UNIVERSITÉ CÔTE D'AZUR

ÉCOLE DOCTORALE
SCIENCES FONDAMENTALES ET APPLIQUÉES

THÈSE DE DOCTORAT

Méthodes isogéométriques pour les équations aux dérivées partielles hyperboliques

Asma GDHAMI

INRIA -LAMSIN

Présentée en vue de l'obtention du grade de docteur en Mathématiques **d'**Université Côte d'Azur
et de l'Université de Tunis El Manar
Dirigée par: Regis DUVIGNEAU / Maher MOAKHER
Soutenue le : 17-12-2018

Devant le jury, composé de :
Amel BEN ABDA, Professeur, Université de Tunis El Manar
Christophe CHALONS, Professeur, Université de Versailles
Regis DUVIGNEAU, Chargé de Recherche, INRIA, Université Cote d'Azur
Maatoug HASSINE, Professeur, Université de Monastir
Maher MOAKHER, Professeur, ENIT, Université de Tunis El Manar
Claire SCHEID, Maître de conférences, Université Cote d'Azur

Inria inventeurs du monde numérique

ENIT LAMSIN

MÉTHODES ISOGÉOMÉTRIQUES POUR LES ÉQUATIONS AUX DÉRIVÉES PARTIELLES HYPERBOLIQUES

Jury:

Rapporteurs:

Mr. Christophe CHALONS, Professeur, Université de Versailles.
Mr. Maatoug HASSINE, Professeur, Université de Monastir.

Examinateurs:

Mrs. Amel BEN ABDA, Professeur, ENIT, Université de Tunis El Manar.
Mrs. Claire SCHEID, Maître de conférences, Université de Côte Azur.

Mr. Regis DUVIGNEAU, Chargé de Recherche, INRIA, Université de Côte Azur.
Mr. Maher MOAKHER, Professeur, ENIT, Université de Tunis El Manar.

MÉTHODES ISOGÉOMÉTRIQUES POUR LES ÉQUATIONS AUX DÉRIVÉES PARTIELLES HYPERBOLIQUES

Resumé: L'nalyse isogéométrique (AIG) est une méthode innovante de résolution numérique des équations différentielles, proposée à l'origine par Thomas Hughes, Austin Cottrell et Yuri Bazilevs en 2005. Cette technique de discrétisation est une généralisation de l'analyse par éléments finis classiques (AEF), conçue pour intégrer la conception assistée par ordinateur (CAO), afin de combler l'écart entre la description géométrique et l'analyse des problèmes d'ingénierie. Ceci est réalisé en utilisant des B-splines ou des B-splines rationnelles non uniformes (NURBS), pour la description des géométries ainsi que pour la représentation de champs de solutions inconnus.

L'objet de cette thèse est d'étudier la méthode isogéométrique dans le contexte des problèmes hyperboliques en utilisant les fonctions B-splines comme fonctions de base. Nous proposons également une méthode combinant l'AIG avec la méthode de Galerkin discontinue (GD) pour résoudre les problèmes hyperboliques. Plus précisément, la méthodologie de GD est adoptée à travers les interfaces de patches, tandis que l'AIG traditionnelle est utilisée dans chaque patch. Notre méthode tire parti de la méthode de l'AIG et la méthode de GD.

Les résultats numériques sont présentés jusqu'à l'ordre polynomial $p = 4$ à la fois pour une méthode de Galerkin continue et discontinue. Ces résultats numériques sont comparés pour un ensemble de problèmes de complexité croissante en $1D$ et $2D$.

Mots clés: problèmes hyperboliques, méthode des éléments finis, méthode de Galerkin discontinue, analyse isogéométrique, fonctions B-splines, extraction de Bézier, ajustement des courbes, méthode de moindres carrés.

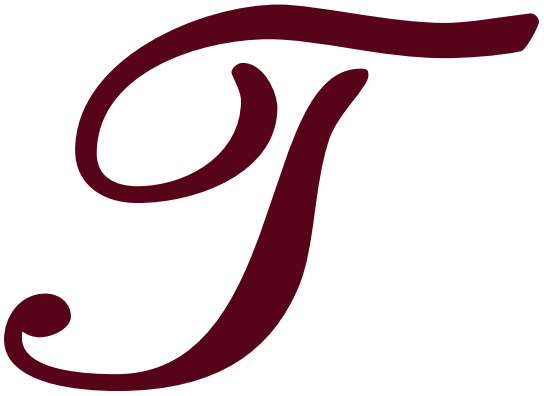
ISOGEOMETRIC METHODS FOR HYPERBOLIC PARTIAL DIFFERENTIAL EQUATIONS

Abstract: Isogeometric Analysis (IGA) is a modern strategy for numerical solution of partial differential equations, originally proposed by Thomas Hughes, Austin Cottrell and Yuri Bazilevs in 2005. This discretization technique is a generalization of classical finite element analysis (FEA), designed to integrate Computer Aided Design (CAD) and FEA, to close the gap between the geometrical description and the analysis of engineering problems. This is achieved by using B-splines or non-uniform rational B-splines (NURBS), for the description of geometries as well as for the representation of unknown solution fields.

The purpose of this thesis is to study isogeometric methods in the context of hyperbolic problems using B-splines as basis functions. We also propose a method that combines IGA with the discontinuous Galerkin (DG) method for solving hyperbolic problems. More precisely, DG methodology is adopted across the patch interfaces, while the traditional IGA is employed within each patch. The proposed method takes advantage of both IGA and the DG method.

Numerical results are presented up to polynomial order $p = 4$ both for a continuous and discontinuous Galerkin method. These numerical results are compared for a range of problems of increasing complexity, in $1D$ and $2D$.

Keywords: hyperbolic problems, Finite Element method, discontinuous Galerkin method, Isogeometric analysis, B-spline functions, Bézier extraction, curve fitting, least squares method.



My dear father Mohamed and my sweet mother Kawther, for your patience, sacrifices and encouragement.

My husband Aymen, I am more thankful than I can possibly put down in words, for giving me energy when mine is running low, for having an open ear for my worries and my successes, for your faith in me, and above all, for your closeness and your love.

My son Dali, for making me smile even on the toughest days.

My sister Haifa & my brother Maher, for your kindness, your love and your concern.

My aunts Emna & Raoudha and my mother in law Jalila

All my friends, for friendship, love and moments spent together.

"***Nothing is lost, everything is transformed***."

Antoine Lavoisier

ACKNOWLEDGMENTS

THese PhD years have been a rich experience from the professional as well as the personal points of view.

First of all I would like to express my gratitude **to my supervisors, Regis Duvigneau & Maher Moakher** for inspiring me, introducing me to new challenges and for helping me carry out this work. I would like to thank them for showing much interest in my work, and for their help and support at the frequently advising sessions. At your sides, I have learnt a lot. You shared with me your technical knowledges and above all your scientific rigorous. In particular, thanks to you, I got the opportunity to attend international congresses. It represents a lot to me. I would like to express my sincere thanks for your patience, your help, your regular availability and your encouragement and advice. I dedicate this work to you reflecting my deep respect.

I express my heartfelt thanks to **Mrs. Amel BEN ABDA & Mrs. Claire SCHEID** for being my thesis examiner and providing valuable suggestions and corrections.

Also, I present my highly express of thanks to **Sir Christophe CHALONS & Sir Maatoug HASSINE** for agreeing to be the reviewers of my work and for your constructive suggestions.

Then, I address a special appreciation for **Sir. Mekki Ayadi** that helped me to accomplish this work by extensive discussions. I would like to express my sincere thanks for your patience, your help, your encouragement and advice.

My next big thank you is to my families Gdhami, Essid and Azaouzi. To my parents Mohamed and Kaouther, my brother and my sister for their unconditional love and support all though my life. To them I owe all that I am and all that I have ever accomplished and it is to them I dedicate this thesis. Moreover, I will never forget my cousins Sana, Ichraf, Boutheina, Soumaya & Olfa who have always encouraged me to accomplish my studies.

On a more personal note, I would like to express my deep gratitude to my husband Aymen for his support. This work is dedicated to him and to my son with my deepest love.

I address a special message for my second family: my friends from the Modeling Laboratory in Engineering Sciences (LAMSIN) National Engineering School of Tunis (ENIT), Boutheina, Imen, Rabeb, Maroua, hamouda, Anis, ... and my friends of the National Institute for Research in Computer Science and Automatic of Nice (INRIA), I would like to express my gratitude for their continuous encouragement.

In the life path, we meet people that can change our life in a different measure - some change it forever. I am sure I would have forgotten some people in this acknowledgement section, that is why for the sake of completion I am thanking all the people who shared with me parts of these years. Thanks for being part of my path. In a certain way, that you maybe do not measure, you contribute to allow me to arrive here today.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

List of Abbreviations

CAD	Computer aided design.
CAGD	Computer aided geometric design.
CAE	Computer aided engineering.
CFL	Courant-Friedrichs-Lewy.
DGM	Discontinuous Galerkin method.
d.o.f.	Degrees of freedom.
FDM	Finite difference method.
FEM	Finite element method.
FVM	Finite volume method.
GLS	Galerkin Least-Squares.
IGA	Isogeometric analysis.
NURBS	Non-uniform-rational B-spline.
ODE	Ordinary differential equation.
PDE	Partial differential equation.
PG	Petrov-Galerkin.
PSPG	Pressure-stabilizing Petrov-Galerkin.
RK	Runge-Kutta.
SUPG	Streamline-Upwind Petrov-Galerkin.

List of Symbols

p	Degree of the polynomial approximation.
n	Number of B-spline functions of degree p.
n_G	Number of nodes for Gauss quadrature methods.
$\check{n}$	Dimension of the time steps.
N_{el}	Finite number of cells of DG method.
$\mathfrak{N}_{el}$	Finite number of patchs of IGDG method.
N	Dimension of the polynomial approximation.
B_p^k	k-th Bernstein polynomial of degree p in parameter space.
$\mathfrak{B}_p^k$	k-th Bernstein polynomial of degree p in physical space.
$\mathscr{N}_{i,p}$	i-th B-spline function of degree p in parameter space.
$N_{i,p}$	i-th B-spline function of degree p in physical space.
$\mathscr{R}_i^p$	i-th NURBS function of degree p.
C_p	Bézier curve of degree p.
R_p	Rational Bézier curve of degree p.
S	Bézier surface.
$\mathscr{C}_p$	B-spline curve of degree p.
$\mathscr{S}$	B-spline surface.
V	B-spline volume.
$\mathfrak{C}_p$	NURBS curve of degree p.
$\mathfrak{S}$	NURBS surface.
Ξ	Knot vector.
Ω	Physical space.
$\partial\Omega$	Boundary of the domain Ω.
$\widetilde{\Omega}$	Parameter space.
$\widehat{\Omega}$	Reference element.

Δt	Time steps.
$h_1, h_2, \mathfrak{h}$	Mesh size.
ω^G	The weights of nodes for Gauss quadrature methods.
v	Test function.
$u_h^{k^+}$	The field on the exterior of the element boundary Γ^k.
$u_h^{k^-}$	The field on the interior of the element boundary Γ^k.
C_{CFL}	Courant-Friedrichs-Lewy (CFL) number.
c	Advection velocity.
$\overrightarrow{c} = (c_x, c_y)$	Velocity field.
$\overrightarrow{n}$	The outward unit normal.
$\overrightarrow{n}^e$	The outer unit normal to Γ^e of the element Ω^e.
T	Final time.
t^0	Initial time.
τ	SUPG stabilization parameter.
r	Convergence rate.
M	Mass matrix for the $\mathbb{P}_1$ FEM.
M^{-1}	Inverse of mass matrix.
M_L^s	SUPG lumped mass matrix.
R	Stiffness matrix for the $\mathbb{P}_1$ FEM.
M_1^s	Mass matrix for the $\mathbb{P}_1$ SUPG-FEM.
M_2^s	Mass matrix for the $\mathbb{P}_2$ SUPG-FEM.
R_1^s	Stiffness matrix for the $\mathbb{P}_1$ SUPG-FEM.
R_2^s	Stiffness matrix for the $\mathbb{P}_2$ SUPG-FEM.
M^B	Mass matrix for the B-spline IGFEM.

$(M^B)^T$	Transpose of the mass matrix for the B-spline IGFEM.
R^B	Stiffness matrix for the B-spline IGFEM.
Q	The matrix defined by the least square method.
$\mathbb{M}^k$	Local mass matrix.
$\mathbf{R}^k$	Local stiffness matrix.
$\mathscr{M}^k$	Local mass matrix for IGDGM.
$\mathscr{R}^k$	Local stiffness matrix for IGDGM.
J	The Jacobian matrix.
$\mid J \mid$	Determinant of the Jacobian.
J^e	The elemental Jacobian matrix in the physical domain Ω^e.
$\partial_x u = \frac{\partial u}{\partial x}$	Partial derivative of u with respect to space x.
$\partial_t u = \frac{\partial u}{\partial t}$	Partial derivative of u with respect to time t.
$\otimes$	Tensor product.
C^k	The set of functions with $k-th$ order continuous derivatives.
$\mathscr{L}$	Differential operator.
$\mathbb{T}$	Transformation of the parametric domain to the physical domain.
$\mathbb{T}^{-1}$	Inverse of transformation $\mathbb{T}$.
$\widehat{F}_e$	Numerical flux in the patch Ω^e.
$\widehat{f}_{CEN}$	Central flux.
$\widehat{f}_G$	Godunov flux.
$\widehat{f}_{LF}$	Lax-Friedrichs flux.

CHAPTER 1

INTRODUCTION

Hyperbolic systems of partial differential equations (PDEs) are mathematical models expressing the conservation of a physical quantity, as for instance mass, energy, etc. They arise naturally from the conservation laws in physics. In particular, they describe a wide variety of phenomena that involve wave motion (such as acoustic, elastic, electromagnetic) or the advective transport of substances.

The wide range of applications of hyperbolic PDEs led to a very intense research activity in this field. It allowed to develop very early a set of numerical methods for accurate and computationally efficient approximations to the solutions of such problems. There are three major families of methods which are widely used: the finite difference method, the finite volume method and the finite element method. These methods have proved to be extremely useful in modeling a broad set of phenomena. To keep this thesis self-contained, we briefly introduce each of these three methods in the context of hyperbolic PDEs.

Historically, the finite difference method (FDM) was the first method used to produce approximations of the solutions of hyperbolic PDEs. They were introduced by Euler in the $18th$ century and represent the easiest method to solve problems on simple geometries [64]. The main idea of this method, is to replace the functional derivatives of the unknown by their FD approximations. The FDM is notable for the large variety of schemes that can be used to approximate a given PDE; e.g. explicit schemes (Forward Euler, Upwind, Lax-Friedrichs, Lax-Wendroff, Leapfrog, ...) and implicit schemes (Backward Euler, Crank-Nicolson,...) [88]. Although this method can be easily formulated and implemented, its application to problems with realistic geometries is rather cumbersome, thus making the method not very attractive for industrial problems. This fact urged the need for other methods with more flexibility, such as finite volume and finite element methods.

The finite volume method (FVM) can handle complex geometries which makes it more attractive for complex problems than the FDM. It is based on the conservative form instead of the differential form to estimate the values of unknown fields. We split the domain into grid cells and approximate the total integral over each grid cell, or actually the cell average, which is this integral divided by the volume of the cell. The links between the cell quantities in the FVM rely on the flux between neighbouring control volumes. This means that the FVM represents the flux of information within the structure of the mesh in a conservative way [56]. It allows the use of unstructured grids to handle complex geometries.

The finite element method (FEM), had its origins in the early 1960*s* and is nowadays the predominating method in analysis of elliptic or parabolic problems due to its flexibility to represent complex geometric domains and its strong theoretical basis. The idea consists in decomposing the domain into many small, "finite" elements which are defined by a set of nodal points and interpolating basis functions.
Although the FEM has been used widely in simulating many physical phenomena due to its flexibility to represent complex geometric domains, it is well known in the FE literature that numerical difficulties arise when solving hyperbolic PDEs. Indeed, when using the standard Galerkin FE method applied to hyperbolic PDEs, unwanted spurious (non-physical) oscillations (Gibbs phenomenon) are frequently detected in the numerical solutions. A cure to this drawback, widespread in the literature, is to add some "artificial" viscosity to a standard (unstable) numerical scheme. On the one hand, this artificial viscosity should damp the oscillations but, on the other hand, it should not smear the numerical solution. In the late 1970*s* and early 1980*s*, a large number of so–called stabilized methods have been developed with different ideas [14] [32] [48]. This is achieved through the use of a Petrov-Galerkin formulation [35], where the test functions are modified such that they weight the upstream node more than the downstream node [17] [38]. Among them, the most popular, so called Streamline-Upwind Petrov–Galerkin (SUPG) method, was introduced by Brooks and Hughes. It was first proposed in the context of advection–diffusion equations and incompressible Navier–Stokes equations [14], and then extended to various other problems, e.g., coupled multidimensional advective–diffusive systems [44], first–order linear hyperbolic systems [49] or first–order hyperbolic systems of conservation laws [45].
Later, Galerkin/least-squares (GLS) has emerged as a generalization of the SUPG method, developed by Hughes et al. for convective transport problems [9] [30] [78], in which residuals of the equations in least-squares form are added to the standard Galerkin formulation. GLS has been successfully employed in a wide variety of applications where enhanced stability and accuracy properties are needed, including problems governed by Navier-Stokes and the compressible Euler equations in fluid mechanics [78].
We can also mention the pressure-stabilizing/Petrov-Galerkin (PSPG) [91] formulation which has been introduced for the stabilization of the Stokes equations [44] and incompressible Navier-Stokes equations [85].

The main idea of all these methods is to transform the original Galerkin method into a Petrov-Galerkin formulation adapted to the physics considered. In fact, these formulations stabilize the method without introducing excessive numerical dissipation. Because its symptoms are not necessarily qualitative, excessive numerical dissipation is not always easy to detect. This concern makes it desirable to seek and employ stabilized formulations developed with objectives that include keeping numerical dissipation to a minimum. In these stabilized formulations, judicious selection of the stabilization parameter, which is almost known as τ, plays an important role in determining the accuracy of the formulation. This yielded a significant amount of attention and research [14] [30] [82] [83]. Typically this stabilization parameter involves a measure of the local length scale (also known as "element length") and other parameters such as the local Reynolds and Courant numbers. However, this stabilization parameter requires special attention, as it strongly depends on the problem under consideration and the chosen numerical method.

More recently, an alternative approach has emerged, the discontinuous Galerkin method (DGM), which shares some features with both the FVM and the FEM. Indeed, discontinuous polynomial functions are used and a numerical flux is defined at the interface between cells to reconstruct the solution. It has been proved very useful in solving a large range of problems. It was first introduced in 1973 by Reed and Hill for solving a time-independent linear hyperbolic equations [74] and, later on, it has been extended for solving nonlinear time-dependent equations. Subsequently, during the nineties of the last century the DGM experienced a series of developments by Cockburn and Shu, where numerical schemes for hyperbolic problems were proposed by combining DG approximation in space with Runge-Kutta time stepping strategies [18] [19] [21].
In fact, the DG method combines the advantages of stability of FVM and the accuracy of continuous FEM. Within each element, the solution is approximated by a polynomial of degree $p \geq 0$ (as in FEM), while the continuity conditions applied to the solution are relaxed at the boundaries of elements (as in FVM), however, motivated by the FVM, interface terms of the problem are approximated by a consistent, monotone and Lipschitz continuous numerical flux. This ensures that the scheme obtained is conservative, which does not hold in case of the classical FEM. In particular, the increasing interest in these kind of formulations are due to the following interesting features: they have good stability properties, they offer flexibility in the mesh construction (irregular meshes are admissible) and in the handling of boundary conditions (Dirichlet boundary conditions are weakly imposed), the accuracy is obtained by means of high-order polynomials within elements, without any regularity constraint at element interfaces. Furthermore, they are locally conservative.

The fundamental difference between the DGM and the classical FEM relies on the continuity of basis functions. In comparison with the classical FEM, in DGM the basis functions are completely discontinuous across each element interface and they consist of local piecewise polynomials. Due to the fact that of basis function have compact support, integration can be achieved locally in each element. This simplifies the implementation of the method, since the mass matrix becomes block diagonal and the solution of a large system is avoided. In addition, the discontinuity across each element allows the use of different degrees of freedom in each element independently, which is not allowed in classical finite element method. Consequently, we can easily apply adaptivity strategies by increasing the degrees of freedom near phenomena of

interest to obtain better approximations to the solution.

As explained above, FEM decomposes the computational domain into many small, "finite" elements with simple shapes. A drawback is that with such elements there is usually no continuity higher than C^0 between elements. Even with higher-order polynomials it is difficult to guarantee C^1 continuity for arbitrarily shaped elements. With the advancement in design technology a more accurate and flexible handling of the geometry becomes necessary.

The design of free-from shapes by mathematical methods is a discipline, named computer-aided-design (CAD). It had its origins slightly later than the computer-aided-engineering (CAE). In fact, CAD is the use of computer technology for design: it allows the creation, modification, analysis and optimization of drawings and geometric modeling. The Bézier curve was the first method used to construct free-form curves and surfaces, and is named according to its inventor, Dr. Pierre Bézier. Bézier was an engineer in the Renault car company and developed this method in 1966. Actually, another French engineer, Paul de Casteljau at Citroën developed the same technology some years earlier. A further development to Bézier's method were B-splines which provide more flexibility in the modeling of free-form curves and surfaces. Since 1975 non-uniform rational B-splines (NURBS) have been used in CAD programs, as a generalization of B-splines. The development of NURBS provided a technology that can exactly describe circular shapes (cylinders, spheres, etc.) which are basic elements in geometric modeling, but also allows very flexible modeling of free-form surfaces [43].

Today, there is a strong need for reducing the gap between CAD and FEM in terms of geometric representations to gain in accuracy, flexibility and ease of interaction. A new form of analysis, named isogeometric analysis (IGA), tries to close this gap between CAD and FEA in such a way that both disciplines work on the same geometric models.

IGA is an extension of the FEM for solving PDEs. It was first introduced in 2005 by Hughes, Cottrell, and Bazilevs [43], and expanded in 2006 [24] in an effort to bridge the gap between FEM and CAD. The key idea is to use for analysis the same geometry used for geometric modeling. In fact, we use the same basis functions, which are used for the representation of the geometry in computer aided design (CAD) models, also for the approximation of the solution of the PDE or the system of PDEs describing the physical phenomenon. The idea of using Bézier, B-splines or NURBS as basis functions is driven by the desire to integrate CAD within FEM, and to have a strategy to replace a huge number of little cells (the FEs) by a reduced set of larger patches covering the entire domain. Moreover, there are several advantages of this approach over the FEM: easily control of the continuity, as C^{p-1}-continuity is obtained using p-th order NURBS, exact representation of the underlying NURBS geometry on the coarsest level of discretisation, as well as exact representation of the geometry as the mesh is refined [43].

IGA has been applied to a wide variety of different physical phenomena, including computational solid dynamics problems, computational fluid dynamics [23], coupled solid–fluid interaction problems [7] and the diffusion equation [37]. In last years, there has been an increasing interest in DG-IGA for the numerical solution of elliptic PDEs [40] [54] [69][95]. The advantages of the local approximation spaces without continuity requirements that DG methods offer [5] [28] is thus employed to manage multi-patch computations.

The main purpose of this thesis is to study the use of IGA to solve some hyperbolic problems. In particular, we describe the continuous and discontinuous Galerkin method using a B-spline basis. Special emphasis is on the discontinuous Galerkin method, since it is considered as one of the most powerful and fastest growing methods with applications in various problems, not necessarily hyperbolic. The discontinuity of basis functions, which provides more flexibility in analysis, makes the method tedious however for handling realistic geometries from CAD.

The thesis is structured in four main parts: the first gives particular focus on the Bernstein and B-splines basis functions used in CAD. It is devoted to giving their definitions and basic properties. We present in the second and third parts the extension from classical analysis to IGA for the FE and DG methods, for the one-dimensional advection problem. In the last part, we deal with two dimensional hyperbolic problems by combining the IGA with the DG method. It should be mentioned, in all this work, that the discretization of equations in time is done by means of high-order explicit Runge-Kutta methods [18] [19] [34] [80].

More precisely, chapter 2 and chapter 3 provide a comprehensive introduction to the main ideas and properties of the Bernstein, B-splines and NURBS, which form the basis for the IGA. In the same context, an analysis of fitting B-spline curve and surface in the least squares sense is presented in chapter 4. The analysis is illustrated by examples of univariate and bivariate problems. Since IGA is an extension of FEM, we start by revisiting the original analysis framework in chapter 5, i.e. FEM. The need for stabilization is outlined and stabilization ideas based on the Petrov-Galerkin concept are discussed. We focus on the SUPG stabilization method and a special attention is given to the study of the stabilization parameter τ.

In chapter 6, the various computational procedures for IGA are reviewed in the context of FEM, by revisiting the one-dimensional advection problem that is given in the previous chapter. While in this chapter we use B-splines (due to the simplicity of the domain) as a basis function, it is not hard to generalize it to other splines such as NURBS. Detailed comparisons between both IGA and classical FEM are discussed.
In this context of IGA, we consider then the application of DG methods. Indeed, the major argument for using DG methods lies with their ability to provide stable numerical methods for hyperbolic PDE problems, for which classical FEM is well known to perform poorly. Therefore, in chapter 7, we deal with one-dimensional advection problem by combining IGA method with the DG method. We note that the DG methodology is adopted at patch level, i.e., we employ the classical IGA within each patch, and employ the DG method across the patch interfaces. Moreover, a transformation of the B-spline basis is necessary to introduce discontinuities at the interfaces, without modifying the geometry of the domain. The advantageous features of both IGA and DG method enable us to design a promising formulation.

With some adjustments, chapter 8 and chapter 9 are devoted to the study of two numerical examples in $2D$. The advection problem is first presented, followed by the acoustic wave equations, where both systems are solved over several domains (Cartesian, linear and curvilinear).

Finally, in chapter 10 we end with some concluding remarks and outlooks. The results of the various studies performed are summarized and discussed, and ideas for future research are proposed.

INTRODUCTION

Les systèmes d'équations aux dérivées partielles (EDPs) hyperboliques sont des modèles mathématiques permettant d'exprimer la conservation d'une quantité physique, comme par exemple la masse, l'énergie, etc. Ils découlent naturellement des lois de conservation en physique. En particulier, ils décrivent une grande variété de phénomènes impliquant le mouvement des ondes (acoustiques, élastiques, électromagnétiques) ou le transport advectif de substances. Le large éventail d'applications des EDPs hyperboliques a conduit à une activité de recherche très intense dans ce domaine. Cela a permis de développer un ensemble de méthodes numériques pour des approximations précises et efficaces du point de vue du calcul des solutions à ces problèmes. Il existe trois grandes familles de méthodes qui sont largement utilisées: la méthode des différences finies (MDF), la méthode des volumes finis (MVF) et la méthode des éléments finis (MEF). Ces méthodes se sont révélées extrêmement utiles pour modéliser un large éventail de phénomènes. Pour garder cette thèse autonome, nous présentons brièvement chacune de ces trois méthodes dans le contexte des EDPs hyperboliques.

Historiquement, la méthode des différences finies (MDF) était la première méthode utilisée pour produire des approximations des solutions des EDPs hyperboliques. Elle a été introduite par Euler au 18ème siècle et représente la méthode la plus simple pour résoudre des problèmes sur des géométries simples [64]. L'idée principale de cette méthode est de remplacer les dérivées partielles de l'inconnu par leurs approximations par des différences finis. La méthode DF est remarquable par la grande variété de schémas qui peuvent être utilisés pour approcher une EDP donnée, par exemples, schémas explicites (Forward Euler, Upwind, Lax-Friedrichs, Lax-Wendroff, Leapfrog, ...) et schémas implicites (Backward Euler, Crank-Nicolson, ...) [88]. Bien que cette méthode puisse être facilement formulée et mise en oeuvre, son application à des problèmes avec des géométries réalistes est assez lourde, ce qui rend la méthode peu attrayante pour les problèmes industriels. Cela a nécessité d'autres méthodes plus flexibles, telles que les méthodes de FV et d'EF.

La méthode des VF peut gérer des géométries complexes ce qui la rend plus attrayante que la méthode de DF pour les problèmes complexes. Elle est basée sur la forme conservative au lieu de la forme différentielle pour estimer les valeurs des champs inconnus. On divise le domaine en cellules et on approxime l'intégrale totale sur chaque cellule de la grille. Le liens entre les quantités de cellules dans la MVF dépendent du flux entre les volumes de contrôle voisins. Cela signifie que la MVF représente le flux d'informations dans la structure du maillage de manière conservative [56]. Elle permet l'utilisation de grilles non struc-

turées pour gérer des géométries complexes.

La méthode des éléments finis a ses origines au début des années 1960, elle est aujourd'hui la méthode prédominante dans l'analyse des problèmes elliptiques ou paraboliques en raison de sa flexibilité à représenter des domaines géométriques complexes et sa base théorique solide. L'idée consiste à décomposer le domaine en plusieurs éléments «finis», définis par un ensemble de points nodaux et des fonctions de base interpolantes.

Bien que la MEF a été largement utilisée pour simuler de nombreux phénomènes physiques en raison de sa flexibilité à représenter des domaines géométriques complexes, il est bien connu dans la littérature de la MEF que des difficultés numériques se posent lors de la résolution des EDPs hyperboliques. En effet, lors de l'utilisation de la méthode d'EF standard appliquée à de tels EDPs, des oscillations parasites (non physiques) indésirables (phénomène de Gibbs) sont fréquemment détectées dans les solutions numériques. Un remède à cet inconvénient, répandu dans la littérature, consiste à ajouter une viscosité "artificielle" à un schéma numérique standard (instable). D'une part, cette viscosité artificielle doit amortir les oscillations mais, d'autre part, elle ne doit pas entacher la solution numérique. À la fin des années 1970 et au début des années 1980, un grand nombre de méthodes dites stabilisées ont été développées avec des idées différentes [14] [32] [48]. Ceci est réalisé grâce à l'utilisation d'une formulation Petrov-Galerkin [35], où les fonctions test sont modifiées de telle sorte qu'elles pondèrent le noeud en amont plus que le noeud en aval [17] [38]. Parmi eux, la plus populaire, dite Streamline-Upwind Petrov-Galerkin (SUPG), a été introduite par Brooks et Hughes. Elle a été proposée d'abord dans le contexte des équations d'advection–diffusion et les équations de Navier–Stokes incompressible [14], puis elle a été étendue à divers autres problèmes, par exemples les systèmes advectifs–diffusifs multidimensionnels couplés [44], les systèmes hyperboliques linéaires du premier ordre [49], les systèmes hyperboliques de lois de conservation [45].

Plus tard, la méthode de Galerkin/moindres carrés (GLS) est apparue comme une généralisation de la méthode SUPG, développée par Hughes et al. pour les problèmes de transport convectif [9] [30] [78], dans laquelle les résidus des équations des moindres carrés sont ajoutés à la formulation de Galerkin standard. La méthode GLS a été utilisée avec succès dans une large variété d'applications où des propriétés de stabilité et de précision améliorées sont nécessaires, y compris les problèmes régis par Navier-Stokes et les équations d'Euler compressible en mécanique des fluides [78]. On peut également citer la formulation de stabilisation de la pression/Petrov-Galerkin (PSPG) [91] introduite pour la stabilisation des équations de Stokes [44] et des équations de Navier-Stokes incompressibles [85].

L'idée principale de toutes ces méthodes est de transformer la méthode originale de Galerkin en une formulation de Petrov-Galerkin adaptée à la physique considérée. En fait, ces formulations stabilisent le procédé sans introduire de dissipation numérique excessive. Parce que les symptômes ne sont pas nécessairement qualitatifs, une dissipation numérique excessive n'est pas toujours facile à détecter. Cette préoccupation rend la recherche et l'emploi de formulations stabilisées développées souhaitable avec des objectifs tels que le maintien au minimum de la dissipation numérique. Dans ces formulations stabilisées, la sélection du paramètre de stabilisation, qui est souvent connu sous le nom de τ, joue un rôle important dans la détermination de la précision de la formulation. Cela a suscité beaucoup d'attention et de recherche [14] [30] [82] [83]. Ce paramètre de stabilisation implique généralement une mesure de l'échelle de longueur locale (également appelée «longueur d'élément») et d'autres paramètres tels que les nombres locaux de Reynolds et de Courant. Cependant, ce paramètre de stabilisation nécessite une attention particulière, car il dépend fortement du problème considéré et de la méthode numérique stabilisée choisie.

Plus récemment, une approche alternative a émergé, la méthode de Galerkin discontinue (MGD), qui partage certaines fonctionnalités avec MVF et MEF. En effet, des fonctions polynomiales discontinues sont utilisées et un flux numérique est défini à l'interface entre les cellules pour reconstruire la solution. Elle s'est avérée très efficace pour résoudre un large éventail de problèmes. Elle a été introduite pour la première fois en 1973 par Reed et Hill pour résoudre des équations hyperboliques linéaires indépendantes du temps [74] et plus tard, elle a été étendue pour résoudre des équations non linéaires dépendant du temps. Par la suite, au cours des années quatre-vingt-dix du siècle dérnier, Cockburn et Shu ont développé une série de schémas numériques pour les problèmes hyperboliques en combinant l'approximation de GD dans l'espace et les stratégies de Runge-Kutta pour la discrétisation temporelle [18] [19] [21].
En fait, la méthode de GD combine les avantages de la stabilité de la MVF et la précision de la MEF continue. Dans chaque élément, la solution est approchée par un polynôme de degré $p \geq 0$ (comme dans la MEF), tandis que les conditions de continuité appliquées à la solution sont relâchées aux limites des éléments (comme dans la MVF), mais comme pour la MVF, les termes d'interface du problème sont approximés par un flux numérique continu, monotone et Lipschitz. Cela garantit que le schéma obtenu est conservatif, ce qui n'est pas le cas de la MEF classique. En particulier, l'intérêt croissant pour ce type de formulations est dû aux caractéristiques intéressantes suivantes: elles ont de bonnes propriétés de stabilité, elles offrent une flexibilité dans la construction du maillage (les maillages non structurés sont admissibles) et dans les conditions aux limites (les conditions aux limites de Dirichlet sont faiblement imposées), la précision est obtenue au moyen de polynômes d'ordre élevé dans les éléments, sans aucune contrainte de régularité aux interfaces d'éléments.

La différence fondamentale entre la MGD et la MEF classique repose sur la continuité des fonctions de base. En comparaison avec la MEF classique, dans la MGD, les fonctions de base sont complètement discontinues pour chaque interface d'élément et elles sont constituées de polynômes locaux par élément. Grâce au fait que des fonctions de base ont un support compact, l'intégration peut être réalisée localement dans chaque élément. Cela simplifie la mise en oeuvre de la méthode, puisque la matrice de masse devient diagonale en bloc et que la résolution d'un grand système est évitée.

De plus, la discontinuité entre chaque élément permet d'utiliser différents degrés de liberté dans chaque élément indépendamment, ce qui n'est pas autorisé dans la méthode des EF classiques. Par conséquent, nous pouvons facilement appliquer des stratégies d'adaptation en augmentant les degrés de liberté à proximité de phénomènes d'intérêt pour obtenir de meilleures approximations de la solution.

Comme expliqué ci-dessus, la MEF décompose le domaine en plusieurs éléments «finis» avec des formes simples. Un inconvénient est qu'avec de tels éléments, il n'y a généralement pas de continuité supérieure à C^0 entre les éléments. Même avec des polynômes d'ordre supérieur, il est difficile de garantir la continuité C^1 pour des éléments de forme arbitraire. Avec les progrès de la technologie de conception, une manipulation plus précise et flexible devient nécessaire.

La conception de les formes libres par des méthodes mathématiques est une discipline appelée conception assistée par ordinateur (CAO). Ses origines étaient un peu plus tardives que l'ingénierie assistée par ordinateur (IAO). En fait, la CAO est l'utilisation de l'informatique pour la conception: elle permet la création, la modification, l'analyse et l'optimisation de dessins et modélisations géométriques. Les courbes de Bézier ont été la première méthode utilisée pour construire une forme libre des courbes et des surfaces, elles sont nommées d'après leur inventeur Pierre Bézier. Bézier était ingénieur dans la société Renault automobile et a développé cette méthode en 1966 [**?**]. En fait, un autre ingénieur français, Paul de Casteljau chez Citroën a développé la même technologie quelques années auparavant. Un autre développement de la méthode de Bézier concerne les B-splines qui offrent plus de flexibilité dans la modélisation des courbes et des surfaces de forme libre. Depuis 1975, les B-splines rationnelles non uniformes (NURBS) ont été utilisées dans les programmes de CAO comme une généralisation de B-splines. Le développement de NURBS a fourni une technologie capable de décrire exactement les formes circulaires (cylindres, sphères, etc.) qui sont des éléments de base de la modélisation géométrique, mais permet également une modélisation très flexible des surfaces à forme libre [43].

Aujourd'hui, il existe un fort besoin de réduire l'écart entre la CAO et la MEF en termes de représentations géométriques pour gagner en flexibilité, précision et en facilité d'interaction. Une nouvelle forme d'analyse, appelée analyse isogéométrique (AIG), tente de combler cet écart entre la CAO et la MEF de telle manière que les deux disciplines fonctionnent avec les mêmes modèles géométriques.
L'AIG est une extension de la MEF pour la résolution des EDPs. Elle a été introduite pour la première fois en 2005 par Hughes, Cottrell et Bazilevs [43], et développé dans 2006 [24] pour tenter de combler le fossé entre la MEF et la CAO.

L'idée principale est d'utiliser pour l'analyse la même géométrie utilisée pour la modélisation géométrique. En fait, on utilise les mêmes fonctions de base, qui sont utilisées pour la représentation de la géométrie dans les modèles de conception assistée par ordinateur (CAO), également pour l'approximation de la solution des EDPs ou des systèmes des EDPs décrivant le phénomène physique. L'idée d'utiliser Bézier, B-splines ou NURBS comme fonctions de base est motivée par le désir d'intégrer la CAO dans la MEF, et d'avoir une stratégie pour remplacer un grand nombre de petites cellules (les EF) par un ensemble réduit de plus gros patches couvrant tout le domaine. De plus, cette approche présente plusieurs avantages par rapport à la MEF: contrôle facile de la continuité, car la continuité C^{p-1} est obtenue en utilisant une NURBS de degré p, représentation exacte de la géométrie en utilisant une NURBS au niveau de discrétisation le plus grossier, ainsi que la représentation exacte de la géométrie lorsque le maillage est affiné [43].

L'AIG a été appliquée à une grande variété de phénomènes physiques, y compris la dynamique des fluides computationnelle [23], les problèmes d'interaction couplé fluide-structure [7] et l'équation de diffusion [37]. Au cours des dernières années, la méthode de GD dans le cadre IG a manifesté un intérêt croissant pour la solution numérique des EDPs elliptiques [40] [54] [69][95]. Les avantages des espaces d'approximation locaux sans exigences de continuité offerts par les méthodes de GD [5] [28] sont alors utilisés pour gérer les calculs multi-patch.

L'objectif principal de cette thèse est d'étudier l'utilisation des AIG pour résoudre certains problèmes hyperboliques. En particulier, nous décrivons la méthode de Galerkin continue et discontinue en utilisant la base des B-splines. Un accent particulier est mis sur la méthode de GD, car elle est considérée comme l'une des méthodes les plus efficaces et à la croissance la plus rapide, avec des applications dans divers problèmes, pas nécessairement hyperboliques. Les discontinuités des fonctions de base, qui offrent une plus grande souplesse d'analyse, rendent cependant la méthode délicate pour gérer des géométries réalistes à partir de la CAO.

La thèse est divisée en quatre parties principales: la première met l'accent sur les fonctions de base de Bernstein et B-splines utilisées en CAO. Elle est consacré à donner leurs définitions et propriétés de base. Nous présenterons dans la deuxième et troisième parties l'extension de l'analyse classique à l'AIG pour les méthodes d'EF et GD, pour le problème d'advection unidimensionnel. Dans la dernière partie, nous traitons des problèmes hyperboliques en deux dimensions en combinant l'AIG avec la méthode de GD. Il convient de mentionner que dans tout ce travail, la discrétisation des équations dans le temps se fait au moyen de méthodes de Runge-Kutta explicites d'ordre élevé [18] [19] [34] [80].

Plus précisément, les deuxième et troisième chapitres fournissent une introduction complète aux principales idées et propriétés des fonctions Bernstein, B-splines et NURBS, qui constituent la base de l'AIG. Dans le même contexte, la construction des courbes B-spline et des surfaces au sens des moindres carrés est présentée au quatrième chapitre. L'analyse est accompagnée d'exemples de problèmes univariés et bivariés.

L'AIG étant une extension de la MEF, nous commençons par revoir le cadre de l'analyse originale au cinquième chapitre, à savoir: la MEF. Le besoin de stabilisation est souligné et les idées de stabilisation basées sur le concept Petrov-Galerkin sont discutés. Nous nous concentrons sur la méthode de stabilisation SUPG et une attention particulière est accordée à l'étude du paramètre de stabilisation τ.

Au niveau du sixième chapitre, les différentes procédures de calcul pour l'AIG sont passées en revue dans le contexte de la MEF, en revisitant le problème d'advection unidimensionnel qui est présenté dans le chapitre précédent. Dans ce chapitre nous utilisons les B-splines (en raison de la simplicité du domaine) comme fonctions de base, il n'est pas difficile de le généraliser pour d'autres splines telles que les NURBS. Des comparaisons détaillées entre l'AIG et la MEF classiques sont discutées. Dans ce contexte de l'AIG, nous considérons alors l'application des méthodes GD. En effet, l'argument majeur pour utiliser les méthodes de GD réside dans leur capacité à fournir des méthodes numériques stables pour les EDPs hyperboliques, pour lesquelles la MEF classique est bien connue pour ses performances médiocres. Par conséquent, au septième chapitre, nous traitons le problème d'advection unidimensionnelle en combinant la méthode de l'AIG avec la méthode de GD. Nous notons que la méthodologie de GD est adoptée au niveau du patch, c'est-à-dire que nous employons l'AIG classique dans chaque patch et utilisons la méthode de GD à travers les interfaces de patch. De plus, une transformation de la base B-spline est nécessaire pour introduire des discontinuités aux interfaces, sans modifier la géométrie du domaine. Les caractéristiques avantageuses des deux méthodes AIG et GD nous permettent de concevoir une formulation prometteuse.

Avec quelques ajustements, les huitième et neuvième chapitres sont consacrés à l'étude de deux exemples numériques en $2D$, le problème d'advection est d'abord présenté, suivi par les équations d'ondes acoustiques, où les deux systèmes sont résolus sur plusieurs domaines (cartésien, linéaire et curviligne).

Enfin, au niveau du dernier chapitre, nous terminons avec quelques remarques et perspectives finales. Les résultats des différentes études réalisées sont résumées et discutées et des idées de recherches futures sont proposées.

Part I

CAD REPRESENTATIONS

CHAPTER 2

BÉZIER CURVES

Bézier curves are parametric curves commonly used in computer graphics and related fields. They are named after their inventor, Dr. Pierre Bézier, an engineer from the Renault car company who developed in the early $1960's$ a curve formulation for use in shape design. The main interest of Bernstein-Bézier patches is that they lend to an easy geometric understanding of the underlying mathematical concepts. Some basic properties and a brief discussion of Bernstein polynomials and Bézier curves [8] are presented in the present chapter.

2.1 Bernstein basis

Bézier curves are expressed in terms of Bernstein polynomials which where introduced by Sergei Bernstein in order to formulate a constructive proof of the Weierstrass approximation theorem.

Definition 2.1.1. *(Univariate Bernstein)*
The Bernstein polynomials of degree p over the interval $[0,1]$ *are defined explicitly by:*

$$B_p^k(\zeta) = C_p^k \zeta^k (1-\zeta)^{p-k} \qquad \forall \quad k = 0, ..., p,$$

with the binomial coefficients C_p^k *given by:*

$$C_p^k = \begin{cases} \frac{p!}{k!(p-k)!} & \text{if} \quad 0 \le k \le p, \\ 0 & \text{otherwise.} \end{cases}$$

An example of constant, linear, quadratic and cubic Bernstein polynomials are presented in Fig. 2.1:

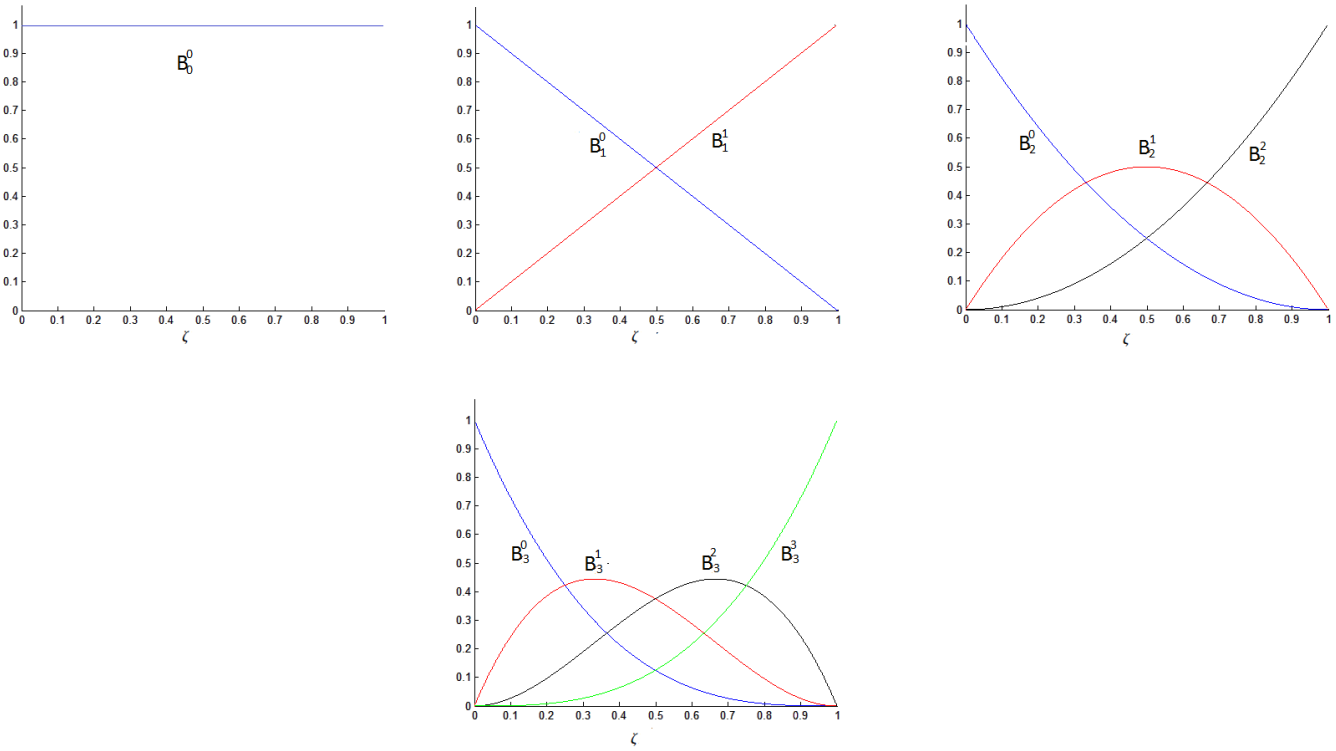


Figure 2.1: Constant, linear, quadratic and cubic Bernstein polynomials.

Definition 2.1.2. *(Multivariate Bernstein)*

In order to define Bernstein basis in higher dimensions, we make use of the tensor product construction. Let $p = (p_1, p_2, ..., p_d)$ *be a vector in* $\mathbb{N}^d$*. The* d*-dimensional Bernstein polynomials are defined by a tensor product of* d *univariate Bernstein polynomials with possibly different degrees* $p = (p_1, p_2, ..., p_d)$ *and multi-indices* $k = (k_1, k_2, ..., k_d)$.

Therefore, $\forall \zeta = (\zeta_1, \zeta_2, ..., \zeta_d) \in [0,1]^d$ *we get:*

$$B_p^k(\zeta) = B_{p_1}^{k_1}(\zeta_1) \otimes B_{p_2}^{k_2}(\zeta_2) \otimes ... \otimes B_{p_d}^{k_d}(\zeta_d).$$

The bivariate Bernstein polynomials are illustrated in Fig. 2.2 for the linear, quadratic and cubic cases.

