical to orthogonalize $A\mathbf{q}_j$ against $\mathbf{q}_1, \ldots, \mathbf{q}_j$ instead of $A^j\mathbf{q}$. The component $\mathbf{r}_j$ of $A\mathbf{q}_j$ orthogonal to $\{\mathbf{q}_1, \ldots, \mathbf{q}_j\}$ then is

$$\mathbf{r}_j = A\mathbf{q}_j - \sum_{i=1}^{j} \mathbf{q}_i(\mathbf{q}_i^* A\mathbf{q}_j).$$

If $\mathbf{r}_j = 0$ the procedure stops and we have found an invariant subspace under $A$, i.e.

$$\mathbf{q} \in \mathcal{K}^j(\mathbf{x}) \Rightarrow A\mathbf{q} \in \mathcal{K}^j(\mathbf{x}).$$

If $\|\mathbf{r}_j\| > 0$ we set

$$\mathbf{q}_{j+1} = \frac{\mathbf{r}_j}{\|\mathbf{r}\|}.$$

It follows that

$$\mathbf{q}_{j+1}^* \mathbf{r}_j = \|\mathbf{r}_j\| = \mathbf{q}_{j+1}^* A\mathbf{q}_j,$$

where the second equation holds true because $\mathbf{q}_{j+1}$ is orthogonal to all previous Arnoldi vectors.. Define

$$h_{i,j} := \mathbf{q}_i^* A \mathbf{q}_j,$$

then

$$A\mathbf{q}_j = \sum_{i=1}^{j+1} \mathbf{q}_i \mathbf{h}_{i,j}. \tag{2.4}$$

**Algorithm 2.1** Arnoldi algorithm

Let $A \in \mathbb{F}^{n \times n}$.
$\mathbf{q}_1 := \mathbf{x}/\|\mathbf{x}\|_2$
**for** $j = 1, \ldots, k$ **do**
  $\mathbf{r} := A\mathbf{q}_j$
  **for** $i = 1, \ldots, j$ **do**
    $h_{i,j} := \mathbf{q}^*\mathbf{r}$
    $\mathbf{r} := \mathbf{r} - \mathbf{q}_i h_{i,j}$
  **end for**
  $h_{j+1,j} := \|\mathbf{r}\|$
  **if** $h_{j+1,j} = 0$ **then**
    **return** $(\mathbf{q}_1, \ldots, \mathbf{q}_j, H \in \mathbb{F}^{j \times j})$
  **end if**
  $\mathbf{q}_{j+1} := \mathbf{r}/h_{j+1,j}$
**end for**
**return** $(\mathbf{q}_1, \ldots, \mathbf{q}_{k+1}, H \in \mathbb{F}^{k+1 \times k})$

The resulting matrix $H_k$ then is an upper Hesenberg matrix. If we define $Q_k := (\mathbf{q}_1, \ldots, \mathbf{q}_k)$ with (2.4) we get

$$AQ_k = Q_k H_k + \mathbf{q}_{k+1} h_{k+1,k} \mathbf{e}_k^* \tag{2.5}$$

which is called Arnoldi relation.

The Arnoldi algorithm is very expensive and gets more expensive in each step (in both memory cost and computational cost). Therefore often a restarted version is used, which in general means that an algorithm is repeated with an initial vector gained from a previous use of the algorithm.

If $h_{m+1,m} = 0$ the algorithm stops and from (2.5) we get

$$AQ_m = Q_m H_m.$$

We see that eigenvalues of $H_m$ are eigenvalues of $A$. Vectors $Q_m\mathbf{y}$ where y is any eigenvector of $H_m$ are called Ritz vectors. The corresponding eigenvalues of $H_m$ are then called Ritz values. In this case Ritz vectors are eigenvectors of $A$.

Most of the time $m$ with $h_{m+1,m} = 0$ will be too big to compute Arnoldi up to this $m$. It often suffices to use a Krylov subspace with small dimension for approximating eigenvalues and eigenvectors. If we only want a small number of eigenvalues then a small dimension of the search space suffices.

With (2.5) we get estimates for the eigenvalue/eigenvector residual. Let $\mathbf{u}_i^{(k)} = Q_k \mathbf{s}_i^{(k)}$ be a Ritz vector with Ritz value $\theta_i^{(k)}$, then

$$\begin{aligned} A\mathbf{u}_i^{(k)} - \theta_i^{(k)}\mathbf{u}_i^{(k)} &= AQ_k\mathbf{s}_i^{(k)} - \theta_i^{(k)}Q_k\mathbf{s}_i^{(k)} \\ &= (AQ_k - Q_kH_k)\mathbf{s}_i^{(k)} = h_{k+1,k}\mathbf{q}_{k+1}\mathbf{e}_k^*\mathbf{s}_i^{(k)}. \end{aligned}$$

Therefore we have

$$\|(A - \theta_i^{(k)}I)\mathbf{u}_i^{(k)}\|_2 = h_{k+1,k}|\mathbf{e}_k^*\mathbf{s}_i^{(k)}|.$$

Since $H_m$ is a small upper Hessenberg matrix the calculation of its eigenvectors and eigenvalues is easily done by QR decomposition.

### 2.6.2 Implicitely restarted Arnoldi

We start by computing $m > n$ steps of Arnoldi where $n$ is the number of eigenvalues we want to compute.

**Algorithm 2.2** m-step Arnoldi iteration

$\mathbf{q}_1 := \mathbf{x}/\|\mathbf{x}\|;\ \mathbf{z} := A\mathbf{q}_1;\ \alpha_1 := \mathbf{q}_1^*\mathbf{z}$
$\mathbf{r}_1 := \mathbf{z} - \alpha_1\mathbf{q}_1;\ Q_1 := (\mathbf{q}_1);\ H_1 := (\alpha_1)$
**for** $j = 1, \ldots, m-1$ **do**
  $\beta_j := \|\mathbf{r}\|$
  $\mathbf{q}_{j+1} := \mathbf{r}_j/\beta_j$
  $Q_{j+1} := (Q_j, \mathbf{q}_{j+1})$
  $\hat{H}_j := \begin{pmatrix} H_j \\ \beta_j\mathbf{e}_j^* \end{pmatrix} \in \mathbb{F}^{j+1,j}$
  $\mathbf{z} := A\mathbf{q}_j$
  $\mathbf{h} := Q_{j+1}^*\mathbf{z};\ \mathbf{r}_{j+1} := \mathbf{z} - Q_{j+1}\mathbf{h}$
  $H_{j+1} := (\hat{H}_j, \mathbf{h})$
**end for**

**Remark.** Classical Gram-Schmidt

$$\mathbf{h} := Q_{j+1}^*\mathbf{z}, \quad \mathbf{r}_{j+1} := \mathbf{z} - Q_{j+1}\mathbf{h}$$

may not lead to sufficient orthogonality. So, often the orthogonalization is iterated

$$\begin{aligned} &\mathbf{h} := Q_{j+1}^*\mathbf{z}, && \mathbf{r}_{j+1} := \mathbf{z} - Q_{j+1}\mathbf{h} && \\ &\mathbf{c} := Q_{j+1}^*\mathbf{r}_{j+1}, && \mathbf{r}_{j+1} := \mathbf{r}_{j+1} - Q_{j+1}\mathbf{c} && \mathbf{h} := \mathbf{h} + \mathbf{c}. \end{aligned}$$

This may be repeated but is seldomly necessary. □
This gives us the Arnoldi relation

$$AQ_m = Q_m H_m + \mathbf{r}_m \mathbf{e}_m^*$$

with

$$\mathbf{r}_m = \beta_m \mathbf{q}_{m+1}, \quad \|\mathbf{q}_{m+1}\|_2 = 1.$$

If $\beta_m = 0$ then we have seen that the Ritz values and vectors are eigenvalues and eigenvectors of $A$.

Realistically we hope that $\beta_m$ is small. Then

$$AQ_m - \mathbf{r}_m \mathbf{e}_m^* = (A - \mathbf{r}_m \mathbf{q}_m^*) Q_m = Q_m H_m.$$

Therefore $\text{ran}(Q_m)$ is invariant under $A + E$ with $E$ being a small perturbation with $\|E\| = \|\mathbf{r}_m\| = |\beta_m|$. This gives us that well-conditioned eigenvalues of $H_m$ are good approximations of eigenvalues of $A$.

Now we ask ourselves how to get a starting vector $\mathbf{q}_1$ such that $\beta_m$ becomes small, while we get closer to our desired eigenvalues?

We do this by applying $k < m$ implicit QR steps with shifts $\mu_1, \ldots, \mu_k$ to $H_m$. This is done in alg. 2.3.

**Algorithm 2.3** $k$ implicit QR-steps on $H_m$

$H_m^+ = H_m$
**for** $i = 1, \ldots, k$ **do**
  $H_m^+ := V_i^* H_m^+ V_i$  with $H_m^+ - \mu_i I = V_i R_i$ as the QR-decomposition
**end for**

How do we choose these shifts? The strategy employed by ARPACK calculates the eigenvalues of $H_m$ and those which fit least our desired eigenvalues are chosen as shifts. We'll see why this makes sense after we have looked at what we get from our QR-steps.

We define

$$V^+ := V_1 \cdots V_k$$

which is a product of $k$ unitary Hessenberg matrices, therefore it is a unitary matrix with $k$ nonzero off-diagonals below the main diagonal. Define

$$Q_m^+ := Q_m V^+, \quad H_m^+ := (V^+)^* H_m V^+.$$

Then

$$AQ_m V^+ = Q_m V^+ (V^+)^* H_m V^+ + \mathbf{r}_m \mathbf{e}_m^* V^+$$

or

$$AQ_m^+ = Q_m^+ H^+ + \mathbf{r}_m \mathbf{e}_m^* V^+. \tag{2.6}$$

$V^+$ has $k$ nonzero off-diagonals, so

$$\mathbf{e}_m^* V^+ = (\underbrace{0,\ldots,0}_{p-1},\underbrace{*,\ldots,*}_{k+1}), \qquad k+p=m.$$

We then discard the last $k$ columns of (2.6), using the matlab notation for slices,

$$\begin{aligned} AQ_m^+[:,1:p] &= Q_m^+ H_m^+[:,1:p] + \mathbf{r}_m \mathbf{e}_m^* V^+[:,1:p] \\ &= Q_m^+[:,1:p] H_m^+[1:p,1:p] + \underbrace{h_{p+1,p}^+}_{\beta_p^+} \mathbf{q}_p + 1^+ \mathbf{e}_p^* + v_{m,p}^+ \mathbf{r}_m \mathbf{e}_p^* \\ &= Q_m^+[:,1:p] H_m^+[1:p,1:p] + \underbrace{(\mathbf{q}_{p+1}^+ h_{p+1,p}^+ + \mathbf{r}_m v_{m,p}^+)}_{=:\mathbf{r}_p^+} \mathbf{e}_p^* \end{aligned}$$

This is again an Arnoldi relation, since $Q_m^+$ is as a product of an ONB with a unitary matrix itself an ONB. It is an ONB of the Krylov space $\mathcal{K}_p(\mathbf{q}_1^+, A)$.

Now we can look at why ARPACK's strategy is sensible. The Arnoldi relation gives

$$(A-\mu_1)Q_m = Q_m(H_m - \mu_1 I) + \mathbf{r}_m \mathbf{e}_m^* = Q_m V_1 R_1 + \mathbf{r}_m \mathbf{e}_m^*.$$

Then we look at the first comlumn of this equation

$$(A-\mu_i I)\mathbf{q}_1 = Q_m V_1 \mathbf{e}_1 r_{1,1} + 0 = \mathbf{q}_1^{(1)} r_{1,1}.$$

This means by multiplying $Q_m$ with $V_1$ we have made $\mathbf{q}_1^{(1)}$ a multiple of $(A-\mu_1)\mathbf{q}_1$.

If $\mu_i$ is an eigenvalue of $A$ then $(A-\mu_1)\mathbf{q}_1$ removes the component of $\mathbf{q}_1$ in the direction of the corresponding eigenvector. In general if $\mu_i$ is close to an eigenvalue then $(A-\mu_1)\mathbf{q}_1$ has only a small component in the direction of the corresponding eigenvector to nearby eigenvalues. Then we apply the next QR steps by multiplying $V_2,\ldots,V_k$ and each time we remove the corresponding eigenvector from $\mathbf{q}_1^{(i)}$.

Therefore the Krylov space $\mathcal{K}_p(\mathbf{q}_1^+, A)$ doesn't contain the approximate eigenspace of $\mu_i$. Since we already have the Arnoldi relation of $\mathcal{K}_p(\mathbf{q}_1^+, A)$ we only need $k$ steps of Arnoldi to gain $\mathcal{K}_m(\mathbf{q}_1^+, A)$

This leads to the IRA algorithm 2.4.

**Remark.** There is the danger of recovering the same Ritz values in each iteration. Thus other strategies have been proposed but experiments show that the original strategy mostly works best. □

**Algorithm 2.4** Implicitely restarted Arnoldi (IRA)

Let the Arnoldi relation $AQ_m = Q_m H_m + \mathbf{r}_m \mathbf{e}_m^*$ be given.
**repeat**
  Determine $k$ shifts $\mu_1, \ldots, \mu_k$
  $\mathbf{v}^* := \mathbf{e}_m^*$
  **for** $i = 1, \ldots, k$ **do**
    $H_m - \mu_i I = V_i R_i$ by QR-decomposition
    $H_m := V_i^* H_m V_i$
    $Q_m := Q_m V_i$
    $\mathbf{v}^T := \mathbf{v}^* V_i$
  **end for**
  $\mathbf{r}_p := \mathbf{q}_{p+1}^+ \beta_p^+ + \mathbf{r}_m \mathbf{v}_{m,p}^+$
  $Q_p := Q_m[:, 1:p]$
  $H_p := H_m[1:p, 1:p]$
  Start with

$$AQ_p = Q_p H_p + \mathbf{r}_p \mathbf{e}_p^*$$

  $k$ additional steps of Arnoldi until

$$AQ_m = Q_m H_m + \mathbf{r}_m \mathbf{e}_m^*$$

**until** convergence

### Convergence criterion

Let $H_m \mathbf{s} = \theta \mathbf{s}$ with $\|\mathbf{s}\| = 1$. Let $\tilde{\mathbf{x}} = Q_m \mathbf{s}$, then

$$\|A\tilde{\mathbf{x}} - \theta\tilde{\mathbf{x}}\| = \|AQ_m\mathbf{s} - Q_m H_m \mathbf{s}\| = \|\mathbf{r}\||\mathbf{e}_m^T| = \beta_m |\mathbf{e}_m^T \mathbf{s}|.$$

We have

$$(A + E)\tilde{\mathbf{x}} = \theta\tilde{\mathbf{x}}, \quad E = -\mathbf{r}_m \mathbf{q}_m^T, \quad \|E\| = \|\mathbf{r}_m\| = \beta_m.$$

**Theorem 6.** *If $\lambda \in \sigma(A)$ is simple and $\theta$ is an eigenvalue of $A + E$ closest to $\lambda$, then*

$$|\lambda - \theta| \leq \frac{\|E\|}{\mathbf{y}^T \mathbf{x}} + O(\|E\|^2),$$

*where $\mathbf{y}, \mathbf{x}$ are left and right eigenvectors of $E$ corresponding to $\lambda$.*

A similar statement holds for eigenvectors, but then the distance to the next eigenvalue is also used.

In ARPACK a pair $(\theta, \tilde{\mathbf{x}})$ counts as converged if for a fixed tolerance tol there holds $\beta_m |\mathbf{e}_m^T \mathbf{s}| \leq \max(\text{eps}\|H_m\|, \text{tol}|\theta|)$. As $|\theta| \leq \|H_m\| \leq \|A\|$ there holds $\|E\| \leq \text{tol}\|A\|$ at convergence. Therefore well-condtioned eigenvalues are well approximated.

### 2.6.3 Shift invert method

To find eigenvalues close to a value $\sigma$ one can use the shift-invert method, which works by searching the largest eigenvalues of $(A-\sigma I)^{-1}$. This works because if $\lambda$ is an eigenvalue of $A$ with corresponding eigenvector $\mathbf{x}$ then

$$(A-\sigma I)^{-1}\mathbf{x} = \frac{1}{\lambda-\sigma}\mathbf{x}$$

and therefore $1/(\lambda-\sigma)$ is an eigenvalue of $(A-\sigma I)^{-1}$ with eigenvector $\mathbf{x}$. The application of $(A-\sigma I)^{-1}$ can be done by using an iterative solver for linear equation systems since the IRA doesn't need the matrix $A$ itself but rather only the matrix-vector-multiplication $A\mathbf{x}=\mathbf{y}$.

## 2.7 Multigrid preconditioning

We'll use multigrid for the preconditioner that I'll describe in section 3.10. Let's look at this method now. For a closer look at iterative solvers and preconditioning see [3].

If one looks at the error of the damped Jacobi iteration solver one observes that high frequencies of the error are reduced very quickly while low frequencies are hardly reduced. Thus we might want to construct a method that uses two principles:

- damped Jacobi for reducing high frequencies,
- another method for reducing low frequencies.

This leads to the two grid method.

### 2.7.1 Two Grid (TG)

Here we'll describe our vectors with $\mathbf{u}_n^m$, where $n$ is the dimension of the problem and $m$ is the number of smoother iterations.

After a few steps of damped Jacobi our error is already smooth. This means it may be large but not strongly varying. The idea then is to construct a good approximation of the error $\mathbf{e}_N^m$ on a coarser mesh.

This leads to the two grid method which uses these two steps:

1. **Smoothing step**: apply $m$-steps of damped Jacobi onto $A_N\mathbf{u}_N=\mathbf{b}_N$ leading to $\mathbf{u}_N^m$.
2. **Coarse grid correction (CGC)**: Use the relation

   $$\mathbf{r}=\mathbf{b}-A\mathbf{u}=A\mathbf{u}^*-A\mathbf{u}=A(\mathbf{u}^*-\mathbf{u})=A(-\mathbf{e}),$$

   where $\mathbf{u}^*$ is the correct solution and solve a related problem of size $n<N$, whose solution approximates the error $\mathbf{e}_N^m=\mathbf{u}_N^m-\mathbf{u}_N^*$. Then utilize this approximation to correct $\mathbf{u}_N^m$.

How is the coarse grid correction done in an algebric way?

We approximate the original matrix $A_N$ by a coarse version $A_n$, $N > n$. Thus we need a restriction operator $R_{n,N}$ and a prolongation operator $P_{N,n}$ between solution spaces of dimension $N$ and $n$. Then CGC is realized by

$$\mathbf{u}_N^m \longmapsto \mathbf{u}_N^{TG} = \mathbf{u}_N^m + P_{N,n}A_n^{-1}R_{n,N}(\underbrace{\mathbf{b}_N - A_N\mathbf{u}_N^m}_{=\mathbf{r}_N^m = A_N(-\mathbf{e})}).$$

We see that $P_{N,n}A_n^{-1}R_{n,N}$ shall approximate $A_N^{-1}$.

**Algorithm 2.5** CGC

Compute residual $\mathbf{r}_N^m := \mathbf{b}_N - A_N\mathbf{u}_N^m$
Restrict $\mathbf{r}_n^m := R_{n,N}\mathbf{r}_N^m$
Solve $A_n\boldsymbol{\delta}_n = \mathbf{r}_n^m$
Prolongate and add $\mathbf{u}_N^{TG} = \mathbf{u}_N^m + P_{N,n}\boldsymbol{\delta}_n$

In algorithm 2.5 the coarse grid correction $P_{N.n}\boldsymbol{\delta}_n$ is an approximation of the error $-\mathbf{e} = \mathbf{u}_N^* - \mathbf{u}_N^m$. The matrix $P_{N,n}A_n^{-1}R_{n,N}$ has reduced rank. Therefore the error amplification matrix

$$G_N^{CGC} = I_N - P_{N,n}A_n^{-1}R_{n,N}A_N$$

cannot be a contraction since $G_N^{CGC}\mathbf{v}_n = \mathbf{v}_n$ for $\mathbf{v}_n \in \ker(R_{n,N})$. The elements of $\ker(R_{n,N})$ are typically unsmooth objects thus this method is only useful in conjunction with a smoother. One can also use the smoother again afterwards for post-smoothing. The smoother may also not be damped Jacobi (eg. Gauß-Seidel).

### Galerkin CGC

In the case of Galerkin methods it is appropriate to use a weak form of CGC, involving Galerkin orthogonality. Let $\mathbf{u}_N^* \in \mathbb{R}^N$ be the exact solution of

$$A_N\mathbf{u}_N = \mathbf{b}_N \quad A_N \in \mathbb{R}^{N\times N}$$

and approximate in a smaller subspace of $\mathbb{R}^N$. The equation is equivalent to

$$\text{Find } \mathbf{u}_N \in V_N, \text{ such that } (A_N\mathbf{u}_N, \mathbf{w}_N) = (\mathbf{b}_N, \mathbf{w}_N) \quad \forall \mathbf{w}_N \in V_N.$$

Now let $V_n \subset V_N$ be a subspace of dimension $n < N$. The corresponding Galerkin approximation then is

$$\text{Find } \mathbf{u}_n \in V_n, \text{ such that } (A_N\mathbf{u}_n, \mathbf{w}_n) = (\mathbf{b}_n, \mathbf{w}_n) \quad \forall \mathbf{w}_n \in V_n. \qquad (2.7)$$

We reformulate this by defining $P_{N,n} := (\mathbf{p}_n^1, \dots, \mathbf{p}_n^n) \in \mathbb{R}^{N \times n}$ with a basis $\{\mathbf{p}_n^1, \dots, \mathbf{p}_n^n\}$ of $V_n$. If we then write $\mathbf{v}_n = P_{N,n}\mathbf{y}_n$ with $\mathbf{y}_n \in \mathbb{R}^n$ then (2.7) is equivalent to

$$\text{Find } \mathbf{y}_n \in \mathbb{R}^n, \text{ such that } (A_N P_{N,n}\mathbf{y}_n, P_{N,n}\mathbf{z}_n) = (\mathbf{b}_N, P_{N,n}\mathbf{z}_n) \quad \forall \mathbf{z}_n \in \mathbb{R}^n,$$

which is equivalent to

$$\text{Find solution } \mathbf{y}_n \in \mathbb{R}^n \text{ of } P_{N,n}^T A_N P_{N,n}\mathbf{y}_n = P_{N,n}^T \mathbf{b}_N.$$

Then we observe that

- $P_{N,n}$ is our prolongation matrix,
- $R_{n,N} := P_{N,n}^T$ is our restriction matrix.

