Isogeometric Analysis for Compressible Flows with Application in Turbomachinery

by

Andrzej Jaeschke

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on Monday August 31, 2015 at 9:00 AM.

Student number:	4419758	
Project duration:	October 1, 2014 – August 31, 2015	
Thesis committee:	Dr. M. Möller,	TU Delft, supervisor
	Prof. dr. ir. K. Vuik,	TU Delft
	Dr. H. Schuttelaars,	TU Delft

TU Delft Delft University of Technology

Preface

Computer-aided design (CAD) and finite element analysis (FEA) tools are extensively used in industrial design processes. The idea of isogeometric analysis (IGA) is to bring those worlds closer together. Combining them significantly increases the efficiency of the design and development processes. The general idea of the IGA approach is to use non-uniform rational B-splines (NURBS) that are used to represent the geometry in the CAD tool also as a basis for numerical analysis of the partial differential equations (PDE) via FEA. The simplest setting is to use the space spanned by NURBS as a search/test space in the standard Galerkin method.

Nowadays, increasing the efficiency of turbo-engines and therefore decreasing the emissions of mainly CO_2 and NO_x is a field of very active research. Therefore, design optimization of turbomachines, largely based on numerical simulations, is extensively performed in industry. An additional motivation for solving the underlying flow problems by IGA is its ability to exactly represent the complex shapes of domains, which is an important feature for flow solvers that need to accurately resolve boundary layers.

The topic of this thesis project: application of isogeometric analysis to compressible flow problems is very broad and it is beyond the scope of this thesis to answer all questions arising during the pilot implementation of an IGA solver for compressible flows. Therefore, it was decided to set the implementation of a B-spline-based IGA solver for the compressible Euler equations as the main goal. To achieve this goal several intermediate milestones were set. The first was the implementation of a simple B-spline basis generator and evaluator. The next step was to solve the Poisson problem with IGA. The next step consisted in implementing the solver for the convection-diffusion equation in stationary and time-dependent variants. Those steps led to the final milestone - the implementation of the compressible inviscid flow solver based on the IGA approach.

It is well known that the standard Galerkin finite element method as well as its isogeometric counterpart suffer from infamous oscillatory behaviour for convection-dominated problems and problems including discontinuities or steep gradients in the domain. Therefore, the algebraic flux correction (AFC) stabilization was implemented to avoid oscillations and non-physical values in the solution. The main novelty of this thesis is to combine AFC with IGA for the compressible Euler equations.

All solvers implemented during the thesis work were successfully validated using common benchmark problems. This work sets a base for further research and development that will hopefully lead to a productive implementation of IGA-based optimisation in industrial turbomachinery design.

I would like to express my sincere gratitude to my advisor Dr. Matthias Möller for the support of my project. His patience and expertise were a source of indispensable help. Besides

my advisor, I would like to thank all members of my thesis committee: Prof. Kees Vuik and Dr. Henk Schuttelaars for their insightful comments.

Last but not the least, I would like to thank Prof. Bert Jüttler from Johannes Kepler University in Linz (JKU) and Dr. Angelos Mantzaflaris from Johann Radon Institute for Computational and Applied Mathematics (RICAM) of Austrian Academy of Sciences (ÖAW) for their extensive support and for answering numerous questions concerning the G+SMO library.

Andrzej Jaeschke
Delft, August 2015

Contents

Chapter 1

Introduction

Nowadays, computer-aided design (CAD) and finite element analysis (FEA) software packages are widely used by the engineering community in design and optimization processes. Preparation of the geometry obtained from CAD software for the FEA is a long, manual process that is not suitable for automation. The aim of the isogeometric analysis (IGA) approach is to bring the CAD and the FEA worlds closer together and to allow to avoid the manual transition between them. This would enable the automation of design optimization processes.

Most of the CAD systems use non-uniform rational B-splines (NURBS) to represent the geometries. The idea of isogeometric analysis is to use the NURBS basis functions that are used to represent the geometry, as a basis for analysis. In the simplest setting, NURBS functions are used as basis functions for test and trial spaces in the standard Galerkin formulation. This method has two significant advantages over the standard FEA. Firstly, the geometry of the domain is represented exactly as designed in the CAD system, while for standard FEA the geometry is typically approximated by a surface triangulation unless special isogeometric curved elements are adopted. Secondly, slight modifications of the shape of the domain do not require a costly re-meshing. However, the isogeometric analysis has one major bottle-neck: the matrix assembly is more costly than that of the standard FEM as the basis functions can have much wider overlap of supports.

The advantages of the isogeometric analysis become even more significant for compressible flow problems, where the behaviour of the fluid close to boundaries is very important. A very special and sophisticated field where IGA can be used to analyse compressible fluid flows is turbomachinery. The optimization of the turbomachines is a very important research topic since many years. Turbomachines are used in many applications, starting from the aerospace engineering, over energy production and ending at the automotive industry. In all those fields efficiency plays a crucial role because of high ecological and economical expectations. Reducing the emissions of, inter alia, CO_2 and NO_x is a field of research of very high priority. This motivates the choice of the topic of this thesis.

1.1 Scope of the thesis

The topic of this thesis: application of isogeometric analysis to the flow problems is very broad and it would not be possible to answer all questions arising during the pilot implementation of IGA solver for compressible flows in the time of one master project. Therefore, it was decided to set the implementation of an IGA-based solver for the compressible Euler equations as the

main goal. To achieve this goal several intermediate milestones were set. The first was to implement a Matlab code capable of building and evaluating the B-spline basis. For the sake of simplicity it was decided to use less complex counterparts of NURBS - B-splines instead of NURBS during this thesis project. Solver for the Poisson's problem was implemented and generalized towards arbitrary geometries. Those two goals were accomplished during the literature study. Our experiments with the Matlab code suggested the use of an external, highly optimized library for efficiency reasons, to enable the numerical simulation of compressible flows. Therefore, the choice of suitable library was done as the next step.

It was considered helpful to implement and investigate the behaviour of solvers of simpler problems that are causing similar difficulties for the numerical schemes as the Euler equations prior to the implementation of the inviscid compressible flow solver. The first problem analysed was the stationary convection-diffusion equation. It is widely known that the Galerkin FEM with standard Lagrange basis functions as well as its IGA counterpart [1] are not capable of solving convection-dominated problems without generating spurious oscillations unless a suitable stabilization method is employed. Although a general practice is to use SUPG stabilization, it was decided to use stabilization methods of AFC-type. Another requirement at this stage was to find a proper projection of initial and boundary data to the considered B-spline spaces. Standard L2 projection generated non-physical under- and over-shoots and therefore it was decided to implement a so-called constrained L2 projection to overcome this deficiency.

The next milestone was to generalize the solver for convection-diffusion equation towards time-dependent problems. For this purpose several time discretization schemes were implemented including Strong Stability Preserving Runge-Kutta (SSP-RK) methods. The implementation of AFC-type stabilization methods was adjusted to be capable of stabilizing time-dependent problems. The last step to achieve the final goal was to generalize the solver towards systems of conservation laws. Implementation of boundary conditions for the Euler equations was an additional difficulty to pass. All solvers were validated using common benchmark problems.

1.2 State of the art

The isogeometric analysis was firstly proposed by Hughes et al. in [2]. The complete description of the method can be found in the book [1] published by this group few years later. The method has been originally developed in the field of mechanical problems, then extended to incompressible flows and fluid structure interaction problems [1]. Currently, there are several groups all over the world that are working on the topics connected with isogeometric analysis. According to the author's best knowledge there were not many published attempts to apply IGA to compressible flow problems. One of them was described by Trontin in [3]. There were successful works towards application of hierarchical splines and adaptive refinement in IGA presented by Vuong in [4]. Brunero in [5] successfully used Discontinuous Galerkin method for multi-patch problems.

It is a well known fact that the standard Galerkin finite element method applied to convection-dominated problems tends to produce non-physical oscillations near steep gradients or discontinuities. It was shown by Hughes et al. in [1] that the same problem occurs also in IGA. There are several stabilization methods that overcome this limitation. The most common is the streamline upwind/Petrov–Galerkin (SUPG) method introduced by Brooks

and Hughes in [6]. Harari and Hughes introduced the Galerkin/least-squares (GLS) and the Galerkin/gradient-least-squares (GGLS) in [7]. Bubble stabilization was proposed by Russo in [8]. Codina in [9] proposed stabilization based on a sub-grid scale approach and an algebraic approximation to the sub-scales. In this thesis an algebraic flux correction (AFC) methodology is used for stabilization. It was introduced by Kuzmin and Turek in [10]. Two types of limiting techniques that are considered in this thesis: TVD-type limiting and FCT-type limiting were introduced in [11] and [12], respectively. The method was generalized to systems of conservation laws in [13]. The comprehensive summary of this family of methods can be found in [14] and [15]. It is worth mentioning that AFC has been successfully generalised to quadratic Lagrangian finite elements but this approach turned out to be impractical and rather inefficient [16]. According to this work, there is a possibility of application of AFC for higher order basis functions, however for FEM it was very hard to implement due to negative values attained by basis functions. For the B-spline (NURBS) basis this problem will not occur.

1.2.1 Software packages

There exist several packages implementing the isogeometric analysis approach. Although most of them are still under development they provide useful toolboxes for solving problems by the IGA approach. In this section examples of existing packages are briefly described.

PetIGA [17] is a free software that implements IGA in C. It can considered as an extension of PETSc (the Portable, Extensible Toolkit for Scientific Computation). An important feature of this package is its support of parallelism [18].

Another package is **GeoPDEs** [19]. It implements IGA (also with T-spline basis) in Octave (compatible with Matlab) and is available under the GNU GPL license. It provides several modules for solving problems from the fields of linear elasticity, fluid mechanics and electromagnetism [20].

Igatools [21] is an open source C++ package implementing IGA. The developers state that it is computationally very efficient and that it supports parallelism. A very important advantage of this package is its extensive documentation [22].

Igafem [23] is an open source implementation of IGA conceived mostly for linear elasticity problems. It is implemented in Matlab and has a user-friendly graphical user interface (GUI). An important feature of this package is the support of not only NURBS basis functions but of T-splines basis functions as well [23].

The last example: **G+SMO** [24] (Geometry + Simulation Modules) is an open source, object-oriented, templated C++ library. It implements IGA with B-spline, Bernstein and NURBS bases and also hierarchical and truncated hierarchical B-spline bases. What makes this library particularly attractive for using it as software basis for this thesis is its assembler of matrices for stationary convection-diffusion-reaction equations, which can be used as robust assembly module for all matrices needed for solving the flow problems considered in this thesis.

Although there exist several well developed software packages, small Matlab codes were written as illustrating examples in the first two chapters. Writing the code from scratch allowed the author of this thesis to get a deeper understanding of IGA. However, more complex problems could not be solved in reasonably short time using the own code so it was decided to use the G+SMO package instead.

Chapter 2

B-splines and NURBS

Short theoretical introduction to B-splines and NURBS is provided in this chapter to acquaint the reader with the general ideas and nomenclature. Several important properties of B-splines are presented here as well. The Matlab code written by the author was used to illustrate the chapter with several examples.

2.1 Piecewise polynomial approximation

Polynomial approximations are widely used in numerical mathematics. Examples of polynomial interpolations are the Lagrange interpolation, the Newton interpolation and the Osculatory interpolation. The polynomial approximations gained high popularity due to the fact that they can be easily evaluated, differentiated and integrated. On the other hand, there are some limitations that become crucial in application to several classes of problems. Namely, if the function that is to be approximated behaves badly anywhere across the interval to be approximated then the approximation is of poor quality everywhere in this interval [25]. This leads to the idea of piecewise polynomial interpolation. The basic definitions connected with the interpolation with use of piecewise polynomial functions (PP functions) are presented below.

It is important to mention here that we define the **degree of polynomial** as the highest degree of its terms when the polynomial is expressed in canonical form consisting of a linear combination of monomials, while the **order of polynomial** is the highest degree among the basis functions used to express the polynomial in terms of a given polynomial vector space. It is, therefore, relative to the basis of the polynomial vector space in which the considered polynomial is included [25].

Let $\boldsymbol{\xi} = (\xi_j)_1^{l+1}$ denote a strictly increasing sequence of points and let k be a positive integer. If $P_1, ..., P_l$ is any sequence of l polynomials, each of order p, then we define the corresponding **piecewise polynomial (PP) function f of order** p as [26]:

$$f(x) = P_j(x) \quad \text{if } \xi_j < x < \xi_{j+1} \quad \text{for } j = 1, ..., l \tag{2.1}$$

The points ξ_j are called **breaks** of f. It is important to mention that for the breaks $\xi_2, ..., \xi_l$ function f has two values: $f(\xi_j^+)$ and $f(\xi_j^-)$. The space of all the PP functions of order p for a break sequence $\boldsymbol{\xi}$ will be denoted by $\Pi_{<p,\boldsymbol{\xi}}$.

In many cases there are additional constraints concerning the continuity between pieces

of the function, i.e. [26]:

$$D^{i-1}f(\xi_j^+) = D^{i-1}f(\xi_j^-) \quad \text{for } i = 1, ..., v_j, \quad j = 2, ..., l \tag{2.2}$$

for some vector $\mathbf{v} = (v_j)_2^l$ of non-negative integers. The operator D^i denotes the i-th derivative. Each component v_j of the vector corresponds to the required continuity at break ξ_j. The subspace of $\Pi_{<p,\boldsymbol{\xi}}$ which fulfils these conditions is denoted by $\Pi_{<p,\boldsymbol{\xi},\mathbf{v}}$ [25].

It would be very convenient to find a universal way to define the basis for arbitrary $\Pi_{<p,\boldsymbol{\xi},\mathbf{v}}$. This leads to the idea of **basis splines** or **B-splines**.

2.2 B-splines

2.2.1 B-spline basis functions

The B-splines were originally defined with the use of divided differences. Let $\boldsymbol{\xi} = (\xi_j)$ be a non-decreasing sequence of **knot values**. The j-th B-spline of order p for the knot sequence $\boldsymbol{\xi}$ is defined as [26]:

$$N_{j,p,\boldsymbol{\xi}}(x) = (\xi_{j+p} - \xi_j)[\xi_j, ..., \xi_{j+p}](\cdots - x)_+^{p-1} \quad \forall x \in \mathbb{R} \tag{2.3}$$

It can be proven that every space $\Pi_{<p,\boldsymbol{\xi},\mathbf{v}}$ has a basis consisting of B-splines defined above [25].

For practical reasons another way of defining the B-splines is more useful. The **Cox-de Boor formula** is a recursive formula allowing to define and evaluate B-splines. Again, let $\boldsymbol{\xi} = (\xi_j)$ be a non-decreasing sequence of knot values. The recursive definition starts with piecewise constant functions ($p = 0$) [1]:

$$N_{i,0,\boldsymbol{\xi}}(x) = \begin{cases} 1 & \text{if } \xi_i \leq x < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.4}$$

For $p > 0$ they are defined by [1]:

$$N_{i,p,\boldsymbol{\xi}}(x) = \frac{x - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1,\boldsymbol{\xi}}(x) + \frac{\xi_{i+p+1} - x}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1,\boldsymbol{\xi}}(x) \tag{2.5}$$

For the simpler representation the index $\boldsymbol{\xi}$ in $N_{i,p,\boldsymbol{\xi}}(x)$ is often omitted.

Knot vectors can contain the same values multiple times. Then the **multiplicity** m_i is the number of occurrences of the i-th knot. A knot vector is called **open** if the first and the last knot value appears $p + 1$ times. The knot vectors are called **uniform** if all the knots are equally spaced and **non-uniform** otherwise. In case of a uniform knot vector the internal basis functions are just translates of each other [1].

Fig. 2.1 presents example: basis functions of orders $p = 0, 1, 2$ for an open, uniform knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 2, 3, 4, 4, 4]$.

While building the basis functions from vectors containing knots with multiplicities higher than one a 0 will occur in the denominator in the formula (2.5). In this case we need to introduce the convention that each term with denominator equal to 0 is equal to 0.

It is important at this point to mention several properties of B-spline basis functions. First of all, each basis function is point-wise non-negative over the whole domain,

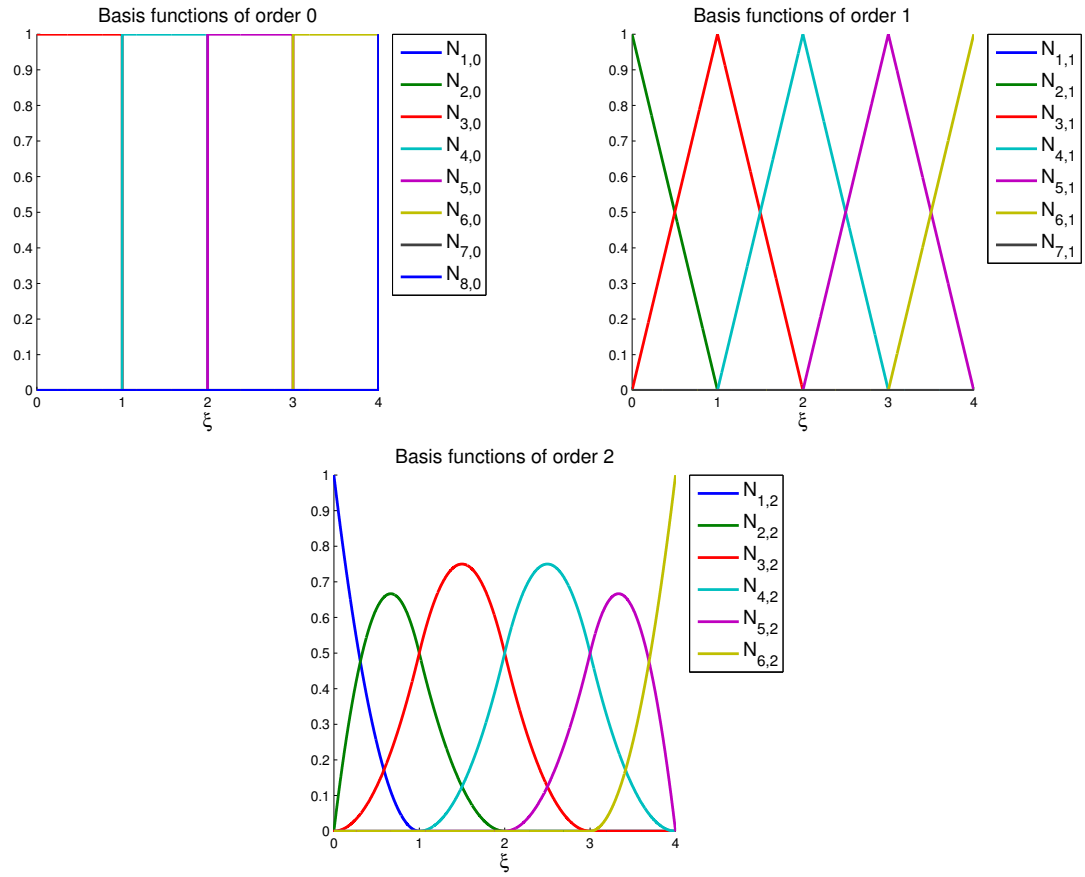


Figure 2.1: Basis function of orders $p = 0, 1, 2$ for open, uniform knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 2, 3, 4, 4, 4]$.

i.e. $N_{i,p}(x) \geq 0 \, \forall \, x, i$ [1]. The second important property of B-splines is the partition of unity, i.e. [1]:

$$\sum_{i=1}^{n} N_{i,p}(x) = 1 \;\; \forall \; x \in [\xi_1, \xi_{n+p+1}] \tag{2.6}$$

The third important property is the size and the placement of the **support of the function** (values of x for which the function is non-zero). The support of B-spline basis functions of order p is always $p + 1$ **knot spans** (intervals between knots, in case of repeated knots some spans can be of length zero). More precisely [1]:

$$N_{i,p}(x) \neq 0 \;\; \text{iff} \;\; \xi_i < x < \xi_{i+p+1} \tag{2.7}$$

A very important property of B-splines is a connection between continuity at knots and their multiplicity. For B-splines of order p at the knot with multiplicity m_i the basis functions have $p - m_i$ continuous derivatives. For example for the uniform knot vector (with multiplicities 1 for each knot value) we have at least C^{p-1} continuity everywhere in the interior of the interval. On the other hand, for open knot vector we have C^{-1} continuity on both ends of the interval which means that the functions are discontinuous at those points which is a natural termination of a domain [1].

Fig. 2.2 presents another example: quadratic basis functions for an open, non-uniform knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 2, 2, 3, 4, 5, 5, 6, 6, 6]$. It is easy to see that the functions are C^1-continuous at knots 1, 3 and 4 , where the multiplicity is 1 and only C^0-continuous at knots 2 and 5 where the multiplicity is 2.

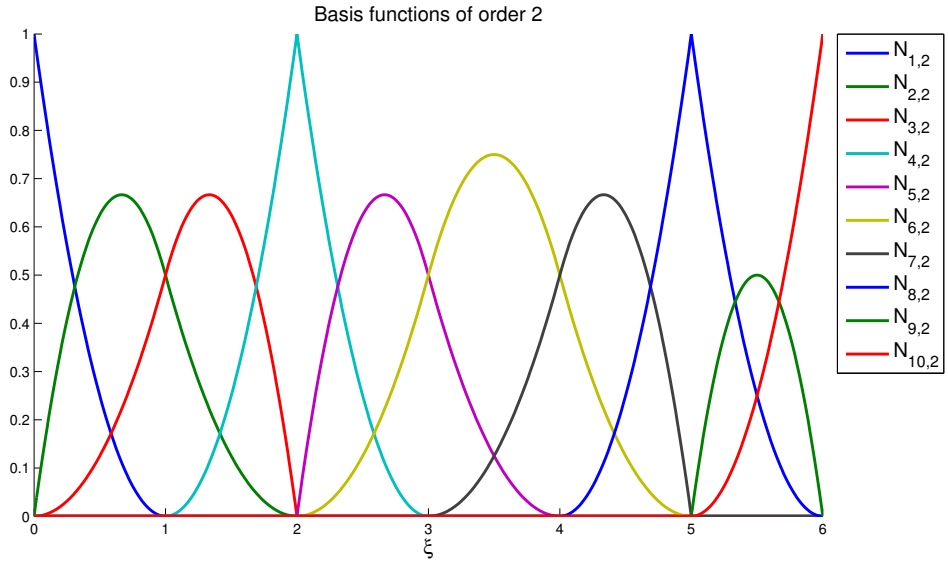


Figure 2.2: Quadratic basis functions for open, non-uniform knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 2, 2, 3, 4, 5, 5, 6, 6, 6]$.

The last important property is that derivatives of B-spline basis functions can be easily represented in terms of lower-order bases. It comes directly from the recursive definition (2.5). The derivative of the i-th basis function of order p is given by [1]:

$$\frac{d}{dx}N_{i,p}(x) = \frac{p}{\xi_{i+p} - \xi_i}N_{i,p-1}(x) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}}N_{i+1,p-1}(x) \tag{2.8}$$

This property can be easily generalized for higher order derivatives [1].

2.2.2 B-spline geometries

First of all, we consider B-spline curves in $\mathbb{R}^d$. They are constructed by taking a linear combination of the basis functions. The coefficients are vector-valued and they are called **control points**.

For a given basis consisting of n functions $N_{i,p}(\xi)$ of order p defined with the use of knot vector $\boldsymbol{\xi}$ and corresponding n control points $\mathbf{B_i} \in \mathbb{R}^d$, $i = 1, ..., n$ a **B-spline curve** is given by [1]:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\mathbf{B_i} \tag{2.9}$$

It is important to mention that from now on x will not be used any more to denote coordinates in the parameter space to avoid confusion with the naturally used notation for the physical space, i.e. x, y, z. This is why we will use ξ, η, ζ to denote the coordinates in the parameter

space. We also keep the index i in $\mathbf{B_i}$ bold to underline the fact that it denotes which control point is considered and is not referring to one of its components. Piecewise linear interpolation of control points is called a **control polygon** [1].

Fig. 2.3 presents the example of the B-spline curve created from the basis functions shown in Fig. 2.2 and the control polygon: $\mathbf{B}_1 = [0,0]^T$, $\mathbf{B}_2 = [3,3]^T$, $\mathbf{B}_3 = [5,1]^T$, $\mathbf{B}_4 = [7,1]^T$, $\mathbf{B}_5 = [7,5]^T$, $\mathbf{B}_6 = [5,5]^T$, $\mathbf{B}_7 = [0,3]^T$, $\mathbf{B}_8 = [1,3]^T$, $\mathbf{B}_9 = [0,2]^T$, $\mathbf{B}_{10} = [0,1]^T$. At the endpoints the curve is C^{-1}-continuous which is a physical termination of the curve and makes it interpolatory at the endpoints. It can be easily seen that at the points corresponding to the knot values with multiplicity 2 the curve is only C^0-continuous, while at all other points it is at least C^1-continuous. The knots mapped onto the physical space partition the curve into elements (knot spans).

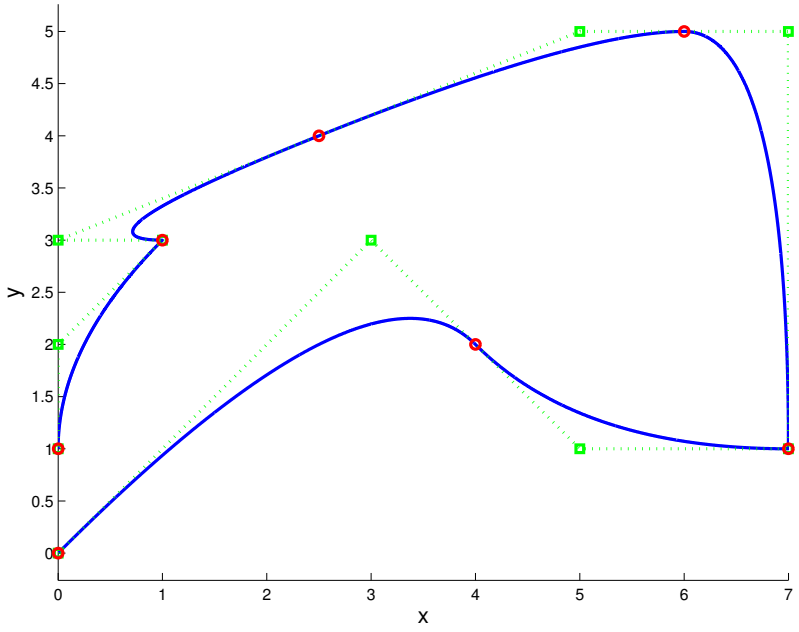


Figure 2.3: The B-spline curve created from quadratic basis functions shown in Fig. 2.2 and the control polygon: $\mathbf{B}_1 = [0,0]^T$, $\mathbf{B}_2 = [3,3]^T$, $\mathbf{B}_3 = [5,1]^T$, $\mathbf{B}_4 = [7,1]^T$, $\mathbf{B}_5 = [7,5]^T$, $\mathbf{B}_6 = [5,5]^T$, $\mathbf{B}_7 = [0,3]^T$, $\mathbf{B}_8 = [1,3]^T$, $\mathbf{B}_9 = [0,2]^T$, $\mathbf{B}_{10} = [0,1]^T$. The blue line represent the curve, the control polygon with control points is marked with green colour. The red points on the curve correspond to the knots mapped to the physical space.

There are several useful properties of B-spline curves that gave this method of representing geometries wide popularity in Computer Aided Design (CAD). First of all, the affine transformation of a B-spline curve (for example translation, rotation, scaling, stretching or shearing) can be done by just applying it to the control points of the curve. This property is called **affine covariance**. Due to the compact support of the basis functions, moving a single control point affects no more than $p+1$ knot spans of the curve. Finally, the curve has $p - m_i$ continuous derivatives at the given knot [1].

Let us consider B-spline surfaces in $\mathbb{R}^d$. Now $\mathbf{B_{i,j}}$, $i = 1, ..., n$, $j = 1, ..., m$ form a **control net**. For knot vectors $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$, n basis functions of order p, $N_{i,p}(\xi)$ and m basis functions of order q, $M_{j,q}(\eta)$, the **B-spline surface** is defined as [1]:

$$\mathbf{S}(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \mathbf{B_{i,j}} \tag{2.10}$$

Fig. 2.4 represents the turbine blade profile described by quadratic B-splines. Knot vectors $\boldsymbol{\xi} = [0, 0, 0, 0.5, 1, 1, 1]$ and $\boldsymbol{\eta} = [0, 0, 0, 1, 1, 1]$ were used to generate the basis. The control net is given by the points presented in the table below:

	j=1	j=2	j=3
i=1	(1.45,0.40)	(-3.55,0.30)	(1.45,-0.70)
i=2	(1.95,0.42)	(3.45,0.00)	(3.45,-1.00)
i=3	(5.95,0.50)	(6.25,0.35)	(6.95,0.30)
i=4	(8.45,1.80)	(8.50,1.80)	(8.45,1.65)

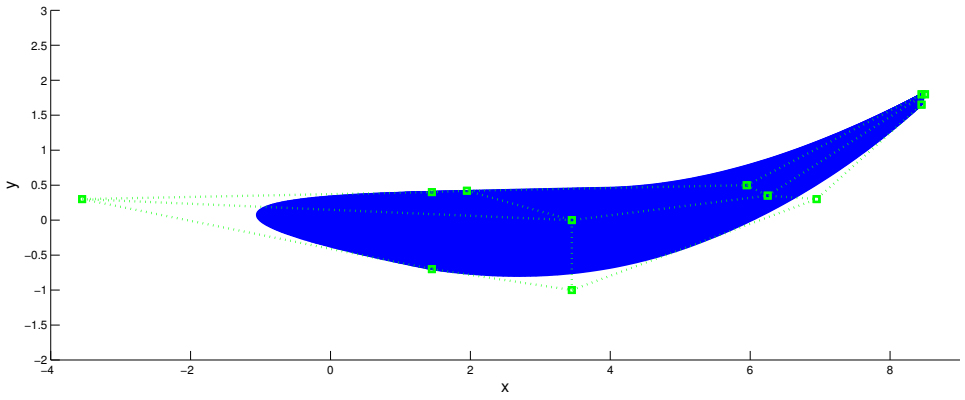


Figure 2.4: The B-spline surface representing the profile of the turbine blade.

Several important properties of B-spline surfaces come from their tensor product nature. The basis, as in case of B-spline curves, is point-wise non-negative and fulfils the partition of unity property, i.e. [1]:

$$\sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) = 1 \quad \forall\, (\xi, \eta) \in [\xi_1, \xi_n + p + 1] \times [\eta_1, \eta_m + q + 1] \tag{2.11}$$

The continuity in given parametric direction can be determined using the knot vector in this direction in the same way as in case of B-spline curves. The support is still compact for $N_{i,p}(\xi) M_{j,q}(\eta)$ it is exactly $[\xi_i, \xi_{i+p+1}] \times [\eta_j, \eta_{j+q+1}]$ [1].

Finally, let us consider B-spline solids. Now $\mathbf{B_{i,j,k}}$, $i = 1, ..., n$, $j = 1, ..., m$, $k = 1, ..., l$ form a **control lattice**. For knot vectors $\boldsymbol{\xi}$, $\boldsymbol{\eta}$ and $\boldsymbol{\zeta}$, n basis functions of order p, $N_{i,p}(\xi)$, m basis functions of order q, $M_{j,q}(\eta)$ and l basis functions of order r, $L_{k,r}(\zeta)$ the **B-spline solid** is defined as [1]:

$$\mathbf{S}(\xi, \eta, \zeta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{l} N_{i,p}(\xi) M_{j,q}(\eta) L_{k,r}(\zeta) \mathbf{B_{i,j,k}} \tag{2.12}$$

Fig. 2.5 presents the B-spline solid representing the shape of a twisted turbine blade. There are three layers (cross-sections) of control lattice at $z = 0, 1, 2$ and each of them is the same as the control net in the Fig. 2.4 and only to the one at $z = 2$ the rotation of $\pi/10$ around $(0, 0)$ is applied. The knot vector in the ζ direction was $\boldsymbol{\zeta} = [0, 0, 0, 1, 1, 1]$ while the two knot vectors in the other directions were the same as in the previous example.

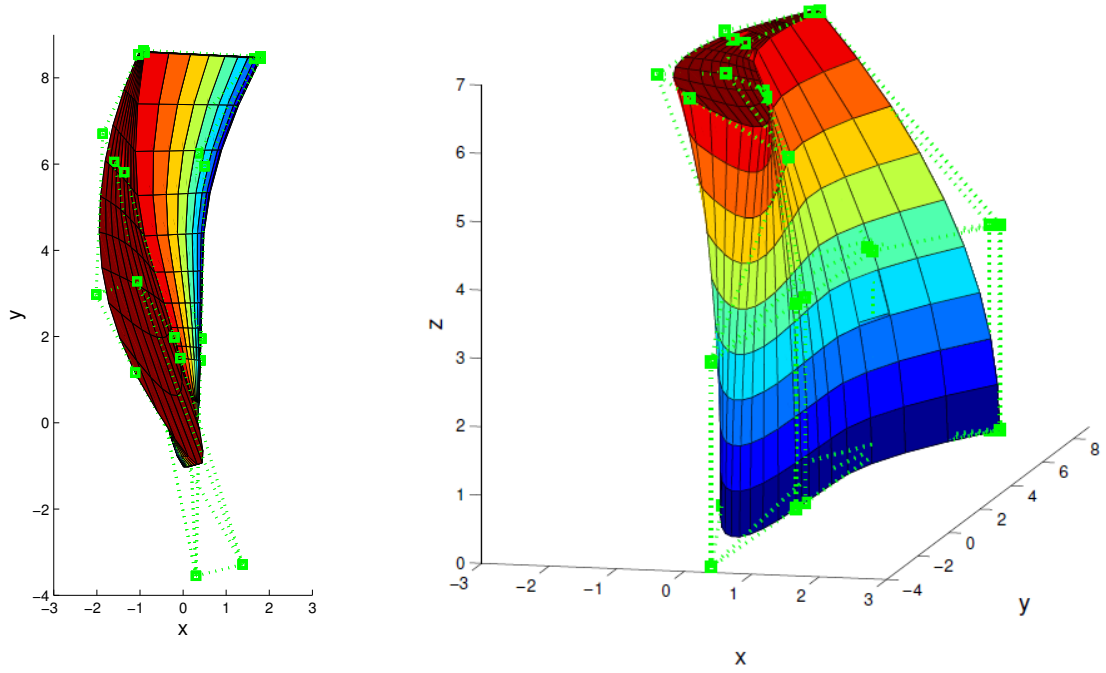


Figure 2.5: The B-spline solid representing the twisted turbine blade.

This is a trivariate generalization of the B-spline surfaces. Therefore, all properties of the B-spline solids are generalizations of the properties of B-spline surfaces [1].

2.2.3 h-, p-, k-refinements of B-splines

The possibility of increasing the order of the basis functions and of inserting additional knots to the knot vector without changing the shape of the curve (surface, solid) is a very important property of B-splines. In other words, we can enrich the basis without changing the shape of the geometry.

The first operation that enriches the basis is **knot insertion**. Inserting additional l knots to the knot vector will result in additional l basis functions (degrees of freedom, abbreviated later as DOFs). This operation correspond to the classical **h-refinement**. It is important to mention that those two operations, although very similar, are not identical. To reproduce the classical h-refinement new knots must be inserted with multiplicity p to achieve C^0 continuity across the new boundaries [1].

Knot insertion can be performed as follows. First of all, we create an extended knot vector $\tilde{\boldsymbol{\xi}} = (\tilde{\xi}_j)$ which consist of all elements of the $\boldsymbol{\xi} = (\xi_j)$ and l new knots to be inserted. The first and the last element of the vector should be the same as in the original knot vector $\boldsymbol{\xi}$. The $n + l$ new basis functions are created with recursion formulas (2.4) and (2.5). The $n + l$

new control points $\tilde{\mathbf{B}}$, that will keep the geometry unchanged, are a linear combination of the original control points $\mathbf{B}$ [1]:

$$\tilde{\mathbf{B}} = L^p \mathbf{B} \tag{2.13}$$

where L^p can be obtained by recursion:

$$L^0_{i,j} = \begin{cases} 1 & \text{if } \tilde{\xi}_i \in [\xi_j, \xi_{j+1}) \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

and for $q = 1, ..., p-1$:

$$L^{q+1}_{i,j} = \frac{\tilde{\xi}_{i+q} - \xi_j}{\xi_{j+q} - \xi_j} L^q_{i,j} + \frac{\xi_{j+q+1} - \tilde{\xi}_{i+q}}{\xi_{j+q+1} - \xi_{j+1}} L^q_{i,j+1} \tag{2.15}$$

It is allowed to increase the multiplicity of the knots to intentionally reduce the continuity of the basis at those knots. By following the procedure described above the continuity of the curve will be however preserved [1].

The left part of Fig. 2.6 shows the curve defined with knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 1, 1]$ and the control points $\mathbf{B}_1 = [0, 0]^T$, $\mathbf{B}_2 = [0.5, 1]^T$, $\mathbf{B}_3 = [1, 0]^T$. The curve, the new knots and the new control polygon obtained after inserting a new knot at $\xi = 0.5$ are presented in the right part of the figure. It can be easily seen that the curve is not modified by this operation. One new control point was added to the control net (as a response for one new basis function).

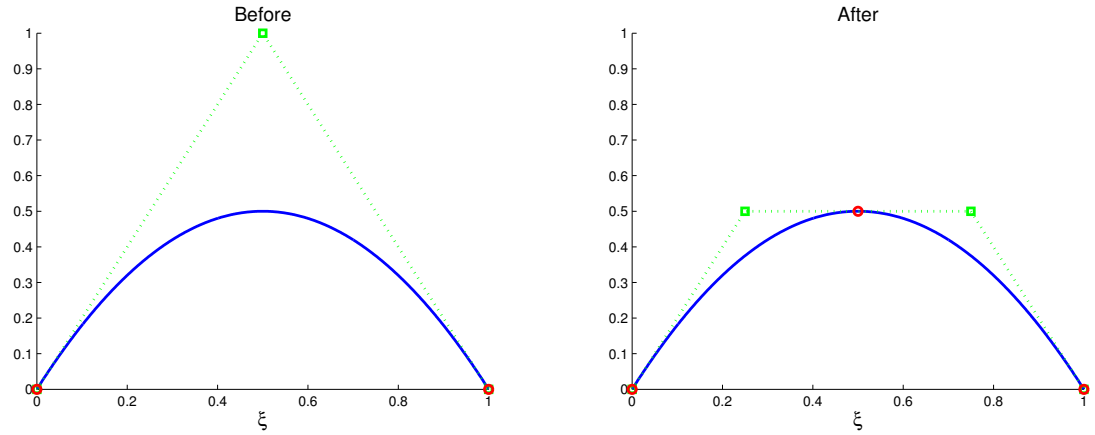


Figure 2.6: The knot insertion at 0.5 into the curve defined originally with the knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 1, 1]$, $p = 2$ and the control points $\mathbf{B}_1 = [0, 0]^T$, $\mathbf{B}_2 = [0.5, 1]^T$, $\mathbf{B}_3 = [1, 0]^T$.