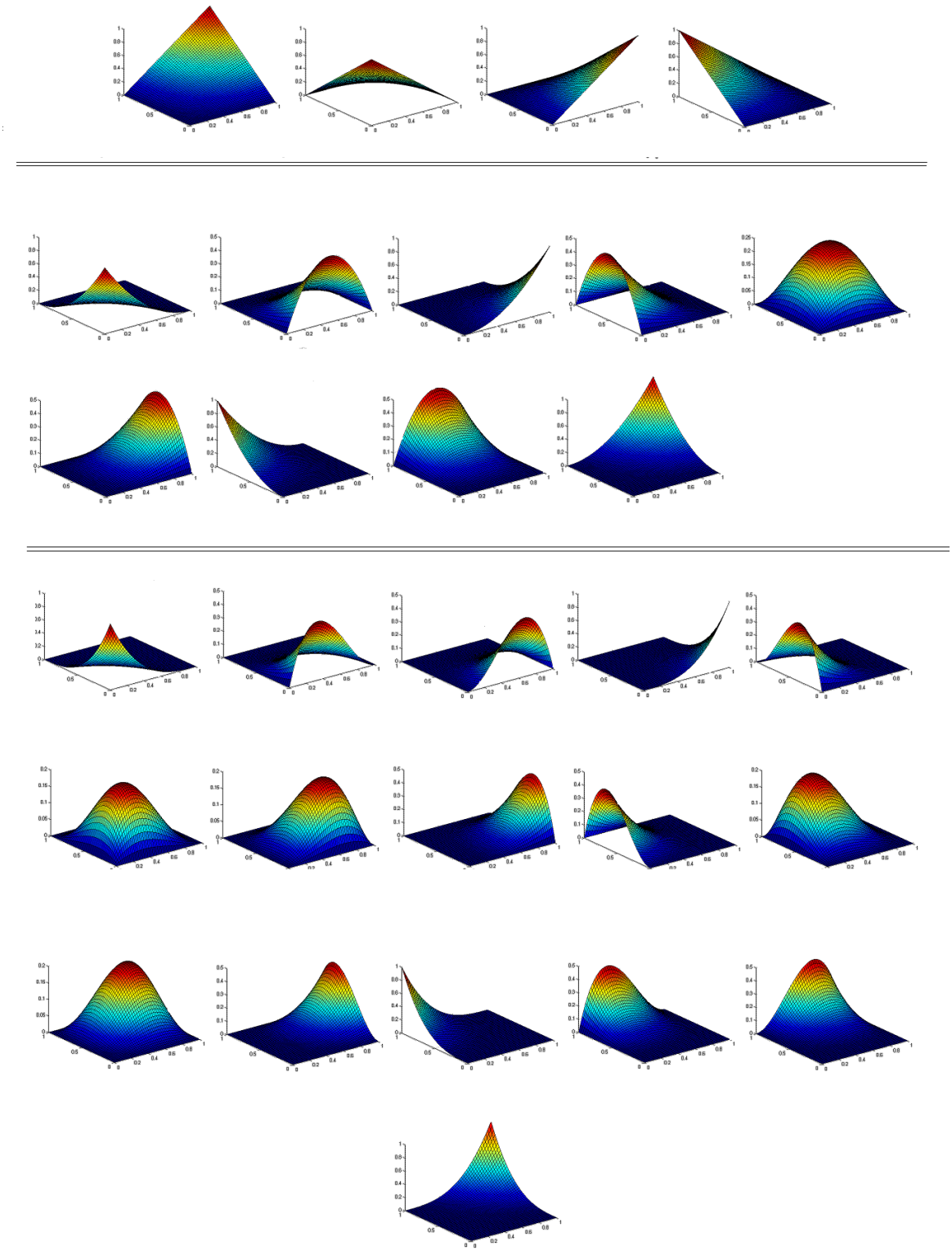


Figure 2.2: Linear, quadratic and cubic bivariate Bernstein polynomials.

2.2 Properties of the Bernstein polynomials

The Bernstein polynomials B_p^k of degree p, have several important properties [50]:

1. **Recursion**

   The Bernstein polynomials satisfy the following recursion relation: for $p > 0$, we have:

$$\begin{cases} B_p^0(\zeta) = (1-\zeta)^p & \text{if} \quad k = 0, \\ B_p^k(\zeta) = (1-\zeta)B_{p-1}^k(\zeta) + \zeta B_{p-1}^{k-1}(\zeta) & \text{if} \quad k = 1, ..., p-1, \\ B_p^p(\zeta) = \zeta^p & \text{if} \quad k = p. \end{cases}$$

   Proof.

   $\forall 1 \leq k \leq p-1$, we have

$$\begin{aligned} (1-\zeta)B_{p-1}^k(\zeta) &= (1-\zeta)C_{p-1}^k \zeta^k (1-\zeta)^{(p-1)-k} \\ &= C_{p-1}^k \zeta^k (1-\zeta)^{p-k} \\ &= \frac{(p-1)!}{k!\Big((p-1)-k\Big)!}\zeta^k(1-\zeta)^{p-k} \\ &= \Big(\frac{p-k}{p}\Big)B_p^k(\zeta). \end{aligned}$$

   Similarly,

$$\begin{aligned} \xi B_{p-1}^{k-1}(\zeta) &= \zeta C_{p-1}^{k-1}\zeta^{k-1}(1-\zeta)^{(p-1)-(k-1)} \\ &= C_{p-1}^{k-1}\zeta^k(1-\zeta)^{p-k} \\ &= \frac{(p-1)!}{(k-1)!\Big((p-1)-(k-1)\Big)!}\zeta^k(1-\zeta)^{p-k} \\ &= (\frac{k}{p})B_p^k(\zeta). \end{aligned}$$

   Therefore,

$$\begin{aligned} (1-\zeta)B_{p-1}^k(\zeta) + \zeta B_{p-1}^{k-1}(\zeta) &= (\frac{p-k}{p})B_p^k(\zeta) + (\frac{k}{p})B_p^k(\zeta) \\ &= B_p^k(\zeta). \end{aligned}$$

   ■

2. **Non-negativity**

   The Bernstein polynomials are positive everywhere in $[0,1]$.

$$B_p^k(\zeta) \geq 0 \qquad \forall \zeta \in [0,1] \quad 0 \leq k \leq p.$$

   Note that the Lagrange interpolating polynomials, commonly used as basis for the numerical solution of partial differential equation (PDE), do not satisfy this property.

3. **Partition of unity**

   The Bernstein polynomials of degree p forms a partition of unity, that is:

$$\sum_{k=0}^{p} B_p^k(\zeta) = 1 \qquad \forall \zeta \in [0,1].$$

   Proof.

$$\sum_{k=0}^{p} B_p^k(\zeta) = \sum_{k=0}^{p} C_p^k \zeta^k \big(1-\zeta\big)^{p-k} = \Big(\xi + (1-\xi)\Big)^p = 1.$$

   ■

4. **Unique maximum**

   The Bernstein polynomial B_p^k has a unique maximum at $\zeta = \frac{k}{p}$ on $[0,1]$.

5. **Symmetry**

   The Bernstein polynomials are symmetric in the sens:

$$B_p^k(1-\zeta) = B_p^{p-k}(\zeta) \qquad \forall 0 \leq k \leq p.$$

6. **Basis for polynomials of degree less or equal p**

   The Bernstein polynomials of degree p form a basis for the space of polynomials of degree less than or equal to p.

7. **Degree elevation**

   Any Bernstein polynomial of degree less than p can be expressed as a linear combination of Bernstein polynomials of degree p. In particular, any Bernstein polynomial of degree $p-1$ can be written as a linear combination of Bernstein polynomials of degree p.

$$B_{p-1}^k(\zeta) = \left(\frac{p-k}{p}\right) B_p^k(\zeta) + \left(\frac{k+1}{p}\right) B_p^{k+1}(\zeta).$$

Proof.

$$\begin{aligned}
\left(\frac{p-k}{p}\right)B_p^k(\zeta)+\left(\frac{k+1}{p}\right)B_p^{k+1}(\zeta) &= \left(\frac{p-k}{p}\right)C_p^k\zeta^k(1-\zeta)^{p-k}+\left(\frac{k+1}{p}\right)C_p^{k+1}\zeta^{k+1}(1-\zeta)^{p-(k+1)} \\
&= \left(C_p^k\zeta^k(1-\zeta)^{p-k}\right)\left(\frac{p-k}{p}+\left(\frac{k+1}{p}\right)\left(\frac{p-k}{k+1}\right)\left(\frac{\zeta}{1-\zeta}\right)\right) \\
&= \left(C_p^k\zeta^k(1-\zeta)^{p-k}\right)\left(\frac{p-k}{p}\right)\left(\frac{1}{1-\zeta}\right) \\
&= \frac{(p-1)!}{k!(p-k-1)!}\zeta^k(1-\zeta)^{(p-1)-k} \\
&= B_{p-1}^k(\zeta).
\end{aligned}$$

■

This property will play an important role, in the context of numerical solution of PDE, by allowing p-refinement process.

2.3 Derivatives

The derivative of the k-th Bernstein polynomial of degree p is given by:

$$\frac{d}{d\zeta}B_p^k(\zeta) = p\Big(B_{p-1}^{k-1}(\zeta)-B_{p-1}^k(\zeta)\Big).$$

Proof.

$$\begin{aligned}
\frac{d}{d\zeta}B_p^k(\zeta) &= \frac{d}{d\zeta}\left(C_p^k\zeta^k(1-\zeta)^{p-k}\right) \quad = C_p^k\Big(k\zeta^{k-1}(1-\zeta)^{p-k}-(p-k)\zeta^k(1-\zeta)^{p-k-1}\Big) \\
&= \frac{k}{\zeta}\Big(C_p^k\zeta^k(1-\zeta)^{p-k}\Big)-\Big(\frac{p-k}{1-\zeta}\Big)\Big(C_p^k\zeta^k(1-\zeta)^{p-k}\Big) \\
&= \frac{k}{\zeta}\Big(\frac{p}{k}C_{p-1}^{k-1}\zeta^k(1-\zeta)^{p-k}\Big)-\Big(\frac{p-k}{1-\zeta}\Big)\Big(C_p^k\zeta^k(1-\zeta)^{p-k}\Big) \\
&= \Big(pC_{p-1}^{k-1}\zeta^{k-1}(1-\zeta)^{p-k}\Big)-\Big((p-k)C_p^k\zeta^k(1-\zeta)^{p-k-1}\Big) \\
&= \Big(pB_{p-1}^{k-1}(\zeta)\Big)-\Big((p-k)\frac{p!}{k!(p-k)!}\zeta^k(1-\zeta)^{p-k-1}\Big) \\
&= \Big(pB_{p-1}^{k-1}(\zeta)\Big)-\Big(p\frac{(p-1)!}{k!(p-k-1)!}\zeta^k(1-\zeta)^{p-k-1}\Big) \\
&= pB_{p-1}^{k-1}(\zeta)-pB_{p-1}^k(\zeta).
\end{aligned}$$

■

This formula is used to evaluate the derivatives of basis functions in the variational formulation, in a recursive way.

2.4 Bézier curves

Definition 2.4.1. *Given $(p+1)$ distinct points $P_0, P_1, ..., P_p$ in space, the Bézier curve (of degree p) defined from these points is the parametric curve C_p defined by:*

$$C_p(\zeta) = \sum_{k=0}^{p} B_p^k(\zeta) P_k \qquad \forall \zeta \in [0,1].$$

The points $(P_i)_{i=0,...,p}$ are called the control points and the line segments P_0P_1, P_1P_2, ..., $P_{p-1}P_p$ form in this order, the control polygon.

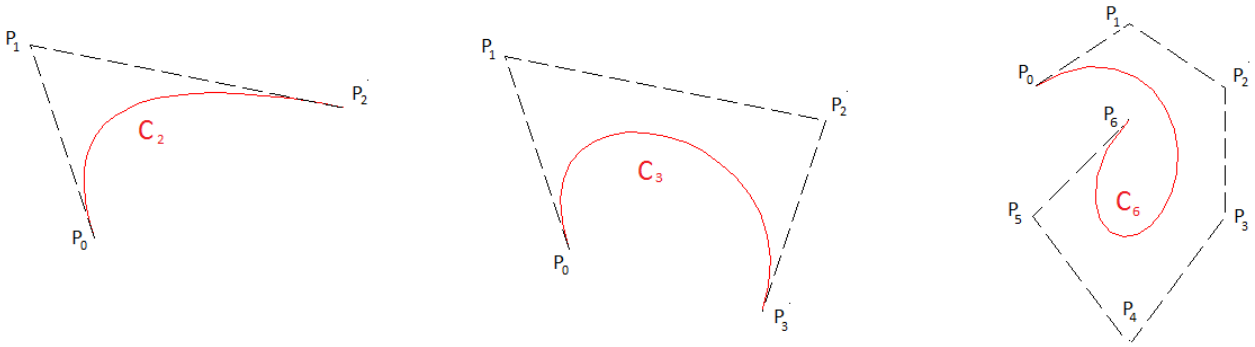


Figure 2.3: Bézier curves of various degrees and their control polygons.

Figure 2.3 depicts three different Bézier curves, with their corresponding control polygons. Each control polygon is composed of its control points that are connected with line segments. Note that these control polygons are not necessary closed.

2.5 Properties

Let us recall some important properties of a Bézier curve [46] [60], that will be useful for the numerical solving PDE systems.

1. **Interpolation at the extremities**

   A Bézier curve C_p of degree p, always starts at the first control point P_0 and ends at last control point P_p.

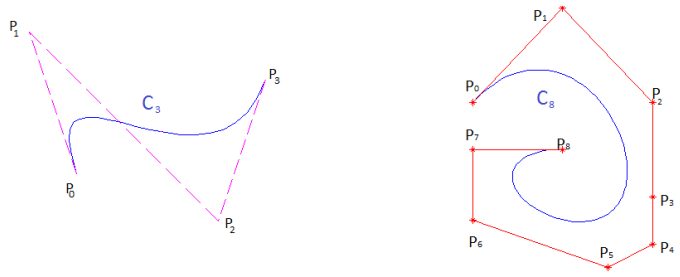


Figure 2.4: Bézier curves with endpoint interpolation.

This property is very important because, even if it is not necessary to fully control the curve in the middle, it is essential to know where it starts and where it ends. If we want to connect several Bézier curves, it is mandatory to know where the ends are. Moreover, this is important for applying Dirichlet boundary conditions when solving PDE systems.

2. **Invariance under affine transformation**

   An affine transformation of a Bézier curve is obtained by applying the transformation to the control points.

   Proof. Let ψ be an affine transformation in $\mathbb{R}^p$:

$$\psi(X) = AX + b \qquad \text{with} \quad A \in \mathbb{M}_p(\mathbb{R}) \quad \text{and} \quad b \in \mathbb{R}^p.$$

   The affine transformation of a Bézier curve C_p of degree p is:

$$\begin{aligned}
\psi(C_p(\zeta)) &= \psi\Big(\sum_{k=0}^{p} B_p^k(\zeta) P_k\Big) \\
&= A\Big(\sum_{k=0}^{p} B_p^k(\zeta) P_k\Big) + b \\
&= \sum_{k=0}^{p} A B_p^k(\zeta) P_k + \sum_{k=0}^{p} b B_p^k(\zeta) \\
&= \sum_{k=0}^{p} \Big(AP_k + b\Big) B_p^k(\zeta) \\
&= \sum_{k=0}^{p} \psi(P_k) B_p^k(\zeta).
\end{aligned}$$

   So $\psi(C_p(\zeta))$ is a Bézier curve. ■

3. **Convex hull**

   An important property of Bézier curves is that they always lies within the convex hull of their control points. To explain this property we need to define the convex hull of a set of points.

Definition 2.5.1. *(Convex hull)*
The convex hull of a set $\mathscr{P} = \{P_0, P_1, ..., P_p\}$ of control points is the smallest convex polygon that contains all the control points of $\mathscr{P}$.

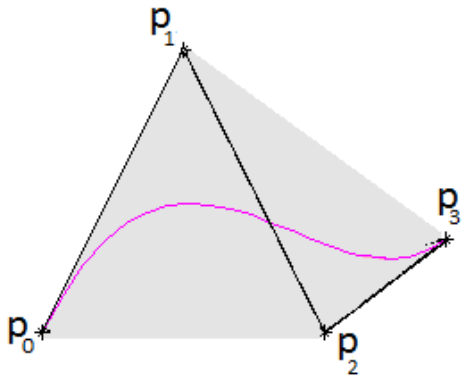


Figure 2.5: Cubic Bézier curve, its control polygon and the convex hull.

In Fig. 2.5, the convex hull of the 4 control points is shown. Again, this property can be exploited when solving PDE, in particular in case of discontinuity capturing.

4. **Local control**
   A Bézier curve is not interpolating the control points and moreover a Bézier curve changes globally when a control point is modified.

An example for a quadric Bézier curve is presented in Fig. 2.6. The original curve is shown on the left. The new Bézier curve after repositioning of the control point P_2 is shown on the right. As we can see, the curve is globally modified. This property can be a drawback and will justify the construction of B-spline curves (see chapter 3).

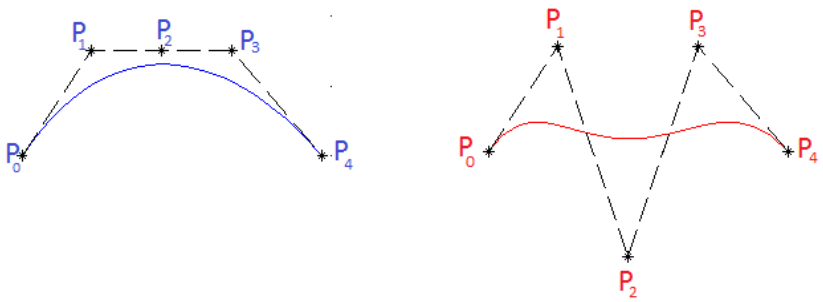


Figure 2.6: Quadric Bézier curve and repositioning of the control point P_2.

2.5.1 Degree elevation

The degree p of a Bézier curve C_p may be increased without changing the curve geometrically or parametrically. The new $p+2$ control points $\{\overline{P}_0, \overline{P}_1, ..., \overline{P}_p, \overline{P}_{p+1}\}$ are formed from the original $p+1$ control points $\{P_0, P_1, ..., P_p\}$ by:

$$\overline{P}_k = \big(\frac{k}{p+1}\big)P_{k-1} + \big(\frac{p+1-k}{p+1}\big)P_k \qquad \forall \quad 0 \le k \le p+1.$$

Proof.

$$\begin{aligned}
C_p(\zeta) &= \Big(\zeta + \big(1-\zeta\big)\Big)C_p(\zeta) \\
&= \zeta C_p(\zeta) + \big(1-\zeta\big)C_p(\zeta) \\
&= \zeta \sum_{k=0}^{p} B_p^k P_k + \big(1-\zeta\big) \sum_{k=0}^{p} B_p^k P_k \\
&= \sum_{k=0}^{p} \Big(\big(\frac{k+1}{p+1}\big)B_{p+1}^{k+1}(\zeta)P_k\Big) + \sum_{k=0}^{p} \Big(\big(\frac{p+1-k}{p+1}\big)B_{p+1}^{k}(\zeta)P_k\Big) \quad \text{(see page 13)} \\
&= \sum_{k=1}^{p+1} \Big(\big(\frac{k}{p+1}\big)B_{p+1}^{k}(\zeta)P_{k-1}\Big) + \sum_{k=0}^{p} \Big(\big(\frac{p+1-k}{p+1}\big)B_{p+1}^{k}(\zeta)P_k\Big) \\
&= \sum_{k=0}^{p+1} \Big(\big(\frac{k}{p+1}\big)B_{p+1}^{k}(\zeta)P_{k-1}\Big) + \sum_{k=0}^{p+1} \Big(\big(\frac{p+1-k}{p+1}\big)B_{p+1}^{k}(\zeta)P_k\Big) \\
&= \sum_{k=0}^{p+1} \Big(\big(\frac{k}{p+1}\big)P_{k-1} + \big(\frac{p+1-k}{p+1}\big)P_k\Big)B_{p+1}^{k}(\zeta). \qquad \blacksquare
\end{aligned}$$

This process can be used to increase the order of the solution, in the context of PDE solving, without modifying the solution itself.

An example of degree elevation is depicted in Fig. 2.7. The original quadratic Bézier curve C_2 is shown in the beginning. The numbers of control points and basis functions increase simultaneously. The locations of the control points change, whereas the elevated curve is geometrically and parametrically identical to the first one.

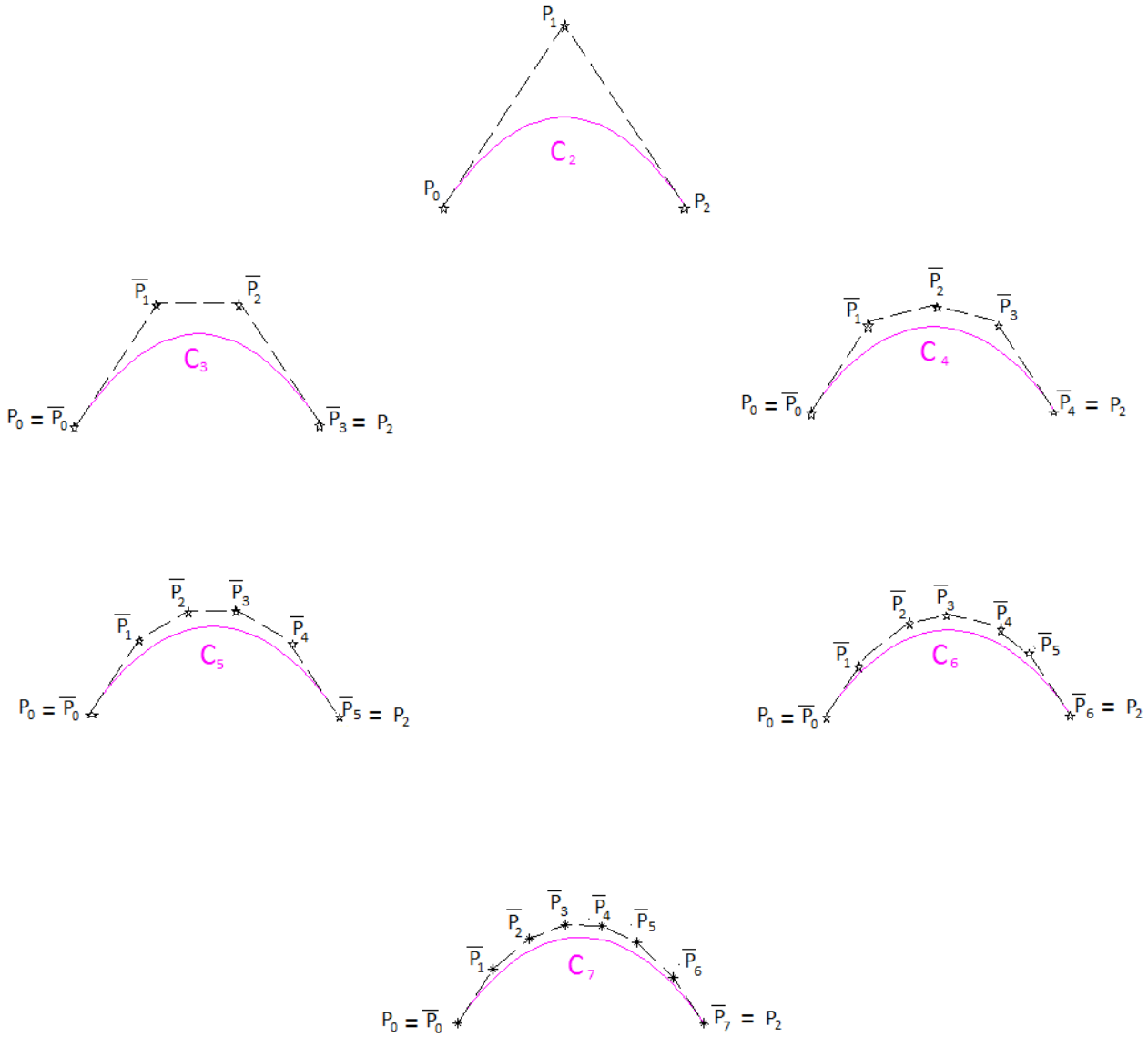


Figure 2.7: Degree elevation of a quadratic Bézier Curve.

2.5.2 Derivatives of a Bézier Curve

The derivative of a Bézier curve of degree p, is another Bézier curve of degree $p-1$ given by:

$$\frac{d}{d\zeta}C_p(\zeta) = p\sum_{k=0}^{p-1} B_{p-1}^k(\zeta)\big(P_{k+1}-P_k\big). \tag{2.1}$$

More generally, derivatives of higher order are given by:

$$\frac{d^r}{d\zeta^r}C_p(\zeta) = \frac{p!}{(p-r)!}\sum_{k=0}^{p-r} B_{p-r}^k(\zeta)\big(P_{k+1}-P_k\big)^r \qquad \forall r \le p. \tag{2.2}$$

This property has important consequences. In particular the curve at extremities is tangent to the first and last control points lines. This could be used to apply Neumann boundary conditions for instance.

Proof.

For the first derivative, we have:

$$\begin{aligned}
\frac{d}{d\zeta}C_p(\zeta) &= \frac{d}{d\zeta}\Big(\sum_{k=0}^{p} B_p^k(\zeta)P_k\Big) \\
&= \sum_{k=0}^{p}\Big(\frac{dB_p^k(\zeta)}{d\zeta}\Big)P_k \\
&= \sum_{k=0}^{p}\Big(p(B_{p-1}^{k-1}(\zeta)-B_{p-1}^{k}(\zeta))\Big)P_k \\
&= \Big(p\sum_{k=0}^{p} B_{p-1}^{k-1}(\zeta)P_k\Big)-\Big(p\sum_{k=0}^{p} B_{p-1}^{k}(\zeta)P_k\Big) \qquad \Big(\text{note that} \quad B_{p-1}^{-1}(\zeta)=B_{p-1}^{p}(\zeta)=0\Big) \\
&= \Big(p\sum_{k=0}^{p-1} B_{p-1}^{k}(\zeta)P_{k+1}\Big)-\Big(p\sum_{k=0}^{p-1} B_{p-1}^{k}(\zeta)P_k\Big) \\
&= p\sum_{k=0}^{p-1} B_{p-1}^{k}(\zeta)\big(P_{k+1}-P_k\big).
\end{aligned}$$

This can be written as:

$$\begin{aligned}
\frac{d}{d\zeta}C_p(\xi) &= p\sum_{k=0}^{p-1} B_{p-1}^{k}(\zeta)P_k \\
&= \frac{p!}{(p-1)!}\sum_{k=0}^{p-1} B_{p-1}^{k}(\zeta)P_k.
\end{aligned}$$

Now assume that (2.2) is true up to the order r. Let us prove it for $r+1$:

$$\begin{aligned}
\frac{d^{r+1}}{d\zeta^{r+1}}C_p(\zeta) &= \frac{d}{d\zeta}\Big(\frac{d^r}{d\zeta^r}C_p\Big) \\
&= \frac{d}{d\zeta}\Big(\frac{p!}{(p-r)!}\Big(\sum_{k=0}^{p-r} B_{p-r}^{k}(\zeta)(P_{k+1}-P_k)^r\Big)\Big) \\
&= \frac{p!}{(p-r)!}\Big(\sum_{k=0}^{p-r}\frac{d}{d\zeta}B_{p-r}^{k}(\zeta)(P_{k+1}-P_k)^r\Big) \\
&= \frac{p!}{(p-r)!}\Big(\sum_{k=0}^{p-r}\Big((p-r)(B_{p-r-1}^{k-1}(\zeta)-B_{p-r-1}^{k}(\zeta))\Big)\Big(P_{k+1}-P_k\Big)^{r+1}\Big) \\
&= \frac{p!}{(p-r-1)!}\Big(\sum_{k=0}^{p-r}\Big(B_{p-r-1}^{k-1}(\zeta)-B_{p-r-1}^{k}(\zeta)\Big)\Big(P_{k+1}-P_k\Big)^{r+1}\Big) \\
&= \frac{p!}{(p-r-1)!}\Big(\sum_{k=1}^{p-r}(P_{k+1}-P_k)^{r+1}B_{p-r-1}^{k-1}(\zeta) \quad - \quad \sum_{k=0}^{p-r-1}(P_{k+1}-P_k)^{r+1}B_{p-r-1}^{k}(\zeta)\Big) \\
&= \frac{p!}{(p-r-1)!}\Big(\sum_{k=0}^{p-r-1}(P_{k+1}-P_k)^{r+1}B_{p-r-1}^{k}(\zeta) \quad - \quad \sum_{k=0}^{p-r-1}(P_{k+1}-P_k)^{r+1}B_{p-r-1}^{k}(\zeta)\Big) \\
&= \frac{p!}{(p-(r+1))!}\Big(\sum_{k=0}^{p-(r+1)}(P_{k+1}-P_k)^{r+1}B_{p-(r+1)}^{k}(\zeta)\Big).
\end{aligned}$$

Therefore, the result is true for $(r+1)$. ■

2.6 Subdivision of Bézier curves

The De Casteljau algorithm is probably the most fundamental algorithm in the field of curve and surface design. This algorithm was devised in 1959 by Paul De Casteljau, a French mathematician from the Citroen automobile company. A main interest of this algorithm is to subdivide a Bézier curve C_p of degree p, defined in $[\zeta_1,\zeta_l]$, into two Bézier curves of degree p, C_p^1 defined in $[\zeta_1,\zeta_{l^1}]$ and C_p^2 defined in $[\zeta_{l^1},\zeta_l]$ whose union is equivalent to the original curve.
Consider now a Bézier curve of degree p defined by the control points $P_0,P_1,...,P_p$. The De Casteljau algorithm is used to obtain a point on the curve at a parameter value $\zeta \in [0,1]$ from the control polygon composed of the points P_i. We construct P_i^j: the new control point during the subdivision step j. Formally, the algorithm of De Casteljau can be written as [65]:

$$\begin{cases} P_i^0 = P_i & i = 0,...,p-r, \\ P_i^j = (1-\zeta)P_i^{j-1} + \zeta P_{i+1}^{j-1} & j = 1,...,p \qquad i = 0,...,p-j. \end{cases}$$

Fig. 2.8 illustrates the De Casteljau evaluation of a point on a cubic Bézier curve at $\zeta = 2/3$. This algorithm can also be represented as a triangular scheme, starting with four control points P_0,P_1,P_2 and P_3, and eventually reducing them to a single point P_0^3.

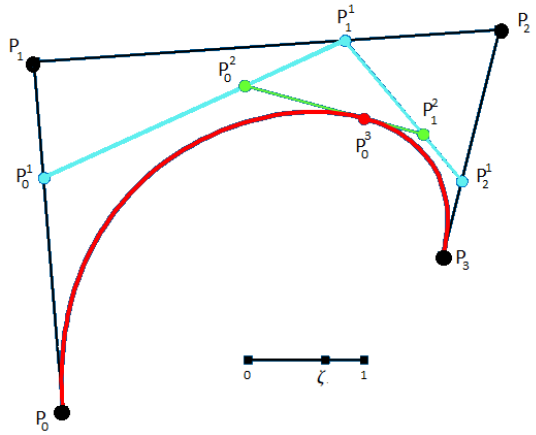


Figure 2.8: Geometric construction according to De Casteljau's algorithm for $p = 3$ and $\zeta = 2/3$.

Thus, it is possible to subdivide a Bézier curve of degree p, into several curves which describe the same curve. Because the resulting Bézier curves must have their own new control points, the original set of control points is discarded. Moreover, since the original Bézier curve C_p is cut into several pieces, each of which is a subset of the original degree p Bézier curve, the resulting Bézier curves must be of degree p.

Note that this algorithm is used in practice to evaluate the curve at a given parameter ζ (for visualization for example).

2.7 Rational Bézier curves

Given $(p+1)$ control points $P_0, P_1, ..., P_p$ and associated nonnegative weights $w_0, w_1, ..., w_p$ we can define, for $\zeta \in [0,1]$, the rational Bézier curve of degree p as:

$$R_p(\zeta) = \frac{\sum_{i=0}^{p} w_i B_p^i(\zeta) P_i}{\sum_{i=0}^{p} w_i B_p^i(\zeta)}.$$

Note that if all weights are 1 (or if all weights are simply equal), a rational Bézier curve reduces to a polynomial Bézier curve. However, almost all properties of Bézier curves still hold for rational Bézier curves (details can be found in [29]). In particular, rational Bézier curves have end-point interpolation, prescribed tangent lines at the endpoints. They also satisfy the convex hull property. Moreover, degree elevation and subdivision can be extended to rational curves. Further, rational Bézier curves have several advantages over polynomial Bézier curves. They provide more control over the shape than does a polynomial Bézier curve. Elsewhere, it is possible to reparametrize the curve by simply changing the weights in a specific manner. Moreover, rational Bézier curves are needed to exactly represent conic sections [55]. Therefore, their use is fundamental in Computer-Aided-Design.

Fig. 2.9 shows several conic curves, which all share the same Bézier points, only the inner weight w changes.

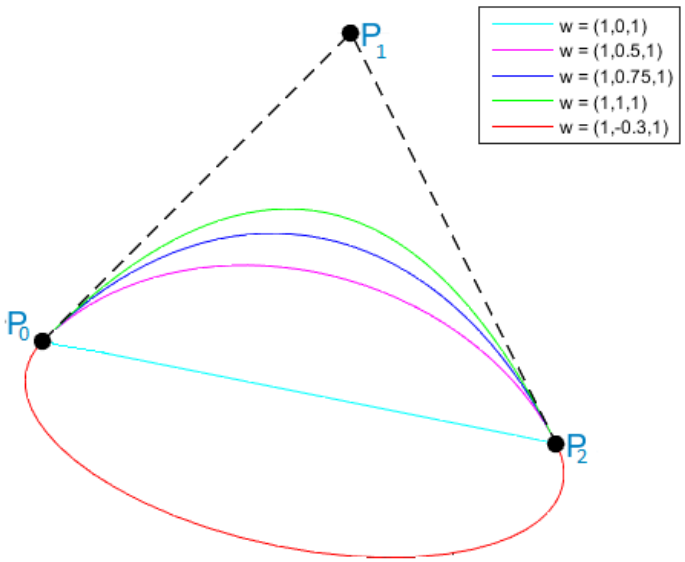


Figure 2.9: Quadratic rational Bézier curves.

2.8 Bézier surface

One can easily extend the previous concepts to define Bézier surfaces, by using a second-order tensor product of Bernstein polynomials. More precisely, a Bézier surface of degree $p_1 \times p_2$ can be formulated as:

$$S(\zeta_1,\zeta_2) = \sum_{i=0}^{p_1} \sum_{j=0}^{p_2} B^i_{p_1}(\zeta_1) B^j_{p_2}(\zeta_2) P_{ij} \qquad \forall \zeta_1,\zeta_2 \in [0,1],$$

where the P_{ij} define a control net. The degrees of Bernstein polynomials p_1 and p_2 do not necessarily have to be the same in the two parameter directions.
Fig. 2.10 gives an example of a tensor product Bézier surface of degree 3×3 and the corresponding control net.

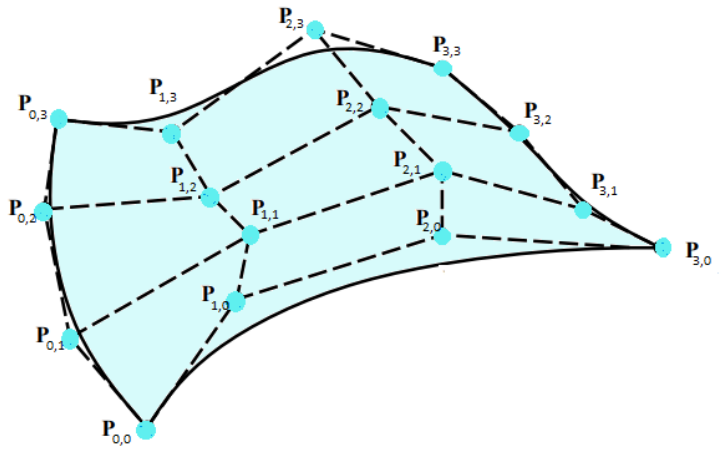


Figure 2.10: Tensor product Bézier patch of degree 3×3 and its control net.

The corresponding properties of the Bézier curve apply to the Bézier surface:

- Invariance under affine transformations.
- Subdivision produces smooth continuous surfaces.
- The Bézier surface is contained within the convex hull of the control nets.
- The Bézier surface can be exactly represented as a Bézier surface of higher degree.
- The Bézier surface does not in general pass through the control nets except for the corners of the control net grid.
- Closed surfaces can be formed by setting the last control net equal to the first.

CONCLUSION

Les courbes de Bézier ont été inventées dans les années 1960 par l'ingénieur Pierre Bézier. Il a étudié le problème de conception de surfaces $3D$ (carrosseries d'automobiles, fuselages d'avion, etc.) pour les premiers programmes de CAO (Conception Assistée par Ordinateur). Le but était de trouver un moyen pour définir des courbes paramétriques de manière précise et simple.

Dans ce chapitre, nous avons examiné les polynômes de Bernstein et les courbes de Bézier détaillées dans les deux cas, rationnel et non rationnel, et les surfaces par produits tensoriels ont été introduites. Diverses exemples de courbes de Bézier ont été générées.

Nous avons discuté des propriétés des courbes de Bézier, ces propriétés sont directement impliquées dans l'algorithme de Casteljau ou dans les propriétés des polynômes de Bernstein. Nous avons mentionné les propriétés les plus fondamentales et nous avons fourni les preuves de ces propriétés brièvement.

En conclusion, nous avons vu que les courbes de Bézier ont un certain nombre de caractéristiques qui les rendent intéressantes pour l'exploration du paramétrage géométrique. Ces courbes sont faciles à décrire paramétriquement. Elles sont faciles à représenter graphiquement elles ont aussi l'avantage d'être incroyablement pratiques, comme en témoigne leur utilisation dans la conception géométrique assistée par ordinateur et infographie depuis les année 1960.

CHAPTER 3

B-SPLINES CURVES

B-spline and Non-Uniform-Rational B-spline (NURBS) (a generalization of B-spline) are the de facto standard for geometric representation in computer aided design (CAD) systems [43]. A brief discussion of B-spline and NURBS is presented in this chapter. We begin by introducing necessary background concepts, defining commonly used notations and introducing B-spline/NURBS curves and surfaces [23] [61] [75].

3.1 B-spline functions

Univariate B-spline functions are piecewise polynomial functions with compact support. They are defined in parametric space using a so-called knot vector denoted Ξ, in one-dimensional space (1D), Ξ is a set of m non-decreasing coordinates:

$$\Xi = \{\xi_1, \xi_2, ..., \xi_m\}.$$

Definition 3.1.1. *The univariate B-spline function $\mathcal{N}_{i,p}$ of degree p is defined according to the Cox–de Boor recursion formula [27]:*

For $p = 0$:

$$\mathcal{N}_{i,0}(\xi) = \begin{cases} 1 & \text{if} \quad \xi_i \leq \xi < \xi_{i+1} \qquad \forall i = 1, ..., m-1, \\ 0 & \text{otherwise.} \end{cases} \tag{3.1}$$

For $p \geq 1$:

$$\mathcal{N}_{i,p}(\xi) = \left(\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i}\right)\mathcal{N}_{i,p-1}(\xi) + \left(\frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}}\right)\mathcal{N}_{i+1,p-1}(\xi). \tag{3.2}$$

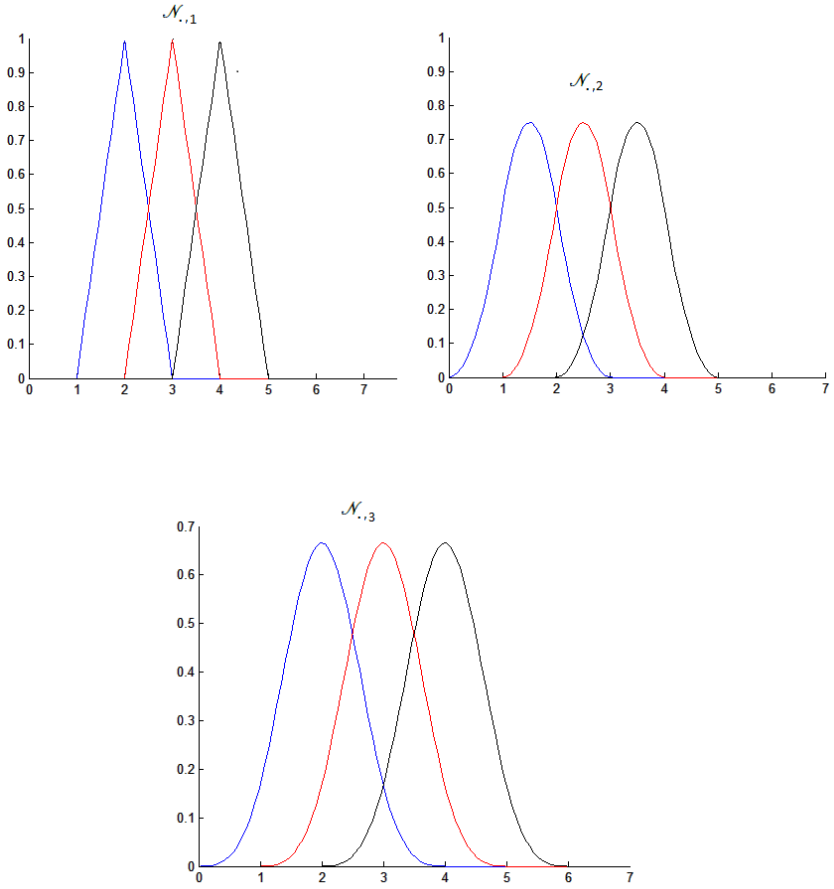


Figure 3.1: Basis functions of degrees 1, 2 and 3 for uniform knot vector $\Xi = \{0, 1, 2, 3, ...\}$.

We use the convention that fraction in front of the basis functions is set equal to zero in the case of the denominator being zero. That is:

$$\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} \equiv 0 \quad \text{if} \quad \xi_{i+p} - \xi_i = 0,$$

$$\frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} \equiv 0 \quad \text{if} \quad \xi_{i+p+1} - \xi_{i+1} = 0.$$

We see that there are exactly n basis functions. This is a direct consequence of the knot vector consisting of $n+p+1$ knots. If we increase the number of elements in the knot vector, we will also increase the number of basis functions. Note that the degree and the number of basis functions are independent, contrary to Bézier functions.

With Eq. (3.1) and Eq. (3.2), it can be noted that for $p = 0$ and $p = 1$, the basis functions of isogeometric analysis are identical to those of the standard piecewise constant and linear finite elements, respectively. In fact, the greatest wealth of B-splines comes from the case $p = 2$.

In order to define multivariate B-splines functions in higher dimensions, we make use of the tensor product.

Definition 3.1.2. *Let $p = (p_1, p_2, ..., p_d)$ be a vector in $\mathbb{N}^d$ and for all $j = 1, ..., d$, let Ξ_j be a 1D knot vector defined by:*

$$\Xi_j = \{\xi_1^j, \xi_2^j, ..., \xi_{n_1+p_1+1}^j\}.$$

Furthermore, if we denote the i_j univariate B-spline of degree p_j defined on the knot vector Ξ_j by $\mathcal{N}_{i_j,p_j}(\xi^j)$, then, with the multi-indices $i = (i_1, i_2, ..., i_d)$, $p = (p_1, p_2, ..., p_d)$ and $n = (n_1, n_2, ..., n_d)$, the d-dimensional tensor product B-spline is defined by:

$$\mathcal{N}_{i,p}(\xi) = \mathcal{N}_{i_1,p_1}(\xi^1) \otimes \mathcal{N}_{i_2,p_2}(\xi^2) \otimes ... \otimes \mathcal{N}_{i_d,p_d}(\xi^d).$$

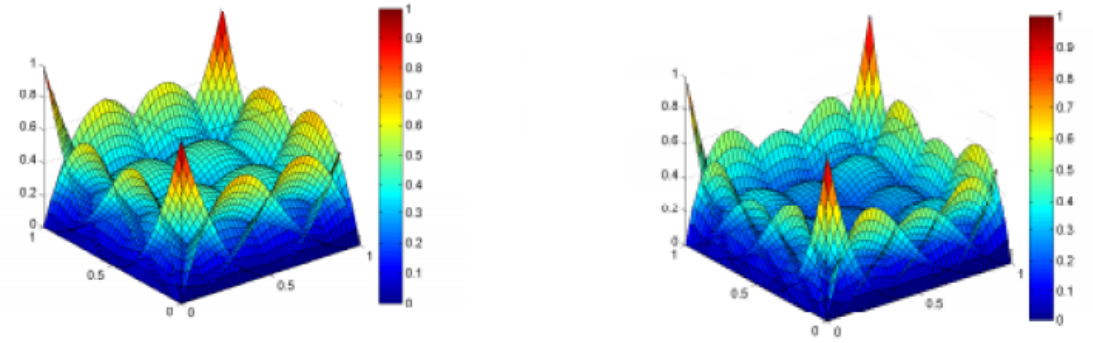


Figure 3.2: Bivariates quadratic and cubic B-spline basis functions [16].

3.1.1 Knot Vectors

A knot vector Ξ in one-dimensional space is a set of finite, real-valued, monotonically increasing sequence of real numbers written,

$$\Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\},$$

where $\xi_i \in \mathbb{R}$ is the $i-th$ knot, $i \in \{1, 2, ..., n+p+1\}$ is the knot index, p is the polynomial degree and n is the number of basis functions which define the B-spline. The interval $[\xi_1, ..., \xi_i]$ is called **a patch** $\forall 1 < i \leq n+p+1$, whereas the interval between two knots $[\xi_i, \xi_{i+1}]$ is called **knot span**. A knot span is called **empty** if $\xi_i = \xi_{i+1}$ and is called **interior** if $\xi_i < \xi_{i+1}$ $\forall 1 \leq i \leq n+p$.

There are different types of knot vectors:

• **Periodic knot vector**: a knot vector Ξ is periodic if there exists an integer **I** and a real **T** such that for every $i \in \mathbb{Z}$,

$$\xi_{i+\mathbf{I}} = \xi_i + \mathbf{T}.$$

• **Uniform knot vector**: if knots are equally-spaced in the parametric space, they are said to be uniform and non-uniform otherwise.

• **Open knot vetor**: a knot vector for a B-spline basis of degree p is said to be open if the first and last knots are repeated $p+1$ times.

$$\Xi = \Big[\underbrace{\xi_1, ..., \xi_1}_{(p+1)\text{times}}, \xi_{p+2}, ..., \underbrace{\xi_{n+p+1}, ..., \xi_{n+p+1}}_{(p+1)\text{times}} \Big].$$

Open knot vectors are standard in the CAD literature. In one dimension, basis functions formed from open knot vectors are interpolatory at the ends of the parametric space interval, $[\xi_1, \xi_{n+p+1}]$, and at the corners of patches in multiple dimension but they are not, in general, interpolatory at interior knots. That is a distinguishing feature between "knots" and "nodes" in FEA.

The equations (3.1) and (3.2) show clearly that the choice of the knot vector has a significant influence on the B-spline functions. An example of quadratic basis functions for an open, uniform knot vector is presented in Fig. 3.3.

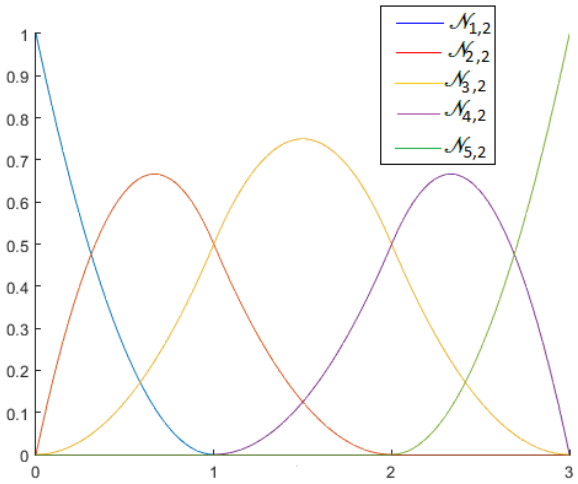


Figure 3.3: Quadratic basis functions for the open-uniform knot vector $\Xi = \{0,0,0,1,2,3,3,3\}$.