**Remark.** The Galerkin residual orthogonality holds. We can see this by rearranging (2.7):

$$\text{Find } \mathbf{v}_n \in V_n \text{ such that } \mathbf{r}_n = \mathbf{b}_N - A_N\mathbf{v}_n \perp V_n.$$

For SPD matrices then the best approximation property follows in the energy norm $\|\cdot\|_{A_N}$:

$$\|\mathbf{v}_n - \mathbf{u}_N^*\|_{A_N} = \min_{\mathbf{w}_n \in V_n} \|\mathbf{w}_n - \mathbf{u}_N^*\|_{A_N}.$$

□

We therefore get the Galerkin CGC by using

$$A_n^{Galerkin} = P_{N,n}^T A_N P_{N,n}$$

with appropriately chosen $P_{N,n}$ for our coarse grid approximation for $A_N$. Due to the remark we have for $A_N$ SPD:

$$\|\mathbf{u}_N^{TG} - \mathbf{u}_N^*\|_{A_N} = \|P_{N,n}\boldsymbol{\delta}_n - (-\mathbf{e}_N^m)\|_{A_N} = \min_{\mathbf{w}_n \in V_n} \|\mathbf{w}_n - (-\mathbf{e}_N^m)\|_{A_N},$$

therefore

$$(P_{N,n}\boldsymbol{\delta}_n - (-\mathbf{e}_N^m)) \perp_{A_N} V_n.$$

**TG in FEM**

Here for notation we use instead of the dimension $N$, $n$ the mesh sizes $h$ (fine) and $H$ (coarse). Our nodal basis functions then are $\hat{\mathbf{v}}_{n,i}$, $i = 1, \dots, N = \dim(\hat{U}_h)$. For all $\hat{\mathbf{u}}_h \in \hat{U}_h$ denote the vector of coefficients in nodal basis as $\mathbf{u}_h$. Let $\mathbf{u}_h^m$ be after $m$ smoothing steps and $\hat{\mathbf{u}}_h^m \in \hat{U}_h$ and the error $\hat{\mathbf{e}}_h^m = \hat{\mathbf{u}}_h^m - \hat{\mathbf{u}}_h^*$ at level $h$. TG seeks

$$\hat{\mathbf{u}}_h^{TG} = \hat{\mathbf{u}}_h^m + \hat{\boldsymbol{\delta}}_H$$

with $\hat{\boldsymbol{\delta}}_H \in \hat{U}_H \subset \hat{U}_h$. We aim for $\hat{\boldsymbol{\delta}}_H$ to be the best approximation to $-\hat{\mathbf{e}}_H^m$. This means that we seek $\hat{\boldsymbol{\delta}}_H$ such that the error in the energy norm is minimized. By Galerkin orthogonality we get:

$$a((\hat{\mathbf{u}}_h^m + \hat{\boldsymbol{\delta}}_H) - \hat{\mathbf{u}}_h^*, \hat{\mathbf{w}}_H) = 0 \quad \forall \hat{\mathbf{w}}_H \in \hat{U}_H.$$

This leads to:

$$\text{Find } \hat{\boldsymbol{\delta}}_H \in \hat{U}_H, \text{ such that } a(\hat{\boldsymbol{\delta}}_H, \hat{\mathbf{w}}_H) = a(-\hat{\mathbf{e}}_h^m, \hat{\mathbf{w}}_H) \quad \forall \hat{\mathbf{w}}_H \in \hat{U}_H \quad (2.8)$$

which is equivalent to

$$\text{Find } \hat{\boldsymbol{\delta}}_H \in \hat{U}_H, \text{ such that } a(\hat{\boldsymbol{\delta}}_H, \hat{\mathbf{w}}_H) = b(\hat{\mathbf{w}}_H) - a(\hat{\mathbf{u}}_h^m, \hat{\mathbf{w}}_H) \quad \forall \hat{\mathbf{w}}_H \in \hat{U}_H$$

where the righthandside is the residual of $\hat{\mathbf{u}}_h$ in a weak sense.

Note that $(\hat{\boldsymbol{\delta}}_H + \hat{\mathbf{e}}_h^m) \perp_{a(\cdot,\cdot)} \hat{U}_H$.

Associate now $\hat{\boldsymbol{\delta}}_H$, $\hat{\mathbf{w}}_h$ with their coefficient vectors and the embedding $\hat{U}_H \subset \hat{U}_h$ with the prolongation matrix $P_{h,H}$. Then (2.8) becomes

$$\text{Find } \boldsymbol{\delta}_h, \text{ such that } (A_h P_{h,H} \boldsymbol{\delta}_H, P_{h,H} \mathbf{w}_H) = (\mathbf{r}_h^m, P_{h,H} \mathbf{w}_H) \; \forall \mathbf{w}_H \in \mathbb{R}^{\dim(\hat{U}_H)}$$

with $\mathbf{r}_h^m = \mathbf{b}_h - A_h \mathbf{u}_h^m$.

**Remark.** $P_{h,H}$ is the Galerkin prolongation matrix. For an SPD problem it makes sense to choose $R_{H,h} = P_{h,H}^T$ because

$$A_H = A_H^{Galerkin} = R_{H,h} A_h P_{h,H} = P_{h,H}^T A_h P_{h,H}$$

is also SPD.

In general $A_H^{Galerkin} = A_H$ does not hold necessarily. General multigrid methods work with $A_H$ given by the discretization from the coarse level. Also different choices for $P_{h,H}$ and $R_{H,h}$ are possible. □

### TG as a preconditioner

If we use TG as a preconditioner we simply use TG applied to $A\varepsilon = \mathbf{r}$ starting from $\varepsilon_0 = 0$ for $\mathbf{r}$ being the residual $\mathbf{r} = \mathbf{b} - A\mathbf{u}$ of any given iterate $\mathbf{u}$ to approximate the error

$$\varepsilon = \varepsilon_0 + T(\mathbf{r} - A\varepsilon_0) = T\mathbf{r} \approx -\mathbf{e}.$$

### 2.7.2 Multigrid (MG)

The coarse grid problem has the same type as the original problem, therefore we get the idea to proceed recursively:

Instead of solving on the coarse grid we use TG again, ie. we apply the smoother and CGC to get a problem on an even coarser grid. We repeat this until the problem is sufficiently small to solve directly.

Suppose we have a sequence of meshes $\mathcal{T}_\ell,\ \ell = 0, 1, \ldots$ with corresponding approximation spaces $\hat{U}_\ell$. For simplicity we assume that the spaces $\hat{U}_\ell \subset \hat{U}_{\ell+1}$ are nested for $\ell = 0, 1, \ldots$ with $N_\ell = \dim(\hat{U}_\ell)$.

The spaces $\hat{U}_\ell$ are spanned by bases. The natural embedding $\hat{U}_\ell \subset \hat{U}_{\ell+1}$ then corresponds to the prolongation $P_{\ell+1,\ell} \in \mathbb{R}^{N_{\ell+1} \times N_\ell}$, $R_{\ell,\ell+1} = P_{\ell+1,\ell}^T$. Let $A_\ell$ be the matrix coming from $a(\cdot,\cdot)$ at level $\ell$.

It may be possible to choose $P_{\ell+1,\ell}$ such that

$$A_{\ell-1} = P_{\ell,\ell-1}^T A_\ell P_{\ell,\ell-1}$$

remains valid. This facilitates the convergence analysis but the MG algorithm doesn't depend on this. It may not be valid or even desirable. Even a choice $R_{\ell-1,\ell} \neq P_{\ell,\ell-1}^T$ may be reasonable.

The algorithm that we get from this strategy is algorithm 2.7.2 with $m_{pre}$ pre-smoothing steps and $m_{post}$ post-smoothing steps. It is called V-cycle (fig. 2.4).

**Algorithm 2.6** Basic MG (V-cycle)

MG($\mathbf{u}_0, \mathbf{b}, \ell$)
**if** level $\ell$ is sufficiently small **then**
  compute $A\mathbf{u} = \mathbf{b}$ directly
**else**
  do $m = m_{pre}$ steps of smoothing with initial guess $\mathbf{u}_0$ to gain $\mathbf{u}_N^m$
  $\mathbf{r}_m = \mathbf{b} - A_\ell \mathbf{u}_N^m$
  $\boldsymbol{\delta}$ =MG($0, R_{\ell-1,\ell}\mathbf{r}_m, \ell - 1$)
  $\tilde{\mathbf{u}} = \mathbf{u}_N^m + P_{\ell,\ell-1}\boldsymbol{\delta}$
  do $m = m_{post}$ steps of smoothing with initial guess $\tilde{\mathbf{u}}$ to gain $\mathbf{u}$
**end if**
**return u**

#### W-cycle, $\mu$-cycle

One may attempt to improve the approximation by $\mu$ repeated recursive calls. This leads to the $\mu$-cycle (algorithm 2.7.2).

The case $\mu = 1$ is the V-cycle, $\mu = 2$ is called the W-cycle (fig. 2.7.2).

For the convergence analysis one gets an iteration matrix that is in recursive form. This leads to a recursive formula for convergence depending

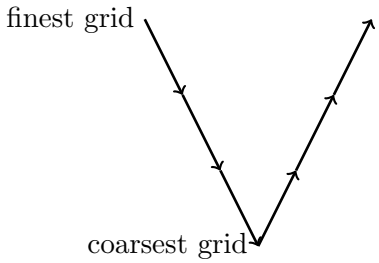


Figure 2.4: V-cycle

**Algorithm 2.7** $\mu$-cycle

MG($\mathbf{u}_0, \mathbf{b}, \ell, \mu$)
**if** level $\ell$ is sufficiently small **then**
  compute $A\mathbf{u} = \mathbf{b}$ directly
**else**
  do $m = m_{pre}$ steps of smoothing with initial guess $\mathbf{u}_0$ to gain $\mathbf{u}_N^m$
  $\mathbf{r}_m = \mathbf{b} - A_\ell \mathbf{u}_N^m$
  $\boldsymbol{\delta}^{(0)} = 0$
  **for** $\nu = 1, \ldots, \mu$ **do**
    $\boldsymbol{\delta}^{(\nu)}$ =MG($\boldsymbol{\delta}^{(\nu)-1}, R_{\ell-1,\ell}\mathbf{r}_m, \ell - 1, \mu$)
  **end for**
  $\tilde{\mathbf{u}} = \mathbf{u}_N^m + P_{\ell,\ell-1}\boldsymbol{\delta}^{(\mu)}$
  do $m = m_{post}$ steps of smoothing with initial guess $\tilde{\mathbf{u}}$ to gain $\mathbf{u}$
**end if**
**return u**

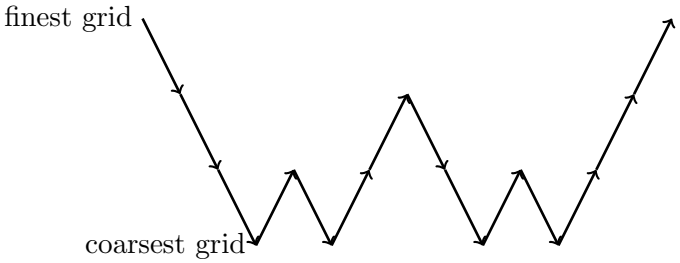


Figure 2.5: W-cycle

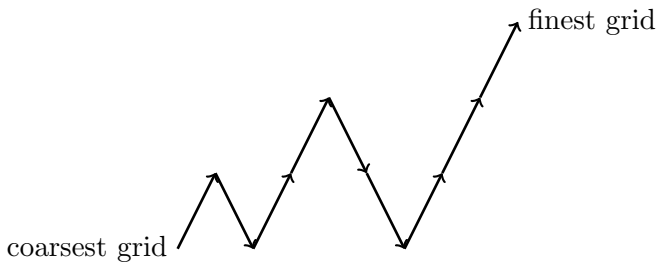


Figure 2.6: FMG

on the TG convergence rate. This makes the convergence analysis not easy (see [3] for more details).

**Remark.** Nested iteration and Full Multigrid (FMG)
If one needs a useful starting vector for the iteration one can use the following strategy:

Start at coarsest level and prolongate upwards to the next finer level where it is used as the starting vector for the next new MG cycle. Repeat this until you arrive at the finest level (fig. 2.7.2). For an in depth description see [3].

□

# Chapter 3

# The Navier-Stokes Equation

The main sources for this chapter are [4], [5] and [6].

Mathematically flow is a continuous transformation of three-dimensional space onto itself. It describes the position of a fluid particle P with starting position $\mathbf{X} = (X, Y, Z)$ (at $t = 0$) at times $t \geq 0$, then it is at position $\mathbf{x} = (x, y, z)$. The transformation is described by

$$\mathbf{x} = \phi(\mathbf{X}, t).$$

Under the assumption that different particles remain distuinguishable (and assumptions about the differentiability of $\phi$) this function is locally invertible,

$$\mathbf{X} = \Phi(\mathbf{x}, t).$$

Most of the time one is interested in certain properties of the flow at fixed points in space, such as density or velocity of the flow

$$\rho = \rho(\mathbf{x}, t), \qquad \mathbf{u} = \mathbf{u}(\mathbf{x}, t).$$

This view with fixed points $(\mathbf{x}, t)$ is called Euler's view and $(\mathbf{x}, t)$ are called (space-fixed) Euler variables. In contrast the variables $(\mathbf{X}, t)$ are called (mass-fixed) Lagrange variables. Everything describable by $(\mathbf{x}, t)$ is also describable by $(\mathbf{X}, t)$. For an arbitrary field $b$ the two different time derivatives are written as

$$\frac{\partial b}{\partial t} = \frac{\partial b(\mathbf{x}, t)}{\partial t}, \qquad \frac{\mathrm{D}b}{\mathrm{D}t} = \frac{\partial b(\mathbf{X}, t)}{\partial t}.$$

$\frac{\mathrm{D}b}{\mathrm{D}t}$ is called material derivative and it is the temporal alteration of $b$ while following a particle. $\frac{\partial b}{\partial t}$ is the alteration of $b$ at the fixed position $\mathbf{x}$. The particle velocity is defined by

$$\mathbf{u} := \frac{\mathrm{D}\mathbf{x}}{\mathrm{D}t}.$$

The particle acceleration then is given by

$$\mathbf{a} := \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t}.$$

This can be calculated

$$\mathbf{a} = \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \frac{\partial \mathbf{u}}{\partial t} + \nabla \mathbf{u} \frac{\mathrm{D}\mathbf{x}}{\mathrm{D}t} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u}.$$

In general there holds

$$\frac{\mathrm{D}b}{\mathrm{D}t} = \frac{\partial b}{\partial t} + (\mathbf{u} \cdot \nabla) b$$

for an arbitrary field $b$.

Fluid dynamics is governed by three laws of conservation:

- conservation of mass,
- conservation of momentum,
- conservation of energy.

The Navier-Stokes equation is derived from conservation of momentum but we'll also give the others here for completeness. One theorem needed to derive the equations is Reynolds transport theorem.

## 3.1 Transport theorem

We'll show the transport theorem but first we also need to see how an infinitesimal volume $\mathrm{d}V$ changes in the flow. This is given by the determinant of the Jacobian of $\phi$

$$\det J(\mathbf{X}, t) = \frac{\partial(x, y, z)}{\partial(X, Y, Z)} = \begin{vmatrix} \frac{\partial x}{\partial X} & \frac{\partial x}{\partial Y} & \frac{\partial x}{\partial Z} \\ \frac{\partial y}{\partial X} & \frac{\partial y}{\partial Y} & \frac{\partial y}{\partial Z} \\ \frac{\partial z}{\partial X} & \frac{\partial z}{\partial Y} & \frac{\partial z}{\partial Z} \end{vmatrix}.$$

We'll need the material derivative of $\det J$. With Laplace expansion of determinants one can show:

$$\frac{\mathrm{D}(\det J)}{\mathrm{D}t} = \det J \nabla \cdot \mathbf{u} = \det J \mathrm{div}(\mathbf{u}).$$

If the flow is incompressible there can't be any change in volume, thus $\frac{\mathrm{D}(\det J)}{\mathrm{D}t} = 0$. This means

$$\mathrm{div}(\mathbf{u}) = 0.$$

**Theorem 7.** *Let $b(\mathbf{x}, t)$ be an arbitrary field. Then there holds*

$$\frac{D}{Dt} \int_V b dV = \int_V \frac{\partial b}{\partial t} dV + \oint_{\partial V} b(\mathbf{u} \cdot \mathbf{n}) dO.$$

*Proof.* Let $V(t)$ be a (mass-fixed) flowing control volume and $b(\mathbf{x}, t)$ an arbitrary field. Then there holds

$$\int_V b\mathrm{d}V = \int_{V_0} b(\mathbf{X}, t)|\mathrm{det}J(\mathbf{X}, t)|\mathrm{d}V_0 \quad , V_0 = V(0).$$

The material derivative then is given by

$$\begin{aligned}\frac{\mathrm{D}}{\mathrm{D}t}\int_V b\mathrm{d}V &= \int_{V_0}\left(|\mathrm{det}J|\frac{\mathrm{D}b}{\mathrm{D}t} + b\frac{\mathrm{D}|\mathrm{det}J|}{\mathrm{D}t}\right)\mathrm{d}V_0 \\ &= \int_{V_0}\left(\frac{\mathrm{D}b}{\mathrm{D}t} + b\mathrm{div}(\mathbf{u})\right)|\mathrm{det}J|\mathrm{d}V_0 \\ &= \int_V \frac{\mathrm{D}b}{\mathrm{D}t} + b\mathrm{div}(\mathbf{u})\mathrm{d}V = \int_V \frac{\partial b}{\partial t} + \mathrm{div}(b\mathbf{u})\mathrm{d}V.\end{aligned}$$

The last equation holds due to $\mathrm{div}(b\mathbf{u}) = b\mathrm{div}(\mathbf{u}) + (\mathbf{u}\cdot\nabla)b$. With Gauß' theorem we then get the result. □

## 3.2 Conservation of mass (continuity equation)

When $\rho(\mathbf{x}, t) > 0$ denotes the density the integral continuity equation is

$$\int_V \left(\frac{\mathrm{D}\rho}{\mathrm{D}t} + \rho\mathrm{div}(\mathbf{u})\right)\mathrm{d}V = \frac{\mathrm{D}}{\mathrm{D}t}\int_V \rho\,\mathrm{d}V = \frac{\mathrm{D}m}{\mathrm{D}t} = 0.$$

For arbitrary $V$ we get the differential continuity equation

$$\frac{\mathrm{D}\rho}{\mathrm{D}t} + \rho\mathrm{div}(\mathbf{u}) = \frac{\partial\rho}{\partial t} + \mathrm{div}(\rho\mathbf{u}) = 0.$$

A nice result one can get with this and Reynolds transport theorem for an arbitrary field $\mathbf{b}$ is

$$\frac{\mathrm{D}}{\mathrm{D}t}\int_V \rho\mathbf{b}\,\mathrm{d}V = \int_V \rho\frac{\mathrm{D}\mathbf{b}}{\mathrm{D}t}\,\mathrm{d}V. \tag{3.1}$$

## 3.3 Conservation of momentum (Equation of motion)

One can alter momentum by putting stress $\sigma\mathbf{n}$ orthogonal to the surface of the control volume, by an outer force $\mathbf{g}$ per mass (like gravity, or electromagnetic force) or a force $\mathbf{K}$ on the flow by a body inside the volume (holding force). This gives us the integral equation of motion

$$\frac{\mathrm{D}}{\mathrm{D}t}\int_V \rho\mathbf{u}\,\mathrm{d}V = \oint_{\partial V}\sigma\mathbf{n}\,\mathrm{d}O + \int_V \rho\mathbf{g}\,\mathrm{d}V + \mathbf{K}.$$

We get the differential form by using $\mathbf{K} = 0$, Gauß' theorem and the result of the transport theorem (3.1) for arbitrary $V$

$$\rho \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = \sum_j \frac{\partial \sigma_{ji}}{\partial x_j} + \rho \mathbf{g}.$$

Very often the stress tensor $\sigma$ is split into the inner friction stress tensor $\sigma'$ and the part $pI \in \mathbb{R}^{3\times 3}$, which comes from the pressure $p$:

$$\sigma := \sigma' - pI.$$

Then the equation of motion reads

$$\rho \frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p + \sum_j \frac{\partial \sigma'_{ji}}{\partial x_j} + \rho \mathbf{g}.$$

## 3.4 Conservation of energy

The kinetic and inner energy are altered by surface- and volume forces, the power $L$ which comes from $\mathbf{K}$ and the heat flow $\mathbf{q}$:

$$\frac{\mathrm{D}}{\mathrm{D}t} \int_V \rho \left( \frac{\mathbf{u}\cdot\mathbf{u}}{2} + e \right) \mathrm{d}V = \oint_{\partial V} \mathbf{u}\sigma\mathbf{n}\,\mathrm{d}O + \int_V \rho \mathbf{u}\cdot\mathbf{g}\,\mathrm{d}V + L - \oint_{\partial V} \mathbf{q}\cdot\mathbf{n}\mathrm{d}O.$$

With Gauß' theorem

$$\rho \frac{\mathrm{D}}{\mathrm{D}t} \left( \frac{\mathbf{u}\cdot\mathbf{u}}{2} + e \right) = \sum_{i,j} \frac{\partial}{\partial x_j}(u_i \sigma_{ji}) + \rho \mathbf{u}\cdot\mathbf{g} - \mathrm{div}(\mathbf{q}).$$

We can get the first law of thermodynamics out of this

$$\rho \left( \frac{\mathrm{D}e}{\mathrm{D}t} - \frac{p}{\rho}\frac{\mathrm{D}\rho}{\mathrm{D}t} \right) = \sum_{i,j} \sigma'_{ij} \frac{\partial u_i}{\partial x_j} - \mathrm{div}(\mathbf{q}).$$

With Gibbs' equation $\mathrm{d}e = T\mathrm{d}s - p\mathrm{d}V$ we get different forms:

$$\begin{aligned} \rho \left( \frac{\mathrm{D}e}{\mathrm{D}t} - \frac{p}{\rho}\frac{\mathrm{D}\rho}{\mathrm{D}t} \right) &= \rho T \frac{\mathrm{D}s}{\mathrm{D}t} = \rho \frac{\mathrm{D}h}{\mathrm{D}t} - \frac{\mathrm{D}p}{\mathrm{D}t} \\ &= \rho c_p \frac{\mathrm{D}T}{\mathrm{D}t} - \beta T \frac{\mathrm{D}p}{\mathrm{D}t} = \sum_{i,j} \sigma'_{ij} \frac{\partial u_i}{\partial x_j} - \mathrm{div}(\mathbf{q}). \end{aligned}$$

The full derivation of these equations can be found in [4].