Another way to enrich the basis is **order elevation**. This results in a richer basis of higher order. When we increase the order from p to $p+1$ we need to increase the multiplicity of every knot by one to preserve the continuity properties of the curve (for example discontinuities). Assuming that we have n distinct knot values in the original knot vector, we have to add n new knots (as multiplicities) which results with $n-1$ new degrees of freedom (one is lost as p is increased by one). The procedure to obtain the new control points is not as simple as in the case of knot insertion. The description of several approaches can be found in [27]. Order elevation corresponds to the classical **p-refinement** as it increases the order of the basis. The main difference is that the classical p-refinement starts with C^0 at every element boundary

which is not a necessity in case of the order elevation which works for all combination of continuities [1].

The left part of Fig. 2.7 presents again the curve defined with knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 1, 1]$ and the control points $\mathbf{B}_1 = [0, 0]^T$, $\mathbf{B}_2 = [0.5, 1]^T$, $\mathbf{B}_3 = [1, 0]^T$. The curve, the new knots and the new control polygon obtained after elevating the order from $p = 2$ to $p = 3$ are presented in the right part of the figure. It can be easily seen that the curve was not changed during this operation. One new control point was added to the control net (as a response for one new basis function).

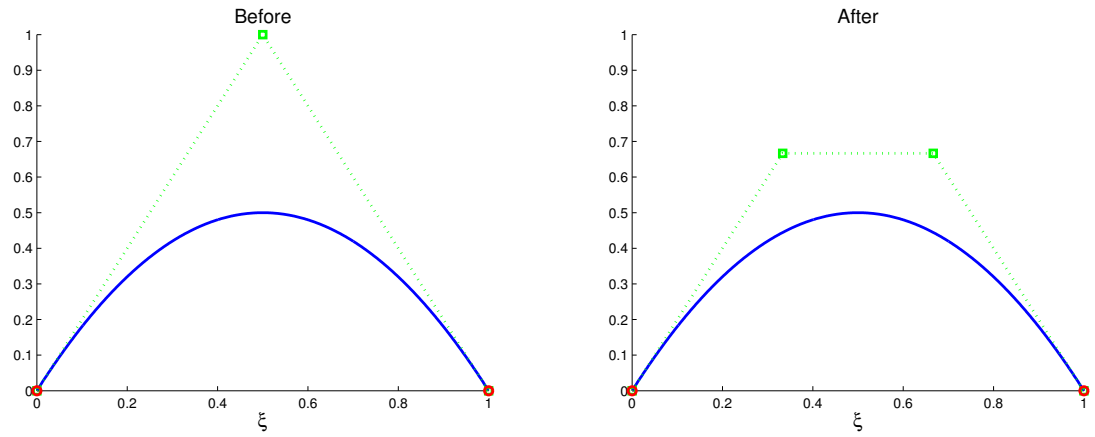


Figure 2.7: The order elevation from $p = 2$ to $p = 3$ for the curve defined originally with the knot vector $\boldsymbol{\xi} = [0, 0, 0, 1, 1, 1]$ and the control points $\mathbf{B}_1 = [0, 0]^T$, $\mathbf{B}_2 = [0.5, 1]^T$, $\mathbf{B}_3 = [1, 0]^T$.

In case of B-splines order elevation and knot insertion do not commute. For example if we firstly insert a new knot value $\tilde{\xi}$ to a curve of order p then we have C^{p-1} continuity at $\tilde{\xi}$ and then we increase the order to q such that the continuity is preserved, we get still C^{p-1} at $\tilde{\xi}$. Changing the order of operations, we firstly increase the order to q and then insert the new knot value $\tilde{\xi}$. Then, the result of those operations is C^{q-1} at $\tilde{\xi}$ [1].

The latter approach (firstly increase the order and then insert the new knot) is called **k-refinement**. This technique is unique for isometric analysis and has no analogue in classical FEA. It results in both higher order and higher continuity.

2.3 NURBS

Although in this thesis we will restrict the discussion to B-splines it is important to mention that there are some geometries that can not be represented exactly with the use of B-splines. An example of such geometry is a circle. Therefore, a further development has been done which resulted in defining **Non-Uniform Rational B-Splines (NURBS)**. NURBS allow to represent much wider array of shapes than B-splines. It is important to note here that the theory and implementations presented in the further chapters of this thesis need to be carefully generalized to NURBS. This generalization is not in the scope of this thesis and therefore the methods presented in the thesis are not proven to work for NURBS directly.

Construction of NURBS can be easily understood as a projective transformation of B-splines in $\mathbb{R}^{d+1}$ to $\mathbb{R}^d$ (see Fig. 2.8). We denote the B-spline curve, called **projective curve**, by $\mathbf{C}^w(\xi)$ and its control points, called **projective control points**, by $\mathbf{B}_i^w$ while the NURBS

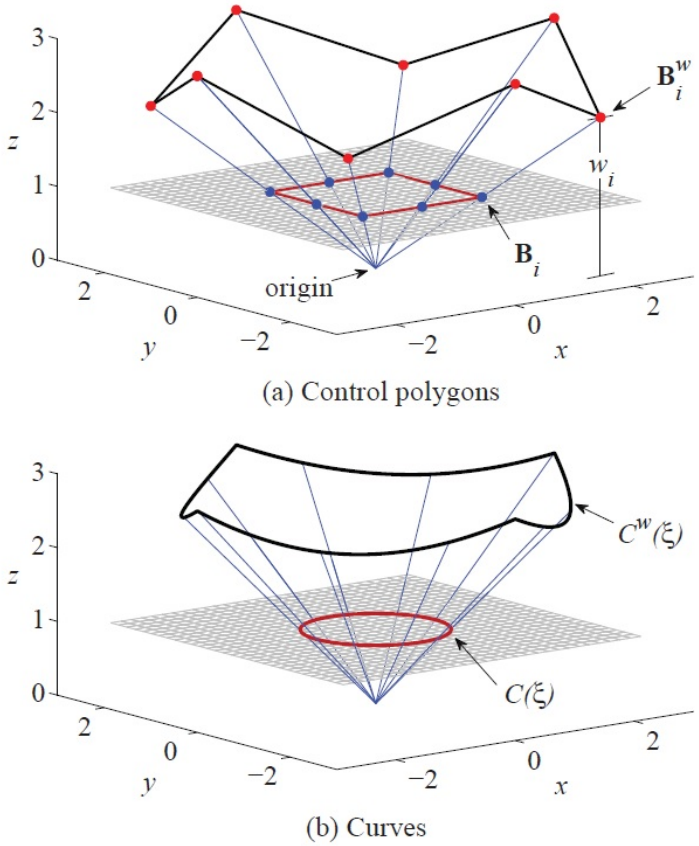


Figure 2.8: A circle in $\mathbb{R}^2$ constructed by the projective transformation of a piecewise quadratic B-spline in $\mathbb{R}^3$. Source: [1].

curve is denoted by $\mathbf{C}(\xi)$ and its control points by $\mathbf{B}_i$. Control points of the NURBS curve can be obtained from the control points of the B-spline curve by following relation [1]:

$$(\mathbf{B}_i)_j = (\mathbf{B}_i^w)_j / w_i \quad j = 1, ..., d \tag{2.16}$$

$$w_i = (\mathbf{B}_i^w)_{d+1} \tag{2.17}$$

where $(\mathbf{B}_i)_j$ is the j-th component of vector $\mathbf{B}_i$ and w_i is a corresponding so-called **weight** (positive in most of the practical applications). We define the **weighting function** as [1]:

$$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i \tag{2.18}$$

where $N_{i,p}(\xi)$ are the standard B-spline basis functions. Now, we can define the NURBS basis functions as [1]:

$$R_i^p(\xi) = \frac{N_{i,p}(\xi) w_i}{W(\xi)} \tag{2.19}$$

The **NURBS curve** can be obtained as [1]:

$$(\mathbf{C}(\xi))_j = \frac{(\mathbf{C}^w(\xi))_j}{W(\xi)} \quad j = 1, ..., d \tag{2.20}$$

or as:

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} R_i^p(\xi)\mathbf{B}_i \tag{2.21}$$

The latter formula is the one used in practice. Analogously, we can obtain **NURBS surfaces** and **NURBS solids** using the rational basis functions, using respectively [1]:

$$R_{i,j}^{p,q}(\xi, \eta) = \frac{N_{i,p}(\xi)M_{j,q}(\eta)w_{i,j}}{\sum_{\hat{i}=1}^{n}\sum_{\hat{j}=1}^{m} N_{\hat{i},p}(\xi)M_{\hat{j},q}(\eta)w_{\hat{i},\hat{j}}} \tag{2.22}$$

$$R_{i,j,k}^{p,q,r}(\xi, \eta, \zeta) = \frac{N_{i,p}(\xi)M_{j,q}(\eta)L_{k,r}(\zeta)w_{i,j,k}}{\sum_{\hat{i}=1}^{n}\sum_{\hat{j}=1}^{m}\sum_{\hat{k}=1}^{l} N_{\hat{i},p}(\xi)M_{\hat{j},q}(\eta)L_{\hat{k},r}(\zeta)w_{\hat{i},\hat{j},\hat{k}}} \tag{2.23}$$

This is only a very basic introduction to NURBS and as they are not in the scope of this thesis more detailed explanation will be omitted. For deeper description see [28].

Chapter 3

B-splines as basis for analysis

The purpose of this chapter is to present the basic idea of isogeometric analysis (IGA) and differences and similarities of IGA and FEA. First of all, the general idea of the isogeometric analysis is presented. Further parts of the chapter are devoted to application of IGA to simple elliptic model problems, i.e. Poisson's problem on the unit interval, Poisson's problem on the unit square and Poisson's problem on an arbitrary surface in 2D.

3.1 General idea

The isogeometric analysis is, like FEM, a numerical method for solving partial differential equations (PDEs). As with the FEM, we develop the variational form of the problem and seek for the approximate solution of this problem in some finite dimensional function space. The general idea of the isogeometric analysis is to replace the classical FEM function spaces (in FEA) by B-spline (or NURBS) spaces [4].

Both in FEA and IGA, the first step is to convert the PDE problem into its weak form, also called the variational formulation, which is then approximated with use of finite-dimensional function spaces. The class of methods that deals with this step is often referred to as Galerkin methods. Among them the standard Galerkin method is the method which is the most widely used as a basis for IGA as well as for FEA. This approach is firstly presented on an abstract problem and, in further part of this chapter, on Poisson's problem. The main idea of this short abstract description is to give a general overview rather than presenting the formal derivation.

First of all, the weak formulation of the problem is derived, i.e.:

$$\text{find } u \in S \text{ such that for all } v \in V, \;\; a(u,v) = f(v). \tag{3.1}$$

where S is called the **trial space** and V is called the **test space**. The weak form is obtained (for linear problems) by multiplying the original problem by a **test function** $v \in V$ and, depending on the type of problem, performing integration by parts. It is important to take care of the boundary conditions at this point. The trial space S and the test space V are the Hilbert spaces (with some additional restrictions caused by Dirichlet boundary conditions).

The next step is to construct the finite-dimensional approximations of S and V, i.e. $S^h \subset S$ and $V^h \subset V$. Additionally, for the standard Galerkin method we assume: $S^h = V^h$. Now, we can formulate the discrete variational problem:

$$\text{find } u^h \in V^h \text{ such that for all } v^h \in V^h, \;\; a(u^h, v^h) = f(v^h). \tag{3.2}$$

Note that the variational forms presented in this chapter are the simplest possible and they do not involve the terms that occur because of the treatment of boundary conditions. Problem (3.2) can be rewritten in matrix form and easily solved numerically.

In this place FEA and IGA diverge. For FEA the space spanned by FEM basis functions is used as V^h while for IGA the space spanned by B-spline (NURBS) basis functions is used [2].

The difference between FEA and IGA that is worth mentioning here is the way the domain is divided into parts. In FEA the physical domain is divided into elements and each element has its own mapping from the reference element (see Fig. 3.1). It means that the parameter space is local to each element [1].

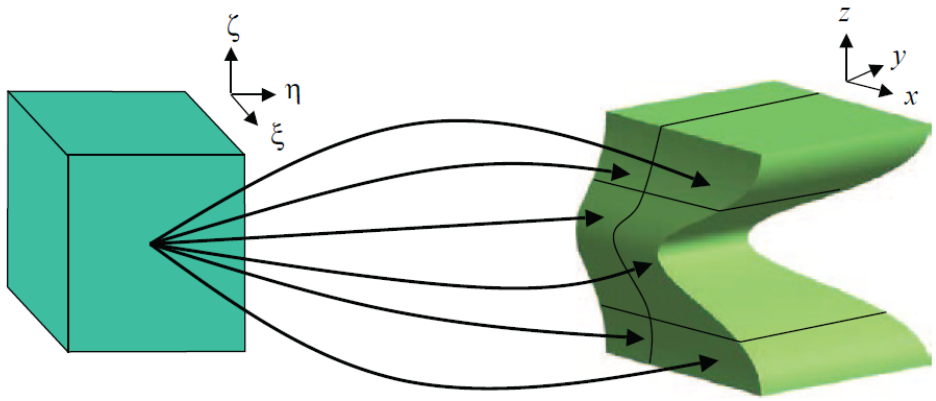


Figure 3.1: Mapping in classical FEA. The parameter space is local to elements. Source: [1].

On the other hand, in IGA the domain is divided into patches that are much larger in size than elements (even the entire domain can be one patch for simple geometries). Mapping takes the entire patch from parameter space to physical space (see Fig. 3.2). The parameter space is local to the patch. In IGA we also use sometimes the word elements to name the knot spans but it can be considered misleading because of this different nature of mapping [1].

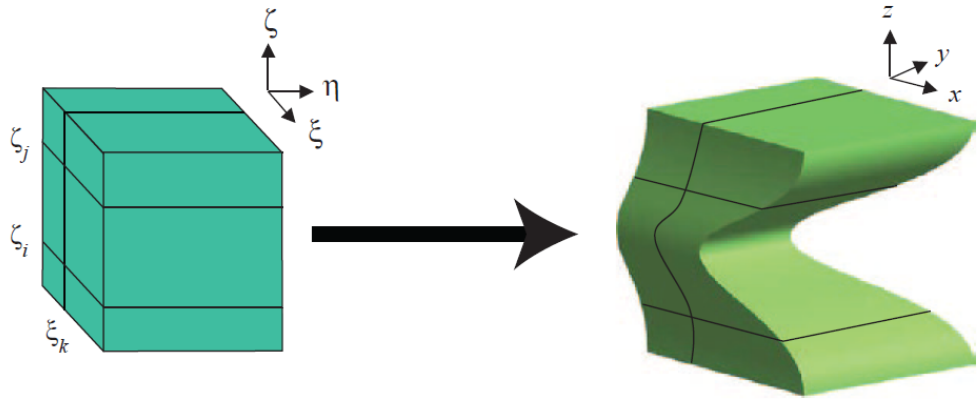


Figure 3.2: Mapping in IGA. The parameter space is local to entire patch. Source: [1].

This different nature of mapping causes the main bottle-neck of the isogeometric analysis. In FEM it is possible to evaluate integrals of basis functions on the reference element and map them to appropriate elements. This mapping is a relatively cheap operation. In IGA one needs to evaluate integrals for all the basis functions numerically. This approach is quite expensive in terms of computational time.

3.2 Poisson's problem in one dimension

The first example to which we apply IGA is the Poisson problem on the unit interval:

$$\begin{aligned} -\frac{d^2u}{dx^2} &= f(x) && \text{in } \Omega = (0,1), \\ u(0) &= 0 && \text{at } \Gamma_D = \{x = 0\}, \\ \frac{du}{dx}(1) &= 0 && \text{at } \Gamma_N = \{x = 1\}, \end{aligned} \tag{3.3}$$

with load vector $f(x) = 6(x+1)$.

First of all, we need to derive the weak form for this problem. Note that both Dirichlet and Neumann boundary condition are homogeneous so neither the Dirichlet lift nor the integral term resulting from the Neumann boundary condition have to be considered. It is easy to see that the weak form of this problem is:

$$\text{find } u \in S \text{ such that for all } v \in V, \;\; a(u,v) = f(v). \tag{3.4}$$

where:

$$a(u,v) = \int_0^1 \frac{du}{dx}\frac{dv}{dx}\,dx \tag{3.5}$$

$$f(v) = \int_0^1 fv\,dx. \tag{3.6}$$

The discrete weak form of this problem, using the same space as trial and test spaces, reads:

$$\text{find } u^h \in V^h \text{ such that for all } v^h \in V^h, \;\; a(u^h,v^h) = f(v^h). \tag{3.7}$$

where:

$$a(u^h,v^h) = \int_0^1 \frac{du^h}{dx}\frac{dv^h}{dx}\,dx \tag{3.8}$$

$$f(v^h) = \int_0^1 fv^h\,dx. \tag{3.9}$$

Analogously as for standard FEM, knowing that $u^h = \sum_{i=1}^n u_i\varphi_i$ (where n is the number of DOFs and φ_i is the i-th basis function (degree of freedom) of the considered trial/test space) and that it is enough to only use basis functions of V^h as test functions v, the discrete weak form can be rewritten in form of the linear system:

$$A\mathbf{u} = \mathbf{b} \tag{3.10}$$

where the entries of matrix A are defined as $a_{ij} = a(\varphi_i, \varphi_j)$, entries of vector $\mathbf{b}$ as $b_i = f(\varphi_i)$ and $\mathbf{u}$ is a vector of unknown coefficients u_i of solution.

The matrix (and the RHS) is assembled in this example using the Gaussian quadrature of 14 points at each knot span (element). From the point of view of computational efficiency of the implementation, it is of utmost importance to consider in the integration at given knot span only the basis functions that have support there. For each of the integration points the basis functions and lower-order basis functions (used to evaluate the derivatives according to (2.8) are evaluated.

The boundary conditions are imposed explicitly in the code. For the homogeneous Neumann boundary condition no special treatment is necessary. For the Dirichlet boundary condition it is assured that all basis functions that have support at this boundary have to have the coefficient equal to 0. Namely, all the entries of matrix A on the rows corresponding to the degrees of freedom that are non-zero at the boundary were set to 0 except for the diagonal entry which was set to 1. The corresponding entry in the RHS was set to 0 as well. Note that in the given 1D problem there is only one basis function that is non-zero on the Dirichlet boundary.

The used basis functions are the B-spline basis functions for an uniform, open knot vector. The numerical results were compared with the exact solution:

$$u_{ex}(x) = -x^3 - 3x^2 + 9x \tag{3.11}$$

in terms of L2 norm defined as:

$$\|u_{ex} - u^h\|_2 := \left(\int_0^1 \left(u_{ex}(x) - u^h(x) \right)^2 dx \right)^{1/2} \tag{3.12}$$

The results for variable numbers of degrees of freedom and different orders of basis are shown in Fig. 3.3. We can see that the higher the order the better the convergence. For $p \geq 3$ we achieve solution of the maximal achievable quality (the order of machine representation) because the exact solution is a polynomial of order 3 so it can be exactly represented with B-spline basis functions of order 3 and the errors are only caused by finite precision of number representation.

3.3 Poisson's problem in two dimensions

In this section we present two examples of application of IGA for 2D problems. The first one is Poisson's problem with homogeneous Dirichlet boundary conditions on the unit square and the second one is the same problem but defined on an arbitrary 2D physical domain.

3.3.1 Poisson's problem on the unit square

Consider Poisson's problem on the unit square with homogeneous Dirichlet boundary conditions, i.e.:

$$\begin{aligned} -\Delta u(\mathbf{x}) &= f(\mathbf{x}) \qquad \text{in } \Omega = (0,1) \times (0,1), \\ u(\mathbf{x}) &= 0 \qquad \text{on } \partial\Omega, \end{aligned} \tag{3.13}$$

with the load vector $f(\mathbf{x}) = 2\pi^2 \sin(\pi x) \sin(\pi y)$. The vector of physical dimensions with entries (x, y) is denoted by $\mathbf{x}$.

The weak form is similar to that of the 1D case, i.e.:

$$\text{find } u \in S \text{ such that for all } v \in V, \;\; a(u, v) = f(v). \tag{3.14}$$

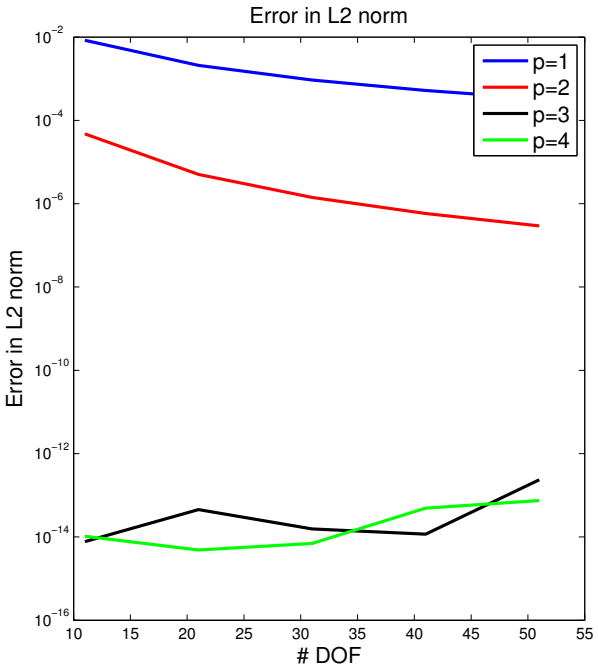


Figure 3.3: Error of numerical solution in L2 norm versus the number of DOFs n, for different orders p.

where:

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v \, d\mathbf{x} \tag{3.15}$$

$$f(v) = \int_\Omega f v \, d\mathbf{x}. \tag{3.16}$$

The discrete weak form, using the same discrete space as trial and test spaces, reads:

$$\text{find } u^h \in V^h \text{ such that for all } v^h \in V^h, \;\; a(u^h, v^h) = f(v^h). \tag{3.17}$$

where:

$$a(u^h, v^h) = \int_\Omega \nabla u^h \cdot \nabla v^h \, dx \tag{3.18}$$

$$f(v^h) = \int_\Omega f v^h \, dx. \tag{3.19}$$

It is worth mentioning that now the basis functions (DOFs) are resulting from the tensor product construction of B-spline basis functions in directions ξ and η. Again, the discrete weak form can be rewritten in terms of the linear system:

$$A\mathbf{u} = \mathbf{b} \tag{3.20}$$

where the entries of matrix A are defined as $a_{ab} = a(\varphi_a, \varphi_b)$, entries of vector $\mathbf{b}$ as $b_a = f(\varphi_a)$ and $\mathbf{u}$ is a vector of unknown coefficients of solution u_a. We need to define appropriate

indexing to be able to get the understanding what the a-th DOF is. Which basis functions in directions ξ and η are involved in constructing this basis function? Let us define it as follows:

$$a = n(j-1) + i \tag{3.21}$$

where i is the index of basis function in direction ξ and j in direction η. This corresponds to line-wise numbering of the DOFs starting with the x-direction [29].

During the matrix and RHS assembly the Gauss quadrature of 14 points in each direction was used. For all the knot spans in the domain the quadrature points were mapped to the span. Then all the basis functions and the lower-order basis functions were evaluated in all the quadrature points in the corresponding direction. From the point of view of efficiency of the code, it is important that the integrals used for the assembly were computed only for the tensor product basis functions that have support on the currently considered knot span. The treatment of the Dirichlet boundary conditions is analogous to the one used in the one-dimensional case.

The tensor product B-spline basis functions, defined with uniform, open knot vectors in both directions were used as basis functions. The numerical results were compared with the exact solution:

$$u_{ex}(\mathbf{x}) = \sin(\pi x)\sin(\pi y) \tag{3.22}$$

in terms of the maximum norm, defined as maximum of absolute values of differences of exact and numerical solutions.

The numerical results are visualised in Fig. 3.4. The number of basis functions in the horizontal direction were $n = 12$, in the vertical direction $m = 10$, the orders in both directions $p = q = 3$, the error in the maximum norm was $7.5681 \cdot 10^{-5}$.

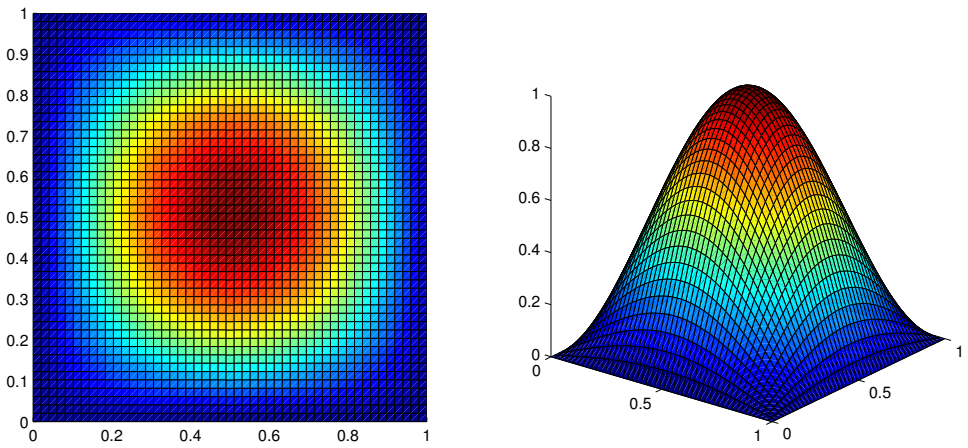


Figure 3.4: Numerical results for the Poisson problem with homogeneous Dirichlet BC, the load vector $f(\mathbf{x}) = 2\pi^2 \sin(\pi x)\sin(\pi y)$, computed with use of open uniform B-spline basis with $n = 12$, $m = 10$ and $p = q = 3$.

The convergence analysis was not performed due to very long computation time even for small number of degrees of freedom. At this point it became clear that more efficient assembly strategies than our naive Matlab implementation are required. Therefore, the G+SMO library was used for all further experiments of this work.

3.3.2 Poisson's problem on an arbitrary geometry in 2D

Consider the Poisson's problem on an arbitrary 2D domain (that can be directly mapped from the square parametric domain and therefore does not require representation with multiple patches) with homogeneous Dirichlet boundary conditions, i.e.:

$$\begin{aligned} -\Delta u(\mathbf{x}) &= f(\mathbf{x}) \qquad \text{in } \Omega, \\ u(\mathbf{x}) &= 0 \qquad\quad \text{on } \partial\Omega, \end{aligned} \tag{3.23}$$

with load vector $f(\mathbf{x}) = 2\pi^2 \sin(\pi\xi(\mathbf{x})) \sin(\pi\eta(\mathbf{x}))$ defined for practical reasons on the parametric domain, with coordinates $\boldsymbol{\xi} = (\xi, \eta)$. For consistency reasons in the definition of the load vector the parametric coordinates ξ and η depend on the physical coordinates x and y and therefore would have to be computed with the inverse mapping. In practice we will evaluate them only after transformation to the parametric domain.

The weak form and the discrete weak form are the same as for the unit square case. As defined in the previous chapter the geometrical mapping in 2D is given by:

$$\mathbf{S}(\boldsymbol{\xi}) = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta) \mathbf{B_{i,j}} \tag{3.24}$$

We will denote the parametric domain by Ω_0 to distinguish it from the physical domain Ω. To solve the weak problem numerically it is necessary first to transform the integrals from the physical domain to the parametric domain via the integration rule [29]:

$$\int_{\Omega} u(\mathbf{x}) d\mathbf{x} = \int_{\Omega_0} u(\boldsymbol{S}(\boldsymbol{\xi})) |\det \boldsymbol{DS}(\boldsymbol{\xi})| d\boldsymbol{\xi} \tag{3.25}$$

Here, $\boldsymbol{DS}(\boldsymbol{\xi})$ is the **Jacobian matrix of the geometrical mapping** defined as [29]:

$$\boldsymbol{DS}(\boldsymbol{\xi}) = \begin{bmatrix} \frac{\partial S_1}{\partial \xi} & \frac{\partial S_1}{\partial \eta} \\ \frac{\partial S_2}{\partial \xi} & \frac{\partial S_2}{\partial \eta} \end{bmatrix} \tag{3.26}$$

where S_i is the i-th component of the mapping and the individual entries are [29]:

$$\frac{\partial S_1}{\partial \xi} = \sum_{i=1}^{n} \sum_{j=1}^{m} N'_{i,p}(\xi) M_{j,q}(\eta) B^1_{i,j} \tag{3.27}$$

$$\frac{\partial S_1}{\partial \eta} = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M'_{j,q}(\eta) B^1_{i,j} \tag{3.28}$$

$$\frac{\partial S_2}{\partial \xi} = \sum_{i=1}^{n} \sum_{j=1}^{m} N'_{i,p}(\xi) M_{j,q}(\eta) B^2_{i,j} \tag{3.29}$$

$$\frac{\partial S_2}{\partial \eta} = \sum_{i=1}^{n} \sum_{j=1}^{m} N_{i,p}(\xi) M'_{j,q}(\eta) B^2_{i,j} \tag{3.30}$$

with $B^k_{i,j}$ being the k-th component of the $\mathbf{B_{i,j}}$ vector. We also need the chain rule of differentiation, namely [29]:

$$\nabla_{\mathbf{x}} u(\mathbf{x}) = \boldsymbol{DS}(\boldsymbol{\xi})^{-T} \nabla_{\boldsymbol{\xi}} u(\boldsymbol{\xi}) \tag{3.31}$$

where the subscript $\mathbf{x}$ in $\nabla_{\mathbf{x}}$ denotes that differentiation takes place in $\mathbf{x}$ coordinates. With all that we can rewrite the weak formulation as:

$$\text{find } u \in V \text{ such that for all } v \in V, \;\; a(u,v) = f(v). \tag{3.32}$$

where:

$$a(u,v) = \int_{\Omega_0} \nabla u(\boldsymbol{\xi})^T |\text{det}\boldsymbol{DS}(\boldsymbol{\xi})| (\boldsymbol{DS}(\boldsymbol{\xi})^{-1}\boldsymbol{DS}(\boldsymbol{\xi})^{-T}) \nabla v(\boldsymbol{\xi})\, d\boldsymbol{\xi} \tag{3.33}$$

$$= \int_{\Omega_0} \nabla u(\boldsymbol{\xi})^T \boldsymbol{G}(\boldsymbol{\xi}) \nabla v(\boldsymbol{\xi})\, d\boldsymbol{\xi} \tag{3.34}$$

$$f(v) = \int_{\Omega_0} v(\boldsymbol{\xi}) f(\boldsymbol{S}(\boldsymbol{\xi})) |\text{det}\boldsymbol{DS}(\boldsymbol{\xi})|\, d\boldsymbol{\xi}. \tag{3.35}$$

with $\boldsymbol{G}(\boldsymbol{\xi}) = |\text{det}\boldsymbol{DS}(\boldsymbol{\xi})|(\boldsymbol{DS}(\boldsymbol{\xi})^{-1}\boldsymbol{DS}(\boldsymbol{\xi})^{-T})$. We can discretize and rewrite the problem as the linear system:

$$A\mathbf{u} = \mathbf{b} \tag{3.36}$$

The matrix assembly process is similar to the one on the unit square except for the fact that the necessary geometry terms, i.e. $\boldsymbol{G}(\boldsymbol{\xi})$ and $|\text{det}\boldsymbol{DS}(\boldsymbol{\xi})|$ have to be evaluated at the quadrature points and taken into account during computing the integrals. The treatment of the Dirichlet boundary condition is analogous to the previous case.

The numerical experiment was performed using the own Matlab code and the G+SMO library [24, 30, 31] to compare the results. The problem was defined on the blade cross-section presented in the previous chapter. The used basis functions were tensor product basis functions defined by the geometry. No refinement was applied. The solution obtained from the Matlab code is presented in Fig. 3.5 and the solution from the G+SMO package in Fig. 3.6.

The results are consistent both qualitatively and quantitatively. In case of the Matlab code the maximum value of the solution is equal to 1.752 while for the G+SMO it is 1.788. It is a pretty good accuracy taking into account the fact that the mesh was very coarse (4×3 degrees of freedom). Again, the convergence analysis could not be performed due to the inefficiency of the Matlab code.

The numerical results presented in the following chapters are computed with solvers using functions of external library G+SMO [24, 30, 31]. Construction of the basis functions, geometry mappings and the assembly of standard matrices and vectors (consistent mass matrices, discrete transport operators, right hand side vectors) is always done with the help of the G+SMO package. Although, the package supports the treatment of boundary conditions as well, in this thesis work their treatment was implemented from scratch to assure that stabilization methods do not influence the consistency of boundary conditions. The resulting system is solved with use of the EIGEN package [32]. The same package is also providing the framework for operations on vectors, full and sparse matrices.

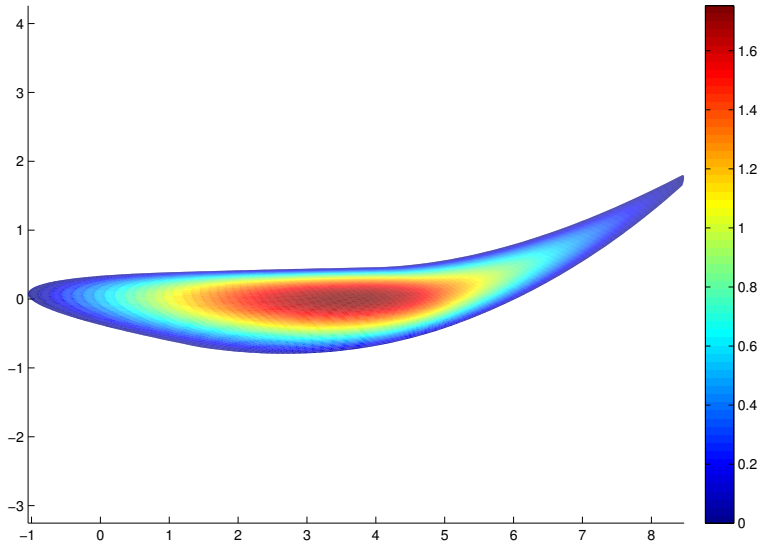


Figure 3.5: Numerical results from the Matlab code for the Poisson's problem with homogeneous Dirichlet BC, the load vector $f(\mathbf{x}) = 2\pi^2 \sin(\pi\xi)\sin(\pi\eta)$, defined on the turbine blade profile. Basis functions defined by geometry, no refinement.

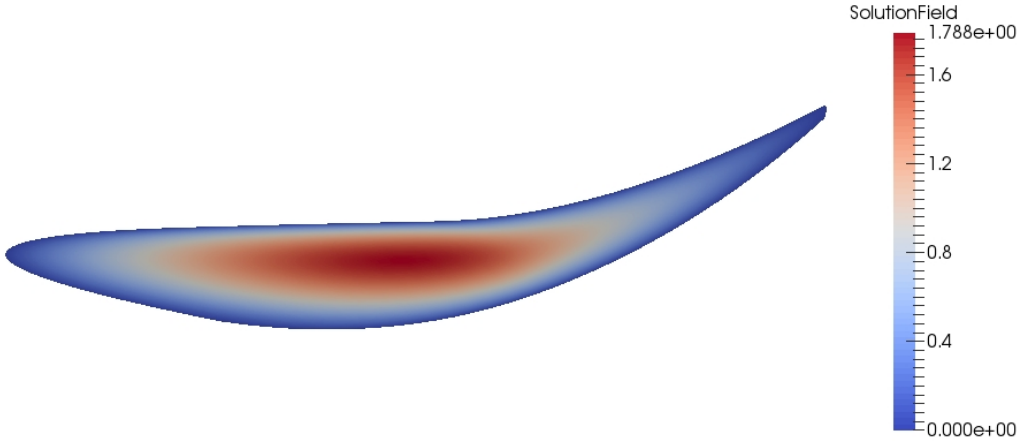


Figure 3.6: Numerical results from the G+SMO package for the Poisson's problem with homogeneous Dirichlet BC, the load vector $f(\mathbf{x}) = 2\pi^2 \sin(\pi\xi)\sin(\pi\eta)$, defined on the turbine blade profile. Basis functions defined by geometry, no refinement.

Chapter 4

Constrained L2 projection

This chapter is devoted to methods for projecting analytical functions to the space of functions used for solving the problem. In IGA and standard FEM the analytically described initial and boundary conditions (depending on the problem) have to be projected to the space spanned by B-splines (NURBS) basis or standard FEM functions. In the standard FEM with nodal degrees of freedom this task is often done by just assigning the nodal values $u_i^0 = u_0(\mathbf{x}_i)$. This approach has significant drawback: it is not conservative and therefore it can be inaccurate if the mesh is too coarse and the function is not smooth enough [33]. In case of non-nodal FEM and IGA it is impossible to use this method as (except of $p = 1$ B-splines) degrees of freedom are not associated with nodal values of the solution.

Therefore, we will use the L2 projection which is described in the first section of this chapter together with some numerical results that reveal the drawbacks of this approach. Further, the cure for those problems - constrained L2 projection is described. The last section contains a comparison of numerical results obtained using those two types of projection.

4.1 Standard L2 projection

The **L2 projection** (we will refer often to it as the standard L2 projection to distinguish it from the constrained L2 projection discussed in the next section) is a method that is used to project a function $f(\mathbf{x})$ defined in the function space A to the function space B. Assuming that $B \subset A$, the ideal projection would be the one that solves the problem (we omit the dependency of functions on $\mathbf{x}$ for clarity of presentation) [34]:

$$R(Pf) = Pf - f = 0, \quad \mathbf{x} \in \Omega \tag{4.1}$$

where P denotes the **projection operator** and $R(Pf)$ is so-called **residual** of the projection. It is easy to see that in most cases we are not able to achieve this goal ($R(Pf) = 0$), for example projection of x^2 from the space of all polynomial function of order 2 to the space of all polynomial functions of order 1 with $R(Pf) = 0$ can not be achieved. Therefore, let us weaken our goal and look for a projection that minimizes the residual $R(Pf)$. We know that the minimal possible residual is orthogonal to B, namely [34]:

$$(Pf - f, v) = 0, \quad \forall v \in B \tag{4.2}$$

where $(\cdot,\cdot)$ denotes the **inner product**. In case of L2 spaces, which are of interest for us, the inner product is defined as (assuming that we operate on real numbers):

$$(f,g) = \int_\Omega f(\mathbf{x})g(\mathbf{x})d\mathbf{x} \tag{4.3}$$

As the inner product is a bilinear operator we can reformulate equation (4.2) as [34]:

$$(Pf,v) = (f,v), \quad \forall v \in B \tag{4.4}$$

In FEM and IGA the L2 projection is widely used to project initial and boundary conditions to the space spanned by the particular basis functions. In this case A is an infinite function space denoted now by V and B is the space of FEM basis functions in case of standard FEM or B-splines (NURBS) in case of IGA, denoted by V^h. Then the objective reads:

$$(u^h,v^h) = (f,v^h), \quad \forall v^h \in V^h \tag{4.5}$$

where u^h is a representation of f in V^h space. We can rewrite the problem as:

$$\text{find } u^h \in V^h \text{ such that for all } v^h \in V^h, \;\; (u^h,v^h) = (f,v^h). \tag{4.6}$$

and knowing that u^h is a linear combination of basis functions φ_i of V^h, namely $u^h = \sum_{i=1}^n u_i\varphi_i$ we can rewrite the problem in the matrix-vector form:

$$M_C\mathbf{u} = \mathbf{b} \tag{4.7}$$

where $M_C = \{m_{ij}\}$ is a **consistent mass matrix** with entries $m_{ij} = \int_\Omega \varphi_i\varphi_j d\mathbf{x}$ and $\mathbf{b} = \{b_i\}$ is the right hand side vector with entries defined as: $b_i = \int_\Omega f\varphi_i d\mathbf{x}$. Vector $\mathbf{u}$ denotes the vector of coefficients u_i. Solving the system above gives the optimal, in the sense of L2 norm, approximation of the analytical function $f \in V$ in the discrete function space V^h.