Note that the basis functions are interpolatory at the ends of the interval and also at $\xi = 1$, the location of a repeated knot, where only C^0-continuity is attained (see Fig. 3.4).

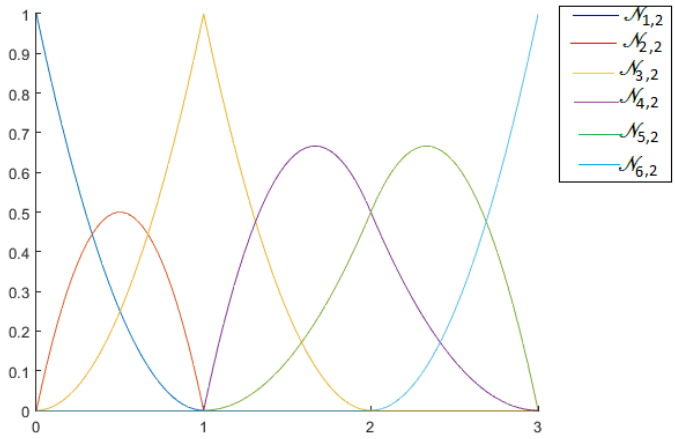


Figure 3.4: Quadratic basis functions with reduced continuity at $\xi = 1$, $\Xi = \{0,0,0,1,1,2,3,3,3\}$.

In general, basis functions of degree p have $p-1$ continuous derivatives. If a knot is repeated k times (with $k < p$), then the number of continuous derivatives decreases by k. When the multiplicity of a knot is exactly p (i.e. $k = p$), the B-spline basis functions are C^0 continuous at that knot point. Obviously, if the multiplicity of the knot point is $p+1$ (i.e. $k = p+1$) the B-spline basis functions are discontinuous at that knot point and as a result two separate B-splines patches are formed.
This property will be used later, to generate a computational domain suitable to Discontinuous Galerkin methods.

3.1.2 Properties of the B-spline functions

The B-spline functions $\mathcal{N}_{i,p}$ defined using the procedure described above are polynomials of degree p. There are several important features of the basis functions that are pointed out by Hughes et al. [23], in the perspective of PDE analysis.

1. B-spline functions form a partition of unity, i.e.

$$\sum_{i=1}^{n} \mathcal{N}_{i,p}(\xi) = 1 \qquad \forall \xi \in \Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}.$$

2. The basis functions are interpolatory at the end points of the knot vector, such that:

$$\mathcal{N}_{i,p}(\xi_1) = \delta_{i,1} \quad \text{and} \quad \mathcal{N}_{i,p}(\xi_{n+p+1}) = \delta_{i,n}$$

where, δ is the Kronecker symbol.

3. The support of each $\mathcal{N}_{i,p}$ is compact and contained in $[\xi_i, \xi_{i+p+1}]$. As seen in Fig. 3.4, the supports of the functions are growing with increasing polynomial degree. In fact, the support will cover exactly $p+2$ knots. Note however that some of these knots may be equal and thus have knot multiplicity greater than one. In these cases, the support will not go over $p+1$ knot spans, which is the maximum support it can have.

4. Each B-spline function is nonnegative over the entire domain, that is:

$$\mathcal{N}_{i,p}(\xi) \geq 0 \quad \forall \xi \in \Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\},$$

which means that all the coefficients of a mass matrix computed from a B-spline basis are also nonnegative, which can be useful for mass lumping schemes [23].

5. The B-spline functions indeed form a basis for the space of polynomials of degree less than or equal to p, $\mathbb{P}_p$. That is, they are all linearly independent i.e.

$$\sum_{i=1}^{n} \alpha_i \mathcal{N}_{i,p}(\xi) = 0 \iff \alpha_i = 0 \qquad \forall i = 1, 2, ..., n.$$

6. The B-spline function $\mathcal{N}_{i,p}(\xi_i)$ is of regularity C^{p-r} at each knot of multiplicity r. When the multiplicity of a knot is exactly p, the basis function is interpolatory.

3.2 Derivatives of B-spline functions

The derivatives of B-spline functions are represented in terms of lower order B-spline basis, as it comes directly from the recursive definition given in equations (3.1) and (3.2). Thus, the first derivative of the i-th B-spline basis function of degree p is given by:

$$\frac{d}{d\xi}\mathcal{N}_{i,p}(\xi) = p\left(\left(\frac{1}{\xi_{i+p} - \xi_i}\right)\mathcal{N}_{i,p-1}(\xi) - \left(\frac{1}{\xi_{i+p+1} - \xi_{i+1}}\right)\mathcal{N}_{i+1,p-1}(\xi)\right). \tag{3.3}$$

Proof. $\frac{d}{d\xi}\mathcal{N}_{i,0}(\xi) = 0 \qquad \Big(\mathcal{N}_{i,0}(\xi)$ and $\mathcal{N}_{i+1,0}(\xi)$ are constants $\Big)$.

$$\begin{aligned}\frac{d}{d\xi}\mathcal{N}_{i,1}(\xi) &= \frac{d}{d\xi}\left(\left(\frac{\xi - \xi_i}{\xi_{i+1} - \xi_i}\right)\mathcal{N}_{i,0}(\xi) + \left(\frac{\xi_{i+2} - \xi}{\xi_{i+2} - \xi_{i+1}}\right)\mathcal{N}_{i+1,0}\right) \\ &= \left(\frac{1}{\xi_{i+1} - \xi_i}\right)\mathcal{N}_{i,0}(\xi) - \left(\frac{1}{\xi_{i+2} - \xi_{i+1}}\right)\mathcal{N}_{i+1,0}.\end{aligned}$$

We assume that for all $0 \le q \le p$, we have:

$$\frac{d\mathcal{N}_{i,q}(\xi)}{d\xi} = \Big(\frac{q}{\xi_{i+q}-\xi_i}\Big)\mathcal{N}_{i,q-1}(\xi) \quad - \quad \Big(\frac{q}{\xi_{i+q+1}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,q-1}(\xi).$$

Let us prove that (3.3) is true for $p+1$.

$$\begin{aligned}
\frac{d}{d\xi}\mathcal{N}_{i,p+1}(\xi) &= \frac{d}{d\xi}\Big(\big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\big)\mathcal{N}_{i,p}(\xi) + \big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\big)\mathcal{N}_{i+1,p}(\xi)\Big) \\
&= \Big(\frac{1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) + \Big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\Big)\Big(\frac{d\mathcal{N}_{i,p}(\xi)}{d\xi}\Big) - \Big(\frac{1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi) \\
&+ \Big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\Big)\Big(\frac{d\mathcal{N}_{i+1,p}(\xi)}{d\xi}\Big) \\
&= \Big(\frac{1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) \quad + \quad \Big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\Big)\Big(\big(\frac{p}{\xi_{i+p}-\xi_i}\big)\mathcal{N}_{i,p-1}(\xi) - \big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\big)\mathcal{N}_{i+1,p-1}(\xi)\Big) \\
&- \Big(\frac{1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi) + \Big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\Big)\Big(\big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\big)\mathcal{N}_{i+1,p-1}(\xi) - \big(\frac{p}{\xi_{i+p+2}-\xi_{i+2}}\big)\mathcal{N}_{i+2,p-1}(\xi)\Big) \\
&= \Big(\frac{1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) \quad + \quad \Big(\frac{p}{\xi_{i+p}-\xi_i}\Big)\Big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p-1}(\xi) \quad - \quad \Big(\frac{1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi) \\
&+ \Big(\big(\frac{-p}{\xi_{i+p+1}-\xi_{i+1}}\big)\big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\big) + \big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\big)\big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\big)\Big)\mathcal{N}_{i+1,p-1}(\xi) \\
&+ \Big(\frac{-p}{\xi_{i+p+2}-\xi_{i+2}}\Big)\Big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+2,p-1}(\xi).
\end{aligned}$$

$$\begin{aligned}
\Big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\Big)\Big(\big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\big) - \big(\frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\big)\Big)\mathcal{N}_{i+1,p-1} &= \Big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\Big)\Big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}} - \frac{\xi_{i+p+2}-\xi_{i+1}}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p-1} \\
&+ \Big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\Big)\Big(\frac{\xi_{i+p+1}-\xi_i}{\xi_{i+p+1}-\xi_i} \quad - \quad \frac{\xi-\xi_i}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i+1,p-1} \\
&= \Big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\Big)\Big(\frac{\xi_{i+1}-\xi}{\xi_{i+p+2}-\xi_{i+1}} + \frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i+1,p-1}.
\end{aligned}$$

Then, we get:

$$\begin{aligned}
\frac{d}{d\xi}\mathcal{N}_{i,p+1}(\xi) &= \Big(\frac{1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) - \Big(\frac{1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi) + \Big(\frac{p}{\xi_{i+p+1}-\xi_i}\Big)\Big(\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i}\mathcal{N}_{i,p-1}(\xi) \\
&+ \Big(\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p-1}(\xi)\Big) + \Big(\frac{p}{\xi_{i+p+1}-\xi_{i+1}}\Big)\Big(\frac{\xi_{i+1}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p-1}(\xi) \\
&+ \Big(\frac{-p}{\xi_{i+p+2}-\xi_{i+2}}\Big)\Big(\frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+2,p-1}(\xi) \\
&= \Big(\frac{p+1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) - \Big(\frac{1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi) \\
&+ \Big(\frac{-p}{\xi_{i+p+2}-\xi_{i+1}}\Big)\Big(\frac{\xi-\xi_{i+1}}{\xi_{i+p+1}-\xi_{i+1}}\mathcal{N}_{i+1,p-1}(\xi) + \frac{\xi_{i+p+2}-\xi}{\xi_{i+p+2}-\xi_{i+2}}\mathcal{N}_{i+2,p-1}(\xi)\Big) \\
&= \Big(\frac{p+1}{\xi_{i+p+1}-\xi_i}\Big)\mathcal{N}_{i,p}(\xi) - \Big(\frac{p+1}{\xi_{i+p+2}-\xi_{i+1}}\Big)\mathcal{N}_{i+1,p}(\xi).
\end{aligned}$$

■

3.3 B-spline curves

B-spline curves are defined as a linear combination of control points and B-spline basis functions. Given n basis functions $\mathcal{N}_{i,p}$, $i = 1,...,n$ and corresponding control points $P_i \in \mathbb{R}$, $i = 1,...,n$, a piecewise-polynomial B-spline curve is obtained as:

$$\mathcal{C}_p(\xi) = \sum_{i=1}^{n} \mathcal{N}_{i,p}(\xi) P_i.$$

The linear interpolation of the control points is called the control polygon. It can be seen that a Bézier curve of order $n+1$ (degree n) is a special case of B-spline curve with no internal knots and the ends knots are repeated $n+1$ times. The knot vector is thus:

$$\Xi = \{ \underbrace{0,0,...,0}_{(n+1)\,times}, \underbrace{1,1,...,1}_{(n+1)\,times} \}.$$

An example is shown in Fig. 3.5 for the quadratic basis functions considered previously. Note that the curve is interpolatory at the first and last control points P_1 and P_5, due to the fact that the knot vector is uniform and open, $\Xi = \{0,0,0,1,2,3,3,3\}$. The curve $\mathcal{C}_2$ is tangent to the control polygon at the first and last control points. The curve is C^1 continuous everywhere.

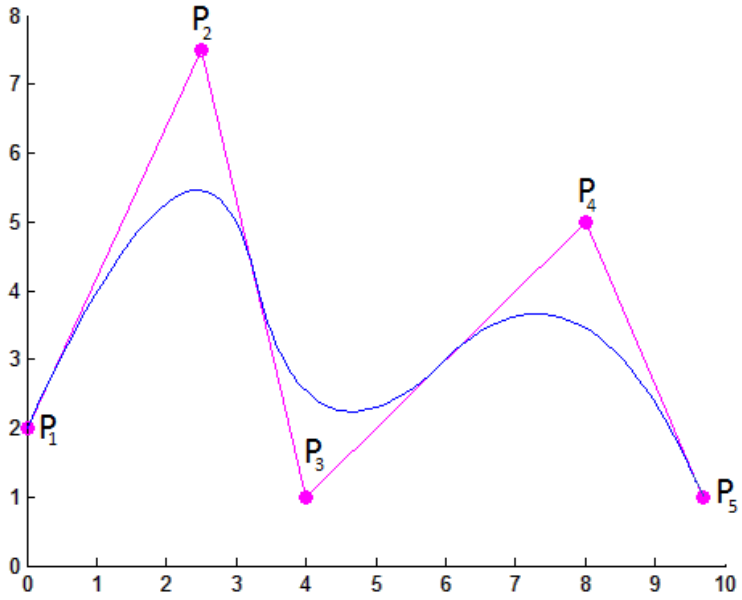


Figure 3.5: Quadratic B-spline curve.

B-spline curves are generalization of Bézier curves and share many important properties with them. Moreover, B-spline curves have additional useful properties [23]. We list below some of the most important properties of B-spline curves:

• The curve $\mathscr{C}_p$ is C^{p-1} continuous everywhere except at the knot or control point of multiplicity r, where it is C^{p-r}.

• Invariance with respect to affine transformations. In fact, an affine transformation of a B-spline curve is obtained by applying the transformation directly to the control points.

• If the knot vector is open, the curve starts at the point P_1 and ends at P_n. Moreover, it is tangent to (P_0P_1) and $(P_{n-1}P_n)$ at the extremities.

• Changing the position of control point P_i affects $\mathscr{C}_p$ only in the interval $[\xi_i, \xi_{i+p}]$, which is a significant difference with respect to Bézier curves.

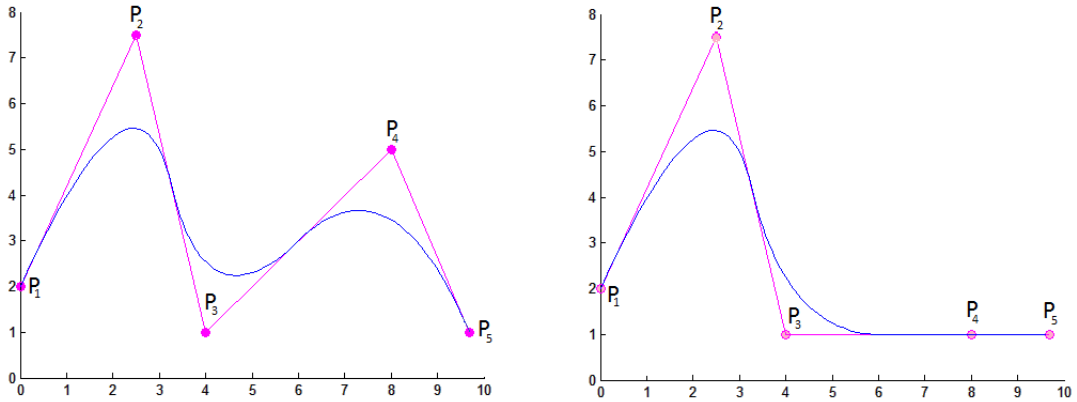


Figure 3.6: A quadratic B-spline curve and its control points. In the right, the curve after moving the control point P_4.

• B-spline curve is contained in the convex hull of its control point. More specifically, if $\xi \in [\xi_i, \xi_{i+1})$, then $\mathscr{C}_p(\xi)$ is in the convex hull of control points $P_{i-p}, P_{i-p+1}, ..., P_i$.

• The derivative of a B-spline curve is given by:

$$\frac{d}{d\xi}\mathscr{C}_p(\xi) = p\Big(\sum_{i=0}^{n-1}\big(\frac{P_{i+1}-P_i}{\xi_{i+p+1}-\xi_{i+1}}\big)\Big)\mathscr{N}_{i+1,p-1}(\xi).$$

3.4 Hierarchical representation

The B-spline curves can be enriched by three types of refinements, which are termed $h-$, $p-$ and $k-$ refinements, without changing the shape of the geometry. For further details, we refer the reader to [23].

3.4.1 Knot insertion

The first mechanism by which one can enrich the basis is knot insertion. Knots may be inserted without changing a curve geometrically or parametrically. More precisely, given a B-spline curve with m control points P_i, the same curve can be obtained with $m+1$ control points $\widetilde{P}_i$ by inserting $\widetilde{\xi} \in [\xi_k, \xi_{k+1}[$ given by:

$$\widetilde{P}_i = \alpha_i P_i + (1-\alpha_i)P_{i-1},$$

where,

$$\alpha_i = \begin{cases} 1 & \text{if} \quad i \le k-p, \\ \frac{\xi-\xi_i}{\xi_{i+p}-\xi_i} & \text{if} \quad k-p+1 \le i \le k, \\ 0 & \text{if} \quad i \ge k+1. \end{cases}$$

An example of knot insertion is presented in Fig. 3.7.

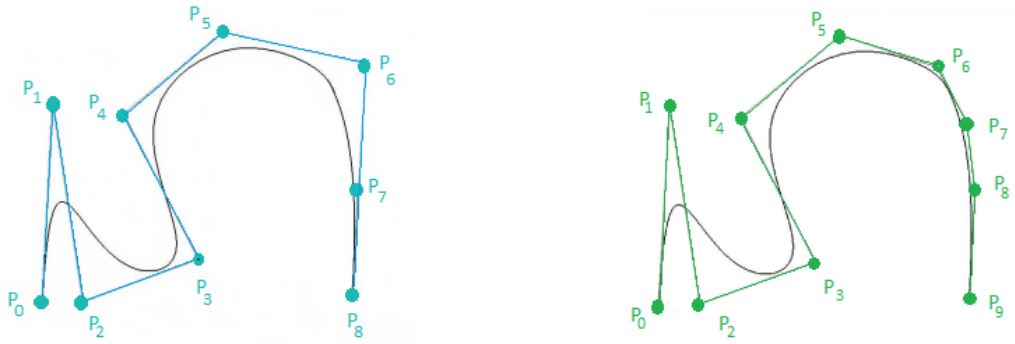


Figure 3.7: Before and after knot insertion (cubic B-spline curve).

The original curve, shown on the left, consists of a cubic B-spline curve with 9 control points and has $9+3+1=13$ knot values. The new curve, shown on the right, is geometrically and parametrically identical to the first one, but the basis functions and control points are changed, there is one more of each. This process may be repeated to enrich the solution space by adding more basis functions of the same order while keeping the curve unchanged. Insertion of new knot values clearly has similarities with the classical $h-$refinement strategy in FEA. As explained in section 3.1.1, inserting a knot on an existing knot results in a decrease of the curve regularity (multiple knots).

3.4.2 Order elevation

The second mechanism by which one can enrich the basis is order elevation (sometimes also called "degree elevation") can be thought of as $p-$refinement in FEA. During this process, the multiplicity of each knot is increased by one but no new knots are added. Let $\Xi = \{\xi_1, ..., \xi_m\}$ be a knot vector, the degree p of a B-spline curve $\mathscr{C}_p$ defined for the knot vector Ξ may be increased without changing the geometry or parametrization. It is possible to define another B-spline curve of degree $p+1$ that is identical to the original one. The number of new control points depends on the multiplicities of existing knots. The process for order elevation is the following:

1. Begins by subdividing the curve into many Bézier curves of degree p by knot insertion (see section 3.7).

2. The next step is to elevate the order of the polynomial on each of these individual segments (see section 2.5.1).

3. Last, excess knots are removed to combine the segments into a B-spline curve of degree $p+1$.

The basic idea of degree raising and knot insertion is to achieve the flexibility without changing the shape of the curve or surface. We refer to [23] for the mathematical details.

3.4.3 $k-$refinement

$k-$refinement refers to the process in which order elevation is followed by knot insertion. It has no analogous in FEA. It is important to point out that the order elevation and knot insertion do not commute. This process results in a higher order and higher continuity basis than the process of knot insertion followed by order elevation. We refer to [23] for a thorough treatment and application examples.

3.5 B-spline surfaces and volumes

Given a control net $(P_{i,j})_{i=1,...,n \quad j=1,...,m}$ and two knot vectors, $\Xi_1 = \{\xi_1, ..., \xi_{n+p+1}\}$ and $\Xi_2 = \{\eta_1, ..., \eta_{m+q+1}\}$, a tensor-product B-spline surface is defined as:

$$\mathscr{S}(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \mathscr{N}_{i,p}(\xi) \mathscr{N}_{j,q}(\eta) P_{i,j}.$$

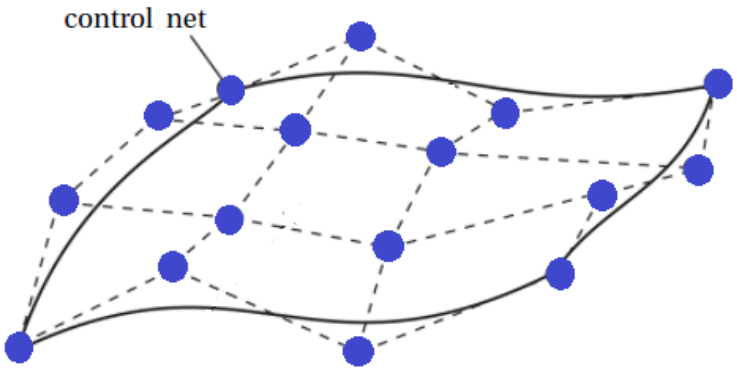


Figure 3.8: B-spline surface example.

B-spline volumes are defined using a third-order tensor product of B-spline functions. Given a control net $P_{i,j,k}$, $i = 1, ..., n$, $j = 1, ..., m$, $k = 1, ..., r$ and knot vectors $\Xi_1 = \{\xi_1, ..., \xi_{n+p+1}\}$, $\Xi_2 = \{\eta_1, ..., \eta_{m+q+1}\}$ and $\Xi_3 = \{\zeta_1, ..., \zeta_{l+l+1}\}$, a B-spline volume is defined by:

$$V(\xi, \eta, \zeta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \sum_{k=1}^{l} \mathscr{N}_{i,p}(\xi) \mathscr{N}_{j,q}(\eta) \mathscr{N}_{k,r}(\zeta) P_{i,j,k}.$$

It is important to note that many properties of a B-spline surface and volume are the results of the tensor product nature. Multivariate B-splines basis functions are nonnegative, have local support, are invariant to affine transformations and form a partition of unity.

3.6 Non-Uniform Rational B-spline (NURBS)

B-splines are convenient for free-form modelling, but they lack the ability to exactly represent some simple shapes such as circles and ellipsoids. NURBS are non-rational functions of B-splines, they allow for the exact parametrization of common curves and surfaces such as circles, cylinders and spheres. They are also extremely flexible and intuitive when dealing with more complex shape creation and deformation. A NURBS entity in $\mathbb{R}^d$ is obtained by the projective transformation of a B-spline entity in $\mathbb{R}^{d+1}$ [23].

3.6.1 NURBS basis functions

Let $(w_i)_{1\leq i\leq n}$ be a sequence of non-negative reals (weights for control point). The i-th NURBS function of degree p, associated to the knot vector Ξ and the weights w, is given by:

$$\mathscr{R}_i^p(\xi) = \frac{\mathscr{N}_{i,p}(\xi)\, w_i}{\sum_{j=1}^n \mathscr{N}_{j,p}(\xi)\, w_j},$$

where $\mathscr{N}_{i,p}(\xi)$ denotes the i-th B-spline basis function of degree p.
The first derivative of a NURBS basis function is given by:

$$\frac{d}{d\xi}\mathscr{R}_i^p(\xi) = w_i \frac{\frac{d}{d\xi}\mathscr{N}_{i,p}(\xi) W(\xi) - \mathscr{N}_{i,p}(\xi)\frac{d}{d\xi}W(\xi)}{W^2(\xi)},$$

where,

$$W(\xi) = \sum_{j=1}^n \mathscr{N}_{j,p}(\xi)\, w_j,$$

$$\frac{d}{d\xi}W(\xi) = \sum_{j=1}^n \frac{d}{d\xi}\mathscr{N}_{j,p}(\xi)\, w_j.$$

3.6.2 NURBS curves and surfaces

The NURBS curve of degree p associated to the knot vector Ξ, the control points $(B_i)_{1\leq i\leq n}$ and the weights $(w_i)_{1\leq i\leq n}$ is defined as:

$$\mathfrak{C}_p(\xi) = \sum_{i=1}^n \mathscr{R}_i^p(\xi) B_i.$$

Similarly, the NURBS surfaces of degree $p_1 \times p_2$ associated to the knot vectors Ξ_1 and Ξ_2, the control nets $(B_{i,j})_{1\leq i\leq n_1, 1\leq j\leq n_2}$ and the weights $(w_{ij})_{1\leq i\leq n_1, 1\leq j\leq n_2}$, are defined by:

$$\mathfrak{S}(\xi,\eta) = \sum_{i=1}^{n_1}\sum_{j=1}^{n_2} \mathscr{R}_{ij}(\xi,\eta) B_{i,j}.$$

With,

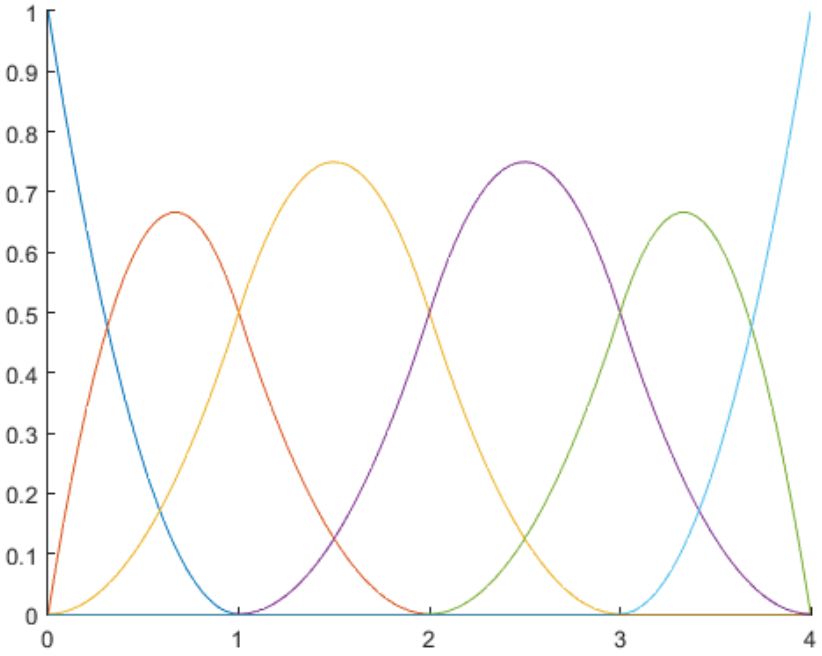
$$\mathscr{R}_{ij}(\xi,\eta) = \frac{w_{ij}\mathscr{N}_{i,p_1}(\xi)\mathscr{N}_{j,p_2}(\eta)}{\sum_{i=1}^{n_1}\sum_{j=1}^{n_2} w_{ij}\mathscr{N}_{i,p_1}(\xi)\mathscr{N}_{j,p_2}(\eta)}$$

We recall some important properties of NURBS (we refer to [23] for more details):

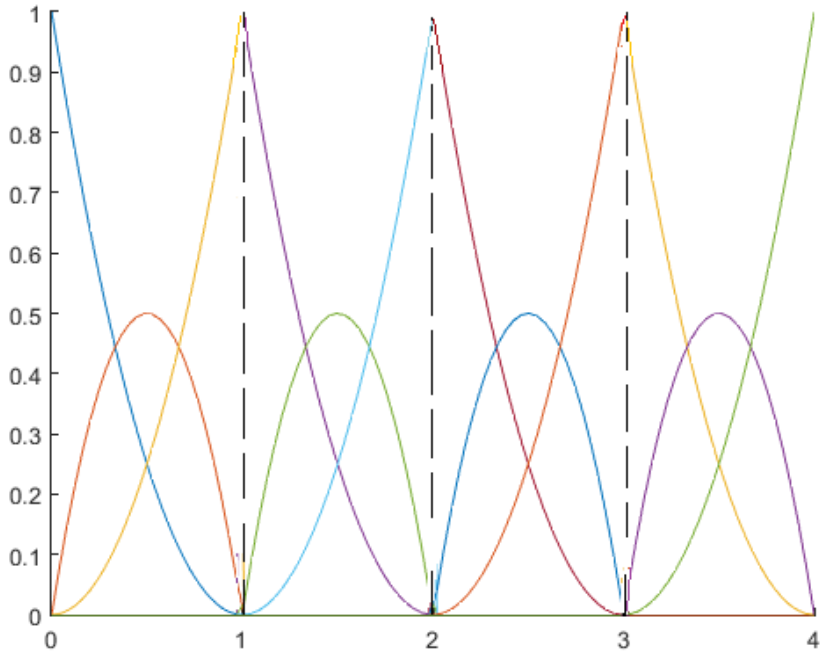
- NURBS basis functions forms a partition of unity.
- The continuity and support of NURBS basis functions are the same as for B-splines.
- Affine transformations in physical space are obtained by applying the transformation to the control points, that is, NURBS possess the property of affine invariance.
- If the weights are equal, NURBS become B-splines.
- NURBS surfaces and solids are the projective transformations of tensor product, piecewise polynomial entities.

3.7 Extracting Bézier curves from B-splines

Let us consider the knot vector $\Xi = \{\xi_1, ..., \xi_{n+p+1}\}$ where, $\xi_1 = ... = \xi_{p+1}$ and $\xi_{n+1} = \xi_{n+p+1}$.
To decompose a B-spline (or NURBS) curve to its Bézier elements, called Bézier extraction, a straightforward approach consists in using the knot-insertion procedure several times, for each of the existing interior knots $(\xi_{p+2}, ..., \xi_n)$ until interior knots have a multiplicity $p+1$. Then, the original B-spline curve is separated in independant Bézier elements. The curve is geometrically unmodified, but its representation is split in a set of discontinuous elements. This is the key procedure to build a computational domain suitable to Discontinuous Galerkin method.
It is important to point out that the Bézier patch is a particular case of B-spline patch, for which the number n of functions (and control points) is equal to $p+1$. An example illustrating Bézier extraction from quadratic B-spline is shown in Fig. 3.9.



$$\Xi = \{0, 0, 0, 1, 2, 3, 4, 4, 4\}$$



$$\Xi = \{0, 0, 0, 1, 1, 1, 2, 2, 2, 3, 3, 3, 4, 4, 4\}$$

Figure 3.9: Bézier decomposition (bottom) from a quadratic B-spline basis (top) by knot insertion.

CONCLUSION

Les courbes B-splines ont été définies dans les années 1970 par Cox et DeBoor, pour remédier à l'inconvénient de la globalité des courbes de Bézier, à travers un algorithme efficace car hiérarchique (analogue à celui de Casteljau), stable numériquement (coefficients multiplicatifs toujours positifs) et interprétable géométriquement: le déplacement d'un point de contrôle de la courbe n'affecte ainsi plus qu'une partie limitée de la courbe, ce qui amène un plus grand confort dans la Conception Assistée par Ordinateur (CAO).
Par conséquent, les courbes B-splines permettent d'approcher des points de manière lisse, comme les coubes de Bézier. Leur avantage est qu'elles sont plus lisses et plus faciles à contrôler.

Dans ce chapitre, nous avons présenté des définitions formelles des fonctions et des courbes B-splines, ainsi qu'une étude détaillée des propriétés les plus fondamentales des B-splines, illustrées par des exemples et des figures.

CHAPTER 4

CURVE AND SURFACE FITTING

Curve or surface fitting, also known as regression analysis, is used to approximate a curve or surface for a series of data points. Most of the time, the curve/surface fitting is used to find points and derivatives anywhere along the curve/surface. This procedure is attractive because of its ability to produce an interpolating curve/surface that retains extremely high accuracy with a minimal number of data to represent the curve/surface, so that the error is minimized in the least squares sense. The purpose of this chapter is to present the problem of fitting a given ordered set of data with a B-spline curve or surface in the least squares sense.

In the context of isogeometric analysis, this technique will be used in the following two cases:

• to define the B-spline computational domain if the geometry of the problem is not defined originally in terms of B-spline,

• to define the B-spline initial conditions.

4.1 Curve fitting

4.1.1 Basic concepts

Curve or surface fitting is a fundamental problem in many fields, such as computer graphics, image processing, shape modeling and data mining. Depending on applications, different types of curves such as parametric curves, implicit curves and subdivision curves are used for fitting [53] [96]. In this chapter, we discuss the problem of B-spline curve and surface fitting [10] [57]. Due to the fact that the curve and surface fitting are closely related topics, we define and discuss the curve fitting problem first and show how to generalize it to surface reconstructions later on.

The general objective of curve fitting is to theoretically describe experimental data with a model (function or equation) and to find the parameters associated with this model.
We recall that a B-spline curve of degree p is defined for a collection of n control points $(P_i)_{i=1,...,n}$ by:

$$\mathscr{C}_p(\xi) = \sum_{i=1}^{n} \mathscr{N}_{i,p}(\xi) P_i.$$

The functions $\mathscr{N}_{i,p}(\xi)$ are the B-spline basis functions of degree p, which are defined recursively and require the selection of a sequence of scalars $\xi_i \in \Xi = \{\xi_1, \xi_2, ..., \xi_{n+p+1}\}$.
Let $(\zeta_k)_{1\leq k\leq n_{eval}}$ such that $\zeta_1 < \zeta_2 < ... < \zeta_{n_{eval}}$ and let Q be a set of ordered and distinct points Q_k, $\forall k = 1, .., n_{eval}$ to be fitted.
The basic aim is to fit a B-spline curve that will approximate n_{eval} measured data points in a least squares sense (with $n_{eval} \gg n$). This leads to find an optimal set of control point $\{P_i\}_{1\leq i\leq n}$ producing an optimal approximating B-spline $\mathscr{C}_p$ with minimal distances to the points Q_k. We seek therefore to minimize the least-squares error, defined as the sum of squares of the points distance expressed as:

$$\frac{1}{2} \sum_{k=1}^{n_{eval}} \left| \sum_{i=1}^{n} \mathscr{N}_{i,p}(\zeta_k) P_i - Q_k \right|^2,$$

where $\sum_{i=1}^{n} \mathscr{N}_{i,p}(\zeta_k) P_i$ is the B-spline curve point at ζ_k and Q_k is the corresponding measured data point.

4.1.2 Description of the least squares method

The least-squares estimation procedure is a mathematical tool that was developed independently by Carl Friedrich Gauss in 1795 and Adrien-Marie Legendre who published it first in 1805 [52]. This theory describes a frequently used approach evolved from statistical methods to estimate values of parameters of a mathematical model from measured data, which are subject to errors.
For a specified set of control points, the least-squares error function between the B-spline curve and sample points is the scalar-valued function:

$$E(P) = \frac{1}{2} \sum_{k=1}^{n_{eval}} \left| \sum_{j=1}^{n} \mathscr{N}_{j,p}(\zeta_k) P_j - Q_k \right|^2. \tag{4.1}$$

As it has been pointed previously, the goal is to find values of the control points that minimize the error. Therefore, we find the values of $\{P_i\}_{1\leq i\leq n}$ in such a way that:

$$\frac{\partial E}{\partial P_i} = 0.$$

Thus, carrying out the differentiation leads to:

$$\begin{aligned}
\frac{\partial E}{\partial P_i} &= \frac{\partial}{\partial P_i}\Big(\frac{1}{2}\sum_{k=1}^{n_{eval}}\Big|\sum_{j=1}^{n}\mathcal{N}_{j,p}(\zeta_k)P_j - Q_k\Big|^2\Big) \\
&= \sum_{k=1}^{n_{eval}}\Big(\sum_{j=1}^{n}\mathcal{N}_{j,p}(\zeta_k)P_j - Q_k\Big)\mathcal{N}_{i,p}(\zeta_k) \\
&= \sum_{k=1}^{n_{eval}}\sum_{j=1}^{n}\mathcal{N}_{i,p}(\zeta_k)\mathcal{N}_{j,p}(\zeta_k)P_j - \sum_{k=1}^{n_{eval}}\mathcal{N}_{i,p}(\zeta_k)Q_k \\
&= \sum_{k=1}^{n_{eval}}\sum_{j=1}^{n}a_{ki}a_{kj}P_j - \sum_{k=1}^{n_{eval}}a_{ki}Q_k,
\end{aligned}$$

where $a_{ki} = \mathcal{N}_{i,p}(\zeta_k)$.

Setting the partial derivatives equal to the zero vector ($0_{\mathbb{R}^n}$) leads to the system of equations:

$$\begin{aligned}
0_{\mathbb{R}^n} &= \sum_{k=1}^{n_{eval}}\sum_{j=1}^{n}a_{ki}a_{kj}P_j - \sum_{k=1}^{n_{eval}}a_{ki}Q_k \\
&= A^tAP - A^tQ,
\end{aligned}$$

where a matrix $A \in \mathbb{R}^{n_{eval}\times n}$ given by: $A = \big(a_{ki}\big)_{1\le k\le n_{eval},1\le i\le n}$.

This system of equations is a least-squares formulation for an over determined problem.

Hence, the minimization leads to the linear system:

$$A^tAP = A^tQ. \tag{4.2}$$

As consequence, the equation 4.2 is equivalent to solving:

$$P = \big(A^tA\big)^{-1}A^tQ. \tag{4.3}$$

4.2 B-spline curve fitting - example

In this section, an example of Least-squares B-spline curve fitting is given to illustrate the performance of the method.

Consider the sinusoidal function:

$$f(x) = \sin(2\pi x) \qquad \forall x \in \Omega = [-1,1].$$

In our example we consider fitting a data set of n_{eval} uniformly distributed parameters $(\zeta_k)_{1\le k\le n_{eval}}$ and points $Q_k = \sin(2\pi\zeta_k)$, $\forall 1 \le k \le n_{eval}$. In this case, we choose $n_{eval} = 80 \gg n = 8$. As seen in Fig. 4.1 an accurate approximation is obtained using only 8 control points.

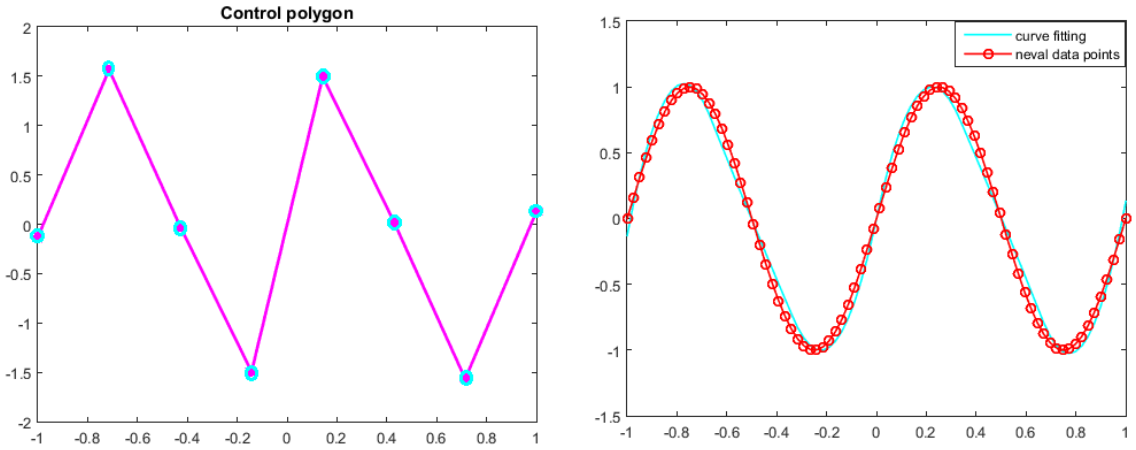


Figure 4.1: The least-squares quadratic B-spline curve fitting.

4.3 Least-squares B-spline surface fitting

A B-spline tensor product surface is defined for a bi-dimensional array of $n \times m$ control points P_{ij} with $1 \le i \le n$ and $1 \le j \le m$,

$$\mathscr{S}(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} \mathscr{N}_{i,p}(\xi)\mathscr{N}_{j,q}(\eta)P_{i,j}. \tag{4.4}$$

The numbers p and q are the degrees for the surface, $\mathscr{N}_{i,p}(\xi)$ and $\mathscr{N}_{j,q}(\eta)$ are the B-spline basis functions. The sample data points are $(\zeta_{k_1},\theta_{k_2},Q_{k_1k_2})$ with $1 \le k_1 \le n_{eval}$ and $1 \le k_2 \le m_{eval}$. It is assumed that:

$$\zeta_1 < \zeta_2 < \cdots < \zeta_{n_{eval}} \quad \text{and} \quad \theta_1 < \theta_2 < \cdots < \theta_{m_{eval}}.$$

The control points may be arranged formally as an $n \times m$ matrix.

$$P = \begin{pmatrix} P_{11} & \cdots & P_{1m} \\ \vdots & \ddots & \vdots \\ P_{n1} & \cdots & P_{nm} \end{pmatrix}.$$

Similarly, the samples $Q_{k_1k_2}$ may be arranged formally as an $n_{eval} \times m_{eval}$ matrix:

$$Q = \begin{pmatrix} Q_{11} & \cdots & Q_{1m_{eval}} \\ \vdots & \ddots & \vdots \\ P_{n_{eval}1} & \cdots & P_{n_{eval}m_{eval}} \end{pmatrix}.$$

For a specified set of control points, the least-squares error function between the B-spline surface and sample points is the scalar-valued function:

$$E(P) = \frac{1}{2}\sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\Big|\sum_{j_1=1}^{n}\sum_{j_2=1}^{m}\mathcal{N}_{j_1,p}(\zeta_{k_1})\mathcal{N}_{j_2,q}(\theta_{k_2})P_{j_1j_2} - Q_{k_1k_2}\Big|^2. \tag{4.5}$$

We determine as previously the control points by minimizing the error function.
The first-order partial derivatives are written in terms of the control points $P_{j_1j_2}$ rather than in terms of the components of the control points:

$$\begin{aligned}
\frac{\partial E}{\partial P_{i_1i_2}} &= \frac{\partial}{\partial P_{i_1i_2}}\Big(\frac{1}{2}\sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\Big|\sum_{j_1=1}^{n}\sum_{j_2=1}^{m}\mathcal{N}_{j_1,p}(\zeta_{k_1})\mathcal{N}_{j_2,q}(\theta_{k_2})P_{j_1j_2} - Q_{k_1k_2}\Big|^2\Big)\\
&= \sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\Big(\sum_{j_1=1}^{n}\sum_{j_2=1}^{m}\mathcal{N}_{j_1,p}(\zeta_{k_1})\mathcal{N}_{j_2,q}(\theta_{k_2})P_{j_1j_2} - Q_{k_1k_2}\Big)\mathcal{N}_{i_1,p}(\zeta_{k_1})\mathcal{N}_{i_2,q}(\theta_{k_2})\\
&= \sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\sum_{j_1=1}^{n}\sum_{j_2=1}^{m}\mathcal{N}_{j_1,p}(\zeta_{k_1})\mathcal{N}_{j_2,q}(\theta_{k_2})\mathcal{N}_{i_1,p}(\zeta_{k_1})\mathcal{N}_{i_2,q}(\theta_{k_2})P_{j_1j_2}\\
&- \sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\mathcal{N}_{i_1,p}(\zeta_{k_1})\mathcal{N}_{i_2,q}(\theta_{k_2})Q_{k_1k_2}\\
&= \sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}\sum_{j_1=1}^{n}\sum_{j_2=1}^{m}a_{k_1j_1}b_{k_2j_2}a_{k_1i_1}b_{k_2i_2}P_{j_1j_2} - \sum_{k_1=1}^{n_{eval}}\sum_{k_2=1}^{m_{eval}}a_{k_1i_1}b_{k_2i_2}Q_{k_1k_2},
\end{aligned}$$

where $a_{k_1j_1} = \mathcal{N}_{j_1,p}(\zeta_{k_1})$ and $b_{k_2j_2} = \mathcal{N}_{j_2,q}(\theta_{k_2})$.
These equations may be written in matrix notation as:

$$\frac{\partial E}{\partial P} = A^tAPB^tB - A^tQB, \tag{4.6}$$

where, $A = \big(a_{kj}\big)_{1\le k\le n_{eval},1\le j\le n}$ is a $n_{eval}\times n$ matrix, and $B = \big(b_{kj}\big)_{1\le k\le m_{eval},1\le j\le m}$ is a $m_{eval}\times m$ matrix.
Setting the partial derivatives equal to the zero matrix $0_{\mathbb{R}^{n\times m}}$ leads to the matrix system of equations,

$$A^tAPB^tB - A^tQB = 0_{\mathbb{R}^{n\times m}}. \tag{4.7}$$

Then, we get:

$$\begin{aligned}
P &= \Big(\big(A^tA\big)^{-1}A^t\Big)Q\Big(B\big(B^tB\big)^{-1}\Big)\\
&= \Big(\big(A^tA\big)^{-1}A^t\Big)Q\Big(\big(B^tB\big)^{-1}B^t\Big)^t.
\end{aligned}$$

4.4 B-spline surface fitting - examples

Figures 4.2 and 4.5 show two examples of least squares quadratic B-spline fitting surfaces. These two cases will be used in forthcoming chapters for isogeometric analysis applications. Note that the least-squares fitting method presented previously is applied patch by patch when the computational domain is composed by a set of patches.

The first example is the two-dimensional Gaussian function.

$$f(x,y) = a\exp\Big(-\Big(\frac{(x-x_0)^2}{2\sigma_x^2}+\frac{(y-y_0)^2}{2\sigma_y^2}\Big)\Big) \quad \forall (x,y)\in\Omega=[-1,1]\times[-1,1].$$

Here the coefficient a is the amplitude, x_0, y_0 is the center and σ_x^2, σ_y^2 are the x and y spreads of the blob. Figure 4.2 created using $a = 1$, $(x_0, y_0) = (0,0)$, and $\sigma_x = \sigma_y = 1/\sqrt{2}$.

The physical domain Ω is a set of quadratic Bézier patch with 5×5 patches. The mesh type considered is a simple Cartesian grid. In each patch, we define local data points $(\zeta_{k_1}, \theta_{k_2}, Q_{k_1 k_2}) \quad \forall 1 \le k_1 \le n_{eval}, 1 \le k_2 \le m_{eval}$, with $n_{eval} = m_{eval} = 30$.

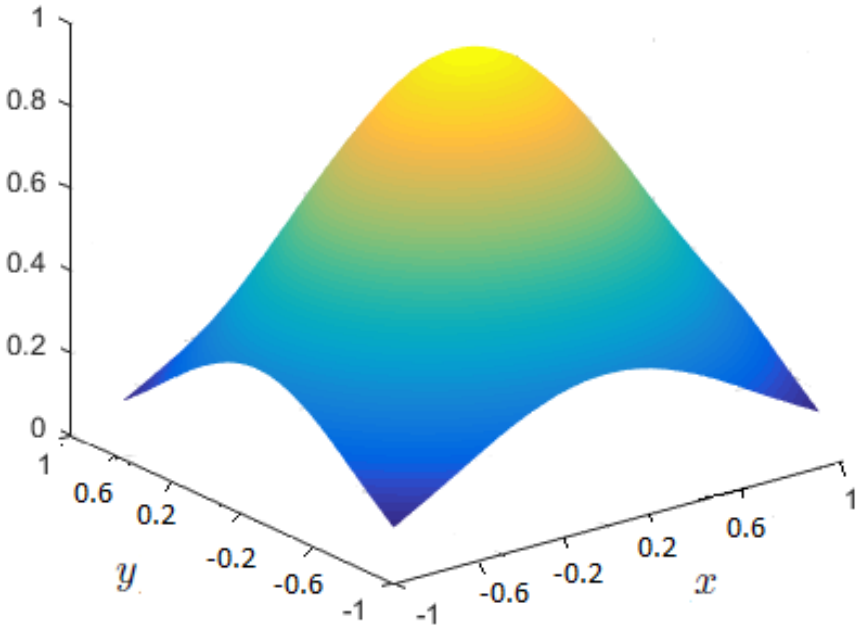


Figure 4.2: The least-squares quadratic B-spline Gaussian surface fitting.

The second example is the Bessel functions of the first kind.

Bessel equation (named according to the astronomer Friedrich Wilhelm Bessel) is a second-order differential equation with two linearly independent solutions: a Bessel function of the first kind of order ν, J_ν and a Bessel function of the second kind of order ν, Y_ν, where ν is a non-negative real number. We refer the reader to [25].