## 3.5 Derivation of the Navier-Stokes equation

We derive the Navier-Stokes equation from the equation of motion. Therefore we need to describe the stress tensor $\sigma$. For a perfect fluid (no inner friction) we get

$$\sigma = -pI, \quad \sigma' = 0. \tag{3.2}$$

In the following we'll need this definition

$$\mathbf{D} := \frac{1}{2}(\nabla \mathbf{u} + \nabla \mathbf{u}^T).$$

A fluid is called a Stokes' fluid if it fulfills Stokes' postulates:

1. $\sigma$ is a continuous function of $\mathbf{D}$ and is independent of other kinematic variables.
2. $\sigma$ is independent of the position $\mathbf{x}$ in space.
3. There is no special direction in space (isotropy).
4. If $\mathbf{D} = 0$ then $\sigma = -pI$.

If we additionally demand linear dependency $\sigma = f(D)$ we get the material equation for Newtonian fluids

$$\sigma = \sigma^T = (-p + \overline{\mu}\ \mathrm{div}(\mathbf{u}))I + 2\mu\mathbf{D}.$$

Here $\overline{\mu}(p,T)$ is the volume viscosity and $\mu(p,T)$ is the dynamic viscosity (shear viscosity). For incompressible flow $\mathrm{div}(\mathbf{u}) = 0$ there holds $\mu = \mu(T)$.

With these we get the compressible Navier-Stokes equation

$$\rho\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p + \nabla(\overline{\mu}\mathrm{div}(\mathbf{u})) + \sum_j (2\mu D_{ij}) + \rho\mathbf{g}.$$

For a perfect fluid (with (3.2)) we get the Euler equation

$$\rho\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\nabla p + \rho\mathbf{g}$$

and for an incompressible Newtonian fluid we get the incompressible Navier-Stokes equation

$$\frac{\mathrm{D}\mathbf{u}}{\mathrm{D}t} = -\frac{1}{\rho}\nabla p + \frac{\mu}{\rho}\Delta\mathbf{u} + \mathbf{g},$$

since

$$\sum_j 2\frac{\partial D_{ij}}{\partial x_j} = \sum_j \frac{\partial^2 u_i}{\partial x_j \partial x_j} + \sum_j \frac{\partial^2 u_j}{\partial x_i \partial x_j} = \Delta u_i + \frac{\partial}{\partial x_i}\mathrm{div}(\mathbf{u}) = \Delta u_i.$$

We call $\nu = \mu/\rho$ the kinematic viscosity. So the incompressible Navier-Stokes equation reads as

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u} = -\frac{1}{\rho}\nabla p + \nu\Delta\mathbf{u} + \mathbf{g}.$$

## 3.6 Dimensionless form

To get a form that is independent of the scale of the problem we use characteristic numbers and reference quantities which are relevant to the problem. We divide our quantities with their corresponding reference quantities (e.g. for the pipe the reference length $\tilde{L}$ would be the pipe diameter)

$$\mathbf{x} = \frac{\tilde{\mathbf{x}}}{\tilde{L}},\ t = \frac{\tilde{t}}{\tilde{\tau}},\ \mathbf{u} = \frac{\tilde{\mathbf{u}}}{\tilde{U}},\ \rho = \frac{\tilde{\rho}}{\tilde{\rho}_r}\ p = \frac{\tilde{p}}{\tilde{p}_r},\ \mathbf{e}_g = \frac{\tilde{\mathbf{g}}}{\tilde{g}_r},\ \mu = \frac{\tilde{\mu}}{\tilde{\mu}_r},\ \overline{\mu} = \frac{\tilde{\overline{\mu}}}{\tilde{\mu}_r}$$

(the tilde and the index $r$ are denoting the original quantities and the reference quantities respectively) and if we then use these in the incompressible Navier-Stokes equation we get the following form:

$$\mathrm{Sr}\rho\frac{\partial u_i}{\partial t} + \sum_j \rho u_j \frac{\partial u_i}{\partial x_j} = \dots$$

$$\dots = -\mathrm{Eu}\frac{\partial p}{\partial x_i} + \frac{1}{\mathrm{Re}}\left(\frac{\partial}{\partial x_i}(\overline{\mu}\mathrm{div}(\mathbf{u})) + \sum_j \frac{\partial}{\partial x_j}(2\mu D_{ij})\right) + \frac{1}{\mathrm{Fr}^2}\rho\mathbf{e}_{g_i}.$$

The characteristic numbers which now describe our problem are the Strouhal-number

$$\mathrm{Sr} := \frac{\tilde{L}}{\tilde{U}\tilde{\tau}} \qquad \text{(reduced frequency)},$$

the Euler-number

$$\mathrm{Eu} := \frac{\tilde{p}_r}{\tilde{\rho}_r\tilde{U}^2},$$

which is important whenever pressure and acceleration forces are a main factor of the problem, the Froude-number

$$\mathrm{Fr} := \frac{\tilde{U}}{\sqrt{\tilde{g}\tilde{L}}},$$

which is similar to the Mach-number, and finally the Reynolds-number

$$\mathrm{Re} := \frac{\tilde{U}\tilde{L}\tilde{\rho}_r}{\tilde{\mu}_r} = \frac{\tilde{U}_r\tilde{L}}{\tilde{\nu}_r},$$

which gives the relation between inertia forces and friction forces in the flow. We will look at the stationary ($\mathrm{Sr} = 0$) incompressible Navier-Stokes equation, where $\rho = 1$, $\mathrm{Eu} = 1$ and $\mathrm{Fr} = 1$. Now we get the form of the Navier-Stokes equation we will look at:

$$-\nabla p + \frac{1}{\mathrm{Re}}\Delta\mathbf{u} - (\mathbf{u}\cdot\nabla)\mathbf{u} = -\mathbf{e}_g,$$
$$\mathrm{div}(\mathbf{u}) = 0.$$

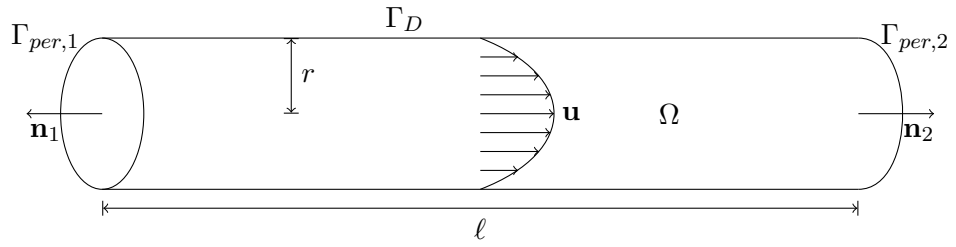


Figure 3.1: Sketch of the pipe

## 3.7 Description of our Problem

We will be looking at pipe flow on the domain $\Omega$ with periodic in- and outflow ($\Gamma_{per,1}$, $\Gamma_{per,2}$) and no-slip condition (Dirichlet boundary $\Gamma_D$) on the pipe wall. With figure 3.1, where $\Gamma_D$ describes the pipe wall we have the following problem:
Find $(\mathbf{u}, p)$, such that

$$\begin{aligned} -\frac{1}{\mathrm{Re}}\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= \mathbf{f} && \text{, on } \Omega, \\ \mathrm{div}(\mathbf{u}) &= 0 && \text{, on } \Omega, \\ \mathbf{u} &= 0 && \text{, on } \Gamma_D, \\ \mathbf{u}|_{\Gamma_{per,1}} &= \mathbf{u}|_{\Gamma_{per,2}}. \end{aligned}$$

This problem is not fully specified since the equation depends on the gradient of the pressure but not the pressure itself. Therefore the pressure is only unique up to additive constants. Thus we need another condition, we fix this by demanding

$$\int_\Omega p = 0.$$

But this still isn't enough, due to the periodic boundary conditions. This can be seen by looking at the linear subspace

$$U = \left[\begin{pmatrix}\mathbf{u}\\ p\end{pmatrix}\right] = \left[\begin{pmatrix} 0 \\ 0 \\ \frac{\mathrm{Re}}{4}(1-(x^2+y^2)) \\ -z \end{pmatrix}\right]$$

Every element $v \in U$ solves the equation for $\mathbf{f} = 0$ and radius of the pipe $r = 1$. To fix this usually a jump condition on the pressure is used

$$p|_{\Gamma_{per,1}} - p|_{\Gamma_{per,2}} = c,$$

with a prescribed constant $c$. So, we have the problem in the following form

Find $(\mathbf{u}, p)$, such that

$$\begin{aligned}
\frac{1}{\mathrm{Re}}\Delta\mathbf{u} - (\mathbf{u}\cdot\nabla)\mathbf{u} - \nabla p &= \mathbf{f} && \text{, on } \Omega,\\
\mathrm{div}(\mathbf{u}) &= 0 && \text{, on } \Omega,\\
\mathbf{u} &= 0 && \text{, on } \Gamma_D,\\
\mathbf{u}|_{\Gamma_{per,1}} &= \mathbf{u}|_{\Gamma_{per,2}}, &&\\
p|_{\Gamma_{per,1}} - p|_{\Gamma_{per,2}} &= c, &&\\
\int_\Omega p &= 0. &&
\end{aligned}$$

## 3.8 Analysis of the Navier-Stokes equation

### 3.8.1 Solving the nonlinear equation

We need to solve the nonlinear Navier-Stokes equation

$$\begin{aligned}
-\frac{1}{\mathrm{Re}}\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{u} + \nabla p &= 0,\\
\mathrm{div}(\mathbf{u}) &= 0.
\end{aligned}$$

which has the form $F(\mathbf{u}) = 0$. Thus we need an iterative method like newton's method to solve this.

$$\mathbf{u}_{n+1} = \mathbf{u}_n - D_{\mathbf{u}}F(\mathbf{u}_n)^{-1}\mathbf{u}_n$$

where $D_{\mathbf{u}}F$ is the Fréchet derivative of $F$.

Recall the definition of the Fréchet derivative.

**Definition 6.** *$F : X \times Y \to Z$ is Fréchet differentiable in $X$ at $(u_0, v_0)$ if there exists a bounded linear operator $D_uF \in L(X, Z)$ such that*

$$\lim_{h\to 0} \frac{\|F(u_0+h, v_0) - F(u_0, v_0) - (D_uF)h\|_Z}{\|h\|_X} = 0.$$

*This operator is called the Fréchet derivative.*

Now we need to calculate the Fréchet derivative of our operator $F$. The only nonlinear term is $(\mathbf{u}\cdot\nabla)\mathbf{u}$, so it suffices to look at this term.

$$\begin{aligned}
&((\mathbf{u}_0 + h)\cdot\nabla)(\mathbf{u}_0 + h) - (\mathbf{u}_0\cdot\nabla)\mathbf{u}_0 - (D_uF)h =\\
&(\mathbf{u}_0\cdot\nabla)\mathbf{u}_0 + (h\cdot\nabla)\mathbf{u}_0 + (\mathbf{u}_0\cdot\nabla)h + (h\cdot\nabla)h - (\mathbf{u}_0\cdot\nabla)\mathbf{u}_0 - (D_uF)h.
\end{aligned}$$

So, if we set $D_uF := (.\cdot\nabla)\mathbf{u}_0 + (\mathbf{u}_0\cdot\nabla)$. and show

$$\lim_{h\to 0} \frac{\|(h\cdot\nabla)h\|_{L^2}}{\|h\|_{H^1}} = 0$$

we have our linearization. With Poincaré's equation we have

$$\frac{\|(h\cdot\nabla)h\|_{L^2}}{\|h\|_{H^1}} \leq \frac{\|h\|_{L^2}\|\nabla h\|_{L^2}}{\|h\|_{H^1}} \preccurlyeq \frac{\|\nabla h\|_{L^2}^2}{\|h\|_{H^1}} \leq \frac{\|\nabla h\|_{L^2}^2}{\|\nabla h\|_{L^2}} = \|\nabla h\|_{L^2} \to 0.$$

This now gives us our linearized problem:

$$\begin{aligned} -\frac{1}{\mathrm{Re}}\Delta\mathbf{u} + (\mathbf{w}\cdot\nabla)\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{w} + \nabla p &= \mathbf{f}, \\ \mathrm{div}(\mathbf{u}) &= 0, \end{aligned}$$

with $\mathbf{w}$ an arbitrary vector. In the case of Newton's method $\mathbf{w}$ would be the solution $\mathbf{u}_n$ from the previous step. In our further analysis we will therefore look at this problem with arbitrary $\mathbf{w}$ with $\mathrm{div}(\mathbf{w}) = 0$. There are often two different main linearizations used, Newton's iteration and the Picard iteration. The Picard iteration omits the term $(\mathbf{u}\cdot\nabla)\mathbf{w}$, the operator arising from this linearization is called Oseen's operator. It converges linearly globally. Newton's method converges quadratically but only if $\mathbf{w}$ is in a small neighbourhood of u. For higher Reynolds numbers this neighbourhood becomes smaller.

### Weak Form

Now we derive the weak form of our linearized problem with our boundary condtitions

$$\begin{aligned} -\frac{1}{\mathrm{Re}}\Delta\mathbf{u} + (\mathbf{w}\cdot\nabla)\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{w} + \nabla p &= \mathbf{f}, \\ \mathrm{div}(\mathbf{u}) &= 0, \\ \mathbf{u}|_{\Gamma_D} &= 0, \\ \mathbf{u}|_{\Gamma_{per,1}} &= \mathbf{u}|_{\Gamma_{per,2}}, \\ p|_{\Gamma_{per,1}} - p|_{\Gamma_{per,2}} &= c. \end{aligned}$$

We start by multiplying the first equation with smooth periodic testfunctions $\mathbf{v}$

$$-\frac{1}{\mathrm{Re}}\int_\Omega \Delta\mathbf{u}\cdot\mathbf{v} + \int_\Omega (\mathbf{w}\cdot\nabla)\mathbf{u}\cdot\mathbf{v} + \int_\Omega (\mathbf{u}\cdot\nabla)\mathbf{w}\cdot\mathbf{v} + \int_\Omega \nabla p\mathbf{v} = \int_\Omega \mathbf{f}\cdot\mathbf{v}.$$

Now we use integration by parts on the diffusion term and the pressure term. First we'll look at the diffusion term. If we denote the boundary as $\Gamma$ we get

$$\int_\Omega \Delta\mathbf{u}\cdot\mathbf{v} = -\int_\Omega \nabla\mathbf{u}\cdot\nabla\mathbf{v} + \int_\Gamma (\nabla\mathbf{u})\mathbf{n}\cdot\mathbf{v}.$$

Take a closer look at the boundary term

$$\int_\Gamma (\nabla \mathbf{u})\mathbf{n}\cdot\mathbf{v} = \int_{\Gamma_D} (\nabla \mathbf{u})\mathbf{n}\cdot\mathbf{v} + \int_{\Gamma_{per,1}} (\nabla \mathbf{u})\mathbf{n}_1\cdot\mathbf{v} + \int_{\Gamma_{per,2}} (\nabla \mathbf{u})\mathbf{n}_2\cdot\mathbf{v},$$
$$\int_{\Gamma_D} (\nabla \mathbf{u})\mathbf{n}\cdot\mathbf{v} = 0,$$
$$\int_{\Gamma_{per,1}} (\nabla \mathbf{u})\mathbf{n}_1\cdot\mathbf{v} = -\int_{\Gamma_{per,2}} (\nabla \mathbf{u})\mathbf{n}_2\cdot\mathbf{v}.$$

The second equation holds because $\mathbf{v} = 0$ on $\Gamma_D$ and the third equation holds because $\mathbf{u},\ \mathbf{v}$ are periodic and $\mathbf{n}_2 = -\mathbf{n}_1$. Therefore

$$\int_\Gamma (\nabla \mathbf{u})\mathbf{n}\cdot\mathbf{v} = 0.$$

Now we'll look at the pressure term. Before we use integration by parts we use the linearity of the term by splitting the pressure into a periodic part and a part which satisfies the jump condition

$$p = p_{per} + p_{[c]}.$$

We'll choose a function for $p_{[c]}$. A simple choice is a linear function along the domain (in our case the pipe). For our specific problem we can choose

$$p_{[c]} = \frac{c}{\ell} z$$

if $\ell$ is the length of the pipe and the point of origin is in the middle of the pipe so that

$$\int_\Omega p_{[c]} = 0$$

is satisfied. We then can split the pressure term

$$\int_\Omega \nabla p\cdot\mathbf{v} = \int_\Omega \nabla p_{per}\cdot\mathbf{v} + \int_\Omega \nabla p_{[c]}\cdot\mathbf{v} = \int_\Omega \nabla p_{per}\cdot\mathbf{v} + \int_\Omega \begin{pmatrix} 0 \\ 0 \\ \frac{c}{\ell} \end{pmatrix}\cdot\mathbf{v}.$$

We put the jump term on the righthandside of the equation. Now we use integration by parts on the periodic term.

$$\int_\Omega \nabla p_{per}\cdot\mathbf{v} = \int_\Omega p_{per}\mathrm{div}(\mathbf{v}) - \int_\Gamma p_{per}\mathbf{v}\cdot\mathbf{n},$$
$$\int_{\Gamma_D} p_{per}\mathbf{v}\cdot\mathbf{n} = 0 \qquad \text{since } \mathbf{v}|_{\Gamma_D} = 0,$$
$$\int_{\Gamma_{per,1}} p_{per}\cdot\mathbf{n}_1 = -\int_{\Gamma_{per,2}} p_{per}\cdot\mathbf{n}_2$$

since $p_{per}$, $\mathbf{v}$ are periodic and $\mathbf{n}_2 = -\mathbf{n}_1$. Similar to before the second equation holds because $\mathbf{v} = 0$ on $\Gamma_D$ and the third equation holds because $p_{per}$, $\mathbf{v}$ are periodic and $\mathbf{n}_2 = -\mathbf{n}_1$. Therefore

$$\int_\Gamma p_{per} \cdot \mathbf{v} = 0.$$

We then get the weak formulation of our first equation

$$\frac{1}{\mathrm{Re}} \int_\Omega \nabla \mathbf{u} \cdot \nabla \mathbf{v} + \int_\Omega (\mathbf{w} \cdot \nabla)\mathbf{u} \cdot \mathbf{v} + \int_\Omega (\mathbf{u} \cdot \nabla)\mathbf{w} \cdot \mathbf{v} - \int_\Omega p_{per} \mathrm{div}(\mathbf{v}) = \dots$$
$$\dots = \int_\Omega \mathbf{f} \cdot \mathbf{v} - \int_\Omega \nabla p_{[c]} \cdot \mathbf{v}.$$

We get our second equation by simply multiplying with a smooth testfunction $q$

$$\int_\Omega \mathrm{div}(\mathbf{u}) q = 0.$$

So we now have our weak form for smooth functions but we can use bigger spaces. For our velocity functions we can use $H^1$ but we have to modify it so that it fulfills the Dirichlet and the periodic boundary conditions

$$(H^1_{0,per})^3 = \{\mathbf{v} \in (H^1)^3 : \mathbf{v}|_{\Gamma_D} = 0,\ \mathbf{v}|_{\Gamma_{per,1}} = \mathbf{v}|_{\Gamma_{per,2}}\}.$$

Since there is no weak derivative of the pressure $p_{per}$ in the equation it seems that we could use $L^2 2$ but since we need to impose a periodic boundary condition on $p_{per}$ and on $L^2$-functions there cannot be prescribed any boundary conditions we need to use a periodic $H^1$ space instead

$$L^2_0 \cap H^1_{per} = \{q \in H^1 : \int q = 0,\ q|_{\Gamma_{per,1}} = q|_{\Gamma_{per,2}}\}.$$

After solving we then get the correct solution by adding

$$p = p_{per} + p_{[c]}.$$

### 3.8.2 Analysis of the linearized problem

Our linearized problem

$$\begin{aligned}
-\frac{1}{\mathrm{Re}}\Delta \mathbf{u} + (\mathbf{u} \cdot \nabla)\mathbf{w} + (\mathbf{w} \cdot \nabla)\mathbf{u} + \nabla p &= f && \text{, on } \Omega, \\
\mathrm{div}(\mathbf{u}) &= 0 && \text{, on } \Omega, \\
\mathbf{u} &= \mathbf{u}_D && \text{, on } \Gamma_D, \\
\mathbf{u}|_{\Gamma_{per,1}} &= \mathbf{u}|_{\Gamma_{per,2}}, \\
p|_{\Gamma_{per,1}} - p|_{\Gamma_{per,2}} &= c,
\end{aligned}$$

with $\operatorname{div}(\mathbf{w}) = 0$ transforms into the weak form

$$\begin{aligned}\frac{1}{\mathrm{Re}}\int \nabla\mathbf{u}\nabla\mathbf{v} + \int (\mathbf{w}\cdot\nabla)\mathbf{u}\cdot\mathbf{v} + (\mathbf{u}\cdot\nabla)\mathbf{w}\cdot\mathbf{v} - \int \operatorname{div}(\mathbf{v})p = \ldots \\ \ldots = \int f\mathbf{v} - \int_\Omega \nabla p_{[c]}\cdot\mathbf{v} \qquad \forall \mathbf{v}\in V, \\ \int \operatorname{div}(\mathbf{u})q = 0 \qquad \forall q \in Q,\end{aligned}$$

with $V = (H^1_{0,per})^3$ and $Q = L^2_0 \cap H^1_{per}$. We have the form

$$\begin{aligned}a(\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) &= \ell(\mathbf{v}),\\ b(\mathbf{u},q) &= 0,\end{aligned}$$

with

$$\begin{aligned}a(\mathbf{u},\mathbf{v}) &= -\frac{1}{\mathrm{Re}}\Delta\mathbf{u} + (\mathbf{u}\cdot\nabla)\mathbf{w} + (\mathbf{w}\cdot\nabla)\mathbf{u},\\ b(\mathbf{v},p) &= \int \operatorname{div}(v)p.\end{aligned}$$

For our problem to be uniquely solvable we have to show continuity of $a$ and $b$, kernel-coercivity of $a$

$$\begin{aligned}a(\mathbf{u},\mathbf{u}) \geq \|\mathbf{u}\|^2_{H^1} \qquad \forall \mathbf{u}\in V_0 &:= \{\mathbf{v}\in V : b(\mathbf{v},p) = 0, \forall p\in Q\}\\ &= \{\mathbf{v}\in V : \operatorname{div}(\mathbf{v}) = 0\}\end{aligned}$$

and the LBB condition for $b$

$$\sup_{\substack{\mathbf{u}\in V,\\ \mathbf{u}\neq 0}} \frac{b(\mathbf{u},q)}{\|\mathbf{u}\|_V} \geq \beta_1\|q\|_Q \qquad \forall q\in Q.$$

Let's start with the continuity and coercivity of $a$. $a$ can be split into three terms:

$$\begin{aligned}a(\mathbf{u},\mathbf{u}) &= a_1(\mathbf{u},\mathbf{v}) + c(\mathbf{w};\mathbf{u},\mathbf{v}) + d(\mathbf{w};\mathbf{u},\mathbf{v}),\\ a_1(\mathbf{u},\mathbf{v}) &= \int \nabla\mathbf{u}\nabla\mathbf{v},\\ c(\mathbf{w};\mathbf{u},\mathbf{v}) &= \int (\mathbf{w}\cdot\nabla)\mathbf{u}\cdot\mathbf{v},\\ d(\mathbf{w};\mathbf{u},\mathbf{v}) &= \int (\mathbf{u}\cdot\nabla)\mathbf{w}\cdot\mathbf{v}.\end{aligned}$$

For Picard iteration the bilinearform $d$ is omitted.