Let us now apply this method to two test problems. Firstly, let us analyse a simple infinitely smooth example. The analytical function is given by:

$$f(\mathbf{x}) = \sin(\pi x_1)\sin(\pi x_2) \tag{4.8}$$

The objective is to represent it with a linear combination of B-spline basis functions on the unit square $\Omega = [0,1]\times[0,1]$. Fig. 4.1 presents the plot of the function. Fig. 4.2 presents the discretization errors in the L2 norm, versus the number of basis functions in one direction. The basis functions used for this problem were B-splines of orders $p = 1$ to $p = 4$. For all orders except $p = 2$ the orders of convergence were at least equal to $p+1$. Therefore we can state that the higher the order of the basis the better it can reproduce the function (4.8). The behaviour of the convergence in case of the basis of order $p = 2$ is an interesting topic for further investigation, being however not in the scope of this thesis. The next problem to be analysed is the projection of a discontinuity to the space of continuous B-spline basis functions. The test problem is defined as the unit interval $\Omega = [0,1]$ with discontinuity at $x = 0.5$, i.e.:

$$f(x) = \begin{cases} 0 & \text{if } x < 0.5 \\ 1 & \text{if } x \geq 0.5 \end{cases} \tag{4.9}$$

to be represented with a linear combination of B-spline basis functions. The results are presented in Fig. 4.3. The left part of the figure demonstrates the behaviour of the projection

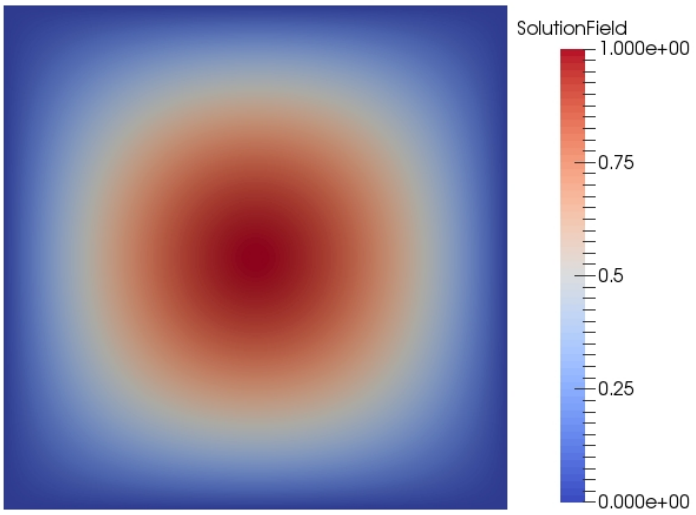


Figure 4.1: Plot of the function (4.8) on the unit square.

with increasing dimension of the basis while the influence of increasing the order of the basis functions is presented in the right part. The number of DOFs for different orders differs slightly in Fig. 4.3b because of the nature of the p-refinement but those additional single DOFs are not affecting the overall conclusions. For all cases considered, under- and over-shoots occur in the vicinity of the discontinuity. With increasing number of basis functions they become narrower but their magnitude is not significantly changing. Increasing the order is not changing significantly the quality of the result of projection neither.

In some classes of problems the existence of under- and over-shoots in the result of projection (for example the projected initial condition) can not be accepted. For example consider the following problem: a container with water that is divided into two equal parts by infinitely thin partition. In one of halves there is salt dissolved in maximum allowed concentration while the other one is completely free of salt. At some point the partition is removed and the diffusion of salt takes place. The initial condition for the diffusion process could be described with the problem discussed above but then over- and under-shoots produce non-physical values: concentration higher than maximal possible or lower than zero. Therefore there is a need for another projection method that will not produce under- and over-shoots. As it was already mentioned, L2 projection is an optimal projection in the sense of the L2 norm, so other projection would be less accurate in this sense but that is the price to be paid for restriction to only physically meaningful values.

4.2 Constrained L2 projection

The constrained L2 projection that is discussed here and used in further parts of this thesis is based on the concept of the **flux-corrected transport (FCT)** and therefore it will be referred to as **FCT-type constrained L2 projection**. The application of FCT to the constrained data projection is described in many publications [33, 35, 36, 37]. A detailed description of FCT is presented in Section 6.3. Here, only a very brief description of the

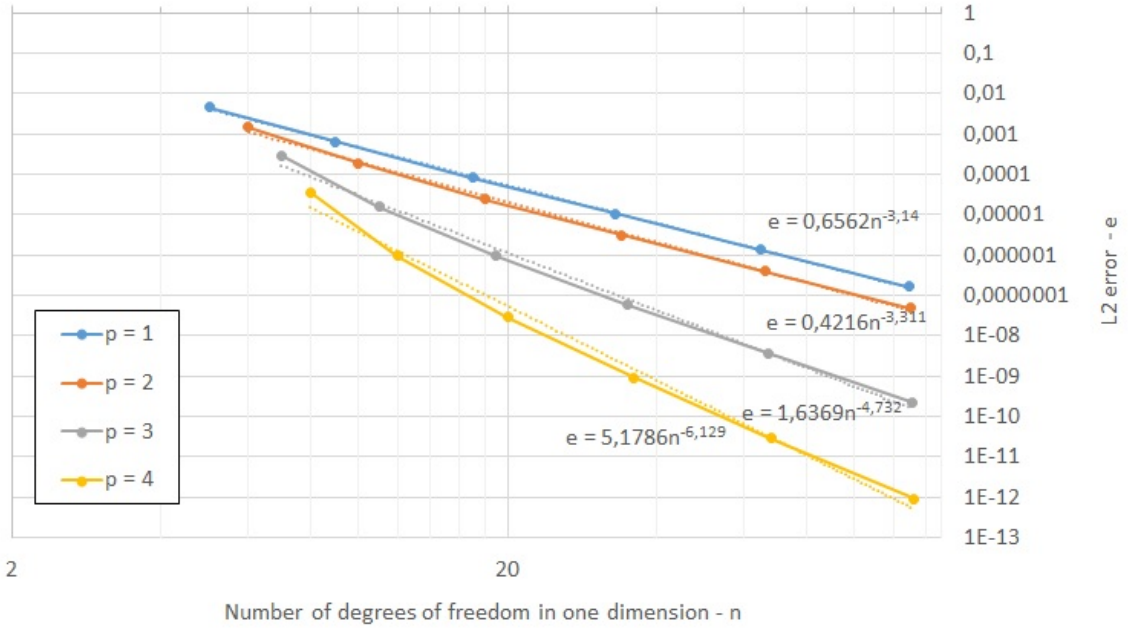


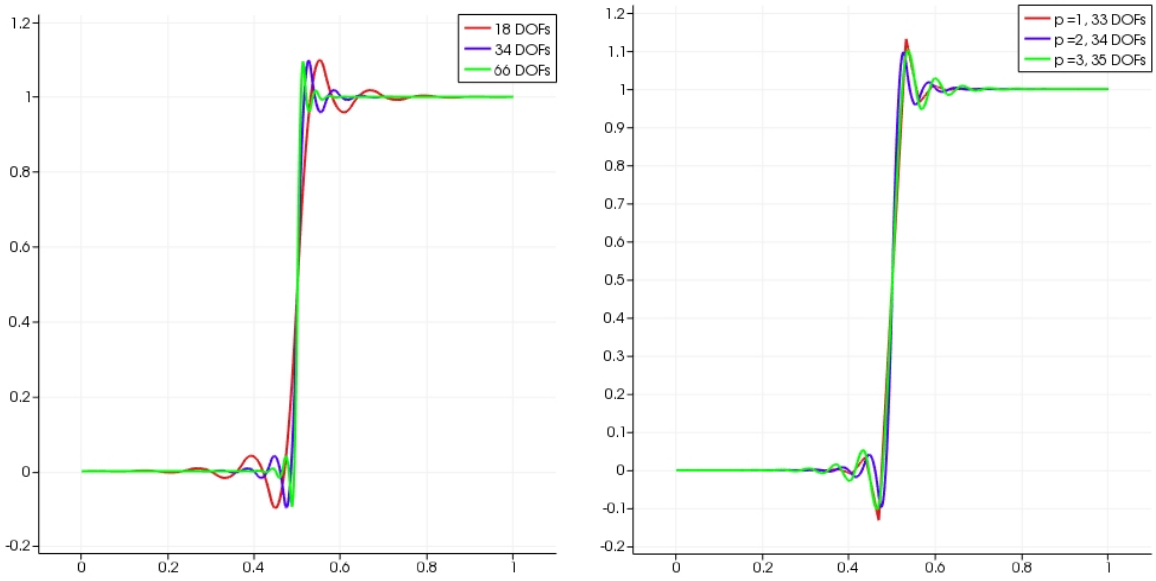
Figure 4.2: The L2 errors of the standard L2 projection applied to the analytical function (4.8) to the space spanned by B-spline bases of orders $p = 1$ to $p = 4$ versus number of DOFs.

general idea is presented as the original FCT algorithm was just an inspiration for creating the constrained L2 projection.

FCT was designed for solving hyperbolic equations with discontinuities occurring in the domain (for example the compressible Euler equations with a shock wave). The general idea is to choose a so-called **high-order** method which guarantees good quality of the solution in regions, where the solution profile is smooth but produces non-physical over- and under-shoots in non-smooth regions. The second ingredient is a **low-order** method which does not produce non-physical values in any part of the domain. Then the **anti-diffusion**, that applied to the low-order solution would give the high-order solution, has to be calculated. The next step is to limit the anti-diffusion such that the solution does not have non-physical values. The final step consist in applying the limited anti-diffusion to the solution obtained by the low-order method [33].

The standard L2 projection described in the previous section is a good choice for the high-order method as it preforms very well in smooth regions but produces undesirable under- and over-shoots in the vicinity of discontinuities. To find the proper low-order method let us look for the inspiration in the so-called **algebraic flux correction (AFC)** framework. Again, the detailed description will follow in Section 6.3.2. Here, let us only state that this is a framework in which the low-order method is obtained from the high-order one by algebraic operations on the matrices of the discrete form. The algebraic operations to be performed on matrices were determined by investigation of the entries that are responsible for the unwanted behaviour of the solution. In case of the consistent mass matrix M it was shown that those are the positive off-diagonal entries [33].

Therefore, in the low-order method we replace the consistent mass matrix M_C by so-called



(a) $p = 2$ and varying number of DOFs (b) varying p and equivalent number of DOFs

Figure 4.3: Results of the standard L2 projection applied to problem (4.9) for different orders and dimension of the B-spline basis.

row-sum lumped mass matrix M_L [33]:

$$M_L = \text{diag}\{m_i\}, \qquad m_i = \sum_j m_{ij} \tag{4.10}$$

Then the low-order problem reads:

$$M_L \mathbf{u} = \mathbf{b} \tag{4.11}$$

The components of the solution $\mathbf{u}^L$ from the scheme above (low-order method) are denoted with u_i^L while the ones of the solution $\mathbf{u}^H$ from the original scheme (high-order method) are denoted with u_i^H. It can be shown that the components of the anti-diffusive contribution (defined as the difference between high- and low-order schemes) are then given by [33]:

$$f_i(\mathbf{u}^H) = (M_L \mathbf{u}^H - M_C \mathbf{u}^H)_i \tag{4.12}$$

The conservative flux decomposition for the discrete diffusion operator $(M_L - M_C)$ yields (the conservative flux decomposition (5.40)-(5.41) is described in detail in Section 5.3.2) [33]:

$$(M_L \mathbf{u}^H - M_C \mathbf{u}^H)_i = \sum_{j \neq i} m_{ij}(u_i^H - u_j^H) \tag{4.13}$$

Therefore, the anti-diffusive contribution can be represented as a sum of raw anti-diffusive fluxes [33]:

$$f_i = \sum_{j \neq i} f_{ij}, \qquad f_{ij} = m_{ij}(u_i^H - u_j^H) \tag{4.14}$$

What is left is to apply the flux limiting technique that assures that there are no non-physical values in the projection and that the result is not overly smeared. For this we use

Zalesak's multidimensional FCT algorithm [38] generalized for non-nodal DOFs in the slightly modified way [33]:

1. Evaluate the sums of positive and negative anti-diffusive fluxes into i-th DOF:

$$P_i^+ = \sum_{j \neq i} \max\{0, f_{ij}\}, \qquad P_i^- = \sum_{j \neq i} \min\{0, f_{ij}\} \tag{4.15}$$

2. Compute the distance to a local maximum and minimum at the bounds:

$$Q_i^+ = m_i(u_i^{\max} - u_i^L), \qquad Q_i^- = m_i(u_i^{\min} - u_i^L) \tag{4.16}$$

   where $u_i^{\max}$ and $u_i^{\min}$ are, respectively, the maximum and minimum coefficient values of the DOFs having supports that overlap with the support of i-th DOF.

   Note: In the nodal version of this algorithm the limiting values are computed using the bounds of solution values. This is, however, not the case for non-nodal basis functions. In this case we consider the bounds of the coefficient values that do not directly correspond to the solution values.

3. Determine the nodal correction factors for the net increment at the i-th DOF:

$$R_i^+ = \min\left\{1, \frac{Q_i^+}{P_i^+}\right\}, \qquad R_i^- = \min\left\{1, \frac{Q_i^-}{P_i^-}\right\} \tag{4.17}$$

4. Control the sign of the raw anti-diffusive flux:

$$\alpha_{ij} = \begin{cases} \min\{R_i^+, R_j^-\}, & \text{if } f_{ij} > 0 \\ \min\{R_i^-, R_j^+\}, & \text{if } f_{ij} < 0 \end{cases} \tag{4.18}$$

Then the limited fluxes are given by:

$$\bar{f}_{ij} = \alpha_{ij} f_{ij} \tag{4.19}$$

The final result of the constrained L2 projection is obtained after adding the sum of the limited fluxes to the low-order solution [33]:

$$u_i = u_i^L + \frac{1}{m_i} \sum_{j \neq i} \bar{f}_{ij} \tag{4.20}$$

4.3 Numerical results

This section presents a comparison of numerical results for the standard L2 projection and its constrained counterpart applied to several test problems. First of all, let us compare the results of applying standard L2 projection (high-order problem), lumped mass matrix approximation (low-order problem) and constrained L2 projection to the discontinuous test problem (4.9). The result of projecting $f(x)$ to the space spanned by $p = 2$, $n = 34$ B-spline basis are presented in Fig. 4.4. It is easy to see that the standard L2 projection gives the steepest approximation of the discontinuity which is an important advantage but it also creates significant under- and over-shoots. The result of the low-order method (lumped mass

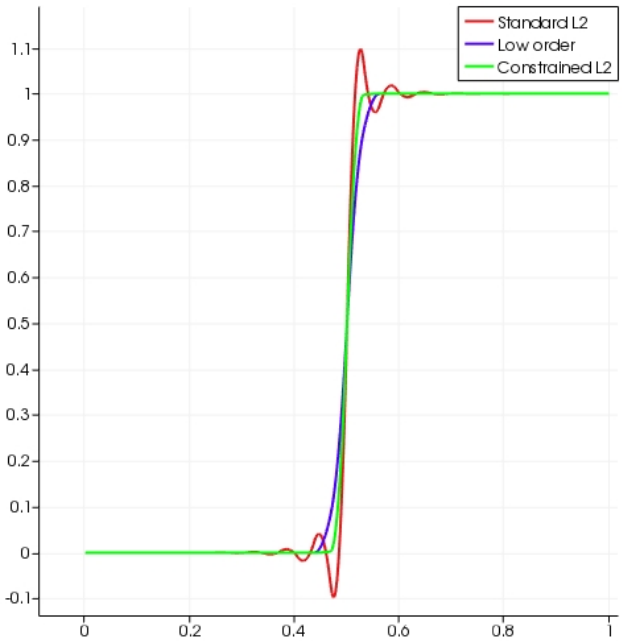
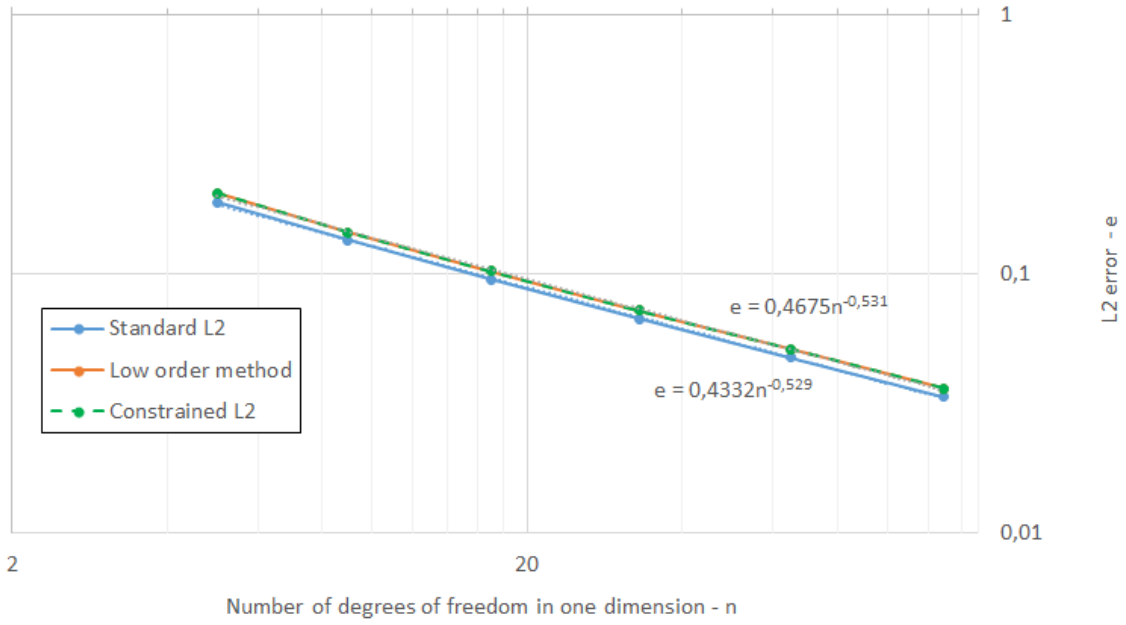


Figure 4.4: Results of the standard L2 projection, low-order method (lumped mass approximation) and constrained L2 projection applied to problem (4.9) on $p = 2$, $n = 34$ B-spline basis.

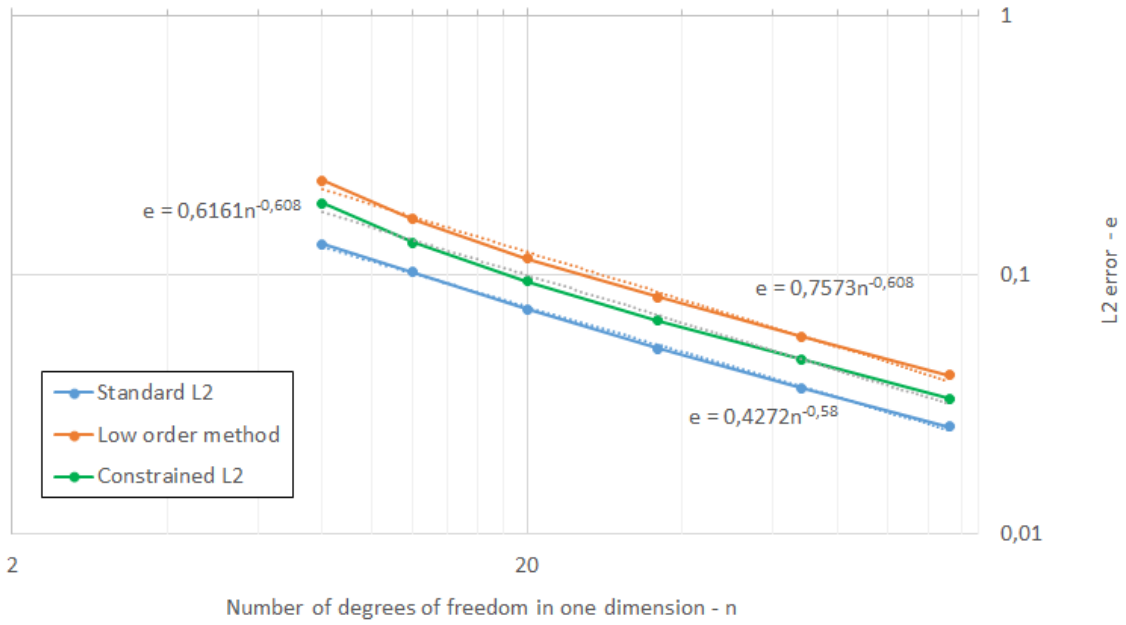
matrix approximation) is free of non-physical under- and over-shoots but the discontinuity is significantly smeared. The constrained L2 projection produces a steeper approximation of the discontinuity than the low-order method and does not produce any non-physical values.

Fig. 4.5 presents the L2 errors of the different approximations obtained by using standard L2 projection, low-order method (lumped mass approximation) and constrained L2 projection versus the dimension of the B-spline basis. Fig. 4.5a presents the errors for the $p = 1$ B-spline basis while Fig. 4.5b illustrates the behaviour for the $p = 4$ B-spline basis. Our first observation is that the orders of convergence are much lower than in the case of the smooth function (4.8) (see Fig. 4.2 for comparison). This is a severe limitation of the robustness of the standard L2 projection and constrained L2 projection. Another conclusion is that the orders of convergence do not improve significantly with increasing the order of the basis. In Fig. 4.5a the errors for the low-order method and the constrained L2 projection are indistinguishable. This is caused by the fact that those methods coincide for this problem in the case when the discontinuity is located at the point that is included in the support of exactly one basis function (for $p = 1$ this is a point where exactly one basis function attains the value 1) and the B-spline basis is of order $p = 1$ (for higher orders and uniform knot vectors there are no situations when an interior point is included in the support of exactly one basis function). Although all methods have orders of convergence limited by 0.5 because of the discontinuity the standard L2 projection is still giving the best results among them in terms of the L2 norm. This proves numerically the fact that standard L2 projection is the best possible projection in the sense of the L2 norm. For $p = 4$ the constrained L2 projection gives better results than the low-order method thanks to its ability to diminish the smearing of the discontinuity.

Let us proceed to more complicated 2D test case. The combination of smooth and non-



(a) $p = 1$



(b) $p = 4$

Figure 4.5: L2 errors of approximations obtained by the standard L2 projection, low-order method (lumped mass approximation) and constrained L2 projection applied to problem (4.9) on a $p = 1$ and $p = 4$ B-spline basis.

smooth regions in one domain is a very interesting case to study. Consider the initial condition of Leveque's solid body rotation benchmark [12] which is frequently used to asses the performance of numerical methods for time-dependent convection-diffusion equations. The function to be projected to the space spanned by B-spline basis functions is defined with the aid of three separate functions, each of them representing one body. The shapes can be expressed easily using the normalized distance function:

$$r(\mathbf{x}) = \frac{1}{r_0}\sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{4.21}$$

with (x_0, y_0) being the centre of the circle of radius $r_0 = 0.15$ in which each body is located. The first body is a slotted cylinder centred at $(x_0, y_0) = (0.5, 0.75)$ defined in $r(\mathbf{x}) < 1$ ($f_1(\mathbf{x}) = 0$ for $r(\mathbf{x}) \geq 1$) as:

$$f_1(\mathbf{x}) = \begin{cases} 1 & \text{if } |x - x_0| \geq 0.025 \vee y \geq 0.85 \\ 0 & \text{otherwise} \end{cases} \tag{4.22}$$

The second one is a cone centred at $(x_0, y_0) = (0.5, 0.25)$ defined in $r(\mathbf{x}) < 1$ ($f_2(\mathbf{x}) = 0$ for $r(\mathbf{x}) \geq 1$) as:

$$f_2(\mathbf{x}) = 1 - r(\mathbf{x}) \tag{4.23}$$

The last one is a hump centred at $(x_0, y_0) = (0.25, 0.5)$ defined in $r(\mathbf{x}) < 1$ ($f_3(\mathbf{x}) = 0$ for $r(\mathbf{x}) \geq 1$) as:

$$f_3(\mathbf{x}) = 0.25\,[1 + \cos(\pi \min\{r(\mathbf{x}), 1\})] \tag{4.24}$$

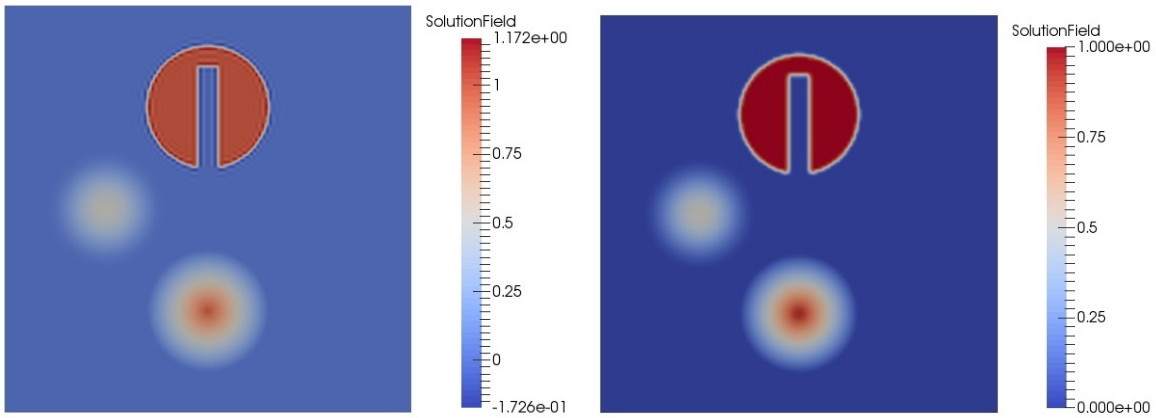
Then the function to be projected is defined as:

$$f(\mathbf{x}) = f_1(\mathbf{x}) + f_2(\mathbf{x}) + f_3(\mathbf{x}) \tag{4.25}$$

The results of the standard L2 projection and the constrained L2 projection of the function defined above to the space spanned by $p = 2$ 130×130 uniform B-spline basis functions are depicted in Fig. 4.6. As expected, the standard L2 projection produces under- and over-shoots while the result obtained from the constrained L2 projection is strictly bounded between 0 and 1.

Fig. 4.7 presents the plots of L2 errors versus the number of basis functions. Fig. 4.7a presents the errors of the standard L2 projection, the low-order method (lumped mass approximation) and the constrained L2 projection for the $p = 2$ basis. It can be easily seen that in this case the orders of convergence are also limited by 0.5 due to the presence of the discontinuity in the domain. Again, the standard L2 projection performs slightly better than the constrained L2 projection. The low-order method produces results of the lowest quality. Fig. 4.7b presents the behaviour of the error for different orders of the basis for the constrained L2 projection. Although the differences are not significant the general trend is that for this problem better approximations are obtained with the basis of lower order. This is caused by the fact that it is harder to represent the discontinuity with higher order functions due to larger support (size of support of B-spline basis functions is $p + 1$ knot spans), and therefore, more DOFs are influenced by the discontinuity.

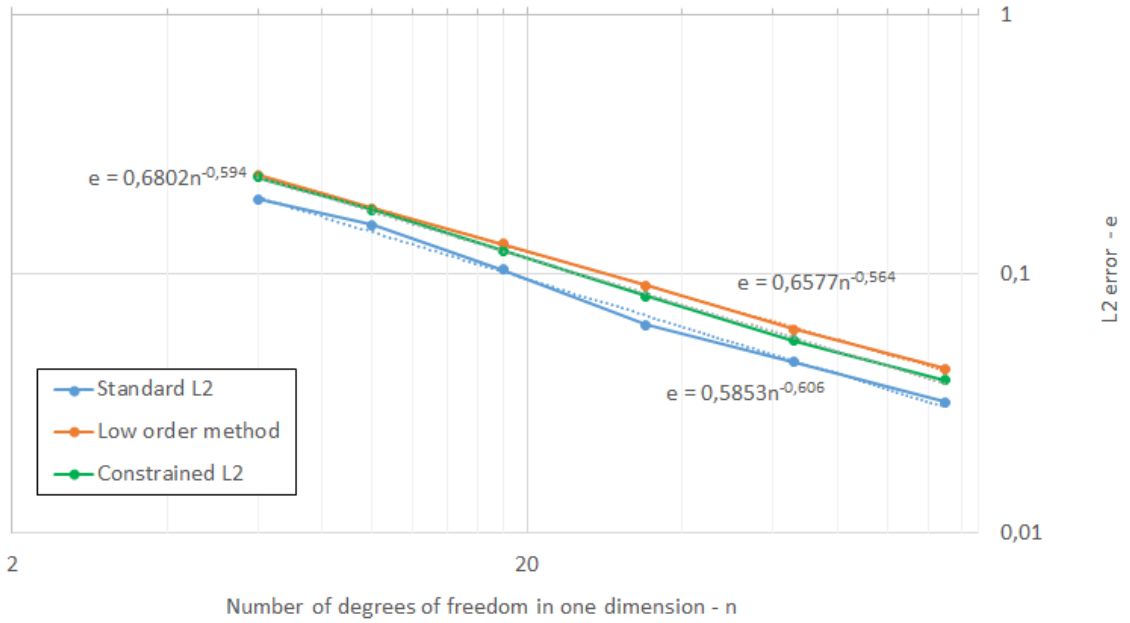
The general conclusion from this chapter is that although the standard L2 projection is provably the best approximation in the sense of minimizing the error in the L2 norm, it produces under- and over-shoots for functions involving discontinuities. For some classes of



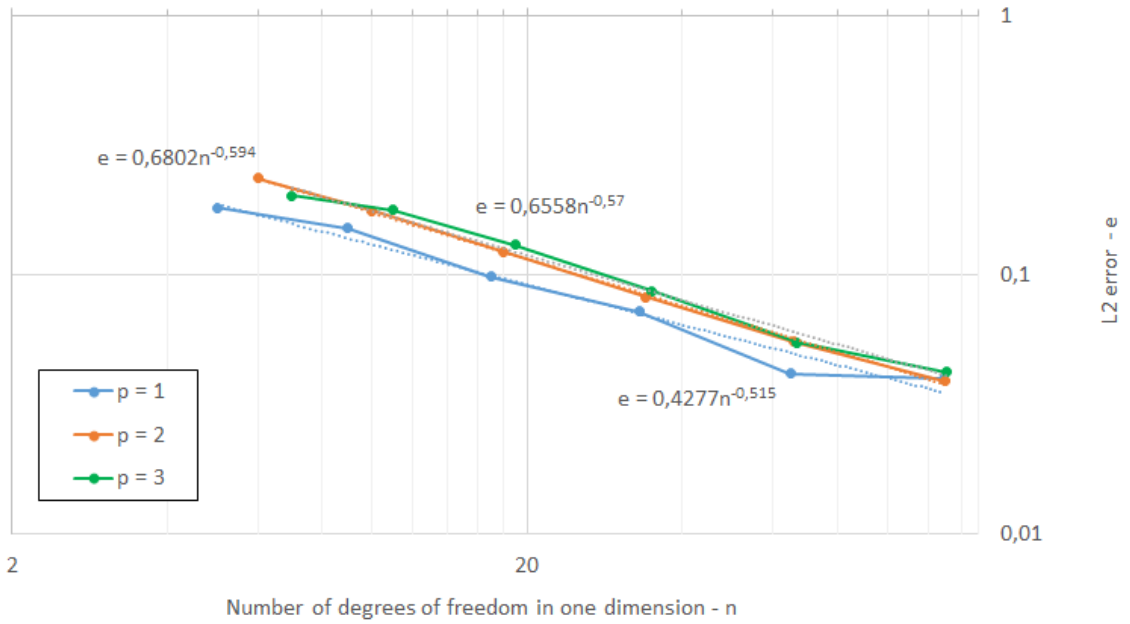
(a) standard L2 projection (b) constrained L2 projection

Figure 4.6: Results of the standard L2 projection and the constrained L2 projection of function (4.21) - (4.25) to the space spanned by $p = 2$ 130×130 uniform B-spline basis functions.

problems those under- and over-shoots are exceeding the possible physical bounds for values attained in the domain. Therefore, to preserve the physical meaning of the problem we have to use a projection method that gives worse approximations in the sense of the L2 norm but does not produce non-physical values such as negative concentrations or concentrations exceeding 100%. A very good method that fulfils this requirement is the FCT-type constrained L2 projection [33]. This method will be used in further chapters of this thesis to project the analytically described boundary conditions and initial values to the space spanned by B-spline basis functions.



(a) standard L2 projection, low-order method (lumped mass approximation) and constrained L2 projection on $p = 2$ basis



(b) constrained L2 projection on $p = 1$ to $p = 3$ basis

Figure 4.7: L2 errors of approximations obtained by the standard L2 projection, low-order method (lumped mass approximation) and constrained L2 projection applied to problem function (4.21) - (4.25) on a space spanned by B-spline basis.

Chapter 5

Stationary convection-diffusion equation

The natural starting point for the development and analysis of numerical methods for flow problems is to apply them to the stationary convection-diffusion equation. In this simple problem one can find a representation of some phenomena and numerical issues that occur also in more complicated systems of equations that govern compressible fluid flows. Therefore, we discuss in this chapter the application of isogeometric analysis to the stationary convection-diffusion equation and postpone the extension to the compressible Euler equations to Chapter 7.

First of all, the description of the problem and the discretization are presented. The second section contains numerical results and reveals the expected problems concerning stability for convection-dominated problems. Further, we give a description of the algebraic flux correction (AFC) methodology which is used for stabilization. The last part of this chapter is devoted to the analysis of results obtained by using IGA along with AFC and a comparison with results from IGA with SUPG stabilization.

5.1 Discretization of the stationary convection-diffusion equation

The **convection-diffusion equation** is a combination of the convection and diffusion equations. It governs the behaviour of particles, energy or other physical quantity transferred due to convection and diffusion processes. Convection is a process of macroscopic motion of a physical quantity regardless of its cause [39], whereas diffusion is a process of movement of a physical quantity from a region of high value (eg. concentration) to a region of low value. The most general form of the convection-diffusion equation reads [40]:

$$-\nabla \cdot (D\nabla u(\mathbf{x})) + \nabla \cdot (\mathbf{v}(\mathbf{x})u(\mathbf{x})) = R(\mathbf{x}) \quad \textbf{in } \Omega \tag{5.1}$$

where $u(\mathbf{x})$ is the variable of interest (concentration of particles, energy, other physical quantity), D is the diffusion tensor (in simpler cases it can be replaced by scalar diffusion coefficient d), $\mathbf{v}(\mathbf{x})$ is an average velocity of quantity (for clarity of presentation the dependency on $\mathbf{x}$ will not be included in the equation in further parts of this chapter) and $R(\mathbf{x})$ is a source term.

The problem is completed with properly chosen boundary conditions (to ensure well-posedness):

$$u(\mathbf{x}) = \gamma(\mathbf{x}) \ \text{ on } \Gamma_D \tag{5.2}$$

$$\frac{du}{d\mathbf{n}}(\mathbf{x}) = \beta(\mathbf{x}) \ \text{ on } \Gamma_N \tag{5.3}$$

where $\mathbf{n}$ is the unit outward normal vector. The weak form of the problem (obtained after multiplication by a test function, integration over the domain and integration by parts applied to the diffusive term only) reads:

$$\text{find } u \in S \text{ such that for all } v \in V, \ \ a(u,v) = f(v). \tag{5.4}$$

where

$$a(u,v) = \int_\Omega ((D\nabla u)\cdot\nabla v + (\nabla\cdot(\mathbf{v}u))v)\, d\mathbf{x} - \int_\Gamma D\frac{du}{d\mathbf{n}} v ds \tag{5.5}$$

$$f(v) = \int_\Omega Rv d\mathbf{x} \tag{5.6}$$

The discretization of the weak form is based on the standard Galerkin method, the discrete variational formulation reads:

$$\text{find } u^h \in S^h \text{ such that for all } v^h \in V^h, \ \ a(u^h, v^h) = f(v^h). \tag{5.7}$$

where

$$a(u^h, v^h) = \int_\Omega ((D\nabla u^h)\cdot\nabla v^h + (\nabla\cdot(\mathbf{v}u)^h)v^h) d\mathbf{x} \tag{5.8}$$

$$f(v^h) = \int_\Omega Rv^h d\mathbf{x} + \int_{\Gamma_N} D\beta v^h ds \tag{5.9}$$

Test functions v^h (because of assumed restrictions on the test space) vanish on the Γ_D and, therefore, we need to only consider the integration over Γ_N instead of the whole Γ [33]. Therefore, we can replace the term $\frac{du}{d\mathbf{n}}$ by the prescribed Neumann boundary condition β. After this operation the term resulting from the treatment of boundary conditions is no longer depending on u^h and therefore belongs now to the linear form $f(v^h)$. In the framework of IGA the considered discrete spaces spanned by B-spline (NURBS) basis. Using Fletcher's group formulation [41], the approximate solution u^h and the convective flux $(\mathbf{v}u)^h$ can be represented in terms of the basis $\{\varphi_j(\mathbf{x})\}$ of V^h as follows [42]:

$$u^h(\mathbf{x}) = \sum_j u_j \varphi_j(\mathbf{x}) \tag{5.10}$$

$$(\mathbf{v}u)^h(\mathbf{x}) = \sum_j \mathbf{v}_j u_j \varphi_j(\mathbf{x}) \tag{5.11}$$

Then, knowing that it is sufficient to test the variational formulation only with the basis functions of V^h one after the other, the discrete problem can be rewritten in the form:

$$\sum_j \left(\int_\Omega ((D\nabla\varphi_j)\cdot\nabla\varphi_i) d\mathbf{x} + \int_\Omega (\nabla\cdot(\mathbf{v}_j\varphi_j))\varphi_i d\mathbf{x} \right) u_j = \int_\Omega R\varphi_i d\mathbf{x} + \int_{\Gamma_N} D\beta\varphi_i ds \tag{5.12}$$

This equation can be rewritten in matrix form (the K operator is multiplied by -1 to preserve consistency of notation with time-dependent problems presented in the next chapters as well as with the common notation used in literature):

$$(S - K)\mathbf{u} = \mathbf{r} \tag{5.13}$$

Here, $S = \{s_{ij}\}$ is the **discrete diffusion operator** with entries:

$$s_{ij} = \int_{\Omega} (D\nabla\varphi_j \cdot \nabla\varphi_i) d\mathbf{x} \tag{5.14}$$

$K = \{k_{ij}\}$ is the **discrete convection operator** with entries:

$$k_{ij} = -\mathbf{v}_j \cdot \mathbf{c}_{ij} \tag{5.15}$$

$$\mathbf{c}_{ij} = \int_{\Omega} \nabla\varphi_j \varphi_i d\mathbf{x} \tag{5.16}$$

$\mathbf{u}$ is a vector of solution coefficients u_i and $\mathbf{r}$ is a right-hand side vector, resulting from the source term and treatment of boundary conditions, with entries r_i:

$$r_i = \int_{\Omega} R\varphi_i d\mathbf{x} + \int_{\Gamma_N} D\beta\varphi_i ds \tag{5.17}$$

The treatment of the mapping from the parametric domain to the physical domain is analogous to the one described in Section 3.3.2.

