universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

„Padé approximation for parametric Helmholtz problems"

verfasst von / submitted by

Konstantin Jung, B.Sc.

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

Master of Science (M.Sc.)

Wien, 2018 / Vienna, 2018

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	A 066 910
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Computational Science
Betreut von / Supervisor:	Univ.-Prof. Ilaria Perugia, PhD
Mitbetreut von / Co-Supervisor:	Francesca Bonizzoni, PhD

Contents

Abstract 3

1 Introduction 7

2 Single point Padé approximation 9
- 2.1 Single point Padé approximation in $\mathbb{C}$ 9
- 2.2 Single point Padé approximation in V 10
- 2.3 Summary of convergence theory 12
- 2.4 Algorithm 13
  - 2.4.1 Calculating p_i and q_i 13

3 Multi point Padé approximation 17
- 3.1 Multi point Padé approximation in $\mathbb{C}$ 17
- 3.2 Multi point Padé approximation in V 18
- 3.3 Algorithm 19

4 Model problems 23
- 4.1 Parametric interior Helmholtz problem 23
- 4.2 Scattering problem 28

5 Numerical results for the single point Padé approximation 31
- 5.1 Remarks on the implementation 31
- 5.2 Parametric interior Helmholtz problem 31
  - 5.2.1 Numerical results in 3D 34
- 5.3 Scattering problem 38
  - 5.3.1 Numerical results in 3D 41

6 Numerical results for the multi point Padé approximation 43
- 6.1 Remarks on the implementation 43
- 6.2 Parametric interior Helmholtz problem 44
- 6.3 Scattering problem 52

7 Conclusions 55

Appendix 57

Bibliography 66

List of Figures 68

Abstract

This master's thesis is concerned with an algorithm to solve parameter-dependent PDEs, which uses rational Padé approximations to reduce the necessary computational effort. We review a single point version, generalize it to multiple points using Newton-Padé approximations and test the algorithm on two examples, a scattering equation and a Helmholtz model problem.

In the first part, we review theoretical results of the algorithm and present the two model problems. In the second part, we test the algorithm numerically. The algorithm is implemented in Python using the library *NGSolve/Netgen*. We try to reproduce theoretical results for the single point method in two dimensions, expand the code to three dimensions and make some first tests for the multi point version. We will also see how the Newton-Padé approximation can be used as an eigenvalue solver for the Laplace operator.

Zusammenfassung

Diese Masterarbeit beschäftigt sich mit einem Algorithmus für parameterabhängige partielle Differentialgleichungen, der rationale Padéapproximationen verwendet, um den Rechenaufwand zu reduzieren. Wir wiederholen eine Version mit einem Punkt, verallgemeinern diese zu mehreren Punkten mit der Hilfe von Newton-Padéapproximationen und testen den Algorithmus numerisch an zwei Beispielen, einer Wellengleichung mit Zerstreuung und einem Modellproblem für die Helmholtzgleichung.

Im ersten Teil wiederholen wir theoretische Resultate des Algorithmus und präsentieren diese zwei Beispiele. Der Hauptfokus liegt auf dem zweiten Teil, wo wir den Algorithmus numerisch testen. Der Algorithmus wurde in Python mit der Bibliothek NGSolve/Netgen implementiert. Wir versuchen die theoretischen Resultate für die Ein-Punkt-Version in zwei Dimensionen zu wiederholen, erweitern den Code auf drei Dimensionen und machen erste Experimente für die Mehr-Punkte-Version. Als zusätzliche Anwendung werden wir sehen, dass man die Newton-Padéapproximation auch als Eigenwertlöser verwenden kann.

1 Introduction

This thesis is about an algorithm to solve partial differential equations (PDEs), which depend on a parameter. We will look at the special case of the Helmholtz equation, which describes a travelling wave and depends on a wave number $\nu^2 \in \mathbb{R}$. Often one needs to solve this PDE multiple times for different values of the parameter belonging to a certain interval of interest. The "direct" method of just solving the PDE multiple times with some Finite-Element solver is computationally very expensive for larger numbers of wave numbers and therefore often not good to use in practice. We will discuss a different algorithm using Padé approximations.

The idea will be to define a solution map T, which maps each wave number to the solution of the PDE for fixed boundary conditions and source term. In the next step we will approximate this solution map T by a Padé approximation. Using evaluation of this Padé approximation one can calculate solutions of the PDE for any wave number.

This will allow us to define an algorithm which consists of an offline and an online part. In the offline part one calculates the necessary data to construct the Padé approximation. This may be computationally expensive, but can be done in advance. The data is saved and a Padé approximation of T is constructed. The second part, the online phase, consists only of the evaluation of the Padé approximation of the solution map T. This can be done nearly instantly and also on less powerful machines. It is also important to note, that the offline part is independent of the wave number, for which the solution should be calculated in the online part.

Traditionally, Padé approximations are defined for real- or complex-valued functions (see [HR00] and [Cla76]). We will generalize this to functions with values in some Hilbert space using a least-square approach (see [BNP17]) in Chapter 2. In this paper, this was done for Padé approximations with one center and we will generalize this theory to multiple centers (also called Newton-Padé approximations) in Chapter 3. Again we will use the the definition for $\mathbb{C}$-valued functions as a basis (see [Cla78]).

The main focus of this thesis will be to test this algorithm numerically. Therefore, we define two model problems, a Helmholtz equation on a squared domain with a source term and Dirichlet boundary conditions and one example, which includes a scattering effect as well. We will introduce these equations in Chapter 4 and review some important theoretical results. For the implementation we will use the software *Netgen/NGSolve* (see [ngs18]) and its Python interface *NGSpy*. For the single point case (one center for the Padé approximation) we will try to repeat some of the numerical results from [BNP17] and [BNPP18] in two space dimensions, expand this to three space dimensions and do some more examples. This is done in Chapter 5. In Chapter 6 we will test, whether the multi-point Padé (Newton-Padé) can outperform the single point Padé especially in the high frequency regime. We will see that, for the implementation of the Newton-Padé, it is more challenging to find a

'good' setup of the centers and derivatives than for the single point Padé. We will see that we can get accurate results by using Chebyshev points. However we will only test this numerically and do not give a theoretical explanation to the problem of placing the centers and the derivatives. We will also see that the Padé approximation can be used to calculate eigenvalues of the Laplace operator, if we approximate a model problem for the Helmholtz equation. We will compare this way of calculating eigenvalues with a traditional method to calculate eigenvalues, the inverse iteration.
In Chapter 7 we will summarize the results and give an outlook on what could be further investigated in the future. In the appendix we will give the important parts of the code and make some comments about the implementation.

At this point I want to thank my two supervisors Prof. Ilaria Perugia and Dr. Francesca Bonizzoni for their great support and the many meetings in the last year.

2 Single point Padé approximation

2.1 Single point Padé approximation in $\mathbb{C}$

The first chapter is about the Padé approximation with one center. We will first look at the Padé approximation for $\mathbb{C}$-valued functions and then expand this theory to Hilbert space-valued functions in the next section.
There are different possibilities to approximate some function $f : \mathbb{C} \to \mathbb{C}$ by evaluating only the function and its derivatives and such an approximation is often needed in praxis. In order to investigate approximations we need the following definition from [Pri03].

Definition 2.1 (Holomorphic function). Let $G \subset \mathbb{C}$ be an open subset of the complex plane and $f : G \to \mathbb{C}$ a function. f is complex differentiable in $z \in G$, if

$$\lim_{h\to 0} \frac{f(z+h) - f(z)}{h}$$

exists. We say that f is holomorphic on G, if it is complex differentiable in every $z \in G$. Then we write $f \in \mathcal{H}(G, \mathbb{C})$.
If we just say that a function is holomorphic without mentioning a set, we mean that it is holomorphic in its whole domain.

One of the most famous approximation is probably the Taylor approximation. In case one does not have a holomorphic f, but f is only meromorphic it is often needed to make a different approach to deal with the singularities. This can be done using a Padé approximation instead of a Taylor approximation, i.e. approximating f not only by a polynomial but a fraction of two polynomials. We will make this idea more clear giving a couple of definitions. We will follow in this chapter the construction from [GHR98] and [BNP17].

Definition 2.2 (Meromorphic function). Let $f : U \subset \mathbb{C} \to \mathbb{C}$ be a complex function and $W \subset U$ a discrete set, i.e. $|W| < \infty$, where $|W|$ denotes the number of elements of W. If f is holomorphic on $U \backslash W$ and for each $\hat{z} \in W$ there exists some $n \in \mathbb{N}$ such that $(\hat{z} - z)^n f$ is holomorphic, we call f meromorphic and write $f \in \mathcal{M}(\mathbb{C}, \mathbb{C})$.

This definition can also be extended to functions with values in some Banach space $V := \mathcal{F}(\mathbb{C}^n, \mathbb{C})$ consisting of functions, i.e. for some $f : \mathbb{C} \to V$. Here $\mathcal{F}(\mathbb{C}^n, \mathbb{C})$ could be for example the space of all L^2- or H^k-functions for some $k \in \mathbb{N}$.
We also define the space of polynomials with degree less equal M.

Definition 2.3 (Space of polynomials). Let be K either $\mathbb{C}$ or $\mathbb{R}$, X some vector space and $M \in \mathbb{N}$ some natural number. We denote by $\mathbb{P}_M(K, X)$ the space of all polynomials with degree smaller equal M mapping from K to X. In case of $K = X$ we only write $\mathbb{P}_M(K)$ instead of $\mathbb{P}_M(K, K)$.

We will now define the Padé approximant for some complex valued function. Later on this definition will be generalized to a larger set of functions.

Definition 2.4 (Padé approximant in $\mathbb{C}$). Let $f : \mathbb{C} \mapsto \mathbb{C}$ be a holomorphic function and f_{i,z_0} its i-th Taylor coefficient in $z_0 \in \mathbb{C}$. Then $f(z) := \sum_{i=0}^{\infty} f_{i,z_0}(z-z_0)^i$ is the complex-valued power series centered in $z_0 \in \mathbb{C}$ and let $P(z) := \sum_{i=0}^{M} p_{i,z_0}(z-z_0)^i$ and $Q(z) := \sum_{i=0}^{N} q_{i,z_0}(z-z_0)^i$ be two polynomials of degree M respectively N. Then $P(z)/Q(z)$ is called a Padé approximant of f in z_0, if

$$\Big|\frac{P(z)}{Q(z)} - f(z)\Big| = \mathcal{O}(|z-z_0|^{M+N+1}). \tag{2.1}$$

We denote the Padé approximant also by $f_{[M/N]}$

Remark. Condition 2.1 can also be formulated in a slightly different way. We can also write that the first $M+N+1$ coefficients in a power series expansion of $fQ-P$ in z_0 should be zero, i.e.

$$(fQ-P)_{i,z_0} = 0 \qquad\qquad \text{for } i = 0, \dots, M+N. \tag{2.2}$$

One should also note that in 2.2 we only have $M+N+1$ conditions for $M+N+2$ unknowns and the trivial solution $P=Q=0$ also exists. Of course, this is not what we want to achieve, when we try to calculate well approximated values of $f(z)$. Therefore, another condition has to be added. This can be for example done by requiring that Q is normalized, i.e. $\sum_{i=0}^{N} |Q_i|^2 = 1$.

2.2 Single point Padé approximation in V

Now we will try to generalize this definition to V-valued functions. We will do this generalization just with the center $x_0 = 0$, but use functions, which are defined on the complex plane. For other centers one can do the definition equivalently, but the notation would be more complicated. First we need another definition. The following definitions are from [BNP17].

Definition 2.5 (Padé functional). Let $(V, \|\cdot\|_V)$ be a Hilbert space, $f : \mathbb{C} \to V$ a map, which is holomorphic around 0 and $\rho \in \mathbb{R}^+$ some positive, real parameter. We also have two polynomials $P \in \mathbb{P}_M(\mathbb{C}, V)$ and $Q \in \mathbb{P}_N(\mathbb{C})$ and a natural number $E \in \mathbb{N}$. Then we define

$$j_{E,\rho}(P,Q) := \left(\sum_{i=0}^{E} \|(Qf(z) - P(z))_i\|_V^2 \rho^{2i}\right)^{1/2}. \tag{2.3}$$

Using this definition we can define the Padé approximant for a V-valued function.

Definition 2.6 (Padé approximant in V). Let be P, Q, f, ρ and E as in the previous definition. Additionally we require $E \geq M+N$ and Q to be normalized, which means that

$\sum_{i=0}^{N}|Q_i|^2 = 1$. We also define $\mathbb{P}_N^1(\mathbb{C}) := \{S \in \mathbb{P}_N(\mathbb{C}) : \sum_{i=0}^{N}|S_i|^2 = 1\}$. Then we say that P/Q is the Padé approximant of f, if

$$j_{E,\rho}(P,Q) = \inf_{\substack{R\in\mathbb{P}_M(\mathbb{C},V),\\ S\in\mathbb{P}_N^1(\mathbb{C})}} j_{E,\rho}(R,S). \tag{2.4}$$

Again we denote the Padé approximant by $f_{[M/N]}$, if the degrees of the two polynomials are M and N.

Remark. In fact the Padé functional and Padé approximant can also be defined, if V is only a Banach space (see [HR00, Remark 2.2]). However, later when we define an algorithm to calculate the Padé approximant, we will need that the norm is induced by a scalar product.

We can see that with condition (2.4) we minimize the coefficients in the power series expansion of $fQ - P$. Therefore this is a natural extension to condition (2.2) for complex valued functions. The next step will be to ensure that the infimum in (2.4) does exist. We need this to make sure that we can calculate later polynomials P, Q for some given f. Therefore we will reformulate $j_{E,\rho}$ as a functional, which depends only on Q and then use the fact that $j_{E,\rho}$ is continuous and $P_N^1(\mathbb{C})$ is compact. This is done in the following theorem, which is from [BNP17]. We will review the proof here, since the derivation in 2.5 is important to understand later the algorithm.

Theorem 2.7 (Existence of a single point Padé approximant). *The infimum in* (2.4) *is always attained by at least one pair of polynomials* (P,Q).

Proof. In the first step (2.3) is rewritten in the following way.

$$\begin{aligned} j_{E,\rho}(P,Q)^2 &= \sum_{i=0}^{M} \|(Q(z)f(z) - P(z))_i\|_{V,}^2 \rho^{2i} \\ &+ \sum_{i=M+1}^{E} \|(Q(z)f(z) - P(z))_i\|_V^2 \rho^{2i} \\ &= \sum_{i=0}^{M} \|(Q(z)f(z) - P(z))_i\|_V^2 \rho^{2i} + \sum_{i=M+1}^{E} \|(Q(z)f(z)\|_V^2 \rho^{2i} \end{aligned} \tag{2.5}$$

We denote by p_i the coefficients of P and by q_i the coefficients of Q, i.e. $P(z) = \sum_{i=0}^{M} p_i z^i$ and $Q(z) = \sum_{i=0}^{N} q_i z^i$. The second equality holds since $p_i = 0$ for $i > M$. Now we can set

$$p_i = \sum_{j=0}^{\min(n,i)} q_j (f)_{i-j} \tag{2.6}$$

and therefore the first summand in (2.5) is always zero and P is uniquely determined by Q and f. Therefore, we have a minimization problem only in Q

$$\tilde{j}_{E_\rho}(Q) = \inf_{S\in\mathbb{P}_N^1(\mathbb{C})} \tilde{j}(S), \tag{2.7}$$

where $\tilde{j}(S) = \sum_{i=M+1}^{E} \|(S(z)f(z)\|_V^2 \rho^{2i}$. Since $\mathbb{P}_N^1(\mathbb{C})$ is homeomorphic to the unit sphere in $\mathbb{C}^{N+1}$ and therefore compact and $\tilde{j}_{E,\rho}(\cdot)$ continuous, the infimum is attained by at least one polynomial Q. Then we reconstruct P from Q and f using (2.6) and therefore have a solution (P, Q) for $j_{E,\rho}(\cdot)$. □

The next question is to find out what kind of approximation property we have for this generalized Padé approximant.

2.3 Summary of convergence theory

In this section, we will summarize some results for the convergence theory of the previously defined single point Padé approximation. All the results are from [BNP17, chapter 5], where one can also find the proofs.
We will set in the following $V = H^1(\Omega)$ for some open and bounded Lipschitz domain $\Omega \subset \mathbb{R}^d$ for $d \in \{1, 2, 3\}$. We define on V the following norm.

Definition 2.8 (Weighted H^1-norm). We define for $u \in H^1(\Omega) =: V$ and $\nu > 0$ a weighted H^1-Norm with

$$\|u\|_{V,\nu}^2 := \|\nabla u\|_{L^2(\Omega)}^2 + \nu^2 \|u\|_{L^2(\Omega)}^2 .$$

The following results have been proven for some map $T : \mathbb{C} \to V$, which fulfills the following conditions:

- Let be $R > 0$. Then T is meromorphic on the closed disk $\overline{B(0, R)}$.
- There is some $h : \mathbb{C} \to V$ holomorphic on $\overline{B(0, R)}$ and $g \in \mathbb{P}_N(\mathbb{C})$, where $g(0) \neq 0$, $\sum_{i=0}^{N} |(g)_i|^2 = 1$ and g is N-maximal, which means that for every polynomial p, $pg \in \mathbb{P}_N(\mathbb{C})$ implies $p \in \mathbb{C}$, such that $T(z) = \frac{h(z)}{g(z)}$ is an irreducible fraction.

The second condition is fulfilled, if a map is meromorphic and we define g in such a way that the singularities of T are the zeros of g. One can also see the connection to the Padé approximation since P as a polynomial is always holomorphic and the denominator Q fulfills the conditions which are given for g.

Theorem 2.9. *Let $G := \{z \in \mathbb{C} : g(z) = 0\}$ and assume that $R > 0$ is large enough such that $G \subset \overline{B(0, R)}$. We also want $h(z) \neq 0$ for every $z \in G$ and $T_{[M/N]}$ is the Padé approximant of T as defined in definition 2.6. Then*

$$\lim_{M \to \infty} \left\|T_{[M/N]}(z) - T(z)\right\|_{V,\nu} = 0$$

uniformly on all compact subsets of $\overline{B(0, R)} \backslash G$.
Let $A \subset \overline{B(0, \rho)} \backslash G$ be a compact subset. Then there exists some $M^\star \in \mathbb{N}$ such that we have the following estimate for all $M \geq M^\star$

$$\left\|T_{[M/N]}(z) - T(z)\right\|_{V,\nu} \leq C \sup_{z \in \partial B(0,R)} \|T(z)\|_{V,\nu} \left(\frac{\rho}{R}\right)^{M+1}, \tag{2.8}$$

where $\rho < R$ is the constant from definition 2.5. The constant $C > 0$ depends on $dist(0, A), \rho, R, N$ and $\min_{z \in A} |g(z)|$, but not on M (only if $dist(0, A) \to \rho$, then $C = \mathcal{O}(M)$).

Regarding the constant C, it is important to note that we always have $dist(0, A) \leq \rho$, since $A \subset \overline{B(0, \rho)}$ and therefore the distance of any point in A to zero smaller equal than ρ. With the last theorem and estimate (2.8), we know under the assumptions in this chapter the Padé approximant is converging to the approximated function exponentially with increasing M. In the numerical experiments later on, we will check whether we achieve exponential convergence also in practice. The theorem also tells us that for approximations P/Q of a function f, where the singularities of f are already the roots of Q, we should rather increase M and not N to get more accurate results.

2.4 Algorithm

We want to approximate our solution function T by a fraction of two polynomials P and Q with degrees M and N, i.e. $T(z) = \frac{P(z)}{Q(z)}$. Therefore our goal is to calculate the coefficients of this two polynomials $p_i \in V$ and $q_i \in \mathbb{C}$. For the construction of the algorithm we will follow [BNPP18, 3].
We have seen in Theorem 2.7, that there exists always a Padé approximant for a meromorphic function $f : \mathbb{C} \to V$, which we can calculate by solving the minimization problem (2.4).