**Theorem 8.** *The bilinearform $a$ is continuous.*

*Proof.* [6] p284 □

**Theorem 9.** *Let* $\mathbf{w} \in V$ *with* $\operatorname{div}(\mathbf{w}) = 0$. *Then it follows:*

1. *For the Picard iteration the bilinearform a is coercive*
$$a(\mathbf{u}, \mathbf{u}) \geq \alpha \|\mathbf{u}\|_{H^1}^2 \qquad \forall \mathbf{u} \in V.$$
2. *For the Newton iteration the bilinearform a is coercive in a neighbourhood of* $\mathbf{w}$
$$a(\mathbf{u}, \mathbf{u}) \geq \alpha \|\mathbf{u}\|_{H^1}^2 \qquad \forall \mathbf{u} \in U_\varepsilon(\mathbf{w}).$$

*Proof.* The coercivity of $a_1$ is easily seen as it is the coercivity of the Laplace operator. The bilinearform $c(\mathbf{w}; \mathbf{u}, \mathbf{v})$ is often said to be skew-self-adjoint $c(\mathbf{w}; \mathbf{u}, \mathbf{v}) = -c(\mathbf{w}; \mathbf{v}, \mathbf{u})$. By using Green's formula (,where $H(\operatorname{div}, \Omega)$ is the space of all $L^2(\Omega)$ functions, with a bounded weak divergence,)

$$(v, \nabla\phi) + (\operatorname{div}(v), \phi) = \langle v \cdot n, \phi \rangle_\Gamma \qquad \forall v \in H(\operatorname{div}, \Omega),\ \forall \phi \in H^1(\Omega), \tag{3.3}$$

we will show this being the case if there are no Neumann conditions.

$$\begin{aligned} c(\mathbf{w}; \mathbf{u}, \mathbf{u}) &= \sum_{i,j=1}^{N} \int_\Omega w_j \frac{\partial u_i}{\partial x_j} u_i \mathrm{d}x = \frac{1}{2} \sum_{i,j=1}^{N} \int_\Omega w_j \frac{\partial (u_i^2)}{\partial x_j} \mathrm{d}x = \\ &= \frac{1}{2} \sum_{i=1}^{N} \int_\Omega \mathbf{w} \cdot \nabla (u_i^2) \mathrm{d}x \\ &= -\frac{1}{2} \sum_{i=1}^{N} \left( \int_\Omega \operatorname{div}(\mathbf{w}) u_i^2 \mathrm{d}x - \int_\Gamma \mathbf{w} \cdot \mathbf{n} (u_i^2) \mathrm{d}s \right) \\ &= 0. \end{aligned} \tag{3.4}$$

In the last step we used $\operatorname{div}(\mathbf{w}) = 0$ and that $\mathbf{w}$ satisfies the Dirichlet boundary condition and the periodic boundary condition.

Finally for the Newton iteration we use continuity of $d$: Let $\mathbf{w} = \mathbf{u} + \mathbf{r}$ with $\|\mathbf{r}\|_{H^1} < \varepsilon$, then

$$d(\mathbf{w}; \mathbf{u}, \mathbf{u}) = d(\mathbf{u}; \mathbf{u}, \mathbf{u}) + d(\mathbf{r}; \mathbf{u}, \mathbf{u})$$

Since $d(\mathbf{u}; \mathbf{u}, \mathbf{u}) = c(\mathbf{u}; \mathbf{u}, \mathbf{u}) = 0$ ($\geq 0$ for outflow boundary) and with continuity of $d$

$$|d(\mathbf{r}; \mathbf{u}, \mathbf{u})| \leq \varepsilon C \|\mathbf{u}\|_{H^1}^2$$

it follows that

$$d(\mathbf{w}; \mathbf{u}, \mathbf{u}) \geq -\varepsilon C \|\mathbf{u}\|_{H^1}^2.$$

Using this we can set $\varepsilon$ small enough that

$$a(\mathbf{u}, \mathbf{u}) - C\varepsilon \|\mathbf{u}\|_{H^1}^2 \geq \alpha \|\mathbf{u}\|_{H^1}^2$$

Therefore $a$ is coercive in a small neighbourhood of $\mathbf{w}$. □

**Remark.** The last part in the proof shows why the Newton iteration only converges if the initial guess is in a small neighbourhood of the solution. □

**Remark.** If we have Neumann conditions we get in (3.4) instead of 0 the term

$$h(\mathbf{u}) = \frac{1}{2}\int_{\Gamma_N} \mathbf{u}^2\mathbf{w}\cdot\mathbf{n}.$$

This is nonnegative on an outflow boundary ($\mathbf{w}\cdot\mathbf{n} \geq 0$). □

Now we'll look at continuity and the LBB-condition of $b$.

**Theorem 10.** *The bilinearform $b$ is continuous in $V$ and $Q$.*

*Proof.*

$$\int_\Omega \operatorname{div}(\mathbf{v})q \leq \|\operatorname{div}(\mathbf{v})\|_{L^2}\|q\|_{L^2} \leq \|\nabla\mathbf{v}\|_{L^2}\|q\|_{L^2} \leq \|\mathbf{v}\|_{H^1}\|q\|_{L^2}.$$

□

For the LBB-condition we start by showing the LBB-condition for the spaces $\mathbf{v} \in (H_0^1)^3$ and $p \in L_0^2$. We need two results first

**Lemma 1.** *Let $f$ be in the dual space $H^{-1}(\Omega)^N$ of $H^{-1}(\Omega)^N$. If it satisfies*

$$\langle f, \mathbf{v}\rangle = 0 \quad \forall \mathbf{v} \in V_0 = \{\mathbf{v} \in (H_0^1)^N : \operatorname{div}(\mathbf{v}) = 0\}$$

*then there exists $p \in L^2$, such that*

$$f = \nabla p$$

*When $\Omega$ is connected, $p$ is unique up to additive constants.*

*Proof.* $\nabla \in \mathcal{L}(L^2, H^{-1}(\Omega)^N)$ is the dual operator of -div $\in \mathcal{L}(H_0^1(\Omega), L^2(\Omega))$ due to Green's formula (3.3). Since $\operatorname{ran}(\nabla)$ is a closed subspace of $H^{-1}(\Omega)^N$, there holds

$$\operatorname{ran}(\nabla) = \overline{\operatorname{ran}(\nabla)} = {}^0(\operatorname{ran}(\nabla)^0) = {}^0(\ker(\operatorname{div})) = {}^0V_0.$$

Where ${}^0V_0$ is:

$${}^0V_0 = \{\mathbf{y} \in H^{-1}(\Omega)^N : \langle \mathbf{y}, \mathbf{v}\rangle = 0 \ \forall \mathbf{v} \in V_0\}$$

Therefore the lemma is proven. □

We split $H_0^1(\Omega)^N$ orthogonally by

$$H_0^1(\Omega)^N = V_0 \oplus V^\perp.$$

**Corollary 1.** *Let $\Omega$ be connected. Then*

1. *the operator $\nabla$ is an isomorphism of $L_0^2(\Omega)$ onto ${}^0V_0$,*
2. *the operator* div *is an isomorphism of $V^\perp$ onto $L_0^2(\Omega)$.*

*Proof.* 1. $\nabla \in \mathcal{L}(L_0^2(\Omega), {}^0V_0)$. Lemma 1 shows that this is a bijection. Since ${}^0V_0$ and $L_0^2$ are Banach spaces, it follows that $\nabla$ is an isomorphism.

2. Since div is the dual operator of $-\nabla$, div is an isomorphism from $({}^0V_0)'$ onto $(L_0^2)'$. Now we prove that $({}^0V_0)$ can be identified with $(V^\perp)'$.

   Take any $g \in (V^\perp)'$, we extend $\mathbf{g}$ to $H_0^1$ by setting

   $$\langle \tilde{\mathbf{g}}, \mathbf{v} \rangle = \langle \mathbf{g}, \mathbf{v}^\perp \rangle \quad \forall \mathbf{v} \in H_0^1$$

   where $\mathbf{v}^\perp$ is the orthogonal projection of $\mathbf{v}$ onto $V^\perp$. Then $\tilde{\mathbf{g}} \in {}^0V_0$ and the linear mapping $\mathbf{g} \mapsto \tilde{\mathbf{g}}$ maps isometrically $(V^\perp)'$ onto ${}^0V_0$.

□

With these results we can show the LBB-condition for $(H_0^1(\Omega))^N$ and $L_0^2(\Omega)$.

**Lemma 2.** *The bilinearform $b$ satisfies the LBB-condition for the spaces $(H_0^1(\Omega))^N$ and $L_0^2(\Omega)$:*

$$\sup_{\mathbf{v} \in (H_0^1(\Omega))^N} \frac{b(\mathbf{v}, q)}{\|\mathbf{v}\|_{H^1}} = \sup_{\mathbf{v} \in (H_0^1(\Omega))^N} \frac{(\operatorname{div}(\mathbf{v}), q)}{\|\mathbf{v}\|_{H^1}} \geq \beta \|q\|_{L^2} \quad \forall q \in L_0^2(\Omega).$$

*Proof.* Let $q \in L_0^2(\Omega)$. Due to corollary 1 above there exists a unique function $\mathbf{v} \in V^\perp$, such that

$$\operatorname{div}(\mathbf{v}) = q, \quad \|\mathbf{v}\|_{H^1} \leq C \|q\|_{L^2}.$$

Thus

$$\frac{(\operatorname{div}(\mathbf{v}), q)}{\|\mathbf{v}\|_{H^1}} = \frac{\|q\|_{L^2}^2}{\|\mathbf{v}\|_{H^1}} \geq \frac{1}{C} \|q\|_{L^2}.$$

With $\beta = 1/C$ the statement follows. □

(see [6].) Now the LBB-condition for the spaces $V$ and $Q$ we use follows from this lemma 2.

**Corollary 2.** *The bilinearform $b$ satisfies the LBB-condition for $V$ and $Q$:*

$$\sup_{\mathbf{v} \in V} \frac{b(\mathbf{v}, q)}{\|\mathbf{v}\|_{H^1}} \geq \beta \|q\|_{L^2} \quad \forall q \in Q.$$

*Proof.* Since $(H_0^1)^3 \subset V$ and $Q \subset L_0^2$ there holds

$$\sup_{\mathbf{v}\in V} \frac{b(\mathbf{v},q)}{\|\mathbf{v}\|_{H^1}} \geq \sup_{\mathbf{v}\in (H_0^1)^3} \frac{b(\mathbf{v},q)}{\|\mathbf{v}\|_{H^1}} \geq \beta \|q\|_{L^2} \quad \forall q \in L_0^2.$$

It follows

$$\sup_{\mathbf{v}\in V} \frac{b(\mathbf{v},q)}{\|\mathbf{v}\|_{H^1}} \geq \beta \|q\|_{L^2} \quad \forall q \in Q.$$

□

### 3.8.3 Numerics of the linearized problem

The first finite element space we'd think about for the Navier-Stokes equation would be piecewise linear continuous basis functions in the velocity space and piecewise constant discontinuous basis functions in the pressure space.

This doesn't work, the inf-sup condition cannot be shown. An easy way to see this is to think about the number of degrees of freedom. There are simply more triangles/tetrahedrons than vertices.

The simplest finite element space that works in 2D is the following: With barycentric coordinates $\lambda_1, \lambda_2, \lambda_3$, which are linear functions that are 1 on the corresponding vertex and 0 on the other vertices, and the outer normal vectors $\mathbf{n}_1, \mathbf{n}_2, \mathbf{n}_3$ on the edges of a triangle define

$$\begin{aligned} \mathbf{p}_1 := \mathbf{n}_1\lambda_2\lambda_3,\ \mathbf{p}_2 := \mathbf{n}_2\lambda_3\lambda_1,\ \mathbf{p}_3 := \mathbf{n}_3\lambda_1\lambda_2,& \\ \mathcal{P}_1(\kappa) &:= P_1^2 + \operatorname{span}\{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3\}, \\ W_h &:= \{\mathbf{w} \in C^0(\overline{\Omega})^2 : \mathbf{w}|_\kappa \in \mathcal{P}_1(\kappa),\ \forall \kappa \in \mathcal{T}_h\}, \\ \overline{X_h} &:= W_h \cap H_0^1(\Omega)^2, \\ Q_h &:= \{q \in L^2(\Omega) : q|_\kappa \in P_0,\ \forall \kappa \in \mathcal{T}_h\}, \\ \overline{M_h} &:= Q_h \cap L_0^2(\Omega). \end{aligned} \tag{3.5}$$

I will not show its inf-sup condition here, for a proof see [6]. Most of the time (if we have subsonic velocities or our setup with the periodic pressure space) we won't see discontinuous pressures. Due to this we would like to have a discretisation that has continuous pressures. This leads us to one of the more famous elements for the Navier-Stokes equation, the Hood-Taylor element:

$$\begin{aligned} X_h &:= \{\mathbf{v} \in C^0(\overline{\Omega})^2 : \mathbf{v}|_\kappa \in P_2^2,\ \forall \kappa \in \mathcal{T}_h,\ \mathbf{v}|_\Gamma = 0\}, \\ Q_h &:= \{q \in C^0(\overline{\Omega}) : q|_\kappa \in P_1,\ \kappa \in \mathcal{T}_h\}, \\ M_h &:= Q_h \cap L_0^2(\Omega). \end{aligned}$$

I will here show the proof for the twodimensional case for $p = 1$. Sadly its not that easy to generalize the proof to tetrahedrons, although according to [5]

it is possible to use a similar approach and it can be done but they didn't give a reference to this proof and I wasn't able to find a proof anywhere else. They give, however, a proof for bricks, which I will not do here because I used tetrahedrons for the calculations.

First we need to look at an important theorem which lets us prove the global inf-sup condition if we are able to prove a local inf-sup condition.

**Theorem 11.** *Let $W_h \subset H^1(\Omega)^N$, $Q_h \subset L^2(\Omega)$ with $\mathbb{R} \subset Q_h$ be two finite dimensional spaces and*

$$X_h := W_h \cap H_0^1(\Omega)^N = \{\mathbf{v} \in W_h : \mathbf{v}_h|_{\Gamma_D} = 0\},$$
$$M_h := Q_h \cap L_0^2(\Omega) = \{q_h \in Q_h : \int_\Omega q_h dx = 0\}.$$

*$\Omega$ shall be able to be partitioned into a finite number of disjoint Lipschitz-continuous open subsets $\Omega_r$ with boundary $\Gamma_r$:*

$$\overline{\Omega} = \bigcup_{r=1}^{R} \overline{\Omega_r}.$$

*For $1 \leq r \leq R$ define*

$$X_h(\Omega_r) := \{\mathbf{v} \in X_h : \mathbf{v} = 0 \text{ in } \Omega \setminus \Omega_r\},$$
$$Q_h(\Omega_r) := \{q|_{\Omega_r} : q \in Q_h\},$$
$$M_h(\Omega_r) := Q_h(\Omega_r) \cap L_0^2(\Omega_r),$$
$$\overline{M_h} := \{q \in L_0^2(\Omega) : q|_{\Omega_r} \text{ is constant}, 1 \leq r \leq R\}.$$

*If the pair $(X_h, M_h)$ satisfies: There exists a constant $\lambda^* > 0$, independent of $h$ and $r$, such that*

$$\sup_{\mathbf{v}_h \in X_h(\Omega_r)} \frac{\int_{\Omega_r} \operatorname{div}(\mathbf{v}_h) q_h dx}{|\mathbf{v_h}_{1,\Omega_r}} \geq \lambda^* \|q_h\|_{0,\Omega_r} \quad \forall q_h \in M_h(\Omega_r), 1 \leq r \leq R$$

*and there exists a subspace $\overline{X_h}$ of $X_h$, such that $(\overline{X_h}, \overline{M_h})$ satisfies the inf-sup condition with a constant $\overline{\beta}$ independent of $h$, then $(X_h, M_h)$ also satisfies the inf-sup condition with a constant $\beta^*$ independent of $h$.*

*Proof.* The proof can be found in [6]. □

So we need to show a local inf-sup condition for Hood-Taylor because the spaces $(\overline{X_h}, \overline{M_h})$ defined in (3.5) already fulfill the corresponding requirements in the theorem.

At first we need to choose a partition that works for our proof. We do this by grouping all elements which share a common vertex (vertex patches). We need to make the following assumption about our triangulation $\mathcal{T}_h$:

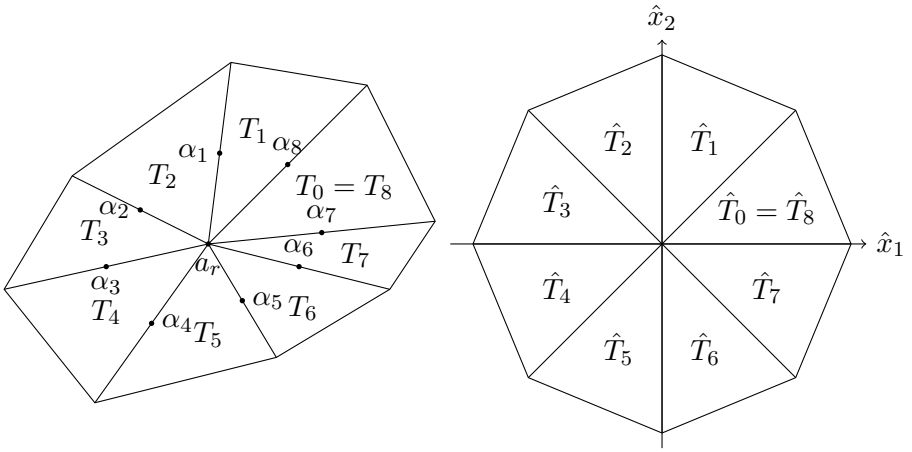


Figure 3.2: One macroelement ($\Omega_r$) with its reference element ($J = 8$).

**Assumption 1.** *$\mathcal{T}_h$ has a set of interior nodes $\{a_r\}_r = 1^R$ such that $\{\Omega_r\}_{r=1}^R$ with*

$$\Omega_r = \left( \bigcup_{a_r \in T} T \right)^\circ$$

*is a partition of $\overline{\Omega}$.*

It is not difficult to construct a triangulation which fulfills this assumption. One simply starts with a coarse mesh and adds interior nodes until the assumption is satisfied.

**Theorem 12.** *Let $\mathcal{T}_h$ be a regular triangulation of $\overline{\Omega}$ that satisfies the assumption. Then there exists a constant $\lambda^* > 0$, independent of $h$ and $r$ such that*

$$\sup_{\mathbf{v} \in X_h} \frac{\int_{\Omega_r} \operatorname{div}(\mathbf{v}) q}{|\mathbf{v}|_{1,\Omega_r}} \geq \lambda^* \|q\|_{0,\Omega_r} \quad \forall q \in M_h(\Omega_r).$$

*Proof.* Let $J$ be the number of elements in $\Omega_r$. We number them with $0 \leq i \leq J$ such that $T_i$ is adjacent to $T_{i-1}$ and $T_0 = T_J$.

The edge shared by $T_i$ and $T_{i+1}$ we name $e_i$ and the midpoint of the edge $\alpha_i$. The vertex common to all elements $T_i$ in $\Omega_r$ is called $a_r$ (see figure 3.2).

We associate with this the reference set:

$$\hat{\Omega} = \bigcup_{i=1}^{J} \hat{T}_i$$

by the continuous piecewise affine function $F_r$ efined by:

$$F_r(\hat{T}_i) = T_i, \; F_r(\hat{\mathbf{x}}) = B_i \hat{\mathbf{x}} + \mathbf{b}_i \quad \forall \hat{\mathbf{x}} \in \hat{T}_i.$$

Since the triangulation is regular the number of elements in $\Omega_r$ $J$ is bounded from above by a fixed number $I$ independent of $r$. This means that there ere at most $I$ different reference sets $\hat{\Omega}_r$. Thus all geometrical constans related to $\hat{\Omega}$ can be bounded independently of $h$ and $r$.