5.2 Numerical results

The IGA-based solver of convection-diffusion equations was applied to two benchmark problems. The first problem deals with a polynomial analytic solution, whereas the second one is characterized by boundary and internal layers. The numerical results are presented below.

5.2.1 Smooth polynomial problem

The first test case to validate the IGA convection-diffusion solver is a smooth polynomial problem given by:

$$-\nabla \cdot (d\nabla u(\mathbf{x})) + \nabla \cdot (\mathbf{v}u(\mathbf{x})) = R(\mathbf{x}) \ \text{ in}\, \Omega = [0,1] \times [0,1] \tag{5.18}$$

$$u(\mathbf{x}) = u_{ex}(\mathbf{x}) \ \text{ on}\, \Gamma \tag{5.19}$$

with:

$$d = 1 \tag{5.20}$$

$$\mathbf{v} = \begin{bmatrix} \frac{1}{5} & \frac{2}{5} \end{bmatrix}^T \tag{5.21}$$

$$R(\mathbf{x}) = 5x^3 - \frac{288}{5}x^2 - \frac{114}{5}x - \frac{26}{5} \tag{5.22}$$

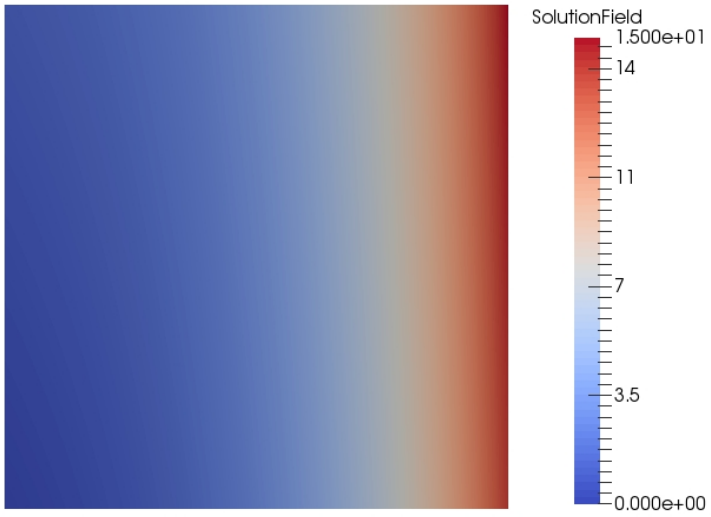


Figure 5.1: The exact solution of the problem (5.18)-(5.23).

$$u_{ex}(\mathbf{x}) = 5x^4 + 4x^3 + 3x^2 + 2x + y \tag{5.23}$$

$u_{ex}(\mathbf{x})$ is the exact solution to this problem which is also used to prescribe the Dirichlet boundary conditions. It is depicted in Fig. 5.1. The computations were made on a space spanned by uniform B-spline basis functions of orders $p = 1$ to $p = 4$. Fig. 5.2 presents the errors in L2 norm (see 3.12) of te obtained numerical solutions versus the number of degrees of freedom in one direction (the numbers of degrees of freedom in both directions are equal). It can be easily seen that the $p+1$ order of convergence was obtained for orders up to $p = 3$. For $p = 4$ the maximum possible precision was obtained as the exact solution is a polynomial of order 4 and it can be exactly represented with the basis functions of order $p = 4$, so the convergence was immediate and the error increases with increasing number of degrees of freedom because of increasing rounding errors. Based on this problem we can conclude that the method performs well for smooth problems.

5.2.2 Problem with internal and boundary layer

The next problem is a common benchmark for stationary convection-diffusion equations in 2D that can be found for example in [1]. It involves internal (skew to the mesh) and boundary (right and partially top boundaries of the domain) layers and is very prone to instabilities [1]. The benchmark problem is defined as follows:

$$-\nabla \cdot d\nabla u(\mathbf{x})) + \nabla \cdot (\mathbf{v}u(\mathbf{x})) = 0 \ \text{ in}\, \Omega = [0,1] \times [0,1] \tag{5.24}$$

$$u(\mathbf{x}) = \beta(\mathbf{x}) \ \text{ on}\, \Gamma \tag{5.25}$$

with:

$$\mathbf{v} = \begin{bmatrix} \cos(\theta) & \sin(\theta) \end{bmatrix}^T, \quad \theta = \tan^{-1}(2) \tag{5.26}$$

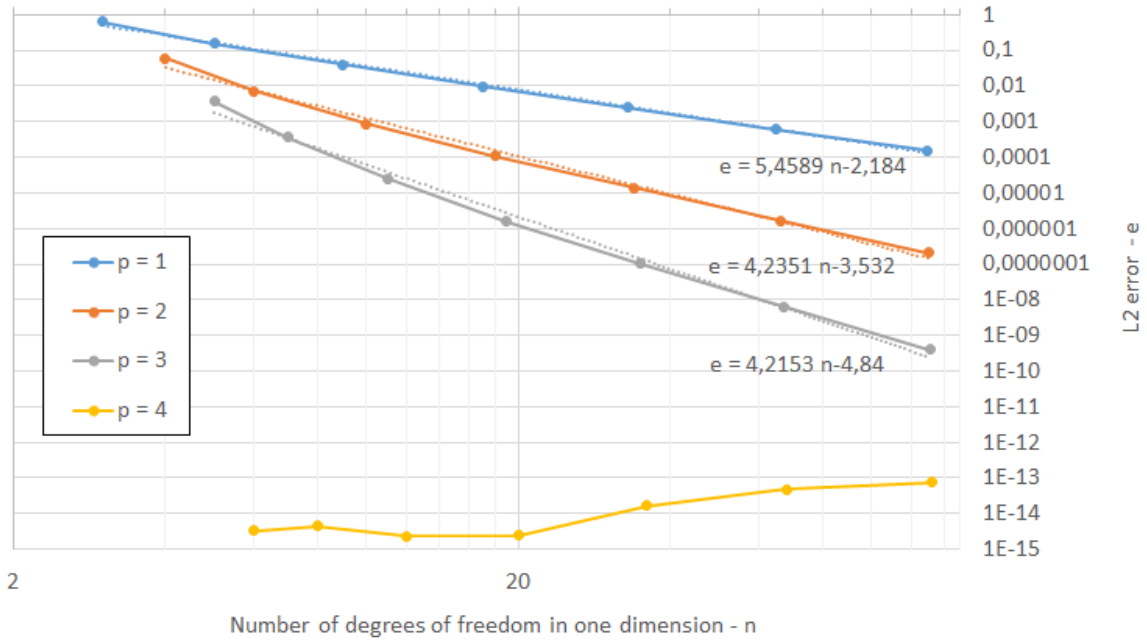


Figure 5.2: The L2 errors of numerical solutions obtained with uniform B-spline basis functions of orders $p = 1$ to $p = 4$ of the problem (5.18)-(5.23) versus the number of DOFs.

$$\beta(\mathbf{x}) = \begin{cases} 1 & \text{if } y \leq \frac{1}{5} - \frac{1}{5}x \\ 0 & \text{else} \end{cases} \tag{5.27}$$

The general setting of the problem is depicted in Fig. 5.3. The **element Péclet number** Pe_h is defined as [1]:

$$Pe_h = \frac{|\mathbf{v}|h}{2D} \tag{5.28}$$

where h denotes the length of knot span. In case of the same number of uniform basis functions in both directions $h = \frac{1}{n}$, with n being the number of basis functions in one direction. The element Péclet number is a non-dimensional measure for the ratio between convective and diffusive phenomena on the mesh scale. For $Pe_h > 1$ the problem is called **convection-dominated**, whereas for $Pe_h < 1$ it is called **diffusion-dominated** [1].

The benchmark problem was solved using the IGA stationary convection-diffusion solver for $d = 0.1$, $d = 0.01$ and $d = 0.001$ which correspond, respectively, to $Pe_h = 0.56$, $Pe_h = 5.56$ and $Pe_h = 55.56$. The used space was spanned by uniform $p = 2$ 18×18 B-spline basis. The Dirichlet boundary values were approximated with constrained L2 projection (see Chapter 4) to avoid over- and under-shoots in the vicinity of the discontinuity. The results are presented in Fig. 5.4.

The unstabilized Galerkin method works well for the diffusion-dominated case, as demonstrated in Fig. 5.4a. After decreasing the diffusion coefficient d by factor 10 the solution becomes oscillatory, the internal and boundary layer become more steep and they cause oscillations that are propagated over the whole domain (see Fig. 5.4b). It is important to mention that those oscillations are non-physical, i.e. the values of the variable of interest (e.g.

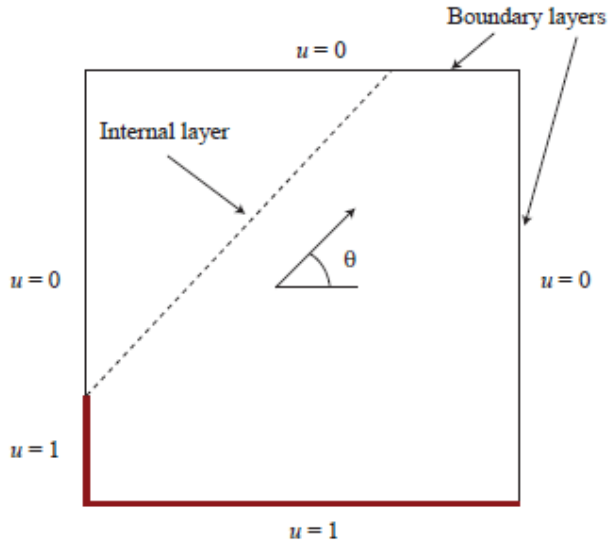


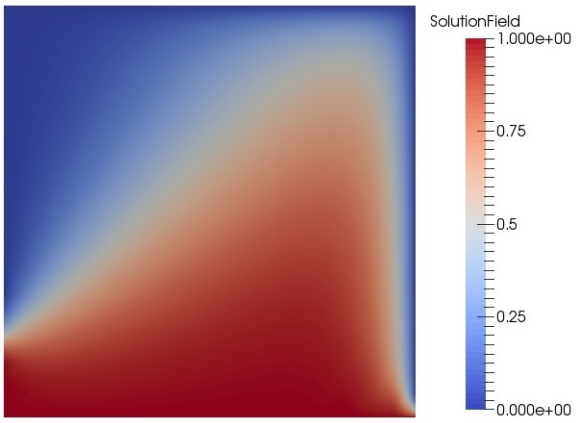
Figure 5.3: Benchmark (5.24) - (5.27) for stationary convection-diffusion equation. Source: [1].

concentration) should be bounded between 0 and 1 in the given problem. The result is even worse if we decrease d by factor of 10 once again, as shown in Fig. 5.4c. We can conclude that the standard Galerkin method is not suitable for convection-dominated problems that feature internal and/or boundary layers. Therefore, there is a necessity for applying some stabilization method.

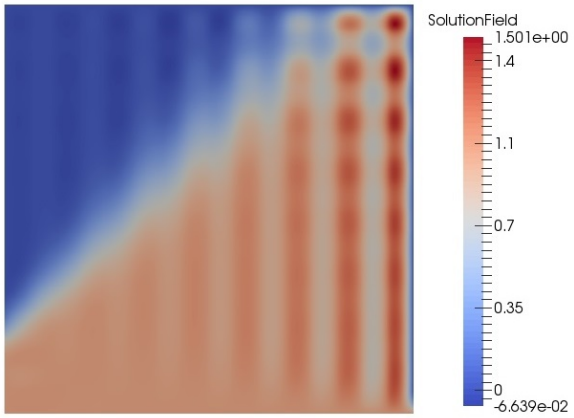
The most common stabilization method for standard FEM is **Streamline Upwind/Petrov-Galerkin method (SUPG)**. It was shown in [1] that this stabilization can be used within the IGA framework. This method has several drawbacks, one of them being that it does not guarantee the prevention of non-physical under- and over-shoots. Therefore, we adopt another stabilization method - **Algebraic Flux Correction method**, which assures that the solution does not involve non-physical values.

5.3 Algebraic flux correction

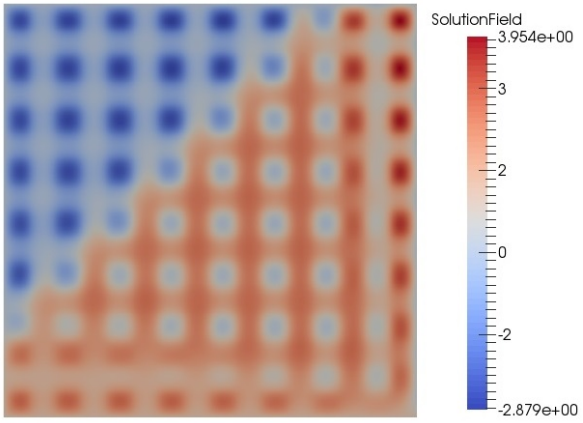
For the stationary convection-diffusion problem we will use algebraic flux correction with **TVD-type limiting**. It was introduced by D. Kuzmin in [43] and D. Kuzmin, S. Turek in [11]. The full description of the method can be found in those papers as well as in [14, 44]. This section presents a brief description of the method applied to the stationary convection-diffusion problem. Although the considered problem is not time-dependent and the stabilization method was proved to work for the stationary problems [11], many definitions used to develop the limiting method have their origin in time-dependent problems. Therefore, the theory will be presented for time-dependent problems and at some point restricted to the considered stationary case.



(a) $d = 0.1$, $Pe_h = 0.56$



(b) $d = 0.01$, $Pe_h = 5.56$



(c) $d = 0.001$, $Pe_h = 55.56$

Figure 5.4: Numerical results for benchmark problem (5.24) - (5.27) for different values of d solved on $p = 2$, 18×18 B-spline basis.

5.3.1 TVD and LED schemes

The detailed description of the **total variation diminishing (TVD) methodology** can be found in [45]. To show the general idea let us analyse the scalar conservation law in one dimension:

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{5.29}$$

with f being a flux function. Lax in [46] showed that any physically admissible solution to this problem is characterized by its **total variation** defined as:

$$TV(u) = \int_{-\infty}^{\infty} \left| \frac{\partial u}{\partial x} \right| dx \tag{5.30}$$

The total variation of the exact solution is always non-increasing [11] therefore it is obvious that the total variation of the numerical solution should be non-increasing as well, namely:

$$TV(u^{n+1}) \leq TV(u^n) \tag{5.31}$$

with u^n denoting the solution at time t^n. Semi-discrete difference schemes that fulfil this requirement are called **total variation diminishing (TVD)** schemes [11]. Harten proved in [47] that a conservative semi-discrete difference scheme of the form:

$$\frac{du_i}{dt} + \frac{f_{i+1/2} - f_{i-1/2}}{\Delta x} = 0 \tag{5.32}$$

is TVD if it can be rewritten in the form:

$$\frac{du_i}{dt} = c_{i-1/2}(u_{i-1} - u_i) + c_{i+1/2}(u_{i+1} - u_i) \tag{5.33}$$

with (possibly non-linear) non-negative coefficients $c_{i-1/2}$ and $c_{i+1/2}$ [11]. Additional restrictions are necessary for the time discretization to avoid instabilities but they are out of the scope of this section. There are several linear methods that satisfy the condition above and guarantee that no oscillations are created in the solution. Unfortunately, as proved by Godunov [48], linear TVD schemes are at most first-order accurate. To overcome this limitation the fluxes $f_{i\pm1/2}$ have to be calculated in non-linear fashion by combining linear high- and low-order approximations, i.e. [11]:

$$f_{i\pm1/2} = f^L_{i\pm1/2} + \Phi_{i\pm1/2}(f^H_{i\pm1/2} - f^L_{i\pm1/2}) \tag{5.34}$$

where $\Phi_{i\pm1/2}$ is so-called adaptive **flux limiter** designed such that the final fluxes satisfy Harten's TVD condition (5.33). There are several methods that are based on this approach and on different ways to define the limiter. The important limitation is the fact that Harten's TVD condition is defined only in one dimension [11]. Therefore we need a generalization of this criterion.

A viable generalization are **local extremum diminishing (LED)** which are required to fulfil the following conditions [11]:

- local maxima are non-increasing: u_i is a local maximum $\Rightarrow$ $\frac{du_i}{dt} \leq 0$
- local minima are non-decreasing: u_i is a local minimum $\Rightarrow$ $\frac{du_i}{dt} \geq 0$

In [11], Kuzmin shows that the one-dimensional LED schemes are necessarily TVD. Thus, LED turns out to be a handy generalization of TVD which is not restricted to one dimension. It is also important to note here that the Godunov theorem limiting the accuracy of linear TVD schemes to first-order also holds for the LED schemes. Namely, linear LED schemes are at most first-order accurate [11, 48].

5.3.2 Algebraic flux correction with TVD-type limiting

The semi-discretized transport equation can be rewritten in the generic form [11]:

$$\frac{du_i}{dt} = \sum_j \sigma_{ij} u_j \tag{5.35}$$

with $\sigma_{ii} = -\sum_{j\neq i} \sigma_{ij}$. As long as the scheme is chosen such that the row sums of the discrete operators are zero we can rewrite the equation above in terms of off-diagonal entries [11], i.e.:

$$\frac{du_i}{dt} = \sum_{j\neq i} \sigma_{ij}(u_j - u_i) \tag{5.36}$$

Jameson showed in [49] that the negative coefficients σ_{ij} in the equation above are responsible for the growth of instabilities. Therefore, the following condition can be stated for the schemes to be LED [11]:

$$\forall j \neq i, \;\; \sigma_{ij} \geq 0 \tag{5.37}$$

Now, we restrict ourselves to the problem to which we have to apply the stabilization - the stationary convection-diffusion equation. Although the definitions of LED and TVD schemes are based on the behaviour of numerical solution in time we can intuitively get the feeling that if the system is not LED it can give non-physical oscillations even for stationary problem. We can consider stationary solutions as the stationary limit to the corresponding pseudo-time-dependent problem. Indeed, all the operations described below are necessary to get a solution free of non-physical oscillations although they correspond to just assuring the scheme to be LED. The stationary convection-diffusion equation discretized with the standard Galerkin method has the following form:

$$(S - K)\mathbf{u} = \mathbf{r} \tag{5.38}$$

This scheme is LED if we assure that condition (5.37) is satisfied for $(S - K)$ operator multiplied by $-I$ (see (5.36) with $\frac{du_i}{dt} = 0$ for stationary problem). Therefore, we need to assure that there are no negative off-diagonal entries in the operator $(K - S)$. It was shown in [42] that the discrete diffusion operator S is not causing problems as all off-diagonal entries are non-positive unless the mesh or the diffusion tensor are highly anisotropic (we restrict ourselves in this thesis to isotropic diffusion tensors). To assure no negative off-diagonal entries in the discrete convective operator K so-called **discrete upwinding** technique [10] is applied. The symmetric **artificial diffusion operator** $D = \{d_{ij}\}$ having zero row and column sums (to preserve conservativity of the scheme), i.e. [11]:

$$d_{ij} = d_{ji} \quad \sum_j d_{ij} = \sum_i d_{ij} = 0 \tag{5.39}$$

is added to the discrete convection operator K. Square matrices that have all the properties (5.39) are called **discrete diffusion operators** [10]. Let us now define the operation of so-called **conservative flux decomposition** for an arbitrary discrete diffusion operator $\tilde{D} = \{\tilde{d}_{ij}\}$ as [11]:

$$\left(\tilde{D}\mathbf{u}\right)_i = \sum_{j \neq i} \tilde{f}_{ij} \tag{5.40}$$

where $\tilde{f}_{ij}$ are **diffusive fluxes** defined as:

$$\tilde{f}_{ij} = \tilde{d}_{ij}(u_j - u_i) \tag{5.41}$$

The correctness of this relation can be proven by:

$$\left(\tilde{D}\mathbf{u}\right)_i = \sum_j \tilde{d}_{ij} u_j \tag{5.42}$$

$$= \tilde{d}_{ii} u_i + \sum_{j \neq i} \tilde{d}_{ij} u_j \tag{5.43}$$

$$\overset{(5.39)}{=} -\sum_{j \neq i} \tilde{d}_{ij} u_i + \sum_{j \neq i} \tilde{d}_{ij} u_j \tag{5.44}$$

Therefore, all the operators that have the properties of a discrete diffusion operator can be decomposed in this way.

The optimal entries of the artificial diffusion operator D are given by [11, 10]:

$$d_{ij} = d_{ji} = \max\{0, -k_{ij}, -k_{ji}\} \tag{5.45}$$

$$d_{ii} = -\sum_{j \neq i} d_{ij} \tag{5.46}$$

Then the modified discrete convection operator $L = \{l_{ij}\}$ is defined as $L = K + D$. In practice the artificial diffusion operator D is not constructed but modification of entries of L is done within an edge-by-edge approach. First of all, the matrix is initialized by: $L := K$. Then each pair of its off-diagonal entries l_{ij} and l_{ji} is analysed. We calculate d_{ij} according to (5.45) and perform modifications of entries in the L matrix as follows [11]:

$$l_{ii} := l_{ii} - d_{ij} \tag{5.47}$$

$$l_{ij} := l_{ij} + d_{ij} \tag{5.48}$$

$$l_{ji} := l_{ji} + d_{ij} \tag{5.49}$$

$$l_{jj} := l_{jj} - d_{ij} \tag{5.50}$$

After those modifications we obtain the linear scheme:

$$(S - L)\mathbf{u} = \mathbf{r} \tag{5.51}$$

which is LED. Unfortunately, according to Godunov's theorem [48], this linear scheme is limited to first-order accuracy. Therefore, we need a workaround for this limitation following a similar approach as (5.34). We need to construct a non-linear LED scheme by a non-linear combination of linear high- and low-order schemes. The original standard Galerkin method (5.38) serves as the linear high-order scheme while the new LED scheme (5.51) obtained from adding artificial diffusion is the low-order scheme. The non-linear scheme that fulfils our expectations is [11]:

$$(S - K^*(\mathbf{u}))\mathbf{u} = \mathbf{r} \tag{5.52}$$

with:

$$K^*(\mathbf{u}) = L + \bar{F}(\mathbf{u}) = K + D + \bar{F}(\mathbf{u}) \tag{5.53}$$

That is, the low-order scheme is augmented by some non-linear anti-diffusion $\bar{F}(\mathbf{u}) = \{\bar{f}_{ij}(\mathbf{u})\}$ to avoid the loss of accuracy in smooth regions due to artificial diffusion. The anti-diffusive operator $\bar{F}(\mathbf{u})$ has the property of a discrete diffusion operator (more precisely discrete anti-diffusion operator, as it has opposite sign as D) [11]. Therefore, we can decompose the anti-diffusion as a sum of anti-diffusive fluxes:

$$(\bar{F}(\mathbf{u})\mathbf{u})_i = \sum_{j \neq i} \bar{f}_{ij}(\mathbf{u}) \tag{5.54}$$

with:

$$\bar{f}_{ij}(\mathbf{u}) = \beta_{ij}(\mathbf{u})(u_i - u_j) \tag{5.55}$$

We need to define the coefficients β_{ij} in such a way that the anti-diffusive fluxes are meaningful. It is worthwhile to use the analogy to (5.34) - define the anti-diffusive fluxes as a limited difference of high- and low-order fluxes. As the difference between the high- and low-order schemes is the artificial diffusion added, which applied to the solution can be easily decomposed into fluxes, we can define β_{ij} as:

$$\beta_{ij}(\mathbf{u}) = \alpha_{ij}(\mathbf{u})d_{ij} \tag{5.56}$$

where $\alpha_{ij}(\mathbf{u})$ is an **adaptive flux limiter**. Clearly, for $\alpha_{ij} = 1$ (no limiting adopted) the anti-diffusive fluxes will be equal to the difference of high- and low-order fluxes. The limiters that assure that the final scheme is LED are calculated using the TVD-type limiting algorithm proposed by Kuzmin in [43] generalized to non-nodal DOFs. In the description of the algorithm the dependency of (u) is omitted to simplify the presentation. Let us define the raw anti-diffusive fluxes to be limited as [11]:

$$f_{ij} = d_{ij}(u_i - u_j) \tag{5.57}$$

Then the **TVD-type limiting algorithm** by Kuzmin [43] generalised for non-nodal DOFs reads: for each pair of overlapping DOFs i and j such that $l_{ji} \geq l_{ij} \geq 0$:

1. Evaluate the sums of positive and negative anti-diffusive fluxes into i-th DOF:

$$P_i^+ = P_i^+ + \max\{0, f_{ij}\}, \qquad P_i^- = P_i^- + \min\{0, f_{ij}\} \tag{5.58}$$

2. Compute the upper and lower bounds $Q_i^\pm$ to be imposed on the sums $P_i^\pm$:

$$Q_i^+ = Q_i^+ + \max\{0, -f_{ij}\} \qquad Q_j^+ = Q_j^+ + \max\{0, -f_{ij}\} \tag{5.59}$$

$$Q_i^- = Q_i^- + \min\{0, -f_{ij}\} \qquad Q_j^- = Q_j^- + \min\{0, -f_{ij}\} \tag{5.60}$$

3. Determine the nodal correction factors at the upwind DOF i:

$$R_i^+ = \min\left\{1, \frac{Q_i^+}{P_i^+}\right\}, \qquad R_i^- = \min\left\{1, \frac{Q_i^-}{P_i^-}\right\} \tag{5.61}$$

4. Compute flux limiters:

$$\alpha_{ij} = \begin{cases} R_i^+, & \text{if } f_{ij} > 0 \\ R_i^-, & \text{otherwise} \end{cases} \tag{5.62}$$

The resulting scheme:

$$(S - L)\mathbf{u} = \mathbf{r} - \bar{F}(\mathbf{u})\mathbf{u} \tag{5.63}$$

can be rewritten as:

$$((S - L)\mathbf{u})_i = r_i + \bar{f}_i(\mathbf{u}) \tag{5.64}$$

for i being the number of the considered degree of freedom, with $\bar{f}_i(\mathbf{u})$ defined as [43]:

$$\bar{f}_i(\mathbf{u}) = \sum_{j \neq i} \alpha_{ij} f_{ij} \tag{5.65}$$

We will denote the vector of all $\bar{f}_i(\mathbf{u})$ as $\bar{\boldsymbol{f}}(\mathbf{u})$. Then, it is easy to see that:

$$\bar{\boldsymbol{f}}(\mathbf{u}) = \bar{F}(\mathbf{u})\mathbf{u} \tag{5.66}$$

With the chosen method the explicit construction of the anti-diffusion operator $\bar{F}(\mathbf{u})$ was avoided. The final problem to be solved reads:

$$(S - L)\mathbf{u} = \mathbf{r} + \bar{\boldsymbol{f}}(\mathbf{u}) \tag{5.67}$$

This is a non-linear algebraic system and therefore we need to employ a suitable method for solving such problems. In this thesis the **defect correction scheme** will be used for solving non-linear algebraic systems.

5.3.3 Defect correction scheme

The defect correction scheme is an iterative method for solving non-linear algebraic systems. The general idea is to start with the initial guess and then with each iteration evaluate the non-linear terms using the solution from the previous iteration. The iteration stops once a prescribed tolerance or the limit for the number of iterations is achieved. For the scheme (5.67) the algorithm reads:

1. Select the initial guess $\mathbf{u}^{(0)}$, for example:

$$\mathbf{u}^{(0)} = 0 \text{ or the solution of the low-order problem: } (S - L)\mathbf{u}^{(0)} = \mathbf{r} \tag{5.68}$$

2. Solve the linear system:

$$(S - L)\mathbf{u}^{(m)} = \mathbf{r} + \bar{\boldsymbol{f}}(\mathbf{u}^{(m-1)}) \tag{5.69}$$

3. Go to the 2. step unless the following condition is fulfilled:

$$||(S - L)\mathbf{u}^{(m)} - \mathbf{r} - \bar{\boldsymbol{f}}(\mathbf{u}^{(m)}|| < tol \text{ or } m = m_{max} \tag{5.70}$$

   where tol is a prescribed tolerance, m_{max} is a prescribed maximal number of iterations and $||\cdot||$ is a norm of interest.

Although the presented algorithm clearly illustrates the idea behind the defect correction approach, in practice an equivalent algorithm [11] is used more often:

1. Select the initial guess $\mathbf{u}^{(0)}$, for example:

$$\mathbf{u}^{(0)} = 0 \text{ or the solution of the low-order problem: } (S - L)\mathbf{u}^{(0)} = \mathbf{r} \tag{5.71}$$

2. Solve the linear system for the correction:

$$(S - L)\Delta\mathbf{u}^{(m)} = \mathbf{r} + \bar{\boldsymbol{f}}(\mathbf{u}^{(m-1)}) - (S - L)\mathbf{u}^{(m-1)} \tag{5.72}$$

   where the right hand side is a residual of the previous approximation.

3. Apply the correction to the solution:

$$\mathbf{u}^{(m)} = \mathbf{u}^{(m-1)} + \Delta\mathbf{u}^{(m)} \tag{5.73}$$

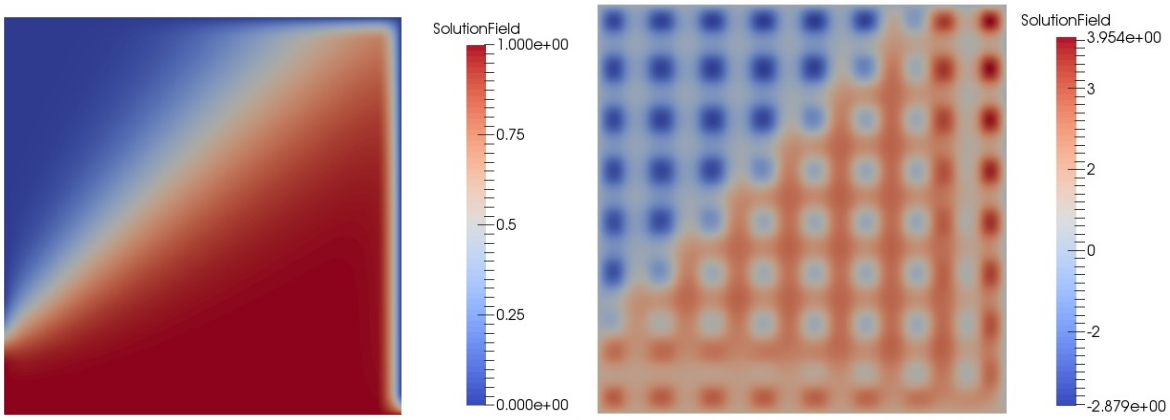
4. Go to the 2. step unless the following condition is fulfilled:

$$\frac{||\Delta \mathbf{u}^{(m)}||}{||\mathbf{u}^{(m)}||} < tol \quad \text{or} \quad m = m_{max} \tag{5.74}$$

Now, we have all necessary ingredients to implement the TVD-type flux limiting approach and apply it to the convection-diffusion equation. The numerical results obtained with this stabilization method are compared with results obtained with commonly used SUPG stabilization in the next section.

5.4 Numerical results - problem with internal and boundary layer

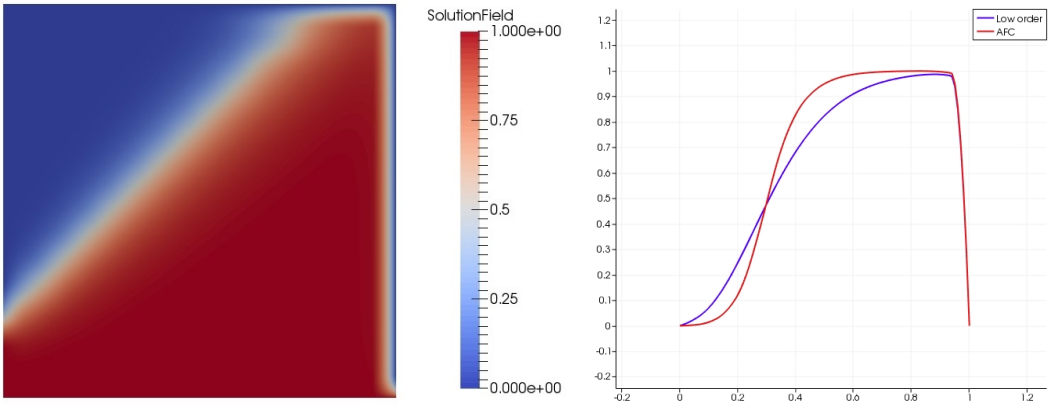
In this section we will discuss the results obtained with the Algebraic Flux Correction method used together with IGA for the benchmark problem (5.24) - (5.27). First of all, let us present the high- and low-order solutions of the benchmark problem with $d = 0.001$ ($Pe_h = 55.56$) to visualize their nature. Again, we use the space spanned by $p = 2$, 18×18 B-spline basis. The results are presented in Fig. 5.5.



(a) Low-order solution (b) High-order solution

Figure 5.5: Numerical results for high- and low-order problems for benchmark (5.24) - (5.27) for $d = 0.001$ solved on $p = 2$, 18×18 B-spline basis

It is easy to see that the low-order solution is free of oscillations which are present in the high-order solution. The price to be paid is a very diffusive nature of solution. There was artificial diffusion added and the result, although not oscillatory, is not accurate due to the smearing of the internal layer. AFC combines both methods to obtain a non-oscillatory and not too diffusive solution. The result of AFC stabilization is shown in Fig. 5.6a. Fig. 5.6b presents the section of the solution with use of AFC and the low-order solution (high-order solution was omitted because of high magnitudes of oscillations that would make the plot less clear) along the cut-line at $y = 0.5$. It is easy to see that the internal layer is much less smeared in case of AFC than in case of the low-order solution. Therefore, we can state that the quality of representation of the internal layer is much better in case of the solution obtained using AFC. The boundary layer is represented by both methods in the same way. It



(a) Result obtained with use of AFC (b) Solution along cut-line at $y = 0.5$.

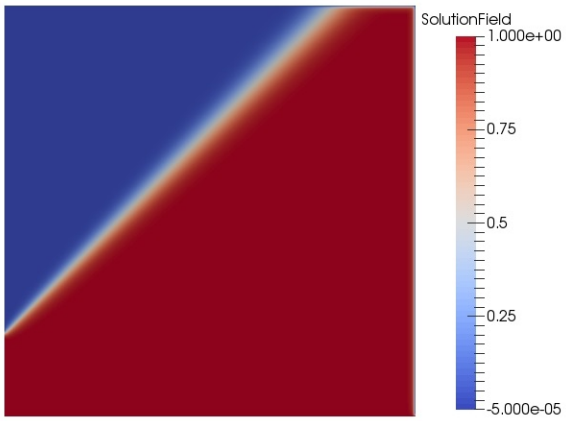
Figure 5.6: Numerical result obtained with use of AFC for benchmark problem (5.24) - (5.27) for $d = 0.001$ solved on $p = 2$, 18×18 B-spline basis.

is caused by the fact that this layer is very thin, much thinner than the single knot span of the basis. The expected layer size was estimated based on results presented in literature [1, 50].

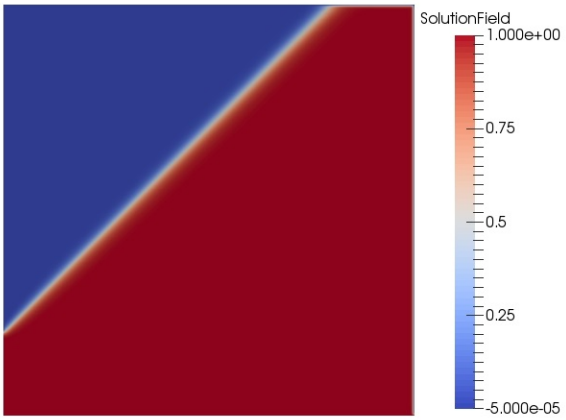
The next step is to validate the stabilization method by comparing the results obtained with AFC for the benchmark problem (5.24) - (5.27) with the results obtained with SUPG. The SUPG stabilization method for the convection-diffusion equation was implemented as part of the G+SMO library [30, 51]. As the unstabilized results are of a very poor quality we will not use them in this discussion. Fig. 5.7 presents the numerical results obtained with IGA stabilized by AFC and SUPG on the space spanned by $p = 2$ 130×130 B-spline basis for $d = 0.001$ and $d = 0.0001$. For the first problem with $d = 0.001$ both stabilization methods give very similar result except the fact that the result from SUPG has broader boundary layer. For $d = 0.0001$, the result from SUPG has non-physical over- and under-shoots. They appear for highly convection-dominated problems when the internal layers are becoming very steep. This is a significant drawback of the SUPG stabilization. On the other hand the internal layer is slightly smeared in case of AFC which is the price to be paid for having no under- and over-shoots. It is important to note that for the SUPG case the L2 projection was used for calculation of boundary values (due to software limitations) instead of constrained L2 used in the AFC case. This explains the under- and over-shoots appearing near the discontinuity on the west boundary.

Let us now have a closer look of how the numerical solution is changing with h- and p-refinements. Only the case for $d = 0.0001$ will be discussed as strongly convection-dominated problems are in general more sensitive to drawbacks of the numerical methods.