The Bessel function of the first kind $J_\nu(x)$ can be written as an infinite polynomial:

$$J_\nu(x) = \sum_{i=0}^{\infty} \frac{(-1)^i}{i!\Gamma(\nu+i+1)} \left(\frac{x}{2}\right)^{\nu+2i},$$

where Γ is the gamma function, satisfy: $\Gamma(i) = (i-1)!$ for i a positive integer.
Some Bessel functions are plotted in Fig. 4.3.

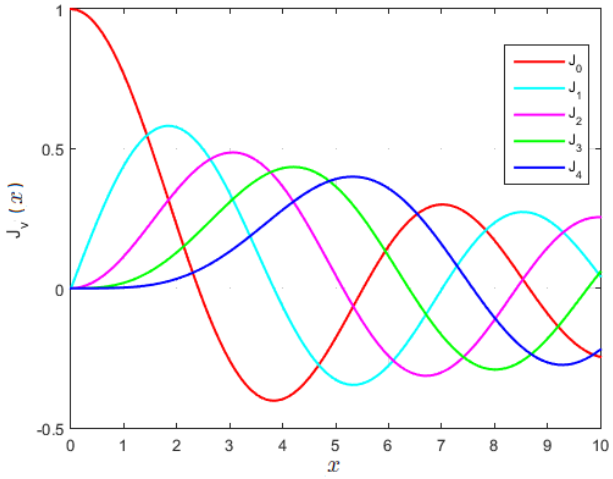


Figure 4.3: Bessel functions of the first kind-1D.

In our example, we consider a fitting a data set of $n_{eval} = 30$ points and $m_{eval} = 30$ points taken from a part of the disc defined by $\big(x(r,\theta), y(r,\theta)\big)$, for $0.25 \le r \le 1$ and $0 \le \theta \le \frac{\pi}{2}$. The computational domain Ω is defined as a set of quadratic Bézier patches with 8×8 patches. The mesh type considered in this example is curvilinear and approximates the geometrical domain. The function to be fitted is defined in cylindrical coordinates as $J_\nu(r)$.

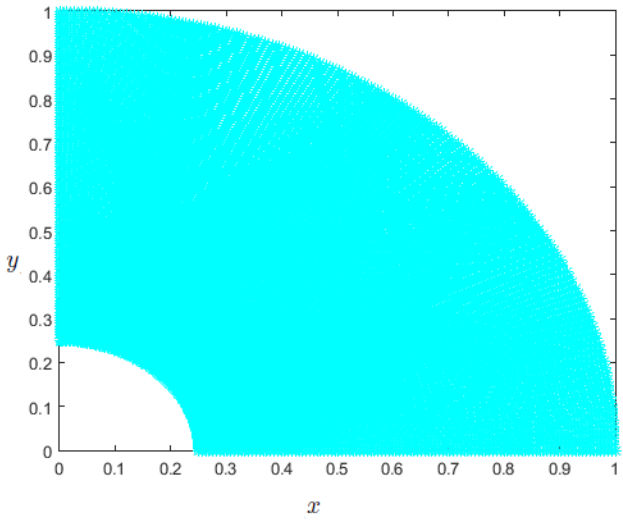


Figure 4.4: Representation of the physical domain Ω.

B-spline fitting curves can be seen in Fig. 4.5, for 8×8 quadratic patches.

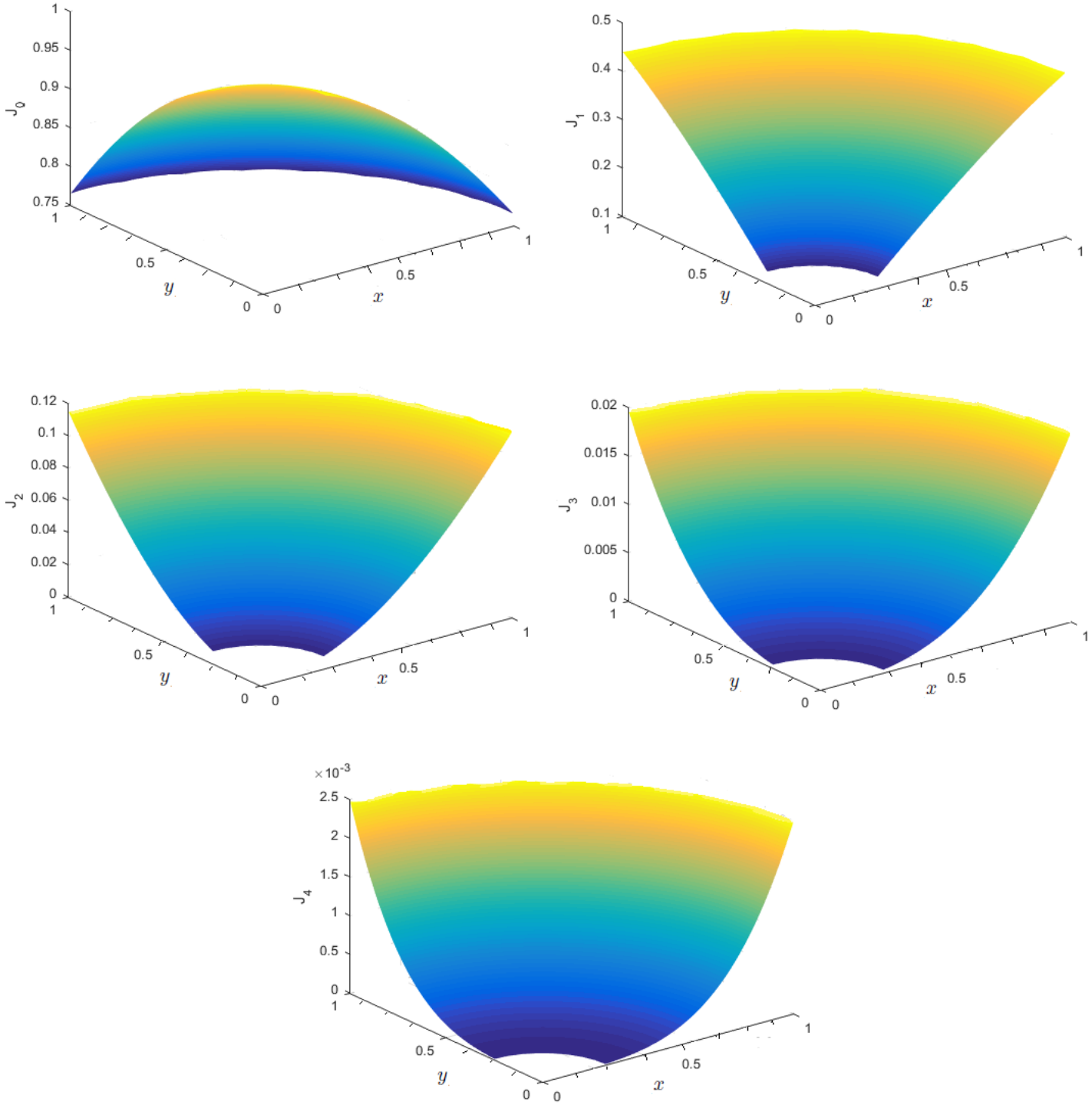


Figure 4.5: The least-squares quadratic B-spline surface fitting.

CONCLUSION

L'ajustement des courbes et des surfaces est un problème fondamental dans de nombreux domaines, tels que l'infographie, le traitement d'images, la modélisation de formes et l'exploration de données. Selon les applications, différents types de courbes, telles que les courbes paramétriques, les courbes implicites et les courbes de subdivision, sont utilisées pour ajuster. Dans ce chapitre, nous avons discuté le problème de l'ajustement de courbes et surfaces B-splines en se basant sur l'approximation par moindres carrés.

La méthode d'approximation par moindres carrés est un outil mathématique, développé indépendamment par Carl Friedrich Gauss en 1795 et Adrien-Marie Legendre qui l'a publié en 1805. Cette théorie décrit une approche fréquemment utilisée à partir de méthodes statistiques pour estimer les valeurs des paramètres d'un modèle mathématique à partir de données mesurées, qui sont sujettes à des erreurs.

Dans le cadre de l'analyse isogéométrique, cette technique est utilisée dans deux cas suivants:

• pour définir le domaine de calcul B-spline si la géométrie du problème n'est pas définie à l'origine en termes de B-spline,

• pour définir les conditions initiales dans l'espace de représentation B-spline.

Part II

ISOGEOMETRIC ANALYSIS - FINITE ELEMENT FRAMEWORK (ILLUSTRATION FOR A $1D$ PROBLEM)

CHAPTER 5

SUPG - FINITE ELEMENT METHOD

THe aim of IGA is to generalize and improve upon classical FEA. In this chapter, we start by giving an introduction to IGA by revisiting the original analysis, i.e. FEA, in the context of hyperbolic PDEs.

The stabilized method Streamline Upwind/Petrov Galerkin (SUPG) [14] [48] [63] had its origin in the late 1970*s* and early 1980*s*. In the present chapter, the capability of this stabilized Finite Element Method (FEM) is illustrated by means of the linear advection problem. A special attention is given to the identification of the stabilization parameter τ of this method which weights the stabilization terms. The need for stabilization is outlined and the basic idea of the Petrov-Galerkin (PG) concept are discussed.

5.1 Preliminaries

For the arguments to follow we would like to introduce some notations and definitions.
The first space we need is the space of square-integrable functions $L^2(\Omega)$ defined by:

$$L^2(\Omega) = \left\{ u \mid \int_\Omega \mid u \mid^2 d\Omega < \infty \right\} \qquad \text{where} \quad \Omega \quad \text{an open internal of} \quad \mathbb{R}.$$

There is a norm related to this space, denoted by $\| \, . \, \|_{L^2(\Omega)}$ which is defined by:

$$\| \, u \, \|_{L^2(\Omega)} = \left(\int_\Omega u^2 d\Omega \right)^{\frac{1}{2}}.$$

We also will need one of the wide family of Sobolev spaces $H^1(\Omega)$ which is defined by:

$$H^1(\Omega) = \left\{ u \in L^2(\Omega) \quad \text{such that} \quad \frac{du}{dx} \in L^2(\Omega) \right\}.$$

$H^1(\Omega)$ is a Hilbert space with inner product

$$(v, w)_{1,\Omega} = \int_\Omega v(x) w(x) dx + \int_\Omega v'(x) w'(x) dx,$$

and the associated norm:

$$\| v \|_{1,\Omega} = \| v \|^2_{L^2(\Omega)} + \| v' \|^2_{L^2(\Omega)} .$$

5.2 Standard Galerkin FEM

Given a one-dimensional domain $\Omega =]\mathrm{a}, \mathrm{b}[$, we consider the one-dimensional hyperbolic model problem of a scalar conservation law, with boundary and initial conditions, which can be written as:

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + \frac{\partial}{\partial x} f(u(x,t)) = 0 & \forall (x,t) \in]\mathrm{a}, \mathrm{b}[\times [0,T], \\ u(x,0) = u_0(x) & \forall x \in]\mathrm{a}, \mathrm{b}[, \\ u(\mathrm{a},t) = u_{\mathrm{a}}(t) & \forall t \in [0,T], \end{cases} \tag{5.1}$$

where u denotes an unknown scalar variable, while $f(u)$ is called flux function and $\frac{\partial u}{\partial t}$ indicates the partial derivative of u with respect to time t, and $t^0 = 0$ indicates initial time. In addition, we represent by $\frac{\partial u}{\partial x}$ the partial derivative of u with respect to space.

The classical variational approach associated with (5.1) is obtained by multiplying this equation by a test function v supposed to be sufficiently regular and by integrating over the domain Ω. We denote by V^1 such space of functions, verifying the boundary conditions:

$$V^1 = \left\{ \mathrm{w} \in H^1(\Omega) \quad \text{such that} \quad \mathrm{w}(\mathrm{a}) = u_{\mathrm{a}} \right\}.$$

We introduce the space $H^1([0,T], V^1)$ of functions v: from $[0,T]$ to V^1, such that $\frac{\partial v}{\partial t} \in L^2([0,T])$.

The weak form of the state equation (5.1) is given by: Find $u \in H^1([0,T], V^1)$ such that:

$$\int_\Omega \frac{\partial u(x,t)}{\partial t} v(x) dx + \int_\Omega \frac{\partial}{\partial x} f(u(x,t)) v(x) dx = 0 \qquad \forall t \in [0,T] \quad \forall v \in H^1(\Omega). \tag{5.2}$$

By integrating by parts and using Dirichlet boundary conditions, the variational formulation reads:

Find $u \in H^1([0,T], V^1)$ such that:

$$\int_\Omega \frac{\partial u(x,t)}{\partial t} v(x) dx = \int_\Omega f(u(x,t)) v'(x) dx + f(u_{\mathrm{a}}) v(\mathrm{a}) - f(u_{\mathrm{b}}) v(\mathrm{b}) \qquad \forall v \in H^1(\Omega). \tag{5.3}$$

As consequence, the weak formulation of the problem (5.1) may be rewritten as:

$$\begin{cases} \text{Find} \quad u \in H^1([0,T], V^1) \quad \text{such that :} \\ a(u,v) = L(v) \qquad\qquad \forall v \in H^1(\Omega), \end{cases} \tag{5.4}$$

where,

$$a(u,v) = \int_\Omega \frac{\partial u(x,t)}{\partial t} v(x) dx - \int_\Omega f(u(x,t)) v'(x) dx,$$

and

$$L(v) = f(u_{\mathrm{a}}) v(\mathrm{a}) - f(u_{\mathrm{b}}) v(\mathrm{b}).$$

5.3 Lagrange $\mathbb{P}_1$ elements

We first consider a mesh of the $1D$ computational domain $\Omega =]\mathrm{a},\mathrm{b}[$, where we want to compute the solution. A mesh is simply a set of points $(x_i)_{1\leq i\leq N}$ or intervals $\Omega_i = [x_i, x_{i+1}]$, $\forall 1 \leq i \leq N-1$ such that:

$$\mathrm{a} = x_1 < x_2 < ... < x_N = \mathrm{b}.$$

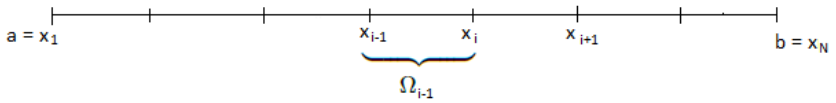


Figure 5.1: Uniform $\mathbb{P}_1$ mesh of $[\mathrm{a},\mathrm{b}]$.

The mesh is said to be uniform if the points $(x_i)_{1\leq i\leq N}$ are uniformly distributed along the segment $[\mathrm{a},\mathrm{b}]$. The cell size or space step is defined by $h_1 = \frac{\mathrm{b}-\mathrm{a}}{N-1}$, where $N-1$ is the number of cells in the mesh. The coordinates of the grid points are then defined by $x_{i+1} = x_1 + ih_1$, $\forall 1 \leq i \leq N-1$.
The FEM for Lagrange $\mathbb{P}_1$ elements involves the space of globally continuous affine functions on each interval:

$$V_h^1 = \left\{ \mathrm{w}_h \in C^0(\Omega) \quad \mathrm{w}_{h|_{\Omega_i}} \in \mathbb{P}_1(\Omega_i), \forall i = 1,2,...,N-1 \quad \text{such that} \quad \mathrm{w}_h(\mathrm{a}) = u_\mathrm{a} \right\},$$

V_h^1 is a subspace of V^1 of dimension N. Moreover, every function $u_h \in V_h^1$ is uniquely determined by:

$$u_h(x,t) = \sum_{j=1}^{N} u_h(x_j,t)\varphi_j(x) \qquad \forall x \in \Omega,$$

where $\forall 2 \leq j \leq N-1$, φ_j is the basis of the shape functions defined as:

$$\varphi_j(x) = \begin{cases} \frac{x-x_{j-1}}{h_1} & \text{if} \quad x \in [x_{j-1}, x_j], \\ \frac{x_{j+1}-x}{h_1} & \text{if} \quad x \in [x_j, x_{j+1}], \\ 0 & \text{otherwise.} \end{cases}$$

These functions are shown in Fig. 5.2. Their construction involves satisfies:

$$\varphi_j(x_i) = \delta_{ij} \quad \forall i,j = 1,...,N.$$

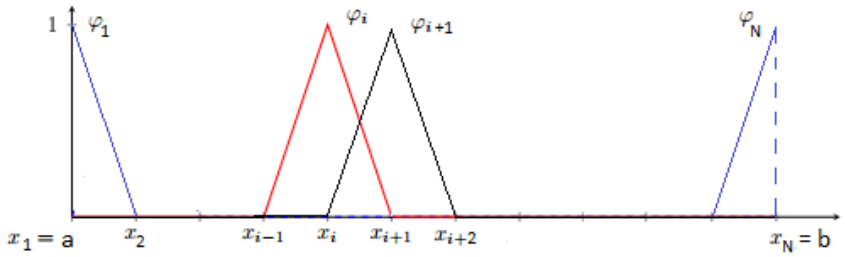


Figure 5.2: Global shape functions for the space V_h^1.

By choosing the test functions v_h equal to the basis functions of V_h^1, one obtains:
Find $u_h \in V_h^1$ such that for all $i = 1, ..., N$,

$$\sum_{j=1}^{N} \frac{\partial u_j(t)}{\partial t} \int_\Omega \varphi_i(x)\varphi_j(x)dx = \int_\Omega f(u_h(x,t))\varphi_i'(x)dx + f(u_a)\varphi_i(a) - f(u_b)\varphi_i(b) \qquad \forall t \in [0,T]. \tag{5.5}$$

Now we rewrite equation (5.1) with $\Omega =]-1,1[$ and $f(u(x,t)) = cu(x,t)$ such that $c > 0$ is the advection velocity. For simplicity in what follows, we assume c to be constant.
Let us consider the problem:

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} & = 0 & \forall (x,t) \in]-1,1[\times[0,T], \\ u(x,0) & = u_0(x) = \sin(2\pi x) & \forall x \in]-1,1[, \\ u(-1,t) & = u_a(t) & \forall t \in [0,T], \end{cases} \tag{5.6}$$

where,

$$u_a(t) = \sin(2\pi(-1-ct)). \tag{5.7}$$

Note that no boundary condition is prescribed at $x = 1$ due to the hyperbolicity of the problem.
The exact solution to this problem is:

$$u_{ex}(x,t) = u_0(x-ct) = \sin(2\pi(x-ct)).$$

We present in Fig. 5.3 the exact solution of the problem (5.6) for a transport speed $c = 1$.

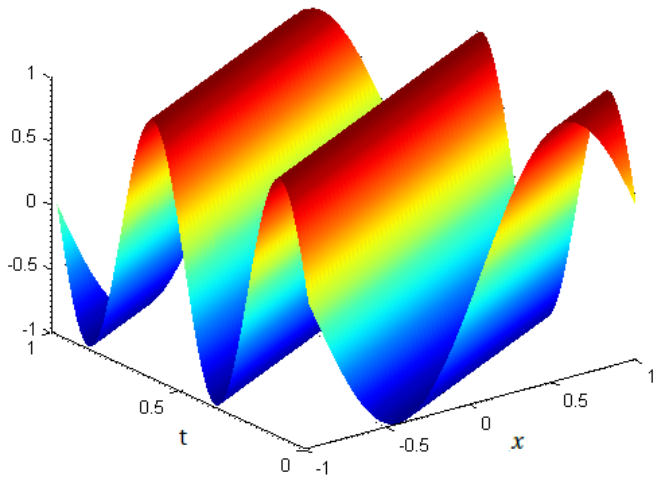


Figure 5.3: The exact sine wave solution for the one-dimensional advection problem.

Thus, we get the discrete weak formulation given by: Find $u_1(t) = u_h(x_1,t),...,u_N(t) = u_h(x_N,t)$ such that for all $i = 1,...,N$

$$\sum_{j=1}^{N} \frac{\partial u_j(t)}{\partial t} \int_\Omega \varphi_i(x)\varphi_j(x)dx = c\sum_{j=1}^{N} u_j(t) \int_\Omega \varphi_j(x)\varphi_i'(x)dx + cu_a\varphi_i(a) - cu(b)\varphi_i(b) \qquad \forall t \in [0,T], \quad (5.8)$$

which can be rewritten in the form of linear system:

$$M\partial_t U = RU + cU_a - cU_b, \quad (5.9)$$

where $U(t)$ is the vector whose components are $u_j(t)$, the unknown values at the grid point at time t, M and R are the mass and stiffness matrices whose coefficients are:

$$M_{ij} = \int_\Omega \varphi_i(x)\varphi_j(x)dx \qquad \forall i,j = 1,...,N,$$

$$R_{ij} = c\int_\Omega \varphi_i'(x)\varphi_j(x)dx \qquad \forall i,j = 1,...,N,$$

and

$$U = \begin{pmatrix} u_1 \\ \vdots \\ u_N \end{pmatrix} \in \mathbb{R}^N, \quad U_a = \begin{pmatrix} \sin(2\pi(-1-ct)) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^N, \quad U_b = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ u_N \end{pmatrix} \in \mathbb{R}^N.$$

In the FEM, the matrices M and R are computed from the corresponding elementary matrices which are obtained by applying a local change of variables to map each cell Ω onto the reference element.
After integration, we get:

$$M = \frac{h_1}{6}\begin{pmatrix} 2 & 1 & 0 & \cdots & \cdots & 0 \\ 1 & 4 & 1 & 0 & \cdots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & 1 & 4 & 1 \\ 0 & 0 & \cdots & 0 & 1 & 2 \end{pmatrix} \quad \text{and} \quad R = \frac{c}{2}\begin{pmatrix} 1 & -1 & 0 & \cdots & \cdots & 0 \\ 1 & 0 & -1 & 0 & \cdots & 0 \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ \ddots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & \cdots & 0 & 1 & 0 & -1 \\ 0 & \cdots & 0 & 0 & 1 & 1 \end{pmatrix}.$$

Therefore, the semi-discrete formulation in space is given by: $\forall 2 \leq i \leq N-1$,

$$\frac{h_1}{6}\big(\partial_t u_{i-1} + 4\partial_t u_i + \partial_t u_{i+1}\big) + c\big(\frac{u_{i+1} - u_{i-1}}{2}\big),$$

where, u_{i-1}, u_i and u_{i+1} are the values of u at nodes $i-1$, i and $i+1$ respectively.
We can conclude that classical Galerkin FEM has similarities with the use of central differenciation in the Finite Difference Method (FDM) for the advection problem for which the advection term writes $c\frac{U_{i+1}-U_{i-1}}{2}$ (identical to $-RU$).

The coefficients u_i are time-dependent while the basis and test functions depend just on spatial coordinates. Further, the time derivative is not discretized in the time domain. One approach would be to use FEM for the time domain as well, but this can be rather computationally expensive. Alternatively, an independent discretization of the time domain is often applied using the method of lines. For example, it is possible to use the explicit Euler scheme:
Given the initial value problem:

$$\begin{cases} \frac{\partial U(x,t)}{\partial t} &= \mathfrak{L}\big(t, U(x,t)\big), \\ U(x,t^0) &= U_0(x), \end{cases} \tag{5.10}$$

we approximate the partial differential equation (5.10) by the finite-difference formulation:

$$\frac{U(x,t+\Delta t) - U(x,t)}{\Delta t} = \mathfrak{L}\big(t, U(x,t)\big).$$

This is exactly one step of the explicit Euler method. Introducing the notation:

$$\begin{cases} t^{\check{n}+1} = t^{\check{n}} + \Delta t, \\ U^{\check{n}} = U(t^{\check{n}}), \end{cases}$$

we have,

$$U^{\check{n}+1} = U^{\check{n}} + \Delta t\,\mathfrak{L}\big(t^{\check{n}}, U^{\check{n}}\big).$$

Euler explicit method is employed to integrate the equations in time with the timestep Δt chosen small enough to ensure that time discretization errors can be neglected. Assuming that M is invertible, we have:

$$U^{\check{n}+1} = U^{\check{n}} + \Delta t M^{-1} R U^{\check{n}} + c\Delta t M^{-1}\left(U_{\mathrm{a}}^{\check{n}} - U_{\mathrm{b}}^{\check{n}}\right). \tag{5.11}$$

This example demonstrates the limitations of the standard Galerkin method for solving advection problem, we observe in Fig. 5.4 that the approximate solution of the problem oscillates.

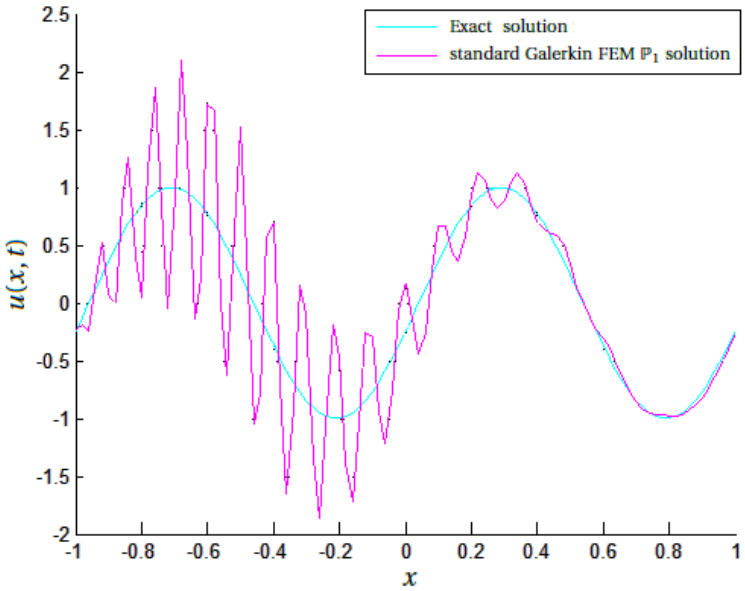


Figure 5.4: Exact and standard Galerkin FEM $\mathbb{P}_1$ solution for the one-dimensional linear advection problem at $T = 0.4s$.

Therefore, standard Galerkin FEM applied to these problems is far from "optimal" and gives unphysical oscillatory solutions, well known as Gibbs phenomena [14]. The attenuation of these oscillations has been the subject of extensive research for several decades during which a huge number of so–called stabilized methods have been developed. The stabilizing effect can be often interpreted as the addition of some artificial diffusion to a standard (unstable) numerical scheme. On the one hand, this artificial diffusion should damp down the oscillations. On the other hand, it should not smear the numerical solution so that the design of a proper stabilization is a very difficult task.

In the late 1970s and early 1980s, a number of methods have been proposed to remove or, at least, to diminish these oscillations (a good summary of the very early literature is contained in [14] [32] [48]). Thus, the attention of FE researchers was turned to the development of Petrov-Galerkin methods (PGM), which are based on searching the test functions that provide exact nodal values for a selected class of solutions. Using these test functions in the general case induces a stabilizing effect which removes the wiggles obtained with the classical Galerkin method [63]. In the context of FEM, there are several approaches, among the most popular techniques, we can name the so called Streamline-Upwind Petrov-Galerkin (SUPG)[13] [84], Galerkin Least-Squares (GLS) [9] [30] [78] and Pressure-Stabilizing Petrov-Galerkin (PSPG) [91]. All these methods are based on a PG FEM. The main idea of all these methods is to add products of suitable perturbation terms and the residuals, thereby maintaining consistency. In these stabilized formulations, a judicious selection of the stabilization parameter, which is often denoted as τ, plays an important role in determining the accuracy of the formulation [1] [15]. This stabilization parameter requires special attention, as it strongly depends on the problem under consideration and the chosen numerical method.
We here give a brief outline of the theoretical basis for PG FEM. Given a differential operator $\mathscr{L}$ and a function G, we consider the problem:

$$\mathscr{L}u(x) = G(x) \qquad \forall x \in \Omega, \tag{5.12}$$

where $\Omega \subset \mathbb{R}$ (without loss of generality on $\mathbb{R}^d$). The weak form of (5.12) is given by:

$$\int_\Omega (\mathscr{L}\widetilde{u} - G)\, v^* \, d\Omega = 0.$$

The problem is discretised based on the following formula:

$$\widetilde{u} = \sum_{j=1}^{N} u_j \varphi_j, \tag{5.13}$$

where $\{\varphi_j\}_{1 \le j \le N}$ are mesh-based shape functions.
Choosing $v^* = \varphi_i$ leads to the classical Galerkin method, whereas for the PGM we can have $v^* \neq \varphi_i$.
In a stabilized FEM, a disturbance is added to the test function of the Galerkin FEM which is given by:

$$v^* = v + \tau \mathscr{L}^*(v),$$

where τ is a nonnegative stabilization parameter and $\mathscr{L}^*$ is a differential operator which may or may not coincide with $\mathscr{L}$. Let us examine some formulations:

Streamline-Upwind/Petrov-Galerkin (SUPG) method: In the context of FEM, a very popular stabilization technique is the SUPG method. This method, developed by Brooks and Hughes [14], can be considered as the first successful stabilization technique to prevent oscillations in convection-dominated problems in the FEM. The basic idea of this method is to add diffusion (or viscosity) which acts only in the flow direction. Extended to a Petrov-Galerkin formulation, the standard Galerkin test functions are modified by adding a streamline upwind perturbation, which again acts only in the flow direction. The modified test function is applied to all terms in the equation, resulting in a consistent weighted residual formulation.

Galerkin/Least-Squares (GLS) method: An alternative stabilization technique, known as Galerkin/Least Squares (GLS) formulation, was introduced in 1988 by Hughes, Franca and Hulbert [91]. It can be interpreted as a generalization of the SUPG method. It is similar to the SUPG in some aspects, and for the hyperbolic equations and/or piecewise linear interpolation functions in the general case, the two methods become identical. The way GLS works is as follows: Least-squares forms of residuals are added to the Galerkin method. These terms enhance the stability of the Galerkin method without degrading accuracy. The result is that practically convenient interpolations, which are unstable within the Galerkin framework, become convergent.

Pressure-Stabilizing Petrov-Galerkin (PSPG) method: Motivated by mathematical analysis, another type of stabilization scheme has been established: the pressure-stabilizing Petrov-Galerkin. It was introduced by Hughes and his collaborators in 1986 for the Stokes problem and incompressible Navier-Stokes equations.

5.4 SUPG FEM for one-dimensional linear advection problem

5.4.1 Selection of the SUPG stabilization parameter

Like many other stabilized methods, the SUPG method contains a stabilization parameter, τ, for which a general "optimal" choice is not known. Since the SUPG method attracted a considerable attention, many research works had also been devoted to the choice of the parameter τ [1] [15]. Theoretical investigations of model problems only provide asymptotic behavior of this parameter (with respect to the local mesh size) and certain bounds for which the SUPG method is stable and leads to (quasi–)optimal convergence of the discrete solution u_h. The choice of τ may dramatically influence the accuracy of the discrete solution and therefore it has been a subject of an extensive research. However, the stabilization parameter τ depends on the problem under consideration and unfortunately, a general optimal definition of τ is still not known. Note that for $1D$ advection-diffusion problems, an optimal value of the stabilization parameter can be defined.

In our computations, for the one-dimensional linear advection problem, we define τ, on any element Ω_i, by the formula:

$$\tau = \alpha \frac{h_i}{c}, \tag{5.14}$$

where, h_i is the length of Ω_i, in the present case is simply $h_i = h_1$ and α is a parameter to be determined in $[0,1]$. We then try to look for the optimal choice of the coefficient $\alpha \in [0,1]$.

5.4.2 SUPG finite element approximation

The standard Galerkin FEM produces non-physical oscillations for the advection problem (5.6), that pollute the whole computational domain. Because of this undesirable feature of the Galerkin method, several approaches have been proposed to cure this problem within the framework of FEMs. In this thesis, we investigate the most favorite one: the SUPG stabilisation method, which adds an additional term to the Galerkin FEM to control the derivatives in the streamline direction. The SUPG weak form of (5.6) can then be written as follows:

$$\int_\Omega \Big(\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x}\Big) v(x)\,dx + \sum_{k=1}^{N-1} \int_{x_k}^{x_{k+1}} \tau c \frac{\partial v(x)}{\partial x} \Big(\frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x}\Big) dx = 0 \qquad \forall v \in V^1. \tag{5.15}$$

For the sake of generality, we adopt here a classical formulation, where stabilization terms are integrated only inside the elements to avoid possible regularity problems at the interfaces.
After integration by parts, one obtains:

$$\int_\Omega \frac{\partial u(x,t)}{\partial t} v(x)\,dx + \sum_{k=1}^{N-1} \tau c \int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial t}\frac{\partial v(x)}{\partial x}\,dx - c\int_\Omega u(x)\frac{\partial v(x)}{\partial x}\,dx + \sum_{k=1}^{N-1} \tau c^2 \int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial x}\frac{\partial v(x)}{\partial x}\,dx$$
$$-c u_{\mathrm{a}} v(\mathrm{a}) + c u_{\mathrm{b}} v(\mathrm{b}) = 0 \quad \forall v \in V^1.$$

Therefore, the weak formulation reads:

$$\begin{cases} \text{Find} \quad u \in H^1([0,T], V^1) \quad \text{such that :} \\ a_{SUPG}(u,v) = L(v) \qquad \forall v \in V^1(\Omega), \end{cases} \tag{5.16}$$

where

$$a_{SUPG}(u,v) = \int_\Omega \frac{\partial u(x,t)}{\partial t} v(x)\,dx - c\int_\Omega u(x,t)\frac{\partial v(x)}{\partial x}\,dx + \tau c \sum_{k=1}^{N-1} \int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial t}\frac{\partial v(x)}{\partial x}\,dx + \tau c^2 \sum_{k=1}^{N-1} \int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial x}\frac{\partial v(x)}{\partial x}\,dx,$$

and

$$L(v) = c u_{\mathrm{a}} v(\mathrm{a}) - c u_{\mathrm{b}} v(\mathrm{b}).$$

Let now the space $H^1(\Omega)$, in which the solution of (5.1) is sought, be approximated by a conforming FE subspace V_h^1.

Thus, the discritized SUPG method reads as follows:

$$\begin{cases} \text{Find} \quad u_h \in H^1([0,T],V^1) \quad \text{such that}: \\ \int_\Omega \frac{\partial u_h(x,t)}{\partial t} v_h(x)dx - c\int_\Omega u_h(x,t)\frac{\partial v_h(x)}{\partial x}dx + \tau c \sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \frac{\partial u_h(x,t)}{\partial t}\frac{\partial v_h(x)}{\partial x}dx + \tau c^2 \sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \frac{\partial u_h(x,t)}{\partial x}\frac{\partial v_h(x)}{\partial x}dx \\ -cu_{\mathrm{a}}(t)v(\mathrm{a}) + cu_{\mathrm{b}}(t)v(\mathrm{b}) = 0. \end{cases}$$

It is important to point out that the SUPG solution with $\tau = 0$ corresponds to the standard semi-discrete Galerkin approximation.

By choosing the test functions v_h as the basis functions of V_h^1, the approximate problem can be written as follows:

$$\begin{cases} \text{Find} \quad u_1, u_2, ..., u_N \quad \text{such} \quad \text{that}: \\ \sum_{j=1}^{N}\Big(\int_\Omega \varphi_j(x)\varphi_i(x)\frac{\partial u_j(t)}{\partial t}dx + \tau c\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}}\varphi_j(x)\varphi_i'(x)u_j(t)dx - c\int_\Omega \varphi_j(x)\varphi_i'(x)u_j(t) \\ +\tau c^2\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}}\varphi_i'(x)\varphi_j'(x)u_j(t)dx\Big) - cu_{\mathrm{a}} + cu_{\mathrm{b}} = 0. \end{cases} \tag{5.17}$$

A consistent SUPG-FE spatial discretization (5.17) leads to the following set of ordinary differential equations:

$$M_1^s \partial_t U = R_1^s U + cU_{\mathrm{a}} - cU_{\mathrm{b}}. \tag{5.18}$$

The elements of mass matrix M_1^s and the stiffness matrix R_1^s are given by:

$$\left(M_1^s\right)_{ij} = \int_\Omega \varphi_j(x)\varphi_i(x)dx + \tau c\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}}\varphi_i'(x)\varphi_j(x)dx \qquad \forall i,j = 1,...,N, \tag{5.19}$$

and

$$\left(R_1^s\right)_{ij} = c\int_\Omega \varphi_i'(x)\varphi_j(x)dx - \tau c^2\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}}\varphi_j'(x)\varphi_i'(x)dx \qquad \forall i,j = 1,...,N. \tag{5.20}$$

By evaluating the integrals, we get:

$$M_1^s = \frac{1}{6}\begin{pmatrix} 2h_1+3\tau c & h_1-3\tau c & 0 & \cdots & \cdots & 0 \\ h_1+3\tau c & 4h_1 & h_1-3\tau c & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & h_1+3\tau c & 4h_1 & h_1-3\tau c \\ 0 & \cdots & \cdots & 0 & h_1-3\tau c & 2h_1+3\tau c \end{pmatrix}.$$

The interpretation of the stabilization term as an additional diffusion is clear here.

$$R_1^s = \begin{pmatrix} \frac{-\tau c^2}{h_1} + \frac{c}{2} & \frac{\tau c^2}{h_1} - \frac{c}{2} & 0 & \cdots & \cdots & 0 \\ \frac{\tau c^2}{h_1} + \frac{c}{2} & \frac{-2\tau c^2}{h_1} & \frac{\tau c^2}{h_1} - \frac{c}{2} & 0 & \cdots & 0 \\ 0 & \ddots & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & \ddots & 0 \\ 0 & \cdots & 0 & \frac{\tau c^2}{h_1} + \frac{c}{2} & \frac{-2\tau c^2}{h_1} & \frac{\tau c^2}{h_1} - \frac{c}{2} \\ 0 & \cdots & \cdots & 0 & \frac{\tau c^2}{h_1} + \frac{c}{2} & \frac{-\tau c^2}{h_1} + \frac{c}{2} \end{pmatrix}.$$

It is noteworthy that M_1^s is not diagonal. It is common in the literature to approximate (5.18) in time by means of explicit time stepping. To avoid having to solve linear systems involving the mass matrix at each time step, it also common to simplify (5.18) by lumping the mass matrix.

5.4.3 Mass lumping

Mass lumping is a numerical technique employed in FEM that has been widely used in different applications (like heat equation, wave equation and time-dependent transport equation). This technique consists of replacing the consistent mass matrix by a diagonal matrix whose entry in row i is the sum of all the entries of the consistent mass matrix in row i, $\forall 1 \leq i \leq N$, usually referred to as the lumped mass matrix (for more details see [36] [90] [91]).

Mass lumping can be shown in one space dimension to be equivalent to approximate the consistent mass matrix by using the following trapezoidal quadrature rule:

$$\int_{x_j}^{x_{j+1}} f(x)dx \approx (x_{j+1} - x_j)\Big(\frac{f(x_i) + f(x_{i+1})}{2}\Big). \tag{5.21}$$

This quadrature is exact for linear polynomials. Using this quadrature, the mass matrix coefficients can be approximated as follows: $\forall 2 \leq i, j \leq N-1$,

$$\begin{aligned} \int_{x_{j-1}}^{x_{j+1}} \varphi_i(x)\varphi_j(x)dx &= \int_{x_{j-1}}^{x_j} \varphi_i(x)\varphi_j(x)dx + \int_{x_j}^{x_{j+1}} \varphi_i(x)\varphi_j(x)dx \\ &\approx \frac{h_1}{2}\Big(\varphi_i(x_i)\varphi_j(x_i)\delta_{i,j} + \varphi_i(x_{i-1})\varphi_j(x_{i-1})\delta_{i-1,j}\Big) + \frac{h_1}{2}\Big(\varphi_i(x_i)\varphi_j(x_i)\delta_{i,j} + \varphi_i(x_{i+1})\varphi_j(x_{i+1})\delta_{i+1,j}\Big) \\ &= \frac{h_1}{2}\Big(\underbrace{(\frac{x_i - x_{i-1}}{h_1})}_{=1}(\frac{x_i - x_{j-1}}{h_1})\delta_{i,j} + \underbrace{(\frac{x_{i-1} - x_{i-1}}{h_1})}_{=0}(\frac{x_{i-1} - x_{j-1}}{h_1})\Big)\delta_{i-1,j} \\ &+ \frac{h_1}{2}\Big(\underbrace{(\frac{x_{i+1} - x_i}{h_1})}_{=1}(\frac{x_{j+1} - x_i}{h_1})\delta_{i,j} + \underbrace{(\frac{x_{i+1} - x_{i+1}}{h_1})}_{=0}(\frac{x_{j+1} - x_{i+1}}{h_1})\Big)\delta_{i+1,j} \\ &= h_1\delta_{ij}. \end{aligned}$$

Where δ_{ij} is the Kronecker symbol.

$$\begin{aligned}\tau c\int_{x_{j-1}}^{x_{j+1}}\varphi_i'(x)\varphi_j(x)dx &= \tau c\int_{x_{j-1}}^{x_j}\varphi_i'(x)\varphi_j(x)dx+\tau c\int_{x_j}^{x_{j+1}}\varphi_i'(x)\varphi_j(x)dx\\ &= \tau c\frac{h_1}{2}\Big(\frac{1}{h_1}\frac{x_{j-1}-x_{j-1}}{h_1}+\frac{1}{h_1}\frac{x_j-x_{j-1}}{h_1}\Big)+\tau c\frac{h_1}{2}\Big(\frac{-1}{h_1}\frac{x_{j+1}-x_j}{h_1}-\frac{1}{h_1}\frac{x_{j+1}-x_{j+1}}{h_1}\Big)\\ &= \tau c\frac{h_1}{2}(\frac{1}{h_1}-\frac{1}{h_1})=0.\end{aligned}$$

Hence, the coefficients of the lumped mass matrix are:

$$\big(M_L^s\big)_{ij}=h_1\delta_{ij}\quad\forall 2\le i,j\le N-1$$

This technique of mass lumping presents a computational advantage as we need mass matrix inversion. Upon replacing the consistent mass matrix M_1^s by the lumped mass matrix M_L^s, we obtain a new matrix form of advection equation which is fully explicit and writes as follows:

$$M_L^s\partial_t U=R_1^sU+cU_{\mathrm{a}}-cU_{\mathrm{b}}. \tag{5.22}$$

$$\partial_t U=(M_L^s)^{-1}R_1^sU+c(M_L^s)^{-1}\Big(U_{\mathrm{a}}-U_{\mathrm{b}}\Big),$$

where, the so-called lumped mass matrix M_L^s thus computed is diagonal, is given by:

$$M_L^s=\frac{h_1}{2}\begin{pmatrix}1&0&0&\cdots&0\\0&2&0&\cdots&0\\\ddots&\ddots&\ddots&\ddots&\\0&\cdots&0&2&0\\0&\cdots&0&0&1\end{pmatrix}.$$

5.4.4 Runge-Kutta time discretization

The space semidiscrete problem (5.22) represents a system of ordinary differential equations (ODEs). This system can be solved by any of the available ODE solvers. In this thesis we focus on the Runge Kutta schemes. These methods have been developed [34] [80] for solving:

$$\partial_t U=\mathfrak{L}\big(t,U(x,t)\big),$$

where $\mathfrak{L}\big(t,U(x,t)\big)$ is a spatial discretization operator.
We divide the time interval $(0,T)$ into $\check{n}$ time steps $[t^k,t^{k+1}]\ \forall k=0,...,\check{n}-1$, where $t^0=0$ and $t^{\check{n}}=T$ and denote the step length of interval $[t^k,t^{k+1}]$ by $\Delta t=t^{k+1}-t^k$.

First-order Runge-Kutta formula ($RK1$)**:**

$$U^{\check{n}+1} = U^{\check{n}} + \Delta t \mathfrak{L}\big(t, U^{\check{n}}\big).$$

Second-order Runge-Kutta formula ($RK2$)**:**
We define an intermediate estimate $U^{\check{n}+\frac{1}{2}}$:

$$U^{\check{n}+\frac{1}{2}} = U^{\check{n}} + \frac{\Delta t}{2}\mathfrak{L}\big(t, U^{\check{n}}\big),$$

$$U^{\check{n}+1} = U^{\check{n}} + \Delta t \mathfrak{L}\big(t + \frac{\Delta t}{2}, U^{\check{n}+\frac{1}{2}}\big).$$

Third-order Runge-Kutta formula ($RK3$)**:**

$$U^{\check{n}+1} = U^{\check{n}} + \frac{\Delta t}{6}\big(K_1 + 4K_2 + K_3\big),$$

where,
$K_1 = \mathfrak{L}\big(t^{\check{n}}, U^{\check{n}}\big)$,

$K_2 = \mathfrak{L}\big(t^{\check{n}} + \frac{\Delta t}{2}, U^{\check{n}} + \frac{\Delta t}{2}K_1\big)$,

$K_3 = \mathfrak{L}\big(t^{\check{n}} + \Delta t, U^{\check{n}} - \Delta t K_1 + 2\Delta t K_2\big)$.

Fourth-order Runge-Kutta formula ($RK4$)**:**

$$U^{\check{n}+1} = U^{\check{n}} + \frac{\Delta t}{6}\big(K_1 + 2K_2 + 2K_3 + K_4\big),$$

where,
$K_1 = \mathfrak{L}\big(t^{\check{n}}, U^{\check{n}}\big)$,

$K_2 = \mathfrak{L}\big(t^{\check{n}} + \frac{\Delta t}{2}, U^{\check{n}} + \frac{\Delta t}{2}K_1\big)$,

$K_3 = \mathfrak{L}\big(t^{\check{n}} + \frac{\Delta t}{2}, U^{\check{n}} + \frac{\Delta t}{2}K_2\big)$,

$K_4 = \mathfrak{L}\big(t^{\check{n}} + \Delta t, U^{\check{n}} + \Delta t K_3\big)$.

In all these formula, $U^{\check{n}+1}$ is an approximation to $U(t^{\check{n}+1})$ and K_1, K_2, K_3 and K_4 are the intermediate evaluations.

5.4.5 Courant-Friedrichs-Lewy (CFL) condition

During the time integration cycles, the length of the time step needs to be chosen, according to a stability criterion ruled by the C_{CFL} number. The concept of the C_{CFL} number was originally published in 1928, the aim was to prove the existence of solutions of some PDEs. Consequently, while proving the existence, Courant, Friedrichs and Lewy found the necessary condition to stabilize the numerical methods. For more detailed discussion of the C_{CFL} number see [45]. The Courant Friedrichs-Lewy (CFL) condition is given by:

$$|\, c\,| \frac{\Delta t}{h_1} \leq C_{CFL},$$

where Δt is the time step, c is the speed and h_1 is the length for the computational element.
Note that this condition may not be sufficient in the presence of the SUPG stabilization term, for which a diffusion stability criterion has to be taken into account.

5.5 Numerical results

In this section, the performance of the SUPG method will be illustrated for the linear one-dimensional advection problem. We use equidistant grids with mesh size $h_1 = \frac{2}{N-1}$. It is known that standart Galerkin discretization results in a strongly non-stable scheme which then leads to a numerical solution exhibiting non-physical phenomena such as spurious numerical oscillations. We get such results on the left in Fig. 5.5. On the right, the SUPG numerical solution is stable thanks to the additional diffusion term.

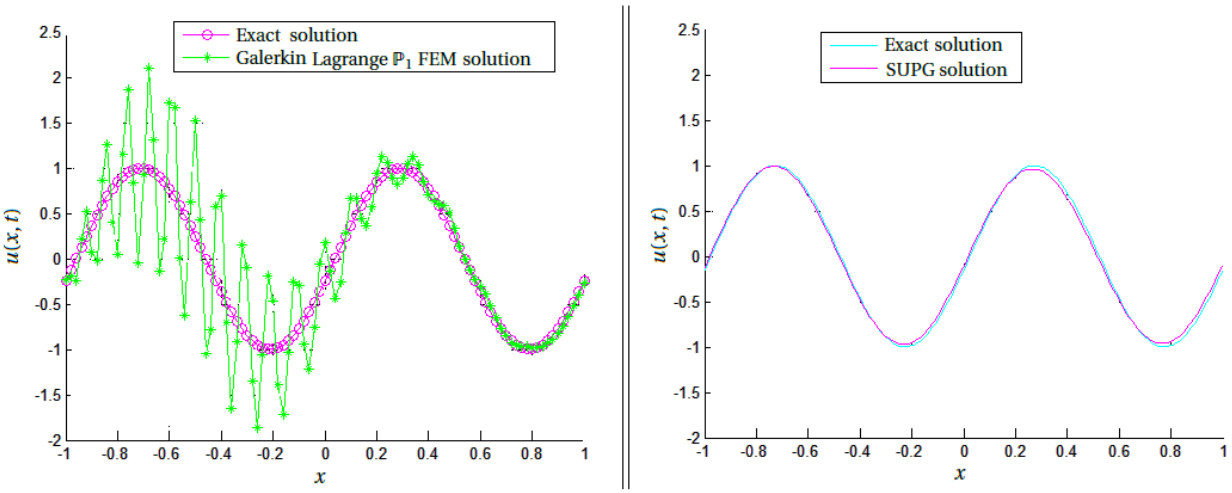


Figure 5.5: Exact, Galerkin and SUPG solutions at $T = 0.4s$.