Let $q_h$ be an arbitrary function in $Q_h(\Omega_r)$ and $\mathbf{v}_h$ a function in $X_h(\Omega_r)$ which fulfills $\mathbf{v}_h(a_r) = 0$. Since $\mathbf{v}_h$ vanishes on $\partial\Omega_r$ and $q_h \in H^1(\Omega_r)$ there holds

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}_h) q_h \mathrm{d}x = -\sum_{i=1}^{J} \int_{\kappa_i} \mathbf{v}_h \cdot \nabla q_h \mathrm{d}x.$$

Each component $v$ of $\mathbf{v}_h$ is a quadratic polynomial on $T_i$ that vanishes at the vertices of $T_i$. Therefore the quadrature formula

$$\int_{T_i} v \mathrm{d}x = \mathrm{meas}(T_i) \frac{1}{3} [v(\alpha_i) + v(\alpha_{i-1})]$$

is valid. Due to $\nabla q_h$ being constant on $T_i$ ($\nabla q_h|_{T_i} = \mathbf{g}_i$) there holds

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}) q_h \mathrm{d}x = -\frac{1}{3} \sum_{i=1}^{J} \mathrm{meas}(T_i) [\mathbf{v}(\alpha_i) + \mathbf{v}(\alpha_{i-1})] \cdot \mathbf{g}_i.$$

Since $\partial q_h / \partial \tau$ is continuous on the edges it seems sensible to choose

$$\mathbf{v}_h(\alpha_i) = -(\mathbf{g}_i \cdot \mathbf{t}_i)\mathbf{t}_i = -(\mathbf{g}_{i+1} \cdot \mathbf{t}_i)\mathbf{t}_i,$$

where $\mathbf{t}_i$ is the tangent vector to $e_i$ with length $\|e_i\|$ pointing outside of $\Omega_r$. This leads to

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}) q_h \mathrm{d}x = \frac{1}{3} \sum_{i=1}^{J} \mathrm{meas}(T_i) [(\mathbf{g}_i \cdot \mathbf{t}_i)^2 + (\mathbf{g}_i \cdot \mathbf{t}_{i-1})^2].$$

$\mathbf{g} \cdot \mathbf{t}$ is preserved by affine transformations. Thus we can write

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}) q_h \mathrm{d}x = \frac{1}{3} \sum_{i=1}^{J} \mathrm{meas}(T_i) [(\hat{\mathbf{g}}_i \cdot \hat{\mathbf{t}}_i)^2 + (\hat{\mathbf{g}}_i \cdot \hat{\mathbf{t}}_{i-1})^2].$$

Since each set $\{\hat{\mathbf{t}}_{i-1}, \hat{\mathbf{t}}_i\}$ is a basis on the reference space the function $\hat{\mathbf{g}} \to [(\hat{\mathbf{g}}_i \cdot \hat{\mathbf{t}}_i)^2 + (\hat{\mathbf{g}}_i \cdot \hat{\mathbf{t}}_{i-1})^2]^{1/2}$ is equivalent to the Euclidean norm. With

$$|\hat{q}|^2_{1,\hat{T}_i} = \mathrm{meas}(T_i) \|\hat{\mathbf{g}}_i\|^2$$

this implies that there exists a constant $\hat{C}_1 > 0$ such that

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}) q_h \mathrm{d}x \geq \hat{C}_1 \sum_{i=1}^{J} \mathrm{meas}(T_i) |\hat{q}|^2_{1,\hat{T}_i}. \tag{3.6}$$

The definition of $\mathbf{v}_h$ gives us

$$\begin{aligned}\|\mathbf{v}_h\|^2_{0,T_i} &\leq \hat{C}_2 \mathrm{meas}(T_i)[\|\mathbf{v}_h(\alpha_{i-1}\|^2 + \|\mathbf{v}_h(\alpha_i)\|^2] \\ &\leq \hat{C}_3 \mathrm{meas}(T_i)[h_{T_i}|\hat{q}|_{1,\hat{T}_i}]^2\end{aligned}$$

Due to (2.3) and since $\hat{\mathbf{v}}$ belongs to $P_2$ on $\hat{T}$ there holds

$$\begin{aligned}|\mathbf{v}_h|_{1,T_i} &\leq C_1\|B_i^{-1}\||\det(B_i)|^{\frac{1}{2}}|\hat{\mathbf{v}}_h|_{1,\hat{T}_i} \\ &\leq C_2\|B_i^{-1}\||\det(B_i)|^{\frac{1}{2}}|\hat{v}_h|_{0,\hat{T}} \\ &\leq C_3\|B_i^{-1}\|\|\mathbf{v}_h\|_{0,T} \\ &\leq C_4 \mathrm{meas}(T_i)[\sigma_{T_i}|\hat{q}|_{1,\hat{T}_i}].\end{aligned} \tag{3.7}$$

With (3.6), (3.7) and the regularity of $\mathcal{T}_h$ we get

$$\int_{\Omega_r} \mathrm{div}(\mathbf{v}_h) q_h \mathrm{d}x \geq \hat{C}_6 \frac{1}{\sigma} |\mathbf{v}_h|_{1,\Omega_r} \left[\sum_{i=1}^{J} \mathrm{meas}(T_i)|\hat{q}|^2_{1,\hat{T}_i}\right]^{\frac{1}{2}}.$$

Now it remains to show that on a regular triangulation there holds

$$\left[\sum_{i=1}^{J} \mathrm{meas}(T_i)|\hat{q}|^2_{1,\hat{T}_i}\right]^{\frac{1}{2}} \geq \hat{C}\|q\|_{0,\Omega_r} \quad \forall q \in H^1(\Omega_r) \cap L^2_0(\Omega_r).$$

$F_r$ maps $H^1(\Omega_r)$ into $H^1(\hat{\Omega})$ but it does not preserve the zero mean value. We handle this by replacing $q \in H^1(\Omega_r) \cap L^2_0(\Omega_r)$ by $\overline{q}$ where

$$\hat{\overline{q}} = \hat{q} - \frac{1}{\mathrm{meas}(\hat{\Omega})} \int_{\hat{\Omega}} \hat{q} \mathrm{d}\hat{x}.$$

Then $q$ and $\overline{q}$ differ by a constant and we have

$$\|q\|_{0,\Omega_r} = \inf_{c \in \mathbb{R}} \|q + c\|_{0,\Omega_r} \leq \|\overline{q}\|_{0,\Omega_r}.$$

This then gives

$$\begin{aligned}\|\overline{q}\|^2_{0,\Omega_r} &= \sum_{i=1}^{J} \mathrm{meas}(T_i)\|\hat{\overline{q}}\|^2_{0,\hat{T}_i} \\ &\leq \hat{C}_7 \sup_{1\leq i\leq J}(h^2_{T_i})|\hat{\overline{q}}|^2_{1,\hat{\Omega}} \quad \text{since } \hat{\overline{q}} \in H^1(\hat{\Omega}) \cap L^2_0(\hat{\Omega}) \\ &\leq \hat{C}_8 \left[\frac{\sup_{1\leq i\leq J}(h^2_{T_i})}{\inf_{1\leq i\leq J}(\rho^2_{T_i})}\right] \sum_{i=1}^{J} \mathrm{meas}(T_i)|\hat{q}|_{1,\hat{T}_i}.\end{aligned}$$

Thus we have

$$\left[\sum_{i=1}^{J} \text{meas}(T_i)|\hat{q}|_{1,\hat{T}_i}\right]^{\frac{1}{2}} \geq \frac{\hat{C}_9}{\sigma_r}\|q\|_{0,\Omega_r}$$

with

$$\sigma_r := \frac{\sup_{1\leq i\leq J}(h_{T_i})}{\inf_{1\leq i\leq J}(\rho_{T_i})}.$$

Due to the regularity of $\mathcal{T}_h$ we have

$$\sigma_r \leq \hat{C}\sigma.$$

□

## 3.9 Convection-diffusion equation

We'll need the convection-diffusion equation as a subproblem in our preconditioner described in the following section. Therefore we'll now look at it more closely. The convection-diffusion equation is

$$-\varepsilon\Delta u + \mathbf{w}\cdot\nabla u = f \tag{3.8}$$

with a constant $\varepsilon > 0$ and boundary conditions

$$\begin{aligned} u &= u_D \quad \text{on } \Gamma_D, \\ \frac{\partial u}{\partial \mathbf{n}} &= g_N \quad \text{on } \Gamma_N, \\ u|_{\Gamma_{per,1}} &= u|_{\Gamma_{per,2}}. \end{aligned}$$

$\mathbf{w}$ is called wind. In the rest of the section we drop the periodic boundary condition, since it doesn't change the arguments presented here.

Most of the time diffusion is less significant than convection, which means

$$\varepsilon << |\mathbf{w}|.$$

This is going to be the case for our problem on most of the volume $\Omega$.

In the following we'll assume that $\varepsilon/(|\mathbf{w}|L)$ is small, where $L$ is a characteristic length scale of the problem. Then the solution $u$ of (3.8) is close to the solution $\hat{u}$ of

$$\mathbf{w}\cdot\nabla\hat{u} = f.$$

We analyse this equation by using characteristics or streamlines, this means curves $\mathbf{c}(s)$ which fulfill $\frac{\mathrm{d}\mathbf{c}}{\mathrm{d}s} = \mathbf{w}$ which leads to the ODE

$$\frac{\mathrm{d}}{\mathrm{d}s}(\hat{u}(\mathbf{c}(s))) = f(\mathbf{c}(s)). \tag{3.9}$$

If $f = 0$ the solution $\hat{u}$ is constant along streamlines.

Let now $\hat{c}(s_0)$ be on an inflow boundary and set $\hat{u}(\mathbf{c}(s_0))$ as an initial condition for (3.9). If the streamline $\mathbf{c}$ intersects the border $\partial\Omega$ at another point $\mathbf{c}(s_1)$ the solution of (3.9) $\hat{u}(\mathbf{c}(s_1))$ might not be the same as the boundary condition of the original problem $u(\mathbf{c}(s_1))$.

Therefore it often happens that the solution $u$ has a steep gradient in some portion $\Omega$. In the most part of $\Omega$ $u$ and $\hat{u}$ are very similar but along streamlines ending in an outflow boundary where $u$ and $\hat{u}$ differ a steep gradient is needed to satisfy the boundary condition. In such a case the problem is called to be singularly perturbed and the solution has an exponential boundary layer.

Also diffusion may lead to a steep gradient transverse to streamlines where $u$ is smoother than $\hat{u}$. A possibility for this are discontinuous boundary conditions on the inflow. The discontinuity then propagates into $\Omega$ along streamlines. $u$ is continuous but rapidly varying across an internal layer.

Such layers (internal and at boundary) lead to problems when trying to construct approximations for cases when the convection is dominant.

It is useful to have a measurement on the relative contributions of convection and diffusion. This leads to the Peclet number, which we get by putting equation (3.8) in its dimensionless form.

Let $L$ be a characteristic length, $\mathbf{w} = W\mathbf{w}_*$ with a positive $W$ (which would be the reference velocity of the wind) and $|\mathbf{w}_*|$ normalized and for each point $\mathbf{x} \in \Omega$ let $\hat{\boldsymbol{\xi}} = \hat{\mathbf{x}}/L$ be its scaled version. With $u_*(\hat{\boldsymbol{\xi}}) = u(L\hat{\boldsymbol{\xi}})$ we get

$$-\Delta u_* + \left(\frac{WL}{\varepsilon}\right)\mathbf{w}_* \cdot \nabla u_* = \frac{L^2}{\varepsilon} f.$$

The Peclet number then is

$$\mathcal{P} := \frac{WL}{\varepsilon}.$$

If $\mathcal{P} \leq 1$ the equation is therefore diffusion dominated and relatively benign, on the other hand if $\mathcal{P} >> 1$ then the problem becomes much more difficult. Due to the steep gradients we get a problem for the standard Galerkin method because the mesh needs to be fine enough to resolve these gradients. This leads to large problems or in the case of our preconditioner is not sensible since the coarser grids of the multigrid method may be too coarse to resolve this.

For a more in-depth analysis of this problem see [5].

Therefore we need a strategy to get useful solutions even on coarse grids. This leads to the streamline diffusion (SD) method.

### Streamline diffusion (SD) method

Here I'll only describe the SD method, for the theory of errors see [5]. The derivation of the SD discretisation uses a Petrov-Galerkin formulation.

Idea of Petrov-Galerkin: Use different spaces for solution and testfunctions, i.e. for an arbitrary operator $\mathcal{L}$ and $\mathcal{L}u = f$ the weak problem is (with suitable product)

$$\text{Find } u \in U \text{ such that } (\mathcal{L}u, v) = (f, v) \quad \forall v \in V.$$

In the case of the SD discretisation $\mathcal{L}$ is the convection-diffusion operator, the trialspace $U$ is the space of trialfunction that fulfill the Dirichlet condition $X_D^h$ and the testspace $V$ is spanned by the functions $v_h + \delta \mathbf{w} \cdot \nabla v_h$ where $v_h \in X_0^h$ (fulfills 0-boundary condition) and $\delta > 0$ is a constant parameter. This gets us to the lefthandside of our equation

$$\begin{aligned}(\mathcal{L}u, v) = \varepsilon \int_\Omega \nabla u_h \cdot \nabla v_h - \varepsilon \int_{\partial\Omega_N} v_h \frac{\partial u_h}{\partial \mathbf{n}} + \int_\Omega (\mathbf{w} \cdot \nabla u_h) v_h \\ + \delta \int_\Omega (\mathbf{w} \cdot \nabla u_h)(\mathbf{w} \cdot \nabla v_h) - \delta\varepsilon \int_\Omega (\Delta u_h)(\mathbf{w} \cdot \nabla v_h).\end{aligned}$$

If $g_N = 0$ this simplifies since the boundary integral vanishes.

One problem here is that $\Delta u_h$ is not defined since $u_h$ doesn't need to have a second derivative. But if we restrict to individual elements and these restrictions have a second derivative, which is true most of the time since most of the time the functions are polynomials on elements, we can construct a legitimate method with an element-wise sum

$$-\delta\varepsilon \sum_k \int_{T_k} (\Delta u_h)(\mathbf{w} \cdot \nabla v_h).$$

This leads to the equation

$$\text{Find } u_h \in X_D^h \text{ such that } a_{SD}(u_h, v_h) = \ell_{SD}(v_h) \quad \forall v_h \in X_0^h,$$

where

$$\begin{aligned}a_{SD}(u, v) &:= \varepsilon \int_\Omega \nabla u \cdot \nabla v + \int_\Omega (\mathbf{w} \cdot \nabla u) v + \delta \int_\Omega (\mathbf{w} \cdot \nabla u)(\mathbf{w} \cdot \nabla v) \\ &\quad - \delta\varepsilon \sum_k \int_{T_k} (\Delta u)(\mathbf{w} \cdot \nabla v), \\ \ell_{SD}(v) &:= \int_\Omega f v + \delta \int_\Omega f \mathbf{w} \cdot \nabla v.\end{aligned}$$

This methodology leads to a different norm, the SD norm

$$\|v\|_{SD} := (\varepsilon \|\nabla v\|^2 + \delta \|\mathbf{w} \cdot \nabla v\|^2)^{\frac{1}{2}}.$$

For linear elements the coercivity bound then is

$$a_{SD}(u_h, v_h) \geq \|v_h\|_{SD}^2$$

which is stronger than the Galerkin bound because it does not degrade in the limit $\varepsilon \to 0$.

We still need to choose the parameter $\delta$ or a locally defined parameter $\delta_k^*$. With the element Peclet number $\mathcal{P}_h^k := |\mathbf{w}|h_k/(2\varepsilon)$ it is useful to choose

$$\delta_k^* = \begin{cases} \frac{h_k}{2|\mathbf{w}_k|}\left(1 - \frac{1}{\mathcal{P}_h^k}\right) & , if\ \mathcal{P}_h^k > 1 \\ 0 & , if\ \mathcal{P}_h^k \leq 1 \end{cases}$$

where $h_k$ is a measure of element length in direction of wind and $|\mathbf{w}_k|$ is the $\ell^2$-norm of wind at the element centroid.

The choice of this parameter is better motivated with a more special derivation of the SD discretisation which can be found in [5], but it is noteworthy that in the limit $\mathcal{P}_h^k \to \infty$ we see $\delta_k^* \to h_k/(2\mathbf{w})$ which is optimal in the convective limit.

Otherwise when the mesh resolves all layers of the problem the SD discretisation becomes the standard Galerkin discretisation.

## 3.10 Preconditioning

To solve the linear equation system arising from our FEM-discretisation efficiently we want to use a preconditioner. The the discretisation of the Navier-Stokes problem has the form

$$\begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \begin{pmatrix} f \\ g \end{pmatrix},$$

where the blocks $F$ and $B$ represent the discrete versions of the bilinearforms $a(\mathbf{u}, \mathbf{u})$ and $b(\mathbf{u}, q)$ respectively

Since our matrix is not symmetric it is not that useful to use a block diagonal matrix as our preconditioner instead we use a block triangular matrix

$$M = \begin{pmatrix} M_F & B^T \\ 0 & -M_S \end{pmatrix}.$$

Where $M_F$ shall be an approximation of $F$ and $M_S$ shall be a suitably chosen Schur complement. We apply this preconditioner by solving

$$\begin{pmatrix} M_F & B^T \\ 0 & -M_S \end{pmatrix} \begin{pmatrix} w \\ s \end{pmatrix} = \begin{pmatrix} v \\ q \end{pmatrix},$$

with given $v$ and $q$, which is relatively inexpensive when solving

$$M_S s = -q, \quad M_F w = v - B^T s.$$

To construct a sensible choice for $M_S$ we assume $M_F = F$ and look at the generalized eigenvalue problem

$$\begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix} = \lambda \begin{pmatrix} F & B^T \\ 0 & -M_S \end{pmatrix} \begin{pmatrix} u \\ p \end{pmatrix}.$$

There are two possibilities: $\lambda = 1$ or $\lambda \neq 1$.

In the case of $\lambda \neq 1$ we get

$$Fu + B^T p = 0 \text{ or } u = -F^{-1}B^T p$$

from the first block equation. Then it follows with the second equation that

$$BF^{-1}B^T p = \lambda M_S p. \tag{3.10}$$

This means the eigenvalues of $M$ consist of unity (case $\lambda = 1$) and the eigenvalues of (3.10). If we define

$$K := \begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix}$$

and use block LU-decomposition we get

$$\begin{pmatrix} F & B^T \\ B & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ BF^{-1} & I \end{pmatrix} \begin{pmatrix} F & B^T \\ 0 & -BF^{-1}B^T \end{pmatrix}.$$

If we'd set the Schur complement

$$M_S = S := BF^{-1}B^T$$

the eigenvalues of the preconditioned matrix $L = KU^{-1}$ would be 1. This would be ideal but is not feasible. Therefore we want to approximate $S$ with a different $M_S$.

We assumed $M_F = F$ but the preconditioner requires $M_F^{-1}$. This means we also need a strategy for $M_F$.

### 3.10.1 Approximating $F$

Here I will outline some of the issues arising from using multigrid for $F$ in a concise manner. For an in depth view on this topic see [5].

We observe that for the Picard iteration the matrix $F$ arising from the Oseen operator is a block-diagonal matrix with a discrete convection diffusion operator in each nonzero block. For the Newton iteration this is not the case but for the preconditioner we'll use an approximation of $F$ instead. We use $\hat{F}$ which shall be the matrix arising from an Oseen operator.

Therefore it is natural to employ multigrid strategies used for convection diffusion equations.

#### Multigrid for convection diffusion

There are two major points which are important for a successful use of multigrid.

1. Choice of discretisation: Due to the fact that standard Galerkin discretisation on coarse grids is ineffective it is not sensible to use it for multigrid, instead use streamline diffusion (SD) discretisation. Even if the original grid is fine enough the coarse grids of multigrid can be too coarse. In experiments this leads to divergent behavior (see [5]).

2. Smoothing operators: Due to the direction of flow simply using a Gauß-Seidel smoother is not sufficient as performance varies widely for different ordering of variables. If the order of variables follows the direction of flow already one step of a Gauß-Seidel smoother gives very good results. But if the order of variables is in the opposite direction the performance is really bad. Each step only smoothes the error on the first cells of the flow.
   So if the direction of flow is known one can choose the ordering accordingly. For more complex flows the ordering could be chosen adaptively but this would be computationally expensive. Instead we can use ordering along each axis in both directions in each step of Gauß-Seidel. For 2-dimensional problems this leads to the 4 directional Gauß-Seidel and for 3-dimensional problems this leads to the 6 directional Gauß-Seidel. This strategy is less effective than following the direction of flow but much better than ordering in the opposite direction.

### 3.10.2 Approximating the Schur complement $S$

With the pressure mass matrix $Q$, setting $M_S = \mathrm{Re}Q$ is a very good choice as long as the Reynolds number is small (order of $\mathrm{Re} \leq 40$). To get to our approximation we start with the Oseen operator

$$L = -\frac{1}{\mathrm{Re}}\Delta + \mathbf{w}_h \cdot \nabla.$$

Suppose we have an analogous operator on the pressure space

$$L_p = \left(-\frac{1}{\mathrm{Re}}\Delta + \mathbf{w}_h \cdot \nabla\right)_p.$$

We now say $p \in H^1$ although $p \in L^2$ and look at the commutator of the convection-diffusion operator with the divergence operator

$$\mathcal{E} = \mathrm{div}\left(-\frac{1}{\mathrm{Re}}\Delta + \mathbf{w}_h \cdot \nabla\right) - \left(-\frac{1}{\mathrm{Re}}\Delta + \mathbf{w}_h \cdot \nabla\right)_p \mathrm{div}.$$

We would like to have $\mathcal{E}$ small in some sense. $\mathcal{E}$ would be 0 if $\mathbf{w}_h$ was constant and $\Omega$ an unbounded domain.

**Pressure convection-diffusion preconditioner**

Let $\mathbf{Q}$ be the velocity mass matrix

$$\mathbf{Q} = (q_{ij}), \quad q_{ij} = \int_\Omega \mathbf{\Phi}_i \mathbf{\Phi}_j,$$

where the $\mathbf{\Phi}_i$ are the velocity basis functions of the finite element method. The matrix representation for the discrete divergence operator is $Q^{-1}B$ and for the discrete gradient operator is $\mathbf{Q}^{-1}B^T$. A similar derivation gets us the matrix representation of the convection-diffusion operator. With these we now get a discrete version of our commutator

$$\mathcal{E}_h = (Q^{-1}B)(\mathbf{Q}^{-1}F) - (Q^{-1}F_p)(Q^{-1}B), \tag{3.11}$$

$$F_p = (f_{p;ij}),\ f_{p;ij} = \tfrac{1}{\mathrm{Re}} \textstyle\int_\Omega \nabla\psi_j \nabla\psi_i + \int_\Omega (\mathbf{w}_h \cdot \nabla\psi_j)\psi_i. \tag{3.12}$$

We isolate the Schur complement in (3.11) by multiplying $QF_p^{-1}Q$ from the left and $F^{-1}B^T$ from the right

$$BF^{-1}B^T \approx QF_p^{-1}B\mathbf{Q}^{-1}B^T.$$

There are some details needed for implementation. The matrix $B\mathbf{Q}^{-1}B^T$ is problematic because it is dense. Therefore we want to use another matrix which is sparse. For enclosed flow we have spectral equivalence to the Laplace operator on the pressure space

$$A_p = (a_{p;ij}),\ a_{p;ij} = \int_\Omega \nabla\psi_j \cdot \nabla\psi_i.$$

For nonenclosed flow this is not useful. Instead we can use $\mathbf{T} = \mathrm{diag}(\mathbf{Q})$ and replace $B\mathbf{Q}^{-1}B^T$ with $B\mathbf{T}^{-1}B^T$. Now we have our approximated Schur complement

$$M_S := QF_p^{-1}\tilde{A}_p$$

whereby $\tilde{A}_p$ means $A_p$ or $B\mathbf{T}^{-1}B^T$ depending on the problem. Our preconditioner requires

$$M_S^{-1} = \tilde{A}_p^{-1}F_pQ^{-1}$$

which can be implemented easily.

This derivation can be altered for stabilized methods and in case of discontinuous pressure also altered with jumps (see [5]).