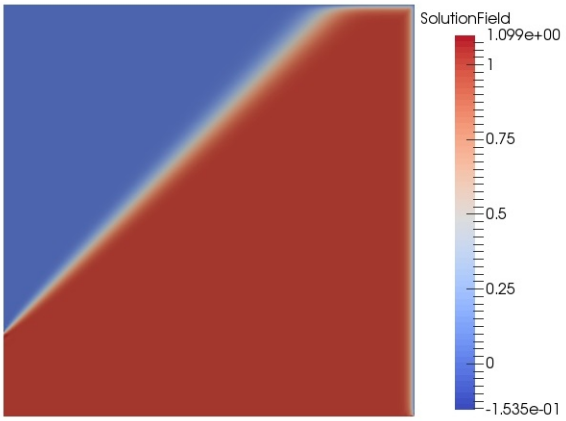
Firstly, let us analyse how the solution is changing for increasing number of degrees of freedom. The computations were made on a space spanned by $p = 2$ uniform B-spline basis with the same number of DOFs in both directions, the results along the cut-line at $y = 0.5$ are shown in Fig. 5.8. It is easy to see that both in case of AFC and SUPG the internal and the boundary layers are getting steeper with increasing number of degrees of freedom. This is caused by the fact that as the layers are very steep for strongly convection-dominated problems they require a fine mesh to be captured correctly. It can be also observed that the finer the mesh the smaller in magnitude are the oscillations in the solution produced by



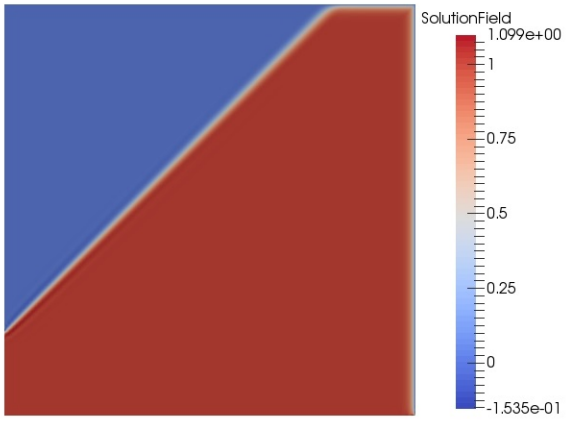
(a) AFC for $d = 0.001$



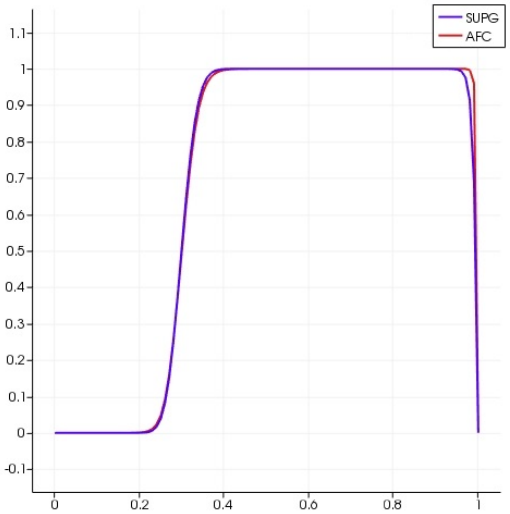
(b) AFC for $d = 0.0001$



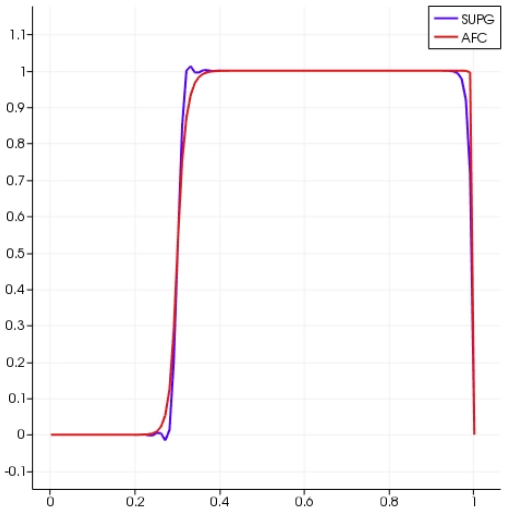
(c) SUPG for $d = 0.001$



(d) SUPG for $d = 0.0001$

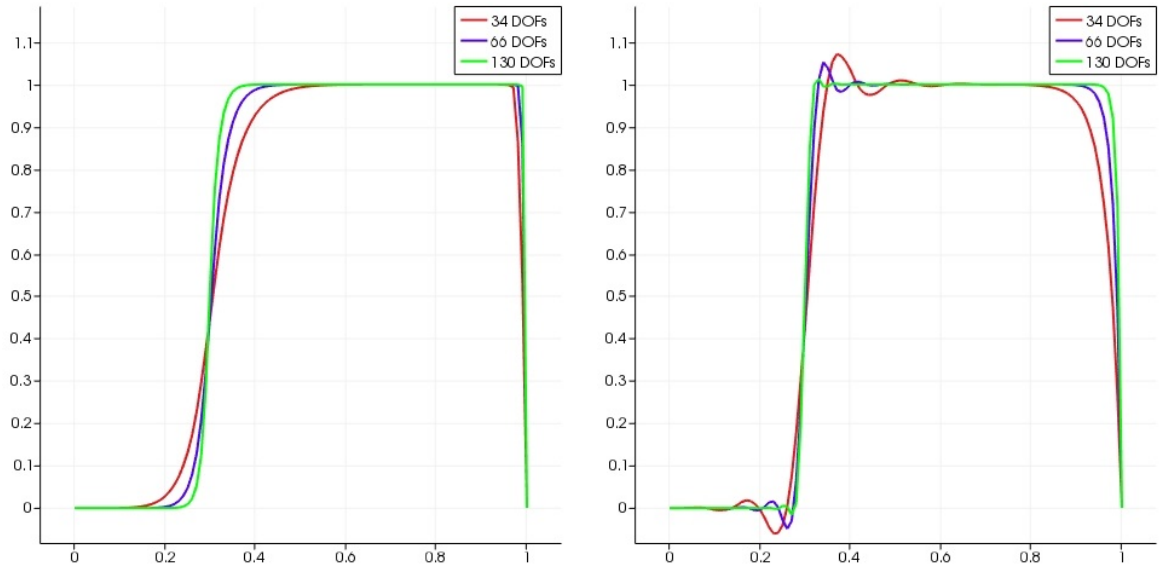


(e) Solution along cut-line at $y = 0.5$ for $d = 0.001$



(f) Solution along cut-line at $y = 0.5$ for $d = 0.0001$

Figure 5.7: Numerical result obtained with IGA with AFC and SUPG stabilizations for benchmark problem (5.24) - (5.27) with $d = 0.001$ and $d = 0.0001$ solved on $p = 2$, 130×130 B-spline basis.



(a) Result obtained with use of AFC

(b) Result obtained with use of SUPG

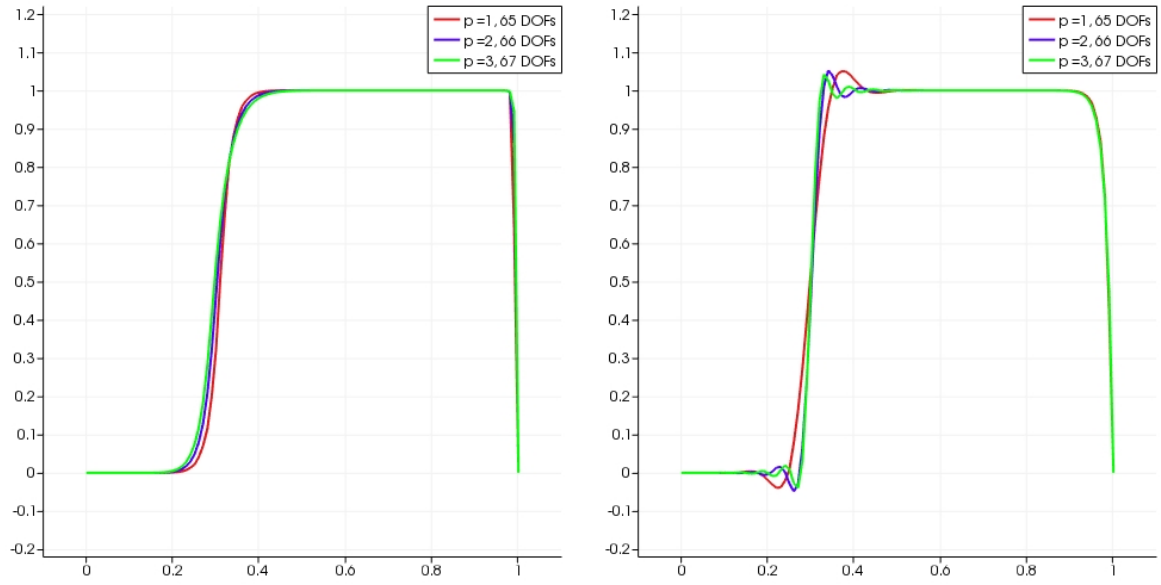
Figure 5.8: Numerical result obtained with IGA with AFC and SUPG stabilizations for benchmark (5.24) - (5.27) for $d = 0.0001$ solved with $p = 2$ B-spline basis and different numbers of degrees of freedom (the numbers presented in the figure are numbers of DOFs in one direction). Solution along cut-line at $y = 0.5$.

SUPG. In case of the solution from AFC the finer the mesh the less smeared is the internal layer.

Secondly, let us analyse how the solution is changing for increasing the order p of the basis. The computations were made with the use of the same number of DOFs in both directions and with uniform basis. The number of DOFs for different orders differs slightly because of the nature of the p-refinement but those additional single DOFs for each p-refinement are not affecting our conclusions. The results along the cut-line at $y = 0.5$ are shown in Fig. 5.9. It is easy to see that for this problem p-refinement does not change the quality of the solution as much as h-refinement. In case of AFC one should notice that the higher the order of the basis the more smeared is the internal layer. This unwanted behaviour is a consequence of increasing continuity - it is harder to represent the discontinuity (or very steep internal layer) without over- and under-shoots with high-order basis and therefore more artificial diffusion is necessary to avoid them. In case of SUPG the layer gets slightly steeper with increasing order and the oscillations at its ends have similar magnitude for different orders. Therefore, the SUPG is performing slightly better for higher orders (in contrast to AFC).

The last step of our comparison between SUPG and AFC is related to convergence analysis. Unfortunately, there is no exact solution known for benchmark (5.24) - (5.27). Therefore, the order of convergence will be estimated based on three consecutive numerical results u^{4h}, u^{2h} and u^h. The estimation is done using the formula [52]:

$$q = \frac{\log\left(\frac{\|u^{2h} - u^{4h}\|_2}{\|u^h - u^{2h}\|_2}\right)}{\log(2)} \tag{5.75}$$



(a) Result obtained with use of AFC

(b) Result obtained with use of SUPG

Figure 5.9: Numerical result obtained with IGA with AFC and SUPG stabilizations for benchmark (5.24) - (5.27) for $d = 0.0001$ solved with different orders of B-spline basis and equivalent number of DOFs (the numbers presented in the figure are numbers of DOFs in one direction). Solution along cut-line at $y = 0.5$.

where h denotes the grid spacing (width of knot span) in one direction. It is assumed that with each refinement step we half h. Therefore u^{4h} is the coarsest grid, u^{2h} is the middle and u^h is the finest one. The formula gives a good estimation only if all of the analysed mesh sizes are fine enough [52]. It is important to mention that the global h-refinement of a uniform basis is not exactly doubling the number of DOFs but resulting in $2n - p$ DOFs, with n being a number of DOFs and p the order of basis. Therefore, the formula gives a good approximation only for fine enough grids when $\frac{p}{n}$ is sufficiently small.

The orders of convergence calculated with the use of the above formula are presented in the table below. The calculations were done with uniform basis of orders $p = 1$ to $p = 4$ for the finest grid of 129 DOFs in one direction (for $p = 1$, for each p-refinement one additional DOF was added) for the benchmark problem (5.24) - (5.27).

p	AFC	SUPG
1	1.71	0.69
2	1.03	0.40
3	0.71	0.47
4	0.48	0.50

The highest orders of convergence were obtained for $p = 1$ basis due to the presence of very steep internal layer which is easier to be represented with low-order functions. For AFC the orders of convergence are significantly decreasing with increasing order of basis functions due to smearing of the internal layer to avoid oscillations. In contrast, the SUPG stabilization performs similarly for different orders of the basis as the magnitudes of under- and over-shoots do not change significantly. It is important to note that this is only an estimation of the orders

of convergence. Nevertheless, the obtained orders of convergence confirm quantitatively the conclusions derived from previous qualitative results.

It was shown that isogeometric analysis is a suitable tool to solve the stationary convection-diffusion equation. As in the case of the standard FEM the method performs well for very smooth and diffusion-dominated problems but applying it to convection-dominated problems requires application of some stabilization method. The method of choice in this thesis is algebraic flux correction with a limiter of TVD-type. The comparison of results obtained with AFC with the results from SUPG stabilization method show that AFC produces results free of non-physical over- and under-shoots that occur in case of SUPG. It was also shown that IGA with AFC is getting less robust with increasing orders of the basis for problems with internal layers. Let us now proceed to more complex problem: the time-dependent convection-diffusion equation.

Chapter 6

Time-dependent convection-diffusion equation

As shown in the previous chapter IGA with AFC stabilization is capable of solving stationary convection-diffusion equations. The next step to be taken is to consider time-dependent problems in which additional attention has to be paid to choosing suitable time discretization methods. Therefore, we discuss in this chapter the application of isogeometric analysis to the time-dependent convection-diffusion equation.

First of all, the description of the problem and the discretization is presented. The second section contains numerical results and reveals the expected stability problems for convection-dominated (in the limit - purely convective) problems. Further, we give a description of algebraic flux correction methods for time-dependent problems, which are used for stabilization. The last part of the chapter is devoted to numerical results.

6.1 Discretization of the time-dependent convection-diffusion equation

The **time-dependent convection-diffusion equation** describes the time-dependent behaviour of the fluid flow due to convective and diffusive phenomena. It is sometimes referred to as **scalar transport equation**. The time-dependent convection-diffusion equation is defined as [42]:

$$\frac{\partial u(\mathbf{x},t)}{\partial t} - \nabla \cdot (D \nabla u(\mathbf{x},t)) + \nabla \cdot (\mathbf{v}(\mathbf{x},t) u(\mathbf{x},t)) = R(\mathbf{x},t) \quad \textbf{in } \Omega \tag{6.1}$$

where $u(\mathbf{x},t)$ is the variable of interest (concentration of particles, energy, other physical quantity), D is the diffusion tensor (in simpler cases it can be replaced by scalar diffusion coefficient d), $\mathbf{v}(\mathbf{x},t)$ is an average velocity field (for the clarity of presentation the dependency on $\mathbf{x}$ will be not included in the equation in further parts of this chapter) and $R(\mathbf{x},t)$ is a source term.

The problem has to be completed with properly chosen boundary conditions (to assure well-posedness):

$$u(\mathbf{x},t) = \gamma(\mathbf{x},t) \ \text{ on } \Gamma_D \tag{6.2}$$

$$\frac{\partial u}{\partial \mathbf{n}}(\mathbf{x},t) = \beta(\mathbf{x},t) \ \text{ on } \Gamma_N \tag{6.3}$$

where $\mathbf{n}$ is the unit outward normal vector, and with the initial condition:

$$u(\mathbf{x}, 0) = u_0(\mathbf{x}), \quad \forall \mathbf{x} \in \Omega \tag{6.4}$$

The weak form of the problem (obtained after multiplication by a test function and integration by parts applied to the diffusive term only) reads:

$$\text{find } u \in S \text{ such that for all } v \in V,$$
$$\int_\Omega \frac{\partial u}{\partial t} v d\mathbf{x} = -\int_\Omega ((\nabla \cdot (\mathbf{v}u))v + (D\nabla u) \cdot \nabla v) d\mathbf{x} + \int_\Omega R v d\mathbf{x} + \int_\Gamma D \frac{du}{d\mathbf{n}} v ds \tag{6.5}$$

6.1.1 Spatial discretization

The spatial discretization of the weak form is performed using the standard Galerkin method. Then, the semi-discretized variational formulation reads:

$$\text{find } u^h \in S^h \text{ such that for all } v^h \in V^h,$$

$$\int_\Omega \frac{\partial u^h}{\partial t} v^h d\mathbf{x} = -\int_\Omega ((\nabla \cdot (\mathbf{v}u)^h)v^h + (D\nabla u^h) \cdot \nabla v^h) d\mathbf{x} + \int_\Omega R v^h d\mathbf{x} + \int_{\Gamma_N} D\beta v^h ds \tag{6.6}$$

Test functions v^h (because of assumed restrictions on the test space) vanish on the Dirichlet part of the boundary Γ_D and, therefore, we only need to consider the integration over Γ_N instead of the whole Γ [42]. In the framework of IGA the considered discrete spaces are spanned by B-spline (NURBS) basis functions. Using the Fletcher's group formulation [41], as in the previous chapter, the approximate solution u^h and the convective flux $(\mathbf{v}u)^h$ can be represented in terms of the basis $\{\varphi_j(\mathbf{x})\}$ of V^h as follows [42]:

$$u^h(\mathbf{x}, t) = \sum_j u_j(t)\varphi_j(\mathbf{x}) \tag{6.7}$$

$$(\mathbf{v}u)^h(\mathbf{x}, t) = \sum_j \mathbf{v}_j(t)u_j(t)\varphi_j(\mathbf{x}) \tag{6.8}$$

Then, knowing that it is sufficient to test the variational formulation only with the basis functions of V^h, the semi-discrete problem can be rewritten in the form:

$$\sum_j \left(\int_\Omega (\varphi_j \varphi_i d\mathbf{x} \right) \frac{du_j}{dt} =$$
$$\sum_j \left(-\int_\Omega ((D\nabla\varphi_j) \cdot \nabla\varphi_i) d\mathbf{x} - \int_\Omega (\nabla \cdot (\mathbf{v}_j\varphi_j))\varphi_i d\mathbf{x} \right) u_j + \int_\Omega R\varphi_i d\mathbf{x} + \int_{\Gamma_N} D\beta\varphi_i ds \tag{6.9}$$

This equation can be rewritten in matrix form:

$$M_C \frac{d\mathbf{u}}{dt} = (K - S)\mathbf{u} + \mathbf{r} \tag{6.10}$$

Where $M_C = \{m_{ij}\}$ is the **consistent mass matrix** with entries:

$$m_{ij} = \int_\Omega \varphi_j \varphi_i d\mathbf{x} \tag{6.11}$$

$S = \{s_{ij}\}$ is the **discrete diffusion operator** with entries:

$$s_{ij} = \int_{\Omega} (D\nabla\varphi_j \cdot \nabla\varphi_i) d\mathbf{x} \tag{6.12}$$

$K = \{k_{ij}\}$ is the **discrete diffusion operator** with entries:

$$k_{ij} = -\mathbf{v}_j \cdot \mathbf{c}_{ij} \tag{6.13}$$

$$\mathbf{c}_{ij} = \int_{\Omega} \nabla\varphi_j \varphi_i d\mathbf{x} \tag{6.14}$$

$\mathbf{u}$ is the vector of solution coefficients u_i and $\mathbf{r}$ is a right-hand side vector, resulting from the source term and treatment of boundary conditions, with entries r_i::

$$r_i = \int_{\Omega} R\varphi_i dx + \int_{\Gamma_N} D\beta\varphi_i ds \tag{6.15}$$

The treatment of the mapping from the parametric domain to the physical domain in the assembly of matrices is analogous to the one used in the previous chapters.

6.1.2 Temporal discretization

The next step is to discretize the semi-discrete problem (6.10) in time. There is a large variety of time-discretization schemes. For this thesis work, three schemes were chosen: first-order forward Euler method and optimal strong stability-preserving (SSP) explicit Runge-Kutta (RK) methods of orders 2 and 3. We briefly introduce them below. Let the semi-discrete problem (6.10) be expressed in the more general form:

$$\frac{d\mathbf{u}}{dt} = F(\mathbf{u}) \tag{6.16}$$

That is, we formally define:

$$F(\mathbf{u}) = M_C^{-1} \left[(K - S)\mathbf{u} + \mathbf{r}\right] \tag{6.17}$$

We denote the time discretization step by Δt.

First-order Forward Euler method

The first-order forward Euler is the simplest time-discretization method. It is an explicit method of the form [53]:

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \Delta t F(\mathbf{u}^n) \tag{6.18}$$

Second-order Strong Stability-Preserving explicit Runge-Kutta method

The SSP-eRK-2 method is a time discretization method of the form [54]:

$$\mathbf{u}^{(1)} = \mathbf{u}^n + \Delta t F(\mathbf{u}^n) \tag{6.19}$$

$$\mathbf{u}^{n+1} = \frac{1}{2}\mathbf{u}^n + \frac{1}{2}\mathbf{u}^{(1)} + \frac{1}{2}\Delta t F(\mathbf{u}^{(1)}) \tag{6.20}$$

Third-order Strong Stability-Preserving explicit Runge-Kutta method

The SSP-eRK-3 method is a time discretization method of the form [54, 55]:

$$\mathbf{u}^{(1)} = \mathbf{u}^n + \Delta t F(\mathbf{u}^n) \tag{6.21}$$

$$\mathbf{u}^{(2)} = \frac{3}{4}\mathbf{u}^n + \frac{1}{4}\mathbf{u}^{(1)} + \frac{1}{4}\Delta t F(\mathbf{u}^{(1)}) \tag{6.22}$$

$$\mathbf{u}^{n+1} = \frac{1}{3}\mathbf{u}^n + \frac{2}{3}\mathbf{u}^{(2)} + \frac{2}{3}\Delta t F(\mathbf{u}^{(2)}) \tag{6.23}$$

The **Courant-Friedrichs-Levy (CFL) condition** is a necessary condition for the convergence of any explicit numerical scheme applied to partial differential equation. In the 2D case, which is considered in the thesis, the condition reads [56]:

$$\frac{|v_1|\Delta t}{h_1} + \frac{|v_2|\Delta t}{h_2} \leq C_{\max} \tag{6.24}$$

where h_1 and h_2 are grid sizes and v_1 and v_2 are the velocity components in x and y directions, respectively. The limit value $C_{\max}$ depends on the adopted method. For explicit methods (all the methods considered in the thesis are explicit) $C_{\max} = 1$. It is important to note that it is not a sufficient condition [54].

The choice of those three methods is not random. Let us assume that a spatial discretization scheme is combined with the forward Euler temporal discretization with sufficiently small time step size to satisfy the CFL condition has the TVD property (see section 5.3.1). The higher-order SSP-eRK methods are designed in such a way that they achieve higher order of convergence in time than the forward Euler method and do not loose the TVD property if applied together with the same spatial discretization scheme [54].

6.2 Numerical Results

The IGA-based solver for time-dependent convection-diffusion equations was applied to two convection-dominated problems: convection of a smooth hump and convection of a rectangular wave. The numerical results are presented below.

6.2.1 Convection of a smooth hump

The first problem analysed deals with the convection of a smooth hump:

$$\frac{\partial u(\mathbf{x}, t)}{\partial t} + \nabla \cdot (\mathbf{v} u(\mathbf{x}, t)) = 0 \quad \text{in } [0, 1] \times [0, 2] \tag{6.25}$$

$$\mathbf{v} = [10, 0]^T \tag{6.26}$$

with homogeneous Dirichlet boundary condition at the Western boundary Γ^W and homogeneous Neumann boundary condition at Northern and Southern boundaries $\Gamma^N \cup \Gamma^S$ (for the

hyperbolic problem no boundary condition is prescribed at the outflow boundary, in this case the Eastern boundary Γ^E), i.e.:

$$u(\mathbf{x}, t) = 0 \ \text{ on } \Gamma^W \tag{6.27}$$

$$\frac{\partial u}{\partial \mathbf{n}}(\mathbf{x}, t) = 0 \ \text{ on } \Gamma^N \cup \Gamma^S \tag{6.28}$$

The initial condition is a hump centred at $(x_0, y_0) = (0.5, 0.5)$ defined as:

$$u(\mathbf{x}, 0) = 0.25\,[1 + \cos(\pi \min\{r(\mathbf{x}), 1\})] \tag{6.29}$$

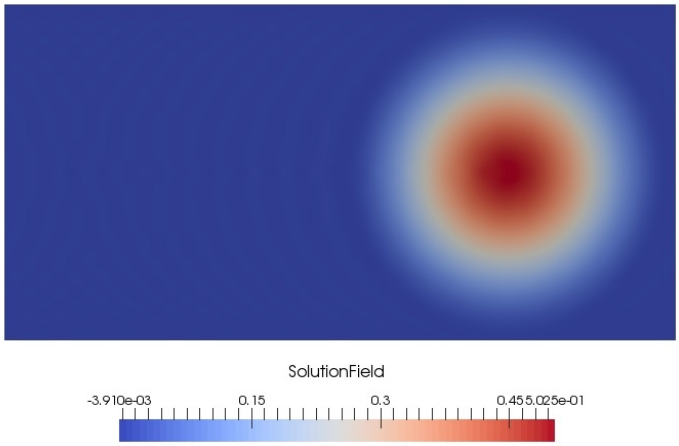
with:

$$r(\mathbf{x}) = \frac{1}{r_0}\sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{6.30}$$

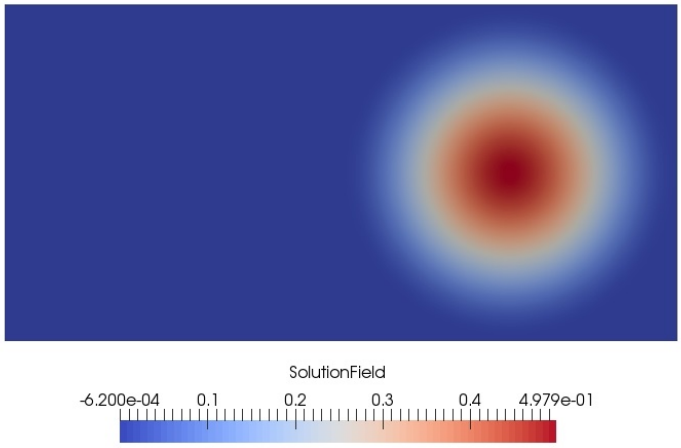
and $r_0 = 0.5$. It is easy to notice that the exact solution to this problem is a hump centred at $(x_t, y_t) = (0.5 + 10t, 0.5)$. In this section we are interested in the numerical solution to the problem at $t = 0.1$. At this time the exact solution is just the translation the initial condition by $\Delta x = 1$ in x direction.

Fig. 6.1 presents the numerical solutions to the problem defined above using the Forward Euler, SSP-eRK-2 and SSP-eRK-3 methods. The computations were done on the space spanned by uniform $p = 2$, 66×66 B-spline basis. The used time step was of size $\Delta t = 0.0001$. All results in general represent the exact solution correctly. However, in the solution obtained with the Forward Euler method there are some oscillations visible. The more clear comparison of results obtained with those three methods is presented in Fig. 6.2 by plotting the solution along the x direction at $y = 0.5$. It is important to note that the exact solution presented in the figures is an optimal, with respect to L2 norm, representation of the analytical solution on the considered B-spline space, and therefore, the best solution achievable on a given mesh. The results from SSP-eRK methods of orders 2 and 3 coincide almost perfectly with the exact solution. The result from the Forward Euler method is of significantly worse quality. The conclusion is that the chosen time step is sufficiently small in case of SSP-eRK methods of orders 2 and 3 (assuming that our goal is a visual agreement of exact and numerical solutions) while for the Forward Euler method smaller time step sizes are necessary.

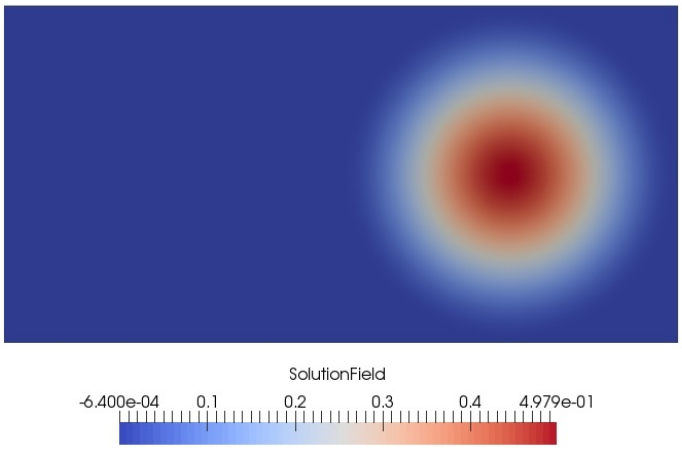
To understand the behaviour of the Forward Euler method and the SSP-eRK methods with varying time step sizes consider Fig. 6.3. For each of the three methods, the curve has a similar structure. Firstly, the error is increasing when the time step is decreased. This happens for a very coarse time discretization, which does not fulfil the stability condition (CFL condition). Theoretically, the value of the largest time step that fulfils the CFL condition is approximately $\Delta t_{\max} \approx 0.003$. It can be easily seen that all three methods start to converge once the value of Δt is smaller than that limiting value. The SSP-eRK methods start to converge even for larger values of Δt. In the second part of the curve the methods give better results when the time step is decreased. As expected the steepest slope in this region is recorded for the third-order SSP-eRK method. The slowest convergence takes place for the Forward Euler method, which is a first-order method. As expected we can deduct that the higher the order of the method the steeper is the slope of the curve in this region. The third region of the curve is a plateau below the steep part. In this region the methods are not converging any further. Decreasing the time step after achieving this plateau is not improving the result any more. This is caused by the fact that the spatial error starts to dominate and further decreasing the time step does not decrease the overall error. All the considered time



(a) Forward Euler method



(b) SSP-eRK-2 method



(c) SSP-eRK-3 method

Figure 6.1: Numerical results for the problem of convection of a smooth hump solved on uniform $p = 2$, 66×66 B-spline basis with different time discretization schemes, with time step $\Delta t = 0.0001$.

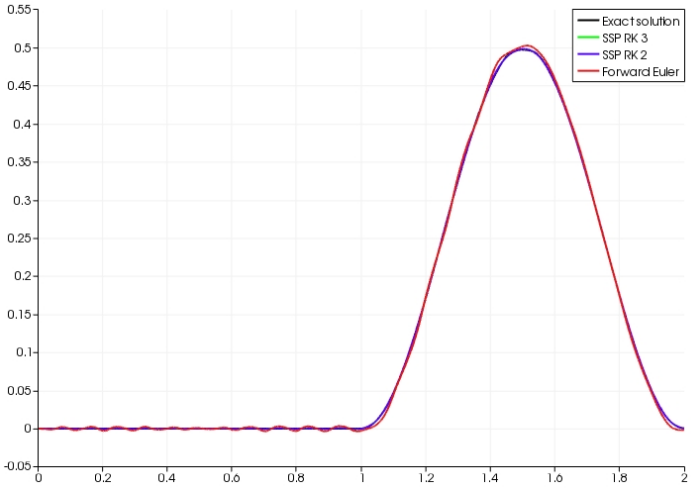


Figure 6.2: Numerical results at $t = 0.1$ for the problem of convection of a smooth hump solved on uniform $p = 2$, 66×66 B-spline basis with different time discretization schemes, with time step $\Delta t = 0.0001$. Section along x direction at $y = 0.5$.

discretization methods have the plateau at the same L2 error. The only difference is how small the time steep needs to be to get there.

The results from Fig. 6.1 and 6.2 were obtained with $\Delta t = 0.0001$. We can see from Fig. 6.3 that for this time step size the SSP-eRK-2 and SSP-eRK-3 methods already achieved that maximum quality of solution possible with the given spatial discretization while the Forward Euler method is still converging towards it. That explains the overlap of solutions from SSP-eRK methods with the exact solution and oscillations that occur in the solution from the Forward Euler method. Obviously, the optimal value of Δt for a given method and a given spatial discretization is the one at which the plateau starts. The SSP-eRK-3 method has the significant advantage that the convergence is very quick. Unfortunately, the price to be paid is that it requires more computations per time step than SSP-eRK-2 method and the Forward Euler method. The SSP-eRK-2 method converges slower but the computational costs per iteration are lower as well. The Forward Euler method converges very slowly which disqualifies it from most of the practical applications but the simplicity of the idea behind it, ease of implementation and small computational effort per iteration make it a perfect choice for the first method to be implemented for test purposes.

We can conclude the results obtained from this test problem that the IGA approach works fine for simple smooth time-dependent problems. It is however very important to carefully choose the time step size to avoid instabilities or excessive computational effort spent on time steps that do not improve the quality of the solution.

6.2.2 Convection of a rectangular wave

The second problem deals with the convection of a rectangular wave (it is a 1D problem prolonged artificially in the second dimension to be consistent with other test problems presented in this chapter). A similar problem was used by Kuzmin in [44]. The problem is defined exactly as the previous one (6.25)-(6.28), the only difference is the initial condition defined now

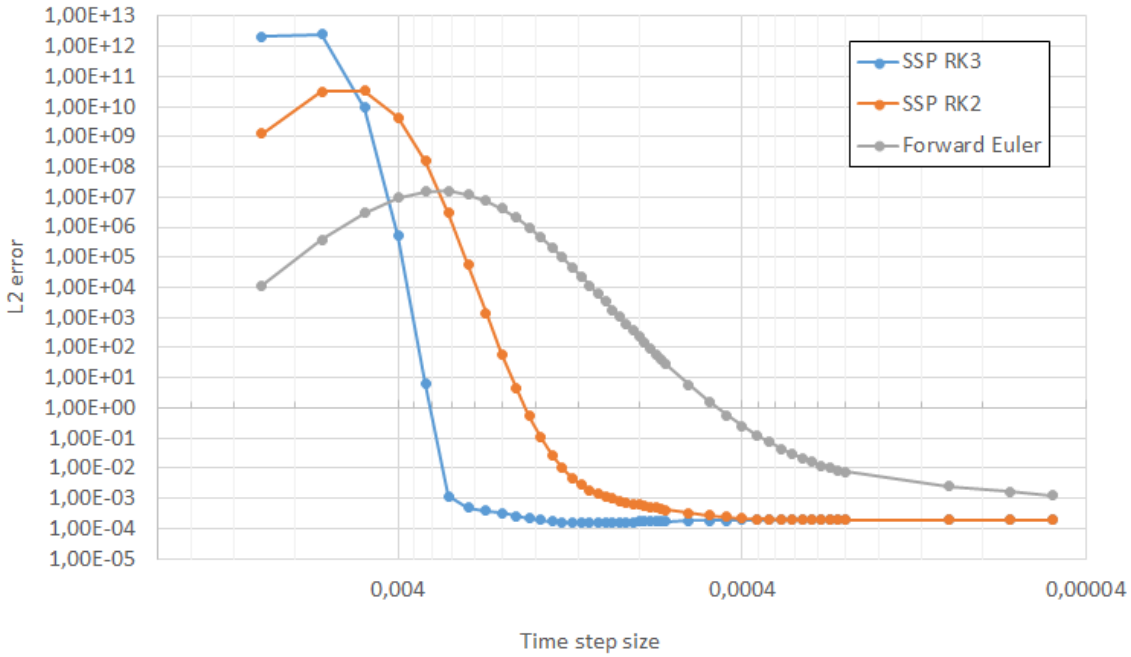


Figure 6.3: L2 errors of numerical solutions at $t = 0.1$ of the problem of convection of a smooth hump solved on uniform $p = 2$, 66×66 B-spline basis with different time discretization schemes versus the time step size.

by:

$$u(\mathbf{x}, 0) = \begin{cases} 1 & \text{if } x \in (0.25, 0.75) \\ 0 & \text{otherwise} \end{cases} \tag{6.31}$$

Obviously, the exact solution of this problem is given by:

$$u(\mathbf{x}, t) = \begin{cases} 1 & \text{if } x \in (0.25 + 10t, 0.75 + 10t) \\ 0 & \text{otherwise} \end{cases} \tag{6.32}$$

In this section we are interested in the numerical solution of the problem at $t = 0.1$. Again, at this time the exact solution is just the translation of the initial condition with by $\Delta x = 1$ in x direction.

Fig. 6.4 presents the exact solution and the numerical solution computed by SSP-eRK-3 to the problem defined above in form of the section along the x direction at $y = 0.5$. We restrict ourself to this method as it (as shown above) yields the fastest convergence. The computations were performed on the space spanned by uniform $p = 2$, 66×66 B-spline basis. The time step was of size $\Delta t = 0.0001$. The initial condition was projected to the considered B-spline space using the constrained L2 projection to avoid non-physical under- and over-shoots. The "exact solution" is here a result of constrained L2 projection of the analytical solution to the considered space. The obtained result is very oscillatory although for the previous problem the SSP-eRK methods with the same time step and spatial grid sizes produced results of very good quality. The discontinuities in the problem caused the method to fail.

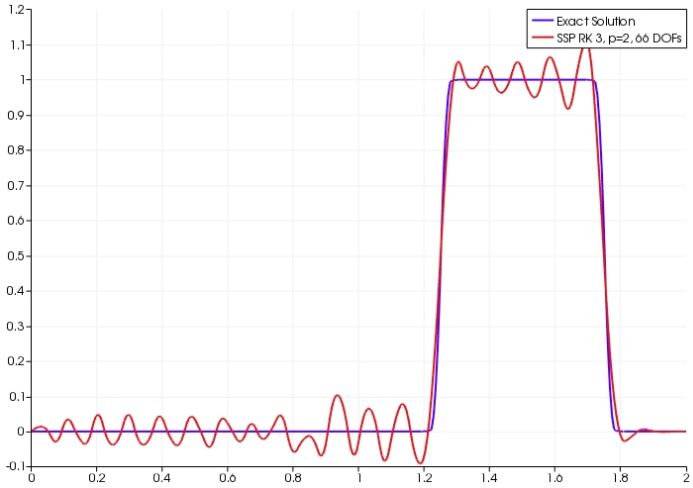


Figure 6.4: Exact and numerical solution at $t = 0.1$ to the problem of convection of a rectangular wave solved on uniform $p = 2$, 66×66 B-spline basis with SSP-eRK-3 time discretization method, with time step $\Delta t = 0.0001$. Section along x direction at $y = 0.5$.

One might think that the oscillations would disappear or decrease for small enough time step. Fig. 6.5 shows clearly that this is not the case. The region in which the method is quickly converging has not moved significantly. The time step $\Delta t = 0.0001$ used above is already at the plateau so further refinement of the time step size is pointless.

From the example above one can easily conclude that in order to be able to solve the time-dependent convection-diffusion equations with discontinuities or very steep gradients (that can not be represented by the mesh), there is a need to find a suitable stabilization method. As in the stationary case we use the stabilization technique from the family of AFC algorithms.

6.3 Algebraic flux correction

For the time-dependent convection-diffusion equation we will use algebraic flux correction with **flux limiting of FCT-type** (in contrast to limiting of TVD-type used for stationary problems in Chapter 5). It was described by D. Kuzmin in [12] as well as in [42]. This section presents a brief description of the method applied to the time-dependent convection-diffusion problem.