2.4.1 Calculating p_i and q_i

As we have seen in the proof of theorem 2.7 once we have calculated the q_i we can reconstruct the coefficients p_i using the formula

$$p_i = \sum_{j=0}^{\min(N,i)} q_j (f)_{i-j}.$$

We take the minimum since $q_j = 0$ for $j > N$ since Q is a polynomial of degree N.
In order to calculate the coefficients q_i, we need the following theorem from [BNPP18].

Theorem 2.10 (Calculation of the coefficients q_i). *The minimization in* (2.7) *is equivalent to finding the normalized eigenvector corresponding to the smallest eigenvalue of the matrix $G_{E,\rho} \in \mathbb{C}^{N+1 \times N+1}$ with entries*

$$(G_{E,\rho})_{i,j} = \sum_{k=M+1}^{E} \langle (f)_{k-j}, (f)_{k-i} \rangle_{V, \sqrt{Re(z_0)}} \rho^{2k}, \qquad \textit{with } i, j = 0, \dots, N.$$

$G_{E,\rho}$ is Hermitian and positive semidefinite.

Proof. By the definition of the Taylor coefficients we know that $q_k = (Q)_k$ for $k = 0, \dots, n$ and from the product rule we get that $(Qf)_k = \sum_{n=0}^{k} q_n (f)_{k-n}$. We define that $(f)_k = 0$

if $k < 0$. Then we can rewrite the Padé functional in the following way.

$$\begin{aligned}
\tilde{j}_{E,\rho}(Q)^2 &= \sum_{k=M+1}^{E} \langle (Qf)_k, (Qf)_k \rangle_{V,\sqrt{Re(z_0)}} \rho^{2k} \\
&= \sum_{k=M+1}^{E} \langle \sum_{i=0}^{k} q_i (f)_{k-i}, \sum_{j=0}^{k} q_j (f)_{k-j} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2k} \\
&= \sum_{i,j=0}^{N} q_i^* q_j \sum_{k=0}^{E} \langle (f)_{k-j}, (f)_{k-i} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2k} \\
&= \vec{q}^* G \vec{q},
\end{aligned}$$

where $\vec{q} = \{q_0, q_1, \dots, q_N\}$ denotes the vector containing all the coefficients of Q. Therefore, minimizing $\tilde{j}_{E,\rho}(Q)^2$ is equivalent to minimizing $\vec{q}^* G \vec{q}$ and this product is minimized by taking $\vec{q}$ as the smallest eigenvector. In order to fulfill the constraint $\sum_{i=0}^{N} |q_i|^2 = 1$, we normalize $\vec{q}$.

$G_{E,\rho}$ is hermitian by its definition. Since $\vec{q}^* G_{E,\rho} \vec{q} = \tilde{j}_{E,\rho}(Q)^2$ for all $q \in \mathbb{C}^{N+1}$, we obtain with the definition of $\tilde{j}_{E,\rho}(\cdot)^2$, that $\vec{q}^* G_{E,\rho} \vec{q} \geq 0$ for all $q \in \mathbb{C}^{N+1}$. Therefore, $G_{E\rho}$ is positive semidefinite. □

Using this theorem, we know that in order to get the coefficients q_i, we have to calculate the normalized eigenvector corresponding to the smallest eigenvalue of $G_{E,\rho}$.

Now we have all the necessary tools to calculate the Padé approximation. We will summarize this in the following algorithm.

Algorithm 1 Single point Pade approximant

```
Require: some meromorphic function $f : \mathbb{C} \to V$ with $\Lambda$ being the set of singularities of
  $f$, $z_0 \in \mathbb{C} \backslash \Lambda, \rho \in \mathbb{R}^+, M, N, E \in \mathbb{N}$ with $M + N \leq E$
  for $k = 0, \dots, E$ do
    calculate Taylor coefficients $(f)_{k,z_0}$
  end for
  for $i, j = 0, \dots, N$ do
    $(G_{E,\rho})_{i,j} \leftarrow \sum_{k=M+1}^{E} \langle (f)_{k-j}, (f)_{k-i} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2k}$
  end for
  $\vec{q} \leftarrow$ normalized eigenvector to smallest eigenvalue of $G_{E,\rho}$
  for $i = 1, \dots, M$ do
    $p_i \leftarrow \sum_{j=0}^{\min(N,i)} q_j (f)_{i-j,z_0}$
  end for
  $Q(z) \leftarrow \sum_{i=0}^{N} q_i (z - z_0)^i$
  $P(z) \leftarrow \sum_{i=0}^{M} p_i (z - z_0)^i$
  return Pade approximant $P(z)/Q(z)$
```

Remark. Looking at the entries of $G_{E,\rho}$, we can see that they are in fact weighted sums of entries of the Gram matrix $(G(f))_{ij} = \langle (f)_i, (f)_j \rangle_{V,\sqrt{Re(z_0)}}$. Therefore, one could also first construct the Gram matrix and then calculate the entries of $G_{E,\rho}$. Depending on the value of N this may be computationally cheaper, since each entry in $G_{E,\rho}$ is a sum of $N+1$ entries of $G(f)$. Of course both ways of calculating $G_{E,\rho}$ lead to the same result.
The parameter ρ attaches weight to the summands of $(G_{E,\rho})_{ij}$. Larger values of ρ attach more weight to the bottom right of $G(f)$ and smaller values of ρ attach more weight to the top left.

3 Multi point Padé approximation

As an extension to the single-point Padé approximation in the last chapter, we will construct in a similar way a multi point version. This means that instead of one center z_0, we will have multiple centers $z_0, \dots, z_k$. This approximation is also called Newton-Padé approximation. We introduce this extension, since we hope to get a larger convergence area by using multiple centers instead of using only one center.

3.1 Multi point Padé approximation in $\mathbb{C}$

First we want to motivate the approximation in V by defining the Newton-Padé approximation for $\mathbb{C}$-valued functions. The following definitions are from [Cla76], [Cla78] and [FL07]. Also [BGM96] provides some introduction into the multi point Padé.
We look at some holomorphic function $f : \mathbb{C} \to \mathbb{C}$ and have a set of point $\{z_i\}_{i=0}^{\infty} \subset \mathbb{C}$. We define divided differences in the following way.

Definition 3.1 (Divided Differences). Let be f and $\{z_i\}_{i=0}^{\infty}$ as above. Then we define the divided differences f_{ij} for $i \leq j$ as

$$f_{i,i} = f(z_i) \qquad i = j \tag{3.1}$$

$$f_{i,j} = \frac{f_{i+1,j} - f_{i,j-1}}{z_j - z_i} \qquad i < j \tag{3.2}$$

For $i > j$ we set $f_{i,j} := 0$.
In case we have $z_i = z_j$, then we have $f_{ij} = (f)_{z_i,j-i}$, where $(f)_{z_i,j-i}$ denotes the Taylor coefficient of f of degree $j-i$ in z_i (see [FL07]). Then we can write f in the following series expansion

$$f(z) = f_{0,0} + f_{0,1}(z - z_0) + f_{0,2}(z - z_0)(z - z_1) + \cdots .$$

To shorten the notation in the following we define recursively $w_{00}(z) = 1$ and $w_{0i}(z) = (z - z_{i-1})w_{0,i-1}$ and write

$$f(z) = \sum_{i=0}^{\infty} f_{0,i} w_{0i}(z),$$

since the set $\{w_{0,i}\}_{i \leq N}$ forms a basis of $\mathbb{P}_N$. This is called the Newton basis. In the following we are only interested in a finite expansion of f and therefore we restrict ourselves to some $(z_0, \dots, z_E)$. Now we try to find two polynomials $P(z) = \sum_{i=0}^{M} p_{0,i} w_{0i}(z)$ and $Q(z) = \sum_{i=0}^{N} q_{0,i} w_{0i}(z)$, which approximate f. Since $w_{0,i}$ is a polynomial of degree i, P and Q are polynomials of degree M respectively N.

Definition 3.2 (Multi point Padé approximation in $\mathbb{C}$). Let be $f \in \mathcal{H}(\mathbb{C}, \mathbb{C})$, $P \in \mathbb{P}_M(\mathbb{C})$ and $Q \in \mathbb{P}_N(\mathbb{C})$ such that $(Qf - P)_{0,i} = 0$ for $i = 0, \dots, m+n$. Then we call $P/Q(z)$ a multi point Padé or Newton-Padé approximation of f.

In a similar way as in the single point case, we formulate this approximation as a linear system of equations and will try to formulate a least-square minimization problem in the case when f is V-valued. Since we want $(Qf - P)_{0i} = 0$ for $i = 0, \dots, m+n$, the following equations have to be fulfilled:

$$\sum_{j=0}^{i} q_{0,j} f_{j,i} = p_{0,i} \qquad\qquad i = 0, 1, \dots, m$$

$$\sum_{j=0}^{i} q_{0,j} f_{j,i} = 0 \qquad\qquad i = m+1, m+2, \dots, m+n$$

In order to derive this two equations one uses the product formula for divided differences

$$(gh)_{i,j} = \sum_{k=i}^{j} (g)_{i,k} (h)_{k,j},$$

for two functions g, h and $i < j$ (see [FL07]).

3.2 Multi point Padé approximation in V

In order to generalize this definition of the Newton-Padé approximation to V-valued functions, we define like in the single-point case a functional, which has to be minimized. Instead of the Taylor coefficient we will use divided differences and therefore the functional is called d instead of j.

Definition 3.3 (Newton-Padé functional). Let V be a Hilbert space, $f : \mathbb{C} \to V$ a function, which is holomorphic around some points $\{z_i\}_{i=0}^{E} \subset \mathbb{C}$ and $\rho \in \mathbb{R}^+$ some positive parameter. We also have two polynomials $P \in \mathbb{P}_M(\mathbb{C}, V)$ and $Q \in \mathbb{P}_N(\mathbb{C})$ and a natural number $E \in \mathbb{N}$. Then we define

$$d_{E,\rho}(P, Q, \{z_i\}_{i=0}^{E}) := \left(\sum_{i=0}^{E} \| (Qf(z) - P(z))_{0,i} \|^2_{V, \sqrt{Re(z_0)}} \rho^{2i} \right)^{1/2}, \tag{3.3}$$

where the divided differences are defined in the points $\{z_i\}_{i=0}^{E}$.

Remark. In case all the z_i are the same (i.e. $z_0 = z_1 = \cdots = z_E$), this is the same functional as in the single point case.
The coefficients $p_{0,i}$ of P are again elements of the Hilbert space V and the coefficients $q_{0,i}$ of Q are complex numbers.

We conclude similar to the last chapter and define the Newton-Padé approximant as the minimum of the functional (3.3).

Definition 3.4 (Newton-Padé approximant in V). Let V, P, Q, F, ρ, E and $\{z_i\}_{i=0}^{E}$ be as in the previous definition. Additionally, we want $N + M \leq E$ and Q to be normalized, when they are represented in the Newton basis. This means if $Q = \sum_{i=0}^{N} Q_{0,i} w_{0,i}$, we want $\sum_{i=0}^{N} |Q_{0,i}|^2 = 1$. We also define $\mathbb{P}_N^{1,N} = \{S \in \mathbb{P}_N(\mathbb{C}) : \sum_{i=0}^{N} |Q_{0,i}|^2 = 1\}$ the space of polynomials of degree N which are normalized in the Newton basis. Then we say that P/Q is the Newton-Padé approximation of f in $\{z_i\}_{i=0}^{E}$, if

$$d_{E,\rho}(P, Q, \{z_i\}_{i=0}^{E}) = \inf_{\substack{R \in \mathbb{P}_M(\mathbb{C}, V),\\ S \in \mathbb{P}_N^{1,N}(\mathbb{C})}} d_{E,\rho}(R, S, \{z_i\}_{i=0}^{E}). \tag{3.4}$$

In order to prove that the multi point Padé approximation exists, we will continue in a similar way as in the single point case.

Theorem 3.5 (Existence of a Newton-Padé approximation). *The infimum in* (3.4) *always exists and is attained by at least one pair of polynomials (P,Q).*

Proof. The proof runs in a very similar way as in theorem 2.7. First we rewrite (3.3).

$$\begin{aligned} d_{E,\rho}^2(P, Q, \{z_i\}_{i=0}^{E}) &= \sum_{i=0}^{M} \|(Qf(z) - P(z))_{0,i}\|_{V,\sqrt{Re(z_0)}}^2 \rho^{2i} \\ &+ \sum_{i=M+1}^{E} \|(Qf(z) - P(z))_{0,i}\|_{V,\sqrt{Re(z_0)}}^2 \rho^{2i} \\ &= \sum_{i=0}^{M} \|(Qf(z) - P(z))_{0,i}\|_{V,\sqrt{Re(z_0)}}^2 \rho^{2i} \\ &+ \sum_{i=M+1}^{E} \|(Qf(z))_{0,i}\|_{V,\sqrt{Re(z_0)}}^2 \rho^{2i} \end{aligned}$$

Here we use that $P_{0,i} = 0$ for $i > M$. Now we set $P_{0,i} = \sum_{j=0}^{\min(N,i)} Q_{0,j} f_{j,i}$ and therefore the first summand is zero and P is uniquely determined by f and Q. We can again write it as a minimization problem only in Q

$$\tilde{d}_{E,\rho}(Q, \{z_i\}_{i=0}^{E}) = \inf_{S \in \mathbb{P}_N^{1,N}(\mathbb{C})} d_{E,\rho}(S, \{z_i\}_{i=0}^{E}), \tag{3.5}$$

where $\tilde{d}_{E,\rho}(Q, \{z_i\}_{i=0}^{E}) = \left(\sum_{i=M+1}^{E} \|(Qf(z))_{0,i}\|_{V,\sqrt{Re(z_0)}}^2 \rho^{2i}\right)^{1/2}$. Now we know that $\mathbb{P}_N^{1,N}$ is compact, since it is homeomorphic to the unit sphere in $\mathbb{C}^{N+1}$, and $\tilde{d}_{E,\rho}$ continuous and therefore we know that the infimum exists. Then we reconstruct P from Q and f and have a solution (P, Q). □

3.3 Algorithm

Now we conclude in a similar way as before to construct a matrix, which eigenvalues are the divided differences $Q_{0,i}$ of the polynomial Q. The following theorem is the version for multiple centers of theorem 2.10.

Theorem 3.6 (Calculation of the coefficients $Q_{0,i}$). *The minimization in* (3.5) *can be solved by calculating the normalized eigenvector of the smallest eigenvalue of the matrix* $G_{E,\rho}$ *with entries*

$$(G_{E,\rho})_{ij} = \sum_{l=M+1}^{E} \langle f_{i,l}, f_{j,l} \rangle^2_{V,\sqrt{Re(z_0)}}.$$

$G_{E,\rho}$ *is hermitian and positive semidefinite.*

Proof. The proof is again very similar to the one of theorem 2.10. One can calculate that $(Qf)_i = \sum_{j=0}^{i} Q_{0,j} f_{j,i}$ and therefore calculate

$$\begin{aligned}
\tilde{d}_{E,\rho}(Q, \{z_i\}_{i=0}^{E})^2 &= \sum_{l=M+1}^{E} \langle (Qf)_{0,l}, (Qf)_{0,l} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2l} \\
&= \sum_{l=M+1}^{E} \langle \sum_{i=0}^{l} Q_{0,i}(f)_{i,l}, \sum_{j=0}^{l} Q_{0,j}(f)_{j,l} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2l} \\
&= \sum_{i,j=0}^{N} Q^*_{0,i} Q_{0,j} \sum_{l=0}^{E} \langle (f)_{i,l}, (f)_{j,l} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2l} \\
&= \vec{q}^* G_{E,\rho} \vec{q},
\end{aligned}$$

where $\vec{q} = \{Q_{0,0}, Q_{0,1}, \ldots, Q_{0,N}\}$ denotes the vector containing the coefficients of Q in the Newton basis.Therefore minimizing $\tilde{d}_{E,\rho}(Q, \{z_i\}_{i=0}^{k})^2$ is equivalent to minimizing $\vec{q}^* G_{E,\rho} \vec{q}$ and this product gets minimal for the smallest eigenvector of $G_{E,\rho}$. The normalization is done by normalizing $\vec{q}$.
$G_{E,\rho}$ is hermitian by its definition. Since $\vec{q}^* G \vec{q} = \tilde{d}_{E,\rho}(Q)^2$ for all $q \in \mathbb{C}^{N+1}$, we can see with the definition of $\tilde{d}_{E,\rho}(\cdot)^2$, that $\vec{q}^* G \vec{q} \geq 0$ for all $q \in \mathbb{C}^{N+1}$. Therefore $G_{E\rho}$ is positive semidefinite. □

Now we have all necessary tools to calculate a Newton-Padé approximation. For the following algorithm we will use a slightly different notation for the $\{z_i\}$. We denote by $\hat{z} = (z_0, z_1, \ldots, z_{\hat{k}})$ a vector of all distinct z_i, i.e. $z_i \neq z_j$ for $i \neq j$. We have a second vector $dev = (dev\_z_0, \ldots, dev\_z_k) \subset \mathbb{N}_0^k$, which denotes how many evaluations (derivatives) we take for some z_i. $dev\_z_i > 1$ (i.e. a derivative is calculated for z_i) would mean in the notation from before that some z_i appeared multiple times and therefore divided difference and derivative are the same. Note that we need $\sum_{i=0}^{k} dev\_z_i = E + 1$ in order to have the right number of Taylor coefficients such that we can calculate the entries of $G_{E,\rho}$. Another point is that it is important to calculate the values $f_{i,j}$ in the correct order, since the definition is recursive. It is a good idea to save the values $f_{i,j}$ in a triangular matrix to make the calculation of $G_{E,\rho}$ more efficiently (as given in Algorithm 2). We do not say anything here about the ordering of the z_i. We will see in the numerical experiments in chapter 6 that this choice is very important. As in the single point case the following algorithm can be used for any meromorphic function $f : \mathbb{C} \to V$ and not only for our solution map.

Algorithm 2 Newton-Padé approximant

```
Require: some meromorphic function $f : \mathbb{C} \to V$ with $\Lambda$ being the set of singularities of
    $f$, $\hat{z} = (z_0, z_1, \ldots, z_k) \subset \mathbb{C}\backslash\Lambda, dev = (dev\_z_0, \ldots, dev\_z_k) \subset \mathbb{N}^k$ with $\sum_{i=0}^{k} dev\_z_i =$
    $E + 1, \rho \in \mathbb{R}^+, M, N, E \in \mathbb{N}$ with $M + N \leq E$
  for $z_i$ in $\hat{z}$ do
      for $\beta = 0, \cdots, dev\_z_i - 1$ do
          Calculate the Taylor coefficient $(f)_{z_i,\beta}$
      end for
  end for
  for $i = 0, 1, \ldots, E$ do
      for $j = i, i-1, \ldots, 0$ do
          Calculate $f_{i,j}$ according to definition 3.1
      end for
  end for
  for $i, j = 0, \ldots, N$ do
      $(G_{E,\rho})_{i,j} \leftarrow \sum_{l=M+1}^{E} \langle (f)_{i,l}, (f)_{j,l} \rangle_{V,\sqrt{Re(z_0)}} \rho^{2l}$
  end for
  $\vec{q} \leftarrow$ normalized eigenvector to smallest eigenvalue of $G_{E,\rho}$
  for $i = 1, \ldots, M$ do
      $p_i \leftarrow \sum_{j=0}^{\min(N,i)} q_j (f)_{j,\min(N,i)}$
  end for
  $Q(z) \leftarrow \sum_{i=0}^{N} q_i w_{0,i}$
  $P(z) \leftarrow \sum_{i=0}^{M} p_i w_{0,i}$
  return Pade approximant $P(z)/Q(z)$
```

Remark. If we compare the complexity of the Padé approximation (Algorithm 1) and of the Newton-Padé approximation (Algorithm 2), the algorithms differ in two aspects. The first difference is, that only in Algorithm 2, we have to build up an extra matrix from the calculated Taylor coefficients, which contains the divided differences. Note that the matrix $(f)_{ij}$ is only a triangular matrix, since the divided differences $(f)_{i,j}$ are zero for $j < i$.
Let now $Ax = b$ be the system of equations, which arises from modelling the PDE with Finite Elements. The second difference between the single and the multi point Padé is that we have to build up more matrices A in Algorithm 2 (one for each center). In Algorithm 1 we have to construct only one matrix A, make some decomposition of it and then solve the linear system multiple times for different right sides. This strategy is less effective here and may even be not good at all. This is the case, if we have only a few derivatives in each center and therefore only have to solve each linear system a few times, which could mean that the decomposition is even more expensive than using a direct solver.