# Chapter 4

# Bifurcation theory

The main sources for this background are [8], [10] and [14].

## 4.1 Bifurcation theory

We want to look at the stability of our equation. Therefore we look at the steady states.

### 4.1.1 Bifurcation theory for ODEs

First we introduce bifurcation theory for ODEs

$$u' = f(u, p).$$

We need the definition of topological equivalence.

**Definition 7.** *A dynamical system* $(A)$ *is topological equivalent to another one* $(B)$ *if there is a homeomorphism mapping trajectories of* $(A)$ *to* $(B)$ *preserving the direction of time.*

Now we can define what a bifurcation is.

**Definition 8.** *The appearence of a topologically nonequivalent phase portrait under parameter variation is called bifurcation.*

This definition can be used to look at normal forms of bifurcations, e.g.

$$u' = u(p - u) \quad u : \mathbb{R} \to \mathbb{R},\ p \in \mathbb{R}$$

is the normal form of transcritical bifurcations (see fig. 4.1).

We want to look at stability, therefore we need to define stable and unstable manifolds.

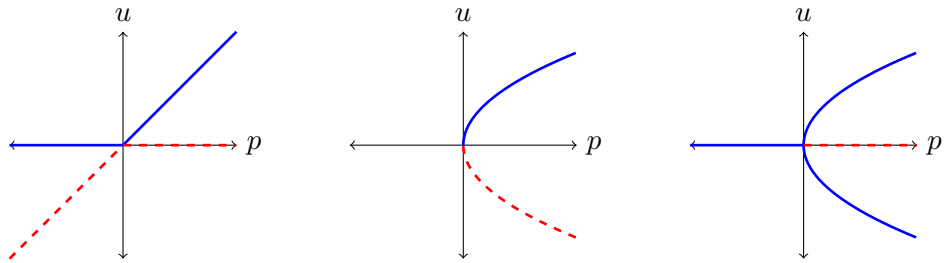


Figure 4.1: Phase portraits of the normal forms for the transcritical bifurcation ($u' = u(p-u)$), the fold bifurcation ($u' = p - u^2$) and the pitchfork bifurcation ($u' = u(p-u^2)$). The blue lines are stable steady states and the red dashed lines are unstable steady states.

**Definition 9.** *Let $\phi(u_0, t)$ be the flow of an ODE with steady state $u^*$. We then define stable and unstable manifolds as*

$$W^s(u^*) := \{v \in \mathbb{R}^d : \phi(v,t) \to u^*,\ t \to \infty\},$$
$$W^u(u^*) := \{v \in \mathbb{R}^d : \phi(v,t) \to u^*,\ t \to -\infty\}.$$

We can get informations about stability from the eigenvalues of our linearized problem. We need to define hyperbolic steady states.

**Definition 10.** *A steady state $u^*$ is called hyperbolic steady state if the derivative $(D_u f)(u^*) \in \mathbb{R}^{d\times d}$ has only eigenvalues $\lambda_i$ with $Re(\lambda_i) \neq 0$.*

**Theorem 13** (Stable-Unstable Manifold Theorem)**.** *Suppose an ODE has a hyperbolic steady state $u^*$ and $(D_u f)(u^*)$ has $k$ real-part negative and $d-k$ real-part positive eigenvalues with corresponding eigenspaces $E^s(u^*)$ and $E^u(u^*)$ for the linearized system. Then there exists a neighbourhood $\mathcal{U}$ of $u^*$ with local stable and unstable manifolds $W^s_{loc}(u^*)$ and $W^u_{loc}(u^*)$*

$$W^s_{loc}(u^*) = \{v \in \mathcal{U} : \phi(v,t) \to u^*,\ t \to \infty,\ \phi(v,t) \in \mathcal{U}\ \forall t \geq 0\},$$
$$W^u_{loc}(u^*) = \{v \in \mathcal{U} : \phi(v,t) \to u^*,\ t \to -\infty,\ \phi(v,t) \in \mathcal{U}\ \forall t \leq 0\}.$$

*These manifolds are tangent to $E^s(u^*)$ and $E^u(u^*)$ at $u^*$ and are as smooth as $f$.*

*Proof.* The proof can be found in [10]. □

It follows that if $p - k = 0$ the steady state is stable.

### 4.1.2 Lyapunov-Schmidt theorem

We want to generalize this theory to PDEs. We do this by focusing on local bifurcations. We look at problems of the form:

$$F(u,v) = 0, \quad F : X \times Y \to Z, \quad (u,v) \in X \times Y \tag{4.1}$$

where $X$, $Y$, $Z$ are Banach spaces.

We use a generalization of the implicit function theorem to gain information.

**Theorem 14** (Implicit Function Theorem)**.** *Suppose* $(u_0, v_0)$ *satisfies* (4.1)*. The Fréchet derivative* $D_uF(u_0, v_0)$ *is bijective,* $F \in C(X \times Y, Z)$ *and* $D_uF \in C(X \times Y, \mathcal{L}(X, Z))$*. Then there exists a neighbourhood* $\mathcal{U} \times \mathcal{V}$ *of* $(u_0, v_0)$ *and a continuous map* $f : \mathcal{V} \to \mathcal{U}$ *such that* $f(v_0) = u_0$ *and*

$$F(f(v), v) = 0 \quad \forall v \in \mathcal{V}.$$

*Moreover all solutions in* $\mathcal{U} \times \mathcal{V}$ *are of this form.*

This theorem gives us local uniqueness of a branch of solutions and fails if two branches cross. Therefore interesting points for bifurcations are where the implicit function theorem fails.

We want to reduce our infinite dimensional problem to a finite dimensional problem. If we assume that $\mathcal{V} \subset \mathbb{R}$ (or $\mathbb{R}^d$) it remains to reduce the $u$-component. This is done using the Lyapunov-Schmidt method.

We need to know what a Fredholm operator is.

**Definition 11.** *Let* $F : \mathcal{U} \subset X \to Z$ *and* $F$ *shall be Fréchet differentiable. Let* $u_0 \in \mathcal{U} \subset X$*.* $F$ *is a nonlinear Fredholm operator if*

- $dim(ker((D_uF)(u_0))) < \infty$,
- $codim(\mathcal{R}((D_uF)(u_0))) < \infty$,

*where* $codim(S) := dim(Z - S)$*. We define the Fredholm index as*

$$\textit{Fredholm index} := dim(ker((D_uF)(u_0))) - codim(\mathcal{R}((D_uF)(u_0))).$$

Fredholm operators have 'relatively small nullspace' and miss a 'relatively small part' of the range.

**Theorem 15.** *Assume*

$$F(u_0, v_0) = 0, F,\ D_uF \in C,\ F(\cdot, v_0) : X \to Z$$

*is a Fredholm operator. Then there exists a neighbourhood* $\mathcal{U} \times \mathcal{V}$ *of* $(u_0, v_0)$ *such that* $F(u, v) = 0$ *is equivalent* $\mathcal{U} \times \mathcal{V}$ *to the finite dimensional problem*

$$\Phi(\tilde{u}, v) = 0, \quad (\tilde{u}, v) \in \tilde{\mathcal{U}}_1 \times \mathcal{V}_1 \subset \mathcal{N} \times \mathcal{V}$$

*and* $\Phi$ *is continuous with* $\Phi(\tilde{u}_1, v_1) = 0$ *for* $(\tilde{u}_1, v_1) \in \tilde{\mathcal{U}}_1 \times \mathcal{V}_1$*.*

*Proof.* The proof can be found in [10]. □

The function $\Phi$ is called bifurcation function.

### 4.1.3 Crandall-Rabinowitz

Now we search for local bifurcations of certain PDEs. We use the following form:

$$F(u,p)=0,\ F:X\times\mathbb{R}\to Z,\ F(0,p)=0.$$

The last condition gives that we always have a trivial solution branch. We assume:

- $\dim(\ker((D_uF)(0,p)))=1=\operatorname{codim}(\mathcal{R}((D_uF)(0,p)))$
- $F\in C^3$ in an open neighbourhood of the trivial branch.

Without loss of generality we can assume that the critical point is at $p=0$.

**Theorem 16** (Crandall-Rabinowitz Theorem)**.** *The assumptions from this subsection shall hold and*

$$ker((D_uF)(0,0))=span(e_0),\qquad (D^2_{up}F)(0,0)e_0\notin\mathcal{R}((D_uF)(0,0))$$

*for $e_0\in X$ and $\|e_0\|=1$. Then there is a nontrivial branch of solutions described by a $C^1$-curve through $(u,p)=(0,0)$*

$$\{(u(s),p(s)):s\in(-s_0,s_0),\ (u(0),p(0))=(0,0)\}$$

*which satisfies $F(u(s),p(s))=0$ locally and all solutions in a neighbourhood of $(0,0)$ are either on the trivial branch or on the nontrivial curve.*

The situation in this theorem is also called bifurcation from a simple eigenvalue as $\dim(\mathcal{N})=1$. And it gives us the existence of a nontrivial solution branch.

**Corollary 3.** *The tangent vector to the nontrivial solution curve at $(u,p)=(0,0)$ is given by $(e_0,\dot{p}(0))$.*

This is directly used in some numerical methods for branch switching (see [8] Method IV)

### 4.1.4 Stability

Now we still need to look at the stability of the evolution problem

$$\partial_t u=F(u,p),\ u:[0,\infty)\to X,\ u=u(t)\in X,\ p\in\mathbb{R}.$$

**Definition 12.** *A solution branch $(u^*(p),p)$ for $F(u,p)=0$ is called (linearly) stable at $p^*$ if*
*$\sigma((D_uF)(u^*,p^*))$ is properly contained in the left half of the complex plane.*

## 4.2 Arclength continuation

For the calculation of solution branches we use the algorithm described in [14] called (pseudo)arclength continuation. The aim is to calculate a solution branch of

$$G(u, \lambda) = 0$$

for $G : X \times \mathbb{R} \to X$ a $C^1$-function and $X$ being a Banach space. The branch $z(s)$ shall be parametrized by $s$ and we look at the extended system

$$H(u, \lambda) = \begin{pmatrix} G(u, \lambda) \\ p(u, \lambda, s) \end{pmatrix} = 0 \qquad \in X \times \mathbb{R},$$

where $p$ is used to make $s$ an approximation of the arclength.

We assume now that $X$ is a Hilbert space with inner product $\langle \cdot, \cdot \rangle$. The standard choice then is: If we have $(u_0, \lambda_0) = (u(s_0), \lambda(s_0))$ with $s_0$ given and we know the tangent $\tau_0 := (\dot{u}_0, \dot{\lambda}_0) := \frac{\mathrm{d}}{\mathrm{d}s}(u(s), \lambda(s))$ we use

$$p(u, \lambda, s) := \xi \langle \dot{u}_0, u(s) - u_0 \rangle + (1 - \xi)\dot{\lambda}_0(\lambda(s) - \lambda_0) - (s - s_0),$$

where $0 < \xi < 1$ is a weight and $\tau_0$ shall be normalized in

$$\|\tau\|_\xi := \sqrt{\langle \tau, \tau \rangle_\xi}, \qquad \left\langle \begin{pmatrix} u \\ \lambda \end{pmatrix}, \begin{pmatrix} v \\ \mu \end{pmatrix} \right\rangle := \xi \langle u, v \rangle + (1 - \xi)\lambda\mu.$$

With a fixed $s$ and $\|\tau_0\|_\xi = 1$ then $p(u, \lambda, s) = 0$ defines a hyperplane perpendicular (in $\langle \cdot, \cdot \rangle_\xi$) to $\tau_0$ at distance $ds := s - s_0$ from $(u_0, \lambda_0)$.

We now use a predictor $(u^1, \lambda^1) = (u_0, \lambda_0) + ds\tau_0$ for a solution on that hyperplane. Using Newton's method we correct this estimate

$$\begin{pmatrix} u^{\ell+1} \\ \lambda^{\ell+1} \end{pmatrix} = \begin{pmatrix} u^\ell \\ \lambda^\ell \end{pmatrix} - A(u^\ell, \lambda^\ell)^{-1} H(u^\ell, \lambda^\ell), \qquad A = \begin{pmatrix} G_u & G_\lambda \\ \xi \dot{u}_0^T & (1 - \xi)\dot{\lambda}_0 \end{pmatrix}. \quad (4.2)$$

Since $\partial_s p = -1$ on a smooth solution arc it follows that

$$A(u(s), \lambda(s)) \begin{pmatrix} \dot{u}(s) \\ \dot{\lambda}(s) \end{pmatrix} = - \begin{pmatrix} 0 \\ -\partial_s p \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix}.$$

Therefore after convergence of (4.2) we can get our new tangent $\tau_1$ at the newly found point $(u_1, \lambda_1)$ with the Jacobian $A^1$ from

$$A^1 \tau_1 = \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \quad (4.3)$$

with normalization $\|\tau_1\|_\xi \leq 1$. This process conserves the orientation of $\tau_0$, i.e. $\langle \tau_0, \tau_1 \rangle = 1$.

One can choose to use a chord method where $A = A(u^1, \lambda^1)$ remains fixed during the iteration.

$$\begin{pmatrix} u^{\ell+1} \\ \lambda^{\ell+1} \end{pmatrix} = \begin{pmatrix} u^\ell \\ \lambda^\ell \end{pmatrix} - A(u^1, \lambda^1)^{-1} H(u^\ell, \lambda^\ell).$$

This avoids the costly assembly of $G_u$ at the price of an increased count of iterations during the Newton correction.

### 4.2.1 Role of weight $\xi$

There are two major roles of $\xi$.

If $\mathbf{G}(\mathbf{u}, \lambda) = 0$ comes from a discretization of a PDE $G(u, \lambda) = 0$ over a domain with $n_p$ spatial points then $\mathbf{u}$ is an element of $\mathbb{R}^p$ with large $p$. Here let's use $p = Nn_p$, with $N$ being the dimension of the image of $u$. In this case we want to choose $\xi$ such that $\xi\|\mathbf{u}\|^2_{\mathbb{R}^p}$ is an approximation of $(1/\text{meas}(\Omega))\|u\|^2_{L^2(\Omega)}$.

If $u \equiv 1$ corresponds to $\mathbf{u}_j,\ j = 1, \ldots, n_p$ then by assuming $u_i,\ i = 1, \ldots, N$ we get an estimate

$$\frac{1}{\text{meas}(\Omega)}\|u\|^2_{(L^2)^N} = N \stackrel{!}{=} \xi\|\mathbf{u}\|^2_{\mathbb{R}^p} = \xi N n_p \quad \Rightarrow \quad \xi = \frac{1}{n_p}.$$

Therefore we get a basic formula for $\xi$.

But if we choose different $\xi$ we get different continuations. Small $\xi$ favors changes in $u$ and large $\xi$ favors changes in $\lambda$. $\xi$ can therefore also be seen as a parameter to tune continuation and may be changed during runs. This

**Algorithm 4.1** Basic continuation algorithm

Given $\xi,\ (u_0, \lambda_0, \tau_0),\ ds$.
Predictor: $(u^1, \lambda^1) := (u_0, \lambda_0) + ds\tau_0$
Newton-corrector: Iterate (4.2) until convergence. Decrease $ds$ if it fails to converge, back to predictor step. Increase $ds$ for next step if it converges quickly.
New tangent: Calculate $\tau_1$ from (4.3), set $(u_0, \lambda_0, \tau_0) = (u_1, \lambda_1, \tau_1)$ and return to the predictor step.

algorithm doesn't work at bifurcation points where $A$ is singular, which generally doesn't happen since the algorithm shoots past these points most of the time.

**Definition 13.** *A simple bifurcation point is a point* $(u, \lambda)$ *where* $\det(A)$ *changes sign. The assumption is that this happens due to a simple eigenvalue of* $A$ *crossing zero.*

This excludes folds, where a simple eigenvalue reaches zero, but $\det(A)$ doesn't change sign (can be seen in diagramm anyway). It also excludes points with an even number of eigenvalues crossing $i\mathbb{R}$. **Remark.** [14] says that $\xi = 1/2$ is the numerically most robust.

In pde2path, the matlab toolbox described in [14], by default uses LU-decomposition to detect a sign change of $\det(A)$. □

Due to the size of our problem this approach is not feasible. The evaluation of $\det(A)$ is too costly. Instead we calculate a small number of eigenvalues

close to $i\mathbb{R}$. The problem of this approach is that it can detect a change without an eigenvalue crossing $i\mathbb{R}$. This can happen if an eigenvalue with a positive (or negative) real part drops out of the list of eigenvalues while an eigenvalue with negative (or postive respectively) real part comes into the list. One way to restrict the possibility of this happening is by starting on a branch with known stability. E.g. on the stable branch we know that all eigenvalues are on the same side of the imaginary axis, therefore the next detected bifurcation can't be due to this effect.

After we detect a bifurcation between $s_k$ and $s_{k+1}$ the bifurcation is located by a bisection method.

### 4.2.2 Switching branches

When we detect a bifurcation we want to be able to switch the branch we are following. We do this by using 'Method I' of [8].

$z(s)$ shall be the solution branch. $z(s_0)$ is a singular point if $\operatorname{rank}(G_z(z(s_0))) = N-1$. Since $G_z \in \mathbb{R}^{N\times N+1}$ there exist two linearly independent null vectors $\Phi_1, \Phi_2 \in \mathbb{R}^{N+1}$. One can set $\Phi_i^T\Phi_j = \delta_{ij}$, therefore we have an orthonormal system of $\ker(G_z(z(s_0)))$. Also $G_z^T(z(s_0)) \in \mathbb{R}^{N+1\times N}$ has rank $N-1$ therefore

$$\ker(G_z^T(z(s_0))) = \operatorname{span}(\psi).$$

Since

$$G(z(s)) = 0$$

it follows that

$$G_z(z(s))\dot{z}(s) = 0,$$

thus at $s = s_0$, $\dot{z}(s_0)$ can be described by

$$\dot{z}(s_0) = \alpha\Phi_1 + \beta\Phi_2 \quad \alpha,\ \beta \in \mathbb{R}.$$

Through differentiation we get

$$G_z(z(s))\ddot{z}(s) + G_{zz}(z(s))\dot{z}(s)\dot{z}(s) = 0.$$

We multiply by $\psi^T$ and evaluate at $s = s_0$, then the first term vanishes and we get

$$\psi^T G_{zz}(z(s_0))\dot{z}(s_0)\dot{z}(s_0) = 0.$$

Then we substitute the presentation of $\dot{z}(s)$ and get

$$a_{11}\alpha^2 + 2a_{12}\alpha\beta + a_{22}\beta^2 = 0$$

where $a_{ij} = \psi^T G_z z(z(s_0))\Phi_i\Phi_j$. This quadratic equation has depending on

$$\Delta = a_{12}^2 - a_{11}a_{22}$$

up to two different solutions. Since $z(s)$ is a smooth solution branch $\Delta < 0$ is not possible. If $\Delta > 0$ there exist 2 nontrivial tangents therefore $z(s)$ is a bifurcation point. 'Method I' of [8] uses this equation. Use the tangent vector

$$\dot{z}(s_0) = \alpha\Phi_1 + \beta\Phi_2$$

and

$$\Phi_1 = \begin{pmatrix} \phi \\ 0 \end{pmatrix}, \quad \Phi_2 = \begin{pmatrix} \phi_0 \\ 1 \end{pmatrix}.$$

We know already one branch of solutions, therefore we already know one tangent. This determines one solution of the equation. Let the known tangent be $(\dot{u}_0, \dot{\lambda}_0)$ then

$$\beta_0 = \dot{\lambda}_0, \quad \dot{u}_0 = \alpha_0\phi + \beta_0\phi_0.$$

We can then get the second solution easily by setting

$$\alpha_1 = -\left(\frac{a_{11}\alpha_0}{\beta_0} + 2a_{12}\right), \quad \beta_1 = a_{11}$$

because

$$\begin{aligned} 0 &\stackrel{!}{=} a_{11}\alpha_1^2 + 2a_{12}\alpha_1\beta_1 + a_{22}\beta_1^2 \\ &= a_{11}\left(\frac{a_{11}^2\alpha_0^2}{\beta_0^2} + 4a_{11}a_{12}\frac{\alpha_0}{\beta_0} + 4a_{12}^2\right) - 2a_{12}\left(\frac{a_{11}\alpha_0}{\beta_0} + 2a_{12}\right)a_{11} + a_{22}a_{11}^2 \\ &= \frac{a_{11}^3\alpha_0^2}{\beta_0^2} + 2a_{12}a_{11}^2\frac{\alpha_0}{\beta_0} + a_{22}a_{11}^2 \\ &\Leftrightarrow a_{11}\alpha_0^2 + 2a_{12}\alpha_0\beta_0 + a_{22}\beta_0^2 = 0. \end{aligned}$$

Therefore we only need to know $a_{11}$ and $a_{12}$, but this needs evaluation of $G_{uu}$ and $G_{u\lambda}$. We can estimate these with the finite differences

$$\begin{aligned} a_{11} &= \frac{1}{\delta}\psi^T(G_u(u_0 + \delta\phi, \lambda_0) - G_u(u_0, \lambda_0))\phi \\ a_{12} &= \frac{1}{\delta}\psi^T((G_u(u_0 + \delta\phi, \lambda_0) - G_u(u_0, \lambda_0))\phi_0 + \dots \\ &\dots + G_\lambda(u_0 + \delta\phi, \lambda_0) - G_\lambda(u_0, \lambda_0)) \end{aligned}$$

This means we need $\psi^T, \phi, \phi_0$. We get $\phi$ and $\psi$ by calculating

$$G_u\phi = 0, \quad G_u^T\psi = 0$$

and normalize by setting $\|\phi\| = 1$, $\langle\psi, \phi\rangle = 1$. We then get $\phi_0$ through the presentation of $\dot{u}_0$

$$\phi_0 = \beta_0^{-1}(\dot{u}_0 - \psi^T\dot{u}_0\phi).$$

We then get algorithm 4.2.