6.3.1 The family of FCT algorithms

Flux-corrected transport (FCT) algorithms were introduced by Boris and Book in [57, 58]. They are non-linear high-resolution schemes that assure the monotonicity (and therefore prevent the generation of non-physical oscillations) even for pure convection problems involving discontinuities. They belong to the class of so-called **diffusion-anti-diffusion (DAD)** methods. The main idea of the algorithm can be described as predictor-corrector method consisting of the following steps [12, 42]:

1. Advance the solution in time with the use of a low-order method.

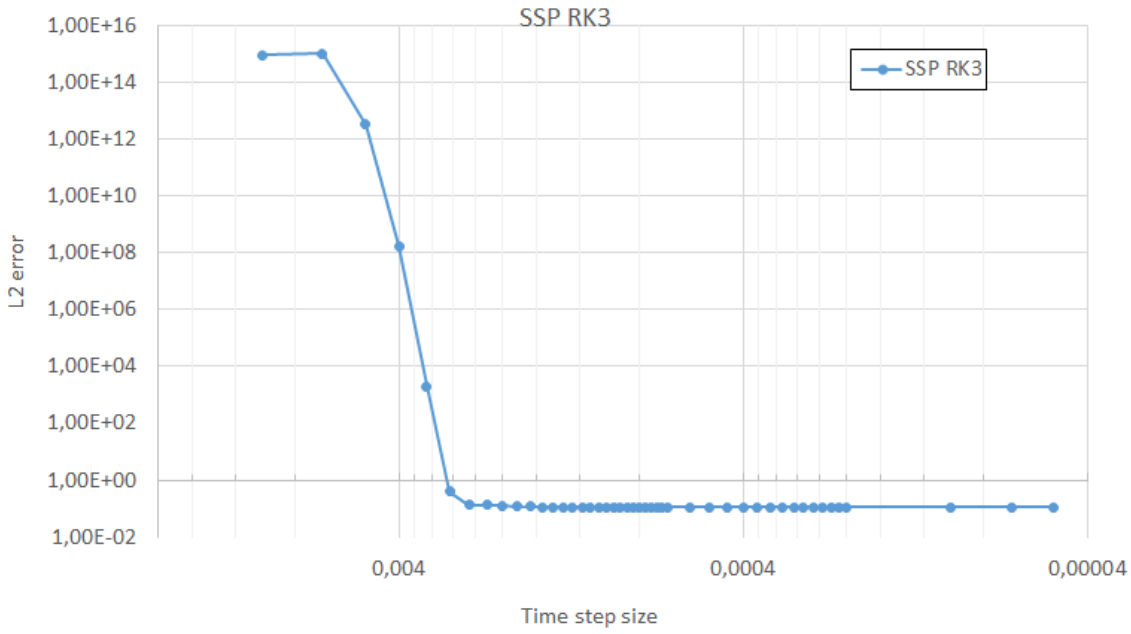


Figure 6.5: L2 errors of the numerical solution at time $t = 0.1$ to the problem of convection of a rectangular wave solved on uniform $p = 2$, 66×66 B-spline basis with SSP-eRK-3 time discretization method versus time step.

2. Correct the solution by applying limited **anti-diffusion**.

The low-order method must be designed to assure that no non-physical under- and over-shoots are generated by incorporating enough **numerical diffusion** [12]. Raw anti-diffusion is defined such that applied to the low-order scheme it should restore the original high-order scheme [42]. The anti-diffusion has to be limited in such a way that, after applying it to the solution computed by the low-order method, no new extrema are created and the existing ones do not grow in magnitude (i.e. to preserve the LED property defined in Section 5.3.1) [12, 42].

6.3.2 Algebraic flux correction with limiter of FCT-type

Zalesak's fully multidimensional FCT algorithm [38] forms the basis for most AFC schemes based on the idea of FCT. The general idea is to use the scheme resulting from the Galerkin FEM (IGA) discretization as the high-order method. The low-order method is then obtained from the high-order method by performing purely algebraic operations on the discrete operators. Kuzmin in [42] states that the non-physical oscillations are caused by:

- positive off-diagonal entries in the consistent mass matrix M_C
- negative off-diagonal entries in the discrete convection operator K
- positive off-diagonal entries in the discrete diffusion operator S

As in the previous chapter, the discrete diffusion operator S is not causing problems as all off-diagonal entries are non-positive unless the mesh or the diffusion tensor are highly

anisotropic [42] (in our case D is a scalar coefficient so the diffusion tensor is diagonal and therefore isotropic by definition). To avoid positive off-diagonal entries in the mass matrix we perform row-sum mass lumping according to (4.10). To avoid negative off-diagonal entries in the discrete convection operator K we add to it the artificial diffusion operator D to obtain L according to the procedure (5.45)-(5.50). After those operations have been performed we obtain the low-order scheme as [42]:

$$M_L \frac{d\mathbf{u}}{dt} = (L - S)\mathbf{u} \tag{6.33}$$

The solution to this low-order problem will be denoted by $\mathbf{u}^L$ with entries u_i^L.

The **anti-diffusion** defined as the difference between the low-order method (6.33) and the high-order method (6.10) is given by:

$$\mathbf{f}(\mathbf{u}) = (M_L - M_C)\frac{d\mathbf{u}}{dt} + (K - L)\mathbf{u} = (M_L - M_C)\frac{d\mathbf{u}}{dt} - D\mathbf{u} \tag{6.34}$$

By construction, the operators $(M_L - M_C)$ and D are discrete diffusion operators (see (5.39)) and can therefore be decomposed according to (5.40)-(5.41). That is [42]:

$$(M_L\mathbf{u} - M_C\mathbf{u})_i = \sum_{j \neq i} m_{ij}(u_i - u_j) \tag{6.35}$$

$$-(D\mathbf{u})_i = \sum_{j \neq i} d_{ij}(u_i - u_j) \tag{6.36}$$

We can now represent the anti-diffusive term $\mathbf{f}(\mathbf{u}) = \{f_i(\mathbf{u})\}$ as the net sum of fluxes between DOFs [42]:

$$f_i(\mathbf{u}) = \sum_{j \neq i} f_{ij}(\mathbf{u}) \tag{6.37}$$

with raw anti-diffusive fluxes:

$$f_{ij}(\mathbf{u}) = \left(m_{ij}\frac{d}{dt} + d_{ij}\right)(u_i - u_j) \tag{6.38}$$

Obviously, the scheme:

$$M_L \frac{d\mathbf{u}}{dt} = (L - S)\mathbf{u} + \mathbf{f}(\mathbf{u}) \tag{6.39}$$

is equivalent to the high-order scheme (6.10). According to the framework of FCT we add only a limited portion of the anti-diffusion [33]:

$$M_L \frac{d\mathbf{u}}{dt} = (L - S)\mathbf{u} + \bar{\mathbf{f}}(\mathbf{u}) \tag{6.40}$$

where the entries $\bar{f}_i(\mathbf{u})$ of $\bar{\mathbf{f}}(\mathbf{u})$ are given by sums of constrained anti-diffusive fluxes:

$$\bar{f}_i(\mathbf{u}) = \sum_{j \neq i} \alpha_{ij}(\mathbf{u}) f_{ij}(\mathbf{u}) \tag{6.41}$$

The **limiting coefficients** $\alpha_{ij}(\mathbf{u})$ are calculated by Zalesak's multidimensional FCT limiter [38, 12, 42] generalized for non-nodal DOFs. The notation of dependency on the solution $(\mathbf{u})$ is dropped in the following description of the algorithm for clarity of presentation. Zalesak's limiting algorithm generalised for non-nodal DOFs consist of the following steps:

1. Evaluate the sums of positive and negative anti-diffusive fluxes into i-th DOF:

$$P_i^+ = \sum_{j \neq i} \max\{0, f_{ij}\}, \qquad P_i^- = \sum_{j \neq i} \min\{0, f_{ij}\} \tag{6.42}$$

2. Compute the distance to a local maximum and minimum at the bounds:

$$Q_i^+ = \frac{m_i}{\Delta t}(u_i^{\max} - u_i^L), \qquad Q_i^- = \frac{m_i}{\Delta t}(u_i^{\min} - u_i^L) \tag{6.43}$$

3. Determine the nodal correction factors for the net increment at i-th DOF:

$$R_i^+ = \min\left\{1, \frac{Q_i^+}{P_i^+}\right\}, \qquad R_i^- = \min\left\{1, \frac{Q_i^-}{P_i^-}\right\} \tag{6.44}$$

4. Control the sign of the raw anti-diffusive flux:

$$\alpha_{ij} = \begin{cases} \min\{R_i^+, R_j^-\}, & \text{if } f_{ij} > 0 \\ \min\{R_i^-, R_j^+\}, & \text{if } f_{ij} < 0 \end{cases} \tag{6.45}$$

6.3.3 Linearisation of anti-diffusive fluxes

After applying a suitable time discretization the resulting scheme is non-linear. It is possible to use the defect-correction scheme to solve this system, however, as stated by Kuzmin in [12] this approach is not efficient due to high computational cost. Therefore, a suitable linearisation would significantly increase efficiency of the method for the price of slightly less accurate solutions. Although linearisation about the low- or high-order solutions seems the most natural approach, it was shown by Kuzmin in [12] that both of those approaches are inefficient. Therefore, another linearisation approach was proposed in the same paper: to compute the low-order end-of-step solution u^L and correct it explicitly (similar to the case of classical DAD methods), namely [12]:

$$M_L \mathbf{u}^{n+1} = M_L \mathbf{u}^L + \Delta t \bar{\mathbf{f}}(\mathbf{u}^L, \mathbf{u}^n) \tag{6.46}$$

where $\mathbf{u}^n$ is a numerical solution at time step n and $\mathbf{u}^L$ is the end-of-step low-order solution. It can be calculated with any time discretization method (eg. Forward Euler method, SSP-eRK methods). The net sums of raw anti-diffusive fluxes after linearisation are defined as [12]:

$$f_i(\mathbf{u}^L, \mathbf{u}^n) = \sum_{j \neq i} f_{ij}(\mathbf{u}^L, \mathbf{u}^n) \tag{6.47}$$

with

$$f_{ij}(\mathbf{u}^L, \mathbf{u}^n) = m_{ij}(\dot{u}_i^L - \dot{u}_j^L) + d_{ij}^{n+1}(u_i^L - u_j^L) \tag{6.48}$$

Here $\dot{u}_i^L$ denotes i-th entry of the finite difference approximation of the time derivative $\dot{\mathbf{u}}^L$ and d_{ij}^{n+1} denotes the values of d_{ij} computed for operator K evaluated at time $n+1$. In the constant-coefficient, time-dependent convection-diffusion equations, considered in this thesis, the operator K is constant in time and therefore d_{ij} does not have to be recomputed for every time step. This, however, is not the case for the compressible Euler equations considered in the further part of this thesis. There are several methods to compute this value. In this thesis

we simply solve the low-order problem which amounts to scaling the RHS by the inverse of diagonal values of the lumped mass matrix, namely [12]:

$$M_L \dot{\mathbf{u}}^L = (L - S)\mathbf{u}^L \tag{6.49}$$

Note that because of this approximation adding the raw anti-diffusive fluxes to the low-order solution will no longer restore the original high-order solution [42]. This approximation guarantees that the corrected end-of-step solution is free of oscillations. However, in some cases more accurate and still not oscillatory results can be obtained using other approximations as proposed by Kuzmin in [12]:

$$M_L \dot{\mathbf{u}}^L = (K - S)\mathbf{u}^L \tag{6.50}$$

or the original high-order solution which is unfortunately prone to oscillations:

$$M_C \dot{\mathbf{u}}^L = (K - S)\mathbf{u}^L \tag{6.51}$$

Now, we have all necessary ingredients at hand to implement the AFC method with FCT-type flux limiter and apply it to the time-dependent convection-diffusion equation. The analysis of the numerical results obtained with IGA combined with this stabilization method are presented in the next section.

6.4 Numerical results

In this section we will discuss the results obtained from using the Algebraic Flux Correction stabilization method in the IGA framework applied to two benchmark problems. The first problem describes the convection of a rectangular wave, which was already analysed in this chapter without application of the stabilization method. The second one is a common benchmark for time-dependent convection-diffusion equations - the rotation of three solid bodies.

6.4.1 Convection of rectangular wave

As it was shown in Fig. 6.4, IGA without stabilization is not able to solve the problem of convection of a rectangular wave without generating spurious oscillations. Therefore, the AFC stabilization technique was applied to stabilize the IGA approach for this problem. Fig. 6.6 presents the exact and numerical solutions to this problem. The numerical results were obtained with use of the unstabilized IGA Galerkin method, the low-order scheme and the AFC-stabilized scheme. All computations were done on the space spanned by uniform $p = 2$, 66×66 B-spline basis. Time discretization was done with SSP-eRK-3 time discretization method and with the time step of size $\Delta t = 0.0001$. As it was already mentioned the result from the high-order method is not acceptable due to oscillations. The low-order scheme obtained after mass lumping and adding the artificial diffusion gives overly diffusive and therefore very inaccurate results. The solution obtained with AFC is much less diffusive than the one obtained from the low-order scheme and still free of non-physical oscillations but the price for that is a lower accuracy in representation of discontinuities which are smeared. Use of AFC with FCT-type limiter allows us reach our goal (the same as in case of constrained L2 projection and stationary convection-diffusion equation) - to produce solutions, which are free of non-physical oscillations and still accurate.

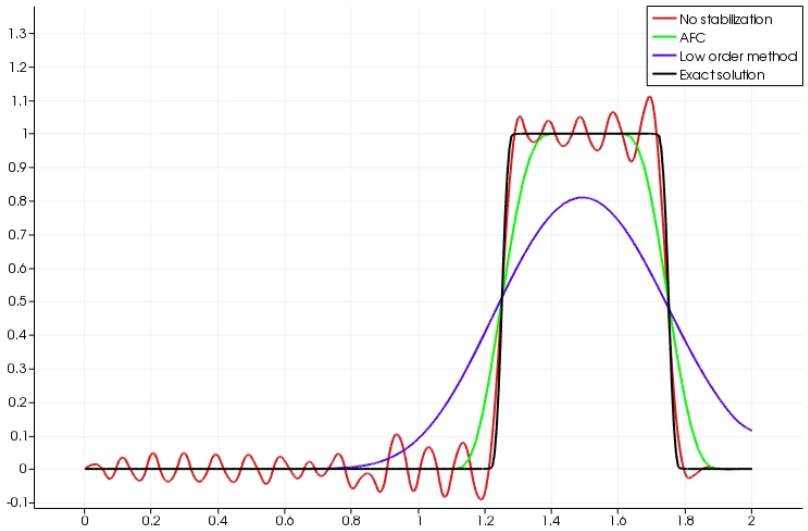


Figure 6.6: Exact and numerical solutions at time $t = 0.1$ to the problem of convection of a rectangular wave solved on a uniform $p = 2$, 66×66 B-spline basis with SSP-eRK-3 time-discretization, with time step $\Delta t = 0.0001$, with use of high-order method (no stabilization), low-order method and AFC stabilization. Section along x direction at $y = 0.5$.

Fig. 6.7 presents the L2 errors of the numerical solutions obtained using IGA with AFC stabilization with use of different time discretization methods versus the time step size. It can be easily seen that all the methods converge much faster than the unstabilized IGA scheme applied to the smooth problem (see for comparison Fig. 6.3). The most significant improvement takes place for the Forward Euler method. It now performs as good as the higher-order SSP-eRK methods. This happens because the AFC stabilization damps the oscillations which are created by the Forward Euler method for large time steps (oscillations are visible in Fig. 6.1a) and, therefore, improves the overall quality of results produced by the method. This gives a very interesting conclusion - there is no significant gain in using higher-order time-discretization schemes together with AFC for this problem. This allows us to use lower-order methods (like the forward Euler method) to save some computational effort on time discretization without loosing the accuracy of the solution.

This is not the only surprising behaviour illustrated in this figure. In some interval of time steps the forward Euler method produces lower error values than SSP-eRK methods and also smaller errors than the optimal value that we expect (value of the plateau). A possible explanation is as follows. Fig. 6.8 presents the numerical solutions obtained with the low-order method of the considered problem. It was solved with the Forward Euler method and SSP-eRK-3 method with two time step sizes $\Delta t = 0.0001$ and $\Delta t = 0.0029$. For $\Delta t = 0.0001$ solutions from both methods almost overlap but for the much larger time step $\Delta t = 0.0029$ the one from the Forward Euler method is much less diffusive. It is due to the fact that the forward Euler method is not accurate for such a large time step. Surprisingly this behaviour, though unwanted in most cases, even improves the quality of the final solution obtained after adding the limited fluxes. This is due to the fact that in general we are interested in having as few artificial diffusion as possible in the low-order solution (but it has to be non-oscillatory still). Numerical anti-diffusion resulting from inaccuracy of the Forward Euler

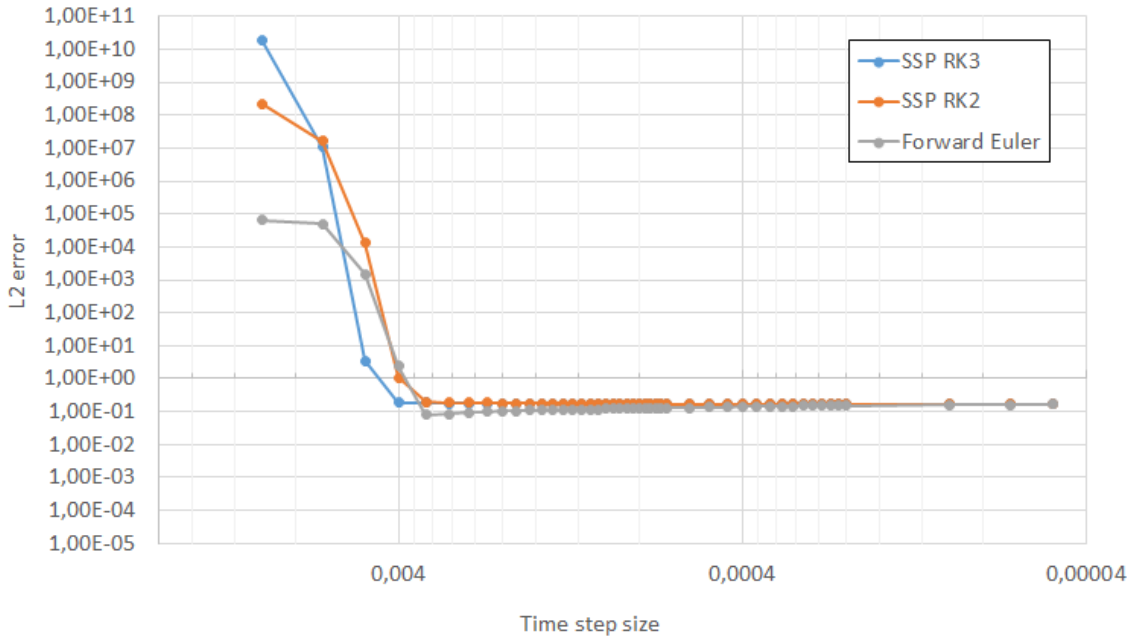


Figure 6.7: L2 errors of numerical solutions at $t = 0.1$ to the problem of convection of a rectangular wave solved on uniform $p = 2$, 66×66 B-spline basis with different time discretization methods versus the time step size.

method removes some of the artificial diffusion added but fortunately the scheme remains still non-oscillatory. One could argue that the anti-diffusion that is accepted by the limiter should improve both low-order solutions (less and more diffusive) to the same final solution. That could be the case if we used the exact values of anti-diffusive fluxes. Here, however, we use the approximation of the time derivative $\dot{\mathbf{u}}^L$ in the computation of fluxes (see (6.49)) and therefore the ability to reconstruct the high-order solution is limited. We will analyse this behaviour in more detailed manner later in this chapter. It is important to note that this behaviour is not a general characteristic but just a special case appearing for this particular problem at hand.

6.4.2 Rotation of three solid bodies

The next problem to is a common benchmark for pure convection problems, the so-called solid body rotation benchmark described for example in [12, 42]. This benchmark problem is defined as:

$$\frac{\partial u(\mathbf{x},t)}{\partial t} + \nabla \cdot (\mathbf{v}(\mathbf{x})u(\mathbf{x},t)) = 0 \quad \text{in } [0,1] \times [0,1] \tag{6.52}$$

$$\mathbf{v}(\mathbf{x}) = [0.5 - y, x - 0.5]^T \tag{6.53}$$

with homogeneous Dirichlet boundary condition at all inflow boundaries and no boundary condition at all outflow boundaries (to assure well-posedness of the hyperbolic problem), i.e.:

$$u(\mathbf{x},t) = 0 \ \text{ on } \Gamma_{in} \tag{6.54}$$

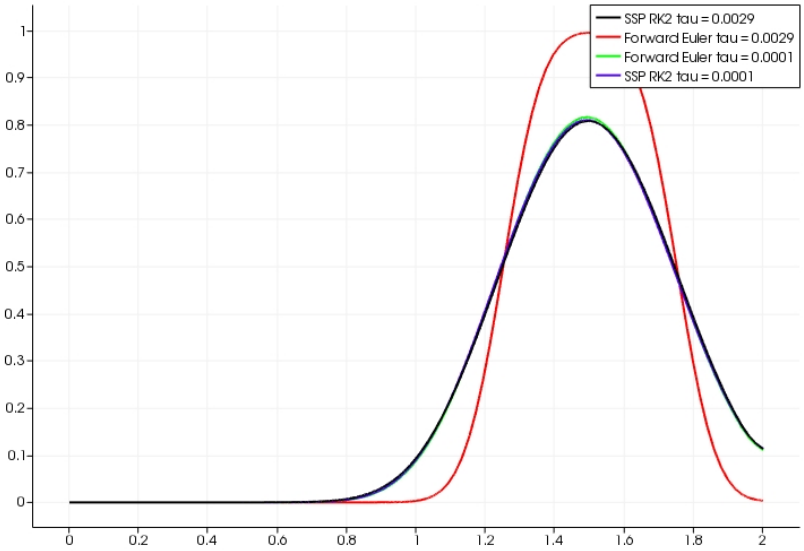


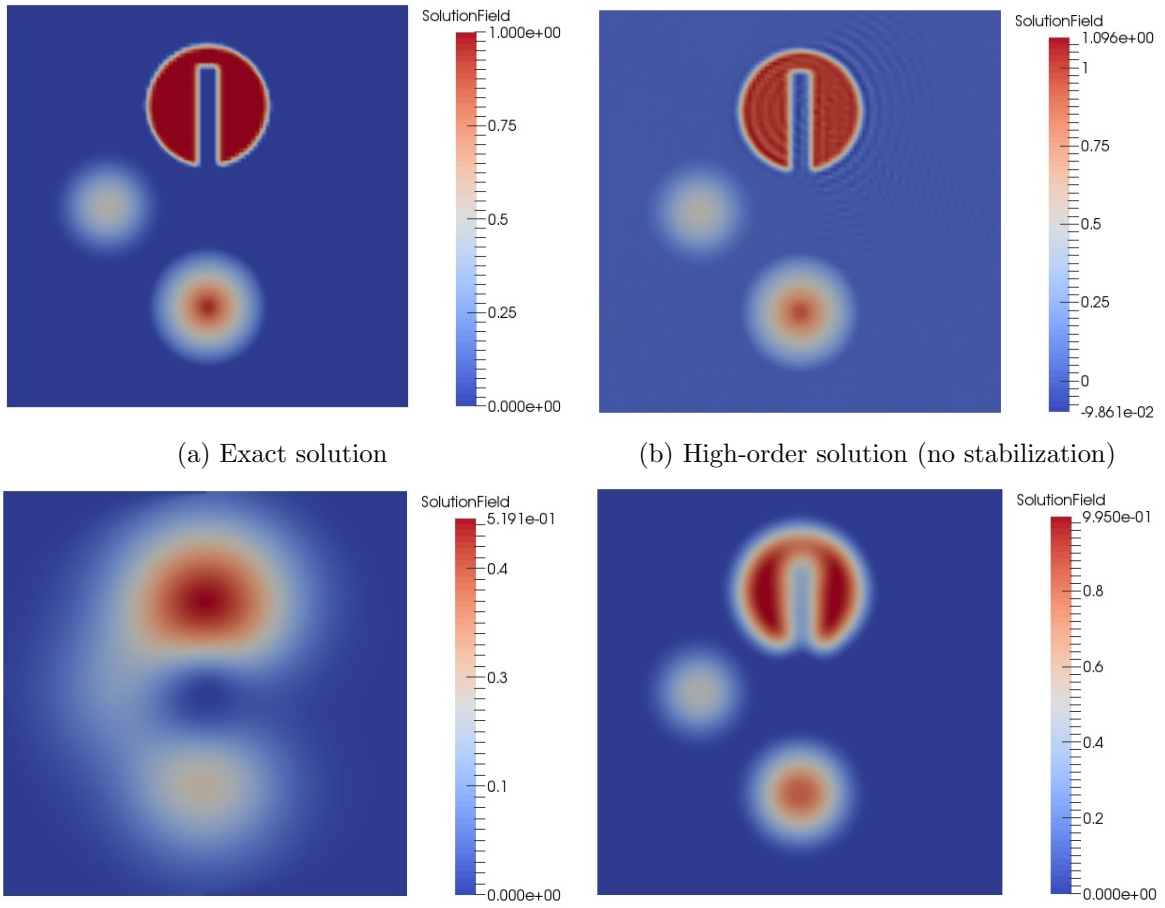
Figure 6.8: Numerical solutions at $t = 0.1$ of the problem of convection of a rectangular wave solved with the low-order scheme (discrete upwinding) on uniform $p = 2$, 66×66 B-spline basis with Forward Euler and SSP-eRK-3 time discretization schemes, with time step sizes $\Delta t = 0.0001$ and $\Delta t = 0.0029$. Section along x direction at $y = 0.5$.

The inflow boundaries are defined as those for which the velocity vector $\mathbf{v}(\mathbf{x})$ at the boundary is directed inward the domain while outflow boundaries are those for which it is directed outward. In the considered case, the inflow boundaries are: lower half of west boundary, upper half of east boundary, right half of south boundary and left half of north boundary. The rest of boundary is considered the outflow boundary.

The initial condition is the function which was already investigated in Section 4.3 as part of the analysis of the constrained L2 projection. It is defined by formulas (4.21)-(4.25). The numerical representation of the initial condition is obtained using the constrained L2 projection to avoid non-physical under- and over-shoots. The initial condition is a compositions of three functions, each describing one shape. Each of them has a significantly different nature and, therefore, the benchmark investigates the behaviour of the applied numerical scheme to problems of different character.

We are interested in the solution of this problem at time $t = 2\pi$ which corresponds to one complete revolution of the initial profile. Thus, the exact solution at this time coincides with the initial condition.

Fig. 6.9a presents the exact solution of the benchmark problem described above. The non-stabilized (high-order) numerical solution to this problem is presented in Fig. 6.9b. Although the shapes are represented correctly, there are oscillations caused by the discontinuity and propagated over the domain. Fig. 6.9c presents the low-order solution to the problem which, as expected, is very diffusive. Finally, Fig. 6.9d illustrates the result obtained with AFC stabilization. It can be easily seen that the solution, as expected, is free of non-physical oscillations. Unfortunately, it is very diffusive as well. All numerical computations were performed on the space spanned by uniform $p = 2$, 130×130 B-spline basis. The temporal discretization was done with SSP-eRK-3 time discretization method, with time step size $\Delta t = 0.006$.



(a) Exact solution

(b) High-order solution (no stabilization)

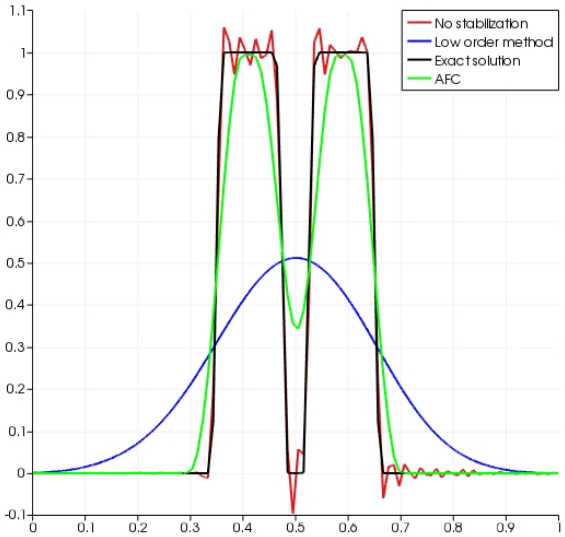
(c) Low-order solution (discrete upwinding)

(d) AFC stabilization

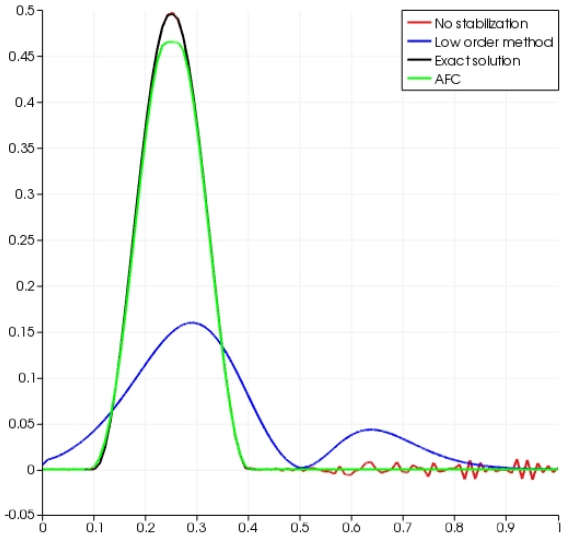
Figure 6.9: Exact and numerical solutions at $t = 2\pi$ to the solid body rotation benchmark solved on uniform $p = 2$, 130×130 B-spline basis with SSP-eRK-3 time discretization method, with time step $\Delta t = 0.006$, with use of high-order method (no stabilization), low-order method and AFC stabilization.

Fig. 6.10 contains sections through each of the shapes involved in the benchmark problem. The computations were performed in the same configuration of spatial and temporal discretizations as in previously discussed figure. The low- and high-order methods as it was visible in the previous figure have respectively diffusive and oscillatory natures. For the slotted cylinder shown in Fig. 6.10a, the solution with AFC stabilization is representing the discontinuities in slightly smeared manner to avoid oscillations. Unfortunately, for the smooth profiles, the hump and cone presented in Fig. 6.10b and Fig. 6.10c, respectively, the solution looses the precision of representation in the area near the extremum. This behaviour is a well known drawback of FCT-type limiters and referred to as peak-clipping phenomenon in the literature [42].

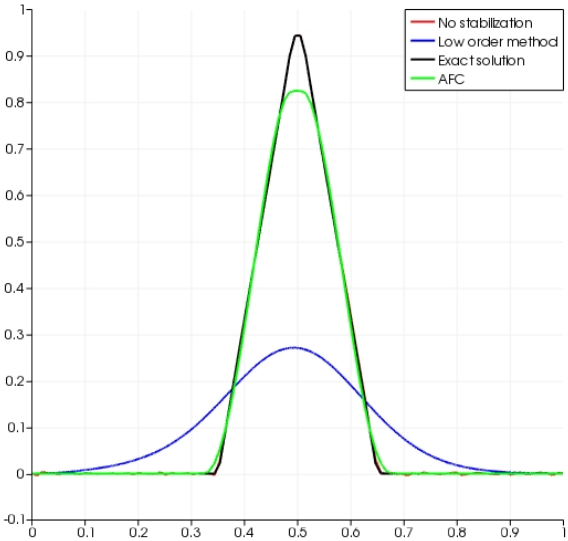
This unwanted overly diffusive behaviour of the scheme stabilized with AFC can be partially explained by analysis of the approximation made during evaluation of the anti-diffusive fluxes. As mentioned before, ideally, the raw anti-diffusive fluxes added to the low-order solution should restore the solution given by the original scheme (high-order solution). In practice



(a) Section along x direction at $y = 0.75$



(b) Section along x direction at $y = 0.5$



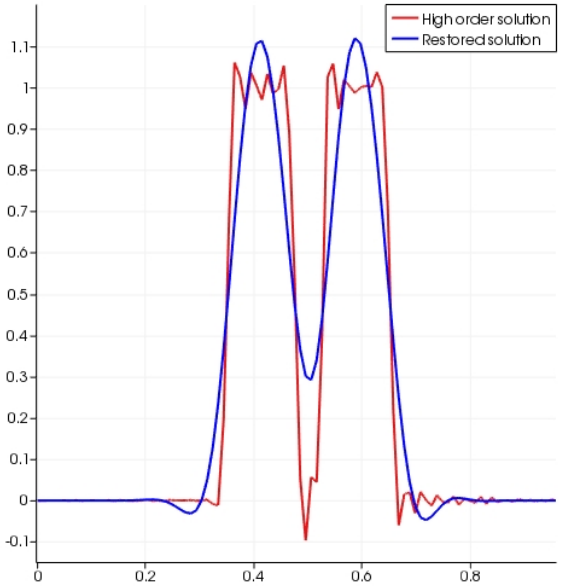
(c) Section along x direction at $y = 0.25$

Figure 6.10: Exact and numerical solutions at $t = 2\pi$ to the the solid body rotation benchmark problem solved on uniform $p = 2$, 130×130 B-spline basis with SSP-eRK-3 time discretization method, with time step $\Delta t = 0.006$, with use of high-order method (no stabilization), low-order method and AFC stabilization. Sections along x direction at $y = 0.75$, $y = 0.5$ and $y = 0.25$.

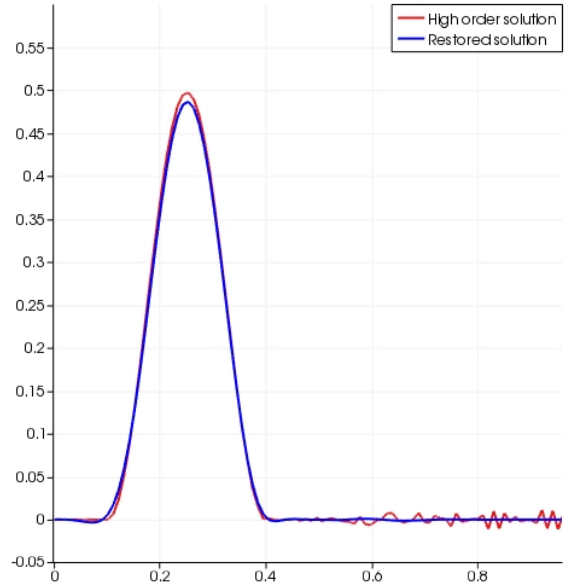
this is not the case due to the error made during the approximation of the time derivative $\dot{\mathbf{u}}^L$. Let us now analyse how large this error is. Fig. 6.11 presents the same three cross-sections as Fig. 6.10. Here, we compare the original high-order solution with the high-order solution restored from the low-order one by adding unlimited anti-diffusive fluxes. The low-order approximation of the time derivative $\dot{\mathbf{u}}^L$ used in the latter case is not very accurate so that both high-order approximations do not coincide. It does not represent the oscillations and discontinuities in a sharp way. Even worse is the fact that the original values of the extrema of the hump and cone are lost.

The easiest way to avoid this and significantly improve the quality of the profile is to use another, more accurate approximation for the time derivative $\dot{\mathbf{u}}^L$. Unfortunately, for this test problem none of the other approximations proposed in Section 6.3.3 gave satisfactory results. Another way to improve the quality of the results is to use either the non-linear AFC scheme instead of the linearised one or to develop better linearisation techniques which, however, goes beyond the scope of this thesis work.

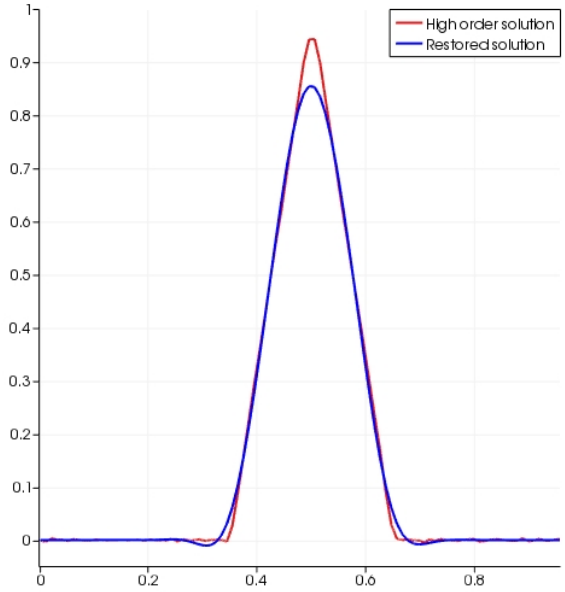
It was shown in this chapter that isogeometric analysis is a suitable tool to solve the time-dependent convection-diffusion equation. Similarly to the standard FEM the method performs well for very smooth problems. However, purely convective problems with shapes involving discontinuities require some sort of stabilization. The method of choice in the thesis is Algebraic Flux Correction. For the time-dependent problems it was used with FCT-type limiting. The stabilization gives results free of non-physical oscillations but overly diffusive. Further developments can be made to improve the quality of the numerical solutions: use of non-linear FCT, other linearisation techniques or different approximations for the time derivative for the evaluation of anti-diffusive fluxes. Those improvements are beyond the scope of this thesis.



(a) Section along x direction at $y = 0.75$



(b) Section along x direction at $y = 0.5$



(c) Section along x direction at $y = 0.25$

Figure 6.11: Numerical solutions at $t = 2\pi$ to the the solid body rotation benchmark problem solved on uniform $p = 2$, 130×130 B-spline basis with SSP-eRK-3 time discretization method, with time step $\Delta t = 0.006$, with use of high-order method (no stabilization) and high-order method restored from low-order method. Sections along x direction at $y = 0.75$, $y = 0.5$ and $y = 0.25$.

Chapter 7

Compressible Euler equations

This chapter is devoted to the application of isogeometric analysis to the compressible Euler equations. Firstly, the compressible Euler equations are defined and a short description of their properties is provided. The second section is devoted to the discretization of the Euler equations using the standard Galerkin/IGA methodology. In the third part of this chapter the boundary conditions for the Euler equations are described together with details of how to implement them. The fourth section presents numerical results for selected benchmark computations. Some of the presented results reveal the existence of instabilities and, therefore, a stabilization of AFC-type to the systems of equations is discussed afterwards. The last part of the chapter presents numerical results obtained using the stabilized scheme.