As in the single point case, one could first build up a matrix $G(f)$ with entries $(G(f))_{i,j} = \langle (f)_{il}, (f)_{jl} \rangle_{V,\sqrt{Re(z_0)}}$ for the calculation of $G_{E,\rho}$ and use these entries for the summation in the calculation of $G_{E,\rho}$. This may save some computational time.
The parameter ρ attaches again weight to the entries of $G(f)$ in the summation of the calculation of $G_{E,\rho}$.

4 Model problems

4.1 Parametric interior Helmholtz problem

First we look at the Helmholtz equation with homogeneous Dirichlet boundary condition. Let $\Omega \subset \mathbb{R}^2$ be an open bounded Lipschitz domain, $f \in L^2(\Omega, \mathbb{C})$ and $\nu \in [\nu_{min}, \nu_{max}] \subset \mathbb{R}^+$ a wave number. We define the equation as

$$\begin{aligned} -\Delta u - \nu^2 u &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega. \end{aligned}$$

Like f, u may be complex valued. We derive the following weak formulation for a general wave number $z \in \mathbb{C}$

Problem 1. Given $f \in L^2(\Omega)$ find $u \in H_0^1(\Omega)$ such that $\forall v \in H_0^1(\Omega)$

$$\int_\Omega \nabla u \nabla \bar{v} dx - z \int_\Omega u\bar{v} dx = \int_\Omega f\bar{v} dx$$

holds.

We also define the corresponding bilinear form as

$$a_z(u, v) := \int_\Omega \nabla u \nabla \bar{v} dx - z \int_\Omega u\bar{v} dx.$$

As before, we define for $u \in H^1(\Omega, \mathbb{C})$ and $\nu > 0$ a weighted H^1-Norm with

$$\|u\|_{V,\nu}^2 = \|\nabla u\|_{L^2(\Omega)}^2 + \nu^2 \|u\|_{L^2(\Omega)}^2 .$$

This norm is equivalent to the standard H^1-norm, since for $c_1 := \sqrt{\min\{1, 1/\nu^2\}}, c_2 := \sqrt{\max\{1, 1/\nu^2\}}$ holds

$$c_1 \|u\|_{V,\nu} \leq \|u\|_{H^1} \leq c_2 \|u\|_{V,\nu} .$$

The next step will be to prove that Problem 1 admits a unique solution. Therefore, we will have a look at the following theorem from [BNP17].

Theorem 4.1. *Problem 1 admits a unique solution, if $z \in \mathbb{C}$ is not an eigenvalue of the Laplace operator Δ with Dirichlet boundary conditions. We denote the set of eigenvalues of Δ by Λ.*

Proof. In the proof we also follow [BNP17]. We will distinguish three different cases depending on the value of z.
Let us first look at the case where z has a negative real part, i.e. $z \in \mathbb{R}^- + i\mathbb{R}$. Here we will use the Lax-Milgram theorem to get existence and uniqueness of the solution. Therefore we have to show continuity and coercivity of a_z. For continuity we take $u, v \in H_0^1(\Omega, \mathbb{C})$ and conclude

$$\begin{aligned}|a_z(u,v)| &= |\int_\Omega \nabla u \nabla \bar{v} dx - z \int_\Omega u\bar{v} dx| \leq |\int_\Omega \nabla u \nabla \bar{v} dx| + |z \int_\Omega u \bar{v} dx| \\ &\leq \|\nabla u\|_{L^2} \|\nabla v\|_{L^2} + |z| \|u\|_{L^2} \|v\|_{L^2} \\ &= \|\nabla u\|_{L^2} \|\nabla v\|_{L^2} + \frac{|z|}{\nu^2} \nu^2 \|u\|_{L^2} \|v\|_{L^2} \\ &\leq \max\{1, |z|/\nu^2\} \|u\|_{V,\nu^2} \|v\|_{V,\nu^2}\end{aligned}$$

For coercivity we make the following estimate.

$$\begin{aligned}|a_z(u,u)| \geq |Re(a_z(u,u))| &\geq \int_\Omega |\nabla u|^2 dx - Re(z) \int_\Omega |u|^2 dx \\ &= \|\nabla u\|_{L^2}^2 + \frac{-Re(z)}{\nu^2} \|u\|_{L^2}^2 \nu^2 \geq \min\{1, \frac{-Re(z)}{\nu^2}\} \|u\|_{V,\nu^2}^2\end{aligned}$$

We also need to prove that the linear form $l(v) := \int_\Omega f v dx$ is continuous. We estimate using the Cauchy-Schwarz inequality

$$l(v) = \int_\Omega f v dx \leq \|f\|_{L^2(\Omega)} \|v\|_{L^2(\Omega)} \leq \|f\|_{V,\nu} \|v\|_{V,\nu} .$$

This concludes the proof in the first case.
In the second case we look at $z \in \mathbb{R}^+ + i\mathbb{R}$ with $\mathrm{Im}(z) \neq 0$. Again we will use the Lax Milgram theorem for the proof. Since we never used any assumptions in the continuity part in the first case, we can prove continuity of a_z here in the same way. The same holds for the continuity of $l(v)$. We have to use a different estimate for the coercivity of a_z. First note, that for $w \in \mathbb{C}$ holds

$$\begin{aligned}(|\mathrm{Re}(w)| + |\mathrm{Im}(w)|)^2 &= \mathrm{Re}(w)^2 + \mathrm{Im}(w)^2 + 2|\mathrm{Re}(w)||\mathrm{Im}(w)| \\ &\leq 2(\mathrm{Re}(w)^2 + \mathrm{Im}(w)^2) = 2|w|^2.\end{aligned}$$

Taking the square root on both sides we get $\sqrt{2}|w| \geq |\mathrm{Re}(w)| + |\mathrm{Im}(w)|$. Since $a_z(u,u) \in \mathbb{C}$, we have for $\varepsilon \in (0,1)$

$$\begin{aligned}\sqrt{2}|a_z(u,u)| &\geq \varepsilon \mathrm{Re}(a_z(u,u)) + |\mathrm{Im}(a_z(u,u))| \\ &= \varepsilon(\|\nabla u\|_{L^2}^2 - \mathrm{Re}(z) \|u\|_{L^2}^2) + |\mathrm{Im}(z)| \|u\|_{L^2}^2 \\ &= \varepsilon \|\nabla u\|_{L^2}^2 + \frac{|\mathrm{Im}(z)| - \varepsilon \mathrm{Re}(z)}{\nu^2} \|u\|_{L^2}^2 \nu^2 \\ &\geq \min\{\varepsilon, \frac{|\mathrm{Im}(z)| - \varepsilon \mathrm{Re}(z)}{\nu^2}\} \|u\|_{V,\nu}^2\end{aligned}$$

Now it is only left to proof that the coercivity constant is positive. Therefore, let $\varepsilon \in (0, \min\{1, \frac{|\mathrm{Im}(z)|}{\mathrm{Re}(z)}\})$. Then $0 < \varepsilon < 1$ still holds and we have that $\frac{|\mathrm{Im}(z)| - \varepsilon \mathrm{Re}(z)}{\nu^2} > 0$. Therefore a_z is coercive and we have a unique solution.
The only case left is $z \in \mathbb{R}^+ \backslash \Lambda$. Here we can apply another theorem from functional analysis, the Fredholm alternative. It says that we have either a solution $u \neq 0$ to the case $f = 0$ or $u = 0$ is the only solution to $f = 0$ and we have a unique solution for every f. Since we assumed that z is not an eigenvalue of Δ, we know that problem 1 with $f = 0$ has only the solution $u = 0$. Therefore, we have a unique solution for the inhomogeneous problem. □

Since we know that problem 1 has for any $z \in \mathbb{C} \backslash \Lambda$ a unique solution, we can define a solution map T, which maps for a fixed right hand side f and fixed boundary conditions a wave number z on its solution.

Definition 4.2 (Solution map). We define the solution map T in the following way:

$$\begin{aligned} T: \quad \mathbb{C} \backslash \Lambda &\to H_0^1(\Omega) \\ z &\mapsto u(z, \cdot) \end{aligned} \tag{4.1}$$

The next step will be to prove an equality for the norm of the solution to understand better the solution. This will be useful for our numerical experiments later. We will follow again the proof in [BNP17].

Lemma 4.3. *Let $z \in \mathbb{C} \backslash \Lambda$ and $u(z, x)$ the unique solution of problem 1. Let $\{\phi_l\}_{l \in \mathbb{N}}$ be the L^2-orthonormal basis of eigenfunctions of the Laplace operator corresponding to the eigenvalues in Λ. Then we have the following equality*

$$\|u(z, \cdot)\|_{V,\nu}^2 = \sum_l \frac{|f_l|^2}{|\lambda_l - z|^2} (\lambda_l + \nu^2) \, \|\phi_i\|_{L^2(\Omega)}^2 \, . \tag{4.2}$$

Proof. We can write $u(z, x) = \sum_{l \in \mathbb{N}} u_l(z) \phi_l(x)$ and $f(x) = \sum_{l \in \mathbb{N}} f_l \phi_l(x)$. Putting these expressions into the Helmholtz equation (problem 1) and choosing $v = \phi_i$ we get the following

$$\begin{aligned} \int_\Omega \nabla \sum_{l \in \mathbb{N}} u_l(z) \phi_l \nabla \overline{\phi_i} dx - z \int_\Omega \sum_{l \in \mathbb{N}} u_l(z) \phi_l \overline{\phi_i} dx &= \int_\Omega \sum_{l \in \mathbb{N}} f_l \phi_l \overline{\phi_i} dx \\ - \int_\Omega \Delta \left(\sum_{l \in \mathbb{N}} u_l(z) \phi_l \right) \overline{\phi_i} dx - z \int_\Omega \sum_{l \in \mathbb{N}} u_l(z) \phi_l \overline{\phi_i} dx &= \int_\Omega \sum_{l \in \mathbb{N}} f_l \phi_l \overline{\phi_i} dx \\ \sum_l u_l \lambda_l \int_\Omega \phi_l \overline{\phi_i} dx - z \sum_l u_l \int_\Omega \phi_l \overline{\phi_i} dx &= \sum_l f_l \int_\Omega \phi_l \overline{\phi_i} dx \\ u_i(\lambda_i - z) &= f_i \end{aligned}$$

Doing this derivation for all $i \in \mathbb{N}$, we get that

$$u_i = \frac{f_i}{\lambda_i - z}. \tag{4.3}$$

The next step is to derive a different expression for the norm of u.

$$\begin{aligned}\|u(z,\cdot)\|_{V,\nu}^2 &= \|\nabla u\|_{L^2(\Omega)}^2 + \nu^2 \|u\|_{L^2(\Omega)}^2 \\ &= \int_\Omega |\sum_{l\in\mathbb{N}} u_l(z)\nabla\phi_l(x)|^2 dx + \nu^2 \int_\Omega |\sum_{l\in\mathbb{N}} u_l(z)\phi_l(x)|^2 dx \end{aligned} \tag{4.4}$$

Since we know that the $(\phi_i)_{i\in\mathbb{N}}$ are orthonormal in $L^2(\Omega)$ (4.4) equals

$$\begin{aligned}&\sum_l |u_l|^2 \int |\nabla\phi_l|^2 dx + \sum_l \nu^2 |u_l|^2 \int_\Omega |\phi_l|^2 dx \\ =&\sum_l |u_l|^2(\lambda_l+\nu^2)\|\phi_l\|_{L^2(\Omega)}^2 .\end{aligned}$$

Using 4.3 we get

$$\|u(z,\cdot)\|_{V,\nu}^2 = \sum_l \frac{|f_l|^2}{|\lambda_l - z|^2}(\lambda_l+\nu^2)\|\phi_i\|_{L^2(\Omega)}^2 .$$

□

Remark. Looking again at (4.2), we now better understand why the solution map T is only defined for $\mathbb{C}\backslash\Lambda$ and not for all complex numbers. Using (4.2), we get for $z \to \lambda_i$ for any i that $\|u(z,\cdot)\|_{V,\nu} \to \infty$. We know for the Laplace operator that all eigenvalues are on the positive real axis. This will be later important when we construct the Padé approximation.

[BNP17, Proposition 3.1] shows that the previously defined solution map T is continuous in V equipped with the norm $\|\cdot\|_{V,\nu}$.

Lemma 4.4. *The solution map T, which was defined in* (4.1)*, is continuous in the space $(V, \|\cdot\|_{V,\nu})$.*

The proof can be found in [BNP17].
In the first step of the Padé algorithm the Taylor coefficients of the solution map T have to be calculated at z_0. We will denote the i-th coefficient by $(T)_{z_0,i}$. The next theorem from [BNP17] shows us how we can calculate the coefficients.

Theorem 4.5. *The solution map T admits a complex derivative $T^{(i)}$, which is the unique solution of the following equation,*

$$\int_\Omega \nabla T^{(i)}\nabla\bar{v}dx - z_0\int_\Omega T^{(i)}\bar{v}dx = \int_\Omega T^{(i-1)}\bar{v}dx, \tag{4.5}$$

for all $v \in H_0^1(\Omega)$.

Proof. First, we note that by Theorem 4.1, we know that (4.5) has a unique solution, since $T^{(i-1)}$ is for all i a L^2-function.

To prove that $T^{(i)}$ is the complex derivative of $T^{(i-1)}$, we have a look at the difference quotient

$$T^{(1)} = \frac{dT}{dz} = \lim_{h\to 0} \frac{u(z+h,\cdot) - u(z,\cdot)}{h} =: \lim_{h\to 0} w_h(z,\cdot).$$

We know that

$$\int_\Omega \nabla u(z+h,x)\nabla\overline{v(x)}dx - (z+h)\int_\Omega u(z+h,x)\overline{v(x)}dx = \int_\Omega f(x)\overline{v(x)}dx \tag{4.6}$$

and

$$\int_\Omega \nabla u(z,x)\nabla\overline{v(x)}dx - z\int_\Omega u(z,x)\overline{v(x)}dx = \int_\Omega f(x)\overline{v(x)}dx \tag{4.7}$$

hold. Now we take the difference of the weak formulation in $z+h$ and z.

$$\begin{aligned}0 = &\int_\Omega \nabla u(z+h,x)\nabla\overline{v(x)}dx - (z+h)\int_\Omega u(z+h,x)\overline{v(x)}dx\\ &- \int_\Omega \nabla u(z,x)\nabla\overline{v(x)}dx + z\int_\Omega u(z,x)\overline{v(x)}dx\\ =& \int_\Omega \nabla(u(z+h,x)-u(z,x))\nabla\overline{v(x)}dx - z\int_\Omega (u(z+h,x)-u(z,x))\overline{v(x)}dx\\ &- h\int_\Omega u(z+h,x)\overline{v(x)}dx\\ =& h\int_\Omega \nabla w_h(z,x)\nabla\overline{v(x)}dx - zh\int_\Omega w_h(z,x)\overline{v(x)}dx - h\int_\Omega u(z+h,x)\overline{v(x)}dx.\end{aligned}$$

Dividing the last equation by h gives

$$\int_\Omega \nabla w_h(z,x)\nabla\overline{v(x)}dx - z\int_\Omega w_h(z,x)\overline{v(x)}dx = \int_\Omega u(z+h,x)\overline{v(x)}dx.$$

Taking the limit $h \to 0$ and using the continuity from lemma 4.4 we get that $T^{(1)}$ is the unique the solution of

$$\int_\Omega \nabla T^{(1)}\nabla\overline{v(x)}dx - z\int_\Omega T^{(1)}\overline{v(x)}dx = \int_\Omega u(z,x)\overline{v(x)}dx.$$

We can do the same derivation inductively for any higher i and therefore equation 4.5 holds. □

Remark. This lemma is very useful for us for two reasons. First, we now know how to calculate derivatives of the solution map T. We can do this by using again a finite-element method (as for $T(z_0)$) and the operator on the left hand side (for matrix A in system of equations) is always the same and therefore the stiffness matrix and the mass matrix do not have to be recalculated for every Taylor coefficient.
We can also see that T is holomorphic in $\mathbb{C}\backslash\Lambda$. In order to see that T is meromorphic on

$\mathbb{C}$ (which we will need for the Padè approximation), we look at $T(z)$ as the sum of the L^2-eigenfunctions as in the proof of theorem 4.3. We know that

$$T(z) = u(z, \cdot) = \sum_l \frac{f_l}{\lambda_l - z}\phi_l.$$

Since the multiplicity of every eigenvalue λ_j is finite, each factor $1/(\lambda_j - z)$ appears only finitely many times in the sum (once for each eigenfunction corresponding to the eigenvalue) and therefore the map is meromorphic.

We can now apply our theory on single-point Padé approximation to this map T. Since we will be interested in some interval of frequencies $K = [k_{min}, k_{max}] \subset \mathbb{R}$, we need to find a good center z_0 to cover that interval with our Padé map. Another important point is that we need $z_0 \notin \mathbb{R}$, since the singularities of T are real valued and we need that T is holomorphic in a neighbourhood of z_0. For this two reasons $z_0 = \frac{k_{max}+k_{min}}{2} + \delta i$ for some $\delta \in \mathbb{R}$ is an obvious choice. In order to have good convergence on the real axis, we do not want δ to be too large. We will use $\delta = 0.5$ in the numerical experiments later, which worked well and is also used in [BNP17] and [BNPP18].

4.2 Scattering problem

This chapter is about the second problem, which we will use for our numerical tests. Again we will present some important theoretical results, which are needed to construct the Padè approximation. In addition to the first problem we will introduce a circle on which the travelling wave gets scattered. In the whole chapter we will follow [BNPP18, 5].
Let $\Omega = (-\pi, \pi)^2 \backslash \mathcal{B}((0,0), 0.5) \subset \mathbb{R}^2$ be our domain, where $\mathcal{B}((0,0), 0.5)$ denotes the circle with center in $(0,0)$ and radius 0.5. We denote the outside boundary by $\Gamma_R = \partial[-\pi, \pi]^2$ and the inner boundary by $\Gamma_D = \partial\mathcal{B}((0,0), 0.5)$. We have some incident wave $u_i = \exp(ik\vec{d} \cdot \vec{x})$, where k is our wave number and $\vec{d} \in \mathbb{R}^2$ is the travelling direction of the wave. We define k, such that $k^2 \in K := [k_{min}^2, k_{max}^2] \subset \mathbb{R}$ holds. We denote by n the outgoing normal vector from Γ_R and set $g_k := \frac{\partial u_i}{\partial n} - iku_i$. Our solution u will consist of two parts, the previously defined incident wave u_i and the scattered part u_{scat}. The sum $u = u_i + u_{scat}$ has to solve the following differential equation and boundary conditions.

$$\begin{aligned} -\Delta u - k^2 u &= 0 && \text{in } \Omega \\ u &= 0 && \text{on } \Gamma_D \\ \frac{\partial u}{\partial n} - iku &= g_k && \text{on } \Gamma_R \end{aligned} \tag{4.8}$$

This PDE with boundary condition describes a model, where the wave u_i travels along the direction $\vec{d}$ and the wave gets scattered at the circle in the center. The boundary condition on the outer boundary is an approximation of the Sommerfeld radiation, which tries to simulate the fact that the wave should scatter to infinity (or in our case the boundary) and

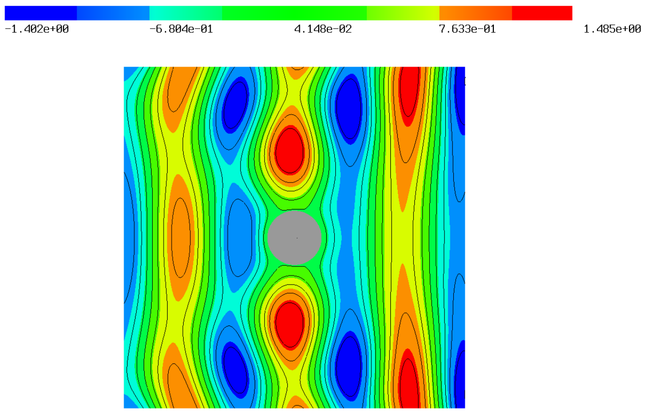


Figure 4.1: Real part of the $\mathbb{P}^3$ finite element solution in $z = 3$

not be reflected from there. In Figure 4.1 we can see the wave travelling from left to right and that there is some "shadow" behind the circle, where the wave is fluctuating much less.