**Algorithm 4.2** switch branch algorithm

$(u_0, \lambda_0)$ is a simple bifurcation point $\tau_0$ tangent along the branch.
Calculate $\phi$, $\psi$ with $G_u(u_0, \lambda_0)\phi = 0$, $G_u(u_0, \lambda_0)^T\psi = 0$, $\|\phi\| = 1$, $\langle\psi, \phi\rangle = 1$
Let $\beta_0 = \dot{\lambda}_0$, $\alpha_0 = \psi^T \dot{u}_0$, $\phi_0 = \beta_0^{-1}(\dot{u}_0 - \alpha_0\phi)$.
Choose some small $\delta > 0$, calculate the finite dimensions

$$a_{11} = \frac{1}{\delta}\psi^T(G_u(u_0 + \delta\phi, \lambda_0) - G_u(u_0, \lambda_0))\phi,$$
$$a_{12} = \frac{1}{\delta}\psi^T((G_u(u_0 + \delta\phi, \lambda_0) - G_u(u_0, \lambda_0))\phi_0 + \dots$$
$$\dots + G_\lambda(u_0 + \delta\phi, \lambda_0) - G_\lambda(u_0, \lambda_0)).$$

Assume $\beta \neq 0$ (see [8] if this is not true), set

$$\alpha_1 = -\left(\frac{a_{11}\alpha_0}{\beta_0} + 2a_{12}\right), \quad \tau_1 = \begin{pmatrix}\alpha_1\phi + a_{11}\phi_0 \\ a_{11}\end{pmatrix}.$$

Choose a weight $\xi$ and stepsize $ds$, set $\tau_0 = \tau_1/\|\tau_1\|_\xi$ and go back to the continuation algorithm. If there is no convergence in the continuation algorithm or it falls back onto the known branch one may change $\xi$ or $ds$.

# Chapter 5

# Experiments

## 5.1 Allen-Cahn equation

For checking the algorithm worked correctly I used a version of the Allen-Cahn equation

$$\begin{aligned} -0.25\Delta u - \lambda u - u^3 + u^5 = 0 \quad & \text{on } \Omega, \\ u = 0 \quad & \text{on } \partial\Omega \end{aligned}$$

on the axisymmetric rectangle $(-1,1)\times(-0.9,0.9)$ and compared the results to the results described in [14].

As it should be expected, we see in figure 5.1 $u = 0$ is the trivial solution. We see the that the bifurcation points are located at the same position, therefore our method of finding bifurcations seems to have worked without problems. Although I have found one extra branch since the searched intervall was a bit larger. Also the found branches look the same. If you use an equivalent norm the plot looks qualitatively the same but it will probably

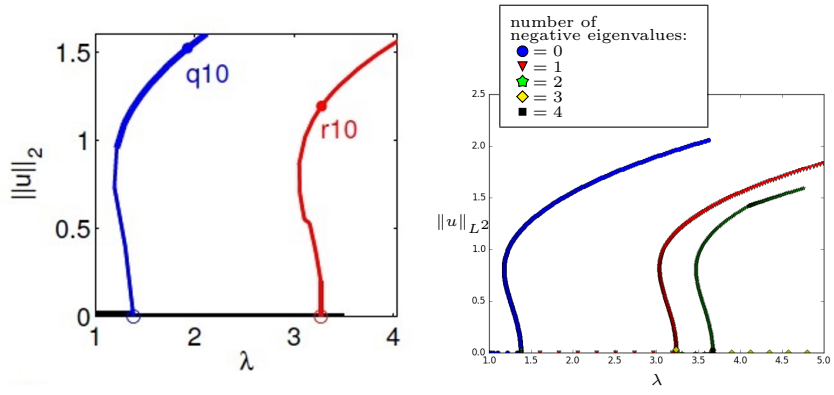


Figure 5.1: The plot from [14] (left) for comparison, next to my plot in the $L^2$-norm.

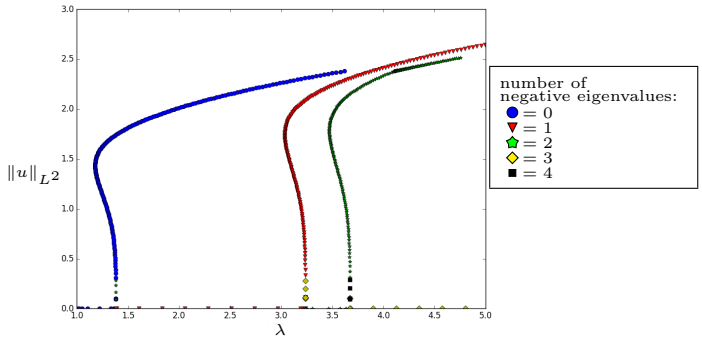


Figure 5.2: The bifurcation diagram in the $H^1$-norm.

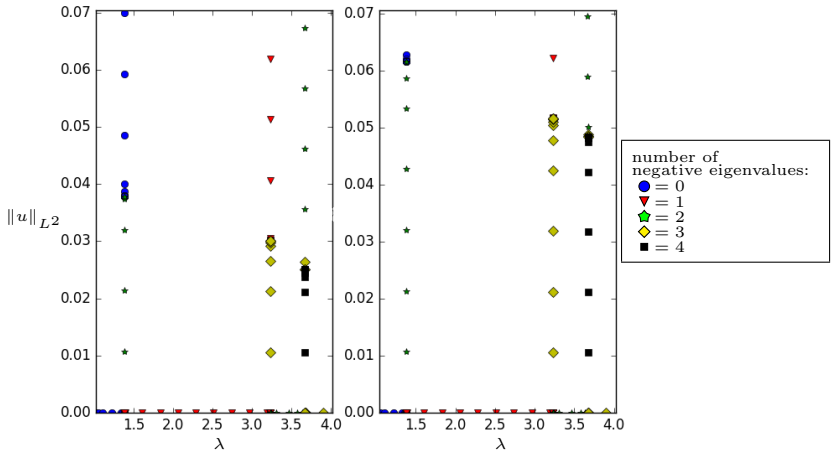


Figure 5.3: Zoomed in versions of the diagrams for different numbers of calculated eigenvalues (left: 15 eigenvalues, right: 20 eigenvalues)

be scaled in some form. In figure 5.2 you can see the plot when using the $H^1$-norm.

On the nontrivial branches close to the bifurcation points we see a difference in the number of eigenvalues with negative real part of two. This seems to be due numerical errors. One thing that seems to support this is that when I computed the diagrams with a different number of calculated eigenvalues these areas changed as can be seen in figure 5.3. Therefore it can't be an effect inherent to the equation but must be brought on by other means.

When looking at the solutions in figure 5.4 we see that they behave the same way as in [14], although the solution one the first branch is the negative one, which is no problem as for a solution $u$ of the Allen-Cahn equation $-u$

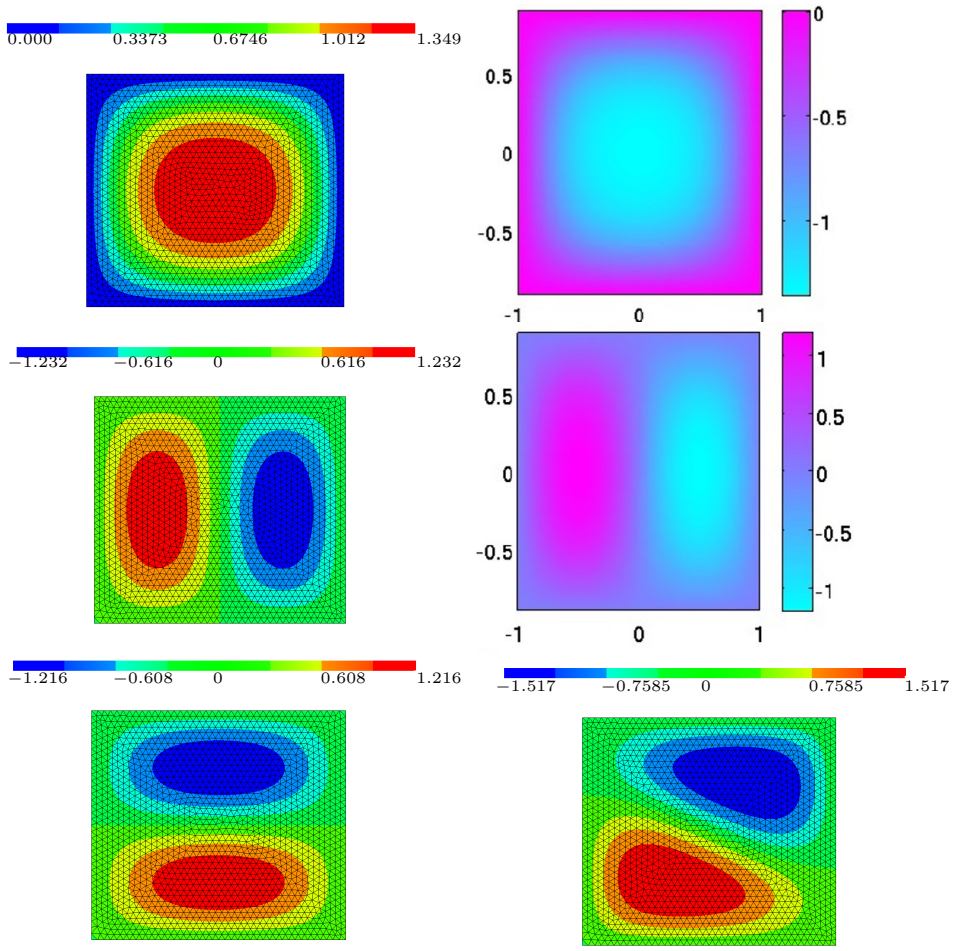


Figure 5.4: Solutions on the three nontrivial branches at $\lambda = 1.93$, $\lambda = 3.33$ and $\lambda = 3.73$ and the last point of the third branch. Next to the solutions from [14] at $\lambda = 1.93$, $\lambda = 3.33$

is also a solution. For the first two nontrivial branches the qualitative look of the solutions is not changing along the branch, while at the third branch the peaks shift into the corners in a clockwise direction.

## 5.2 Boundary layer

Due to the difference in scales between the diffusion and the convection term if the flow velocity is large enough the diffusion term can be neglected:

$$\frac{1}{\mathrm{Re}}\Delta\mathbf{u} = O(\mathrm{Re}^{-1}) \leftrightarrow \mathbf{u}\cdot\nabla\mathbf{u} = O(\mathbf{u}).$$

This makes sense in large areas of the flow but one big area where this will fail is at the Dirichlet boundary (no-slip condition). Flow speeds are small there, therefore also the convection term is small. This means there exists a boundary layer, with thickness of about $O(\mathrm{Re}^{-\frac{1}{2}})$, which behaves differently than most of the rest of the geometry.

One can use this to solve two different equations on two different areas with interface conditions (see section A.1).

If you don't want to neglect diffusion you have to remember that this boundary layer exists. For numerical simulation this means that your mesh should be finer at the boundary such that the boundary layer is taken care of.

To demonstrate this effect I have done some of the following tests on meshes with boundary refinement and without it.

## 5.3 Testing the preconditioner

To test the preconditioner I used the Navier-Stokes equation on the cube $\Omega = (-1,1) \times (-1,1) \times (-1,1)$ with the righthandside $\mathbf{f} = (0,0,x)^T$ with no-slip condition on $\partial\Omega$ with and without boundary refinement.

Also I tested on the periodic pipe with the axis $(0,0,-5) - (0,0,5)$ and radius $r = 1$ with no-slip condition on the pipe wall and periodic boundary conditions on the in- and outflow boundaries.

We want to answer some questions that might be of interest about the preconditioner and the linear equation solver in general.

- Is it worth the effort to implement the preconditioner or does the solver work good enough without it?
- How does the boundary layer influence the solver/Newton algorithm?
- Is there a remarkable difference when using this solver/preconditioner for NSolve or NCorr (see A.2.3)?
- How does this solver/preconditioner fare when used for computing the eigenvalues of the matrix

$$A = \begin{pmatrix} G_u & G_\lambda \\ \xi \dot{\mathbf{u}}^T & (1-\xi)\dot{\lambda} \end{pmatrix}$$

### 5.3.1 Value of the preconditioner

Is it really worth the effort to implement the preconditioner?

Short answer: yes.

It is really easy to realise the enormous value of the preconditioner after only a few tests. I have tested on two relatively coarse meshes to solve with and without the preconditioner, on the cube and on the pipe. (Cube:

max$h$ = 0.3, boundary max$h$ = 0.15, boundary layer thickness = 0.02; Pipe: max$h$ = 0.3, boundary max$h$ = 0.2, boundary layer thickness = 0.05.) When testing on the cube I used a zero-vector as the starting solution, while on the pipe I used the analytic laminar solution. On the cube this setup led to two Newton steps before the tolerance of $10^{-2}$ was reached. On the pipe this lead to instant convergence of the Newton algorithm, therefore in the table I have put the data of the Newton when we let the continuation algorithm run for one extra step. I tested at Re = 20.

| Newton tol $= 10^{-2}$ | preconditioned? | Newton iteration | iterations | achieved residual value |
|---|---|---|---|---|
| cube | False | 0 | 300 | 0.34 |
| cube | False | 1 | 300 | 0.39 |
| cube | True | 0 | 17 | 0.00095 |
| cube | True | 1 | 20 | 0.00048 |
| pipe | False | 0 | 300 | 0.096 |
| pipe | True | 0 | 5 | 0.00095 |

When only setting a tolerance of $10^{-3}$ the gmres isn't able to converge within the maximum number of iterations (300 iterations) when not using the preconditioner. In contrast, with the preconditioner and the same setup the solver achieves the tolerance within a few iterations.

### 5.3.2 Boundary layer influence

When I started the tests it became apparent pretty quickly that the continuation algorithm started to fail at certain Reynolds-numbers due to the Newton-algorithm not achieving the desired tolerance because of the gmres-solver hitting the maximum number of iterations. After some further testing I was able to discern three major influences in this effect.

The first one was almost obvious, the coarseness of the mesh was a factor, finer meshes lead to higher critical Reynolds numbers but the second influence is more important and it is the resolution of the boundary layer.

| boundary layer thickness | boundary maximum h | maximum h | critical Re |
|---|---|---|---|
| 0.05 | 0.15 | 0.3 | 20.01 |
| 0.05 | 0.1 | 0.2 | 21.47 |
| 0.02 | 0.1 | 0.2 | 39.95 |

The third influence was not that obvious but after reviewing the data was also not that hard to see. It was the restart count on the gmres-solver. The number of iterations needed for the solver to achieve its set tolerance steadily increased while getting closer to the critical Reynolds number as seen in figure 5.5. We see two large jump in this plot as the number of iterations arrives at the restart points.

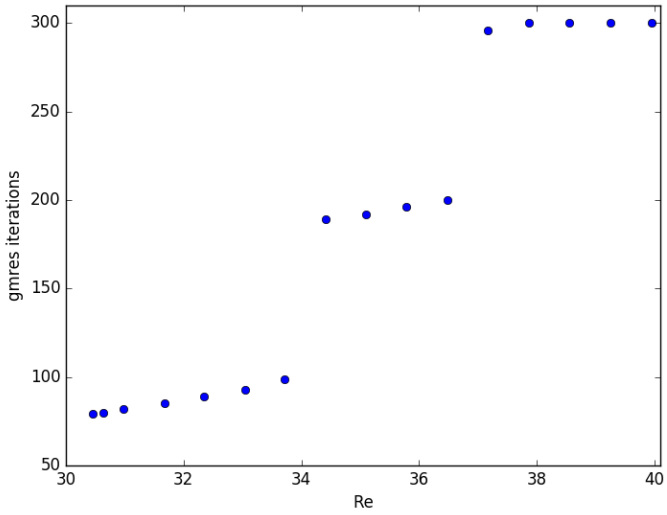


Figure 5.5: The number of gmres-iterations required to get the defined tolerance. Two jumps can be seen when hitting the restart counts of 100 and 200 iterations. At 300 the maximum number of iterations is reached.

Thus the main possibilities to achieve higher break-off points are to resolve the boundary layer finer, use a finer mesh, or in the same type of approach use a higher order discretization or to set a higher restart count or lower tolerances.

We already see that the boundary layer has an enormous influence.

I also wanted to compare the effects of refining for the boundary layer against meshes without boundary refinement. Therefore I tested on the cube with four different meshes: Two with boundary refinement, one with the same resolution on the main part of the cube without the refinement and one with finer resolution overall but also without boundary refinement. The coarser refined mesh has about the same number of degrees of freedom as the finer unrefined mesh.

| max$h$ | boundary thickness | boundary max$h$ | Newton iterations |
|---|---|---|---|
| 0.3 | 0.02 | 0.1 | 6 |
| 0.2 | 0.02 | 0.1 | 5 |
| 0.2 | - | - | 88 |
| 0.1 | - | - | diverged |

It is quite interesting to see what happened. While between the first three tests there was the expected effect, in the fourth test the Newton iteration failed and diverged after the gmres failed to achieve the wanted tolerance. This is probably due to the increased complexity of the linear

equation system while not getting the advantage of better resolving the problematic areas.

To show this I have put here an overview of the number of gmres iterations needed during the Newton iteration. Due to the third mesh having a lot of steps where only one step of gmres was performed the average was skewed quite a bit. Therefore I have also put here the overview of the first five steps of the Newton algorithm since the refined mesh only took five steps and I thought it would be the most interesting comparison.

| | gmres iterations | | | | | |
|---|---|---|---|---|---|---|
| | all Newton steps | | | first 5 steps | | |
| mesh | mean | min | max | mean | min | max |
| h=0.3 refined | 35.67 | 34 | 38 | 35.4 | 34 | 38 |
| h=0.2 refined | 35 | 33 | 37 | 35 | 33 | 37 |
| h=0.2 | 13.43 | 1 | 211 | 46.2 | 44 | 49 |
| h=0.1 | 192.58 | 6 | 300 | 257.6 | 191 | 300 |

### 5.3.3 NSolve versus NCorr

One big question is if the preconditioner still works as well when used in the setting of the (pseudo-)arclength continuation.

There we use the matrix

$$A = \begin{pmatrix} G_{\mathbf{u}} & G_{\lambda} \\ \xi\dot{\mathbf{u}} & (1-\xi)\dot{\lambda} \end{pmatrix}$$

for our Newton correction. Due to preconditioner described in section 3.10 being made for the block $G_{\mathbf{u}}$ we need to adjust it. When $M_{G_{\mathbf{u}}}$ is the preconditioner for $G_{\mathbf{u}}$ then I simply made the preconditioner for $A$ by defining

$$M_A := \begin{pmatrix} M_{G_{\mathbf{u}}} & 0 \\ 0 & 1 \end{pmatrix}.$$

But this begs the question if this still works as well. Therefore I made a test to compare them on the cube with a mesh with boundary refinement ($h = 0.3$, $bh = 0.15$, boundary thickness$= 0.02$).

| NSolve/NCorr | Newton step | gmres iterations | residual value at Newton step |
|---|---|---|---|
| NSolve | 0 | 30 | 0.11 |
| | 1 | 33 | 0.0075 |
| | 2 | 32 | 0.0017 |
| | 3 | 31 | 0.00051 |
| | 4 | 32 | $6.5 \cdot 10^{-5}$ |
| | 5 | 33 | $2.3 \cdot 10^{-5}$ |
| | 6 | 33 | $8.0 \cdot 10^{-6}$ |
| NCorr | 0 | 30 | 0.11 |
| | 1 | 33 | 0.0075 |
| | 2 | 32 | 0.0017 |
| | 3 | 31 | 0.00051 |
| | 4 | 32 | $6.6 \cdot 10^{-5}$ |
| | 5 | 33 | $2.4 \cdot 10^{-5}$ |
| | 6 | 33 | $8.1 \cdot 10^{-6}$ |

It shows that this adjustment didn't make any distinguishable difference.

### 5.3.4 computing eigenvalues

As described in section 4.2 we need to compute eigenvalues close to zero to check if one crosses over. Thus we need to use the shift-invert mode of **eigs** from scipy [7]. Thankfully **eigs** is implemented in a way so that one can specify how this inverse works. (More specifically how the matrix-vector multiplication with the inverse works.)

Therefore we want to use our gmres-precondtioner combo for this. We'll now look at how this fares for calculating the eigenvalues.

I have tested on a mesh on the cube ($h = 0.3$, boundary: $h = 0.15$, thickness= 0.02) and did two full continuation steps.

| gmres steps in **NCorr** | gmres steps in **eigs** | | | | |
|---|---|---|---|---|---|
| max | times called | mean | min | max | median |
| 33 | 322 | 37.11 | 34 | 38 | 37 |
| 34 | 268 | 38.68 | 36 | 39 | 39 |

We see that the solver was called about 300 times in **eigs** and it always was slightly worse than when used for **NCorr** but not really impacting a lot. This reflects what we would have expected, although I have found in early testing stages (when I hadn't set the restart count for gmres myself (default= 20)), that the performance was way worse. Thus it seems to be a good idea to set the restart count as high as possible.

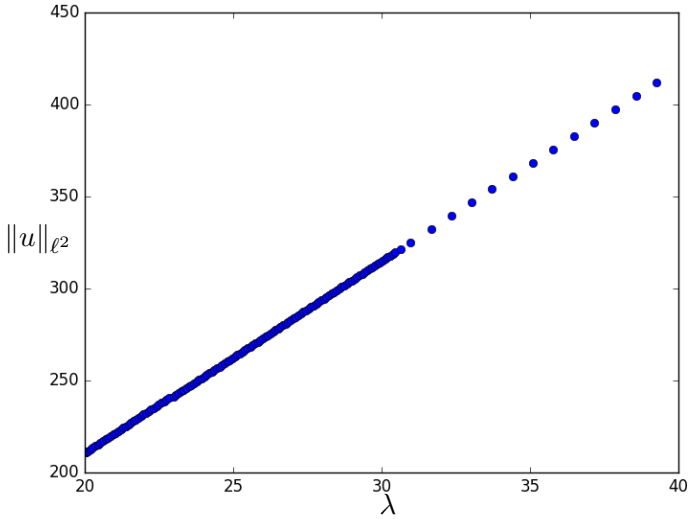


Figure 5.6: Path of the laminar solution, the change of resolution comes from a change of dsmax.

## 5.4 Pipe continuation

Sadly, due to computational problems, I was not able to do a continuation on a long pipe (length = 100 pipe diameters). As mentioned in the introduction one of the bigger problems is getting a fine enough mesh.

Since transitional flow needs a long enough pipe and I never was able to reach a high enough Reynolds number, I didn't get transitional flow.

At least I was able to use the algorithm on the short pipe with length 10 radii.

In figure 5.6 we see what we could expect already. The laminar solution gets linearly bigger as the Reynolds number increases.

### 5.4.1 Ideas for testing in pipe flow

Since I never got to test on the correct setting I was not able to experiment on how to change the continuation algorithm to jump from the laminar solution onto any turbulent flow. However I'd like to list here some ideas that I had. All of them evolve around changing the predictor step

$$\begin{pmatrix} u^1 \\ \lambda \end{pmatrix} = \begin{pmatrix} u_0 \\ \lambda_0 \end{pmatrix} + ds\tau_0.$$

The core idea is, since the transition is brought on by finite amplitude disturbances, to add a vector $b$ onto $\tau_0$ to jump away from the laminar solution

$$\begin{pmatrix} u^1 \\ \lambda^1 \end{pmatrix} = \begin{pmatrix} u_0 \\ \lambda_0 \end{pmatrix} + ds(\tau_0 + b).$$

The question is on how to choose this vector $b$. The problem is that we want to choose $b$ in a way such that it facilitates a jump onto another branch but doesn't hinder the Newton correction too much if no jump occurs.