7.1 Compressible Euler equations

The **compressible Euler equations** describe the behaviour of an ideal inviscid compressible fluid. This class of flows consist of gas flows in regimes in which neglecting the viscosity still assures acceptable quality of results. The compressible Euler equations are a system of conservation laws for the mass, momentum and energy [45]:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0 \tag{7.1}$$

$$\frac{\partial (\rho \mathbf{v})}{\partial t} + \nabla \cdot (\rho \mathbf{v} \otimes \mathbf{v} + p\mathscr{I}) = 0 \tag{7.2}$$

$$\frac{\partial (\rho E)}{\partial t} + \nabla \cdot (\rho E \mathbf{v} + p\mathbf{v}) = 0 \tag{7.3}$$

with ρ denoting the density, t the time, $\mathbf{v}$ the velocity vector, E the total energy, $\mathscr{I}$ the identity tensor and p the pressure given by the **equation of state** (for ideal polytropic gas) [33]:

$$p = (\gamma - 1)\left(\rho E - \frac{\rho |\mathbf{v}|^2}{2}\right) \tag{7.4}$$

Here, γ denotes the **heat capacity ratio** (for air $\gamma = 1.4$). The system of equations (7.1)-(7.3) can be rewritten in divergence form [33]:

$$\frac{\partial U}{\partial t} + \nabla \cdot \mathbf{F} = 0 \tag{7.5}$$

with the vector of **conservative variables** U and the vector of **inviscid fluxes F** defined as:

$$U = \begin{bmatrix} \rho \\ \rho \mathbf{v} \\ \rho E \end{bmatrix} \tag{7.6}$$

$$\mathbf{F} = \begin{bmatrix} \rho \mathbf{v} \\ \rho \mathbf{v} \otimes \mathbf{v} + p\mathscr{I} \\ \rho E \mathbf{v} + p\mathbf{v} \end{bmatrix} \tag{7.7}$$

The Euler equations are completed with initial conditions (given distribution of $U(\mathbf{x}, t)$ in the bounded domain Ω at time $t = 0$) [33]:

$$U(\mathbf{x}, 0) = U_0(\mathbf{x}) \text{ in } \Omega \tag{7.8}$$

and consistent boundary conditions. The boundary conditions for the Euler equation are usually imposed as solution of a Riemann problem for the interior state U and the exterior state U_∞ [33]. A more detailed description of the theory of boundary conditions for the Euler equations and their implementation is provided in section 7.3. In this thesis we consider BCs of the following type: Dirichlet boundary condition on the boundary Γ_D [33]:

$$U = G(U, U_\infty) \text{ on } \Gamma_D \tag{7.9}$$

and Neumann boundary condition on the boundary Γ_N:

$$\mathbf{n} \cdot \mathbf{F} = F_n(U, U_\infty) \text{ on } \Gamma_N \tag{7.10}$$

where $\mathbf{n}$ is the outward unit normal vector.

7.2 Discretization of the Euler equations

This section describes the discretization of the Euler equations. The spatial discretization is done using the group FEM approach in the framework of IGA. The weak formulation of the system (7.5) reads [33]:

$$\begin{gathered} \text{find } U \in S \text{ such that for all } v \in V, \\ \int_\Omega \left(v \frac{\partial U}{\partial t} - \nabla v \cdot \mathbf{F} \right) d\mathbf{x} + \int_\Gamma v F_n ds = 0 \end{gathered} \tag{7.11}$$

Test functions v (because of restrictions on the test space) vanish on the Dirichlet boundary Γ_D and, therefore, we only need to consider the integration over Γ_N instead of the whole boundary Γ [33].

Fletcher's group formulation approximates the solution U by U^h and the vector of inviscid fluxes $\mathbf{F}$ by $\mathbf{F}^h$ using the same function space [41]. For IGA it is a space spanned by B-spline (NURBS) basis functions, i.e. [33]:

$$U^h(\mathbf{x}, t) = \sum_j U_j(t) \varphi_j(\mathbf{x}) \tag{7.12}$$

$$\mathbf{F}^h(\mathbf{x}, t) = \sum_j \mathbf{F}_j(t) \varphi_j(\mathbf{x}) \tag{7.13}$$

Inserting the above expansion into the variational form (7.11) yields the system of semi-discretized equations for the time-dependent unknowns U_j [33]:

$$\sum_j \left(\int_\Omega \varphi_i \varphi_j d\mathbf{x} \right) \frac{dU_j}{dt} = \sum_j \left(\int_\Omega \nabla \varphi_i \varphi_j d\mathbf{x} \right) \cdot \mathbf{F}_j - \int_{\Gamma_N} \varphi_i F_n ds \tag{7.14}$$

According to [59] the Euler fluxes evince the homogeneity property:

$$\mathbf{F} = \mathbf{A}U \tag{7.15}$$

where $\mathbf{A} = \frac{\partial \mathbf{F}}{\partial U}$ is the Jacobian tensor associated with the quasi linear form of (7.5), i.e. [33]:

$$\frac{\partial U}{\partial t} + \mathbf{A} \cdot \nabla U = 0 \tag{7.16}$$

Due to this amenable property we obtain:

$$\mathbf{F}_j = \mathbf{A}_j U_j \tag{7.17}$$

Thus, we can write the semi-discrete problem (7.14) in matrix form [33]:

$$M_C \frac{dU}{dt} = KU + S(U) \tag{7.18}$$

with M_C being the **block consistent mass matrix**, K the **discrete Jacobian operator** and S the **boundary load vector**. All those operators are defined below. For systems the ordering of unknowns is very important. The approach used in this thesis is to order them in the following way:

$$U = \begin{bmatrix} \rho_1 & (\rho \mathbf{v})_1 & (\rho E)_1 & \cdots & \rho_N & (\rho \mathbf{v})_N & (\rho E)_N \end{bmatrix}^T \tag{7.19}$$

For such an ordering, the consistent mass matrix M_C is a block matrix consisting of blocks M_{ij} of size $(n+2) \times (n+2)$ (for problem defined in $\mathbb{R}^n$), where $M_{ij} = m_{ij} I$ with I being the identity matrix and [33]:

$$m_{ij} = \int_\Omega \varphi_i \varphi_j d\mathbf{x} \tag{7.20}$$

The discrete Jacobian operator K is defined as a block matrix consisting of blocks K_{ij} of size $(n+2) \times (n+2)$ given by [33]:

$$K_{ij} = \mathbf{c}_{ji} \cdot \mathbf{A}_j \tag{7.21}$$

with:

$$\mathbf{c}_{ij} = \int_\Omega \varphi_i \nabla \varphi_j d\mathbf{x} \tag{7.22}$$

The Jacobian matrix for the 2D case is given by $\mathbf{A} = (A^1, A^2)$, where [60, 61]:

$$A^1 = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \gamma_2 v_1^2 + \gamma_1 v_2^2 & (3-\gamma) v_1 & (1-\gamma) v_2 & \gamma - 1 \\ v_1 v_2 & v_2 & v_1 & 0 \\ \gamma_1 (v_1^3 + v_2^2 v_1) - H v_1 & H - (\gamma - 1) v_1^2 & (1-\gamma) v_1 v_2 & \gamma v_1 \end{bmatrix} \tag{7.23}$$

$$A^2 = \begin{bmatrix} 0 & 0 & 1 & 0 \\ v_1 v_2 & v_2 & v_1 & 0 \\ \gamma_2 v_2^2 + \gamma_1 v_1^2 & (1-\gamma)v_1 & (3-\gamma)v_2 & \gamma - 1 \\ \gamma_1(v_2^3 + v_1^2 v_2) - H v_2 & (1-\gamma)v_1 v_2 & H - (\gamma-1)v_2^2 & \gamma v_2 \end{bmatrix} \tag{7.24}$$

where γ_1 and γ_2 are the auxiliary quantities given by:

$$\gamma_1 = \frac{\gamma - 1}{2}, \quad \gamma_2 = \frac{\gamma - 3}{2} \tag{7.25}$$

and H is the **total enthalpy** defined as:

$$H = E + \frac{p}{\rho} \tag{7.26}$$

The entries of the boundary load vector S are defined as [33]:

$$S_i = -\int_{\Gamma_N} \varphi_i F_n ds \tag{7.27}$$

Temporal discretization of the semi-discrete problem (7.18) can be performed for example using the Forward Euler method or Strong Stability Preserving Runge-Kutta methods of orders 2 and 3, respectively. Those methods were introduced and described in Section 6.1.2.

7.3 Boundary conditions

The Euler equations form a hyperbolic system. Solutions to such a system are superpositions of several waves travelling in certain directions with certain speeds. A consistent choice of boundary conditions depends on the way the waves propagate in the considered region [33]. This section presents briefly the underlying theory and the practical implementation of the boundary conditions for the Euler equations.

7.3.1 Physical boundary conditions

The number of **physical boundary conditions (PBC)** that have to be prescribed is obtained using a transformation to the local characteristic variables along the unit outward normal vector $\mathbf{n}$. The description of the transformation can be found in [45]. The final result is a set of $n+2$ decoupled equations [33]:

$$\frac{\partial w_k}{\partial t} + \lambda_k \frac{\partial w_k}{\partial \mathbf{n}} = 0 \quad \text{for } k = 1, ..., n+2 \tag{7.28}$$

where w_k are the **Riemann invariants** and λ_k are the eigenvalues of the directional Jacobian $\mathbf{n} \cdot \mathbf{A}$. With $\Lambda = \text{diag}\{\lambda_1, ..., \lambda_{n+2}\}$ and $W = [w_1, ..., w_{n+2}]^T$ we can write the system in matrix form [33]:

$$\frac{\partial W}{\partial t} + \Lambda \frac{\partial W}{\partial \mathbf{n}} = 0 \tag{7.29}$$

For the 2D case W and Λ are given by [33, 61]:

$$W = \left[v_n - \tfrac{2c}{\gamma - 1}, \quad \tfrac{p}{\rho^\gamma}, \quad v_\xi, \quad v_n + \tfrac{2c}{\gamma - 1} \right]^T \tag{7.30}$$

$$\Lambda = \mathrm{diag}\{v_n - c,\ v_n,\ v_n,\ v_n + c\} \tag{7.31}$$

where v_n is the normal component of the velocity, v_ξ the tangential component of the velocity and c is the speed of sound given by [61]:

$$c = \sqrt{\frac{\gamma p}{\rho}} \tag{7.32}$$

It is easy to see from the system (7.28) that the characteristic variable w_k remains constant along the characteristic defined by [45]:

$$\lambda_k = \frac{d\mathbf{n}}{dt} \tag{7.33}$$

The evolution of Riemann invariants has purely convective behaviour, therefore a boundary condition is required for each incoming wave - a Riemann variable for which the characteristic has negative slope. In other words, the number of PBC to be imposed is equal to the number of negative eigenvalues N_λ [33].

It is worthwhile to define here the local **Mach number** [33]:

$$M = \frac{|v_n|}{c} \tag{7.34}$$

The number of the negative eigenvalues N_λ depend on the sign of v_n and the magnitude of M. Five most common types of boundaries with the corresponding values of v_n and M and the required number of PBC in 2D are presented in the table below [33].

Type of boundary	v_n	M	N_λ
Supersonic inlet	< 0	> 1	4
Supersonic outlet	> 0	> 1	0
Subsonic inlet	< 0	< 1	3
Subsonic outlet	> 0	< 1	1
Solid wall	0	0	1

7.3.2 Implementation of boundary conditions

Whenever $0 < N_\lambda < 4$ the physical boundary conditions are not providing enough information to directly impose the boundary conditions (Dirichlet boundary values or Neumann fluxes). In such a case there is a need for numerical boundary conditions (NBC). The missing information is obtained by the procedure presented below. The Dirichlet boundary value $G(U, U_\infty)$ as well as the Neumann normal fluxes $F_n(U, U_\infty)$ are defined as the solution to the **boundary Riemann problem** associated with the internal state U and the external state U_∞. The internal state U is defined as the numerical solution to the Euler equations while the external state U_∞ has to be computed based on the internal state and the physical boundary condition prescribed on this boundary. It can be obtained using the following procedure [33, 59, 62]:

1. Compute the Riemann invariants W corresponding to the numerical solution U
2. Overwrite the incoming Riemann invariants in W by prescribed PBC values to obtain W_∞

3. Convert the modified Riemann invariants W_∞ back to conservative variables to obtain the external state U_∞

In this thesis we only consider Neumann boundary conditions. Although there exists an exact Riemann solver proposed by Toro [60], we use the **approximate solver by Roe** [63] (to avoid excessive computational costs) to solve the Riemann problem for the Neumann normal fluxes $F_n(U, U_\infty)$. The idea behind Roe's approximate Riemann solver is to find a **Roe matrix** $\mathbf{A}(U_1, U_2)$ constant between the states U_1 and U_2 and approximate with it the Jacobian matrix $\mathbf{A}(U)$ depending on the intermediate states. The Roe matrix $\mathbf{A}(U_1, U_2)$ has to fulfil the following conditions [60]:

1. $\mathbf{F}(U_1) - \mathbf{F}(U_2) = \mathbf{A}(U_1, U_2)(U_1 - U_2)$
2. $\mathbf{A}(U_1, U_2) \to A(U)$ as $U_1 \to U$ and $U_2 \to U$
3. $\mathbf{A}(U_1, U_2)$ has only real eigenvalues and a complete system of eigenvectors

Using this approximation we can solve a fully linear problem instead of the quasi-linear one. In this thesis we will compute the Roe matrix by evaluation of the Jacobian matrix $\mathbf{A}(U)$ for the density averaged Roe mean values defined later in this section. For the problem at hand Roe's approximate Riemann solver yields [33]:

$$F_n(U, U_\infty) = \mathbf{n} \cdot \frac{\mathbf{F}(U) + \mathbf{F}(U_\infty)}{2} - \frac{1}{2}|\mathbf{n} \cdot \mathbf{A}(U, U_\infty)|(U_\infty - U) \tag{7.35}$$

with $\mathbf{A}(U, U_\infty)$ being the Roe matrix. According to [61] the following factorization is possible:

$$|\mathbf{n} \cdot \mathbf{A}(U, U_\infty)| = \|\mathbf{n}\| R_n |\Lambda_n| L_n \tag{7.36}$$

where the columns of matrix R_n are given by the set of right eigenvectors [61]:

$$R_n = \begin{bmatrix} 1 & 1 & 1 & 0 \\ v_1 - cn_1 & v_1 & v_1 + cn_1 & n_2 \\ v_2 - cn_2 & v_2 & v_2 + cn_2 & -n_1 \\ H - cv_n & q & H + cv_n & v_1 n_2 - v_2 n_1 \end{bmatrix} = \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{r}_3 & \mathbf{r}_4 \end{bmatrix} \tag{7.37}$$

$$q = \frac{|\mathbf{v}|}{2} \tag{7.38}$$

and L_n is the matrix of left eigenvectors [61]:

$$L_n = \begin{bmatrix} \frac{1}{2}(b_1 + \frac{v_n}{c}) & \frac{1}{2}(-b_2 v_1 - \frac{n_1}{c}) & \frac{1}{2}(-b_2 v_2 - \frac{n_2}{c}) & \frac{1}{2} b_2 \\ 1 - b_1 & b_2 v_1 & b_2 v_2 & -b_2 \\ \frac{1}{2}(b_1 - \frac{v_n}{c}) & \frac{1}{2}(-b_2 v_1 + \frac{n_1}{c}) & \frac{1}{2}(-b_2 v_2 + \frac{n_2}{c}) & \frac{1}{2} b_2 \\ \frac{v_2 - v_n n_2}{n_1} & n_2 & \frac{n_2^2 - 1}{n_1} & 0 \end{bmatrix} = \begin{bmatrix} \mathbf{l}_1 \\ \mathbf{l}_2 \\ \mathbf{l}_3 \\ \mathbf{l}_4 \end{bmatrix} \tag{7.39}$$

$$b_1 = b_2 q, \qquad b_2 = \frac{\gamma - 1}{c^2} \tag{7.40}$$

Λ_n is the diagonal matrix of eigenvalues [61]:

$$\Lambda_n = \mathrm{diag}\{v_n - c,\ v_n,\ v_n + c,\ v_n\} \tag{7.41}$$

All three matrices are evaluated for the **density averaged Roe mean values** [61]:

$$\tilde{\rho}_{ij} = \sqrt{\rho_i \rho_j} \tag{7.42}$$

$$\tilde{\mathbf{v}}_{ij} = \frac{\sqrt{\rho_i}\mathbf{v}_i + \sqrt{\rho_j}\mathbf{v}_j}{\sqrt{\rho_i} + \sqrt{\rho_j}} \tag{7.43}$$

$$\tilde{H}_{ij} = \frac{\sqrt{\rho_i}H_i + \sqrt{\rho_j}H_j}{\sqrt{\rho_i} + \sqrt{\rho_j}} \tag{7.44}$$

$$\tilde{c}_{ij} = \sqrt{(\gamma - 1)\left(\tilde{H}_{ij} - \frac{|\tilde{\mathbf{v}}_{ij}^2|}{2}\right)} \tag{7.45}$$

Here, index i corresponds to the value of the internal state and j to the value of the external state. Let us have a closer look on how to obtain a suitable external state U_∞ for a given PBC. The conversion from conservative variables to Riemann invariants is done according to definition (7.30). The inverse operation can be done using the formulas below [33, 61, 59]:

$$\rho = \left(\frac{c^2}{\gamma w_2}\right)^{\frac{1}{\gamma - 1}} \tag{7.46}$$

$$\rho\mathbf{v} = \rho(v_n\mathbf{n} + v_\xi\tau_\xi) \tag{7.47}$$

$$\rho E = \frac{p}{\gamma - 1} + \frac{\rho}{2}(v_n^2 + v_\xi^2) \tag{7.48}$$

where τ_ξ is the vector tangential to the boundary, and:

$$v_n = \frac{w_4 + w_1}{2}, \qquad v_\xi = w_3 \tag{7.49}$$

$$c = \frac{\gamma - 1}{4}(w_4 - w_1), \qquad p = \frac{\rho c^2}{\gamma} \tag{7.50}$$

The physical boundary conditions are usually prescribed in terms of primitive or conservative variables (sometimes also other quantities) rather than Riemann invariants [33]. Below we will present a practical way to implement the most common types of boundary conditions.

Supersonic inlet boundary condition

At the supersonic inlet four PBCs have to be imposed. Then obviously, all the entries of U_∞ have to be prescribed. Therefore, there is no need for conversion to Riemann invariants [61].

Supersonic outlet boundary condition

At the supersonic outlet no PBCs are imposed. Therefore, the external state U_∞ is equal to the internal state U and the Roe flux reduces to [33]:

$$F_n(U, U_\infty) = \mathbf{n} \cdot \mathbf{F}(U) \tag{7.51}$$

Subsonic inlet boundary condition

At the subsonic inlet three PBCs have to be imposed. It is common practice to prescribe the density ρ_∞, the pressure p_∞ and the tangential velocity $v_{\xi,\infty}$. Based on these quantities we need to compute the Riemann invariants except for w_4, which is calculated from the internal state. The three incoming Riemann variables are calculated according to [33, 61]:

$$w_1 = w_4 - \frac{4}{\gamma - 1}\sqrt{\frac{\gamma p_\infty}{\rho_\infty}} \tag{7.52}$$

$$w_2 = \frac{p_\infty}{\rho_\infty^\gamma} \tag{7.53}$$

$$w_3 = v_{\xi,\infty} \tag{7.54}$$

Subsonic outlet boundary condition

At the subsonic outlet one PBC has to be imposed. It is common practice to prescribe the exit pressure p_∞. All Riemann invariants except w_1 are calculated from the internal state. The first Riemann invariant w_1 is computed according to [33, 64]:

$$w_1 = w_4 - \frac{4}{\gamma - 1}\sqrt{\frac{\gamma p_\infty}{\rho_\infty}} \tag{7.55}$$

where the external density ρ_∞ is not prescribed but calculated according to the formula:

$$\rho_\infty = \rho \left(\frac{p_\infty}{p}\right)^{\frac{1}{\gamma}} \tag{7.56}$$

Wall boundary condition

At walls one PBC has to be imposed. Naturally, that is the no-penetration or free-slip condition [61]:

$$\mathbf{v} \cdot \mathbf{n} = 0 \tag{7.57}$$

In this case the external state variables U_∞ can be obtained without transformation to Riemann invariants. The density, the tangential velocity and the total energy remain unchanged while the normal velocity is calculated using the mirror condition [33]:

$$\mathbf{n} \cdot (\mathbf{v}_\infty + \mathbf{v}) = 0 \tag{7.58}$$

Thus the external state U_∞ is given by [33]:

$$U_\infty = \begin{bmatrix} \rho \\ \rho \mathbf{v}_\infty \\ \rho E \end{bmatrix}, \qquad \mathbf{v}_\infty = \mathbf{v} - 2\mathbf{n}(\mathbf{n} \cdot \mathbf{v}) \tag{7.59}$$

There exists also an alternative way to impose this boundary condition, where Neumann fluxes are defined directly as [33]:

$$F_n = \begin{bmatrix} 0 \\ \mathbf{n}p \\ 0 \end{bmatrix} \tag{7.60}$$

The advantage of this method is that there is no need to solve the Riemann problem. On the other hand, using the Roe fluxes constitutes a more physical BCs. It is important to note that both methods, due to weak imposition of the wall BC, do not guarantee a zero normal velocity on the wall [33]. This issue can be solved by adding a penalty term as proposed in [65].

The boundary integrals can be calculated using for example Gauss quadrature (in this thesis we use 14 point Gauss quadrature for each knot span). The procedure described above must be performed for each quadrature point.

7.4 Numerical results

In this section we analyse three standard benchmark problems for the Euler equations to validate the IGA-based solver. The first test case is a stationary vortex problem, which is later on placed in a convective stream. The last problem is Sod's shock tube.

7.4.1 Stationary isentropic vortex

The first problem to be analysed is the stationary isentropic vortex problem described in [66, 67, 68]. It is defined on the infinite 2D domain with uniform state $\{\rho, v_1, v_2, p\} = \{1, 0, 0, 1\}$ to which the isentropic vortex centred at $(x_0, y_0) = (0, 0)$ is added. The vortex is defined with the use of perturbations of v_1, v_2 and temperature $T = \frac{p}{\rho}$ while the entropy $S = \frac{p}{\rho^\gamma}$ is preserved. The perturbations are given by the following formulas [68]:

$$\delta v_1 = -\frac{\beta}{2\pi} e^{(1-R^2)/2}(y - y_0) \tag{7.61}$$

$$\delta v_2 = \frac{\beta}{2\pi} e^{(1-R^2)/2}(x - x_0) \tag{7.62}$$

$$\delta T = \frac{(\gamma - 1)\beta^2}{8\gamma\pi^2} e^{1-R^2} \tag{7.63}$$

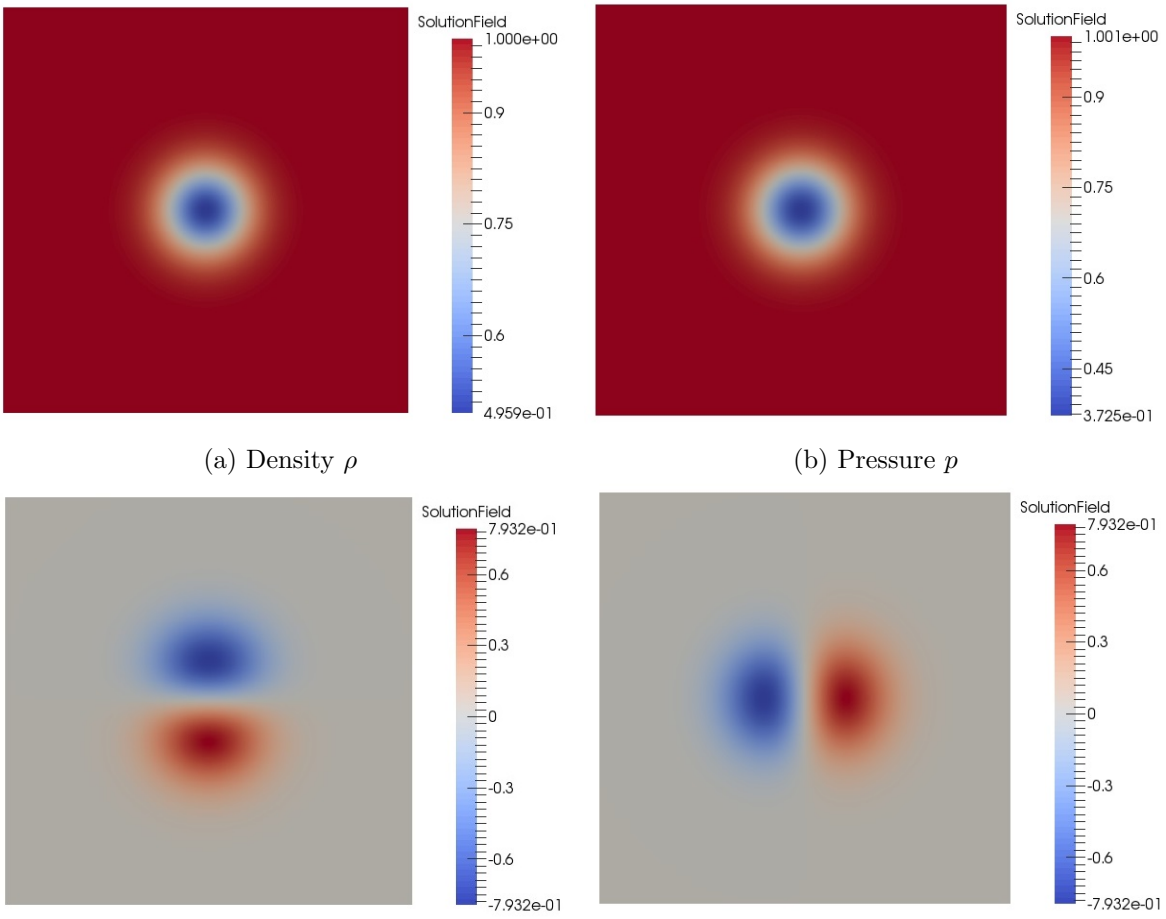
with:

$$R = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{7.64}$$

Here, β is the vortex strength, in this thesis $\beta = 5$. Note that the initial condition must be converted from the primitive variables described above to the conservative variables before solving the problem.

The problem was solved on the square computational domain $[-5, 5] \times [-5, 5]$. The solid wall boundary condition was imposed on all boundaries. Although most of the literature presents this benchmark solved on larger computational domains with periodic boundary condition it was decided to use the setting described above. It was significantly cheaper in terms of computational cost and therefore it was possible to perform the simulation for longer time. Such a setting is obviously more demanding in terms of stability and accuracy than the one with larger computational domain as all oscillations will reflect from the boundaries, overlap and easier spoil the equilibrium of the vortex.

The exact solution to this problem is presented in Fig. 7.1. For the stationary isentropic vortex problem the exact solution is not time dependent. The presented exact solution is a result of a standard L2 projection of the analytical exact solution to the space spanned by



(a) Density ρ

(b) Pressure p

(c) Horizontal component of velocity v_1

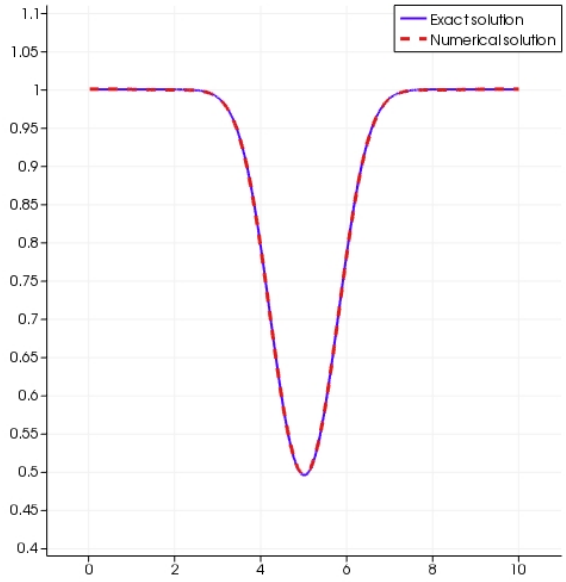
(d) Vertical component of velocity v_2

Figure 7.1: Exact solution for the stationary isentropic vortex benchmark problem.

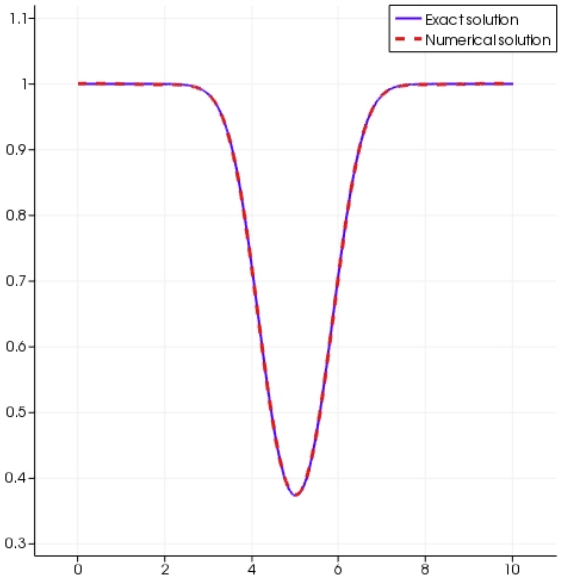
$p = 2$, 66×66 B-spline basis and therefore it is the best possible solution that can be obtained on this space in the sense of the L2 norm. In this thesis we consider solutions converted from conservative variables to primitive variables. The motivation behind it is that in engineering applications the primitive variables are of interest rather than conservative ones, as pressure, density and velocity have explicit physical meaning.

The main objective of this benchmark is to investigate how well the vortex is preserved after long simulation times. Castonguay et al. in [68] do similar tests for $t = 358$. In this thesis we are limited by the efficiency of the code and available computational power. Therefore, we preform the test for $t = 30$.

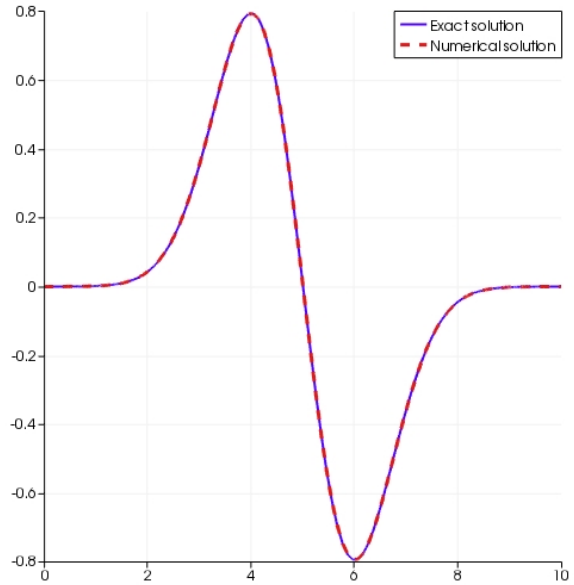
Fig. 7.2 presents the comparison of the exact solution and the numerical solution at $t = 30$. The numerical solution was computed on the space spanned by $p = 2$, 66×66 B-spline basis functions. The time discretization was done with the SSP-eRK-3 scheme with time step $\Delta t = 0.01$. The conclusion from this figure is that the vortex is perfectly preserved over time. The IGA-based solver for the compressible Euler equations passed the first test flawlessly. Let us now proceed to more complex modification of this benchmark - the convection of an isentropic vortex.



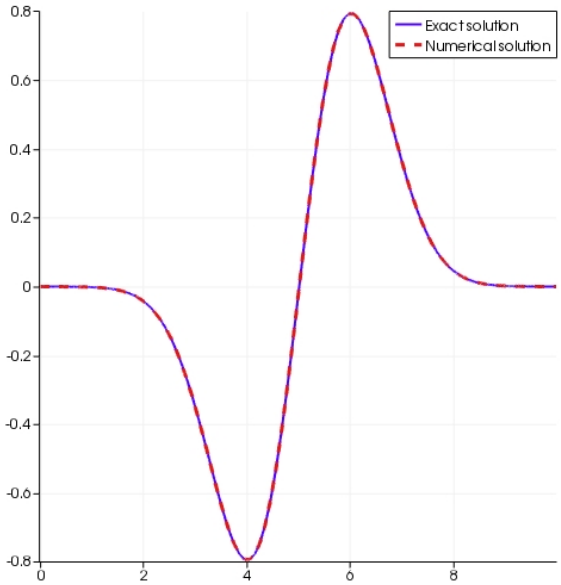
(a) Density ρ along the cut-line $y = 0.5$



(b) Pressure p along the cut-line $y = 0.5$



(c) Horizontal component of velocity v_1 along the cut-line $x = 0.5$



(d) Vertical component of velocity v_2 along the cut-line $y = 0.5$

Figure 7.2: Exact and numerical solutions to the stationary isentropic vortex benchmark problem at $t = 30$. Numerical solution obtained on $p = 2$, 66×66 B-spline basis with SSP-eRK-3 time discretization with time step $\Delta t = 0.01$.

7.4.2 Convection of an isentropic vortex

The second problem is a convection of an isentropic vortex. A similar benchmark problem was presented in [66, 67, 68]. The isentropic vortex is defined as in the previous section by (7.61) - (7.64). Now, the vortex is placed in the uniform flow $\{\rho, v_1, v_2, p\} = \{1, 5, 0, 1\}$. We expect the horizontal convection of the preserved vortex in the infinite domain. The computational domain is again a $[-5, 5] \times [-5, 5]$ box. The solid wall boundary condition is prescribed on the north boundary Γ^N and the south boundary Γ^S, while on the west boundary Γ^W and the east boundary Γ^E periodic boundary conditions are imposed.

Periodic boundary conditions were not discussed in Section 7.3.2 devoted to the implementation of boundary conditions for the Euler equations. The implementation of this boundary condition for this problem was done by imposing the supersonic outlet (the flow is supersonic even at the part of the vortex with the backward flow) at the east boundary Γ^E. On the west boundary Γ^W the supersonic inlet boundary condition was imposed with the external state being equal to the corresponding (the same value of y coordinate) internal state at Γ^E.

At $t = 4$ the vortex is back at the starting position after two cycles of convection along the whole domain. Therefore, the exact solution to this problem at $t = 4$ is similar to the one of the static case presented in Fig. 7.1. The only difference is that the values of the horizontal velocity field are now increased uniformly by 5.0 due to the fact that the vortex is now in the convective stream.

The comparison of the exact and numerical solutions at $t = 4$ looks exactly as the one for the stationary isentropic vortex presented in Fig. 7.2 and therefore was not presented here separately. The numerical solution was computed on the space spanned by $p = 2$, 66×66 B-spline basis functions. The time discretization was done with SSP-eRK-3 scheme with time step $\Delta t = 0.005$. The conclusion from this problem is that the vortex was perfectly preserved during the convective movement.

Fig. 7.3 presents the L2 errors of numerical solution for density ρ at $t = 4$ to the convection of isentropic vortex benchmark problem solved on the space spanned uniform $p = 1, 2, 3$ B-spline basis versus the number of DOFs in one direction. The time discretization was done with SSP-eRK of order 3 and the time step $\Delta t = 0.005$. Although a general practice is to keep the ratio $h/\Delta t$ constant rather than to perform the whole convergence analysis for the same time step size, in this case it was performed for the constant value of Δt which was chosen to lie on the plateau of the convergence curve of the time discretization method even for the finest spatial grid. Therefore, it was assured that any further refinement of the temporal discretization would not influence the result significantly. For all the orders the same order of convergence of 2 was achieved. This surprising, unwanted behaviour needs further investigation to reveal the cause of this limitation. There are several clues what could be the reason. First of them is that the convergence is limited by the implementation of the periodic boundary conditions (with the standard approach the continuity is decreased to C^0 at the boundary). This trace was rejected by analysis of different configurations of the domain, not involving periodicity of BCs and shorter simulation. This change did not improve the results. Other traces which were not investigated in the scope of this thesis are that the limitation can be caused by Fletcher's group formulation, the way of computing the Neumann fluxes, the weak imposition of Neumann boundary conditions or the time discretization procedure. Finding a cure for the limitation of orders of convergence for higher order basis functions is an important step on the way to an effective IGA solver for compressible flow problems.

Figure 7.3: L2 errors of the numerical solution for density ρ at $t = 4$ to the convection of isentropic vortex benchmark problem solved on uniform $p = 1, 2, 3$ B-spline basis with SSP-eRK-3 time discretization method with time step $\Delta t = 0.005$ versus the number of DOFs in one direction.

7.4.3 Sod's Shock tube

The third problem analysed is Sod's shock tube problem introduced in [69]. Although in this position it is considered as 1D problem, we will solve its 2D version. The considered domain $\Omega = [0, 1] \times [0, 1]$ consists of two parts on which separate initial conditions are prescribed [33]:

$$\{\rho_L, v_{1,L}, v_{2,L}, p_L\} = \{1, 0, 0, 1\} \quad \text{on } \Omega_L = [0, 0.5] \times [0, 1] \tag{7.65}$$

$$\{\rho_R, v_{1,R}, v_{2,R}, p_R\} = \{0.125, 0, 0, 0.1\} \quad \text{on } \Omega_R = [0.5, 1] \times [0, 1] \tag{7.66}$$

Solid wall boundary conditions are prescribed at all boundaries of the domain.

This benchmark problem has a concrete physical meaning. The domain is a container with gas at rest, divided into two parts by a thin membrane. When the membrane is instantaneously removed at $t = 0$ the gas starts to flow from the high-pressure left part of the container to the low-pressure right part of the container. The resulting flow is represented by three waves travelling at different speeds [33]. The **shock discontinuity**, at which all primitive variables are discontinuous is travelling to the right into the region of unperturbed low-pressure gas. The **contact discontinuity**, at which only the density is discontinuous and velocity and pressure are constant, is following the shock discontinuity. The **rarefaction wave** is travelling to the left into the region of unperturbed high-pressure gas. At rarefaction wave, the transition of all primitive variables is smooth [33]. The initial condition was pro-

jected to the considered B-spline space using the constrained L2 projection to avoid under- and over-shoots in the vicinity of the discontinuity.

Fig. 7.4 presents the exact and the numerical solutions to this problem at $t = 0.231$. The exact solution was obtained on a grid of 130 points with the exact Riemann solver E1RPEX from the package NUMERICA [70]. The numerical solution was calculated on the space spanned by $p = 1$ 129×129 B-spline basis. The time discretization was done with the SSP-eRK-3 scheme with time step $\Delta t = 0.0005$. Although the used basis is of a very low order and both spatial and temporal discretizations are very fine, the solution is unacceptably oscillatory. Although the largest oscillations are caused by the shock discontinuity, there are also smaller oscillations following the contact discontinuity. In most of the examples in this thesis we use a basis of order $p = 2$ to distinguish IGA from the standard FEM and prove the correctness of approximations obtained with higher order basis functions. However, for this problem in case of higher order basis functions the oscillations were growing in magnitude so rapidly that the results were out of range of machine representation after a few time steps. This happened due to the rule of thumb that the lower the order of the basis the easier it is to represent the discontinuities without creating large over- and under-shoots.

From this benchmark problem we can draw the conclusion that there is a necessity of applying stabilization methods to extend the range of problems that can be solved with the IGA-based Euler solver. As in the case of the time-dependent convection-diffusion equation we use the AFC stabilization with FCT-type limiter. The next section is devoted to the generalisation of this method towards systems of conservation laws.

7.5 Algebraic Flux Correction for the systems of conservation laws

For the Euler equations we use the Algebraic Flux Correction stabilization method with FCT-type limiter. This method was described Section 6.3. Here, we present its generalization towards systems of equations. The full, detailed description of this method is presented in [33].