The first step is to derive a weak formulation for the problem. We set $V := H^1_{\Gamma_D}(\Omega) = \{v \in H^1(\Omega) : u|_{\Gamma_D} = 0\}$, where $u|_{\Gamma_D}$ denotes the trace of u on Γ_D (as defined e.g. in [Eva10]). Since we want to construct a Padé approximation on a part of the complex plane later on, we will define the problem also for complex wave numbers.

Problem 2. Given $z \in \mathbb{C}$ find $u_z \in V$ such that for all $v \in V$

$$\int_\Omega \nabla u_z(x) \cdot \nabla \overline{v(x)} dx - z^2 \int_\Omega u_z(x) \overline{v(x)} dx - iz \int_{\Gamma_R} u_z(x) \overline{v(x)} dS$$
$$= \int_{\Gamma_R} g_z(x) \overline{v(x)} dS.$$

We can also define the solution map in a similar way as before. We set $\mathbb{C}^+ := \{z \in \mathbb{C} : \mathrm{Im}(z) \geq 0\}$ and define

$$\begin{aligned} T_s : \quad \mathbb{C}^+ &\to H^1_{\Gamma_D}(\Omega) \\ z &\mapsto u(z, \cdot), \end{aligned} \tag{4.9}$$

in such a way that $u(z, x)$ solves problem 2. We define T_s only on $\mathbb{C}^+$, since we know that there exists a unique solution on all compact subsets (see [BNPP18, Theorem 5.1]). [BNPP18, Theorem 5.3] shows, that T is meromorphic in all open bounded and connected subsets of $\mathbb{C}$ and that all its singularities have negative imaginary part. This is important to know in order to be able to apply our Padé algorithm. Next, we have to find a formula to calculate the Taylor coefficients of T_s for the scattering problem.

Lemma 4.6. *The solution map for the scattering problem T_s admits a complex derivative*

$T_s^{(i)}$, which is the unique solution of the following equation.

$$\begin{aligned}&\int_\Omega \nabla \frac{d^j T_s}{dz^j} \nabla \bar{v} dx - z^2 \int_\Omega \frac{d^j T_s}{dz^j} \bar{v} dx - iz \int_{\Gamma_R} \frac{d^j T_s}{dz^j} \bar{v} dS \\ =& j(j-1) \int_\Omega \frac{d^{j-2} T_s}{dz^{j-2}} \bar{v} dx + ji \int_{\Gamma_R} \frac{d^{j-1} T_s}{dz^{j-1}} \bar{v} dS + 2jz \int_\Omega \frac{d^{j-1} T_s}{dz^{j-1}} \bar{v} dx \\ &+ \int_{\Gamma_R} \nabla (u_i (id \cdot x)^j) \bar{v} dS - \int_{\Gamma_R} i^j (d \cdot x)^{j-1} u_i (zid \cdot x + j) \bar{v} dS,\end{aligned}$$

holds for every $v \in H^1_{\Gamma_R}(\Omega)$.

Proof. The formula can be derived by simply taking the derivative of problem 2 with respect to z. It is important to note that not only u_z, but also g_z depends on z. □

Remark. As for the Helmholtz equation the operator on the left hand side is the same for each $j \in \mathbb{N}$. The right hand side can again be computed using Taylor coefficients of lower degree. In contrast to the Helmholtz equation we do not only need here $T_s^{(j-1)}$, but also $T_s^{(j-2)}$. The fact that the operator on the left hand side does not change is very useful since we do not have to change the stiffness and mass matrix, when calculating the Taylor coefficients with a FEM, but only have to solve the same linear system with different right hand sides b. This can be used to decrease the computational effort, if one has to calculate many derivatives by using some decomposition to solve the linear system, e.g. a Cholesky or a LU decomposition.

5 Numerical results for the single point Padé approximation

In this chapter we will have a look at results from the single point Padé algorithm presented in Algorithm 1. We will look at results from the interior Helmholtz problem with Dirichlet boundary conditions and the scattering problem.

5.1 Remarks on the implementation

All the numerical experiments in the following were done using the software Netgen/NGSolve and its python interface, which is a free software library which can be used to solve PDEs using the Finite-Element-method(FEM) (see [ngs18]). A very informative introduction into this, which is also the basis of the code used here, are the itutorials (see [itu18]).

Basically we implemented the Padé code as given in Algorithm 1 in Python but used the interface to NGSolve for calculations like solving PDEs or calculating norms (i.e. evaluating integrals numerically). In the following we will describe some important parts of the code in more details.

The first step in NGSolve is defining some domain with some mesh and on this mesh a Finite-Element(FE) space. The mesh consists of triangles and has a parameter h_{max}, which denotes the maximum diameter of a triangle of the mesh. When defining the Finite-Element space one has to define some parameter p, which denotes the maximum degree of the polynomials on the FE space. Since these parameters determine basically the accuracy of the solution of some PDE, we will state them in the following calculations.

A more detailed description of the implementation and the most important parts of the code are given in the appendix at the end of the thesis.

5.2 Parametric interior Helmholtz problem

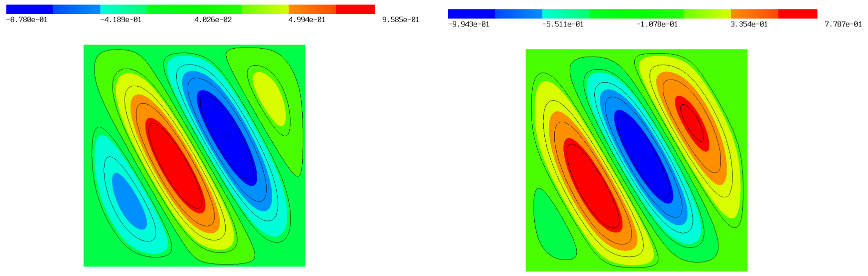
First we want to look at the results from the Helmholtz equation. This means we look for a solution to the following problem (see Problem 1).

$$\begin{aligned} -\Delta u - zu &= f && \text{in } \Omega \\ u &= 0 && \text{on } \partial\Omega \end{aligned}$$

The following example is taken from [BNP17, chapter7]. We fix the domain $\Omega = [0,\pi]^2$ and the right hand side

$$f(\vec{x}) = \frac{16}{\pi^4}\exp(-\nu i\vec{x}\cdot\vec{d})[2i\nu d_1(2x_1x_2^2 - 2\pi x_1x_2 - \pi x_2^2 + \pi^2 x_2)$$
$$+2i\nu d_2(2x_1^2x_2 - \pi x_1^2 - 2\pi x_1x_2 + \pi^2 x_1) - (2x_1^2 - 2x_2\pi + 2x_2^2 - 2x_1\pi)],$$

where $\vec{d} = (d_1, d_2)$ is the direction of the waves. We fix here $\vec{d} = (\cos(\pi/6), \sin(\pi/6))$. This right hand side was choose such that the analytical solution is the product of a bubble $w(x) = \frac{16}{\pi^4}x_1x_2(x_1-\pi)(x_2-\pi)$ and a plane wave $v(x) = \exp(-i\nu\vec{x}\cdot\vec{d})$, which is travelling in the direction $\vec{d}$. For the following calculations we will also fix $\nu = \sqrt{12}$ (to calculate the right hand side f). In Figure 5.1 we can see the solution for $z = 12$. The equation describes a wave travelling in the direction $\vec{d}$, where the boundaries are kept at zero.



(a) real part

(b) imaginary part

Figure 5.1: $\mathbb{P}^3$ finite element solution in $\nu^2 = 12$

Let us denote in the following the $\mathbb{P}_3$-FEM solution by u_h and the Padé approximation by $u_{P,h}$. $z_0 \in \mathbb{C}$ will always denote the point, where the Padé is centered and $z \in \mathbb{C}$ the point where u_h and $u_{P,h}$ are compared. Whenever a norm is taken, it is the $\|\cdot\|_{V,\sqrt{Re(z_0)}}$-norm and the relative error is always calculated by

$$\text{relative error} = \frac{\|u_h - u_{P,h}\|_{V,\sqrt{Re(z_0)}}}{\|u_h\|_{V,\sqrt{Re(z_0)}}}.$$

Some of the experiments with the Helmholtz problems were already done in [BNP17] and are reproduced here.

We will run two different types of tests. First we will compare along an interval of wave numbers the Padé approximant with a solution, which we will calculate by using a normal Finite-Element-method (FEM). Since the Taylor coefficients are also computed via the same FEM, we can compare how well the Padé approximant approximates here a function.

The second test will check the convergence order when we compare FEM against Padé for some fixed z and z_0 and increase the degree M of P. We will compare this convergence also with the convergence of a simple Taylor approximation to check when it is worth in practise to calculate the more expensive Padé approximation. Another interesting aspect

will be to check how we can influence the accuracy of the Padé approximation by increasing the accuracy of the FEM, which calculates the input data (Taylor coefficients).
All the calculation in NGSolve in this section are done with polynomial degree 3 and a maximum mesh size of 0.02, if not stated differently. This affects solving equations with FEM (for comparison for the relative error and calculation of the Taylor coefficients) and also calculating the integrals for the norm, which we use for example for the entries of G or the relative error.

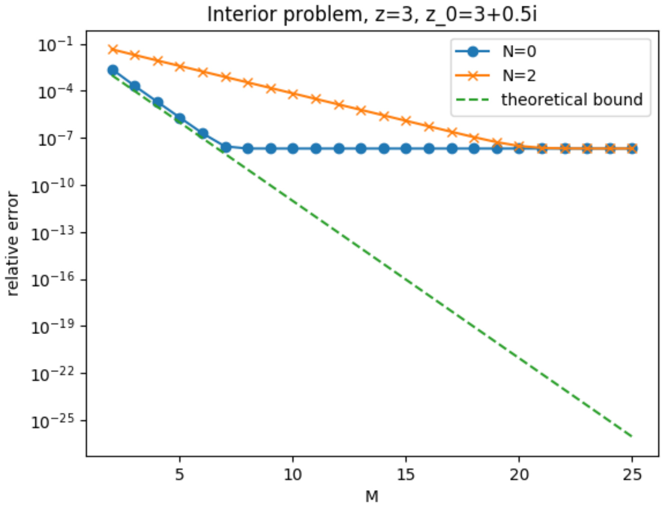


Figure 5.2: Center $z_0 = 3 + 0.5i$, approximation in $z = 3$, M growing, $N = 2$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{M+1}$ we have $\rho = |3 + 0.5i - 3|$ and $R = |3 + 0.5i - 8|$

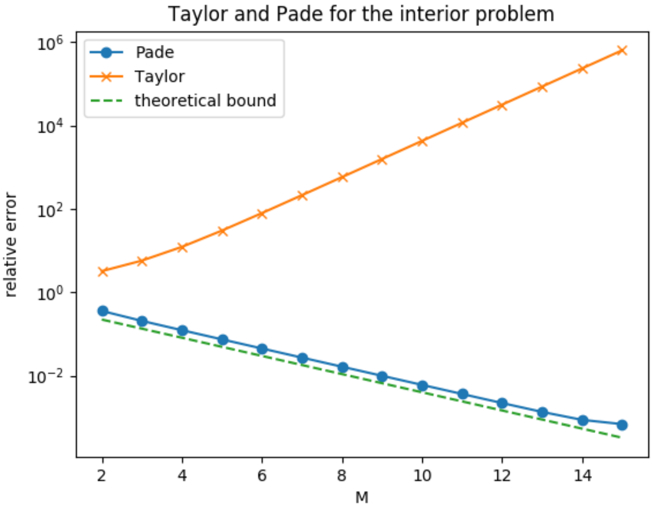


Figure 5.3: Center $z_0 = 3 + 0.5i$, approximation in $z = 6$, M growing, $N = 2$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{M+1}$ we have $\rho = |3 + 0.5i - 6|$ and $R = |3 + 0.5i - 8|$

In Figure 5.2 and Figure 5.3 we compared the convergence of the Padé approximation with the theoretical bound from theorem 2.9, which is $\left(\frac{\rho}{R}\right)^{M+1}$. Remember that we have used $\rho = |z - z_0|$ and R is chosen, such that the Padé approximation is meromorphic on $B(z_0, R)$. Therefore, we have to check how many poles of T can be resolved by Q. Since we have $N = 2$ here, we know that two poles can be resolved. We know that all Dirichlet eigenvalues of the Laplacian in two dimension on the square $[0, \pi]^2$ can be calculated with the formula $m^2 + n^2$, where $m, n \in \mathbb{N} := \{1, 2, 3, \dots\}$(see [KS84]). The two nearest eigenvalues are 2 and 5 and therefore 8 has to be boundary of $B(z_0, R)$.

In Figure 5.2 and Figure 5.3 we can also see in which situation the Padé approximation has a clear advantage and in which situations also a Taylor approximation can be used. While in Figure 5.2 the Taylor approximation can also deliver good results, in Figure 5.3 only the Pade works. In order to understand the difference we have to look at the eigenvalues of the Laplacian, which are the singularities of the solution function. While there are no singularities between 3 and $3 + 0.5i$, we have a singularity at 5, which disturbs the convergence of Taylor in $z = 6$. Since the Pade approximation consists also of a polynomial in the denominator, it can deal with this and therefore also deliver good results in $z = 6$.

We can see as well that the theoretical bound from Theorem 2.9 is achieved in Figure 5.2 and Figure 5.3 for the Padé approximant. In Figure 5.2 this is only the case as long as the FE error is not to large and the convergence stops.

Another interesting aspect, which we can see in Figure 5.2, is that for both the Taylor and the Padé approximation there seems to be some barrier for the accuracy of the relative error. Therefore, we have some M_{max}, where it is not worth to increase M any more. Looking at Figure 5.4, we can see that one of the factors influencing this maximal accuracy of the Padé is the accuracy of the input data. Here the example of Figure 5.2 is done with different mesh sizes, when calculating the input data with $\mathbb{P}_3$-FEM. While with a maximum mesh size of $h_{max} = 0.02$ we can reach a relative error of around 10^{-8}, with a much rougher mesh ($max_h = 0.16$), we can only get a relative error of around 10^{-3}. This error is due to the error of the results of the Finite-Element method. This error is not produced by the Padé approximation, since the Padé approximation is an approximation of the FE solution. This shows that for an accurate Padé approximation one not only needs high degrees for the polynomials P and Q, but also good accuracy when calculating the input data.

We did not only check the convergence in one point, but also the accuracy on an intervall of frequencies. In Figure 5.5 we plotted the $\|\cdot\|_{V,\sqrt{Re(z_0)}}$-norm of the $\mathbb{P}_3$-FEM solution and the Padé approximation centered in $z_0 = 9 + 0.5i$ with parameters $M = 4$ and $N = 2$. One can see that the Padé approximation is capable of reproducing the norm on a whole intervall accurately.

5.2.1 Numerical results in 3D

One can also run the code on a three dimensional example. The only difference is, that instead of the the spline geometry in 2D, we use Constructive Solid Geometry (CSG), which is also part of the Netgen package, to build our 3D domain. The idea is to use standard geometries in 3D (e.g. half-spaces, spheres,...) to build up more complex domains. We will use 6 half-spaces to build up a cube, which, of course, is the 3D extension of the

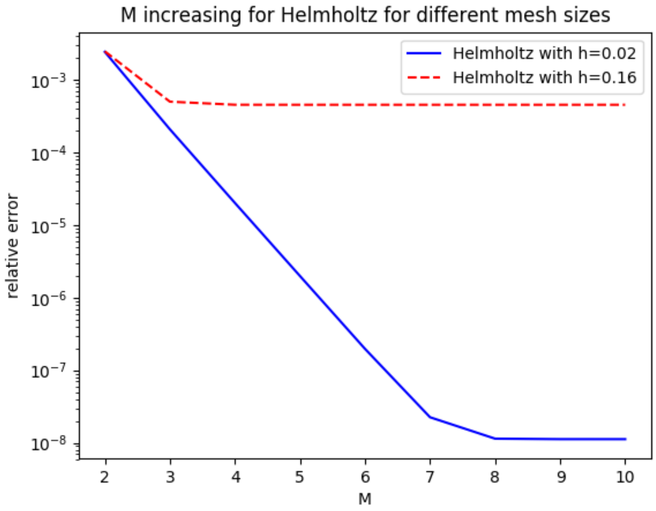


Figure 5.4: Comparison of different mesh sizes, Center $z_0 = 3 + 0.5i$, approximation in $z = 3$

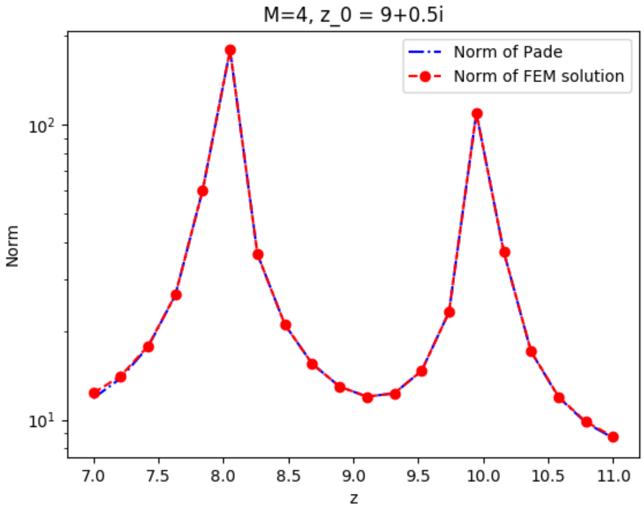


Figure 5.5: Comparison of FEM solution and Padé approximation centered in $z_0 = 9 + 0.5i$ with parameters $M = 4$ and $N = 2$ in 2D

square in 2D. One advantage of NGSolve is that the rest of the code does not depend on the dimension at the beginning. So we just have to construct the right geometry at the beginning and can reuse the rest of the code.
In the code we used the cube $[0, \pi]^3$. We did not change the right hand side f, i.e. f does not depend on the third coordinate. In order to keep the computation time low, the mesh is a bit rougher than in 2D ($h_{max} = 0.6$). Then we ran again the Padé code and compared it to the FEM solution of this problem. In Figure 5.6 we compared the FEM solution with the Padé approximation centered in $9 + 0.5i, 10 + 0.5i$ and $11 + 0.5i$. As parameters we used $M = 4$ and $N = 2$.