I had three ideas on how to choose $b$:

- Simply a random vector with a fixed norm. This might lead to the highest probability of jumps but also of a diverging Newton correction.
- A vector that only brings a disturbance on the periodic boundary for the influence of inflow disturbances.
- Or a vector that gives a disturbance on or near the pipe wall (Dirichlet boundary conditions) to look for an influence from the pipe itself.

# Appendix A

# Appendix

## A.1 Boundary layer approximation

sources: [2], [4]

Due to the existence of a boundary layer for the Navier-Stokes equation one can try to approximate the solution with two different equations on two different domains. Here I will give the idea for two-dimensional flow on a flat plate ($\Omega = \mathbb{R} \times \mathbb{R}^+$). Let $\mathbf{u} = (v, w)^T \in \mathbb{R}^2$ and $\varepsilon = 1/\mathrm{Re} \ll 1$ be a fixed value. Then the Navier-Stokes equation looks as follows:

$$\begin{aligned} v_t + vv_x + wv_y + p_x &= \varepsilon \Delta v, \\ w_t + vw_x + ww_y + p_y &= \varepsilon \Delta w, \\ v_x + w_y &= 0, \\ v|_{y=0} = w|_{y=0} &= 0, \\ v(0,x,y) = v_I(x,y),\ w(0,x,y) &= w_I(x,y). \end{aligned} \tag{A.1}$$

### Outer expansion

We start in the outer area, which means outside the boundary layer where we can think of the fluid as frictionless as $\varepsilon$ is small. This should result in the Euler-equation. We start by making an ansatz:

$$\begin{aligned} v &= v_0 + \varepsilon v_1 + \varepsilon^2 v_2 + \dots, \\ w &= w_0 + \varepsilon w_1 + \varepsilon^2 w_2 + \dots, \\ p &= p_0 + \varepsilon p_1 + \varepsilon^2 p_2 + \dots. \end{aligned}$$

When we insert this into our equation (A.1) we get in the lowest order the Euler equations:

$$\begin{aligned} \partial_t v_0 + v_0\partial_x v_0 + w_0\partial_y v_0 + \partial_x p_0 &= 0, \\ \partial_t w_0 + v_0\partial_x w_0 + w_0\partial_y w_0 + \partial_y p_0 &= 0, \\ \partial_x v_0 + \partial_y w_0 &= 0, \\ v(0,x,y) = v_I(x,y),\ w(0,x,y) &= w_I(x,y). \end{aligned} \tag{A.2}$$

**Inner expansion**

Inside the boundary layer we expect that there are large differences in the $y$-direction, while there are little differences in the $x$-direction. Thus we apply a scaling only on the $y$-direction $T := t,\ X := x,\ Y = y/\varepsilon^\alpha$ with an unknown $\alpha > 0$, which we will determine later,

$$\begin{aligned} V(T,X,Y) &= v(T,X,\varepsilon^\alpha Y), \\ W(T,X,Y) &= w(T,X,\varepsilon^\alpha Y), \\ P(T,X,Y) &= p(T,X,\varepsilon^\alpha Y). \end{aligned}$$

Inserting this into our equation (A.1) gives us

$$\begin{aligned} \partial_T V + V\partial_X V + \varepsilon^{-\alpha} W\partial_Y V + \partial_X P &= \varepsilon\partial_X^2 V + \varepsilon^{1-2\alpha}\partial_Y^2 V, \\ \partial_T W + V\partial_X W + \varepsilon^{-\alpha} W\partial_Y W + \partial_X P &= \varepsilon\partial_X^2 W + \varepsilon^{1-2\alpha}\partial_Y^2 W, \\ \partial_X V + \varepsilon^{-\alpha}\partial_Y W &= 0, \\ V|_{Y=0} = W|_{Y=0} &= 0. \end{aligned} \tag{A.3}$$

We then use a general expansion for $V,\ W,\ P$

$$\begin{aligned} V &= V_0 + \varepsilon^\beta V_1 + \varepsilon^{2\beta} V_2 + \dots, \\ W &= W_0 + \varepsilon^\beta W_1 + \varepsilon^{2\beta} W_2 + \dots, \\ P &= P_0 + \varepsilon^\beta P_1 + \varepsilon^{2\beta} P_2 + \dots. \end{aligned}$$

with unknown $\beta > 0$, still to determine.

When we then insert this in the third equation in (A.3) we get

$$[\partial_X V_0 + \varepsilon^\beta \partial_X V_1 + \dots] + \varepsilon^{-\alpha}[\partial_Y W_0 + \varepsilon^\beta \partial_Y W_1 + \dots] = 0.$$

The leading $\varepsilon$-exponent is at $\partial_Y W_0$ which suggests

$$\partial_Y W_0 = 0, W_0(T,X,0) = 0 \forall T, X \Rightarrow \boxed{W_0 = 0}.$$

The next smallest $\varepsilon$-exponent suggests $\alpha = \beta$, thus this gives us

$$\boxed{\partial_X V_0 + \partial_Y W_1 = 0}.$$

We then insert the expansion into the first equation in (A.3) and get

$$\begin{aligned}
&[\partial_T V_0 + \varepsilon^\beta \partial_T V_1 + \dots] + [V_0 + \varepsilon^\beta V_1 + \dots] \cdot [\partial_X V_0 + \varepsilon^\beta \partial_X V_1 + \dots] \\
&+ \varepsilon^{-\alpha}[0 + \varepsilon^\alpha W_1 + \dots] \cdot [\partial_Y V_0 + \varepsilon^\alpha \partial_Y V_1 + \dots] + [\partial_X P_0 + \varepsilon^\alpha \partial_X P_1 + \dots] \\
&= \varepsilon[\partial_X^2 V_0 + \varepsilon^\alpha \partial_X^2 V_1 + \dots] + \varepsilon^{1-2\alpha}[\partial_Y^2 V_0 + \varepsilon^\alpha \partial_Y^2 V_1 + \dots].
\end{aligned}$$

If we choose $\alpha > 1/2$ then $1 - 2\alpha < 0$ and we only have one leading term $\partial_Y^2 V_0 = 0$. But if we choose $\alpha = 1/2$ such that $1 - 2\alpha = 0$ we get the highest number of leading terms.

$$\boxed{\partial_T V_0 + V_0 \partial_X V_0 + W_1 \partial_Y V_0 + \partial_X P_0 = \partial_Y^2 V_0}.$$

With $\alpha = \beta = 1/2$ we have the thickness of the boundary layer as $\delta(\varepsilon) = O(\varepsilon^{\frac{1}{2}})$

It remains to insert the expansion into the second equation in (A.3)

$$\begin{aligned}
&[\partial_T W_0 + \varepsilon^{\frac{1}{2}} \partial_T W_1 + \dots] + [V_0 + \varepsilon^{\frac{1}{2}} V_1 + \dots] \cdot [0 + \varepsilon^{\frac{1}{2}} \partial_X W_1 + \dots] \\
&+ \varepsilon^{-\frac{1}{2}}[0 + \varepsilon^{\frac{1}{2}} W_1 + \dots] \cdot [0 + \varepsilon^{\frac{1}{2}} W_1 + \dots] + \varepsilon^{-\frac{1}{2}}[\partial_Y P_0 + \varepsilon^{\frac{1}{2}} \partial_Y P_1 + \dots] \\
&= \varepsilon[0 + \varepsilon^{\frac{1}{2}} \partial_X^2 W_1 + \dots] + [0 + \varepsilon^{\frac{1}{2}} \partial_Y^2 W_1 + \dots].
\end{aligned}$$

In the leading order we now get

$$\boxed{\partial_Y P_0 = 0}.$$

Thus the pressure is constant in the $y$-direction in the boundary layer.

**matching**

Our two expansions need to be compatible with each other we start with a condition in the limit for $\varepsilon \to 0$. Then $\delta(\varepsilon) \to 0$ and $V_0$, $W_0$, $P_0$ give the boundary connection between the boundary conditions and $v_0(\delta(\varepsilon))$, $w_0(\delta(\varepsilon))$, $p_0(\delta(\varepsilon))$.

$$\begin{aligned}
\lim_{Y\to\infty} V_0(T, X, Y) &\stackrel{!}{=} \lim_{y\to 0} v_0(t, x, y), \\
\lim_{Y\to\infty} W_0(T, X, Y) &\stackrel{!}{=} \lim_{y\to 0} w_0(t, x, y), \\
\lim_{Y\to\infty} P_0(T, X, Y) &\stackrel{!}{=} \lim_{y\to 0} p_0(t, x, y).
\end{aligned} \tag{A.4}$$

But it is then still discontinuous for $\varepsilon > 0$. Due to the second condition and $W_0 = 0$ we have $w_0(t, x, 0) = 0$ which corresponds to the typical boundary condition of the Euler equations ($\mathbf{u} \cdot \nu = 0$).

**Solving for outer expansion**

The outer expansion doesn't need information from the inner expansion therefore we start by solving the Euler equation (A.2) with $w_0(t,x,0)=0$ in the outer area. Due to the third condition in (A.4) and $\partial_Y P_0 = 0$ we get $P_0(T,X,Y) = p_0(t,x,0)\ \forall T=t, X=x, Y$.

This means that the pressure in the boundary layer is fully described by the pressure from the outer expansion at the boundary ($y=0$).

**Solving for inner expansion**

With $p|_{y=0}$ and $v_0|_{y=0}$ given from the outer flow we can solve for the inner expansion. The equations are called Prandtl's boundary layer equations.

$$\begin{aligned}
\partial_T V_0 + V_0\partial_X V_0 + W_1\partial_Y V_0 + \partial_X(p_0|_{y=0}) &= \partial_Y^2 V_0,\\
V_0|_{Y=0} = W_1|_{Y=0} &= 0,\\
\lim_{Y\to\infty} V_0(T,X,Y) &= v_0(t,x,0),\\
\partial_X V_0 + \partial_Y W_1 &= 0,\\
V_0(0,X,Y) = v_I(x,0) \quad &\text{if } \mathbf{u}_I \text{ has no boundary layer.}
\end{aligned}$$

**full approximation**

If we'd define our approximation like this:

$$\hat{v} := \begin{cases} V_0(t,x,\frac{y}{\varepsilon^{\frac{1}{2}}}) & , y\in(0,\delta(\varepsilon)) \\ v_0(t,x,y) & , y\in(\delta(\varepsilon)),\infty) \end{cases}$$

it would be discontinuous and therefore not a suitable approximation.

We instead define it through

$$\hat{v}(t,x,y) := V_0\left(t,x,\frac{y}{\varepsilon^{\frac{1}{2}}}\right) + v_0(t,x,y) - \lim_{y\to 0} v_0(t,x,y).$$

Where the limit is used such that $\hat{v}$ fulfills the boundary condtition. Analogously we define $\hat{w}$ and $\hat{p}$

$$\begin{aligned}
\hat{w}(t,x,y) &:= w_0(t,x,y) \quad (\text{since } W_0 \text{ and } w_0(t,x,y)=0),\\
\hat{p}(t,x,y) &:= p_0(t,x,y)
\end{aligned}$$

(since the pressure is constant in $Y$ in the boundary layer).

Result: In a boundary layer of thickness $O(\sqrt{\varepsilon})$ the horizontal velocity $v_0$ is corrected such that at $y=0$ the no-slip condition $\mathbf{u}=0$ is fulfilled. The vertical velocity already fulfills $w_0(t,x,0)=0$ and thus doesn't need to be corrected.

For a slightly more physical view on that subject look at [4].

## A.2 Used code

I used Netgen/NGSolve [13] and its Python interface for implementing the continuation algorithm. NGSolve is a FEM-Solver using C++. The whole implementation of my code is done in Python and follows the description in [14].

### A.2.1 ContCollection

ContCollection is the base class for everything in this code. It collects everything that is needed for running the algorithm. Therefore all functions used are called with an object of this class.

The constructors signature is

```
p = ContCollection(spa, grf, amat, amatinv,
                   applyBLF, rhs, invlam=False).
```

I will go through these arguments now.

#### spa, grf

**spa** takes the finite element space from NGSolve on which the FEM is defined. **grf** is the Gridfunction (also from NGSolve) defined on the FESpace **spa** for the solution.

#### amat and amatinv

**ContCollection** takes the matrix $A$ and its inverse (or a solver) as linear operators **amat** and **amatinv** which follows the rules of the scipy-linear operators with an added necessary update function. Due to this implementation one can use any desired form of the matrix and its solver. It only needs to give the result of the matrix-vector multiplication. **amat** and **amatinv** needs to be able to give the multiplication with $G_u$ or $G_u^{-1}$ respectively because these are needed in swibra (see A.2.5) and NSolve. Also **amat** needs to be able to give $G_\lambda$ as it is needed in swibra. Again due to the requirements of swibra both operators need to be able to give a transposed version.

There are default implementations. These use the multiplication and inverse given from the bilinearform in NGSolve.

#### applyBLF

**applyBLF** is a Bilinearform from NGSolve used for the calculation of the residuum of the nonlinear problem. The last component given by the number-FESpace is not relevant for this, thus it can be used to define a norm for the solution (e.g. $L^2$, $H^1$-norm). If this option is used the parameter **L2n** has to be set as true.

**rhs**

This is the shortened vector for the righthandside of the equation given as a numpy array.

**Parameters**

There are some parameters saved in **ContCollection**. Some of them are boolean ones that change how the computation works (righthandside is the default value).

- **invlam = False**: Describes if the parameter $\lambda$ of the equation is given in its reciprocal value. Since it is already needed in the constructor it can be given as a parameter in the constructor.
- **chord = True**: Defines if a chord method is used.
- **calctangent = True**: Gives if the tangent $\tau$ is calculated by using $A\tau = (\mathbf{0}, 1)^T$ or by using the secant defined by the last two points calculated.
- **L2n = False**: If this is true the norm of $u$ is calculated using **applyBLF**. Otherwise the vectornorm specified by **normord** is taken.
- **calceigs = True**: Since calculating the eigenvalues is the most expensive operation this value can be used to to turn it off if you only want to follow the branch without detecting bifurcation points.
- **saveu = False**: Defines if each calculated point is saved. Often it is not sensible for problems with a high number of degrees of freedom.

Other parameters are, e.g. in which area the continuation is done, the order of the used norm, the names of the autosaves (done with pickle) or a filename to save certain benchmarks to.

### A.2.2 cont

**cont** is the implemented function of the main continuation algorithm described in section 4.2. Its only argument is an object of **ContCollection**.

### A.2.3 NSolve and NCorr

**NSolve** is a Newton iteration using the matrix $G_u$ with a fixed $\lambda$. **NCorr** is the Newton correction used in **cont** using the matrix $A$.

Both again use **ContCollection** as their argument. **NCorr** also has a boolean argument that is used to describe if **NCorr** is called by **cont** or by the bisection method.

### A.2.4 biseccont

This is the bisection method that is called after a bifurcation is detected. Its arguments are the base class and the number of negative eigenvalues at the old point. After it has gotten close enogh to the bifurcation point it creates an object of the class **bifurpoint** which saves $\mathbf{u}, \lambda, \dot{\mathbf{u}}, \dot{\lambda}$ at the point.

### A.2.5 swibra

**swibra** is the implementation of the function described in subsection 4.2.2. Its arguments are **ContCollection** and a **bifurpoint** and it calculates the branch switching at that point.

### A.2.6 Example script

Here I give an example script using the Allen-Cahn equation.

```
from ngsolve import *
import sys
sys.path.append('..')
import netgen.geom2d as g2d
from numpy import zeros, inf
from LinOps import amatLinOp, amatinvLinOp
from contcoll import ContCollection
from cont import cont
from swibra import swibra
import matplotlib.pyplot as plt
import pickle

#Creating the geometry on which the equation is solved
#using netgen
geo = g2d.SplineGeometry()
p1 = geo.AppendPoint(-1,-0.9)
p2 = geo.AppendPoint(1,-0.9)
p3 = geo.AppendPoint(1,0.9)
p4 = geo.AppendPoint(-1,0.9)

geo.Append(["line", p1, p2], bc=1)
geo.Append(["line", p2, p3], bc=2)
geo.Append(["line", p3, p4], bc=3)
geo.Append(["line", p4, p1], bc=4)

#Generating the mesh using netgen
mesh = Mesh(geo.GenerateMesh(maxh=0.05))
```

```
#Defining the FESpace with ngsolve
V = H1(mesh, order=1, dirichlet=[1,2,3,4])
N = FESpace(mesh = mesh,type = "number")

fes = FESpace([V,N])

u,lam = fes.TrialFunction()
v,mu = fes.TestFunction()

#Defining the GridFunction with ngsolve
gfu = GridFunction(fes)
gfu.components[0].Set(0*x)
gfu.components[1].Set(0*x)

gfu0 = gfu.components[0]
gfu1 = gfu.components[1]

#Define a vector of indices to shorten
#the vectors to get only the free dofs
BA = fes.FreeDofs()
FDinds = zeros(gfu.vec.size,dtype=int)
n=0
for iterat in range(gfu.vec.size):
    if BA[iterat]:
        FDinds[n]=iterat
        n+=1
FDinds=FDinds[0:n]

#Defining the righthandside with
#a linearform from ngsolve
sourceLF = LinearForm(fes)
sourceLF += SymbolicLFI(0*v+0*mu)

sourceLF.Assemble()

#Defining the bilinearform for the linearized problem
Gu = 0.25*grad(u)*grad(v)-gfu1*u*v-...
...-3*(gfu0*gfu0)*u*v+5*(gfu0*gfu0*gfu0*gfu0)*u*v
Glam = -gfu0*lam*v

GuBLF = BilinearForm(fes)
GuBLF += SymbolicBFI(Gu+Glam)

GuBLF.Assemble()
```

```
#Defining the appBLF
#for calculating the residuum and the L2-norm
appu = 0.25*grad(u)*grad(v)-lam*u*v-...
...-(u*u*u)*v+(u*u*u*u*u)*v

appBLF = BilinearForm(fes, nonassemble = True)
appBLF += SymbolicBFI(appu)
appBLF += SymbolicBFI(u*u*mu)

#Defining the linear operators
#for solving the linear equation system
amat = amatLinOp(GuBLF,FDinds)
amatinv = amatinvLinOp(GuBLF,FDinds)

#Constructing the ContCollection
p = ContCollection(fes,gfu,amat,amatinv,appBLF,
                   sourceLF.vec.FV().NumPy()[FDinds])

#Setting different parameters for the continuation
p.eigtol = 1e-12
p.Ntol = 1e-12
p.numeigs = 20
p.normord = inf
p.L2n = True

p.lam = 1.0
p.lamold = 1.0
p.dsmax = 0.3
p.lammax = 5
p.normumax = 20
p.maxit = 200

p.autosavename0 = 'allencahnas0'
p.autosavename1 = 'allencahnas1'

#Setting the first tangent vector
p.lamprime = 1
p.uprime = gfu.components[0].vec.FV().NumPy()
p.uprime = p.uprime[FDinds[:-1]]

p.lamprime = p.lamprime/p.tauXiNorm()
p.uprime /= p.tauXiNorm()
```

```
#starting continuation
cont(p)

#switching branchesat the first bifucation point
swibra(p,p.bifurlist[0])

#continuing continuation after branchswitching
cont(p)

#repeat for other bifucation points
swibra(p,p.bifurlist[1])

cont(p)

swibra(p,p.bifurlist[2])

cont(p)
```

# Bibliography


[1] Peter Arbenz. Lecture notes on solving large scale eigenvalue problems. Zürich, 2016. Lecture Notes, ETH Zürich, available at http://people.inf.ethz.ch/arbenz/ewp/Lnotes/lsevp.pdf [accessed 27.03.2018].

[2] Anton Arnold. Modellierung mit partiellen differentialgleichungen. Vienna, 2016. Lecture Notes, Vienna University of Technology, available at http://www.asc.tuwien.ac.at/~arnold/lehre/zeitabhaengige_probleme/pde-modellierung.pdf [accessed 27.03.2018].

[3] W. Auzinger and J. M. Melenk. Iterative solution of large linear systems. Vienna, 2017. Lecture Notes, Vienna University of Technology, available at http://www.asc.tuwien.ac.at/~winfried/teaching/106.079/ [accessed 27.03.2018].

[4] Stefan Braun. Strömungslehre für tph. Vienna, 2015. Lecture Notes, Vienna University of Technology.

[5] Howard Elman, David Silvester, and Andy Wathen. *Finite Elements and Fast Iterative Solvers*. Oxford University Press, Oxford, second edition, 2014.

[6] Vivette Girault and Pierre-Arnaud Raviart. *Finite Element Methods for Navier-Stokes Equations*. Springer-Verlag.

[7] Eric Jones, Travis Oliphant, Pearu Peterson, et al. SciPy: Open source scientific tools for Python, 2001–. http://www.scipy.org/; [accessed 20.03.2018].

[8] H.B. Keller. *Lectures on Numerical Methods In Bifurcation Problems*. Springer-Verlag, Berlin Heidelberg, 1986.

[9] R.R. Kerswell. Recent progress in understanding the transition to turbulence in a pipe. *IOP Publishing Ltd and London Mathematical Society*.

[10] Christian Kuehn. Lecture notes: Dynamical systems and pdes. Vienna, 2015. Lecture Notes, Vienna University of Technology, avail-

able at http://www-m8.ma.tum.de/personen/kuehn/courses/2014_notes_DynamicsPDE.pdf [accessed 27.03.2018].

[11] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *ARPACK Users'Guide: Solution of Large-Scale Eigenvalue Problems by Implicitely Restarted Arnoldi Methods*. SIAM, Philadelphia, PA, 1998.

[12] Joachim Schöberl. Numerical methods for partial differential equations. Vienna, 2009. Lecture Notes, Vienna University of Technology, available at https://www.asc.tuwien.ac.at/~schoeberl/wiki/index.php/Lecture_Notes [accessed 27.03.2018].

[13] Joachim Schöberl et al. Netgen/NGSolve, 1996–. https://ngsolve.org/ [accessed 20.03.2018].

[14] Hannes Uecker, Daniel Wetzel, and Jens D.M. Rademacher. pde2path - a matlab package for continuation and bifurcation in 2d elliptic systems. 2013.