The time stepping is done with the standard $RK2$ method; this ensures that the error induced by the time approximation is small compared to the spatial error. The solution is computed at $T = 0.4$. Special attention is given to the role of the stabilization parameter τ (more precisely of the coefficient α) given by the formula: $\tau = \alpha \frac{h_1}{c}$.

5.5.1 Influence of the SUPG parameter

The SUPG FEM contains a stabilization parameter τ for which a general "optimal" choice is not known. It is pointed out that the optimal choice of this parameter is still an open question. The aim of this section is to describe how the SUPG stabilization parameter impacts the solution for the one-dimensional advection problem. The influence of this parameter on the numerical results is illustrated in Fig. 5.6.

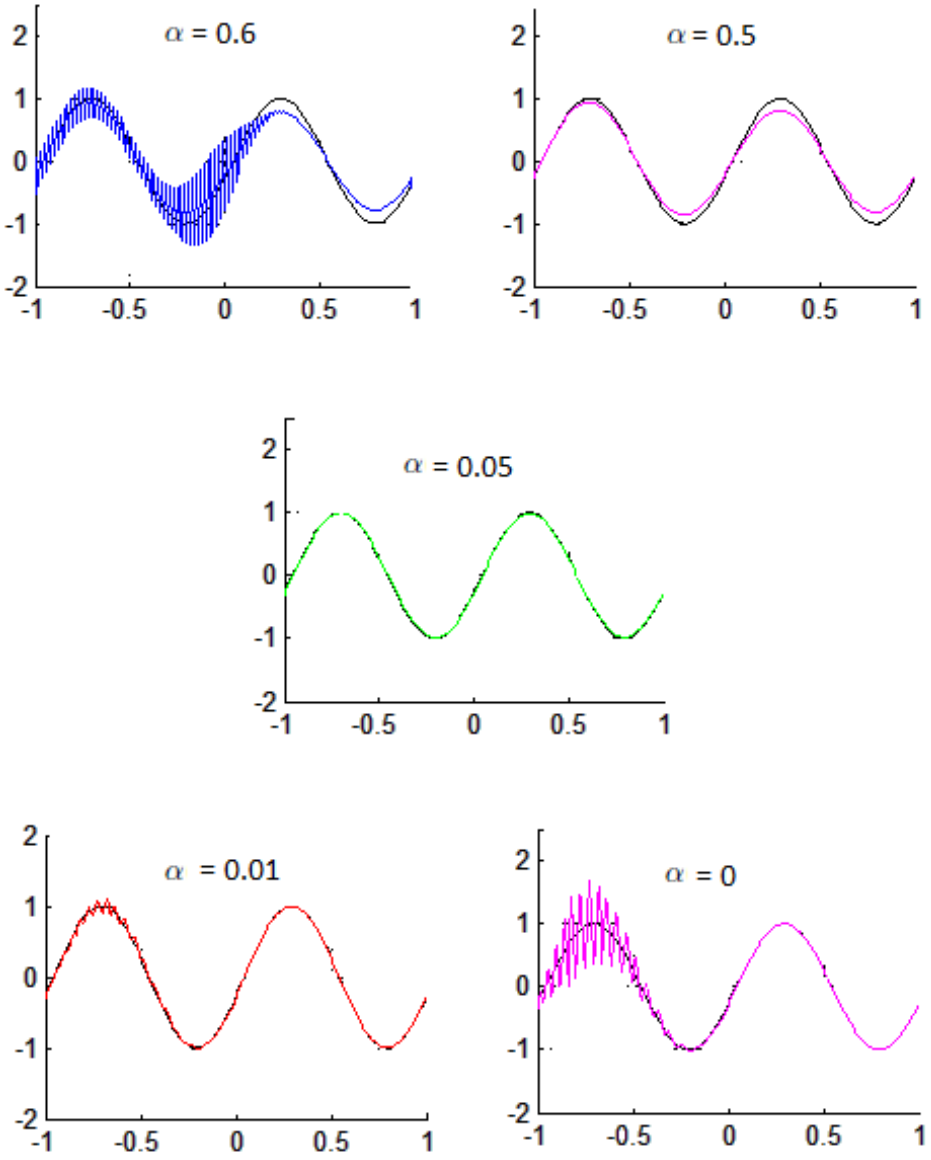


Figure 5.6: SUPG FE $\mathbb{P}_1$ solution for different values of $\alpha \in [0,1]$.

We can see clearly in Fig. 5.6 important oscillations for $0 \leq \alpha \leq 0.01$ and $\alpha \geq 0.6$. For α too small, the stabilization effect is not strong enough to counterbalance the instability of Galerkin scheme. When α is too large, oscillations are due to the violation of the stability criteria in RK time integration for diffusive terms. For $0.01 < \alpha \leq 0.5$, we do not observe oscillations. However, a too large additional diffusion ($\alpha = 0.5$) impacts the accuracy of the solution, even if the scheme is stable. In what follows, we propose to determine the optimal choice by studying the L^2-error.

5.5.2 Error Estimates for the SUPG FE method

We want to evaluate the numerical behavior of the SUPG linear FEM for the problem (5.1), where an analytic solution is explicitly known. The interest here is in the evaluation of the errors committed for different mesh steps h_1 as well as the order of convergence of the method. A mesh convergence study allows to test the accuracy of the numerical method by numerically evaluating an order of accuracy. The error between the approximate solution u_h and the analytic solution u_{ex} for a final time T is determined in the standard L^2-norm:

$$\begin{aligned} \| e(T) \|_{L^2(\Omega)} &= \| u_{ex}(x,T) - u_h(x,T) \|_{L^2(\Omega)} \\ &= \left(\int_\Omega \Big(u_{ex}(x,T) - u_h(x,T) \Big)^2 dx \right)^{\frac{1}{2}} \\ &= \left(\sum_{k=1}^{N-1} \int_{x_k}^{x_{k+1}} \Big(\sin(2\pi(x - cT)) - u_h(x,T) \Big)^2 dx \right)^{\frac{1}{2}}. \end{aligned}$$

Before we proceed further, let us denote by r the convergence rate of the SUPG FEM. We assume that a norm $\| e(T) \|_{L^2(\Omega)}$ of the computational error behaves according to the formula:

$$\| e(T) \|_{L^2(\Omega)} = C_r h_1^r,$$

where $C_r > 0$ is a constant, h_1 the mesh size and $r \in \mathbb{R}$.
Let $\| e_1(T) \|_{L^2(\Omega)}$ and $\| e_2(T) \|_{L^2(\Omega)}$ be computational errors of the numerical solutions obtained on two different meshes h_1 and h_2, respectively. Then we obtain:

$$r = \frac{\ln\big(\| e_2(T) \|_{L^2(\Omega)} \big) - \ln\big(\| e_1(T) \|_{L^2(\Omega)} \big)}{\ln(h_2) - \ln(h_1)}.$$

Table 5.1 shows the error of the SUPG-$\mathbb{P}_1$FEM numerical approximations in L^2-norm for various number of degrees of freedom (d.o.f.) (notice that the number of d.o.f. is equal to N) and coefficient of the stabilization parameter $\alpha \in [0,1]$. The L^2-errors of the numerical approximations are depicted in Fig. 5.7 for different values of α in log scale.

$\| e(T) \|_{L^2}$							
α	$d.o.f. = 16$	$d.o.f. = 32$	$d.o.f. = 64$	$d.o.f. = 128$	$d.o.f. = 256$	$d.o.f. = 512$	$d.o.f. = 1024$
1	$8.001E+003$	$1.706E+008$	$1.703E+017$	$3.509E+035$	$2.997E+072$	$4.359E+146$	–
0.9	$7.027E+002$	$1.237E+006$	$7.943E+012$	$6.610E+026$	$9.141E+054$	$3.474E+111$	–
0.8	$0.475E+002$	$5.341E+003$	$1.366E+008$	$1.694E+017$	$5.118E+035$	$9.248E+072$	$5.991E+147$
0.7	2.464	13.560	$8.560E+002$	$6.146E+006$	$5.871E+014$	$1.035E+031$	$6.346E+063$
0.6	0.846	0.656	0.434	0.255	0.139	$7.299E-002$	$3.737E-002$
0.5	0.807	0.598	0.381	0.218	0.117	$6.131E-002$	$3.127E-002$
0.4	0.752	0.526	0.322	0.180	$9.581E-002$	$4.943E-002$	$2.511E-002$
0.3	0.667	0.435	0.255	0.139	$7.298E-002$	$3.737E-002$	$1.891E-002$
0.2	0.538	0.321	0.179	$9.571E-002$	$4.941E-002$	$2.511E-002$	$1.266E-002$
0.1	0.362	0.177	$9.491E-002$	$4.925E-002$	$2.508E-002$	$1.265E-002$	$6.355E-003$
0.09	0.345	0.161	$8.585E-002$	$4.444E-002$	$2.260E-002$	$1.139E-002$	$5.722E-003$
0.08	0.330	0.145	$7.669E-002$	$3.961E-002$	$2.012E-002$	$1.013E-002$	$5.088E-003$
0.07	0.317	0.129	$6.745E-002$	$3.475E-002$	$1.762E-002$	$8.876E-003$	$4.453E-003$
0.06	0.308	0.113	$5.814E-002$	$2.986E-002$	$1.513E-002$	$7.613E-003$	$3.818E-003$
0.05	0.304	$9.779E-002$	$4.880E-002$	$2.495E-002$	$1.262E-002$	$6.348E-003$	$3.183E-003$
0.04	0.305	$8.323E-002$	$3.948E-002$	$2.001E-002$	$1.011E-002$	$5.082E-003$	$2.547E-003$
0.03	0.313	$7.084E-002$	$3.034E-002$	$1.510E-002$	$7.597E-003$	$3.813E-003$	$1.911E-003$
0.02	0.329	$6.257E-002$	$2.180E-002$	$1.024E-002$	$5.085E-003$	$2.569E-003$	$9.900E-003$
0.01	0.353	$6.107E-002$	$1.53E-002$	$5.669E-003$	$2.662E-003$	$1.837E-002$	84.784

Table 5.1: L^2-errors of the SUPG FE $\mathbb{P}_1$ method for the one-dimensional advection problem.

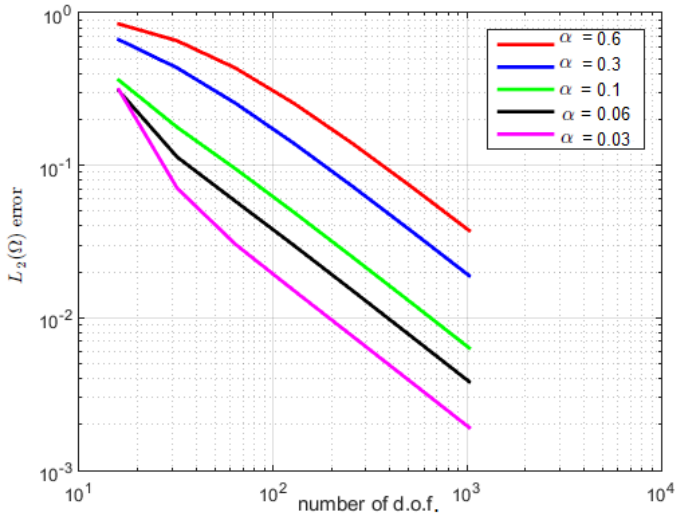


Figure 5.7: Convergence in number of d.o.f. for different choices of α.

We observe that the convergence rates obtained are lower than the optimal rate expected (of value 2). As shown in [58] [91], this reduction of the convergence rate is due to the use of the mass lumping procedure, in addition to the use of the stabilization term (for some τ values).

α	$d.o.f. = 16$	$d.o.f. = 32$	$d.o.f. = 64$	$d.o.f. = 128$	$d.o.f. = 256$	$d.o.f. = 512$	$d.o.f. = 1024$
0.6	–	0.36	0.59	0.76	0.87	0.93	0.97
0.5	–	0.3	0.65	0.8	0.91	0.93	0.97
0.4	–	0.51	0.7	0.84	0.91	0.95	0.98
0.3	–	0.61	0.77	0.87	0.93	0.97	0.98
0.2	–	0.74	0.84	0.91	0.94	0.97	1
0.1	–	1.03	0.92	0.94	0.97	1	1
0.09	–	1.12	0.91	0.95	0.77	1.2	1
0.08	–	1.2	0.92	0.95	0.98	1	1.02
0.07	–	1.3	0.94	0.96	0.98	1	1
0.06	–	1.5	0.96	0.96	0.98	1	1
0.05	–	1.6	1	0.98	0.98	1	1
0.04	–	1.7	1.07	0.98	0.99	0.98	1
0.03	–	2	1.22	1	1	0.99	1

Table 5.2: Convergence rates.

5.6 SUPG FE method for high-order elements

In some applications, the affine approximation on each element of the mesh $\Omega_i = [x_i, x_{i+1}]$ can be considered as not good enough in the sense that it provides an approximate function that is too far from the exact function u. To overcome this problem, we can approximate u on each mesh by polynomials of higher degree. We now extend the above considerations to higher-order finite-elements. We particularly focus our attention in this section on the FE approximation $\mathbb{P}_2$, which consists in approaching the solution u by a continuous function on Ω and a polynomial of degree 2 on each element Ω_i.

Now, let us revisit the one-dimensional advection problem that is given in (5.6) and we examine the quadratic Lagrange bases functions. These are constructed by adding an extra node $x_{i+\frac{1}{2}}$ at the midpoints of each $\Omega_i = [x_i, x_{i+1}]$,

$$x_{i+\frac{1}{2}} = x_i + \frac{h_1}{2}, \quad \forall 1 \leq i \leq N-1.$$

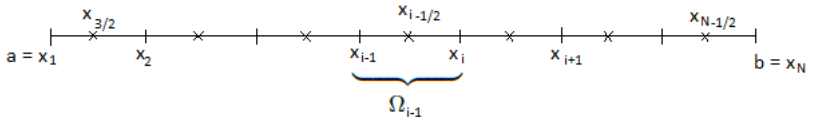


Figure 5.8: Uniform $\mathbb{P}_2$ mesh of [a, b].

The FEM for Lagrange $\mathbb{P}_2$ elements involves the discrete space:

$$V_h^2 = \left\{ w_h \in C^0([-1,1]) \quad w_{h|_{\Omega_i}} \in \mathbb{P}_2 \quad \forall 1 \le i \le N-1 \quad \text{such that} \quad w_h(a) = u_{2a} \right\}.$$

These spaces are composed of piecewise polynomials functions (polynomials of degree less than or equal to 2). The $\mathbb{P}_2$ FEM consists in applying the internal variational approximation approach to these spaces. V_h^2 is a subspace of $H^1(\Omega)$ of dimension $2N-1$. As with the piecewice-linear basis, one basis function is associated with each node. Those associated with vertices are:

$$\phi_j(x) = \begin{cases} 2\frac{(x-x_{j-1})(x-x_{j-\frac{1}{2}})}{h_1^2} & \text{if} \quad x_{j-1} \le x \le x_j, \\ 2\frac{(x_{j+1}-x)(x_{j+\frac{1}{2}}-x)}{h_1^2} & \text{if} \quad x_j \le x \le x_{j+1}, \\ 0 & \text{otherwise,} \end{cases} \tag{5.23}$$

and those associated with element midpoint are:

$$\phi_{j+\frac{1}{2}}(x) = \begin{cases} 4\frac{(x_{j+1}-x)(x-x_j)}{h_1^2} & \text{if} \quad x_j \le x \le x_{j+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{5.24}$$

These functions are shown in Fig. 5.9. Their construction (to be described) involves satisfying:

$$\begin{cases} \phi_j(x_i) = \delta_{ij} \\ \phi_j(x_{i+\frac{1}{2}}) = 0 \end{cases} \quad \text{and} \quad \begin{cases} \phi_{j+\frac{1}{2}}(x_i) = 0 \\ \phi_{j+\frac{1}{2}}(x_{i+1}) = \delta_{ij} \end{cases}$$

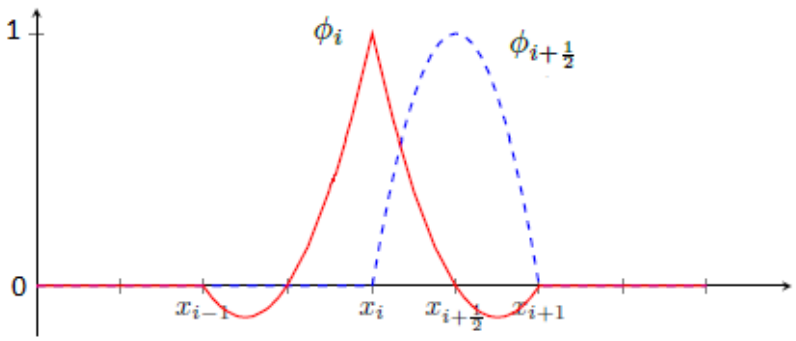


Figure 5.9: Global shape functions for the space V_h^2.

The SUPG variational formulation of the problem (5.6) consists in finding $u_h \in H^1([0,T], V_h^2)$. Every function $u_h \in V_h^2$ defined by its values at the mesh vertices $(x_i)_{1\le i\le N}$ and at the midpoints $(x_{i+\frac{1}{2}})_{1\le i\le N-1}$:

$$u_h(x,t) = \sum_{j=1}^{N} u_j(t)\phi_j(x) + \sum_{j=1}^{N-1} u_{j+\frac{1}{2}}(t)\phi_{j+\frac{1}{2}}(x) \qquad \forall x \in \Omega.$$

Furthermore, since it is enough to use basis functions of V_h^2 as test functions v, the discrete weak formulation can be rewritten in form of linear differential system, wich can be described as follows:

$$M_2^s \partial_t U = R_2^s U + c\Big(U_{2\mathrm{a}} - U_{2\mathrm{b}}\Big). \tag{5.25}$$

Since the shape functions ϕ_i have a compact support, the matrices M_2^s and R_2^s are mostly composed of zeros. In contrast to the Lagrange $\mathbb{P}_1$ FEM, the matrices M_2^s and R_2^s are no longer tridiagonal matrices. Their elements are given by:

$$\begin{aligned}\left(M_2^s\right)_{ij} &= \int_\Omega \phi_j(x)\phi_i(x)dx + \int_\Omega \phi_{j+\frac{1}{2}}(x)\phi_{i+\frac{1}{2}}(x)dx \\ &+ \tau c\Big(\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \phi_i'(x)\phi_j(x)dx + \sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \phi'_{i+\frac{1}{2}}(x)\phi_{j+\frac{1}{2}}(x)dx\Big),\end{aligned}$$

and

$$\begin{aligned}\left(R_2^s\right)_{ij} &= c\Big(\int_\Omega \phi_i'(x)\phi_j(x)dx + \int_\Omega \phi'_{i+\frac{1}{2}}(x)\phi_{j+\frac{1}{2}}(x)dx\Big) \\ &- \tau c^2\Big(\sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \phi_j'(x)\phi_i'(x)dx + \sum_{k=1}^{N-1}\int_{x_k}^{x_{k+1}} \phi'_{j+\frac{1}{2}}(x)\phi'_{i+\frac{1}{2}}(x)dx\Big),\end{aligned}$$

where,

$$U_{2\mathrm{a}} = \begin{pmatrix} \sin(2\pi(-1-ct)) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{2N-1} \qquad \text{and} \qquad U_{2\mathrm{b}} = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ u_N \end{pmatrix} \in \mathbb{R}^{2N-1}$$

In this case, the elementary contributions of the element $\Omega_i = [x_i, x_{i+1}]$ to the stiffness matrix and to the mass matrix are given by the 3×3 elementary matrices $R^s_{2e|\Omega_i}$ and $M^s_{2e|\Omega_i}$:

$$R^s_{2e|\Omega_i} = \begin{pmatrix} \frac{-7\tau c^2}{3h_1} & \frac{-2c}{3} + \frac{8\tau c^2}{3h_1} & \frac{c}{6} - \frac{\tau c^2}{3h_1} \\ \frac{2c}{3} + \frac{8\tau c^2}{3h_1} & \frac{-16\tau c^2}{3h_1} & \frac{-2c}{3} + \frac{8\tau c^2}{3h_1} \\ \frac{-c}{6} - \frac{\tau c^2}{3h_1} & \frac{2c}{3} + \frac{8\tau c^2}{3h_1} & \frac{-7\tau c^2}{3h_1} \end{pmatrix},$$

and

$$M^s_{2e|\Omega_i} = \begin{pmatrix} \frac{2h_1}{15} & \frac{h_1}{15} - \frac{2\tau c}{3} & \frac{-h_1}{30} + \frac{\tau c}{6} \\ \frac{h_1}{15} + \frac{2\tau c}{3} & \frac{8h_1}{15} & \frac{h_1}{15} - \frac{2\tau c}{3} \\ \frac{-h_1}{30} - \frac{\tau c}{6} & \frac{h_1}{15} + \frac{2\tau c}{3} & \frac{2h_1}{15} \end{pmatrix}.$$

In the present case of $\mathbb{P}_2$ elements, Simpson's rule can be derived. In particular, let the basis function ϕ_j be tabulated at points x_j, x_{j+1} $\forall 1 \leq j \leq N-1$ (equally spaced by distance h_1), and midpoint $x_{j+\frac{1}{2}} = \frac{x_j + x_{j+1}}{2}$. Then Simpson's rule states that:

$$\int_{x_j}^{x_{j+1}} f(x)dx \approx \frac{x_{j+1} - x_j}{6}\Big(f(x_j) + 4f(x_{j+\frac{1}{2}}) + f(x_{j+1})\Big). \tag{5.26}$$

Since it uses quadratic polynomials to approximate functions, Simpson's rule actually gives exact results when approximating integrals of polynomials up to cubic degree.

Hence, the so-called lumped mass matrix M^s_2 thus computed is diagonal and is given by:

$$M^s_{2L|\Omega_i} = \frac{1}{6}\begin{pmatrix} h_1 - 3\tau c & 0 & 0 \\ 0 & 4h_1 & 0 \\ 0 & 0 & h_1 + 3\tau c \end{pmatrix}.$$

Upon replacing the consistent mass matrix M^s_2 by the lumped mass matrix M^s_{2L}, we rewrite the matrix form 5.25 as follows:

$$\partial_t U = \big(M^s_{2L}\big)^{-1} R^s_2 U + c\big(M^s_{2L}\big)^{-1}\Big(U_{2\mathrm{a}} - U_{2\mathrm{b}}\Big). \tag{5.27}$$

5.6.1 Matrix assembly

The construction of the global mass and stiffness matrices, M_{2L}^s and R_2^s is often referred to as "assembling" due to its method of construction. The assembly is obtained algorithmically using a loop over all mesh elements Ω_i, $\forall 1 \le i \le N-1$ (which is composed of three nodes x_j, $x_{j+\frac{1}{2}}$ and x_{j+1}) and adding their contributions to the corresponding coefficients of the global system. It is more convenient to loop over each element instead of looping over all basis functions at the outer loop.
The algorithm of constructing the global mass and stiffness matrices M_{2L}^s and R_2^s can be described as follows:

Pseudo-code of matrix assembly:

1. For $k = 1 : N-1$
2. For $i = 1 : 3$
3. $i_g = 2(k-1) + i$
4. For $j = 1 : 3$
5. $j_g = 2(k-1) + j$
6. $M_{2L}^s(i_g, j_g) = M_{2L}^s(i_g, j_g) + m_{2e}^s(i, j)$
7. $R_2^s(i_g, j_g) = R_2^s(i_g, j_g) + r_{2e}^s(i, j)$
8. End (loop j)
9. End (loop i)
10. End (loop k)

The space semidiscrete problem (5.25) represents a system of ODEs, which has to be solved with the RK schemes. The choice of the timestep that ensures a stable scheme, for the quadratic Lagrange $\mathbb{P}_2$ SUPG FE method is based on the condition:

$$\Delta t \le \frac{h_1 C_{CFL}}{2 \mid c \mid}.$$

5.6.2 Numerical results

To give an illustration of the SUPG quadratic Lagrange FEM in conjunction with $RK2$ method for the temporal discretization, some numerical results are presented in this section. The exact and numerical solutions of the problem (5.6) investigated by these methods, are depicted in Fig. 5.10 for different choices of the coefficient α. The numerical behavior of the method is evaluated on the spacial domain $[-1,1]$, which is subdivided into $N-1$ uniformly distributed subintervals with a number of $d.o.f. = 2N-1$, and time domain $[0,T]$ where $T>0$ is considered as the final time.

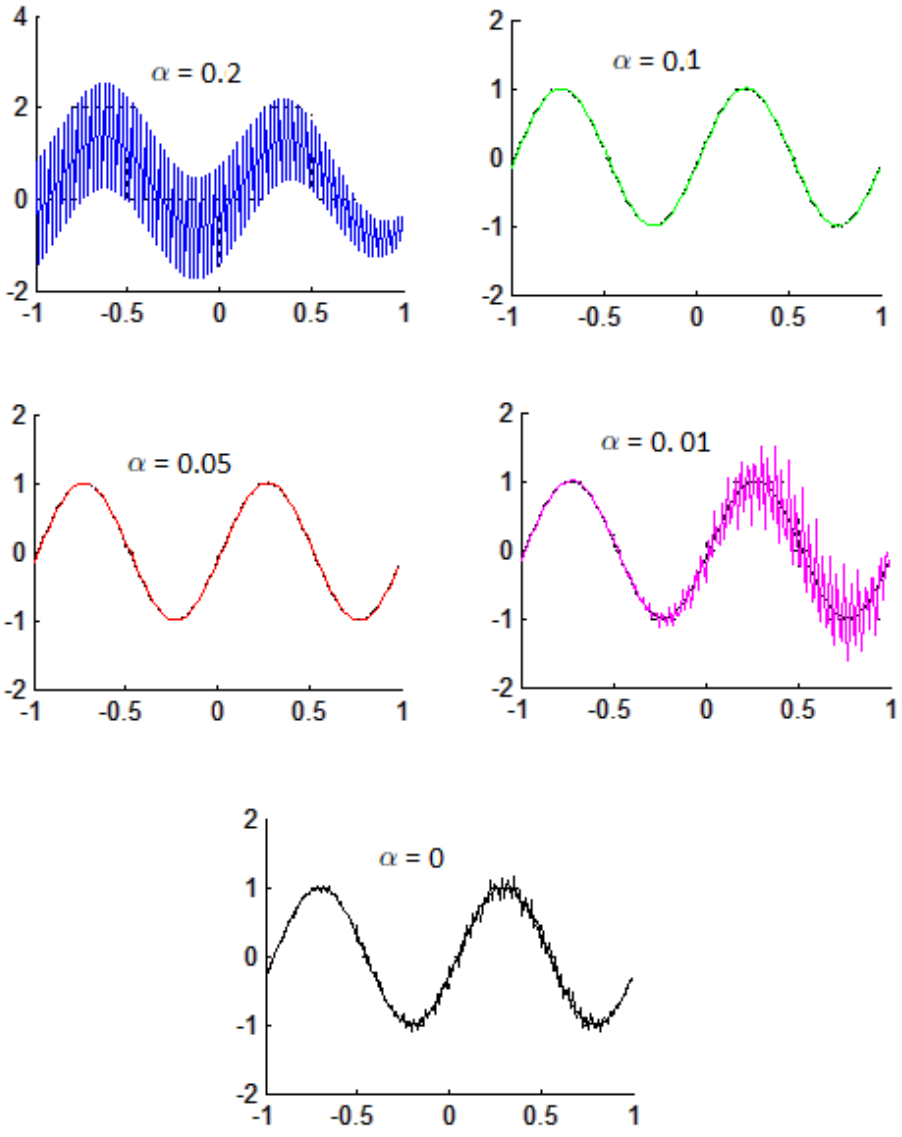


Figure 5.10: SUPG quadratic Lagrange $\mathbb{P}_2$ FEM in conjunction with $RK2$ for the $1D$ advection problem.

Figure 5.10 shows that the SUPG solution obtained for τ defined by $\alpha \geq 0.2$ contains large spurious oscillations whereas they are less important for $\alpha \leq 0.01$. These oscillations disappear if $\alpha \in [0.05, 0.1]$. The results shown in Table 5.3 confirm what has been observed in Fig. 5.10. These results are similar as those obtained with SUPG $\mathbb{P}_1$ method.

5.6.3 Accuracy study

To assess the quality of the numerical approximations, we compare the approximate solutions with the exact solution in terms of L^2-norm. The errors of the numerical approximations for various number of d.o.f. and choice of the coefficient α are shown in Table 5.3 for the $RK2$ time discretization and in Table 5.4 for the $RK4$ time discretization: surprisingly, no stabilization was required in this case.

$e(T)$							
α	$d.o.f. = 16$	$d.o.f. = 32$	$d.o.f. = 64$	$d.o.f. = 128$	$d.o.f. = 256$	$d.o.f. = 512$	$d.o.f. = 1024$
$\alpha = 0.9$	$2.555E+064$	$1.042E+131$	–	–	–	–	–
$\alpha = 0.8$	$9.798E+059$	$1.274E+122$	–	–	–	–	–
$\alpha = 0.7$	$8.426E+054$	$7.731E+111$	–	–	–	–	–
$\alpha = 0.6$	$9.837E+048$	$8.491E+099$	–	–	–	–	–
$\alpha = 0.5$	$7.048E+041$	$3.430E+085$	–	–	–	–	–
$\alpha = 0.4$	$8.064E+032$	$3.402E+067$	$4.392E+137$	–	–	–	–
$\alpha = 0.3$	$1.290E+021$	$5.983E+043$	$9.608E+089$	–	–	–	–
$\alpha = 0.2$	$1.001E+005$	$8.691E+010$	$4.601E+023$	$1.004E+050$	$3.818E+103$	–	–
$\alpha = 0.1$	0.032	$7.875E-003$	$1.921E-003$	$4.739E-004$	$1.176E-004$	$2.932E-005$	$7.318E-006$
$\alpha = 0.09$	0.032	$7.897E-003$	$1.924E-003$	$4.744E-004$	$1.177E-004$	$2.932E-005$	$7.320E-006$
$\alpha = 0.08$	0.032	$7.925E-003$	$1.929E-003$	$4.75E-004$	$1.178E-004$	$2.933E-005$	$7.373E-006$
$\alpha = 0.07$	0.033	$7.958E-003$	$1.935E-003$	$4.757E-004$	$1.179E-004$	$2.935E-005$	$1.528E-005$
$\alpha = 0.06$	0.033	$7.996E-003$	$1.942E-003$	$4.768E-004$	$1.180E-004$	$2.945E-005$	$6.492E-004$
$\alpha = 0.05$	0.033	$8.04E-003$	$1.952E-003$	$4.782E-004$	$1.182E-004$	$4.259E-005$	$1.574E-001$
$\alpha = 0.04$	0.032	$8.308E-003$	$1.965E-003$	$4.803E-004$	$1.190E-004$	$1.772E-003$	$6.227E+002$
$\alpha = 0.03$	0.033	$8.125E-003$	$1.982E-003$	$4.838E-004$	$2.473E-004$	0.845	$1.769E+008$
$\alpha = 0.02$	0.033	$8.141E-003$	$2.003E-003$	$5.043E-004$	$2.504E-002$	$1.494E+004$	$7.326E+016$
$\alpha = 0.01$	0.033	$8.111E-003$	$2.022E-003$	$5.050E-003$	65.454	$1.592E+011$	$1.301E+031$

Table 5.3: The L^2-error norm in function of the choice of the stabilization parameter α.

These results have been obtain without stabilization.

$e(T)$						
$d.o.f. = 16$	$d.o.f. = 32$	$d.o.f. = 64$	$d.o.f. = 128$	$d.o.f. = 256$	$d.o.f. = 512$	$d.o.f. = 1024$
$1.420E-02$	$3.406E-03$	$8.476E-04$	$2.1154E-04$	$5.285E-05$	$1.321E-05$	$3.302E-06$

Table 5.4: The L^2-error norm for $RK4$ time discretization.

In Fig. 5.11 we show the L^2-error norm depending on the numbers of d.o.f for the optimal choice of α.

5.7 Conclusion

In the present chapter we have focuced our attention to the SUPG FEM. As mentioned before, no standard scheme has been formulated to select the stabilisation parameter τ, which depends on the problem and the grid size. In our work, we have assumed that the stabilisation parameter τ takes the form:

$$\tau = \alpha \frac{h}{c},$$

and thus we have focused our attention on the selection of $\alpha \in [0,1]$.
In order to have a precise idea for the linear and quadratic Lagrange SUPG FEM in conjunction with $RK2$ for time discretization, we represented the L^2-error norm for an optimal choice for each method for a successive refinement of the mesh. Even if the convergence rate does not seem to be strongly impacted by the choice of α, the influence of the stabilization on the error level was reported.

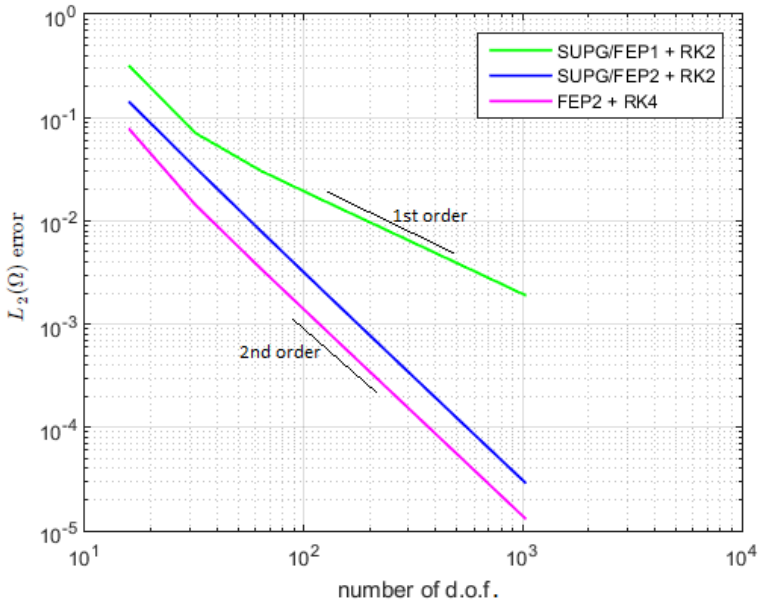


Figure 5.11: L^2-error for the advection problem with the linear and quadratic Lagrange FEM.

CONCLUSION

Dans ce chapitre, l'équation d'advection pure unidimensionnelle est étudiée. Le but est de mettre en évidence l'instabilité de la méthode de Galerkin pour un tel problème et d'analyser la façon dont les méthodes stabilisées apportent une solution pour construire un schéma à la fois robuste et précis. Pour cela, on a introduit une perturbation des fonctions tests pondérées par un paramètre de stabilisation afin d'assurer la précision et la consistance du schéma numérique. C'est dans ce contexte qu'on a introduit les méthodes de type Petrov-Galerkin: SUPG, PSPG ou encore GLS. Comme cela a déjà été dit, une approche éléments finis stabilisés s'appuie sur une formulation éléments finis standard à laquelle s'ajoute un terme de stabilisation. Cela revient à passer d'une méthode de Galerkin à une méthode de Petrov-Galerkin grâce à la modification des fonctions tests de la formulation faible.

Dans le présent chapitre, nous avons concentré notre attention sur la méthode d'EF stabilisé SUPG, pour Lagrange linéaire et quadratique en conjonction avec $RK2$ et $RK4$ pour la discrétisation temporelle. Une attention particulière a été accordée à l'identification du paramètre de stabilisation τ de cette méthode qui pondère les termes de stabilisation, et son influence sur l'erreur de la solution.

CHAPTER 6

ISOGEOMETRIC ANALYSIS: B-SPLINE AS A FEM BASIS

Isogeometric Analysis (IGA) is a generalization of classical FEA. It was first introduced by T.J.R. Hughes, J.A. Cottrell and Y. Bazilevs [43] with the main aim of closing the gap between the geometrical description and the analysis of engineering problems. The isogeometric paradigm consists of using basis functions commonly found in CAD geometries such as B-spline, to represent both the geometry and the physical fields in the solution of problems governed by partial differential equations (PDEs) [43] [67]. However, Hughes and co-authors considered only elliptic or parabolic problems so far. The objective of this chapter is to consider IGA in the hyperbolic context. This chapter reviews the various computational procedures for IGA, by revisiting the one-dimensional advection problem that is given in (5.6). While in this chapter we use B-splines (due to the simplicity of the domain), it is not hard to generalize it to other splines such as NURBS. Detailed comparisons between IGA and FEA approaches are carried out.

6.1 IGA: a B-spline based approach

IGA employs the same mathematical foundations as FEA to obtain the numerical solutions of differential equations, but the idea behind IGA method is to use the same functions that define the physical domain $\Omega \subset \mathbb{R}^2$, to approach the solution. Although the main difference between FEA and IGA lies compactly in the set of basis functions used, this change influences all steps of traditional FEA: preprocessing, solving and postprocessing.

The basis functions of IGA B-splines (more generally NURBS) are defined on a parameter space (or parameter domain) that we denote $\widetilde{\Omega}$. It is defined by control points, and not nodes like discretized domain in FEA. The input of node coordinates is replaced by coordinates of control points. Before we proceed further, let us notice that the B-spline parameter space $\widetilde{\Omega}$ corresponds to **"patches"** instead of elements. A patch can be seen as a **"macro-element"**, whereas knot intervals can be seen as traditional elements. Thus, the

B-spline mapping transforms a patch of multiple elements in the parameter space into the physical space. Each element in the physical space is the image of a corresponding element (knot interval) in the parameter space, but the mapping itself is global to the whole patch, rather than to the elements themselves. We refer to Fig. 6.1 for a schematic overview of this approach.

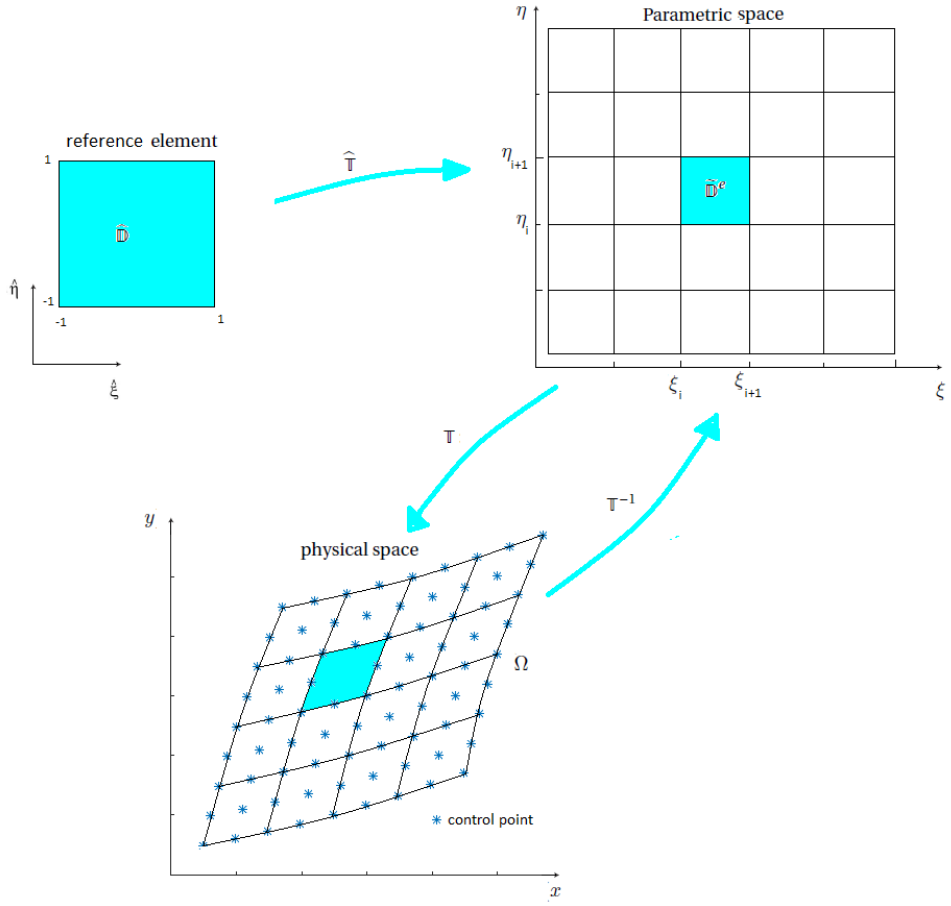


Figure 6.1: An example of a B-spline patch in physical space Ω, parametric space $\widetilde{\Omega}$, and the reference element $\widehat{\Omega}$ used to perform numerical integration.

6.1.1 Isogeometric discretisation

The parameter domain $\widetilde{\Omega}$ is defined by the knot vectors Ξ_1 and Ξ_2:

$$\Xi_1 = \left\{\xi_1 = ... = \xi_{p+1}, \xi_{p+2}, ..., \xi_n, \xi_{n+1} = ... = \xi_{n+p+1}\right\},$$

and

$$\Xi_2 = \left\{\eta_1 = ... = \eta_{q+1}, \eta_{q+2}, ..., \eta_m, \eta_{m+1} = ... = \eta_{m+q+1}\right\},$$

where, p and q are prescribed degrees, ξ_i and η_j are the i-th and j-th knots, i, j are the knots indices, $i = 1, 2, ..., n+p+1$, $j = 1, 2, ..., m+q+1$ and n, m equals the number of basis functions.

IGA allows to control the geometry and variables at the control points, in contrast to FEA which writes these quantities at the nodes. The transformation of the parametric domain $\widetilde{\Omega}$ to the physical domain Ω is introduced:

$$\mathbb{T} : \widetilde{\Omega} \longmapsto \Omega$$

$$(\xi, \eta) \longmapsto \big(x(\xi, \eta), y(\xi, \eta)\big).$$

Note that this transformation is not linear and is simply defined by the parametric representation of the computational domain. Unless otherwise specified, we assume that the mapping $\mathbb{T}$ to be invertible and its inverse:

$$\mathbb{T}^{-1} : \Omega \longmapsto \widetilde{\Omega},$$

transforms points in the physical domain to their corresponding parameter values. Any point of coordinate (x, y) in the physical domain Ω is mapped to a point (ξ, η) in the parametric domain $\widetilde{\Omega}$. The mapping from the parametric domain to the physical domain is defined by associating a control net P_{ij} to each basis function in such a way that:

$$\big(x(\xi, \eta), y(\xi, \eta)\big) = \sum_{i=1}^{n} \sum_{j=1}^{m} \mathcal{N}_{i,p}(\xi) \mathcal{N}_{j,q}(\eta) P_{ij}.$$

Thus, the basis functions in the physical domain Ω are defined by:

$$\mathbb{N}_{ij,pq}(x, y) = \mathbb{N}_{ij,pq}\big(\mathbb{T}(\xi, \eta)\big) = \mathcal{N}_{ij,pq}(\xi, \eta) = \mathcal{N}_{i,p}(\xi) \mathcal{N}_{j,q}(\eta).$$

It is important to point out that in some applications (which uses the simple geometries such as square, circle/ellipse), the computational domain can be modeled straightforwardly with only a single patch. But for more complex geometries, it is not possible to describe the physical computational domain with just one geometrical mapping $\mathbb{T}$, we need to use multi-patch geometries (we refer the reader to [26] [40]). In this thesis we focus on single-patch geometries for the sake of simplicity but the extension is quite simple.
In an isoparametric formulation, the variable field is approximated by the same shape functions:

$$u(\xi, \eta) = \sum_{i=1}^{n} \sum_{j=1}^{m} \mathcal{N}_{i,p}(\xi) \mathcal{N}_{j,q}(\eta) u_{ij}.$$

Hence, the variable u is obtained as functions of ξ and η. Here u_{ij} denotes the value of the variable field u corresponding to the control net P_{ij}. It is therefore referred to as a control variable or more generally a degree of freedom (d.o.f.). Note that B-spline representations are not interpolatory, so u_{ij} is not the solution value at the location P_{ij}.

6.1.2 Computational procedures for IGA

The fact that the functions are defined in the parametric space while the space derivatives are with respect to the coordinates of the physical space makes it necessary to compute the Jacobian matrix of the geometric mapping defined as:

$$J = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial x}{\partial \eta} \\ \frac{\partial y}{\partial \xi} & \frac{\partial y}{\partial \eta} \end{pmatrix}.$$

We also highlight that for the one-dimensional case, the Jacobian matrix reduces to:

$$J_\xi = \frac{\partial x}{\partial \xi} = \sum_{i=1}^{n} \Big(\frac{\partial \mathcal{N}_{i,p}(\xi)}{\partial \xi}\Big) P_i,$$

and we denote, $J_\xi^{-1} = \frac{\partial \xi}{\partial x}$.

We will try to give an idea of the calculation of the integrals. Integrals over the entire geometry (physical domain) are split into elementary integrals over a domain denoted by Ω_e. Let $\mathbf{f}$ be a function of the two variables x and y. Then,

$$\begin{aligned}
\int_\Omega \mathbf{f}(x,y)\,d\Omega &= \sum_{e=1}^{\mathfrak{N}_{el}} \int_{\Omega_e} \mathbf{f}(x,y)\,d\Omega_e \\
&= \sum_{e=1}^{\mathfrak{N}_{el}} \int_{\Omega_e} \mathbf{f}(\mathbb{T}(\xi,\eta))\,d\Omega_e \\
&= \sum_{e=1}^{\mathfrak{N}_{el}} \int_{\widetilde{\Omega}_e} \mathbf{f}(\xi,\eta) \mid J \mid d\widetilde{\Omega}_e \\
&= \sum_{e=1}^{\mathfrak{N}_{el}} \int_{\widehat{\Omega}} \mathbf{f}\big(\widehat{\xi},\widehat{\eta}\big) \mid J \mid\mid \widehat{J} \mid d\widehat{\Omega} \quad \text{(via the mapping } \widehat{\mathbb{T}} (Fig.\ 6.1)).
\end{aligned}$$

We also highlight that $\widehat{\xi}$ and $\widehat{\eta}$ on the parent domain $\widehat{\Omega}$ are given as functions of ξ and η on parametric domain $\widetilde{\Omega}_{ij} = [\xi_i, \xi_{i+1}] \times [\eta_j, \eta_{j+1}]$:

$$\xi = \frac{1}{2}\big((\xi_{i+1} - \xi_i)\widehat{\xi} + (\xi_{i+1} + \xi_i)\big),$$

$$\eta = \frac{1}{2}\big((\eta_{j+1} - \eta_j)\widehat{\eta} + (\eta_{j+1} + \eta_j)\big).$$

Therefore, the Jacobian of this transformation reads:

$$\mid \widehat{J} \mid = \frac{1}{4}\big(\xi_{i+1} - \xi_i\big)\big(\eta_{j+1} - \eta_j\big).$$

For the one-dimensional case $\mid \widehat{J} \mid$ reduces to:

$$\mid \widehat{J} \mid = \frac{1}{2}\big(\xi_{i+1} - \xi_i\big).$$

6.2 Isogeometric FE formulation

After a general description of the physical domain, geometrical mappings, and an introduction of notations for the IG method, we will discusses in this section specific details of the implementation of IGFE method.