One interesting aspect one can see at this example is that the center should not only be not exactly at a singularity, but should be also far enough away. If one would take the center in $z_0 = 9 + 0.5i$, one gets to much "information" of the singularity in 9 into the Padé approximation and there is not enough "information" about the singularity in 11. Therefore this second singularity cannot be resolved. In Figure 5.6 we can see, that we have the same for problem vice versa for $z_0 = 11 + 0.5i$. Still for $M = 4$ and $N = 4$ we can approximate the norm on the interval well for $z_0 = 9 + 0.5i$, as we can observe in the first plot of Figure 5.7, but we have worse results for higher M and the same N due to numerical instability and a higher condition number of G. We can also see that when looking at the roots of Q for growing M and $N = 4$ in Table 5.1. There are the roots for some values of M. While for $M = 4$ there are still roots at around $7 + i$ and $12 + i$, for higher M the real parts of the roots are moving to the center and for $M \geq 11$ all four roots have a real part of around 9. The root at around $12 + i$ already disappears for $M \geq 7$ (largest real part at around 9.4 for $M = 7$).

M	roots of Q rounded to 2 digits
4	$6.95 + 1.01i$, $8.99 + i$, $8.99 + i$, $12.05 + 1.03i$
6	$7 + 1.02i$, $8.99 + i$, $8.99 + i$, $12 + 1.2i$
7	$7 + 1.22i$, $8.99 + i$, $8.99 + i$, $9.41 + 1.76i$
8	$6.9 + 1.12i$, $9 + i$, $8.99 + i$, $9 + 1.02i$
10	$7.03 + 1.09i$, $8.99 + i$, $8.99 + i$, $8.99 + i$
11	$8.99 + i$, $8.99 + i$, $8.99 + i$, $9.02 - 0.91i$

Table 5.1: Roots of Q for Padé centered in $z_0 = 9 + 0.5i$ for $N = 4$ and M growing

As we can also see in Figure 5.6 that the center $z_0 = 10 + 0.5i$ seems to be a good choice here, since it has the same distance to both singularities. Of course in a real-world example one would not know the singularities in advance and therefore would have either to find them with numerical experiments or a theoretical analysis or find a different strategy on how to place the centers. We can also see that the middle of the interval of the real parts $z_0 = 9.5 + 0.5i$ is a good choice for the center.
We can also see that with the Padé approximation and the right choice of the center, we can calculate again the Laplace eigenvalues. For this domain the eigenvalues are at 9 and 11 in the intervall $(7, 12)$. As in the 2D problem we are able to achieve a relatively high accuracy on the whole domain with the right choice of parameters.

In Figure 5.8 we checked the convergence of the relative error for the center $z_0 = 10 + 0.5i$

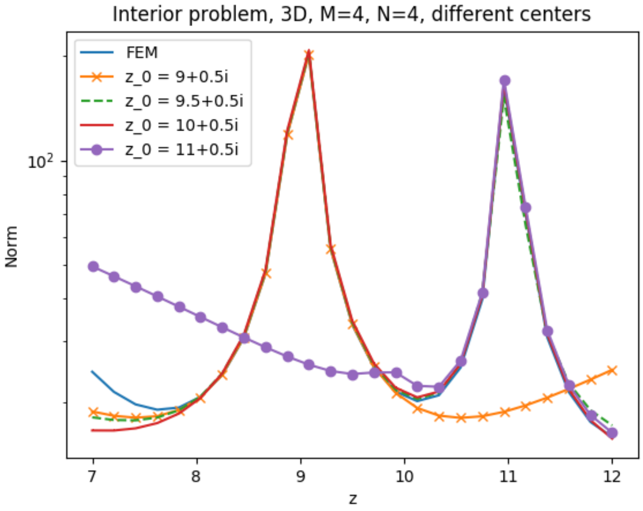
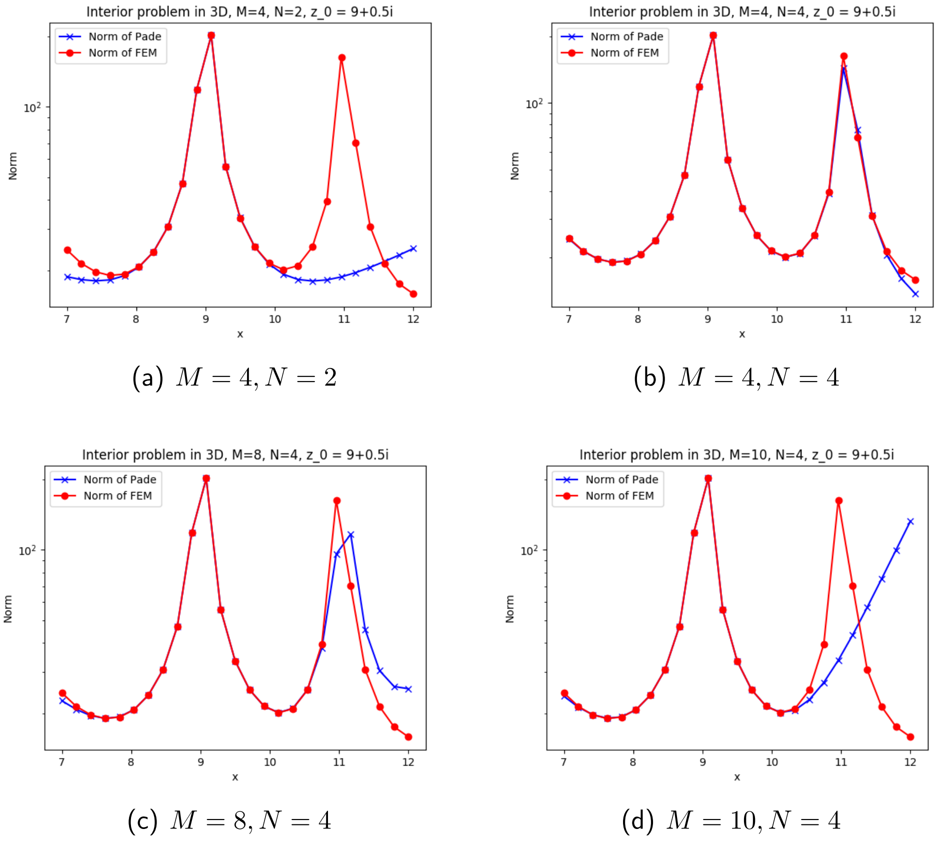


Figure 5.6: Comparison of the FEM solution with the Padé centered in $9 + 0.5i, 10 + 0.5i$ and $11 + 0.5i$ in 3D



(a) $M = 4, N = 2$

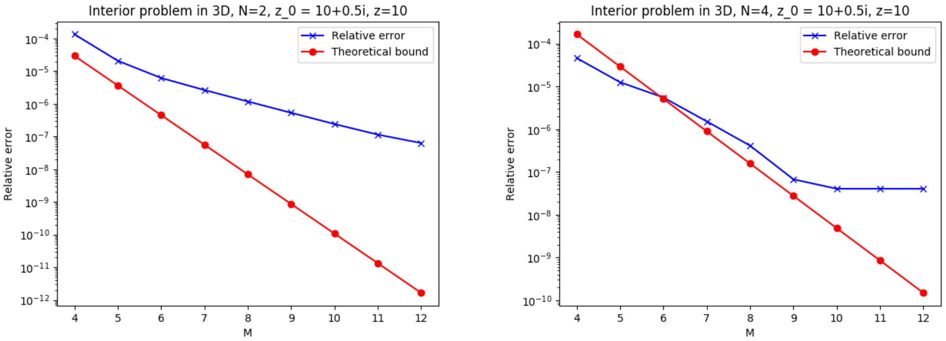
(b) $M = 4, N = 4$

(c) $M = 8, N = 4$

(d) $M = 10, N = 4$

Figure 5.7: Comparison of the FEM solution with the Padé centered in $9 + 0.5i$ in 3D - we have problems to approximate the singularity at 11 well

in $z = 10$. We can see, that in order to reach the theoretical bound from Theorem 2.9, we need a polynomial degree of $N = 4$ for the denumerator. For $N = 2$ we also have convergence towards the FE-solution, but we cannot reach the theoretical bound. The fact that the convergence stops for $N = 4$ for M larger than 10 is again due to the error of the FEM as discussed before in two dimensions.



(a) N=2

(b) N=4

Figure 5.8: Convergence of the Padé approximation for the interior Helmholtz problem in 3 dimensions for growing M for $N = 2, 4$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{(M+1)}$ we have $\rho = |10 + 0.5i - 10|$ and $R = |10 + 0.5i - 14|$

5.3 Scattering problem

In this chapter we want to present the results for the scattering equation. We presented the most important theoretical results in chapter 4.2.

In all the following examples we will set the direction of the wave as $\vec{d} = (1, 0)$, which means that the wave is travelling in the direction of the x-axis. We will do similar experiments as in the previous section for the Helmholtz equation. Most of the settings of the numerical experiments are the same as in [BNPP18, chapter 5].

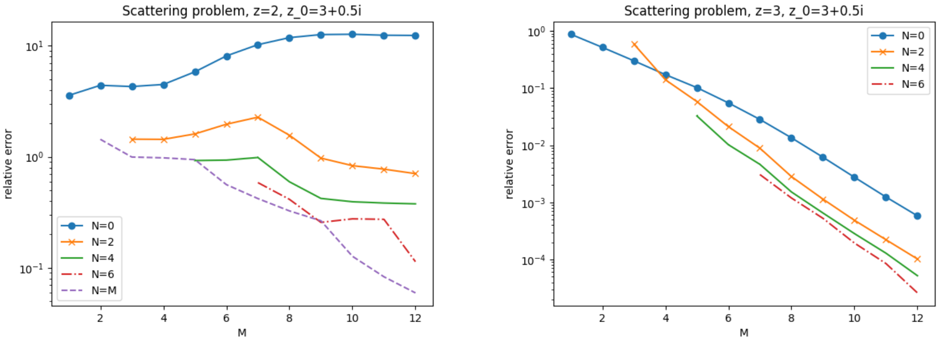
Recall that the singularities of T have negative imaginary part. Therefore, we should choose the center in $\mathbb{C}^+$, since we do not want our center to be equal to a singularity of T.

For the scattering problem, we will use the $\|\cdot\|_{V,Re(z_0)}$-norm and the relative error is then always calculated by

$$\text{relative error} = \frac{\|u_h - u_{P,h}\|_{V,Re(z_0)}}{\|u_h\|_{V,Re(z_0)}}.$$

In this way the results are evaluated in the same way as in [BNPP18]. The parameters for the FEM are always $p = 3$ as a polynomial degree and $h = 0.02$ for the mesh size, if not stated differently. The scattering problem needs a finer mesh than the last problem to be solved accurately enough.

A first big practical difference to the interior Helmholtz problem is that the convergence gets worse a lot quicker, if we have a larger distance to the center. One can see that looking at the next two examples, where the center is in both plots $z_0 = 3 + 0.5i$, but once we look at the convergence in $z = 2$ and once in $z = 3$ (Figure 5.9).



(a) evaluated in $z = 2$

(b) evaluated in $z = 3$

Figure 5.9: Convergence for the scattering problem with center $z_0 = 3 + 0.5i$ evaluated in $z = 2$ and $z = 3$

Looking at this two plots we can see an exponential convergence in $z = 3$ for different values of N. Already for $N = 2$ we have a good convergence when M is increasing and it does not seem to be worth investing computational time in a higher N, if the center is so close. Even the normal Taylor expansion gives us quite accurate results. A theoretical explanation for this is that the disk $B(z_0, |z - z_0|) = B(3 + 0.5i, 0.5)$ is completely in $\mathbb{C}^+$ and therefore does not contain any singularities. Therefore we do not need a high N to resolve singularities.

However it is a completely different situation in the first plot of Figure 5.9. Here we evaluate the same Padé approximation (centered in $z_0 = 3 + 0.5i$) in $z = 2$. Now the disk $B(z_0, |z - z_0|) = B(3 + 0.5i, \sqrt{1.25})$ is bigger and also contains points with negative imaginary part. We tried again the convergence for $N = 0, 2, 4, 6$ and also $N = M$. We can clearly see that the value of N has a huge influence on the accuracy. The biggest difference to the last example is for the normal Taylor expansion $(N = 0)$, where the relative error even diverges. We get the best convergence for $N = M$, which is called diagonal Padé. It is the only one of the four choices of parameters, where the error always decreases for higher M. The big drawback is of course that it is by fare the most expensive one to calculate (for the same M), since we need to calculate many Taylor coefficients. In the end one has to compare relative errors for the same value of $E = M + N$, since they need the same number of Taylor coefficients.

In order to make the last argument more clearly we also plotted the Padé approximant in $z_0 = 2 + 0.5i$ evaluated in $z = 2$ in Figure 5.10 for $N = 0, 2, 4$. As expected we have again exponential convergence, even for $N = 0$. This shows that we can also get good results in $z = 2$, if the center is close enough.

Next, we investigate whether we can see for the scattering problem the same behaviour as in Figure 5.2 and Figure 5.4 for the interior Helmholtz problem, where at some point increasing M does not change anything any more, since the result for the FEM need to be more accurate to get better results for the Padé approximation. In the two plots Figure 5.11

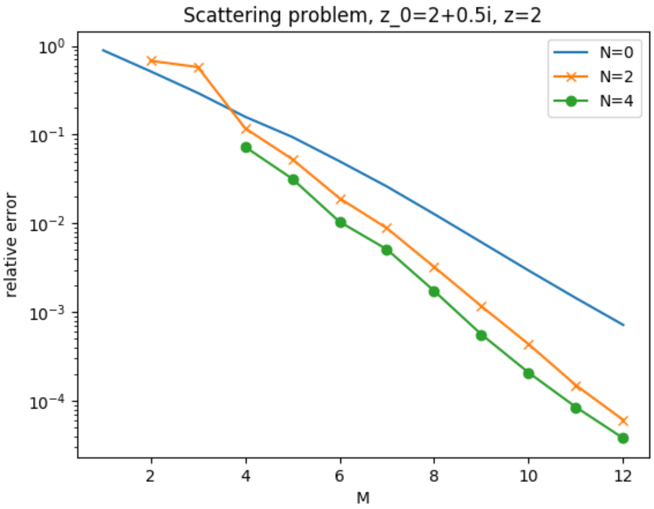


Figure 5.10: Convergence for the scattering problem in $z = 2$ with center in $z_0 = 2 + 0.5i$

and Figure 5.12, the parameter h always denotes the maximum mesh size h_{max}, which was used to construct the mesh with NGSolve. Both examples were done with the center $z_0 = 3 + 0.5i$ and in $z = 3$.

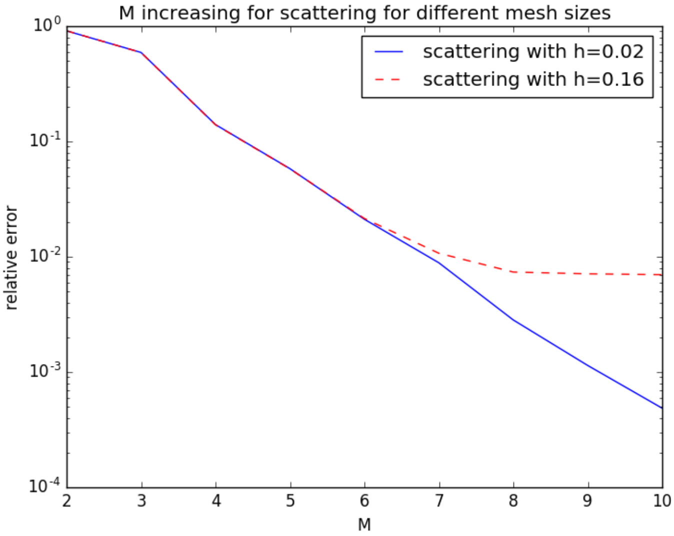


Figure 5.11: Convergence for the scattering problem for different mesh sizes

In Figure 5.11 we can see that for a rough mesh with $h = 0.16$ we reach the point, where increasing M does not help any more at $M = 8$, while with a finer mesh ($h = 0.02$), we can still decrease the error by increasing M further. Furthermore the number of degrees M in the numerator, which decrease the error for a given mesh size, is much higher for the

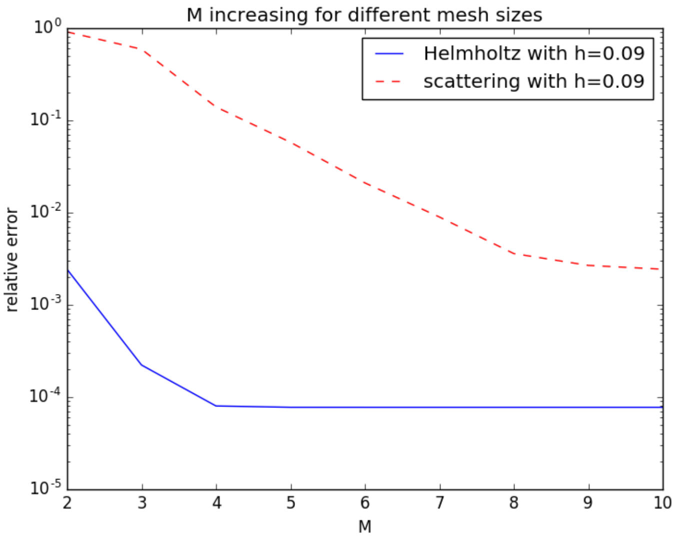


Figure 5.12: Convergence for the interior Helmholtz problem and the scattering problem with the same mesh size

scattering problem than the interior Helmholtz problem. Looking at Figure 5.12, we can see that here for the interior Helmholtz problem the error decreases only up to $M = 4$, while for the scattering problem this is the case up to $M = 9$. This is probably due to the fact that the scattering equation needs a finer mesh than the interior Helmholtz problem to get the same accuracy for the FEM solution. We obtain again that the Padé approximation works much better for the interior Helmholtz problem than for the scattering problem.

5.3.1 Numerical results in 3D

We also run a test for the scattering equation in 3D.
The domain Ω is here $[0, \pi]^3 \backslash B((\pi/2, \pi/2, \pi/2), \pi/8)$, i.e. we take the cube from 0 to π and exclude the ball centered in $(\pi/2, \pi/2, \pi/2)$ with radius $\pi/8$. As for the interior Helmholtz problem we used CS geometry to build up the domain. The cube is again an intersection of half-planes and then the sphere is defined as an extra object. We did not change the direction and therefore the incident wave is, like in the 2D example, travelling in the direction of the first coordinate. Boundary conditions are also the same as before. So we have Dirichlet condition on the sphere and the mixed Robin boundary condition on the boundary of the cube. Since the calculation are more expensive in 3D in terms of calculation time and memory, we only used a mesh with a maximum mesh size of $h = 0.12$. As in 2D we centered the Padé approximation in $z_0 = 3 + 0.5i$ and evaluated it against the FEM $\mathbb{P}_3$ solution in $z = 3$. The relative error is plotted against the degree of the numerator M for different values of the degree of the denumerator N.

We can see in Figure 5.13 that in three dimensions we have also an exponential convergence, if the disk $B(z_0, |z - z_0|)$ is completely in $\mathbb{C}^+$. This is similar to the 2D example evaluated in $z = 3$ in Figure 5.9. The results of the experiments done in 3D for the scattering and

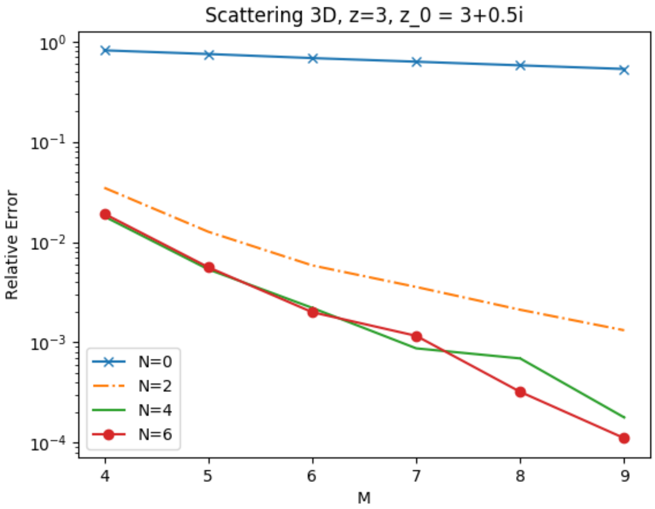


Figure 5.13: Convergence for the scattering problem in $z = 3$ in three dimensions, $N = 0, 2, 4, 6$, M growing, center $z_0 = 3 + 0.5i$

the Helmholtz problem (see Section 5.2.1) seem to be of similar accuracy as in 2D. This is consistent in the sense that the theoretical results do not depend on the dimension of the domain Ω, where the problem is defined. One difference is of course that the calculation takes much longer, since calculating Taylor coefficients and evaluating integrals takes more time in higher dimensions, since we need more nodes in order to have the same mesh size.