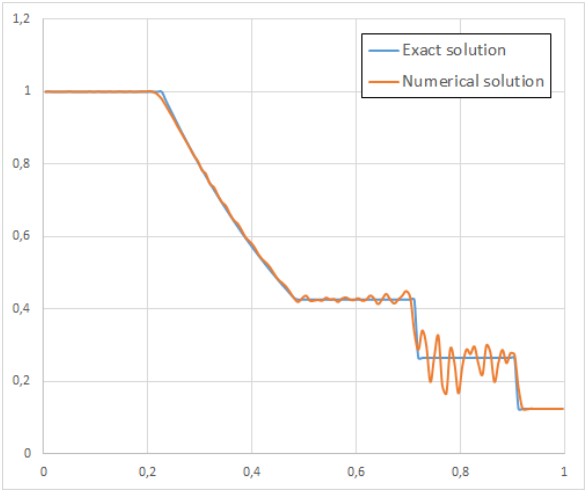
7.5.1 Derivation of the low-order method

The general idea behind the derivation of the low-order method is similar to the scalar case: perform algebraic operations on discrete operators to avoid entries that are causing the growth of instabilities (see Section 6.3.2). Similarly, as in the scalar case we perform row-sum mass lumping which is now defined as [33]:

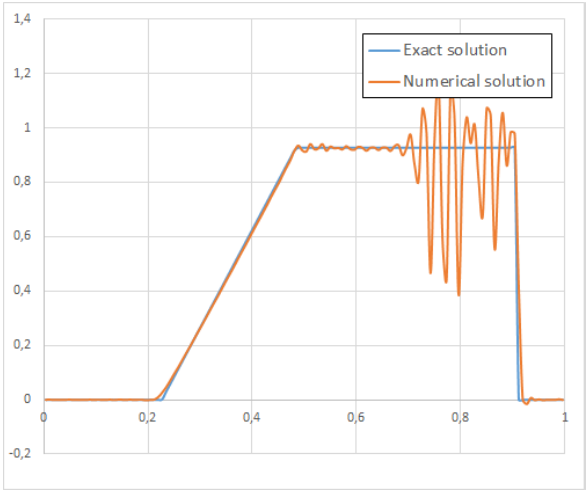
$$M_L = \text{diag}\{m_i I\}, \quad m_i = \sum_j m_{ij} \tag{7.67}$$

where I denotes the identity matrix of dimension of M_C. We also add the artificial diffusion to the discrete Jacobian operator $K = \{K_{ij}\}$. In the case of systems of conservation laws each entry K_{ij} is a block of size $(n+2) \times (n+2)$. Therefore the artificial diffusion operator $D = \{D_{ij}\}$ consists of blocks of the same size. The operator $L = \{L_{ij}\} = K + D$ can be obtained, similarly as in scalar case by edge-by-edge operations [33]:

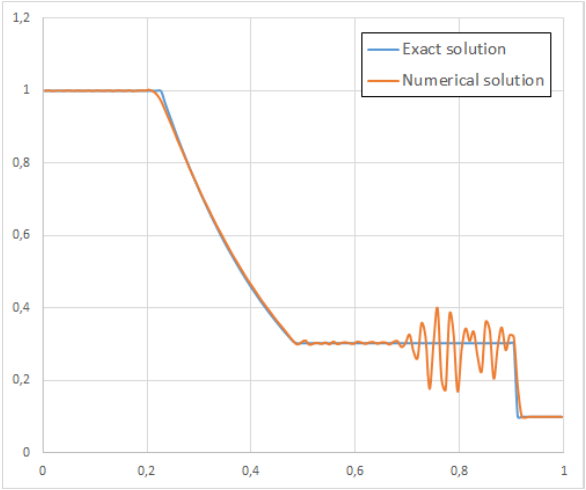
$$L_{ii} = K_{ii} - D_{ij} \tag{7.68}$$



(a) Density ρ along the cut-line $y = 0.5$



(b) Velocity v_1 along the cut-line $y = 0.5$



(c) Pressure p along the cut-line $y = 0.5$

Figure 7.4: Exact and numerical solutions to Sod's shock tube benchmark problem at $t = 0.231$. Numerical solution obtained on $p = 1$, 129×129 B-spline basis with SSP-eRK-3 time discretization with time step $\Delta t = 0.0005$.

$$L_{ij} = K_{ij} + D_{ij} \tag{7.69}$$

$$L_{ji} = K_{ji} + D_{ij} \tag{7.70}$$

$$L_{jj} = K_{jj} - D_{ij} \tag{7.71}$$

Apparently, according to [33, 13] it is not necessary for assuring the LED property generalised for systems of conservation laws (LED with respect to local characteristic variables) that all off-diagonal entries of L are non-negative (like in the scalar case). It is enough that the off-diagonal blocks are positive semi-definite. There are several ways to compute the D_{ij}, they are presented and derived in [33]. In this thesis we use scalar dissipation proportional to the largest eigenvalue which is defined as [33, 71]:

$$D_{ij} = d_{ij}I, \quad d_{ij} = |\mathbf{e}_{ij}| \max_i |\lambda_i| \tag{7.72}$$

where the spectral radius of the Roe matrix can be calculated from the formula [33]:

$$\max_i |\lambda_i| = |\mathbf{e}_{ij}|(|\tilde{v}_{ij}| + \tilde{c}_{ij}) \tag{7.73}$$

Here, $\mathbf{e}_{ij}$ is a virtual edge connecting the DOFs i and j defined as [33]:

$$\mathbf{e}_{ij} = \frac{\mathbf{c}_{ji} - \mathbf{c}_{ij}}{2} \tag{7.74}$$

$\tilde{v}_{ij}$ is a component along the edge $\mathbf{e}_{ij}$ of the Roe density averaged velocity $\tilde{\mathbf{v}}_{ij}$ computed according to (7.43) and [33]:

$$\tilde{v}_{ij} = \frac{\mathbf{e}_{ij} \cdot \tilde{\mathbf{v}}_{ij}}{|\mathbf{e}_{ij}|} \tag{7.75}$$

and $\tilde{c}_{ij}$ is a Roe density averaged speed of sound computed according to (7.45).

Application of these operations on the discrete operators yields the low-order approximation to the problem (7.18) as [33]:

$$M_L \frac{dU}{dt} = LU + S(U) \tag{7.76}$$

7.5.2 FCT-type limiter for systems of conservation laws

By the definition of the low-order and high-order semi-discrete schemes and from the definition of M_L and L, the raw anti-diffusive fluxes are given by [33]:

$$F_i = \sum_{j \neq i} F_{ij}, \qquad F_{ij} = m_{ij} I \left(\frac{dU_i}{dt} - \frac{dU_j}{dt}\right) + D_{ij}(U_i - U_j) \tag{7.77}$$

Similarly to the scalar case, we look for the appropriate limiting coefficients α_{ij} to constrain the fluxes F_{ij}, namely [33]:

$$\bar{F}_i = \sum_{j \neq i} \bar{F}_{ij}, \qquad \bar{F}_{ij} = \alpha_{ij} F_{ij} \tag{7.78}$$

The limiting coefficients must be chosen in the way that preserves the generalised LED property in the final scheme [33]:

$$M_L \frac{dU}{dt} = LU + S(U) + \bar{F}(U) \tag{7.79}$$

with $\bar{F}(U)$ being a vector of non-linear anti-diffusive corrections with entries $\bar{F}_i$. In this thesis we use AFC with linearised FCT as shown in Section 6.3.3. Therefore, the final scheme reads [12]:

$$M_L U^{n+1} = M_L U^L + \Delta t \bar{F}(U^L, U^n) \tag{7.80}$$

where U^n denotes the solution at time t^n and U^L is the end-of-step low-order solution obtained with any time discretization scheme (eg. Forward Euler, SSP-eRK methods described in Section 6.1.2). The linearised raw anti-diffusive fluxes to be limited are computed according to the formula [33, 12]:

$$F_i = \sum_{j \neq i} F_{ij}, \qquad F_{ij} = m_{ij} I(\dot{U}_i^L - \dot{U}_j^L) + D_{ij}^{n+1}(U_i^L - U_j^L) \tag{7.81}$$

In the case of the time-dependent convection-diffusion equation we assumed that the matrix K and therefore entries d_{ij} are constant in time. In the case of the Euler equations the D_{ij}^{n+1} is time dependent and therefore we have to look for some explicit approximation to this term if we want to avoid implicitness and therefore non-linearity of the scheme. A worthwhile approximation is to compute the discrete Jacobian operator K^{n+1} and the artificial diffusion operator D^{n+1} using the low-order solution U^L. We restrict ourselves to this approach. However, investigation of another approximation for this term is an interesting open research topic. The approximation of the time derivative $\frac{dU^L}{dt}$ follows schemes (6.49)-(6.51). The method of choice is again (6.49).

There are several limiting techniques for systems of conservation laws proposed by Kuzmin et al. in [33]. In this thesis **limiting in terms of primitive variables** was used. It has the significant advantage that the primitive variables have an explicit physical meaning. Another advantage is that we separately compute the limiting coefficients for all scalar fluxes corresponding to primitive variables chosen for limiting and then combine them into the final limiting coefficient instead of considering the entire system at once. As the computation of limiting coefficients is done on the decoupled scalar level we are able to adopt Zalesak's fully multidimensional FCT algorithm introduced in the previous chapter. The scalar fluxes for the conservative variables f_{ij}^{ρ}, $f_{ij}^{\rho v_1}$, $f_{ij}^{\rho v_2}$ and $f_{ij}^{\rho E}$ are computed according to the formula (6.48) using respective entries of matrices.

In this thesis we restrict ourselves to limiting in terms of density ρ and pressure p as in most cases such an approach guarantees non-oscillatory solutions. As ρ is both a primitive and conservative variable the calculation of the corresponding flux f_{ij}^{ρ} is straightforward. For the pressure we use the following formulas [33]:

$$p_i = (\gamma - 1)\left[(\rho E)_i - \frac{|(\rho \mathbf{v})_i|^2}{2\rho_i}\right] \tag{7.82}$$

$$f_{ij}^{p} = (\gamma - 1)\left[f_{ij}^{\rho E} + \frac{|\mathbf{v}_i|^2}{2} f_{ij}^{\rho} - \mathbf{v}_i \cdot \mathbf{f}_{ij}^{\rho v}\right] \tag{7.83}$$

It is important to note that the flux f_{ij}^p is not symmetric, i.e. $f_{ij}^p \neq f_{ji}^p$. Therefore we need to slightly modify the limiting technique proposed in Section 6.3.2. Limiters based on each of the primitive variables chosen for limiting are calculated separately. Let us denote with u_i^L a low-order solution to one of the chosen primitive variables ρ or p and with f_{ij}^u the raw anti-diffusive fluxes corresponding to this variable. Then the limiting coefficients α_{ij}^u are calculated with the algorithm presented in [33, 72] generalized for non-nodal DOFs:

1. Evaluate the sums of positive and negative anti-diffusive fluxes into i-th DOF:

$$P_i^+ = \sum_{j \neq i} \max\{0, f_{ij}^u\}, \qquad P_i^- = \sum_{j \neq i} \min\{0, f_{ij}^u\} \tag{7.84}$$

2. Compute the distance to a local maximum and minimum at the bounds:

$$Q_i^+ = \frac{m_i}{\Delta t}(u_i^{\max} - u_i^L), \qquad Q_i^- = \frac{m_i}{\Delta t}(u_i^{\min} - u_i^L) \tag{7.85}$$

3. Determine the nodal correction factors for the net increment at i-th DOF:

$$R_i^+ = \min\left\{1, \frac{Q_i^+}{P_i^+}\right\}, \qquad R_i^- = \min\left\{1, \frac{Q_i^-}{P_i^-}\right\} \tag{7.86}$$

4. Control the sign of the raw anti-diffusive flux:

$$\alpha_{ij}^u = \min\{R_{ij}, R_{ji}\}, \qquad R_{ij} = \begin{cases} R_i^+, & \text{if } f_{ij}^u \geq 0 \\ R_i^-, & \text{if } f_{ij}^u < 0 \end{cases} \tag{7.87}$$

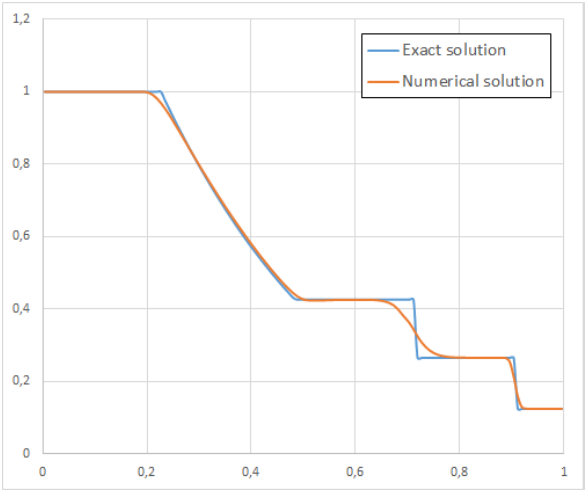
The final coefficient α_{ij} can be calculated by one of the strategies presented in [33, 73]. The method used in this thesis is to choose the minimum value of coefficients computed for the chosen primitive variables, i.e. [33]:

$$\alpha_{ij} = \min\{\alpha_{ij}^\rho, \alpha_{ij}^p\} \tag{7.88}$$

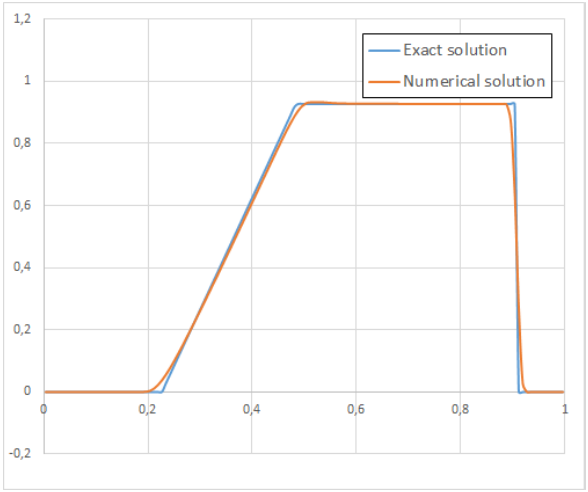
Now, we have all necessary ingredients to implement AFC with FCT-type flux limiter for systems of conservation laws and apply it to the Euler equations. The analysis of numerical results obtained with IGA with this stabilization method are presented in the next section.

7.6 Numerical results - Sod's shock tube

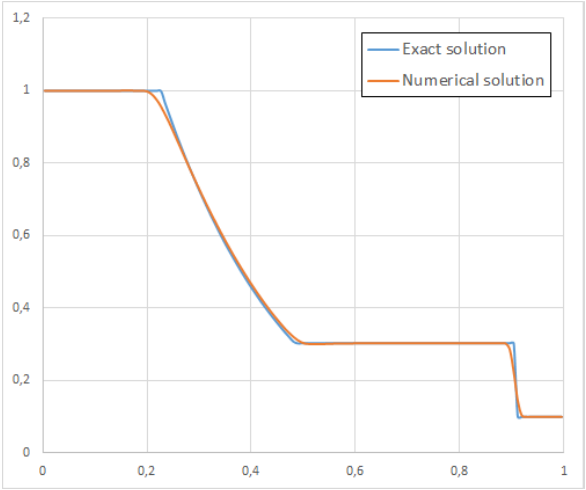
Fig. 7.5 presents the exact solution and numerical solution, obtained with IGA along with AFC stabilization, to Sod's shock tube problem (also at $t = 0.231$). The numerical solution was calculated on the space spanned by $p = 2$, 66×66 B-spline basis. The time discretization was done with the SSP-eRK scheme of order 3 with time step $\Delta t = 0.001$. One can notice that although coarser time and spatial discretizations were used, the result is of much better quality. It is free of oscillations but unfortunately the discontinuities are smeared. Especially large smearing occurs for the contact discontinuity. As in the case of the time-dependent convection-diffusion equation the result could be significantly improved by using a better approximation for the time derivative used for evaluation of anti-diffusive fluxes or by using non-linear FCT instead of linearised one.



(a) Density ρ along the cut-line $y = 0.5$



(b) Velocity v_1 along the cut-line $y = 0.5$



(c) Pressure p along the cut-line $y = 0.5$

Figure 7.5: Exact and numerical solutions to the Sod's shock tube benchmark problem at $t = 0.231$. Numerical solution obtained on $p = 2$, 66×66 B-spline basis with SSP-eRK-3 time discretization with time step $\Delta t = 0.001$.

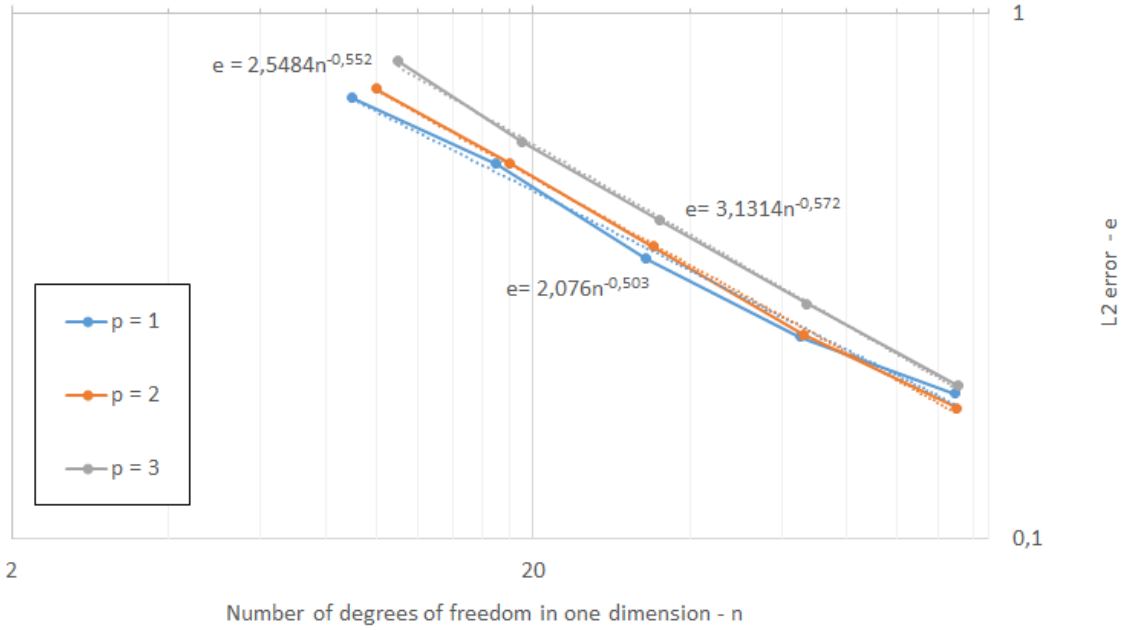


Figure 7.6: L2 errors of the numerical solution for density ρ at $t = 0.231$ to the Sod's shock tube benchmark problem solved on uniform $p = 1, 2, 3$ B-spline basis with SSP-eRK-3 time discretization scheme with time step $\Delta t = 0.001$ versus the number of DOFs in one direction.

Fig. 7.6 presents the L2 errors of numerical solution for the density ρ at $t = 0.231$ to Sod's shock tube problem solved on the space spanned uniform $p = 1, 2, 3$ B-spline basis versus the number of DOFs in one direction. The temporal discretization was done with the SSP-eRK-3 time discretization scheme and the time step size $\Delta t = 0.001$. As in the case of the convergence analysis of the problem of convection of an isentropic vortex, the time step was not refined during the analysis but it was chosen such as to assure that any further refinement of the temporal discretization would not influence the result significantly. The orders of convergence for all orders of basis functions are limited by 0.5. This was caused by the discontinuity in the domain. One can notice from the Chapter 4 that the projection of an analytical initial condition to the considered B-spline space limits the orders of convergence and therefore the scheme can not achieve higher orders of convergence than those achieved by projecting the initial data. Therefore, we can conclude that there is a need for a method that will workaround this limitation. There are several ideas how that could be achieved. Local refinement would significantly improve the robustness of the method for the problems including the discontinuities. Another promising approach is to modify the multiplicities of knot values to achieve locally adjusted continuity properties which are required in the given region. Further investigation and development of those ideas is an interesting open research topic being beyond the scope of this thesis.

It was shown in this chapter that isogeometric analysis is a suitable tool for solving the compressible Euler equations. Similarly to the standard FEM the method performs well for problems not involving discontinuities and shock waves but problems that contain them require some type of stabilization. The method of choice in the thesis is Algebraic Flux

Correction. IGA applied together with AFC with FCT-type limiting generalised for systems of conservation laws gave results free of non-physical oscillations but overly diffusive. This issue could, however, be fixed by additional developments proposed in the previous chapters. The orders of convergence of the method are significantly limited by the discontinuity in the domain. Elaborating the way to avoid this limitation is of utmost importance. Another important research question that needs to be answered is the surprising limitation of the orders of convergence of the IGA-based solver applied to smooth problems to $p + 1$.

Chapter 8

Conclusions

Isogeometric analysis is a very powerful tool for the analysis of PDE problems. It is a very promising approach for optimization processes in the turbomachinery design. It has two important advantages over classical FEA. Firstly, it gives the possibility to exactly represent the geometry. This is very important for gas turbines as most of the flow problems require an accurate resolution of flow phenomena close to the walls. Secondly, it creates a bidirectional link between CAD and analysis. There is no need for computationally expensive re-meshing in each optimization cycle since geometry modifications can be realised by shifting the control points and modifying the geometrical mapping.

The first part of the project work, carried out during the literature study period, included writing Matlab codes for building and evaluation of B-spline basis functions as well as solving the Poisson problem on arbitrary 2D geometries. This Matlab code, written for the needs of the literature study, turned out to be not efficient enough for more complex applications. Thus, it was decided to use the C++ library G+SMO.

Before solving more complex problems we had to consider the projection of the analytical data to the space spanned by B-splines. This could not be achieved by nodal assignment of function values due to the fact that degrees of freedom in IGA are not nodal. Therefore, we considered the L2 projection. It was very efficient for smooth functions but discontinuities were causing non-physical under- and over-shoots. The constrained L2 projection was a perfect cure for this issue. Unfortunately, the price to be paid was a less accurate, in sense of L2 norm, approximation compared to standard L2 projection.

The next step was to implement a solver for the stationary convection-diffusion equation. This solver was working perfectly for diffusion-dominated problems achieving $p + 1$ order of convergence. However, convection-dominated problems could not be solved by pure IGA and required proper stabilization. The idea of using AFC stabilization with TVD-type limiting instead of SUPG turned out to be very good. AFC achieved higher orders of convergence than SUPG and produced solutions that were free of non-physical values in the domain.

A generalization to time-dependent problems required the selection of a suitable time discretization scheme. We considered the forward Euler method and explicit strong stability preserving Runge-Kutta methods of orders 2 and 3, respectively. For smooth initial conditions the solver was capable of achieving non-oscillatory solutions even in the limit of purely convective problems. A convergence analysis of the different time discretization schemes was performed. As expected, the higher the order of the time discretization method the faster the reduction of the L2 error. For non-smooth initial conditions the unstabilized Galerkin

method was not able to produce non-oscillatory solution even for very fine spatial and temporal discretizations. Therefore, the AFC approach with linearised FCT-type limiter was used for stabilization. The results obtained with this approach were free of non-physical oscillations but overly diffusive. This unwanted feature was caused by the approximation of the time derivative during the computation of anti-diffusive fluxes in the linearised FCT scheme. Therefore, there is a need for better approximations of the time derivative or implementing the more accurate non-linear FCT-type limiters. It was also shown that all considered time discretization schemes converge at the same speed when the AFC stabilization is enabled, and therefore, there is no gain in using time-stepping schemes of higher order for the types of problems considered in this thesis. This surprising observation is very promising as decreasing the computational costs of the time discretization would be a good balance for the higher costs of the IGA/AFC approach (comparing to standard finite volumes).

As the final step, the IGA solver for the compressible Euler equations was successfully implemented and validated for the isentropic vortex benchmark problem. Unfortunately, the order of convergence was limited to 2 for all orders of basis functions. This surprising and unwanted behaviour requires further investigation. Another conclusion from this problem is that the solver is not optimized enough to handle more complex test cases. AFC stabilization with linearised FCT-type limiting generalized for systems of conservation laws allowed to solve problems involving shock waves. Sod's shock tube benchmark problem was successfully solved using this stabilization method. As in the case of the time-dependent convection-diffusion equation the results can be significantly improved by using better approximations of the time derivative in the evaluation of anti-diffusive fluxes or by using non-linear FCT-type limiting instead of the linearised one.

This thesis project resulted in an interesting side conclusion. It is possible to extend the algebraic flux correction methodology to higher order basis functions. Similar work was already done before by Kuzmin in [16] where quadratic Lagrange FEM basis functions were used. The paper concluded with the statement: "the involved programming effort and the overhead cost are quite significant, whereas the improvements are often marginal, if any". From the point of view of application along with AFC, IGA is superior to quadratic FEM as the method works perfectly for higher order basis functions without any additional implementation effort due to the positivity of the basis functions over their entire support. Therefore, we can conclude that B-splines (and possibly NURBS) are a viable tool for extending the AFC framework to higher order functions.

We can draw the final conclusion that the goals of the thesis were successfully achieved but more research required to apply IGA for real industrial aerodynamic flow problems. The compressible Euler equations provide a good approximation of flow phenomena of viscous fluids only inside the domain, at a distance from the walls. In turbines, with their very narrow channels, viscosity plays a crucial role. Therefore, the full Navier-Stokes equations need to be solved to obtain solutions of good quality. The next section contains a summary of further developments that are required to apply IGA to real, industrial turbomachinery problems as well as side research questions that raised up during this project work and are worth further investigation.

8.1 Future developments

The final result of this thesis is a functional IGA solver for the compressible Euler equations. The main development necessary for industrial application is its generalization to the full Navier-Stokes equations. Another significant limitation of the current code is efficiency. To be able to consider more complex test cases it has to be optimized significantly. Furthermore, a generalization of the solver to multi-patch problems is required since most industrial applications require complicated domains that can not be directly mapped from the square domain. Considering the geometry of the domain, the solver must also be generalised to three dimensional problems. The IGA approach to compressible flow problems presented in this thesis should be generalized from B-splines to NURBS to be capable of, for example, solving problems defined on circular domains.

The AFC stabilization method needs to be improved by implementing alternative approximations of the time derivative in the computation of anti-diffusive fluxes since the current one result in overly diffusive solution profiles. Implementation of non-linear FCT is also worth considering. Application of different time-discretization schemes together with AFC should be investigated in more detail to assure fast convergence and to prevent excessive computations. Finally, investigation of the source of limitations of orders of convergence for the IGA Euler solver in case of the isentropic vortex benchmark is of utmost importance.

To conclude, this thesis set the base for further research in field of isogeometric methods for compressible flow problems and raised several interesting research questions.

Bibliography

[1] J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, Ltd., 2009.

[2] T.J.R. Hughes, J.A. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39–41):4135 – 4195, 2005.

[3] P. Trontin. Isogeometric analysis of Euler compressible flow. Application to aerodynamics. *50th AIAA*, 2012.

[4] A.V. Vuong. *Adaptive Hierarchical Isogeometric Finite Element Methods*. PhD thesis, TU München, 2011.

[5] F. Brunero. Discontinuous galerkin methods for isogeometric analysis. Master's thesis, University of Milano, 2012.

[6] A.N. Brooks and T.J.R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32(1–3):199 – 259, 1982.

[7] I. Harari and T.J.R Hughes. Stabilized finite element methods for steady advection—diffusion with production. *Computer Methods in Applied Mechanics and Engineering*, 115(1–2):165 – 191, 1994.

[8] A. Russo. Bubble stabilization of finite element methods for the linearized incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 132(3–4):335 – 343, 1996.

[9] R. Codina. On stabilized finite element methods for linear systems of convection–diffusion-reaction equations. *Computer Methods in Applied Mechanics and Engineering*, 188(1–3):61 – 82, 2000.

[10] D. Kuzmin and S. Turek. Flux correction tools for finite elements. *Journal of Computational Physics*, 175(2):525 – 558, 2002.

[11] D. Kuzmin and S. Turek. High-resolution FEM-TVD schemes based on a fully multidimensional flux limiter. *Journal of Computational Physics*, 198(1):131 – 158, 2004.

[12] D. Kuzmin. Explicit and implicit FEM-FCT algorithms with flux linearization. *Journal of Computational Physics*, 228(7):2517–2534, 2009.

[13] D. Kuzmin, M. Möller, and S. Turek. High-resolution FEM–FCT schemes for multidimensional conservation laws. *Computer Methods in Applied Mechanics and Engineering*, 193(45–47):4915 – 4946, 2004.

[14] D. Kuzmin and M. Möller. *Flux-Corrected Transport*, chapter Algebraic flux correction I. Scalar conservation laws. Springer, 2005.

[15] D. Kuzmin and M. Möller. *Flux-Corrected Transport*, chapter Algebraic flux correction II. Compressible Euler equations. Springer, 2005.

[16] D. Kuzmin. On the design of algebraic flux correction schemes for quadratic finite elements. *Journal of Computational and Applied Mathematics*, 218(1):79 – 87, 2008. Special Issue: Finite Element Methods in Engineering and Science (FEMTEC 2006)Special Issue: Finite Element Methods in Engineering and Science (FEMTEC 2006).

[17] PetIGA on Bitbucket. `https://bitbucket.org/dalcinl/petiga/`, February 2015.

[18] N. Collier, L. Dalcin, and V.M. Calo. PetIGA: High-performance isogeometric analysis. *CoRR*, (1305.4452), 2013.

[19] Official webpage of GeoPDEs. `http://geopdes.apnetwork.it/`, February 2015.

[20] C. de Falco, A. Reali, and R. Vázquez. GeoPDEs: a research tool for isogeometric analysis of pdes. *Adv. Eng. Softw.*, (42(12)), 2011.

[21] Igatools on GoogleCode. `https://code.google.com/p/igatools/`, February 2015.

[22] M.S. Pauletti, M. Martinelli, N. Cavallini, and P. Antolín. Igatools: an isogeometric analysis library. *I.M.A.T.I.-C.N.R. Technical Report*, (3PV14/1/0), 2014.

[23] IgaFEM on SourceForge. `http://sourceforge.net/projects/cmcodes/`, February 2015.

[24] Official G+SMO webpage. `http://www.gs.jku.at/gs_gismo.shtml`, February 2015.

[25] C. de Boor. *A practical guide to splines*. Springer, 2001.

[26] X. Li. Iga seminar: B-splines and nurbs. *Lecture Notes*, Delft University of Technology, 2014. `http://ta.twi.tudelft.nl/nw/users/matthias/teaching/iga_2014/BsplineIntroduction_Xiaozhou.pdf`.

[27] L. Piegl and W. Tiller. *The NURBS Book (Monographs in Visual Comunicatin), Second Edition*. Springer, 1997.

[28] G.E. Farin. *NURBS Curves and Surfaces: from Projective Geometry to Practical Use, Second Edition*. A.K. Peters, Ltd., 1999.

[29] M. Möller. Iga seminar: Assembly strategies in isogeometric analysis. *Lecture Notes*, Delft University of Technology, 2014. `http://ta.twi.tudelft.nl/nw/users/matthias/teaching/iga_2014/Assembly_Matthias.pdf`.

[30] A. Mantzaflaris, A. Jaeschke, S. Kleiss, et al. G+Smo (Geometry plus simulation modules) v. 0.8. `http://gs.jku.at/gismo`, 2014.

[31] B. Juettler, U. Langer, A. Mantzaflaris, Stephen Moore, and Walter Zulehner. Geometry + simulation modules: Implementing isogeometric analysis. *Proc. Appl. Math. Mech.*, 14(1):961–962, 2014. Special Issue: 85th Annual Meeting of the Int. Assoc. of Appl. Math. and Mech. (GAMM), Erlangen 2014.

[32] EIGEN on Tuxfamily. `http://eigen.tuxfamily.org/`, July 2015.

[33] D. Kuzmin, M. Möller, and M. Gurris. *Flux-Corrected Transport*, chapter Algebraic flux correction II. Compressible flow problems. Springer, 2012.

[34] J. Jansson. The finite element method. *Lecture Notes*, KTH Royal Institute of Technology, 2012. `http://www.csc.kth.se/utbildning/kth/kurser/DN2260/fem12/M2.pdf`.

[35] P.K. Smolarkiewicz and G.A. Grell. A class of monotone interpolation schemes. *Journal of Computational Physics*, 101(2):431 – 440, 1992.

[36] R. Liska, M. Shashkov, P. Váchal, and B. Wendroff. Optimization-based synchronized flux-corrected conservative interpolation (remapping) of mass and momentum for arbitrary Lagrangian–Eulerian methods. *Journal of Computational Physics*, 229(5):1467 – 1497, 2010.

[37] P.E. Farrell, M.D. Piggott, C.C. Pain, G.J. Gorman, and C.R. Wilson. Conservative interpolation between unstructured meshes via supermesh construction. *Computer Methods in Applied Mechanics and Engineering*, 198(33–36):2632 – 2642, 2009.

[38] S.T. Zalesak. Fully multidimensional flux-corrected transport algorithms for fluids. *Journal of Computational Physics*, 31(3):335 – 362, 1979.

[39] B. Munson. *Fundamentals of Fluid Mechanics*. John Wiley & Sons, Ltd., 1990.

[40] S.A. Socolofsky and G.H. Jirka. Advective diffusion equation. *Lecture Notes*, Texas A&M University, 2004. `https://ceprofs.civil.tamu.edu/ssocolofsky/cven489/downloads/book/ch2.pdf`.

[41] C.A.J. Fletcher. The group finite element formulation. *Comput. Methods Appl. Mech. Eng*, 37:225–243, 1983.

[42] D. Kuzmin. *Flux-Corrected Transport*, chapter Algebraic flux correction I. Scalar conservation laws. Springer, 2012.

[43] D. Kuzmin. Algebraic flux correction for finite element discretizations of coupled systems. *Computational Methods for Coupled Problems in Science and Engineering II, CIMNE, Barcelona*, pages 653–656, 2007.

[44] D. Kuzmin. On the design of general-purpose flux limiters for finite element schemes. I. Scalar convection. *Journal of Computational Physics*, 219(2):513 – 531, 2006.

[45] C. Hirsch. *Numerical Computation of internal and external flows; Volume 2: Computational Methods for Inviscid and Viscous Flows*. John Wiley & Sons, Ltd., 1990.

[46] P.D. Lax. *Hyperbolic Systems of Conservation Laws and the Mathematical Theory of Shock Waves*. Number nos. 11-16 in CBMS-NSF Regional Conference Series in Applied Mathematics. Society for Industrial and Applied Mathematics, 1973.

[47] A. Harten. High resolution schemes for hyperbolic conservation laws. *Journal of Computational Physics*, 49(3):357 – 393, 1983.

[48] S.K. Godunov. Finite difference method for numerical computation of discontinous solutions of the equations of fluid dynamics. *Mat. Sbornik*, 47:271–306, 1959.

[49] A. Jameson. Positive schemes and shock modelling for compressible flows. *International Journal for Numerical Methods in Fluids*, 20(8-9):743–776, 1995.

[50] V. John and P. Knobloch. On spurious oscillations at layers diminishing (SOLD) methods for convection–diffusion equations: Part i – a review. *Computer Methods in Applied Mechanics and Engineering*, 196(17–20):2197 – 2215, 2007.

[51] C. Giannelli, B. Juettler, S. Kleiss, A. Mantzaflaris, B. Simeon, and J. Speh. THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis. May 2015 (submitted).

[52] J.H. Ferziger and M. Peric. *Computational Methods for Fluid Dynamics*. Springer Berlin Heidelberg, 2001.

[53] R.L. Burden. and J.D. Faires. *Numerical Analysis*. Number v. 1 in Numerical Analysis. Brooks/Cole, 2001.

[54] S. Gottlieb, C. Shu, and E. Tadmor. Strong stability-preserving high-order time discretization methods. *SIAM Review*, 43(1):89–112, 2001.

[55] D. Kuzmin. A vertex-based hierarchical slope limiter for -adaptive discontinuous Galerkin methods. *Journal of Computational and Applied Mathematics*, 233(12):3077 – 3085, 2010. Finite Element Methods in Engineering and Science (FEMTEC 2009).

[56] R. Courant, K. Friedrichs, and H. Lewy. Über die partiellen Differenzengleichungen der mathematischen Physik. *Mathematische Annalen*, 100:32–74, 1928.

[57] D.L. Book. *Flux-Corrected Transport*, chapter The conception, cestation, cirth, and infancy of FCT. Springer, 2005.

[58] J.P. Boris and D.L. Book. Flux-corrected transport. I. SHASTA, a fluid transport algorithm that works. *Journal of Computational Physics*, 11(1):38 – 69, 1973.

[59] P. Wesseling. *Principles of Computational Fluid Dynamics*. Springer, 2001.

[60] E.F. Toro. *Riemann Solvers and Numerical Methods for Fluid Dynamics. A Practical Introduction*. Springer, 1999.

[61] M. Möller. *Adaptive high-resolution finite element schemes*. PhD thesis, TU Dortmund, 2008.

[62] M. Feistauer and V. Kučera. On a robust discontinuous Galerkin technique for the solution of compressible flow. *Journal of Computational Physics*, 224(1):208 – 221, 2007.

[63] P.L. Roe. Approximate Riemann solvers, parameter vectors, and difference schemes. *Journal of Computational Physics*, 43(2):357 – 372, 1981.

[64] R.A. Shapiro. *Adaptive Finite Element Solution Algorithm for the Euler Equations*. Notes on Numerical Fluid Mechanics and Multidisciplinary Design. Vieweg+Teubner Verlag, 2013.

[65] M. Gurris. *Implicit finite element schemes for compressible gas and particle-laden gas flows*. PhD thesis, TU Dortmund, 2009.

[66] H.C Yee, N.D Sandham, and M.J Djomehri. Low-dissipative high-order shock-capturing methods using characteristic-based filters. *Journal of Computational Physics*, 150(1):199 – 238, 1999.

[67] F. Davoudzadeh, H. McDonald, and B.E. Thompson. Accuracy evaluation of unsteady (CFD) numerical schemes by vortex preservation. *Computers Fluids*, 24(8):883 – 895, 1995.

[68] P. Castonguay, P.E. Vincent, and A. Jameson. Application of high-order energy stable flux reconstruction schemes to the Euler equations. In *49th AIAA Aerospace Sciences Meeting*, volume 686, 2011.

[69] G.A. Sod. A survey of several finite difference methods for systems of nonlinear hyperbolic conservation laws. *Journal of Computational Physics*, 27(1):1 – 31, 1978.

[70] Official webpage of NUMERICA. `http://eleuteriotoro.com/software/`, July 2015.

[71] S.T. Zalesak. *Flux-Corrected Transport*, chapter The design of flux-corrected transport (FCT) algorithms for structured grids. Springer, 2005.

[72] D. Kuzmin, M. Möller, J. N. Shadid, and M. Shashkov. Failsafe flux limiting and constrained data projections for equations of gas dynamics. *Journal of Computational Physics*, 229(23):8766 – 8779, 2010.

[73] R. Löhner, K. Morgan, J. Peraire, and M. Vahdati. Finite element flux-corrected transport (FEM–FCT) for the Euler and Navier–Stokes equations. *International Journal for Numerical Methods in Fluids*, 7(10):1093–1109, 1987.