The IGM based on B-splines is not much different than the classical FEM. Subtle differences are introduced due to the non-interpolatory character of B-splines and the definition of an element.
We focus now on the one-dimensional advection problem that is given in the previous chapter by:

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} &= 0 & \forall (x,t) \in [\mathrm{a},\mathrm{b}] \times [0,T], \\ u(x,0) &= u_0(x) & \forall x \in [\mathrm{a},\mathrm{b}], \\ u(\mathrm{a},t) &= u_{\mathrm{a}}(t) & \forall t \in [0,T], \end{cases} \tag{6.1}$$

with,

$$\begin{cases} u_0(x) &= \sin(2\pi x) & \forall x \in [\mathrm{a},\mathrm{b}], \\ u_{\mathrm{a}}(t) &= \sin\big(2\pi(-1-ct)\big) & \forall t \in [0,T]. \end{cases}$$

In order to solve Eq. (6.1), we consider a spatial discretization of its domain by means of B-spline functions IGA in the framework of the Galerkin method and we need to write the weak form. Before we proceed further, let us define:

$$\mathbf{V}_{\mathcal{N}} := span\Big\{\mathcal{N}_i, \quad \text{for} \quad i = 1,..,n\Big\},$$
$$\mathbf{V}_{\mathbb{N}} := span\Big\{\mathbb{N}_i, \quad \text{for} \quad i = 1,..,n\Big\}.$$

According to the IGA concept, these spaces will be used to build the test function spaces for the approximation of (6.1). However, we define a subspace V^B which only span a finite number of B-spline basis functions. That is:

$$V^B = \mathbf{V} \cap \mathbf{V}_{\mathbb{N}},$$

where,

$$\mathbf{V} := \Big\{ w^B \in H^1(\Omega) \quad \text{such that} \quad \mathrm{w}^B(\mathrm{a}) = u_{\mathrm{a}} \Big\}.$$

For IGA in the framework of the Galerkin method, we start by multiplying the strong form (6.1) by a B-spline test function v^B and integrate over the domain Ω, then using Green's first identity and applying the boundary conditions we finally get:

$$\int_\Omega \partial_t u(x,t) v^B(x) dx - c\int_\Omega u(x,t)\partial_x v^B(x) dx = c u_{\mathrm{a}} v^B(\mathrm{a}) - c u_{\mathrm{b}} v^B(\mathrm{b}).$$

As in the previous chapter, we introduce a SUPG stabilization term yielding:

$$\begin{cases} \text{Find} \quad u \in H^1([0,T], V^B) \quad \text{such that}: \\ \int_\Omega \dfrac{\partial u(x,t)}{\partial t} v^B(x) dx - c\int_\Omega u(x,t)\dfrac{\partial v^B(x)}{\partial x} dx + \tau c \sum_{k=1}^{n-p} \int_{x_k}^{x_{k+1}} \dfrac{\partial u(x,t)}{\partial t}\dfrac{\partial v^B(x)}{\partial x} dx \\ +\tau c^2 \sum_{k=1}^{n-p} \int_{x_k}^{x_{k+1}} \dfrac{\partial u(x,t)}{\partial x}\dfrac{\partial v^B(x)}{\partial x} dx = c u_{\mathrm{a}} v^B(\mathrm{a}) - c u_{\mathrm{b}} v^B(\mathrm{b}) \qquad \forall v^B \in H^1([\mathrm{a},\mathrm{b}]). \end{cases}$$

The solution u is obtained by solving the finite-dimensional problem:

$$\begin{cases} \text{Find} \quad u \in H^1([0,T], V^B) & \text{such that :} \\ a_{sp}(u, v^B) = L_{sp}(v^B) & \forall\, v^B \in H^1([\mathrm{a},\mathrm{b}]). \end{cases} \tag{6.2}$$

Where, $a_{sp} : V^B \times H^1([\mathrm{a},\mathrm{b}]) \longmapsto \mathbb{R}$ is given by:

$$\begin{aligned} a_{sp}(u, v^B) &= \int_\Omega \frac{\partial u(x,t)}{\partial t} v^B(x)\,dx - c\int_\Omega u(x,t)\frac{\partial v^B(x)}{\partial x}\,dx + \tau c \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial t}\frac{\partial v^B(x)}{\partial x}\,dx \\ &+ \tau c^2 \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \frac{\partial u(x,t)}{\partial x}\frac{\partial v^B(x)}{\partial x}\,dx, \end{aligned}$$

and

$L_{sp} : H^1([\mathrm{a},\mathrm{b}]) \longmapsto \mathbb{R}$, that contains the right hand side term of (6.2) and

$$L_{sp}(v^B) = c u_{\mathrm{a}} v^B(\mathrm{a}) - c u_{\mathrm{b}} v^B(\mathrm{b}).$$

Galerkin's method consists in constructing finite-dimensional approximations of V^B, denoted V_h^B. Strictly speaking, these will be subsets such that: $V_h^B \subset V^B$. Therefore, the SUPG formulation of this problem can be written:

$$\begin{cases} \text{Find} \quad u_h \in H^1([0,T], V_h^B) \quad \text{such that :} \\ \int_\Omega \frac{\partial u_h(x,t)}{\partial t} v_h^B(x)\,dx - c\int_\Omega u_h(x,t)\frac{\partial v_h^B(x)}{\partial x}\,dx + \tau c \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \frac{\partial u_h(x,t)}{\partial t}\frac{\partial v_h^B(x)}{\partial x}\,dx \\ +\tau c^2 \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \frac{\partial u_h(x,t)}{\partial x}\frac{\partial v_h^B(x)}{\partial x}\,dx - c u_{\mathrm{a}}(t) v_h^B(\mathrm{a}) + c u_{\mathrm{b}}(t) v_h^B(\mathrm{b}) = 0. \end{cases}$$

According to the isogeometric paradigm, the solution u_h from the IGA space $H^1([0,T], V_h^B)$ can be represented using B-spline basis functions for the one-dimensional case, in the form:

$$u_h(x,t) = \sum_{j=1}^{n} \mathbb{N}_{j,p}(x)\, u_j(t),$$

where n is the number of basis functions which is equal to the number of control points and the degrees of freedom (d.o.f.) of u_h associated with the control points, respectively. This representation is similar to that used in classical FEM. By choosing the test function v_h^B equal to the B-spline basis function $\mathbb{N}_{i,p}$, the Galerkin IGA scheme reads as follows:

$$\begin{cases} \text{Find} \quad u_1, u_2, ..., u_n \quad \text{such that :} \\ \sum_{j=1}^{n} \Big(\int_\Omega \mathbb{N}_{j,p}(x)\mathbb{N}_{i,p}(x)\frac{\partial u_j(t)}{\partial t}\,dx + \tau c \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \mathbb{N}_{j,p}(x)\mathbb{N}'_{i,p}(x) u_j(t)\,dx - c\int_\Omega \mathbb{N}_{j,p}(x)\mathbb{N}'_{i,p}(x) u_j(t) \\ +\tau c^2 \sum_{k=1}^{n-p}\int_{x_k}^{x_{k+1}} \mathbb{N}'_{i,p}(x)\mathbb{N}'_{j,p}(x) u_j(t)\,dx \Big) = c u_{\mathrm{a}} \mathbb{N}_{i,p}(\mathrm{a}) - c u_{\mathrm{b}} \mathbb{N}_{i,p}(\mathrm{b}) \qquad \forall\, i = 1, ..., n. \end{cases}$$

Finally, we obtain the linear system which is:

$$M^B \partial_t U = R^B U + c U_{\mathrm{a}}^B - c U_{\mathrm{b}}^B, \tag{6.3}$$

similar to that resulting from the FEM in conjunction with SUPG.

The elementary mass and stiffness matrices are defined by:

$$M_{ij}^B = \int_\Omega \mathbb{N}_{j,p}(x)\mathbb{N}_{i,p}(x)dx + \tau c \sum_{k=1}^{n-p} \int_{x_k}^{x_{k+1}} \mathbb{N}_{j,p}(x)\mathbb{N}'_{i,p}(x)dx \qquad \forall i,j = 1,...,n,$$

and

$$R_{ij}^B = c\int_\Omega \mathbb{N}_{j,p}(x)\mathbb{N}'_{i,p}(x)dx - \tau c^2 \sum_{k=1}^{n-p} \int_{x_k}^{x_{k+1}} \mathbb{N}'_{j,p}(x)\mathbb{N}'_{i,p}(x)dx \qquad \forall i,j = 1,...,n,$$

and the right-hand side given by:

$$U_{\mathrm{a}}^B = \begin{pmatrix} \sin(2\pi(-1-ct)) \\ 0 \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^n, \qquad \text{and} \qquad U_{\mathrm{b}}^B = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ u_n \end{pmatrix} \in \mathbb{R}^n.$$

We note that on the element Ω_j, only $p+1$ B-splines functions $\mathbb{N}_{j,p}, \mathbb{N}_{j+1,p}, ..., \mathbb{N}_{j+p,p}$ non zero and only $p+1$ derivatives $\mathbb{N}'_{j,p}, \mathbb{N}'_{j+1,p}, ..., \mathbb{N}'_{j+p,p}$ are non zero. The correspond matrices are therefore sparse.
By performing the integration in the parametric space and using the Jacobian of the B-spline mapping, we get:

$$M_{ij}^B = \int_{\widetilde{\Omega}} \mathcal{N}_{j,p}(\xi)\mathcal{N}_{i,p}(\xi)J_\xi d\xi + \tau c \sum_{k=1}^{n-p} \int_{\widetilde{\Omega}_k} \mathcal{N}_{j,p}(\xi)\mathcal{N}'_{i,p}(\xi)d\xi,$$

and

$$R_{ij}^B = c\int_{\widetilde{\Omega}} \mathcal{N}_{j,p}(\xi)\mathcal{N}'_{i,p}(\xi)d\xi - \tau c^2 \sum_{k=1}^{n-p} \int_{\widetilde{\Omega}_k} \mathcal{N}'_{j,p}(\xi)\mathcal{N}'_{i,p}(\xi)J_\xi^{-1} d\xi.$$

By interpreting the integration in the parent element, we get:

$$M_{ij}^B = \int_{\widehat{\Omega}} \mathcal{N}_{j,p}(\hat{\xi})\mathcal{N}_{i,p}(\hat{\xi})\hat{J}_{\hat{\xi}} d\hat{\xi} + \tau c \sum_{k=1}^{n-p} \int_{\widehat{\Omega}_k} \mathcal{N}_{j,p}(\hat{\xi})\mathcal{N}'_{i,p}(\hat{\xi})d\hat{\xi},$$

and

$$R_{ij}^B = c\int_{\widehat{\Omega}} \mathcal{N}_{j,p}(\hat{\xi})\mathcal{N}'_{i,p}(\hat{\xi})d\hat{\xi} - \tau c^2 \sum_{k=1}^{n-p} \int_{\widehat{\Omega}_k} \mathcal{N}'_{j,p}(\hat{\xi})\mathcal{N}'_{i,p}(\hat{\xi})\hat{J}_{\hat{\xi}}^{-1} d\hat{\xi}.$$

Contrary to Lagrange elements, the element integrals appearing are evaluated using Gauss points via Gaussian quadrature which can be described as follows:

$$\int_{\widehat{\Omega}} \mathcal{N}_{j,p}(\hat{\xi})\mathcal{N}_{i,p}(\hat{\xi})d\hat{\xi} \simeq \sum_{k=1}^{n_G} \mathcal{N}_{j,p}(X_k^G)\mathcal{N}_{i,p}(X_k^G)\omega_k^G,$$

where n_G are the number of integration points X_k^G, ω_k^G is the weight corresponding to the k-th integration point (see Appendix A). This approach is more flexible than the use of analytical integration.
The global mass and stiffness matrices are obtained using the assembly technique.

As mentionned before, we focus in this thesis on the RK method for the time integration. The ODE (6.3) is integrated in time by means of a second and fourth-order RK schemes. In one spatial dimension, a space discretization using polynomials basis functions of degree p, associated to a $(p+1)$-stage RK time integrator of order $p+1$, the stability limit for the CFL number is defined by:

$$\frac{\Delta t \mid c \mid}{\mathfrak{h}} \leq C_{CFL}.$$

where $\mathfrak{h}$ is the size of the image of the knot interval considered.

6.3 Numerical results

The standard Galerkin formulation for the advection problem produces unstable discretizations. This is well known in FEA for the Lagrange $\mathbb{P}_1$ and $\mathbb{P}_2$ elements, and unfortunately, is also observed for the linear and quadratic B-splines-based approach as we can see in this section. We investigate the ability of the isogeometric approach based on B-splines of arbitrary degree, in conjunction with SUPG, to solve advection problem. In all calculations the mesh is uniform with element size length $\mathfrak{h} = \frac{2}{n-p}$. The solutions are calculated from $p = 1$ to $p = 4$. Notice that in the case of linear and quadratic B-spline, the standard SUPG formulation is used with the choice of the stabilization parameter $\tau = \frac{\alpha \mathfrak{h}}{c}$ while for the cubic and quartic B-spline it is important to point out that the method has been found stable, and therefore $\tau = 0$.

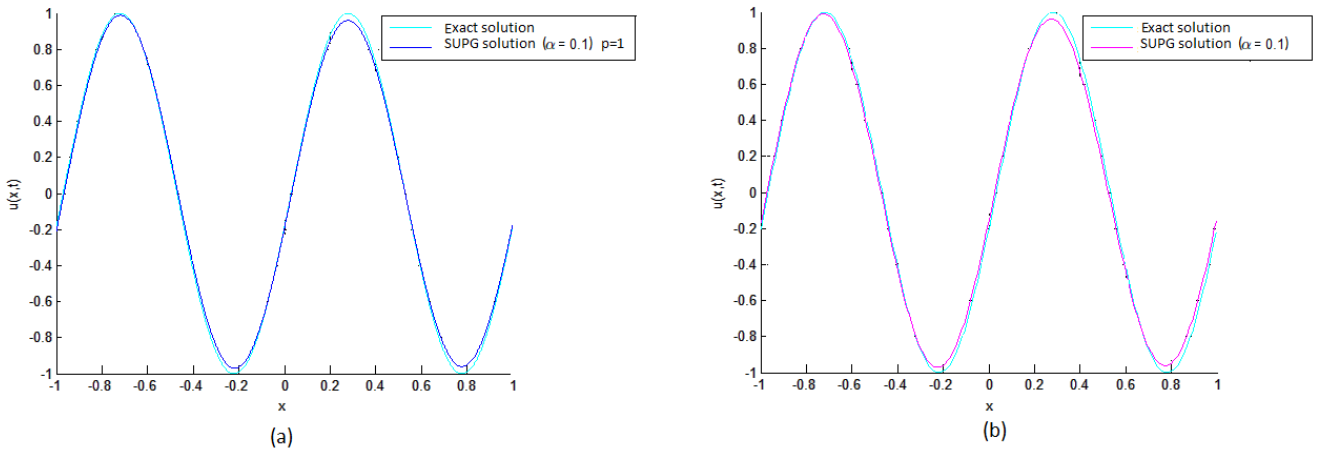


Figure 6.2: (a) SUPG B-spline linear solution for the advection problem ($\alpha = 0.1$). (b) SUPG FEM $\mathbb{P}_1$ for the advection problem ($\alpha = 0.1$) at $T = 0.4s$.

Due to the equivalence of hat shape functions and linear B-spline functions, the linear FEM and IGA give the same results (as shown in Fig. 7.3). However, this is not the case for representations of higher degree. If one compares quadratic B-spline basis functions with basis functions based on Lagrange interpolating polynomials, one can notice some critical differences.

6.3.1 Influence of the SUPG stabilization parameter τ for the quadratic B-spline

We first consider the case of quadratic B-spline. The results for the advection problem for different choices of $\alpha \in [0,1]$ are presented in the Fig. 6.3. The plotting routine sampled the solution with a grid of uniformly distributed points.

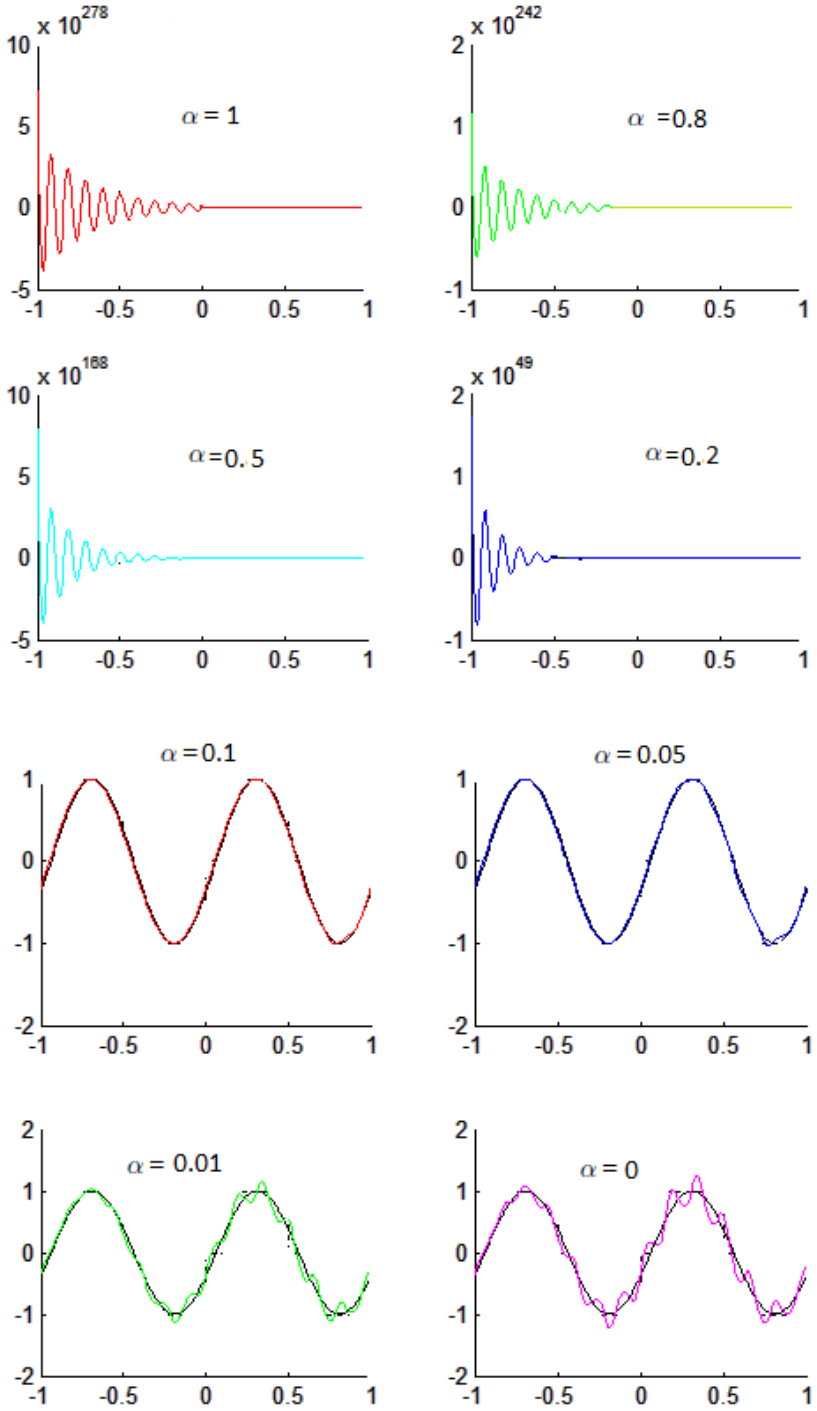


Figure 6.3: SUPG quadratic B-spline solutions in conjunction with *RK*2 for the advection problem.

These results are very similar to those obtained with Lagrange quadratic elements in Fig. 5.10.

6.3.2 Error estimates for the quadratic B-spline

We define the error of the numerical solution as $e(T) = u_{ex}(x,T) - u_h(x,T)$. The $L^2(\Omega)$-norm of the error is depicted in the Table 6.1 which is defined by:

$$\| e(T) \|_{L^2([-1,1])} = \| u_{ex}(x,T) - u_h(x,T) \|_{L^2([-1,1])} .$$

$\| e \|_{L_2}$							
α	$n=16$	$n=32$	$n=64$	$n=128$	$n=256$	$n=512$	$n=1024$
0.9	$6.7674E+057$	$3.972E+123$	–	–	–	–	–
0.8	$6.773E+052$	$1.683E+113$	–	–	–	–	–
0.7	$1.783E+047$	$4.567E+101$	–	–	–	–	–
0.6	$8.831E+040$	$3.967E+088$	–	–	–	–	–
0.5	$5.093E+033$	$4.032E+073$	–	–	–	–	–
0.4	$1.721E+025$	$1.018E+056$	$4.240E+110$	–	–	–	–
0.3	$1.321E+015$	$5.248E+034$	$3.301E+073$	–	–	–	–
0.2	$1.400E+003$	$0.001E+008$	$7.434E+008$	$8.070E+019$	$1.180E+093$	–	–
0.1	0.036	$5.293E-003$	$2.552E-003$	$4.285E-005$	$1.671E-005$	$7.584E-006$	$2.860E-06$
0.09	0.032	$4.417E-003$	$2.185E-004$	$4.050E-005$	$1.669E-005$	$7.583E-006$	$2.860E-06$
0.08	0.029	$3.816E-03$	$1.938E-004$	$3.932E-005$	$1.668E-005$	$7.583E-006$	$2.860E-00$
0.07	0.027	$3.385E-003$	$1.765E-004$	$3.871E-005$	$1.668E-005$	$7.583E-006$	$2.860E-06$
0.06	0.026	$3.067E-003$	$1.639E-004$	$3.839E-005$	$1.667E-005$	$7.583E-006$	$2.860E-06$
0.05	0.026	$2.829E-003$	$1.547E-004$	$3.824E-005$	$1.667E-005$	$7.583E-006$	$2.860E-06$
0.04	0.025	$2.656E-003$	$1.478E-004$	$3.817E-005$	$1.667E-005$	$7.583E-006$	$2.860E-06$
0.03	0.026	$2.541E-003$	$1.428E-004$	$3.815E-005$	$1.667E-005$	$7.583E-006$	$2.859E-06$
0.02	0.027	$2.492E-003$	$1.401E-004$	$3.816E-005$	$1.667E-005$	$7.594E-006$	$1.322E-04$
0.01	0.029	$2.535E-003$	$1.418E-004$	$3.818E-005$	$1.670E-005$	$1.5091E-005$	0.357

Table 6.1: The L^2-error as function of the choice of the number of control points n and the stabilization parameter α, for quadratic B-splines in conjunction with $RK2$.

These results have been obtain without stabilization.

$\| e \|_{L_2}$						
$n=16$	$n=32$	$n=64$	$n=128$	$n=256$	$n=512$	$n=1024$
$2.433E-02$	$1.246E-03$	$6.970E-05$	$6.195E-06$	$1.043E-06$	$1.235E-07$	$1.324E-08$

Table 6.2: The L^2-error as function of the choice of the number of control points n for quadratic B-splines in conjunction with $RK4$.

Convergence results in the L^2-norm are shown in Tab. 6.1 and Tab. 6.2 for the quadratic B-spline. These results are also visualized in Fig. 6.4. As a result, the continuity of the basis is C^1 everywhere, for the quadratic B-spline case. As can be seen, the L^2-convergence for quadratic B-spline in conjunction with the $RK4$ for the temporal discretization is approximately 3. This is the best one could reasonably hope for.

6.4 Higher order B-spline

Table 6.3 shows the L^2-error of the numerical approximations for various number of control point n and order of B-spline. Those results are depicted in Figure 6.4. These results have been obtain without stabilization.

Table 6.4 shows the convergence rates observed for different degrees. As can be seen, a sub-optimal rate of the value 2 is obtained for quadratic basis in conjunction with $RK2$, certainly due to the stabilization. For quadratic basis in conjunction with $RK4$, the optimal rate of the value 3 is obtained. For the degree $p = 4$, the rate is limited by the use of $RK4$ time integrator. For $p = 3$ a sub-optimal rate of value 3.5 is also observed.

	$n = 16$	$n = 32$	$n = 64$	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$p = 3 + RK4$	$2.620E-03$	$4.502E-05$	$8.860E-07$	$3.024E-08$	$1.889E-09$	$1.900E-10$	$1.917E-11$
$p = 4 + RK4$	$2.528E-04$	$3.380E-06$	$1.223E-07$	$5.631E-09$	$3.211E-10$	$2.059E-11$	$1.483E-12$

Table 6.3: Error measured in the L^2-norm.

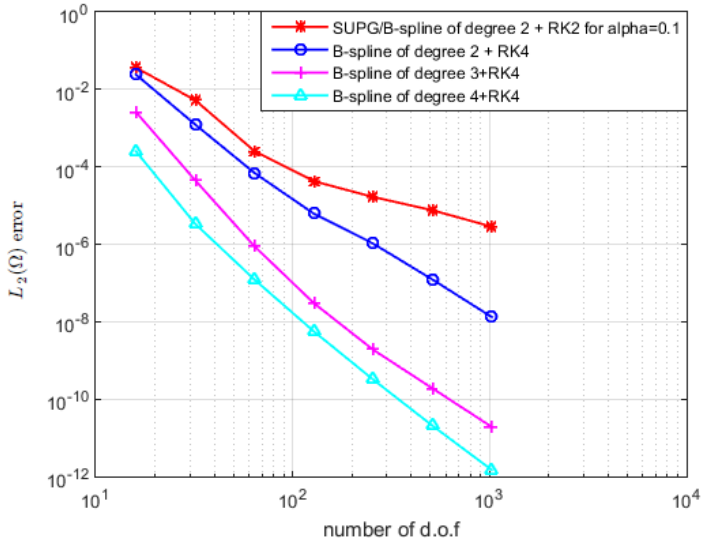


Figure 6.4: Convergence rates in the L^2-norm.

p	$n = 16$	$n = 32$	$n = 64$	$n = 128$	$n = 256$	$n = 512$	$n = 1024$
$2(for RK2)$	–	2.77	2.11	2.1	2.01	2.01	2
$2(for RK4)$	–	4.28	4.16	3.49	2.57	3.07	3.17
3	–	5.88	5.68	4.88	4.5	3.5	3.5
4	–	6.22	4.8	4.5	4.14	3.99	4

Table 6.4: Convergence rates.

6.5 IGFEA and classical FEA: a comparisons

In this section we present a comparison of the performance of IGFEM and the classical FEM for the numerical simulation of the one-dimensional advection problem.

These two methods employ the same mathematical foundations. Therefore, they have many similarities. However, some important differences lie in the choice of basis functions. Obviously, in classical FEM the basis which is chosen to approximate the unknown field is interpolatory. This often takes the form of polynomial functions and the geometry is in most cases only approximated.

Whereas the IGA approach has an advantage that the basis is chosen to exactly capture the geometry and this is also used to approximate the field of unknown quantities.

Another benefit is that the approximation is smooth. In fact, when the multiplicity at a knot is m, for $1 \leq m \leq p$ in IGA, the basis functions are then C^{p-m} at the interfaces of the involved elements, whereas basis functions based on Lagrange interpolating polynomials are only C^0.

Also, it is important to point out that, due to the support of a B-spline, function of order p is always $p+1$ knot spans. Therefore, a higher-order B-spline function has support over much larger portion of the domain when compared to classical FEM. Therefore, the computational efficiency is reduced.

We proceed now further to numerically compare the IGFEM and FEM. A good comparison we might perform is to compare the numerical approximations error with respect to the number of d.o.f. However, a special attention should be paid to the accuracy of the SUPG stabilization method.

As mentioned before, the linear FEM and IGFEM give the same results (due to the equivalence of hat shape functions and linear B-spline functions).

Note that the error for the one-dimensional advection problem is plotted in Fig. 6.5 as a function of the global d.o.f. in conjunction with SUPG stabilization (if necessary), to be able to compare IGFEM and FEM.

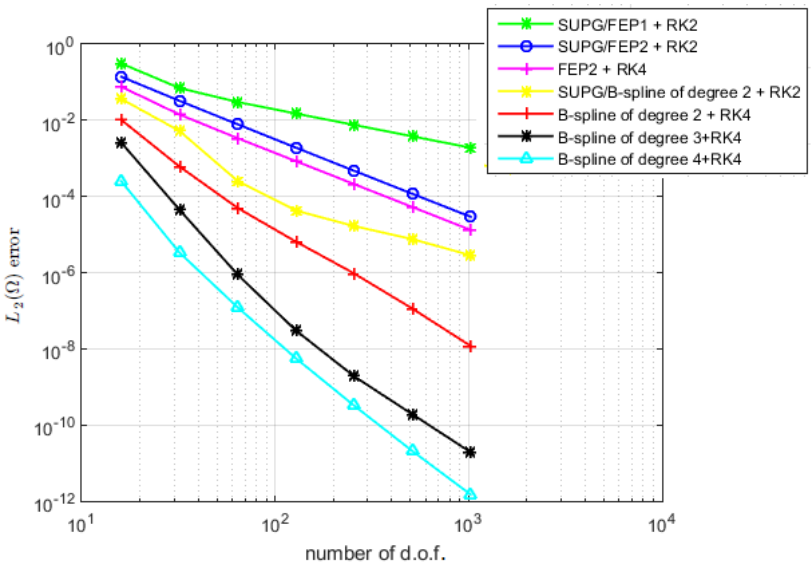


Figure 6.5: Error in the L^2-norm of IGFEM and classical FEM vs. number of d.o.f.

As one can see, results of this simple numerical test allow to make some conclusions. At first sight, the graphs of Fig. 6.5 indicate that the slope of the error lines are not exactly 1, 2, 3 and 4 for $1st$, $2nd$, $3rd$ and $4th$ order elements, respectively. Moreover, we highlight that the values of the errors from the IGFEM are far lower than those of the FEM.

We also note that, increasing the order of Lagrangian polynomials may increase the amplitude of oscillations in the FEM. This problem is eliminated in IGA as a result of non-negativity and non-interpolatory nature of the B-splines shape functions. In conclusion, we believe that the isogeometric approach has considerable potential in practical problem solving and is a promising alternative to current analysis procedures. However, a possible drawback of the SUPG method is the sensitivity of the solution to the stabilization parameter, whose optimal value is not determined precisely by the available theory.

In the context of IGA, another approach we could go for is the application of DG method. Indeed, the major argument for using DG methods lies in their ability to provide stable numerical methods for first order PDE problems, for which classical FEM is well known to perform poorly. Therefore, we will introduce in the next chapter a new DG method in the IG context, called IGDGM.

CONCLUSION

L'analyse isogéométrique (AIG) trouve ses origines depuis 2005 dans les travaux de Hughes, Cottrell, et Bazilevs [43] avant d'être détaillée et formalisée dans le livre de Cottrell et al. [24] dans un contexte de développement de nouveaux outils pour le calcul numérique en ingénierie basée sur la simulation, celle-ci introduit un nouveau paradigme pour tenter de combler le fossé entre la MEF et la CAO.

L'idée principale de l'AIG est de modéliser exactement la géométrie avec des fonctions qui vont servir à approximer la solution, ces fonctions permettant une description paramétrique d'un domaine. Elles sont définies dans un espace appelé espace paramétrique; la géométrie est définie dans un espace nommé espace physique. La paramétrisation de la géométrie est obtenue par un morphisme non linéaire entre l'espace paramétrique et l'espace physique défini à l'aide des fonctions B-splines. S'il est nécessaire d'effectuer des intégrations lors du processus de résolution, elles sont effectuées élément par élément sur un élément de référence obtenu par une transformation linéaire à partir de l'espace paramétrique.

On a introduit brièvement dans ce chapitre le contexte de l'AIG. Pour cela, on a commencé par rappeler les fondements et les premiers objectifs de la méthode. On se restreint aux B-splines mais on ne doute pas de l'applicabilité des méthodes développées ici à d'autres représentation de l'AIG (comme NURBS), ces fonctions possèdent en fait une continuité de classe supérieure, ce qui permet d'obtenir plus de précision pour un même nombre de degrés de libertés comparé aux EF classiques.

Les travaux présentés dans ce chapitre ont pour objectif de mettre au point une méthode d'EF stabilisée, la méthode SUPG pour un problème d'advection dans le cadre de l'AIG. L'idée principale est donc d'utiliser les fonctions de base B-spline représentant la géométrie pour générer l'espace de recherche de la solution au problème hyperbolique souhaité.

.

Part III

ISOGEOMETRIC DISCONTINUOUS GALERKIN METHOD (IGDGM)

CHAPTER 7

DISCONTINUOUS GALERKIN METHOD (DGM): FROM CLASSICAL TO ISOGEOMETRIC

We propose a method that combines isogeometric analysis (IGA) with the discontinuous Galerkin (DG) method for solving hyperbolic problems, namely the isogeometric discontinuous Galerkin (IGDG) method that merges exact geometry with high-order solution accuracy [54] [61]. In this chapter we formulate and analyze this method for the one-dimensional advection problem. The solution of the problem is approximated in every sub-domain without any continuity requirement for the discrete solution at the interfaces. Finally, we numerically compare the performance of the IGDG method with the DGFE method.

7.1 Introduction and background

The discontinuous Galerkin finite element (DGFE) method was originally introduced in 1973 by Reed and Hill [74], for the numerical solution of the nuclear transport PDE problem. Subsequently, the method has found broad applications in large-scale data intensive science and engineering problems. DG is a class of FEM that uses completely discontinuous basis functions. Thanks to their flexibility in local approximation, they offer good stability properties when approximating convection dominated problems [79] [93].
In contrast to the stabilized continuous Galerkin FEM, DG method produces stable discretizations for hyperbolic problems without the need for stabilization parameters, stabilization resulting from the use of upwind fluxes. Therefore, this method combines the best properties of the finite volume (FV) method and continuous Galerkin FEM.

In fact, the FV method, which is well suited to hyperbolic conservation laws, can only use low-degree polynomials to locally represent the solution. In contrast, FEM are unstable for hyperbolic problems and, as seen in previous chapters, stabilization relies on tedious choice of a parameter. Therefore, the idea of this method is to decompose the original problem into a set of subproblems, solved by using a FEM approach, that are connected using an appropriate transmission condition (known as the numerical flux).
Although DG methods have gained increasing attention in large-scale modeling applications, a shortcoming of the conventional DG methodology is the inability to fully recover complex underlying geometries in the meshing domain. To overcome this problem, we combine IGA method with the DG method. As mentioned before, IGA is a computational technique that improves and generalizes the classical FE method. The main benefit of this method is the exact representation of the geometry in the language of computer aided design (CAD) tools. This simplifies the meshing as the computational mesh is directly created by the engineer using the CAD tools.
The proposed IGDG method is the DG method formulated on elements that exactly preserve the geometry generated by CAD tools while the PDE solution exhibits discontinuities at element interfaces. An important property of B-spline in the context of DG is the ability to perform Bézier extraction. Bézier extraction provides the capability of recovering a local Bernstein-Bézier representation of the geometry from the global B-spline CAD. In this chapter, we will discuss specific details of the implementation of IGDG method for the one-dimensional linear advection problem in contrast to the DGFE method.

7.2 DGFE framework for one-dimensional scalar conservation law

The DGFE method was first designed as an effective numerical method for solving hyperbolic conservation laws [72] [93]. In this section, we will present the details of the DGFE method, the stability analysis, and the error estimates for the one-dimensional scalar conservation law given by:

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + \frac{\partial f(u(x,t))}{\partial x} & = 0 & \forall (x,t) \in \Omega \times [0,T], \\ u(x,0) & = u_0(x) & \forall x \in \Omega, \\ u(\mathrm{a},t) & = u_{\mathrm{a}}(t) & \forall t \in [0,T]. \end{cases} \tag{7.1}$$

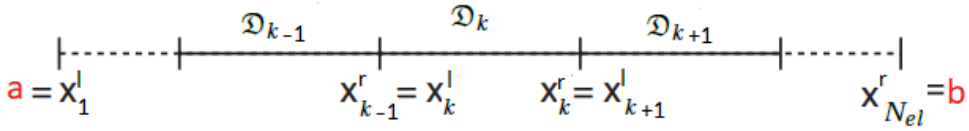
(with $\Omega = [\mathrm{a},\mathrm{b}]$ and $f'(u) > 0 \quad \forall x \in \Omega$).

7.2.1 Discontinuous Galerkin-space discretization

In the DG method, the domain $\Omega = [\mathrm{a},\mathrm{b}]$ is subdivided into a union of finite number N_{el} of cells $\{\mathfrak{D}_k\}_{k=1}^{N_{el}}$, each element being delimited by two nodes of coordinates x_k^l and x_k^r, such that:

$$\Omega = [\mathrm{a},\mathrm{b}] = \bigcup_{k=1}^{N_{el}} \mathfrak{D}_k \qquad \text{with} \quad \mathfrak{D}_k \bigcap \mathfrak{D}_l = \emptyset \quad \forall 1 \leq k \neq l \leq N_{el}$$

We note that: $x_1^l = \text{a}$ and $x_{N_{el}}^r = \text{b}$.



We define uniform N_{el} cells $\mathfrak{D}_k$ of length $\mathfrak{h}$ by:

$$\mathfrak{D}_k = [x_k^l, x_k^r] \quad \text{and} \quad \mathfrak{h} = x_k^r - x_k^l, \quad \forall 1 \le k \le N_{el}.$$

We denote by $\mathcal{T}$ the subdivision of Ω into N_{el} elements $\mathfrak{D}_k$,

$$\mathcal{T} = \Big\{\mathfrak{D}_k, \quad 1 \le k \le N_{el}\Big\}.$$

The starting point for the derivation of a DG scheme is obtained by multiplying Eq. (7.1) by a polynomial test function v on each cell and then integrating over an arbitrary subset $\mathfrak{D}_k$ of $\mathcal{T}$:

$$\int_{\mathfrak{D}_k} \frac{\partial u(x,t)}{\partial t} v(x) dx + \int_{\mathfrak{D}_k} \frac{\partial f(u(x,t))}{\partial x} v(x) dx = 0, \qquad \forall t \in [0,T].$$

We underline that this integration is carried-out over a single element $\mathfrak{D}_k$ and not over the whole computational domain Ω, as done in a classical FE approach. After integration by parts, one obtains:

$$\int_{\mathfrak{D}_k} \frac{\partial u(x,t)}{\partial t} v(x) dx = \int_{\mathfrak{D}_k} f(u(x,t)) \frac{\partial v(x)}{\partial x} dx + f\big(u(x_k^l,t)\big) - f\big(u(x_k^r,t)\big), \qquad \forall t \in [0,T].$$

The DG method represents the unknowns like the FEM by piecewise polynomial functions, but unlike FEM the polynomials are discontinuous at the cell interfaces. A numerical flux is defined at the cell interface in the same way as for Finite Volume (FV) methods. So, on each cell $\mathfrak{D}_k$, the discrete unknown u_h^k is represented as a linear combination of well chosen basis functions of the space of polynomials of degree p. Then, the finite-dimensional subspace $\mathcal{V}_h^p$ is defined as:

$$\mathcal{V}_h^p = \Big\{ v \in L^2(\Omega) \quad | \, v_{|_{\mathfrak{D}_k}} \in \mathbb{P}_p(\mathfrak{D}_k) \quad \forall 1 \le k \le N_{el}, \quad \mathfrak{D}_k \in \mathcal{T} \Big\},$$

where $\mathbb{P}_p(\mathfrak{D}_k)$ represents the space of polynomials of degree up to p defined on the element $\mathfrak{D}_k$. Notice that functions in $\mathcal{V}_h^p$ are discontinuous across cell interfaces.

As a consequence, the flux f evaluated at the interfaces of the element $\mathfrak{D}_k$ is replaced by a numerical flux function $\widehat{f}$, due to the fact that u_h is a priori discontinuous at the interfaces. Therefore, the local approximate solution u_h^k is then determined as the unique solution of the following weak formulation:

For each element $\mathfrak{D}_k \in \mathcal{T}$:

$$\int_{\mathfrak{D}_k} \frac{\partial u_h^k(x,t)}{\partial t} v_h(x) dx = \int_{\mathfrak{D}_k} f\big(u_h^k(x,t)\big) \frac{\partial v_h(x)}{\partial x} dx + \widehat{f}_k^l - \widehat{f}_k^r, \quad \forall t \in [0,T], \quad \forall v_h \in \mathcal{V}_h^p. \tag{7.2}$$

Note that, for the boundary element $\mathfrak{D}_1$, the numerical flux for the left edge is defined using the given boundary condition:

$$\widehat{f}_1^l = f(u_{\mathrm{a}}(t)).$$

For the element $\mathfrak{D}_{N_{el}}$, the numerical flux for the right edge is evaluated using interior solution:

$$\widehat{f}_{N_{el}}^r = f(u_h^{N_{el}}(t)).$$

For interior interfaces, a basic idea is to use a numerical flux $\widehat{f}$ that is defined according to the left and right values of the solution at the interface, as in a classical FV approach:

$$\widehat{f}_k^l = \widehat{f}\big(u_h^{k-1}(x_k^{l-}, t), u_h^k(x_k^{l+}, t)\big),$$

$$\widehat{f}_k^r = \widehat{f}\big(u_h^k(x_k^{r-}, t), u_h^{k+1}(x_k^{r+}, t)\big),$$

with, $u_h^k = u_{h|_{\mathfrak{D}_k}}$.
Naturally, the choice of the flux is important.

7.2.2 Numerical flux

To complete the definition of the approximate solution u_h, it only remains to choose the numerical flux $\widehat{f}$. An approximation to the true flux [5] at an element's interface is defined by considering the following conditions:

- A numerical flux is defined using interface solution values regardless of the polynomial space chosen for the solution,

  $$\widehat{f} = \widehat{f}\big(u_h^{k^+}, u_h^{k^-}\big),$$

  where $u_h^{k^+}$ and $u_h^{k^-}$ are the fields on the two sides of the element interface Γ^k.

- Consistency: For continuous solutions, the numerical flux must be equivalent to the normal flux at the interface:

  $$\widehat{f}(u_h^k, u_h^k) = f(u_h^k).$$

- Conservation: If a piecewise constant approximation is used, the discretization results in a monotone FV scheme. This is ensured if we have a conservative flux,

  $$\widehat{f}\big(u_h^{k^+}, u_h^{k^-}\big) - \widehat{f}\big(u_h^{k^-}, u_h^{k^+}\big) = 0.$$

There are many possible choices of the numerical flux [71] satisfying the above properties:

Central flux:

$$\widehat{f}_{CEN} = \frac{1}{2}\big(f(u_h^{k-}) + f(u_h^{k+})\big).$$

Upwind flux:

$$\widehat{f}_{Up} = \begin{cases} f(u_h^{k-}) & \text{if} \quad f'(u_h^k) \geq 0, \\ \\ f(u_h^{k+}) & \text{else.} \end{cases}$$

Lax-Friedrichs flux:

$$\begin{aligned} \widehat{f}_{LF} &= \frac{1}{2}\big(f(u_h^{k-}) + f(u_h^{k+})\big) + \frac{C_{LF}}{2}\big(u_h^{k-} - u_h^{k+}\big) \\ &= \widehat{f}_{CEN} + \frac{C_{LF}}{2}\big(u_h^{k-} - u_h^{k+}\big), \end{aligned}$$

where C_{LF} is the maximum wave speed at the interface:

$$C_{LF} = \max |\, f'(u_h)\,| \qquad \text{for} \quad \min(u_h^{k-}, u_h^{k+}) \leq u_h \leq \max(u_h^{k-}, u_h^{k+}).$$

The Lax-Friedrichs flux adds an extra diffusive term to the central flux in an attempt to smear out instabilities. We will focus on the Lax-Friedrichs flux, in the following work, which is a classical choice in DG methods.

7.2.3 Elementary linear system

Returning now to Eq. (7.2) and substituting the local solution approximation in element $\mathfrak{D}_k$, there are $(p+1)$ equations to be solved for each component of the field corresponding to the $(p+1)$ degrees of freedom. Indeed, if a local basis of $\mathbb{P}_p(\mathfrak{D}_k)$ is chosen and denoted as $\big(\varphi_j^k(x)\big)_{j=1,\dots,p+1}$, for $x \in \mathfrak{D}_k$, we can express the local numerical solution u_h^k as:

$$u_h^k(x,t) = u_{h|_{\mathfrak{D}_k}}(x,t) = \sum_{j=1}^{p+1} u_j^k(t)\varphi_j^k(x), \quad \forall x \in \mathfrak{D}_k, \quad \forall t \in [0,T].$$

If Lagrange polynomials are used as local basis, u_j^k corresponds to the solution value in element k at interpolation point j.

Since the support of the basis functions is restricted to the element $\mathfrak{D}_k$, the left-hand side of equation 7.2 is simply the product of a local mass matrix $\mathbb{M}^k$ of size $(p+1)\times(p+1)$ with the vector of the time-derivative of the degrees of freedom $\partial_t u^k$ for the element $\mathfrak{D}_k$. The right-hand side represents the residual for the element $\mathfrak{D}_k$, composed of a volume integral $\mathbf{R}^k$ and fluxes at the interfaces F_l and F_r, which ensures the coupling with solution in neighboring elements.

Therefore, we should solve the following local system for each element k:

$$\mathbb{M}^k \partial_t u^k = \mathbf{R}^k(u^k) + \widehat{f}_k^l - \widehat{f}_k^r \qquad \forall\, t \in [0,T] \quad k = 1,...,N_{el}, \tag{7.3}$$

for the coefficients:

$$u^k = \begin{pmatrix} u_1^k \\ u_2^k \\ \vdots \\ u_{p+1}^k \end{pmatrix}.$$

The global solution $u(x,t)$ is then assumed to be approximated by the piecewise p-th order polynomial approximation $u_h(x,t)$:

$$u(x,t) \simeq u_h(x,t) = \bigoplus_{k=1}^{N_{el}} u_h^k(x,t),$$

defined as the direct sum of the N_{el} local polynomial solutions $u_h^k(x,t)$.

7.3 Computation of residual and mass matrix

Contrary to the FEM, only local matrices on each element (in practice only the elementary matrices on the reference interval $[-1,1]$) need to be assembled. The coefficients of the local mass matrix for the element $\mathfrak{D}_k$ are:

$$(\mathbb{M}^k)_{ij} = \int_{\mathfrak{D}_k} \varphi_i^k(x)\varphi_j^k(x)\,dx \qquad i,j = 1,...,p+1.$$

The first contribution to the residual is the integral over each element $\mathfrak{D}_k$ of the flux multiplied by the test function gradient:

$$\left(\mathbf{R}^k(u^k)\right)_{ij} = \int_{\mathfrak{D}_k} f\left(u_h^k(x,t)\right)_j \partial_x \varphi_i^k(x)\,dx \qquad i,j = 1,...,p+1.$$

The numerical integration is performed using the Gauss-Legendre quadrature rules described above.

7.4 CFL condition for DG Method

An important feature of the above-presented method is the use of a local mass matrix, which results from the local support of basis functions. As a consequence, the local mass matrix can be inverted easily before time integration and the method is well suited to high-order explicit time integration, like RK methods, and is highly parallelizable. Therefore, in the present work, $RK2$ and $RK4$ are used for time integration. Because we are focusing on DG schemes, we discuss the limitation on the C_{CFL} number when the DG method is used in conjunction with the RK time integration approach. A stability condition on the size of the timestep must indeed be satisfied. It corresponds to the Courant Friedrichs-Lewy (CFL) condition:

$$\lambda \frac{\Delta t}{\mathfrak{h}} \leq C_{CFL},$$

where $\lambda = \max_u \mid f'(u) \mid$, $\mathfrak{h}$ is the smallest element width, and Δt is the length of the time step.

Physically this condition bounds the size of the timestep to ensure the physical features of the solution are resolved over the mesh. It is also noteworthy to realise that numerical stability is ensured by bounding the CFL number by $(2p+1)^{-1}$, i.e.:

$$\lambda \frac{\Delta t}{\mathfrak{h}} \leq \frac{1}{2p+1},$$

where p is the degree of the approximating polynomial (when polynomial of degree p is used, a RK of order $(p+1)$ must be used to recover optimal convergence rate). This condition has been proven for the polynomial order $p = 0$ and $p = 1$, there is no analytical proof for higher order polynomials as for as we know [20].

7.5 Numerical results

We study the DGFE method in conjunction with RK for the time disretization for the one-dimensional advection problem:

$$\begin{cases} \frac{\partial u(x,t)}{\partial t} + c\frac{\partial u(x,t)}{\partial x} & = 0 & \forall (x,t) \in [-1,1] \times [0,T] \quad c > 0, \\ u(x,0) & = u_0(x) = \sin(2\pi x) & \forall x \in [-1,1], \end{cases} \tag{7.4}$$

and left boundary condition,

$$u(-1,t) = \sin(2\pi(-1-ct)).$$

The effect of $1D$ advection is thus to move an initial distribution with speed $c > 0$. To complete the numerical scheme, the Lax-Friedrichs flux was chosen, where $C_{LF} = c$. We get:

$$\widehat{f}_{LF}(u_h^{k-}, u_h^{k+}) = \frac{1}{2}\big(f(u_h^{k-}) + f(u_h^{k})\big) + \frac{C_{LF}}{2}\big(u_h^{k-} - u_h^{k+}\big) = cu_h^{k-} \qquad \forall 2 \leq k \leq N_{el} - 1.$$

which is equivalent to the upwind flux in this case.
Setting $c = 1$, an exact solution can be found,

$$u_{ex}(x,t) = \sin(2\pi(x-t)).$$

To test the validity of the classical DG method in conjunction with the RK method for the time discretisation, the L^2-error of the difference between the numerical and exact solution across the domain was computed. Hesthaven et al. [39] show that at time T the error measured in the L^2-norm should be of the form:

$$\| u_{ex}(T) - u_h(T) \|_{L^2(\Omega)} \leq C(N_p)(\mathfrak{h})^{N_p}(1 + C_1(N_p)T),$$

where $\mathfrak{h}$ is the smallest elements size, $N_p (= p+1)$ is the number of nodes per element and C, C_1 are constants independent of $\mathfrak{h}$.
Table 7.1 displays the L^2-error as a function of the number of elements N_{el}, and polynomial degree, $p = N_p - 1$.