6 Numerical results for the multi point Padé approximation

6.1 Remarks on the implementation

In this section we will investigate some practical results of the Newton-Padé approximation. As in the single point case, Netgen/NGSolve and its python interface NGSpy are used. We basically implemented Algorithm 2. Still there are, as in the single point case, a couple of choices one has to make, which can have a huge influence on the accuracy of the results. We will have a look at some results from the interior parametric Helmholtz problem with Dirichlet boundary conditions (Problem 1) and the scattering problem (Problem 2). For the interior parametric Helmholtz problem we will have a special look at higher frequencies ($\nu^2 \in (39, 55)$), since in [BNPP18] it is shown numerically that the single point Padé cannot resolve all singularities for this frequencies and we will see that with the right choice of points the Newton-Padé performs better.
Regarding the distribution of the real parts of the points, different strategies could be used. We decided here to take Chebyshev points of first kind. k Chebyshev nodes $(z_1, z_2, \dots, z_k)$ in some interval $[a, b]$ are calculated with the formula

$$z_i = \frac{1}{2}(a+b) + \frac{1}{2}(b-a)\cos\left(\frac{2i-1}{2k}\pi\right) \qquad i = 1, \dots, k \tag{6.1}$$

Chebyshev nodes of the first kind are chosen since it is known that they are a good choice for polynomial interpolation for $\mathbb{R}$-valued functions and, in contrast to second kind, they exclude the boundaries (see [Xu16]). This seemed an useful property since the convergence is always an area around the centers and therefore, if we would choose one center on the boundary of the interval, we would loose some of the convergence area.
For the imaginary parts we take a constant value of 0.5. One could take also some other (small) positive real number in order to make sure, that our centers do not coincide with the singularities. In the future one could investigate whether it is possible to achieve better results with other imaginary parts.
After calculating the points, we do a reordering of them. Therefore, let us look at some vector $z = (z_1, z_2, \dots, z_k)$, which will be reordered to $(z_{k/2}, z_{k/2-1}, z_{k/2+1}, z_{k/2-2}, z_{k/2+2}, \dots)$. If k is not even, $\frac{k}{2}$ is always rounded up. The calculation of the Chebyshev points and the reordering is done with the function in Listing 6.1.

```
#calculate chebyshev points and reorder them
#a,b boundary of interval, n number of point
#imag imaginary part of points
import numpy as np
import math
pi=math.pi
def calculate_chebyshev(a,b,n,imag):
        p = np.zeros(n) + np.zeros(n)*1j
        #calculate points
        for ind in range(n):
                p[ind] = 0.5*(a+b)+0.5*(b-a)\
                *np.cos((2*(ind+1)-1)*pi/(2*n)) + imag*1j
        #make reordering
        p2 = np.zeros(n) + np.zeros(n)*1j
        if (n%2 == 0):
                for ind in range(n):
                        if (ind%2 == 0):
                                p2[ind] = p[int((n/2)-1-ind/2)]
                        if (ind%2 != 0):
                                p2[ind] = \
                                p[int(n/2 + int(ind/2))]
        else:
                for ind in range(n):
                        if (ind%2 == 0):
                                p2[ind] = p[int(int(n/2)-ind/2)]
                        if (ind%2 != 0):
                                p2[ind] = \
                                p[int(int(n/2) + int(ind/2)+1)]

        return p2
```

Listing 6.1: Calculation of centers

We do the reordering since points with a lower index in z influence the result more and since the points in the middle of the interval have more "important information", we want those points in the beginning of the vector. This is probably due to the definition of the basis $\{w_{0i}\}$ and the fact that for the construction of P and Q, we only use the first M respectively N elements. If we only use the first M of totally E basis elements, also only the first M centers z_i are used to construct this basis. We will discuss other reorderings in Section 6.3 and see that this reordering seems to work quite well compared to others. Still there may be other reordering, which are not tested here and lead to more accurate results. Therefore this is a potential area for improvements if one wants to further develop this method.

For the functional (3.3), we will always use $\rho = 1$ for the Newton-Padé approximation.

A more detailed description of the implementation and the most important functions are given in the appendix.

6.2 Parametric interior Helmholtz problem

First of all, one has to choose two parameters for the Finite-Element space, the maximum mesh size h_{max} and the polynomial order p. In the following example $h_{max} = 0.1$ and $p = 5$ are chosen. Using this paramaters we achieve for a wavenumber of $\nu^2 = 47$ a relative error for the FE solution compared with the analytical solution of around $1.2 * 10^{-6}$.

In the first example we check whether we can achieve again exponential convergence for growing degree M of the numerator. We take one, four and six Chebyshev points on the interval $[2,4]$ with 12 evaluations in total and evaluate the Newton-Padé in $z = 3$. We use as norm $\|\cdot\|_{V,\sqrt{3}}$. In Figure 6.1 one can see that we achieve a convergence for multiple centers which bit slower than in the single point case, but still a straight line on a log-scale. We will later see that this is due to the fact that we have a relatively small interval and that in the case of one center this center $z_0 = 3 + 0.5i$ is quite close to the evaluation in $z = 3$.

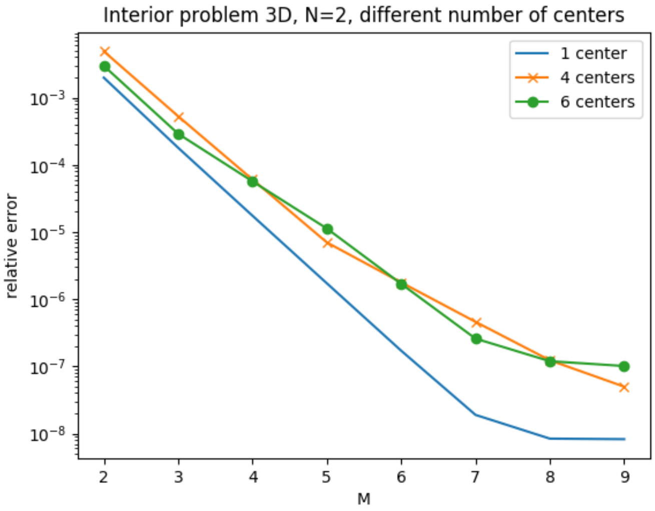


Figure 6.1: Convergence for the Helmholtz problem in $z = 3$ with Newton-Padé and one, four and six Chebyshev points on $[2,4]$

Next, we have a look at how well the Newton-Padé works, when one tries to approximate a whole interval. One would expect that the Newton-Padé potentially outperforms here the single-point, since one can cover a larger region with multiple points. We will see that this is in fact the case, if we choose the right number and location of the points.

As discussed before we will choose the example from [BNPP18, Section 6]. We have the interval $[39,55]$ and calculate the Newton-Padé with different number of Chebyshev points. We will take the parameters $M = 10$ and $N = 6$. The number of evaluations in each point will be $18/k$, where k denotes the number of points. In Table 6.1 there is an overview of the different choices for the number of centers and the number of evaluations in each center. In all the cases we have 18 evaluations and therefore, a similar computational effort in the offline phase of the Newton-Padé approximation. When comparing the results for one center with [BNPP18, Section 6] one should keep in mind that we have used here $\rho = 1$, while in the other paper $\rho = |z - z_0|$ is used.

We first check the convergence in three different points $z = 43, 47$ and 51 for growing degree of the numerator M for different number of centers in Figure 6.2. The relative error

number of centers	number of evaluations in each center
1	18
2	9
3	6
6	3
9	2
18	1

Table 6.1: Number of centers and iterations for the following experiments with the Newton-Padé approximations

is calculated as

$$\text{relative error} = \frac{\|u_h - u_{P,h}\|_{V,\sqrt{\mathrm{Re}(47+0.5i)}}}{\|u_h\|_{V,\sqrt{\mathrm{Re}(47+0.5i)}}},$$

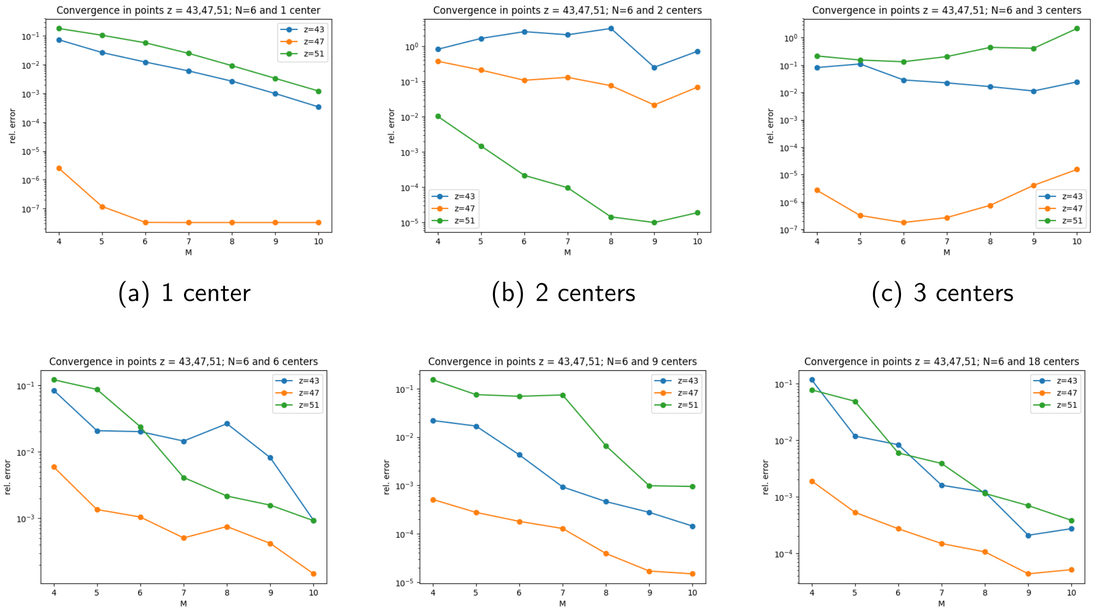
where u_h denotes the FEM solution and $u_{h,P}$ denotes the Newton-Padé approximation. For one center the result is equivalent to the single point Padé. Therefore, the convergence is a lot better for the middle point ($z = 47$), since the one center is located in the middle of the interval ($z_0 = 47 + 0.5i$) and therefore the distance is smaller to 47 than to the other two points (43 and 53).

Two and three centers seem to be not a good choice, since they do not outperform the single point method and then there is no reason to calculate the more complicated multi point Padé approximation. We will also see this in Figure 6.4 and Figure 6.6, when we further discuss the results.

The situation is different for a higher number of centers ($6, 9$ and 18). We can achieve there convergence in all three points. Still, the accuracy is in a similar range (around 10^{-3}) as for the single point Padé. Comparing all plots with more than one center, we can also see that the best convergence results can be achieved for 18 centers, since we then have a relative error smaller than 10^{-3} in all three points.

We can see the difference between the single point and the multi point Padé better, when we look at a larger interval. In Figure 6.3 we can see that the advantage of the multi point Padé to the single point Padé grows, when we look at a larger interval. Here we have the interval $(36, 71)$. We look at the convergence in three points $43, 47$ and 51. We set $N = 11$, since there are 11 eigenvalues of the Laplacian in this interval. We did the calculation with one center with 36 derivatives and with 36 centers with one derivative for each center. The number of evaluations are doubled compared to the other example in order to be able to cover such a large interval. We can see that in $z = 51$ both methods still work quite good, since the point is near to the middle of the interval (55) where the single center is located (plus $0.5i$). For the other two points, which are farther away from the middle of the interval, only the version with 36 centers works out. This shows that one needs a multi point version of the Padé approximation, if one wants to cover a larger interval of frequencies with one approximation. However, we can also see that we do not have a steady convergence here for growing degree of the numerator M, but the relative errors are fluctuating for higher M.

We will now look again at the example on the interval $(36, 51)$ and discuss the localiza-



(a) 1 center

(b) 2 centers

(c) 3 centers

(d) 6 centers

(e) 9 centers

(f) 18 centers

Figure 6.2: Relative error of the Newton-Padé approximation for a different number of points chosen by Chebyshev nodes, the parameters are always $M = 10$ and $N = 6$

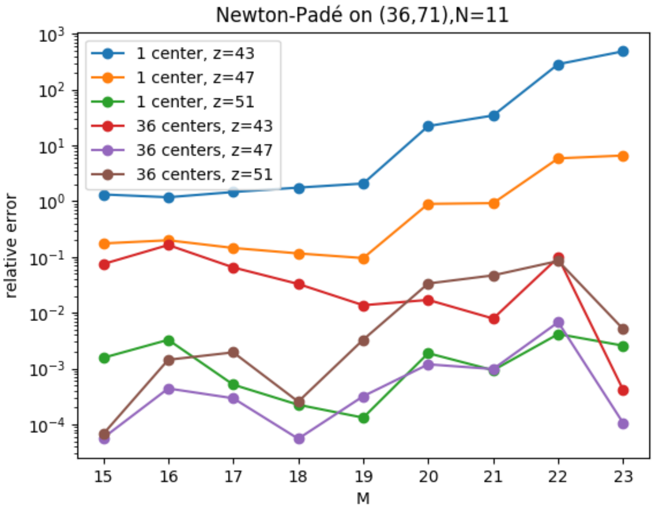
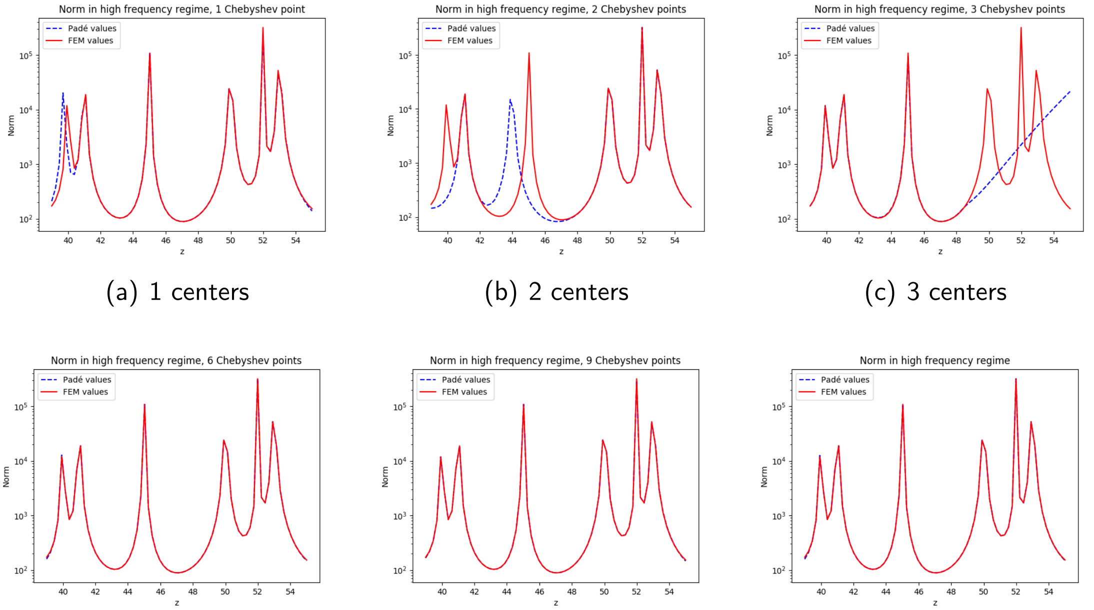


Figure 6.3: Convergence of the relative error for the parametric interior Helmholtz problem in $z = 43, 47, 51$ of the Newton-Padé approximation compared to the FE-solution on the interval $(36, 71)$ for one and 36 centers for growing M, $N = 11$



(a) 1 centers

(b) 2 centers

(c) 3 centers

(d) 6 centers

(e) 9 centers

(f) 18 centers

Figure 6.4: Newton-Padé approximation and FEM solution for a different number of points chosen by Chebyshev nodes, we always set $M = 10$ and $N = 6$

tion of the singularities by the Newton-Padé approximation. In Figure 6.4 there are the comparisons of the FEM-solution and the Newton-Padé approximation for the different numbers k of Chebyshev nodes. We can see that for $k = 1, 2, 3$ we cannot find all the singularities correctly, while for $k = 6, 9, 18$ the Newton-Padé is able to localize them correctly. This means that these methods could be used for the calculation of the eigenvalues of the Laplace operator instead of some eigenvalue solver. Here one would determine the eigenvalues by calculating the zeros of the polynomial Q in the denominator. These results can also be justified theoretically. Remember that all Dirichlet eigenvalues of the Laplacian in two dimension on the square $[0, \pi]^2$ can be calculated with the formula $m^2 + n^2$, where $m, n \in \mathbb{N} := \{1, 2, 3, \dots\}$(see [KS84]). On the interval $(39, 55)$, which we look at here, the eigenvalues are $40, 41, 45, 50, 52$ and 53.

It is interesting to note that the most inaccurate Newton-Padé is for $k = 3$. This is due to the way we reordered the centers. Let $z = (z_0, z_1, z_2)$ be the three centers in ascending order. Then we would have first the six evaluations for z_1, then for z_0 and then for z_2. Since the evaluations of z_2 are all at the end of the vector z, they contribute less to the approximation. Therefore, we have a half of the interval, which is not enough covered, since we have to little evaluations in the middle point z_1 (more evaluations in z_1 would lead to a better results as we see it for one center) and z_0 is to far away. Another strategy to order the points would be to duplicate the vector $\hat{z}$, where each distinct z_i appears only once, i.e. a vector $z = (z_0, z_1, \dots, z_k, z_0, \dots, z_k, \dots, z_k)$, where each z_i appears once for each evaluation. We will call this a mixed ordering. At first sight this seems a logical way to avoid the problem for three centers described before. We tried this for the three centers, but as we can see in Figure 6.5 this is not a good strategy at all.

Now we will have a closer look at the fact that the Newton-Padé can be used as an alternative to a traditional eigenvalue solver for the Laplace operator. For the examples in Figure 6.4, the roots of q r_Q were calculated and compared to the analytical roots λ. Similar numerical experiments for the convergence of the roots were already done in [BNPP18] for the single point Padé approximation. The eigenvalues of the Laplace on the interval $(39, 55)$, are, as discussed before, $40, 41, 45, 50, 52$ and 53. We will always look at the relative error $|r_Q - \lambda|/|\lambda|$. In the results in Figure 6.6 we can see that we have the accurate results for all roots only for a higher number of centers. We already assumed this looking at the plots in Figure 6.4.

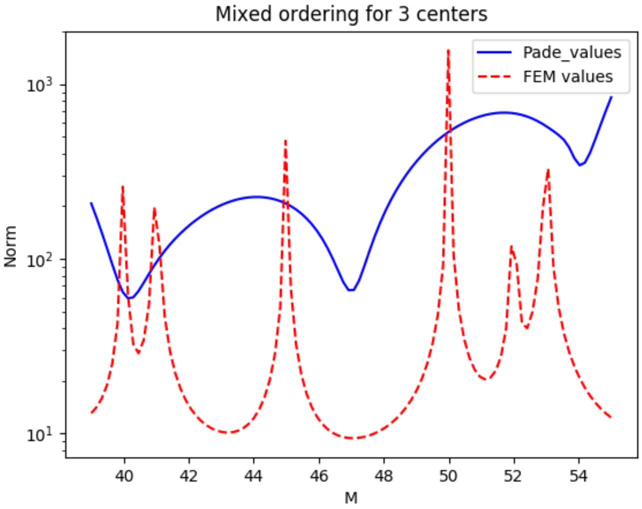
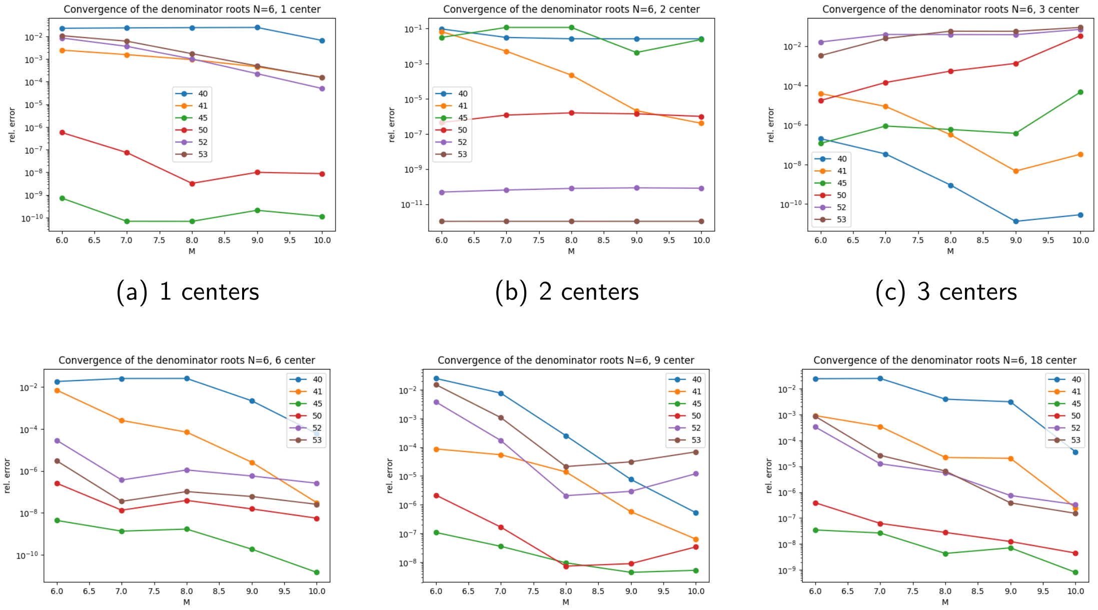


Figure 6.5: Mixed ordering, three centers, parameters as in Figure 6.4

Now we want to compare this way of calculating eigenvalues with a traditional eigenvalue solver, the inverse iteration with shift. The inverse iteration with shift is a derivative of the power method and calculates always the eigenvalue which is the nearest to the shift parameter σ, while the classical power method calculates only the eigenvalue with largest absolute value of a matrix (see [Bor16]). In the following we will construct a stiffness matrix A and a mass matrix M using NGSolve, export these matrices and then solve the generalized eigenvalue problem $Ax = \lambda Mx$ using inverse iteration. The code which we used for this can be found in the appendix in Listings 7.5.

We will investigate how many iterations we need to get an accuracy, which is similar to the eigenvalues calculated with the Newton-Padé.

When exporting matrices from NGSolve, one has to keep in mind that they are created without boundary conditions. We have stated before, that eigenvalues have the form m^2+n^2 for $m, n \in \mathbb{N}$. Since we do not include boundary conditions, we will have eigenvalues of the form $m^2 + n^2$ for $m, n \in \mathbb{N} \cup \{0\}$. An intuitive explanation for this additional, unwanted eigenvalues is, that normally we would rule out constant function $(\neq 0)$ with the zero



(a) 1 centers

(b) 2 centers

(c) 3 centers

(d) 6 centers

(e) 9 centers

(f) 18 centers

Figure 6.6: Convergence of the roots r_Q of the denominator Q to the analytical roots of the Laplacian λ, plotted is always the relative error $|r_Q - \lambda|/|\lambda|$, the parameters are always $M = 10$ and $N = 6$

boundary conditions. However we can look at an eigenvalue, which has the same neighbours in both cases, by choosing a suitable shift parameter. The convergence of the inverse iteration in such a case is not affected by the additional eigenvalues.

In the following we will create the matrices using $h_{max} = 0.1$ and polynomial degree $p = 5$. Then we export stiffness and mass matrix and start the eigenvalue solver with an accuracy of 10^{-7}, since this is roughly the accuracy that we can achieve with the Newton-Padé for a larger number of points (see Figure 6.6). Since the convergence speed of the inverse iteration is highly dependent on how well the eigenvalue is approximated by the shift parameter (see [Bor16],20.8), we will give the number of iterations for different shift parameters. The starting vector is always a vector with random numbers on the interval $(0, 1)$.

In the following we try to approximate the eigenvalue at $\lambda = 52$ by different shifts. The next eigenvalues, which influence the convergence, are 50 and 53 and therefore the results are not disturbed by the additional eigenvalues because of the missing boundary conditions. The results are given in Table 6.2.

Here we can see that the inverse iteration quickly gets inaccurate, if we do not know a good shift parameter. Therefore, the Newton-Padé approximation may be a good alternative, if one does not have a good estimate of the eigenvalue in advance. Another advantage of the method is that we can calculate multiple eigenvalues with one calculation.

In the future one could also try to investigate how the Newton-Padé approximation performs in comparison to other solvers which calculate multiple eigenvalues on a given interval (i.e. Lanczos algorithm). The eigenvalues of some other differential operator $\mathfrak{L}$ could be

Shift parameter σ	number of iterations
51.1	35
51.3	11
51.5	5
51.7	7
51.9	3
52.1	4
52.3	8

Table 6.2: Number of iterations needed in the inverse iteration to get an relative error smaller than 10^{-7}

calculated as well by applying the Newton-Padé to the equation $\mathfrak{L}u - \lambda u = 0$ with some suitable boundary conditions.

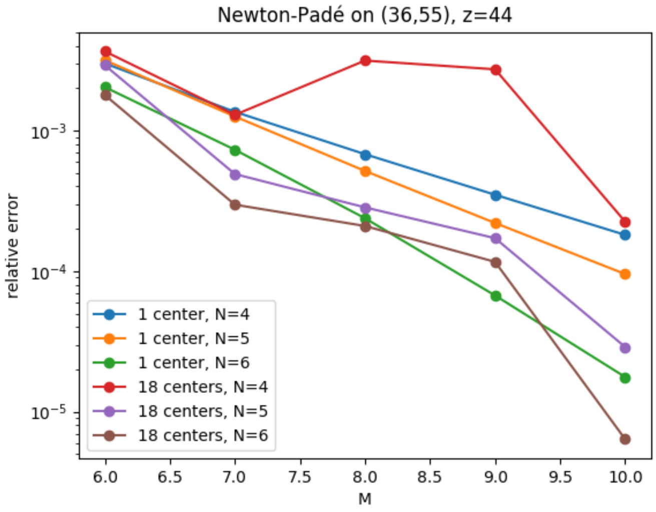


Figure 6.7: Convergence of the relative error for the parametric interior Helmholtz problem in $z = 44$ of the Newton-Padé approximation on the interval $(36, 55)$ for one and 18 centers for growing M and $N = 4, 5, 6$

We also investigated in high frequency regime how the multi and the single point Padé react to a change of the degree of the denominator N. We look again at the interval $(39, 55)$ and compare a single point Padé with 18 evaluations and a multi point Padé with 18 centers and one evaluation at each center. We compare the convergence in $z = 44$ for $N = 4, 5, 6$ for growing M. We can see in Figure 6.7 that the convergence is more uniform for one center than for 18. The other difference is that the multi point Padé seems to have problems for $N = 4$ as we can see for $M = 8, 9$. For higher N both methods work similarly well for this kind of distance to the middle of the interval.

Looking at all the experiments in this chapter one can see that covering a small interval of frequencies works fine for the interior parametric Helmholtz problem with the single and the multi point method or is even better with the single point version. As soon as we have to cover a large interval or want to calculate eigenvalues (i.e. roots of the denominator) one should use the multi point Padé approximation.

6.3 Scattering problem

In this section we will present some numerical results of the Newton-Padé approximation for the scattering problem introduced in Section 4.2. We set again $h_{max} = 0.1$ and $p = 5$. First, we have a look at the convergence rate for growing M. We take five Chebyshev points on the interval $[2, 5]$, set $N = 4$ and evaluate the Newton-Padé in $z = 3.5$ against the FEM solution. The norm used is $\|\cdot\|_{V,3.5}$. In Figure 6.8 we can see that we get indeed exponential convergence.

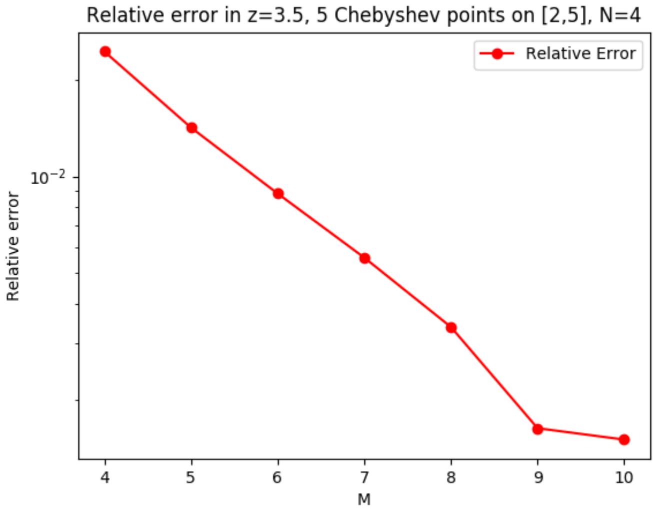


Figure 6.8: Convergence for the scattering problem in $z = 3.5$ with Newton-Padé and five Chebyshev points on $[2, 5]$

In Figure 5.9 we have seen that in the single point Padé we have problems approximating the scattering problem in $z = 2$ and $z = 3$ with one Padé approximation. We will now try to do this with a multi point Padé. Since we can place centers near both points here, we would expect better results.

In Figure 6.9 we have the results for the multi point Padé. We place six Chebyshev points on the interval $(1, 4)$ and three derivatives each. Then we set $N = 6$ and let M grow from 6 to 10.
We can see that except for $z = 2$ and $M = 8$ we have a steady convergence in both points. This is better than with the single point Padé and it highlights again the advantage of the multi point Padé, if one wants to cover a whole interval.

In the next example we will illustrate, why it is important to reorder the centers as described in Section 6.1. Assume we would have the centers and its evaluations in an

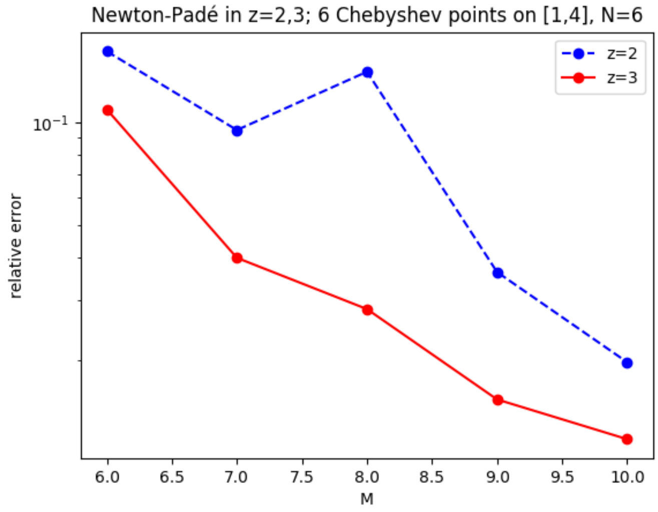
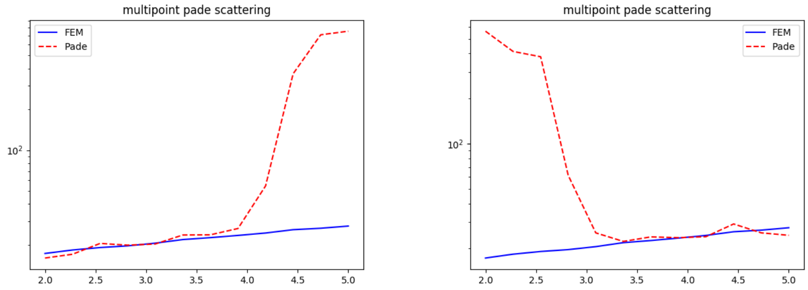


Figure 6.9: Relative error for the scattering problem in $z = 2$ and $z = 3$ with Newton-Padé and six Chebyshev points on $[1, 4]$, $N = 6$ and M growing

ascending order. Since points in the beginning have a greater influence on the result, we would have more information about smaller wave numbers and vice versa if we would store the centers in a descending order. We can see this in Figure 6.10. In both examples we have taken the parameters $M = 8$ and $N = 4$, which would be enough for more accurate results with a correct ordering of the points, as one can see in the "correct" half of the interval. We also try out again the mixed ordering described in Section 6.2. This is not working as

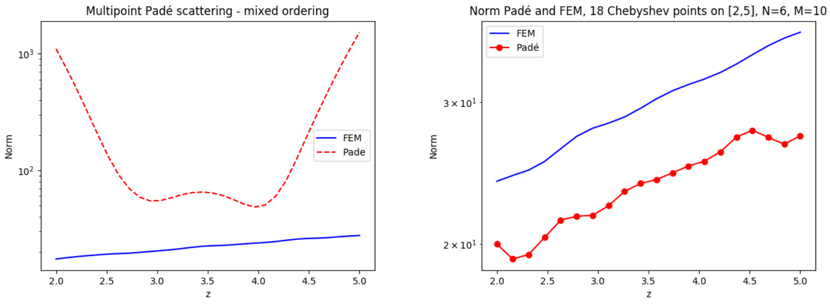


(a) ascending centers

(b) descending center

Figure 6.10: Newton-Padé for the scattering problem without reordering of the centers

we see in the left part of Figure 6.11. It is the same example as before and we can see that with this mixed ordering we are nowhere near the right solution on the whole interval. Therefore, this is not a useful strategy. In the right part of Figure 6.11 we can see the strategy from the code in Listing 6.1, which was used in this thesis. Here we can, at least, approximate the growth of the norm in most parts of the interval. However we also see that covering accurately a whole interval for the scattering problem is a challenging task and the approximation is already getting more inaccurately near the boundaries of the interval. As mentioned before there may be other reorderings which produce a more accurate solution.



(a) Mixed ordering for the Newton-Padé approximation - not a good choice at all

(b) Ordering used in this thesis

Figure 6.11: Mixed ordering and ordering according to the code in listing 6.1

7 Conclusions

We presented a method to solve a parameter-dependet PDE in a computationally cheap way for multiple values of the parameter, which was introduced in [BNP17]. We defined a solution map, which maps parameter values onto its solution for fixed right hand side and boundary conditions. Then we defined and calculated a Padé approximation of this solution map. The main focus was on numerical experiments. For the numerical experiments we defined two model problems, the Helmholtz equation with Dirichlet boundary conditions and a source term and a wave equation with additional scattering on a circle in the domain. We reviewed first the Padé approximation in one center (as in [BNP17] and [BNPP18]) and then extended it to multiple centers (multipoint Padé/Newton-Padé). For both strategies there are numerical experiments in two dimensions and for the single point Padé also in three dimensions. We figured out numerically that the Padé approximation works also in three dimensions. For the multipoint Padé we tried different strategies how to order the centers and we found in the numerical experiments with Chebyshev points one reordering which seems to achieve exponential convergence. For some numbers of centers we were able to outperform the single point Padé in a high frequency regime for the Helmholtz equation. We also saw that the Newton-Padé can be used as a eigenvalue solver for the Laplacian and is in the high frequency regime superior to the single point Padé in this perspective. The algorithm may be useful for calculating eigenvalues, if one wants to calculate multiple eigenvalues on a specific interval.

However especially for the Newton-Padé approximation there are still a lot of open questions. The ordering and choice of the centers was only investigated numerically and not theoretically as well as the ration number of points/derivatives, i.e. how one should spent the evaluations which are available. We also did not talk about the convergence rate or the convergence area. We have only seen in the experiments that in some cases exponential convergence can be achieved.

Appendix

In the following the code for some important functions used in the numerical experiments and a short description of it is given.

Listing 7.1 shows two functions. *calculate_pq* calculates the coefficients for the two polynomials P and Q and the second one *calculate_pade* evaluates the two polynomials. *IP* calculates the weighted H^1-inner product of two function. We can also calculate $\|\cdot\|_{V,\sqrt{Re(z_0)}}$ with this function. *IP* will also be used in the other pieces of code as well. We will change it slightly for the scattering problem, since we need then the $\|\cdot\|_{V,Re(z_0)}$-norm instead in order to be consistent with the results in [BNPP18].

```
#calculates coefficients for P and Q for single point Pade
#m1 degree of P, n degree of Q
#list_coeff contains Taylor coefficients (saved as GridFunction)
#z point in intervall, where Pade is likely to be evaluated
#z_0 center of Pade (used for weighted norm)

#function for Inner Product
def IP(a,b,mesh1):
        return Integrate(InnerProduct(grad(a),Conj(grad(b))) + ((z_0.real
           ))*(InnerProduct(a,Conj(b))),mesh1)

def calculate_pq(m1,n,list_coeff,z,z_0):
        #set up matrix G
        G=np.zeros((n+1,n+1),dtype=complex)
        rho = abs(z_0-z)

        #set up for loop over alpha
        for ind1 in range(n+1):
                for ind2 in range(n+1):
                        for alpha in range(m1+1,e+1,1):
                                G[ind2][ind1] += IP(list_coeff[alpha-ind1
                                    ],list_coeff[alpha-ind2],mesh)*rho
                                    **(2*alpha)


        w,v = linalg.eigh(G)
        #save eigenvector to smallest eigenvalue in q
        q=v[:,0]

        #calculate coefficients of p
        p=[]

        for ind in range(m1+1):
                p_cur = 0.*x+1j*0.*x
                #calculate p_ind = (QT)_ind
                for ind2 in range(min(ind+1,n+1)):

                        p_cur = p_cur + list_coeff[ind-ind2]*q[ind2]

                #put coefficient p_ind to list p
                p.append(p_cur)
        return p,q

#evaluates Pade approximation of degree m/n in z
#Pade centered in z_0
def calculate_pade(p,q,z,m,n,z_0):

        abstand = z-z_0
        q_val = 0.+0.*1j
        for ind in range(n+1):
                q_val = q_val + q[ind]*(abstand)**(ind)
        p_val = 0.*x+1j*0.*x
        for ind in range(degree_p+1):
                p_val = p_val + p[ind]*(abstand)**(ind)
        pade = p_val/q_val
        return pade
```