N_{el}	16	32	64	128	256	512	1024
$p=1$	$6.0454E-02$	$1.528E-02$	$3.797E-03$	$9.449E-04$	$2.450E-04$	$5.963E-05$	$1.495E-05$
$rate$	–	1.98	2	2	1.94	2.03	1.99
$p=2$	$5.521E-03$	$6.896E-04$	$8.635E-05$	$1.077E-05$	$1.360E-06$	$1.593E-07$	$1.950E-08$
$rate$	–	3	2.99	3	2.98	3.09	3.03

Table 7.1: L^2-errors for the $1D$ advection problem.

As we can see, Table 7.1 demonstrates that greater accuracy can be achieved by either increasing the polynomial order or increasing the number of elements in the domain. The convergence rates were estimated by fixing the polynomial degree, and measuring the L^2-error as a function of the number of elements. Assuming a relationship of the form:

$$\| u_{ex}(T) - u_h(T) \|_{L^2(\Omega)} = C(T)(\mathfrak{h})^{rate}, \tag{7.5}$$

the convergence rate is estimated as the best slope fit of $\ln\left(\| u_{ex}(T) - u_h(T) \|_{L^2(\Omega)} \right)$ against $\ln\left(N_{el}\right)$ for each polynomials degree, as demonstrated in Fig. 7.1. The graphs indicate that $rate \approx p+1$, in agreement with the estimate (7.5), which corresponds to the optimal rate.

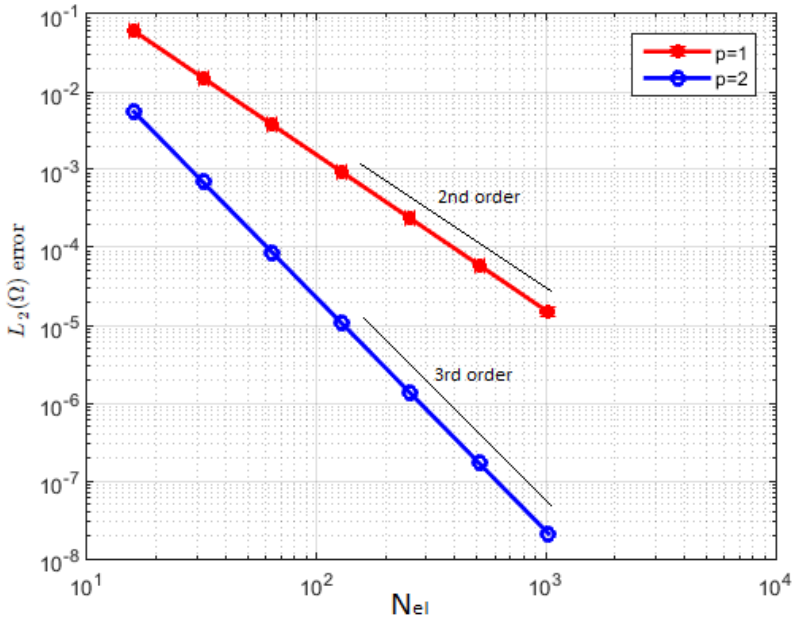


Figure 7.1: L^2-errors for the $1D$ advection problem using the DGFE method in conjunction with the RK method for a sinusoidal initial condition and Lax-Friedrichs flux.

7.6 Isogeometric - discontinuous Galerkin framework (IGDG)

In this section, we present a method that combines isogeometric analysis (IGA) with the discontinuous Galerkin (DG) method for solving hyperbolic equations. The basis functions are continuous within each patch, and discontinuous only on patch boundaries. We have already mentioned that the IGA space is defined using patches rather than elements as in FEA spaces. Therefore, the DG application in IGA is a patch-to-patch relation instead of an element-to-element one. This fact is important to remember, since every time we refer to partitions in the domain, we are in fact referring to patches and not to elements.
As in the previous part of the present chapter we have already seen the technical part of DGFEM, it is now sufficient to focus on the necessary adjustments for IGDG.

7.6.1 Construction of the DG basis

In order to apply the IGDG methodology for the problem (5.1), we assume that the physical domain Ω is represented by a set of non overlapping B-splines patches $\mathfrak{D}_l$:

$$\Omega = \bigcup_l \mathfrak{D}_l \qquad \mathring{\mathfrak{D}}_l \bigcap \mathring{\mathfrak{D}}_{l'} = \emptyset \qquad \forall \quad l \neq l'.$$

As it is common in the IGA analysis, we assume a parametric domain $\widetilde{\mathfrak{D}}_l$ composed of the knots $(\xi_1, ..., \xi_{n+p+1})$. We propose to consider as DG patches the intervals delimited in the parametric domain $\widetilde{\mathfrak{D}}_l$ by all the knots $(\xi_1, ..., \xi_{n+p+1})$.
Therefore, in the framework of the DG method, we can define a set of "elements" delimited by the interfaces x_k^l and x_k^r (assuming $p+1$ equal knots at domain extremities) for each patch $\mathfrak{D}_l$:

$$x_k^l = x(\xi_{k+p}) \quad \text{and} \quad x_k^r = x(\xi_{k+p+1}), \qquad k = 1, ..., n-p = N_{el}.$$

Thus, this baseline discretization fits the definition of the computational domain. If a finer discretization is required, additional knots can be inserted locally in the representation, without any modification of the geometry, using the knot-insertion procedure described in chapter 3.
To complete the construction of the DG framework, it only remains to define for each interval $\mathbb{D}_k = [x_k^l, x_k^r]$ the basis functions $(\Phi_j^k)_{j=1,...,p+1}$. The B-spline basis functions cannot be used directly, because they do not exhibit discontinuities at the interfaces. A modification of the basis should therefore be achieved before. A straightforward approach consists in using again the knot-insertion procedure p times, for each of the existing interior knots $(\xi_{p+2}, ..., \xi_n)$ as illustrated in section 3.7.
By doing so, the computational domain is split into a set of Bézier patches, without modification of the geometry. A Bézier patch is a particular case of B-spline patch, for which the number n of functions (and control points) is equal to $p+1$. As a consequence, $p+1$ basis functions $(\Phi_j^k(x))_{j=1,...,p+1}$ are defined in each interval $\mathbb{D}_k$, which can be identified with Bernstein polynomials of degree p in the parametric domain (with a change of parameter from $[0,1]$ to $[\xi_{k+p}, \xi_{k+p+1}]$).

In practice, basis functions used to represent the solution in the physical domain are defined in the parametric domain by:

$$\Phi_j(x)_{|_{\mathbb{D}_k}} = \Phi_j^k(x(\xi)) \begin{cases} \mathfrak{B}_j^{k,p}(x) = B_j^{k,p}(\xi) & \text{if} \quad x \in \mathfrak{D}_k, \\ 0 & \text{otherwise}, \end{cases} \tag{7.6}$$

where $B_j^{k,p}$ is the j-th Bernstein polynomial of degree p defined over the interval $]\xi_{k+p}, \xi_{k+p+1}[$.
The generation of the Discontinuous Galerkin basis from a B-spline (or NURBS) representation is illustrated in Fig. 7.2:

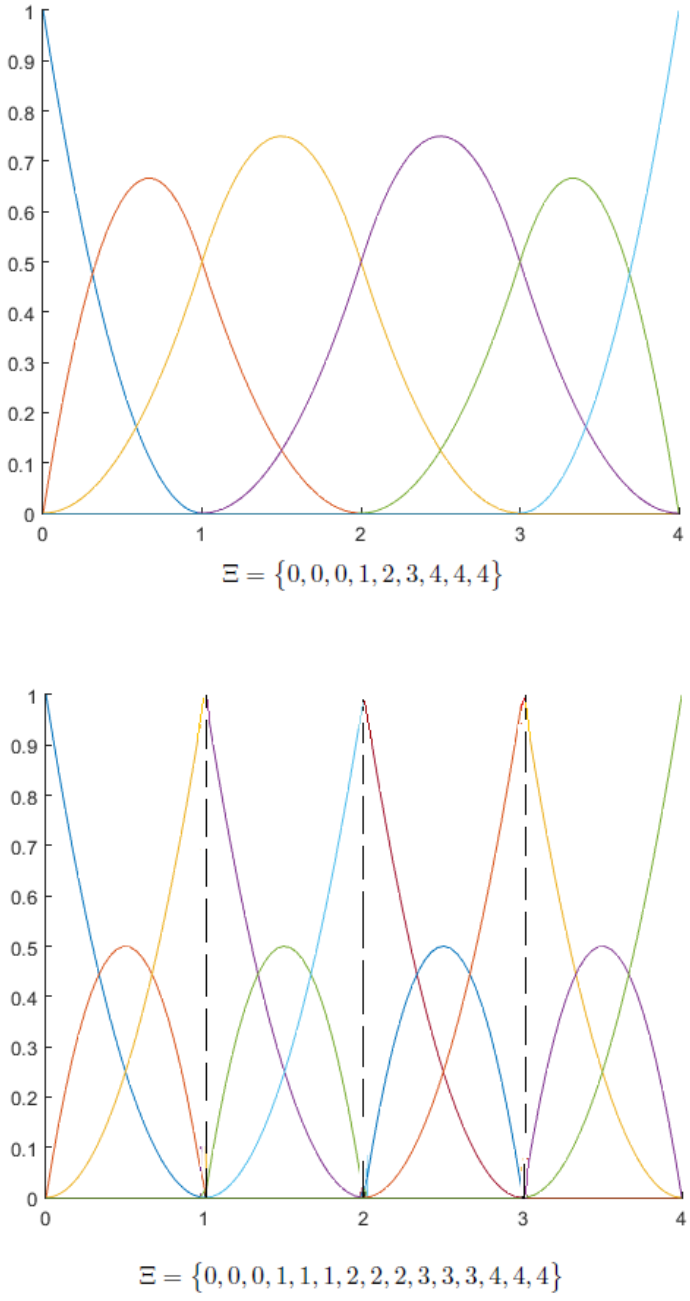


Figure 7.2: Bézier decomposition (bottom) from a quadratic B-spline basis (top) by knot insertion.

The resulting representation has several suitable properties:

- $(\Phi_j^k)_{j=1,...,p+1}$ are non-zero for $x \in]x(\xi_{k+p}), x(\xi_{k+p+1})[$.
- $(\Phi_j^k)_{j=1,...,p+1}$ are C^∞ for $x \in]x(\xi_{k+p}), x(\xi_{k+p+1})[$.
- Φ_1^k and Φ_{p+1}^k are equal to one at $x^+(\xi_{k+p})$ and $x^-(\xi_{k+p+1})$.
- $\sum_{j=1}^{p+1} \Phi_j^k(x) = 1 \ \forall x \in]x(\xi_{k+p}), x(\xi_{k+p})[$ (partition of unity).
- $u_{h|_{\mathbb{D}_k}}(x^+(\xi_{k+p})) = u_1^k$ (local solution interpolates first degree of freedom).
- $u_{h|_{\mathbb{D}_k}}(x^-(\xi_{k+p+1})) = u_{p+1}^k$ (local solution interpolates last degree of freedom).
- $\min_j u_j^k \leq u_{h|_{\mathbb{D}_k}} \leq \max_j u_j^k$ (convex hull property).

These properties will be exploited to define an efficient numerical procedure, as described below.

7.6.2 Isogeometric discontinuous Galerkin approximation spaces

Let us now consider the physical element $\mathbb{D}_k$, that can be exactly parametrized with a mapping $\mathbb{T}$:

$$\mathbb{T} : \widetilde{\mathbb{D}}_k \quad \longmapsto \mathbb{D}_k, \qquad \xi \longmapsto x(\xi).$$

Thus, any point of coordinate x in the physical domain $\mathbb{D}_k$ is mapped to a point of parameter ξ in the parametric domain $\widetilde{\mathbb{D}}_k$. The transformation is defined by associating a control point to each basis function:

$$x(\xi) = \sum_{j=1}^{p+1} B_{j,p}^k(\xi) P_j.$$

P_j, for $j = 1, ..., p+1$ are defined by the knot insertion procedure as described above.

We will revisit the one-dimensional advection problem that is given in (5.1). Using the Bernstein basis functions in the element $\mathbb{D}_k$, $u_{h|_{\mathbb{D}_k}}$ is described as:

$$u_{h|_{\mathbb{D}_k}}(x,t) = u_h^k(x,t) = \sum_{j=1}^{p+1} \mathfrak{B}_{j,p}^k(x) u_j^k(t).$$

For the derivation of a IGDG scheme, equation (5.1) is multiplied by a polynomial test function on each local patch $\mathbb{D}_k$, $\forall 1 \leq k \leq N_{el}$ which can be choosen equal to the Bernstein basis function and after integration by parts and introducing the numerical flux $\widehat{f}$, we get:

$$\sum_{j=1}^{p+1} \partial_t u_j^k(t) \int_{\mathbb{D}_k} \mathfrak{B}_{i,p}^k(x) \mathfrak{B}_{j,p}^k(x) dx = c \sum_{j=1}^{p+1} u_j^k(t) \int_{\mathbb{D}_k} \mathfrak{B}_{j,p}^k(x) \partial_x \mathfrak{B}_{i,p}^k(x) dx + \widehat{f}(x_k^l, t) - \widehat{f}(x_k^r, t) \forall t \in [0,T] \quad k = 1, ..., N_{el}.$$

Since the support of the basis functions is restricted to the element $\mathbb{D}_k$, the left-hand side of the previous equation is simply the product of a local mass matrix $\mathscr{M}^k$ of size $(p+1)\times(p+1)$ with the vector of the time-derivative of the degrees of freedom $\partial_t u^k$ for the element $\mathbb{D}_k$. The right-hand side represents the residual for the element $\mathbb{D}_k$, composed of a volume integral $\mathfrak{R}^k$ and fluxes at the interfaces $\widehat{f}_k^l$ and $\widehat{f}_k^r$, which ensures the coupling with solution in neighboring elements. Therefore, the problem can be expressed in local matrix form:

$$\mathscr{M}^k \partial_t u^k = \mathfrak{R}^k u^k + \widehat{f}_k^l - \widehat{f}_k^r \qquad \forall\, t\in[0,T] \quad k=1,...,N_{el}. \tag{7.7}$$

As for the DGFEM, for the first and last elements, the boundary fluxes $f(u_\mathrm{a})$ and $f(u_\mathrm{b})$ replace $\widehat{f}_1^l$ and $\widehat{f}_{N_{el}}^r$ respectively. Note that, if one uses a piecewise constant representation $p=0$, the mass matrix is just a scalar equal to the element volume, the volumic residual is zero and one recovers a classical first-order FV method.

7.6.3 Computation of residual and mass matrix

The coefficients of the local mass matrix and stiffness matrix for the element $\mathbb{D}_k$ are:

$$(\mathscr{M}^k)_{ij} = \int_{\mathbb{D}_k} \mathfrak{B}_{i,p}^k(x)\mathfrak{B}_{j,p}^k(x)\,dx \qquad i,j=1,...,p+1,$$

$$(\mathfrak{R}^k)_{ij} = c\int_{\mathbb{D}_k} \mathfrak{B}_{j,p}^k(x)\partial_x\mathfrak{B}_{i,p}^k(x)\,dx \qquad i,j=1,...,p+1.$$

By performing the integration in the parametric cell $\widetilde{\mathbb{D}}_k$, we obtain:

$$(\mathscr{M}^k)_{ij} = \int_{\widetilde{\mathbb{D}}_k} B_{i,p}^k(\xi)B_{j,p}^k(\xi)\big(J_\xi^k\big)\,d\xi \qquad i,j=1,...,p+1,$$

and

$$(\mathfrak{R}^k)_{ij} = c\int_{\widetilde{\mathbb{D}}_k} B_{j,p}^k(\xi)\partial_\xi B_{i,p}^k(\xi)\,d\xi \qquad i,j=1,...,p+1,$$

where we denote $J_\xi^k = \left(\frac{\partial x}{\partial\xi}\right)_{|_{\mathbb{D}_k}}$.

The computation is achieved using Gauss-Legendre quadrature rules:

$$(\mathscr{M}^k)_{ij} = \sum_{l=1}^{n_G} B_{i,p}^k\big(X^G(l)\big)B_{j,p}^k\big(X^G(l)\big)\omega_l^G\big(J_{X^{\hat{G}}(l)}^k\big) \qquad i,j=1,...,p+1,$$

where $(X^G(l))_{l=1,...,n_G}$ and $(\omega_l^G)_{l=1,...,n_G}$ are quadrature abscissae and weights. Its inverse is computed numerically, in a pre-processing phase.

$$(\mathfrak{R}^k)_{ij} = c\sum_{l=1}^{n_G} B_{j,p}^k\big(X^G(l)\big)\partial_{\hat{\xi}}B_{i,p}^k\big(X^G(l)\big)\omega_l^G \qquad i,j=1,...,p+1.$$

The solution in each element $\mathbb{D}_k$ interpolates the local first and last degrees of freedom u_1^k and u_{p+1}^k. Therefore, the flux computation only depends on the two degrees of freedom located at each interface:

$$\widehat{f}(x_k^l,t) = \widehat{f}\big(u_{p+1}^{k-1},u_1^k\big),$$

$$\widehat{f}(x_k^r,t) = \widehat{f}\big(u_{p+1}^k,u_1^{k+1}\big).$$

7.7 Numerical studies

Now, we will revisit the one-dimentional advection problem (5.1) by using the developed IGDG method. We will give the numerical results demonstrating the performance of the IGDG method in conjunction with RK time discretization.

We start with an initial B-spline patch which is formed on $\Omega = [-1, 1]$. This uniform patch is used to solve the test problem. The solution is computed up to $T = 0.4$. The initial patch is composed of equal knot intervals of size $\mathfrak{h}$, and is split into a set of Bézier elements as explained in section 7.6.1 to apply the DG formulation.

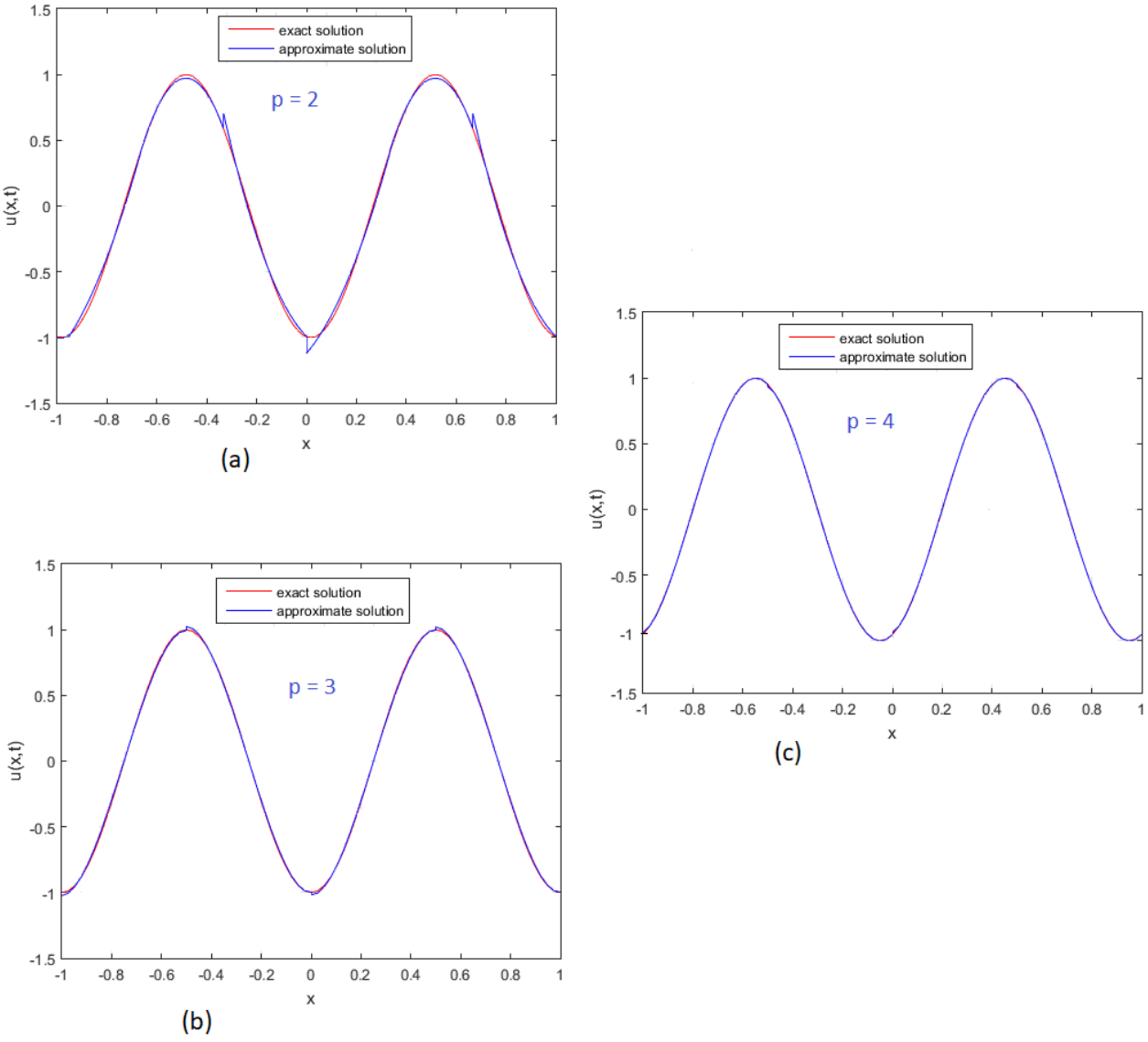


Figure 7.3: IGDG solution of the $1D$ advection problem for quadratic (a), cubic (b) and quartic (c) basis and exact solution with 4 Bézier elements. $RK4$ time discretization and Lax-Friedrichs flux were used.

The solutions obtained numerically are compared with the exact solutions in terms of L^2 norm, for basic functions of degrees 0, 1, 2, 3 and 4. The plots of the numerical solutions obtained using only 4 elements and a quadratic, cubic and quartic basis are depicted in Figure 7.3.

We underline the accuracy of the solution obtained with a so small number of elements. Note also that the discontinuities at the interfaces are decreasing as p increases.

One method to test the convergence of a spatial discretisation is by computing the error between the numerical solution and the analytical one. The numerical approximation error in L^2-norm:

$$\| e \|_{L^2(\Omega)} = \| u_{ex}(T) - u_h(T) \|_{L^2(\Omega)},$$

for various element sizes $\mathfrak{h}$ and degrees p of the Bernstein basis and it is depicted in Fig.7.3. The convergence rates are shown in Table 7.2 and Table 7.3. An optimal convergence rate is observed, the method being of order $p+1$ with respect to L^2- norm.

We notice that, as expected, the convergence rate is limited when using $RK2$ time integrator. Therefore, we employ $RK4$ method to recover optimal convergence rate for $p \geq 2$.

p	$\mathfrak{h}$	$\frac{\mathfrak{h}}{2}$	$\frac{\mathfrak{h}}{4}$	$\frac{\mathfrak{h}}{8}$	$\frac{\mathfrak{h}}{16}$	rate
0	$4.649E-01$	$3.054E-01$	$1.699E-01$	$8.841E-02$	$4.477E-02$	1
1	$5.137E-02$	$1.322E-02$	$3.255E-03$	$8.176E-04$	$2.024E-04$	2
2	$4.255E-03$	$7.179E-04$	$1.594E-04$	$3.581E-05$	$8.652E-06$	2.5

Table 7.2: L^2-error for the IGDG method in conjunction with $RK2$ time discretisation for various element sizes and degree of Bézier basis $p = 0,1,2$.

p	$\mathfrak{h}$	$\frac{\mathfrak{h}}{2}$	$\frac{\mathfrak{h}}{4}$	$\frac{\mathfrak{h}}{8}$	$\frac{\mathfrak{h}}{16}$	rate
2	$3.692E-03$	$4.671E-04$	$5.758E-05$	$7.083E-06$	$8.709E-07$	3
3	$9.299E-05$	$5.724E-06$	$3.322E-07$	$1.849E-08$	$1.062E-09$	4
4	$8.069E-06$	$2.497E-07$	$7.767E-09$	$2.359E-10$	$7.241E-12$	5

Table 7.3: L^2-error for the IGDG method in conjunction with $RK4$ time discretisation for various element sizes and degree of Bézier basis $p = 2,3,4$.

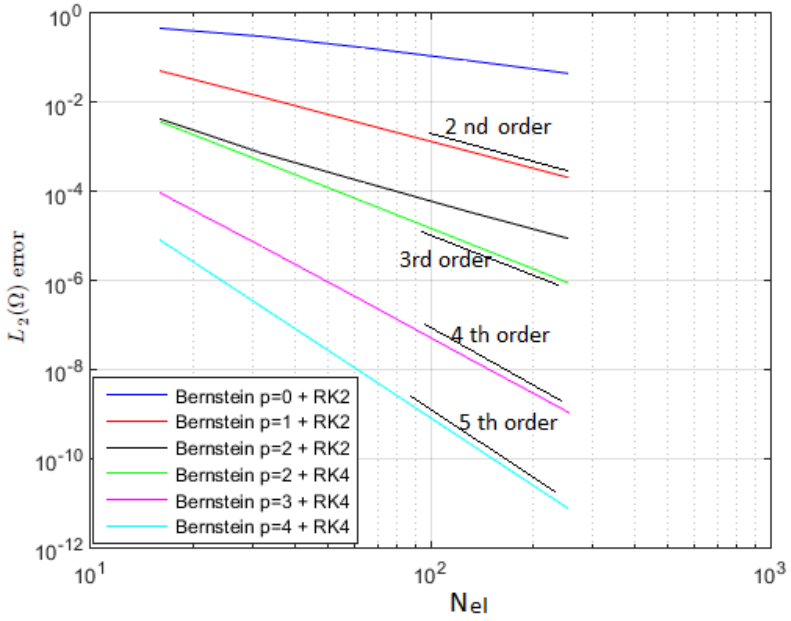


Figure 7.4: L^2-errors for the $1D$ advection problem with a sinusoidal initial condition, $RK2$ and $RK4$.

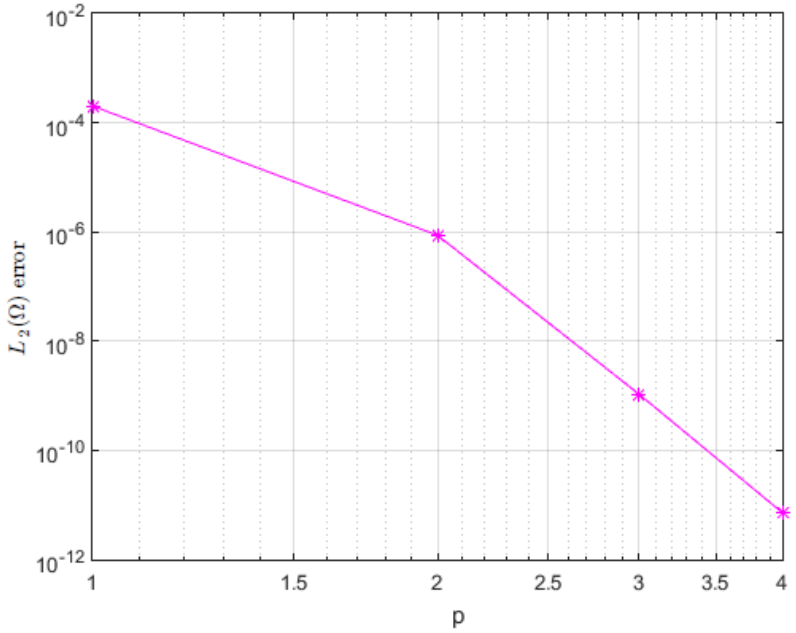


Figure 7.5: Convergence rates in IGDG method as a function of the Bernstein function is degree p for the finest grid.

7.8 Conclusion and comparison

In this chapter a new family of discontinuous Galerkin methods which combines the IGA with the DG method, called IGDG method, has been developed for the one-dimensional advection problem. Our method takes advantage of both IGA and DG methods. In fact, DG methodology is adopted at Bézier patch level, i.e., we employ the traditional IGA within each Bézier patch, and employ the DG method across the patch interfaces to glue the multiple patches. Bézier patches, considered as elements, are constructed by transformation of the initial B-spline domain. Due to IGA, all conic sections can be represented exactly, thus eliminating the geometrical errors by the construction. Obviously, this property will be more evident for $2D$ problems.

As mentioned before, the major reason for using DG methods lies with their ability to provide stable numerical methods for first order PDE problems, for which classical FEM is well known to perform poorly. Due to the piecewise discontinuity of basis functions, the DG method can be applied locally in each element. This simplifies the implementation of the method, since the mass matrix becomes block diagonal and the solution of a large system is avoided. The solution for the whole computational domain is achieved by summing over all the elements of the mesh.

Compared to SUPG-FEM for solving hyperbolic problems, an attractive feature of the DG method is that it still uses the basic Galerkin method for the volume integrals, although this is extended by element boundary integrals to achieve the 'upwinding'. Because of the simple concept of the method it is basically the same for any space dimension. The treatment of boundary conditions is also relatively easy.

Now we focus our attention on the comparison of the numerical methods combining IGFEM, DGFEM and IGDG space discretization and explicit RK time integration, for the one-dimensional advection problem.

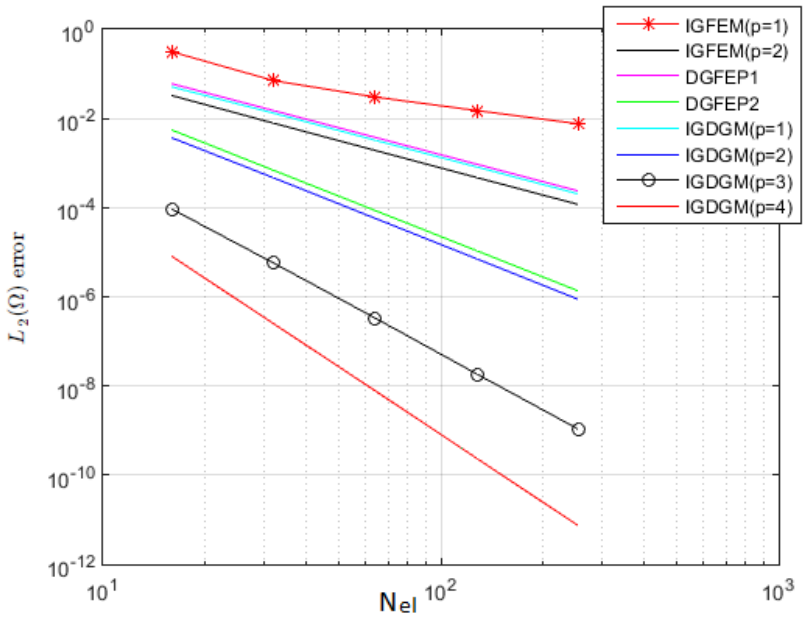


Figure 7.6: Error in the L^2-norm combining IGFEM, DGFEM, IGDG space discretization and explicit RK time integration.

Figure 7.6 shows that DG methods are far more accurate than FEM. The comparison between the classical DG and IGDG methods is less clear, because both method gives almost the same accuracy with optimal convergence results with respect to the L^2-norm.

CONCLUSION

La méthode de Galerkin discontinue a initialement été introduite par Reed et Hill [74] en 1973, pour la discrétisation des équations caractérisant le transport de neutrons. C'est une méthode de Galerkin dont les fonctions tests sont polynomiales par morceaux, mais sans contrainte de continuité. La base de fonctions tests est définie localement sur des sous-domaines par des polynômes d'ordre au plus p, et est identiquement nulle partout ailleurs. En conséquence, la différence majeure avec les EF classiques provient de la non imposition de la continuité de la solution numérique au niveau des frontières inter-éléments. L'intégration par parties conduit donc à l'apparition d'intégrales sur ces frontières, qui sont évaluées de manière analogue à l'approche des volumes finis, en introduisant des flux numériques des valeurs des fonctions de part et d'autre de l'interface.

Grâce à une grande flexibilité d'approximation locale, la méthode de GD offre de bonnes propriétés de stabilité [79] [93]. Contrairement à la méthode de Galerkin continue stabilisée, la méthode de GD produit des discrétisations stables pour les problèmes hyperboliques sans la nécessité de paramètres de stabilisation, la stabilisation résultant de l'utilisation de flux décentrés. Par conséquent, cette méthode combine les meilleures propriétés de la méthode des volumes finis (VF) et de la méthode d'EF standard.

Bien que la méthode de GD ait attiré de plus en plus d'attention dans les applications de modélisation à grande échelle, l'incapacité à récupérer exactement les géométries sous-jacentes complexes dans le domaine du maillage constitue une lacune de la méthodologie conventionnelle. Pour surmonter ce problème, nous combinons l'AIG avec la méthode de GD. Comme indiqué précédemment, l'AIG est une technique de calcul qui améliore et généralise la méthode d'EF classique. Le principal avantage de cette méthode est la représentation exacte de la géométrie dans le langage des outils de conception assistée par ordinateur (CAO).

La méthode de GD dans le cadre IG (IGGD) proposée est en fait une méthode de GD formulée sur des éléments qui préservent exactement la géométrie générée par les outils de CAO, tandis que la solution de l'EDP présente des discontinuités aux interfaces d'éléments. Une propriété importante des B-splines dans le contexte de GD est la capacité à effectuer une extraction de Bézier. L'extraction de Bézier permet de récupérer une représentation locale de Bernstein-Bézier de la géométrie à partir d'une B-spline globale. Dans ce chapitre, on a discuté des détails spécifiques de l'implémentation de la méthode IGGD pour le problème unidimensinnel d'advection linéaire, ainsi que la méthode de GD classique.

Part IV

$2D$ PROBLEM STUDY

CHAPTER 8

IGDG: 2D ADVECTION PROBLEM

In the previous chapter, we have described the various computational procedures for IGDG analysis especially for the one-dimensional case. This chapter is devoted to the extension to the two-dimensional case. The geometrical representation will obviously play a more critical role in $2D$ cases and we will especially underline the tretment of the geometry. Numerical experiments validate the presented methodology.

8.1 Computational procedures in two dimensions

8.1.1 Preliminaries - IGDG notation

In order to apply the IGA methodology, the physical domain Ω is subdivided into B-spline patches $\mathfrak{D}^e$,

$$\mathscr{S}(\Omega) := \{\mathfrak{D}^e\}_{e=1}^{N_{pa}},$$

such that:

$$\overline{\Omega} = \bigcup \overline{\mathfrak{D}^e} \quad \text{with} \quad \overset{\circ}{\mathfrak{D}}{}^e \bigcap \overset{\circ}{\mathfrak{D}}{}^l = \emptyset \quad \forall \quad e \neq l.$$

Moreover, we denote the interpatch boundary between the two patches $\mathfrak{D}^e$ and $\mathfrak{D}^l$ by $\Gamma^{e,l}$ and the collection of all interfaces by Γ, i.e.,

$$\Gamma^{e,l} = \overline{\mathfrak{D}^e} \bigcap \overline{\mathfrak{D}^l} \quad \text{and} \quad \Gamma^e := \bigcup_{l>e} \Gamma^{e,l}.$$

Furthermore, the boundary of the domain Ω is denoted by $\partial\Omega$.

8.1.2 Isogeometric analysis (IGA): physical domain and geometrical mappings

The main idea behind the isogeometric approach is to discretize the unknowns of the problem with the same set of basis functions that CAD employs for the construction of the geometries.

We use the definitions from IGA that we introduced in chapter 6. In our specific case the physical domain $\Omega \subset \mathbb{R}^2$ is represented by a set of B-spline patches. Each B-spline patch denoted by $\mathfrak{D}^e$, is an image under a B-spline mapping of a parametric domain $\widetilde{\mathfrak{D}}^e$.

$$\mathfrak{D}^e = \left\{ X^e = (x^e, y^e) \in \mathbb{R}^2 \quad | \, X^e = \mathbb{T}(\xi, \eta) \quad \text{such} \quad \text{that} \quad (\xi, \eta) \in \widetilde{\mathfrak{D}}^e \right\},$$

where the transformation $\mathbb{T}$ is defined for all $(\xi, \eta) \in \widetilde{\mathfrak{D}}^e$ by:

$$\mathbb{T} : \widetilde{\mathfrak{D}}^e \quad \longmapsto \quad \mathfrak{D}^e$$

$$(\xi, \eta) \quad \longmapsto \quad \big(x^e(\xi, \eta), y^e(\xi, \eta)\big).$$

Note that this transformation is non linear and is simply defined by the parametric representation of the computational domain. Unless otherwise specified, we assume this mapping to be invertible. Its inverse:

$$\mathbb{T}^{-1} : \mathfrak{D}^e \longmapsto \widetilde{\mathfrak{D}}^e,$$

takes points in the physical domain to their corresponding parameter values. Figure 8.1 gives a schematic overview of our proposed approach.

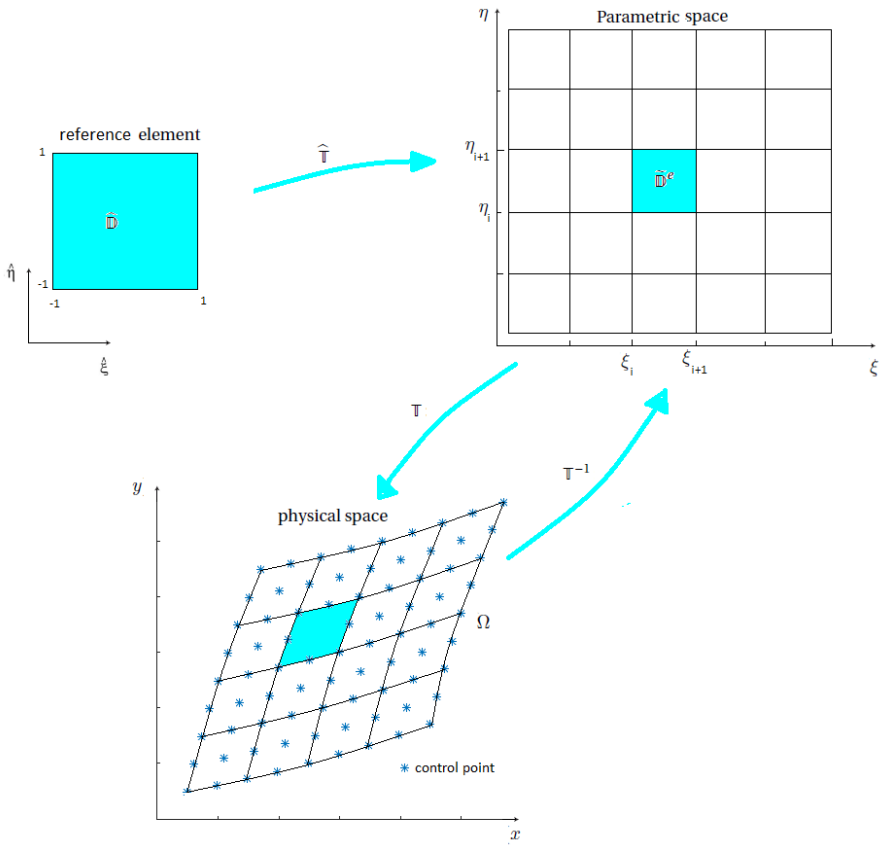


Figure 8.1: An example of a B-spline patch in physical space, parametric space, and the parent element used to perform numerical integration.

8.1.3 Basic function space for the parametric domain and physical domain

In this section we focus on the construction of the basis functions. A C^{p-1} continuous basis $\widetilde{\Phi}(\xi,\eta)$ is constructed over the parameteric domain $\widetilde{\mathfrak{D}}^e$. The basis is used to construct a geometric map $\mathbb{T}$ so that it maps a point $(\xi,\eta)\in\mathbb{R}^2$ in the parametric domain $\widetilde{\mathfrak{D}}^e$ to a point in $\mathfrak{D}^e$.

We have already stated that we consider elements to be the images of knot spans under the Bézier mapping. We will denote these knot spans in the parameter space by $\widetilde{\mathbb{D}}^e$, and their image in the physical space as $\mathbb{D}^e$, where $e=1,...,\mathfrak{N}_{el}$, with $\mathfrak{N}_{el}$ being the total number of elements.

To implement the DG method in the isogeometric framework, i.e. based on a computational domain defined from a B-spline representation, we must first define a set of elements, which are the supports of a polynomial representation with discontinuities at each interface between elements. Given a B-spline surface defining the computational domain, the insertions of knots p times are used for each of the existing interior knot for each parametric direction sequentially. In doing so, the computational domain is divided into a set of Bézier patches, without modification of the geometry. We remind that a Bézier patch is a special case of B-spline patch, for which the number n of functions (and control points) is equal to $p+1$ (with p is the degree of basis function).

Finally, all the Bézier patches created by the insertion process are considered as elements. Each element $\mathbb{D}^e$ is therefore defined by $(p+1)\times(q+1)$ basis functions, $\left(\mathfrak{B}^{p,q}(x,y)\right)^e$, which can be identified with Bernstein polynomials of degrees p and q:

$$\left(\mathfrak{B}^{p,q}(X)\right)^e=\left(\mathfrak{B}^{p,q}(x,y)\right)^e=\left(\mathfrak{B}^{p,q}\left(\mathbb{T}(\xi,\eta)\right)\right)^e=\left(B^{p,q}(\xi,\eta)\right)^e=\begin{cases}\left(B^p(\xi)\right)^e\otimes\left(B^q(\eta)\right)^e & \forall(x,y)\in\mathbb{D}^e,\\ 0 & \text{otherwise.}\end{cases}$$

where, $\left(B_i^p\right)_{i=1,...,p+1}^e$ and $\left(B_j^q\right)_{j=1,...,q+1}^e$ are respectively the $i-$th and the $j-$th Bernstein polynomials of degree p and q, defined on the interval $[0,1]$ by:

$$\left(B_i^p(\xi)\right)^e=C_p^i\xi^i\left(1-\xi\right)^{p-i}\qquad\forall t\in[0,1]\quad i=0,...,p.$$

A given control point P^e will have local indices associated to the Bézier patch $\mathbb{D}^e$. Thus, the geometrical mapping local to element e can be defined as:

$$X^e(\xi,\eta)=\left(x^e(\xi,\eta),y^e(\xi,\eta)\right)=\sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\left(B_i^p(\xi)\right)^e\left(B_j^q(\eta)\right)^eP_{ij}^e.$$

Before we proceed further, we denote by J^e the elementary Jacobian matrix of this transformation defined in $\mathbb{D}^e$ by:

$$J^e=\begin{pmatrix}\frac{\partial x}{\partial\xi} & \frac{\partial x}{\partial\eta}\\ \frac{\partial y}{\partial\xi} & \frac{\partial y}{\partial\eta}\end{pmatrix}.$$

Thus, we can also calculate the inverse of the elementary Jacobian matrix $(J^e)^{-1}$ given by:

$$(J^e)^{-1} = \frac{1}{\mid J^e \mid}\begin{pmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{pmatrix} = \frac{1}{\frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi}}\begin{pmatrix} \frac{\partial y}{\partial \eta} & -\frac{\partial x}{\partial \eta} \\ -\frac{\partial y}{\partial \xi} & \frac{\partial x}{\partial \xi} \end{pmatrix} = \begin{pmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \xi}{\partial y} \\ \frac{\partial \eta}{\partial x} & \frac{\partial \eta}{\partial y} \end{pmatrix}.$$

Then, we define the test functions in the physical domain $\mathbb{D}^e$ by using the same representation as the geometry:

$$\begin{aligned} \Phi^{p,q}(x,y)_{|_{\mathbb{D}^e}} = \Big(\Phi^{p,q}(x,y)\Big)^e = \Big(\mathfrak{B}^{p,q}(x,y)\Big)^e &= \Big(\mathfrak{B}^{p,q}\big(\mathbb{T}(\xi,\eta)\big)\Big)^e \\ &= \big(B^{p,q}(\xi,\eta)\big)^e \\ &= \Big(B^p(\xi)\Big)^e \otimes \Big(B^q(\eta)\Big)^e \\ &= \Big(\widetilde{\Phi}^p(\xi)\Big)^e \otimes \Big(\widetilde{\Phi}^q(\eta)\Big)^e. \end{aligned}$$

We also highlight that, $\forall 1 \le i \le p+1$ and $\forall 1 \le j \le q+1$,

$$\begin{aligned} \Big(\Phi^{p,q}(x,y)\Big)^e_{i,j} &= \Big(B^p(\xi)\Big)^e_i \Big(B^q(\eta)\Big)^e_j \\ &= \Big(\widetilde{\Phi}^p(\xi)\Big)^e_i \Big(\widetilde{\Phi}^q(\eta)\Big)^e_j. \end{aligned}$$

In addition, we need also to compute $\nabla\Phi^e$, which is the gradient in the Cartesian form:

$$\nabla\Phi^e = \begin{pmatrix} \left(\frac{\partial \Phi^{p,q}(x,y)}{\partial x}\right)^e \\ \\ \left(\frac{\partial \Phi^{p,q}(x,y)}{\partial y}\right)^e \end{pmatrix} = \begin{pmatrix} \frac{\partial \widetilde{\Phi}^e}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial \widetilde{\Phi}^e}{\partial \eta}\frac{\partial \eta}{\partial x} \\ \\ \frac{\partial \widetilde{\Phi}^e}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial \widetilde{\Phi}^e}{\partial \eta}\frac{\partial \eta}{\partial y} \end{pmatrix} = \begin{pmatrix} \frac{\partial \xi}{\partial x} & \frac{\partial \eta}{\partial x} \\ \\ \frac{\partial \xi}{\partial y} & \frac{\partial \eta}{\partial y} \end{pmatrix}\begin{pmatrix} \frac{\partial \widetilde{\Phi}^e}{\partial \xi} \\ \\ \frac{\partial \widetilde{\Phi}^e}{\partial \eta} \end{pmatrix}$$

$$\nabla\widetilde{\Phi}^e = \begin{pmatrix} \frac{\partial \widetilde{\Phi}^e}{\partial \xi} \\ \frac{\partial \widetilde{\Phi}^e}{\partial \eta} \end{pmatrix} = \begin{pmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{pmatrix}\begin{pmatrix} \frac{\partial \Phi^e}{\partial x} \\ \frac{\partial \Phi^e}{\partial y} \end{pmatrix} = \big(J^e\big)^T \nabla\Phi^e.$$

Finally, one obtains:

$$\nabla\widetilde{\Phi}^e = \big(J^e\big)^T \nabla\Phi^e,$$

$$\nabla\Phi^e = \big((J^e)^T\big)^{-1} \nabla\widetilde{\Phi}^e.$$

8.1.4 Numerical integration

In this thesis we use Gaussian quadrature as a choice of numerical integration. To do this, we will need to map our functions over to a reference square $(\widehat{\xi},\widehat{\eta}) \in \widehat{\mathbb{D}} = [-1,1] \times [-1,1]$. While this is common for most FEM as well, we have the additional mapping from the parametric space to the physical space. The first one is an affine mapping (see Fig. 8.1):

$$\widehat{\mathbb{T}}: \quad \widehat{\mathbb{D}} \longmapsto \widetilde{\mathbb{D}}^e.$$

The numerical integration has to take place in the reference square, while all basis functions are defined over the parametric domain and the differential equation is formulated in the physical space. Therefore, a typical integration writes:

$$\begin{aligned}\int_{\mathbb{D}^e}\left(\Phi_{i,j}^{p,q}(x,y)\right)^e d\mathbb{D}^e &= \int_{\mathbb{D}^e}\left(\mathfrak{B}_{i,j}^{p,q}(\mathbb{T}(\xi,\eta))\right)^e d\mathbb{D}^e \\ &= \int_{\widetilde{\mathbb{D}}^e}\left(B_{i,p}(\xi,\eta)\right)^e\left(B_{j,q}(\xi,\eta)\right)^e |\,J^e(\xi,\eta)|\,d\widetilde{\mathbb{D}}^e \\ &= \int_{\widehat{\mathbb{D}}}\left(B_{i,p}(\widehat{\xi})\right)^e\left(B_{j,q}(\widehat{\eta})\right)^e |\,\widehat{J}^e(\widehat{\xi},\widehat{\eta})|\,|\,J^e(\xi,\eta)|\,d\widehat{\mathbb{D}},\end{aligned}$$

8.2 $2D$ advection problem: IGDG space semi-discretization

In the following, we describe the discretization of the transient, bi-dimensional, linear conservation law or transport equation over a domain $\Omega \subset \mathbb{R}^2$ with periodic boundary conditions by the IGDG method:

$$\begin{cases} \partial_t u + \nabla\cdot\left(\overrightarrow{c}\,u(x,y,t)\right) &= 0 \qquad\qquad \forall (x,y,t)\in\Omega\times[0,T], \\ u(x,y,0) &= u_0(x,y) \qquad \forall (x,y)\in\Omega, \end{cases} \tag{8.1}$$

where $u(x,y,t)$ is a scalar quantity transported by a continuous velocity field $\overrightarrow{c} = (c_x, c_y)^t$.
Similarly as in the previous chapter, we derive the IGDG space semidiscretization leading to a system of ordinary differential equations. As a matter of fact, the IGDG method for multi-dimensional conservation law has the same structure it has for one-dimensional scalar conservation laws, we only need to describe the DG-space discretization.
Applying a IGDG method, the solution u is approximated by $u_h \in \mathbb{V}^p$, which we assume to have the following form:

$$u_h^e(x,y,t) = \sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\left(B_{i,p}(\xi)\right)^e\left(B_{j,q}(\eta)\right)^e u_{ij}^e(t),$$

where $u_{ij}^e : [0,T] \longmapsto \mathbb{R}, \quad \forall \quad 1\le i\le p+1, \quad 1\le j\le q+1$ are local unknown coefficients.
Before we proceed further, let us put $v_h^e \in \mathbb{V}_h^p$, the approximate solution that will be sought for each $t\in[0,T]$ in the finite-dimensional space:

$$\mathbb{V}_h^p = \left\{ v := v(x(\xi,\eta), y(\xi,\eta)) \in L^2(\Omega), \quad v_{|_{\mathbb{D}^e}} \in span\{B^e\}\right\},$$

where B^e is the set of $(p+1)\times(q+1)$ Bernstein polynomials defined over $\widetilde{\mathbb{D}}^e$.