Listing 7.1: Calculation of P and Q for the single point Padé approximation

Listing 7.2 shows the main part of the code for the single point Padé approximation, where the Taylor coefficients are calculated. We look here at the Helmholtz equation with Dirichlet boundary conditions on the square $[0,\pi]^2$. In the beginning there are calls to the NGSolve library. This part is a modified version of example 1.7 from [itu18]. In order to build up the linear form, we use the function *calc_source*, which calculates the function f as defined in Chapter 5.2. Then we solve the PDE and afterward inductively calculate the derivatives. Here the right hand side is calculated according to Theorem 4.5. Then we save the Taylor coefficients in the list *list_coeff*. This data can be used to calculate the single point Padé with the code in Listing 7.1. In order to repeat the calculations in this thesis one needs to combine the code in Listing 7.1 and Listings 7.2.

```
#constructs vector T containing derivatives
#of the solution function

#import packages
import math
import cmath
from ngsolve import *
from netgen.geom2d import SplineGeometry
import numpy as np
from numpy import linalg as LA
import matplotlib.pyplot as plt
from scipy import linalg

#function that returns source term for wave number
def calc_source(nu):
        erg = 16./(pi**4.0)*exp(-(nu**0.5)*1j*(x*d1+d2*y))*(2*1j*(nu
           **0.5)*d1*(2*x*y*y-2*pi*x*y-pi*y*y+pi**2*y)+2*1j*(nu**0.5)*d2
           *(2*x*x*y-pi*x*x-2*pi*x*y+pi**2*x)\
         -(2*y*y - 2*y*pi) - (2*x*x - 2*x*pi))
        return erg


#main function
if (__name__ == '__main__'):
        #list for solutions
        list_sol=[]

        #start calculating solution at z_0

        #Define mesh
        geo = SplineGeometry()
        geo.AddRectangle((0,0),(pi,pi),bcs=["b","r","t","l"])
        mesh = Mesh(geo.GenerateMesh(maxh=0.06))

        # H1-conforming finite element space
        fes = H1(mesh, order=3, dirichlet="l|r|t|b", complex=True)

        # define trial- and test-functions
        u = fes.TrialFunction()
        v = fes.TestFunction()

        #source
        source = calc_source(omega)


        # Forms
        a = BilinearForm(fes)
        a += SymbolicBFI(grad(u)*grad(v)-z_0*u*v) #Helmholtz problem
        c = Preconditioner(a, type="multigrid", flags= {"inverse" : "
           sparsecholesky" })
        a.Assemble()

        #RHS
        f = LinearForm(fes)
        f += SymbolicLFI(source * v)
        f.Assemble()

        #solve system
        gfu = GridFunction(fes, name="u")
        inv = CGSolver(a.mat,c.mat,complex=True,printrates=False,maxsteps
           =200)
        gfu.vec.data = inv * f.vec

        #put solution in list
        list_sol.append(gfu)


        #calculate derivatives
        for num_dev in range(1,e+1):
                gfu2 = GridFunction(fes, name = 'u2')

                #new rhs from old solution
                f2 = LinearForm(fes)
                rhs = list_sol[(num_dev-1)]*num_dev
                f2 += SymbolicLFI(rhs*v)
                f2.Assemble()
                gfu2.vec.data = inv * f2.vec
                list_sol.append(gfu2)


        #multiply by 1/i! and make GridFunction again
        list_coeff = []
        for ind in range(len(list_sol)):
                coeff = GridFunction(fes,name='coeff')
                div = list_sol[ind]*(1/math.factorial(ind))
                coeff.Set(div)
                list_coeff.append(coeff)
```

Listing 7.2: Calculation of vector with derivatives for the single point Padé approximation

Listing 7.3 contains three functions. The function *calc_pq* calculates the coefficients of the polynomials P and Q, *w_ret* calculates the values of the Newton basis in x and *calculate_pade* evaluates P/Q in z_calc

```
#Multipoint Pade
#calculating coefficients of the polynomials P and Q
#F matrix with divided differences
#m1 degree of P, n degree of Q
def calc_pq(F,m1,n):

        G=np.zeros((n+1,n+1),dtype=complex)
        #make choice for rho
        rho = 1

        for ind1 in range(n+1):
                for ind2 in range(n+1):
                        for alpha in range(m1+1,e+1,1):
                                        G[ind1][ind2] += IP(F[alpha][
                                            alpha-ind2],\
                                        F[alpha][alpha-ind1],mesh)*rho
                                            **(2*alpha)


        p=[]
        for ind1 in range(m1+1):
                p_cur = CoefficientFunction(0.*x+1j*0.*x,1)

                #calculate p_ind = (QT)_ind
                for ind2 in range(min(ind1+1,n+1)):

                        p_cur = p_cur + F[ind1][ind1-ind2]*q[ind2]

                #put coefficient p_ind to list p
                p.append(p_cur)
        return p,q

#calcuates Newton basis for evaluation in x with centers saved in z
#x is double, z is vector of doubles
def w_ret(x,z):
        w = np.zeros(len(z))+1j*np.zeros(len(z))
        for i in range(len(z)):
                if i == 0:
                        w[i] = 1
                else:
                        w[i] = w[i-1]*(x-z[i-1])
        return w

#calculates Pade approximation of degree degree_p
#z_calc point where pade is calculated
#z2 vector with centers, appear multiple times if derivative
def calculate_pade(p,q,z_calc,z2,m,n):

        w = w_ret(z_calc,z2)
        q_val = 0.+0.*1j
        for ind in range(n+1):
                q_val = q_val + q[ind]*w[ind]
        p_val = CoefficientFunction(0.*x+1j*0.*x,1)
        for ind in range(m+1):
                p_val = p_val + p[ind]*w[ind]
        pade = p_val/q_val
        return pade
```

Listing 7.3: Calculation of P and Q for the Newton-Padé approximation

Listing 7.4 holds the main part of the calculation of the Newton-Padé approximation. This is an example for the scattering equation (problem 2). First the centers are calculated with Chebyshev points as described in Chapter 6.1. Then we set the number of derivatives in each center. Here we want a total of 18 derivatives equally distributed over the center (num_z denotes the number of centers). Then we set up a vector $z2$, which contains each center once for each derivative, that should be calculated.
The next part consist of calls to the NGSolve library. These calls are partly taken from [itu18]. We build up the domain and define a mesh and a FE-space. Then we loop over all different centers (contained in vector z) and solve the PDE. If we need to calculate derivatives, we have to take the formula from Lemma 4.6. In the end we we have to make Taylor coefficient out of the derivatives and save them as GridFunctions (data format from NGSolve) in *results2*. In the last step we construct the matrix F, which contains the divided differences. After these steps we have enough information to construct the Newton-Padé approximation with the functions in Listing 7.3.

```
#construct matrix F containing divided
#differences of the solution function

#importing packages
import math
import cmath
```

```
from ngsolve import *
from netgen.geom2d import SplineGeometry
import numpy as np
from numpy import linalg as LA
import matplotlib.pyplot as plt
from scipy import linalg

#auxiliary function to calculate index for calculation of F
def get_cur_z(j,d):
        ind = 0
        while(j-d[ind] > 0):
                j = j - d[ind]
                ind = ind + 1
        return ind

#main function
if (__name__ == '__main__'):
        #calculate centers
        z = np.zeros(num_z)+np.zeros(num_z)*1j
        z = calculate_chebyshev(2,5,num_z,0.5)

        #number of derivatives, 1 means zero derivatives,
        #i.e. minimum is 1 in vector
        devs = np.ones(num_z)
        devs = devs*(18/num_z)

        sum_dev = 0
        for num in devs:
                sum_dev=sum_dev+num;

        #set up vector where each z appears once for each derivative/
            result
        #that has to be computed for it
        z2 = np.zeros(int(sum_dev))+np.zeros(int(sum_dev))*1j
        index = 0
        for ind in range(num_z):
                for ind2 in range(int(devs[ind])):
                        z2[index] = z[ind]
                        index = index + 1



        # Geometry
        geo = SplineGeometry()
        geo.AddRectangle((-pi, -pi), (pi,pi), leftdomain=1, rightdomain
            =0,  bc="out_bound")
        #call rectangle 1 and inner cirle 2
        geo.AddCircle((0, 0), 0.5, leftdomain=2, rightdomain=1, bc="
            scat_bound")

        geo.SetMaterial (1, "outer")
        geo.SetMaterial (2, "inner")
```

```
#build mesh
mesh = Mesh(geo.GenerateMesh(maxh=0.1))
#define Finite Element space
fes = H1(mesh, order = 5, dirichlet = 'scat_bound', definedon = '
    outer', complex = True)

# define trial- and test-functions
u = fes.TrialFunction()
v = fes.TestFunction()


#list for derivatives
results = []

for z_num in range(len(z)):
        results.append([])

        #u_i
        u_i_coeff = calc_u_i(z[z_num])
        u_i = GridFunction(fes, name = 'u_i')
        u_i.Set(u_i_coeff)

        #bilinear form
        print(z)
        a = BilinearForm(fes)
        a += SymbolicBFI(grad(u)*grad(v))
        a += SymbolicBFI(-z[z_num]*z[z_num]*u*v)
        a += SymbolicBFI(-1j*z[z_num]*u*v, definedon=mesh.
            Boundaries('out_bound'))
        c = Preconditioner(a, type="multigrid", flags= {"inverse"
            : "sparsecholesky" })
        a.Assemble()

        #define outward normal
        normal = specialcf.normal(mesh.dim)

        #calculation for z_0
        #RHS
        f = LinearForm(fes)
        f += SymbolicLFI((grad(u_i)*normal-1j*z[z_num]*u_i)*v,
            definedon = mesh.Boundaries('out_bound'))
        f.Assemble()




        for num_dev in range(int(devs[z_num])):
                #calculate derivative
                if num_dev == 0:
                        #solve system
                        gfu = GridFunction(fes, name="u")
                        inv = CGSolver(a.mat,c.mat,complex=True,
                            printrates=False,maxsteps=200)
                        gfu.vec.data = inv * f.vec
                        results[z_num].append(gfu)

                else:


                        #new rhs from old solution
                        gfu = GridFunction(fes, name="u")
                        f2 = LinearForm(fes)
                        if(num_dev > 1):
                                f2 += SymbolicLFI(num_dev*(
                                    num_dev-1)*results[z_num][
                                    num_dev-2]*v)
                        f2 += SymbolicLFI(2*num_dev*z[z_num]*
                            results[z_num][num_dev-1]*v)
                        f2 += SymbolicLFI(num_dev*1j*results[
                            z_num][num_dev-1]*v, definedon = mesh.
                            Boundaries('out_bound'))

                        u_i_j_coeff = u_i*(1j*(d1*x+d2*y))**(
                            num_dev)
                        u_i_j = GridFunction(fes)
                        u_i_j.Set(u_i_j_coeff)
                        f2 += SymbolicLFI((grad(u_i_j)*normal)*v,
                             definedon = mesh.Boundaries('
                            out_bound'))
                        f2 += SymbolicLFI((-1*(1j)**num_dev*(d1*x
                            +d2*y)**(num_dev-1)*(z[z_num]*1j*(x*d1
                            +y*d2)+num_dev))*u_i*v, definedon =
                            mesh.Boundaries('out_bound'))
                        f2.Assemble()

                        gfu.vec.data = inv * f2.vec
                        results[z_num].append(gfu)

#make Taylor coeffients out of derivatives
results2 = []
for ind1 in range(len(results)):
        results2.append([])
        for ind2 in range(len(results[ind1])):
                div = results[ind1][ind2]*(1/math.factorial(ind2)
                    )
                gfu2 = GridFunction(fes)
                gfu2.Set(div)
                results2[ind1].append(gfu2)


F=[]


#construct matrix F with entries f_ji
for i in range(m+n+1):
        F.append([])
```

```
                for j in range(i,-1,-1):
                        if (z2[i] == z2[j]):
                        #we have to take derivative of f or f(z2[
                            i])
                                gridf = GridFunction(fes)
                                gridf.Set(results2[int(get_cur_z(
                                    i+1,devs))][int(i-j)])
                                F[i].append(gridf)
                        else:
                        #we have to take divided difference
                                gridf = GridFunction(fes)
                                gridf.Set((F[i][i-j-1]-F[i-1][i-j
                                    -1])/(z2[i]-z2[j]))
                                F[i].append(gridf)
```

Listing 7.4: Calculation of matrix F consisting of divided differences

The next piece of code in Listing 7.5 was used for the eingevalue calculation in Chapter 6. First one construct with *NGSolve* a stiffness and a mass matrix and exports them to Python using the csc format in *Scipy*.
The function *inverse_iteration* in the second part calculates the eigenvalue of the generalized eigenvalue problem $Ax = \lambda Bx$, which has the smallest distance to $shift$. eps gives the maximal relative error to the analytical value $real\_ev$ and $num\_iterations$ the maximal number of iterations. The function returns the number of iterations needed to get a relative error smaller than eps, since we evaluated this for the results in Table 6.2.

```
#calculating eigenvalues of Laplacian using
#inverse iteration
import math
import cmath
from ngsolve import *
from netgen.geom2d import SplineGeometry
import numpy as np
from scipy import linalg
import scipy.sparse as sp
import scipy.sparse.linalg as spla

pi=math.pi
#Define mesh
geo = SplineGeometry()
geo.AddRectangle((0,0),(pi,pi),bcs=["b","r","t","l"])

mesh = Mesh(geo.GenerateMesh(maxh=0.1))
fes = H1(mesh, order=5, dirichlet="l|r|t|b", complex=True)

# define trial- and test-functions
u = fes.TrialFunction()
v = fes.TestFunction()
a = BilinearForm(fes)
a += SymbolicBFI(grad(u)*grad(v))
a.Assemble()
```

```
m = BilinearForm(fes)
m += SymbolicBFI(u*v)
m.Assemble()

#make matrix to scipy sparse format
rows,cols,vals = a.mat.COO()
A = sp.csc_matrix((vals,(rows,cols)))

rows,cols,vals = m.mat.COO()
M2 = sp.csc_matrix((vals,(rows,cols)))

#calculate eigenvalue using inverse iteration
def inverse_iteration(A,B,num_iterations,eps,shift,real_ev):
        #start point
        x = np.random.rand(A.shape[1])

        #make shift
        A = A - shift*B
        #construct sparse LU decomp to solve system
        solve = spla.factorized(A)

        #iterations
        for ind in range(num_iterations):


                #calculate relative error
                rhs = B.dot(x)
                Ax = A.dot(x)
                eigen = x.dot(Ax)/(x.dot(rhs))

                err = linalg.norm(eigen*rhs - Ax)
                err = abs(real_ev-(eigen+shift))/real_ev

                if ( err > eps):
                        x = solve(rhs)
                        x = x/linalg.norm(x)
                else:
                        return ind
                if (ind == num_iterations-1):
                        eigen = x.dot(A.dot(x))/(x.dot(B.dot(x)))

        return ind
```

Listing 7.5: Calculating eigenvalues of Δ using *NGSolve* and inverse iteration

Bibliography

[BGM96] George A. Baker and Peter Graves-Morris. *Padé Approximants*. Cambridge University Press, 1996.

[BNP17] Francesca Bonizzoni, Fabio Nobile, and Ilaria Perugia. Convergence anaylsis of padé approximations for helmholtz frequency response problems. *ESAIM: Mathematical Modelling and Numerical Analysis*, 2017.

[BNPP18] Francesca Bonizzoni, Fabio Nobile, Ilaria Perugia, and Davide Pradovera. Least-squares padé approximation of parametric and stochastic helmholtz maps. *ArXiv e-prints*, May 2018.

[Bor16] Folker Bornemann. *Numerische lineare Algebra - Eine konzise Einführung mit MATLAB und Julia*. Springer Spektrum, 2016.

[Cla76] Guido Claessens. The rational hermite interpolation problem and some related recurrence formulas. *Computers & Mathematics with Applications*, 2(2):117–123, 1976.

[Cla78] Guido Claessens. On the newton-padé approximation problem. *Journal of Approximation Theory*, pages 150–160, 1978.

[Eva10] Lawrence C. Evans. *Partial differential equations*. American Mathematical Society, 2010.

[FL07] Michael S. Floater and Tom Lyche. Two chain rules for divided differences and faà di bruno's formula. *Mathematics of Computation*, 76(258):8, 2007.

[GHR98] Philippe Guillaume, Alain Huard, and Vincent Robin. Generalized multivariate pade approximants. *Journal of Approximation Theory 95*, 1998.

[HR00] Alain Huard and Vincent Robin. Continuity of approximation by least-squares multivariate pade approximants. *Journal of Computational and Applied Mathematics 115*, 2000.

[itu18] *Interactive NGSolve Tutorial*. `https://ngsolve.org/docu/nightly/i-tutorials/`, July 2018.

[Kal18] Josef Kallrath. On rational function techniques and padé approximants. `https://www.astro.ufl.edu/~kallrath/files/pade.pdf`, July 2018.

[KS84] J. Kuttler and V. Sigillito. Eigenvalues of the laplacian in two dimensions. *SIAM Review*, 26(2):163–193, 1984.

[ngs18] Homepage of netgen/ngsolve. `https://ngsolve.org/`, July 2018.

[Pri03] H.A. Priestley. *Introduction to Complex Analysis*. OUP Oxford, second edition, 2003.

[Xu16] Kuan Xu. The chebyshev points of the first kind. *Applied Numerical Mathematics*, 102:17–30, 2016.

List of Figures

4.1 Real part of the $\mathbb{P}^3$ finite element solution in $z = 3$ 29

5.1 $\mathbb{P}^3$ finite element solution in $\nu^2 = 12$ 32

5.2 Center $z_0 = 3 + 0.5i$, approximation in $z = 3$, M growing, $N = 2$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{M+1}$ we have $\rho = |3 + 0.5i - 3|$ and $R = |3 + 0.5i - 8|$ 33

5.3 Center $z_0 = 3 + 0.5i$, approximation in $z = 6$, M growing, $N = 2$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{M+1}$ we have $\rho = |3 + 0.5i - 6|$ and $R = |3 + 0.5i - 8|$ 33

5.4 Comparison of different mesh sizes, Center $z_0 = 3 + 0.5i$, approximation in $z = 3$ 35

5.5 Comparison of FEM solution and Padé approximation centered in $z_0 = 9+0.5i$ with parameters $M = 4$ and $N = 2$ in 2D 35

5.6 Comparison of the FEM solution with the Padé centered in $9+0.5i, 10+0.5i$ and $11 + 0.5i$ in 3D 37

5.7 Comparison of the FEM solution with the Padé centered in $9 + 0.5i$ in 3D - we have problems to approximate the singularity at 11 well 37

5.8 Convergence of the Padé approximation for the interior Helmholtz problem in 3 dimensions for growing M for $N = 2, 4$, for the theoretical bound $\left(\frac{\rho}{R}\right)^{(M+1)}$ we have $\rho = |10 + 0.5i - 10|$ and $R = |10 + 0.5i - 14|$ 38

5.9 Convergence for the scattering problem with center $z_0 = 3 + 0.5i$ evaluated in $z = 2$ and $z = 3$ 39

5.10 Convergence for the scattering problem in $z = 2$ with center in $z_0 = 2 + 0.5i$ 40

5.11 Convergence for the scattering problem for different mesh sizes 40

5.12 Convergence for the interior Helmholtz problem and the scattering problem with the same mesh size 41

5.13 Convergence for the scattering problem in $z = 3$ in three dimensions, $N = 0, 2, 4, 6$, M growing, center $z_0 = 3 + 0.5i$ 42

6.1 Convergence for the Helmholtz problem in $z = 3$ with Newton-Padé and one, four and six Chebyshev points on $[2, 4]$ 45

6.2 Relative error of the Newton-Padé approximation for a different number of points chosen by Chebyshev nodes, the parameters are always $M = 10$ and $N = 6$ 47

6.3 Convergence of the relative error for the parametric interior Helmholtz problem in $z = 43, 47, 51$ of the Newton-Padé approximation compared to the FE-solution on the interval $(36, 71)$ for one and 36 centers for growing M, $N = 11$ 47

6.4 Newton-Padé approximation and FEM solution for a different number of points chosen by Chebyshev nodes, we always set $M = 10$ and $N = 6$ . . . 48

6.5 Mixed ordering, three centers, parameters as in Figure 6.4 49
6.6 Convergence of the roots r_Q of the denominator Q to the analytical roots of the Laplacian λ, plotted is always the relative error $|r_Q - \lambda|/|\lambda|$, the parameters are always $M = 10$ and $N = 6$ 50
6.7 Convergence of the relative error for the parametric interior Helmholtz problem in $z = 44$ of the Newton-Padé approximation on the interval $(36, 55)$ for one and 18 centers for growing M and $N = 4, 5, 6$ 51
6.8 Convergence for the scattering problem in $z = 3.5$ with Newton-Padé and five Chebyshev points on $[2, 5]$ 52
6.9 Relative error for the scattering problem in $z = 2$ and $z = 3$ with Newton-Padé and six Chebyshev points on $[1, 4]$, $N = 6$ and M growing 53
6.10 Newton-Padé for the scattering problem without reordering of the centers . 53
6.11 Mixed ordering and ordering according to the code in listing 6.1 54