8.2.1 Isogeometric discontinuous Galerkin space semi-discretization

The starting point for a DG discretization is the weak formulation, which is obtained by multiplying Eq. (8.1) by a local arbitrary test function $v^e(x,y) \in \mathbb{V}^p$, and then integrating on each patch $\mathbb{D}^e$ separately. Note that, in this framework, no continuity on the state vector u^e and the test function v^e is enforced along the interfaces between patches. The variational formulation is given for $e = 1, ..., \mathfrak{N}_{el}$ by:

$$\int_{\mathbb{D}^e} \partial_t u^e(x,y,t)\, v^e(x,y) d\mathbb{D}^e + \int_{\mathbb{D}^e} \nabla \cdot \big(\overrightarrow{c}\, u^e(x,y,t)\big) v^e(x,y) d\mathbb{D}^e = 0. \tag{8.2}$$

By applying Green's formula and introducing the numerical flux $\widehat{f}$, the weak formulation can be written as:

$$\int_{\mathbb{D}^e} \partial_t u^e(x,y,t)\, v^e(x,y) d\mathbb{D}^e = \int_{\mathbb{D}^e} u^e(x,y,t) \overrightarrow{c} \cdot \nabla v^e(x,y) d\mathbb{D}^e - \int_{\Gamma^e} v^e(x,y) \overrightarrow{c}\, u^e \cdot \overrightarrow{n}^e d\Gamma^e. \tag{8.3}$$

We denote by $\overrightarrow{n}^e$ the outer unit normal to Γ^e for the element $\mathbb{D}^e$. We define the restriction of the approximate solution $u_h^e \approx u^e$ to $\mathbb{D}^e$ via:

$$u_h^e(x,y,t)_{|_{\mathbb{D}^e}} = u_h^e(x,y,t) \quad = \quad \mathfrak{B}^e(x,y) u^e(t). \tag{8.4}$$

By discretizing the problem on the Bézier basis associated with the element, the problem can be written:

$$\begin{aligned} \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \partial_t u_{i,j}^e(t) \Big(\int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(x,y) \mathfrak{B}_{k,l}^e(x,y) d\mathbb{D}^e \Big) &= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} u_{i,j}^e(t) \Big(\int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(x,y) \overrightarrow{c} \cdot \nabla \mathfrak{B}_{k,l}^e(x,y) d\mathbb{D}^e \Big) \\ &- \int_{\Gamma^e} \mathfrak{B}_{k,l}^e(x,y) (\overrightarrow{c}\, u^e) \cdot \overrightarrow{n}^e d\Gamma^e \\ \forall 1 \le k \le p+1 \qquad 1 \le l \le q+1. \end{aligned}$$

Therefore, the local problem takes the form of a linear system of size $(p+1)^2 \times (q+1)^2$, which can be written in the following matrix form:

$$\mathbf{M}^e \partial_t u^e = \mathbf{R}^e u^e - F^e \qquad \forall\, t \in [0,T] \quad \forall 1 \le e \le \mathfrak{N}_{el}. \tag{8.5}$$

8.2.2 Elementary linear system

The coefficients of the local mass and stiffness matrix for the element $\mathbb{D}^e$ are written as:

$$\begin{aligned} \mathbf{M}_{kl,ij}^e &= \int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(x,y) \mathfrak{B}_{k,l}^e(x,y) d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(\mathbb{T}(\xi,\eta)) \mathfrak{B}_{k,l}^e(\mathbb{T}(\xi,\eta)) d\mathbb{D}^e, \end{aligned}$$

and,

$$\begin{aligned}\mathbf{R}^e_{kl,ij} &= \int_{\mathbb{D}^e} \big(\vec{c}\,\mathfrak{B}^e_{i,j}(x,y)\big).\nabla\mathfrak{B}^e_{k,l}(x,y)\,d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}(x,y)\Big(c_x\frac{\partial\mathfrak{B}^e_{k,l}(x,y)}{\partial x}+c_y\frac{\partial\mathfrak{B}^e_{k,l}(x,y)}{\partial y}\Big)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}(\mathbb{T}(\xi,\eta))\Big(c_x\frac{\partial}{\partial x}\mathfrak{B}^e_{k,l}(\mathbb{T}(\xi,\eta))+c_y\frac{\partial}{\partial y}\mathfrak{B}^e_{k,l}(\mathbb{T}(\xi,\eta))\,d\mathbb{D}^e.\end{aligned}$$

By performing the integration in the local parametric space $\widetilde{\mathbb{D}}^e$, we get:

$$\mathbf{M}^e_{kl,ij} = \int_{\widetilde{\mathbb{D}}^e}\Big(\big(B_{i,p}(\xi)\big)^e\big(B_{j,q}(\eta)\big)^e\Big)\Big(\big(B_{k,p}(\xi)\big)^e\big(B_{l,q}(\eta)\big)^e\Big)\mid J^e(\xi,\eta)\mid d\widetilde{\mathbb{D}}^e$$
$$\forall i,k=1,...,p+1,\quad j,l=1,...,q+1.$$

and

$$\begin{aligned}\mathbf{R}^e_{kl,ij} &= \int_{\widetilde{\mathbb{D}}^e}\Big(\big(B_{i,p}(\xi)\big)^e\big(B_{j,q}(\eta)\big)^e\Big)\Big[\Big(c_x\big(\frac{\partial B^e_{k,p}(\xi)}{\partial\xi}\big)\big(B^e_{l,q}(\eta)\frac{\partial\xi}{\partial x}\big)+\big(c_xB^e_{k,p}(\xi)\big)\big(\frac{\partial B^e_{l,q}(\eta)}{\partial\eta}\frac{\partial\eta}{\partial x}\big)\Big)\\ &+ \Big(\big(c_y\frac{\partial B^e_{k,p}(\xi)}{\partial\xi}\big)\big(B^e_{l,q}(\eta)\frac{\partial\xi}{\partial y}\big)+\big(c_yB^e_{k,p}(\xi)\big)\big(\frac{\partial B^e_{l,q}(\eta)}{\partial\eta}\frac{\partial\eta}{\partial y}\big)\Big)\Big]\mid J^e(\xi,\eta)\mid d\widetilde{\mathbb{D}}^e,\end{aligned}$$

where $\mid J^e \mid$ is the local Jacobien determinant which is described in section 8.1.3.
By performing the integration in the reference square $\widehat{\mathbb{D}}$, we get:

$$\mathbf{M}^e_{kl,ij} = \int_{\widehat{\mathbb{D}}}\Big(\big(B_{i,p}(\hat{\xi})\big)^e\big(B_{j,q}(\hat{\eta})\big)^e\Big)\Big(\big(B_{k,p}(\hat{\xi})\big)^e\big(B_{l,q}(\hat{\eta})\big)^e\Big)\mid\hat{J}^e(\xi,\eta)\mid\mid J^e(\xi,\eta)\mid d\widehat{\mathbb{D}}$$
$$\forall i,k=1,...,p+1\qquad j,l=1,...,q+1,$$

and,

$$\begin{aligned}\mathbf{R}^e_{kl,ij} &= \int_{\widehat{\mathbb{D}}}\Big(\big(B_{i,p}(\hat{\xi})\big)^e\big(B_{j,q}(\hat{\eta})\big)^e\Big)\Big[\Big(c_x\big(\frac{\partial B^e_{k,p}(\hat{\xi})}{\partial\hat{\xi}}\big)\big(B^e_{l,q}(\hat{\eta})\frac{\partial\hat{\xi}}{\partial x}\big)+\big(c_xB^e_{k,p}(\hat{\xi})\big)\big(\frac{\partial B^e_{l,q}(\hat{\eta})}{\partial\hat{\eta}}\frac{\partial\hat{\eta}}{\partial x}\big)\Big)\\ &+ \Big(\big(c_y\frac{\partial B^e_{k,p}(\hat{\xi})}{\partial\hat{\xi}}\big)\big(B^e_{l,q}(\hat{\eta})\frac{\partial\hat{\xi}}{\partial y}\big)+\big(c_yB^e_{k,p}(\hat{\xi})\big)\big(\frac{\partial B^e_{l,q}(\hat{\eta})}{\partial\hat{\eta}}\frac{\partial\hat{\eta}}{\partial y}\big)\Big)\Big]\mid\hat{J}^e(\hat{\xi},\hat{\eta})\mid\mid J^e(\xi,\eta)\mid d\widehat{\mathbb{D}}.\end{aligned}$$

The computation is achieved using Gauss-Legendre quadrature rules. To complete the scheme (8.5), we choose the Lax–Friedrichs flux.

8.3 Numerical Lax–Friedrichs fluxes

In the IGDG method, continuity is not enforced between elements. The flux $f_n(u^e) = f_n^e = \vec{c}\, u^e \cdot \vec{n}^e$ along the boundaries Γ^e must be approximated by a numerical flux $\widehat{f}_n^e$. In the present case for the Bézier elements, we use the local Lax- Friedrichs flux which can be defined by:

$$\widehat{f_n^e} = \frac{1}{2}\big(f_n(u_l^e) + f_n(u_r^e)\big) + \frac{|\, c_n \,|}{2}\big(u_l^e - u_r^e\big) \qquad \text{with} \quad c_n = \vec{c}\cdot\vec{n}^e \tag{8.6}$$

$$= \frac{1}{2}\big(\vec{c}\, u_l^e \cdot \vec{n}^e + \vec{c}\, u_r^e \cdot \vec{n}^e\big) + \frac{|\, \vec{c}\cdot\vec{n}^e \,|}{2}\big(u_l^e - u_r^e\big) \tag{8.7}$$

Due to the fact that the weak form of the DG method is written elementwise, the numerical flux between adjacent elements must be defined. For this purpose, it is possible to write for each element $\mathbb{D}^e$:

$$\begin{aligned}
\widehat{F_e} &= \sum_{k=1}^{4} \int_{\Gamma_k^e} \widehat{f_n^e} \mathfrak{B}_{k,l}^e(x,y)\, d\Gamma_k^e \\
&= \int_{[0,1]} \widehat{f_n^e}_{|_{\Gamma_1^e}} \big(B_{k,p}(\xi)\big)^e \big(\underbrace{B_{l,p}(0)}_{=1}\big)^e \,|\, J^e(\xi,0) \,||\, \widehat{J}^e(\widehat{\xi},0) \,|\, d\xi \\
&+ \int_{[0,1]} \widehat{f_n^e}_{|_{\Gamma_2^e}} \big(\underbrace{B_{k,p}(1)}_{=1}\big)^e \big(B_{l,p}(\eta)\big)^e \,|\, J^e(1,\eta) \,||\, \widehat{J}^e(1,\widehat{\eta}) \,|\, d\eta \\
&+ \int_{[0,1]} \widehat{f_n^e}_{|_{\Gamma_3^e}} \big(B_{k,p}(\xi)\big)^e \big(\underbrace{B_{l,p}(1)}_{=1}\big)^e \,|\, J^e(\xi,1) \,||\, \widehat{J}^e(\widehat{\xi},1) \,|\, d\xi \\
&+ \int_{[0,1]} \widehat{f_n^e}_{|_{\Gamma_4^e}} \big(\underbrace{B_{k,p}(0)}_{=1}\big)^e \big(B_{l,p}(\eta)\big)^e \,|\, J^e(0,\eta) \,||\, \widehat{J}^e(0,\widehat{\eta}) \,|\, d\eta
\end{aligned}$$

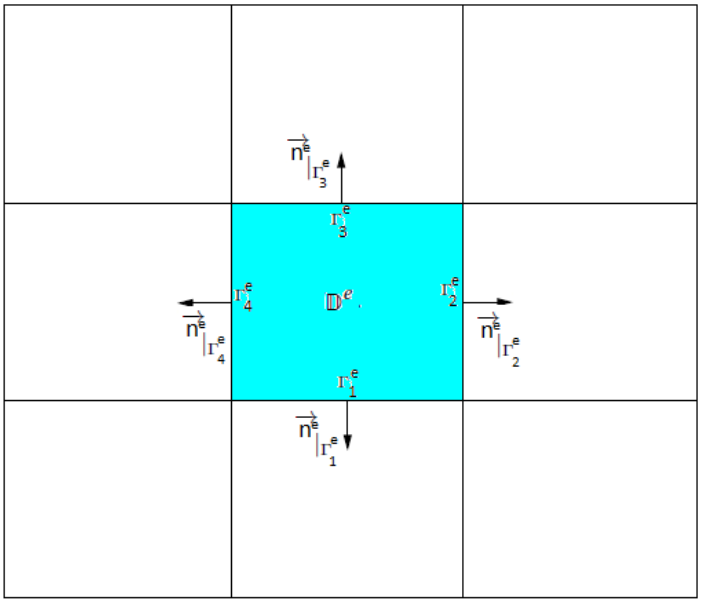


Figure 8.2: Element $\mathbb{D}^e$, its faces $\big(\Gamma_k^e\big)_{k=1,\dots,4}$ and the corresponding normals $\vec{n}^e_{|_{\Gamma_k^e}}$.

It is also necessary to define for the patch $\mathbb{D}^e$, the normal vectors on the four interfaces Γ_1^e, Γ_2^e, Γ_3^e and Γ_4^e given by:

$$\overrightarrow{n}^e_{|_{\Gamma_1^e}} = \begin{pmatrix} n^e_{x|_{\Gamma_1^e}} \\ n^e_{y|_{\Gamma_1^e}} \end{pmatrix} = \begin{pmatrix} \frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \xi} \end{pmatrix}^e,$$

$$\overrightarrow{n^e}_{|_{\Gamma_2^e}} = \begin{pmatrix} n^e_{x|_{\Gamma_2^e}} \\ n^e_{y|_{\Gamma_2^e}} \end{pmatrix} = \begin{pmatrix} \frac{\partial y}{\partial \eta} \\ -\frac{\partial x}{\partial \eta} \end{pmatrix}^e,$$

$$\overrightarrow{n^e}_{|_{\Gamma_3^e}} = \begin{pmatrix} n^e_{x|_{\Gamma_3^e}} \\ n^e_{y|_{\Gamma_3^e}} \end{pmatrix} = \begin{pmatrix} -\frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \xi} \end{pmatrix}^e,$$

$$\overrightarrow{n^e}_{|_{\Gamma_4^e}} = \begin{pmatrix} n^e_{x|_{\Gamma_4^e}} \\ n^e_{y|_{\Gamma_4^e}} \end{pmatrix} = \begin{pmatrix} -\frac{\partial y}{\partial \eta} \\ \frac{\partial x}{\partial \eta} \end{pmatrix}^e.$$

Obviously, these vectors are normalized for the computations.

8.4 The RK time discretization

The space semidiscrete problem (8.5) represents a system of ordinary differential equations (ODEs), which has to be solved with the Runge Kutta schemes of order 2 and 4. As mentioned before, for the $1D$ case the time step Δt is strongly restricted by the Courant-Friedrichs-Lewy (CFL) stability condition. In the context of a general multi-dimensional hyperbolic system, the combination of RK of order $p+1$ with DG at order p, the CFL condition can be written as:

$$\Delta t \leq C_{CFL} \min \frac{\mid d_e \mid}{\parallel \overrightarrow{c}\,(2p+1) \parallel}, \tag{8.8}$$

where $\mid d_e \mid$ is a typical length scale for the Bézier element $\mathbb{D}^e$. Obviously, $0 < C_{CFL} \leq 1$ is the Courant-Friedrichs-Lewy (CFL) coefficient and $\parallel \overrightarrow{c} \parallel = \sqrt{c_x^2 + c_y^2}$.
In what follows, we consider a partition $0 = t^0 < t^1 < ... < t^{\check{n}} = T$ of the time interval $[0, T]$ and set $\Delta t = t^{k+1} - t^k$ for $k = 1, ..., \check{n} - 1$.

8.5 Numerical results

In order to demonstrate the performance of the present method, we consider an example of $2D$ advection problem given above by Eq. (8.1) whose initial solution is given by:

$$u_0(x, y) = \exp\big(-5(x^2 + y^2)\big). \tag{8.9}$$

The analytical solution to this problem is:

$$u_{\text{analytic}}(x, y, t) = \exp\big(-5(x - c_x t)^2 - 5(y - c_y t)^2)\big).$$

We present in Fig. 8.3 the exact solution of the problem (8.1) for $\overrightarrow{c} = (c_x, c_y) = (1, 1)$.

The results will be presented for 3 different types of patches: cartesian patches, linear patches and curvilin-

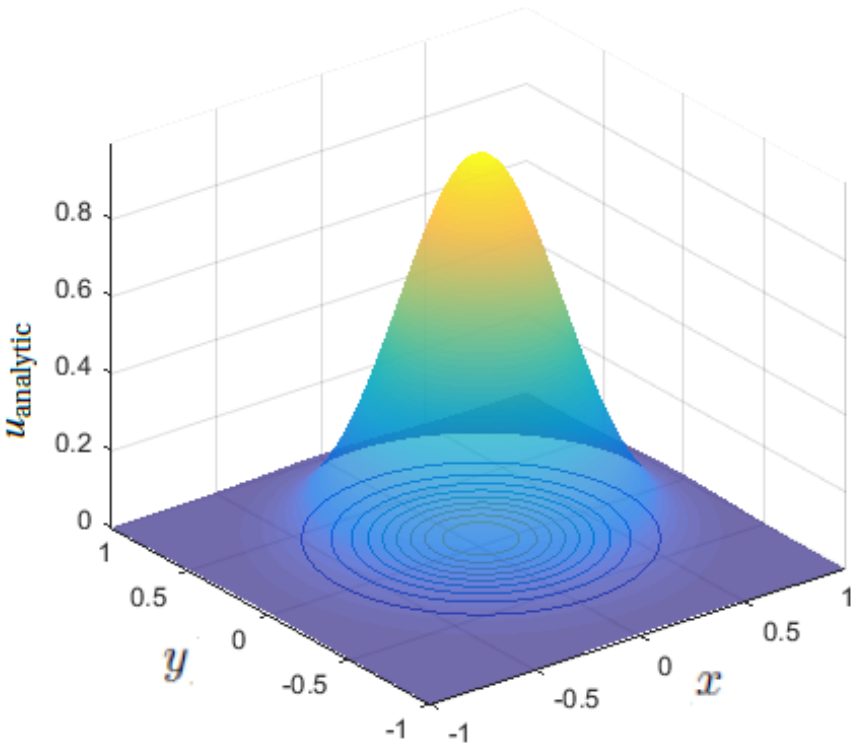


Figure 8.3: Analytical solution for the bi-dimensional advection problem, for $\mathfrak{N}_{el} = 4 \times 4$ at $T = 0.5s$.

ear patches. Thus, we use the RK time stepping method, with respect the CFL condition given by Eq. (8.8). The physical domain $\Omega = [-1, 1] \times [-1, 1]$ is a initial Bézier patch with $\mathfrak{N}_{el} = \mathfrak{N}^1_{el} \times \mathfrak{N}^2_{el} = 4 \times 4$ patches $\mathbb{D}^e$ which is plotted with the corresponding numerical solution for each choice of patches.
In addition, we consider uniform mesh sizes $\mathfrak{h}_x = \frac{2}{\mathfrak{N}^1_{el}}$ and $\mathfrak{h}_y = \frac{2}{\mathfrak{N}^2_{el}}$. The bivariate Bernstein functions are taken to be of bi-degree (p, q). We focus for the case $p = q$ which will be specified in each example.

8.5.1 Cartesian grids

The first mesh type considered are simple cartesian grids. Figure 8.4 depicts the IGDG numerical solutions for the bivariate quadratic Bernstein case for the initial mesh with $\mathfrak{N}_{el} = 4 \times 4$ patches which is plotted in the top of Fig. 8.4. We subsequently add a reffinement of $\frac{h}{2}$ as shown in the bottom of Fig. 8.4.

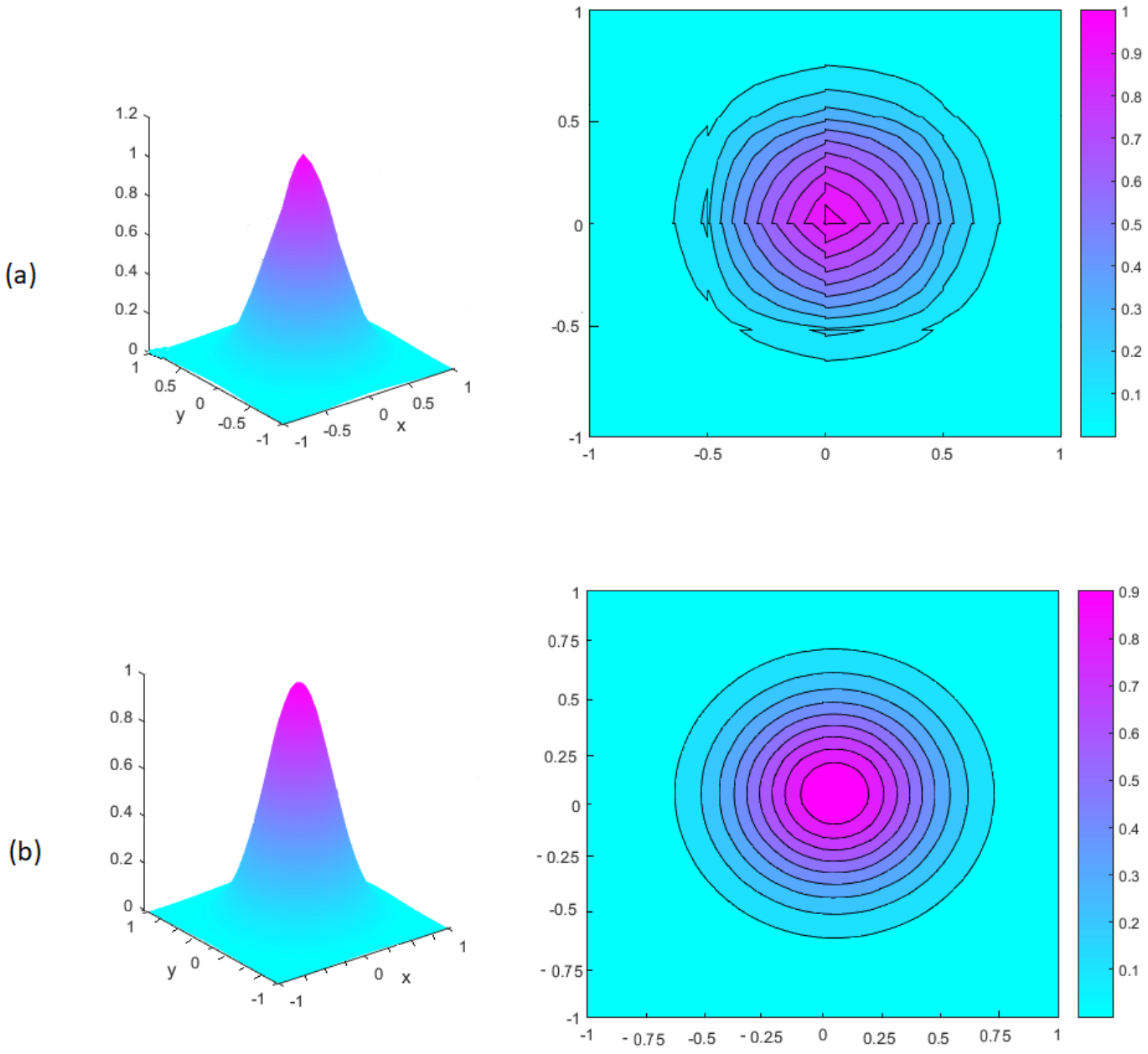


Figure 8.4: Plots and contour plots of numerical results for bivariate quadratic Bernstein basis with (a) $\mathfrak{N}_{el} = 4 \times 4$ patches and (b) $\mathfrak{N}_{el} = 8 \times 8$ patches at $T = 0.05s$.

Figure 8.5 shows the numerical solutions from IGDG space discretization and explicit $RK4$ time integration for the linear, quadratic, cubic and quartic Bernstein cases. We can see the effect of the degree elevation of the bivariate Bernstein basis function on the accuracy.

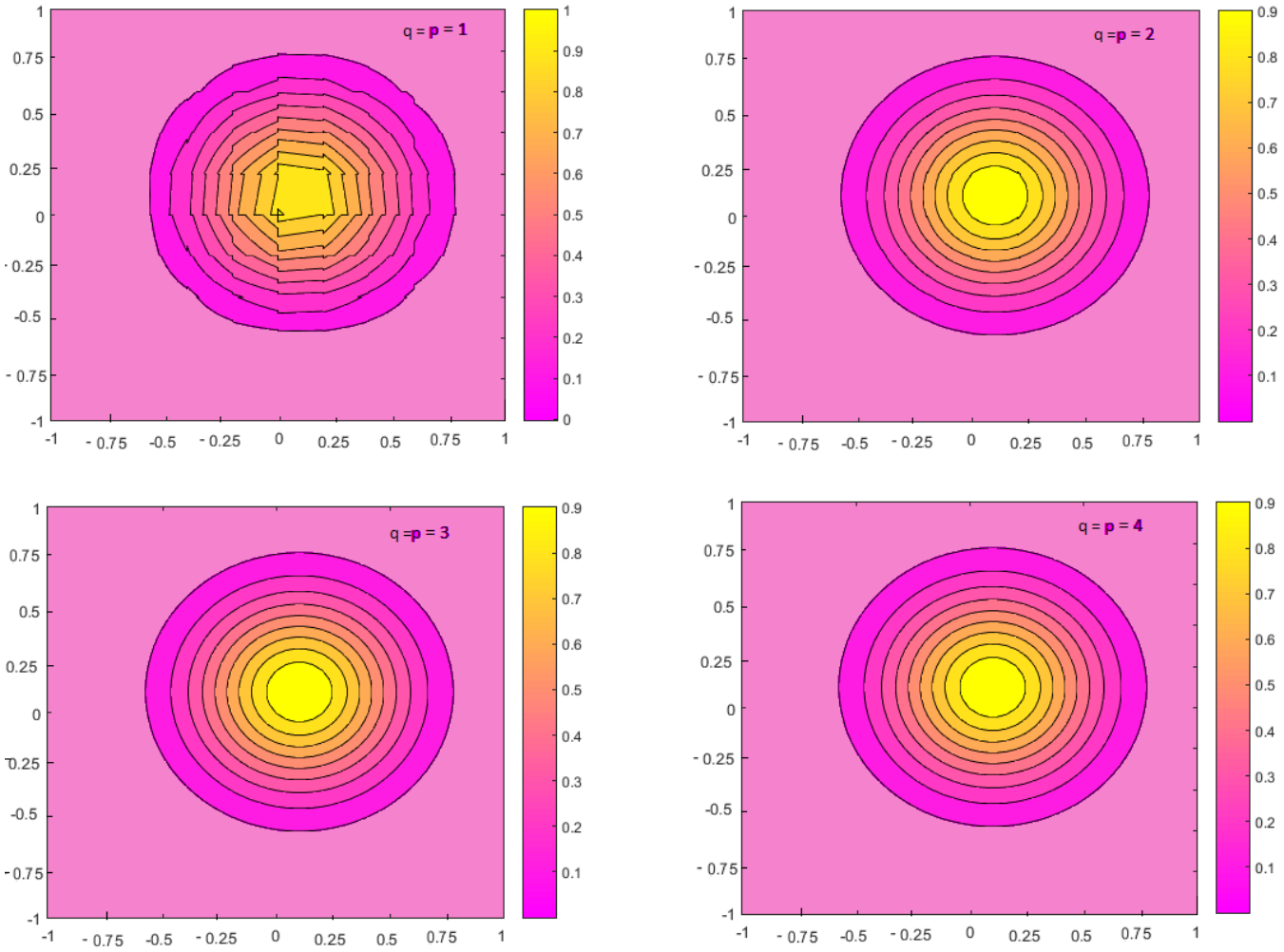


Figure 8.5: IGDG solution for $\mathfrak{N}_{el} = 8 \times 8$ patches for different degrees (p, q).

Before we proceed further, let us recall that we denote by $\| u \|_{L^2(\Omega)}$, the $L^2(\Omega)$-norm of a function u:

$$\| u \|_{L^2(\Omega)} = \Big(\int_\Omega u^2 d\Omega \Big)^{\frac{1}{2}}.$$

We measure the convergence of the numerical methods in the L^2-norm for the Cartesian, linear and curvilinear grids. Our aim is to identify the numerical order of convergence. We define the error of the numerical solution as:

$$e_h = u_h - u_{ex}, \tag{8.10}$$

and its L^2-norm is:

$$\| e_h \|_{L^2(\Omega)} = \Big(\int_\Omega \big(u_h(x,y) - u_{ex}(x,y) \big)^2 d\Omega \Big)^{1/2}. \tag{8.11}$$

Tab. 8.1 and Tab. 8.2 summarize the convergence results in the L^2-norm for the bivariate linear,quadratic, cubic and quartic Bernstein basis functions, from which we can see that the L^2-convergence rate is approximately $p+1$.

Mesh	L^2-error	rate
$\mathfrak{h}$	$2.235E-01$	–
$\frac{\mathfrak{h}}{2}$	$1.012E-01$	1.14
$\frac{\mathfrak{h}}{4}$	$2.586E-02$	1.96
$\frac{\mathfrak{h}}{8}$	$5.807E-03$	2.15

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.136E-01$	–
$\frac{\mathfrak{h}}{2}$	$1.476E-02$	2.94
$\frac{\mathfrak{h}}{4}$	$1.713E-03$	3.10
$\frac{\mathfrak{h}}{8}$	$2.047E-04$	3.06

Table 8.1: L^2-error for the $2D$ advection problem and convergence order for the IGDG method for the linear (left) and quadratic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

Mesh	L^2-error	rate
$\mathfrak{h}$	$3.170E-02$	–
$\frac{\mathfrak{h}}{2}$	$5.090E-03$	2.63
$\frac{\mathfrak{h}}{4}$	$2.6488E-04$	4.26
$\frac{\mathfrak{h}}{8}$	$1.650E-05$	4.00

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.706E-02$	–
$\frac{\mathfrak{h}}{2}$	$3.639E-04$	5.55
$\frac{\mathfrak{h}}{4}$	$9.763E-06$	5.22
$\frac{\mathfrak{h}}{8}$	$2.851E-07$	5.09

Table 8.2: L^2-error for the $2D$ advection problem and convergence order for the IGDG method for the cubic (left) and quartic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

The corresponding convergence data for uniform refinement $\frac{\mathfrak{h}}{2}$, $\frac{\mathfrak{h}}{4}$ and $\frac{\mathfrak{h}}{8}$ are shown in Fig. 8.6. The convergence rates of the L^2-norm of the error are shown in the legend.

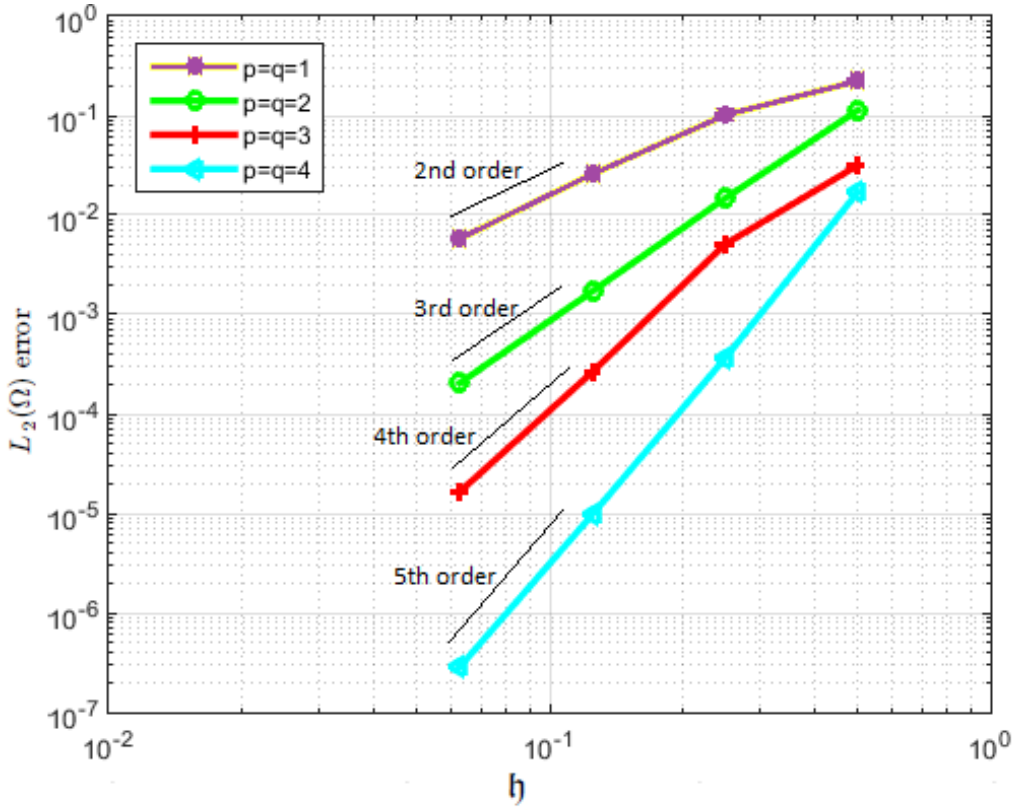


Figure 8.6: L^2-error for the $2D$ advection problem using the IGDG method in conjunction with $RK4$.

We consider $p = q \in \{1,2,3,4\}$ and asses the quality of numerical approximations through the L^2-norm. Table 8.1 and table 8.2 show the error of numerical approximations in L^2-norm for various size of element $\mathfrak{h}$ and order of basis p. Those results are depicted in Figure 8.6. We obtain the convergence rates $r = p+1$, $p \in \{1,2,3,4\}$.

8.5.2 Linear grids

Linear patches are still employed in this case. The initial configurations of $\mathfrak{N}_{el} = \mathfrak{N}^1_{el} \times \mathfrak{N}^2_{el}$ patches and degrees (p, q) of bivariate Bernstein function are the same with those described in the previous case.

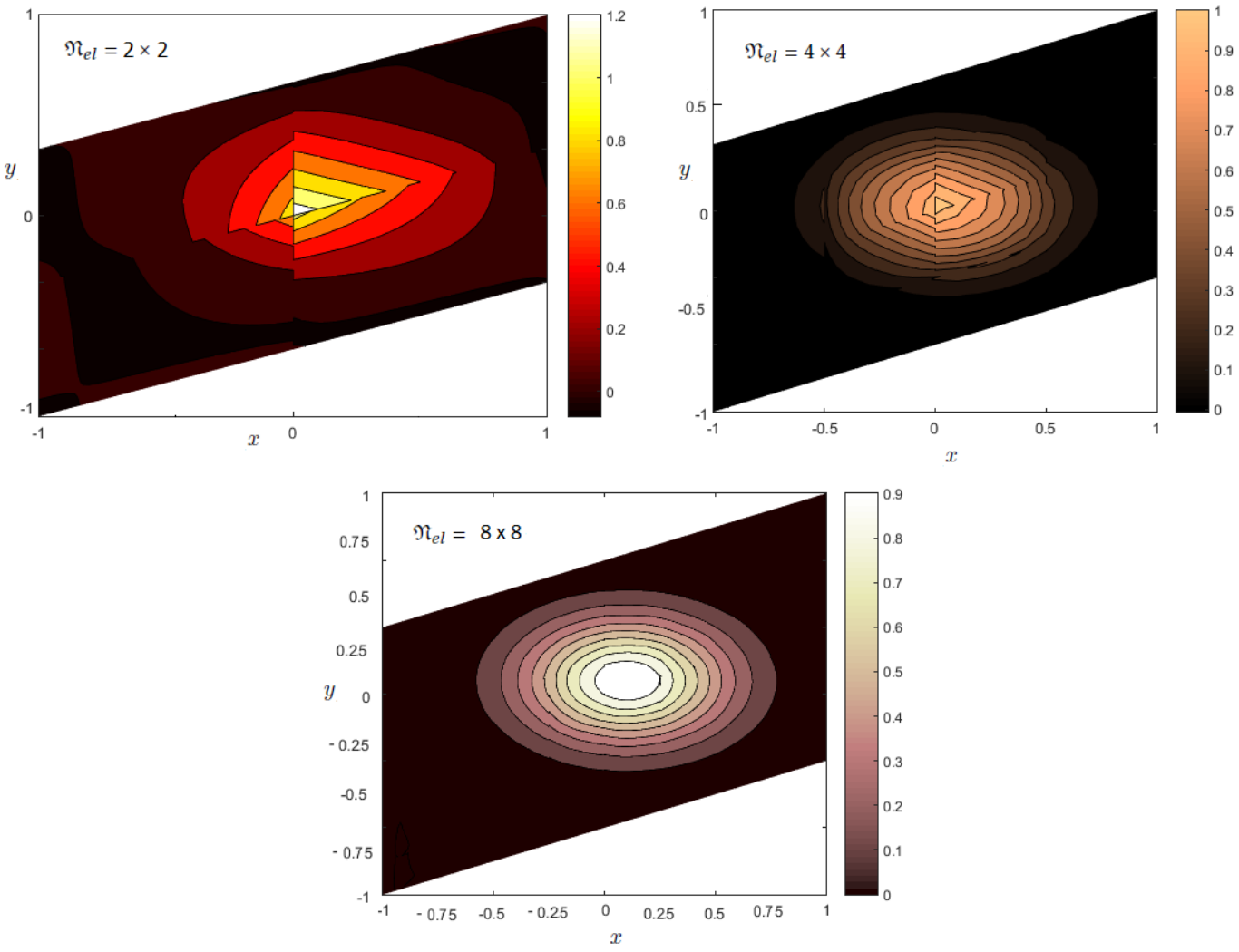


Figure 8.7: Contour plots of numerical results for bivariate quadratic Bernstein basis at $T = 0.05s$.

We refer to Figure 8.8 for comparisons of bases functions for the bivariate linear, quadratic, cubic and quartic Bernstein functions, our mesh is a Bézier patch with $\mathfrak{N}_{el} = 8 \times 8$ elements.

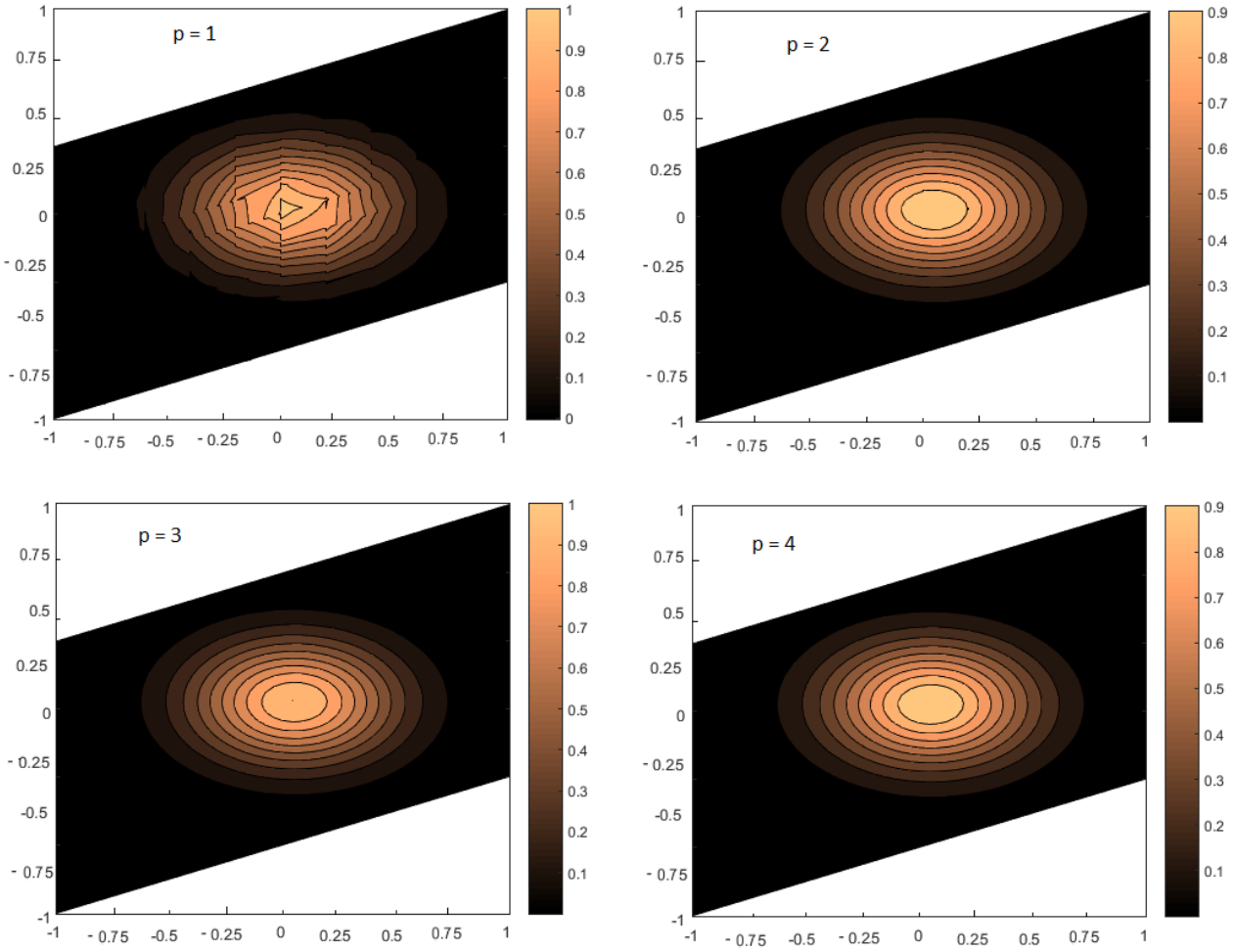


Figure 8.8: IGDG solutions for different degrees for $\mathfrak{N}_{el} = 8 \times 8$ uniform elements.

Convergence results for the $2D$ advection problem with different choices of $\mathfrak{N}_{el}$ are given in Tab. (8.3) for the linear and quadratic Bernstein and in Tab. (8.4) for the cubic and quartic Bernstein.

Mesh	L^2-error	rate
$\mathfrak{h}$	$2.248E-01$	–
$\frac{\mathfrak{h}}{2}$	$1.043E-01$	1.10
$\frac{\mathfrak{h}}{4}$	$2.931E-02$	1.83
$\frac{\mathfrak{h}}{8}$	$6.818E-03$	2.10

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.222E-01$	–
$\frac{\mathfrak{h}}{2}$	$2.220E-02$	2.46
$\frac{\mathfrak{h}}{4}$	$3.410E-03$	2.70
$\frac{\mathfrak{h}}{8}$	$3.874E-04$	3.13

Table 8.3: L^2-error for the $2D$ advection problem and convergence order for the IGDG method for the linear (left) and quadratic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

Mesh	L^2-error	rate
$\mathfrak{h}$	$4.397E-02$	–
$\frac{\mathfrak{h}}{2}$	$5.362E-03$	3.03
$\frac{\mathfrak{h}}{4}$	$3.473E-04$	3.94
$\frac{\mathfrak{h}}{8}$	$1.915E-05$	4.18

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.845E-02$	–
$\frac{\mathfrak{h}}{2}$	$8.373E-04$	4.46
$\frac{\mathfrak{h}}{4}$	$2.871E-05$	4.86
$\frac{\mathfrak{h}}{8}$	$7.553E-07$	5.19

Table 8.4: L^2-error for the $2D$ advection problem and convergence order for the IGDG method for the cubic (left) and quartic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

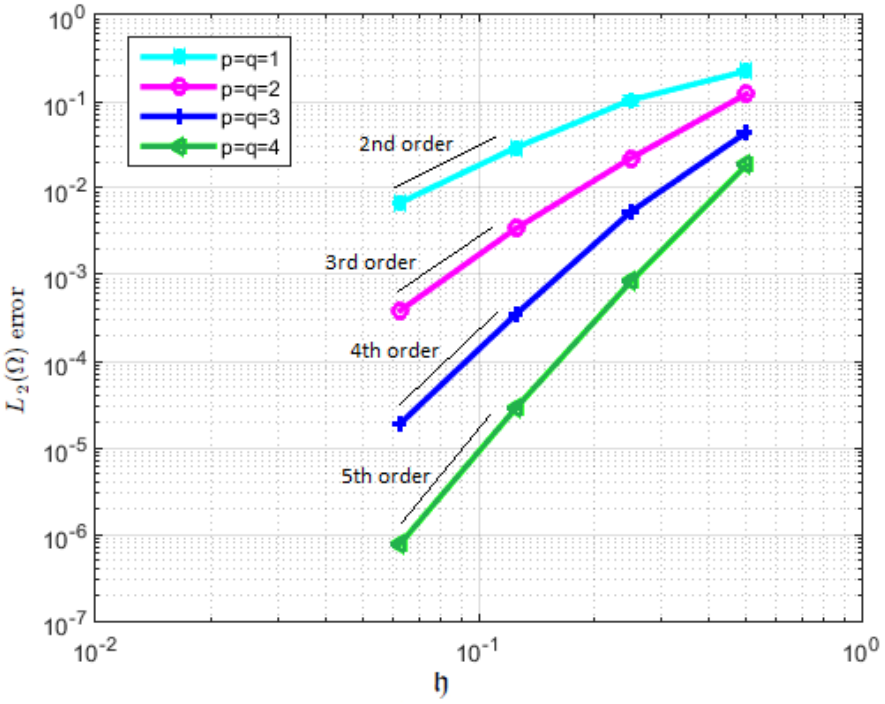


Figure 8.9: L^2-error for the $2D$ advection problem using the IGDG method in conjunction with $RK4$.

8.5.3 Curvilinear grids

In this case, we consider $\mathfrak{N}_{el}$ curvilinear patches. Contrary to the previous cases, the geometry curvature is taken into account here. We underline that the boundary geometry is identically maintained whatever the basis degree (except for $p = 1$) and d.o.f number.

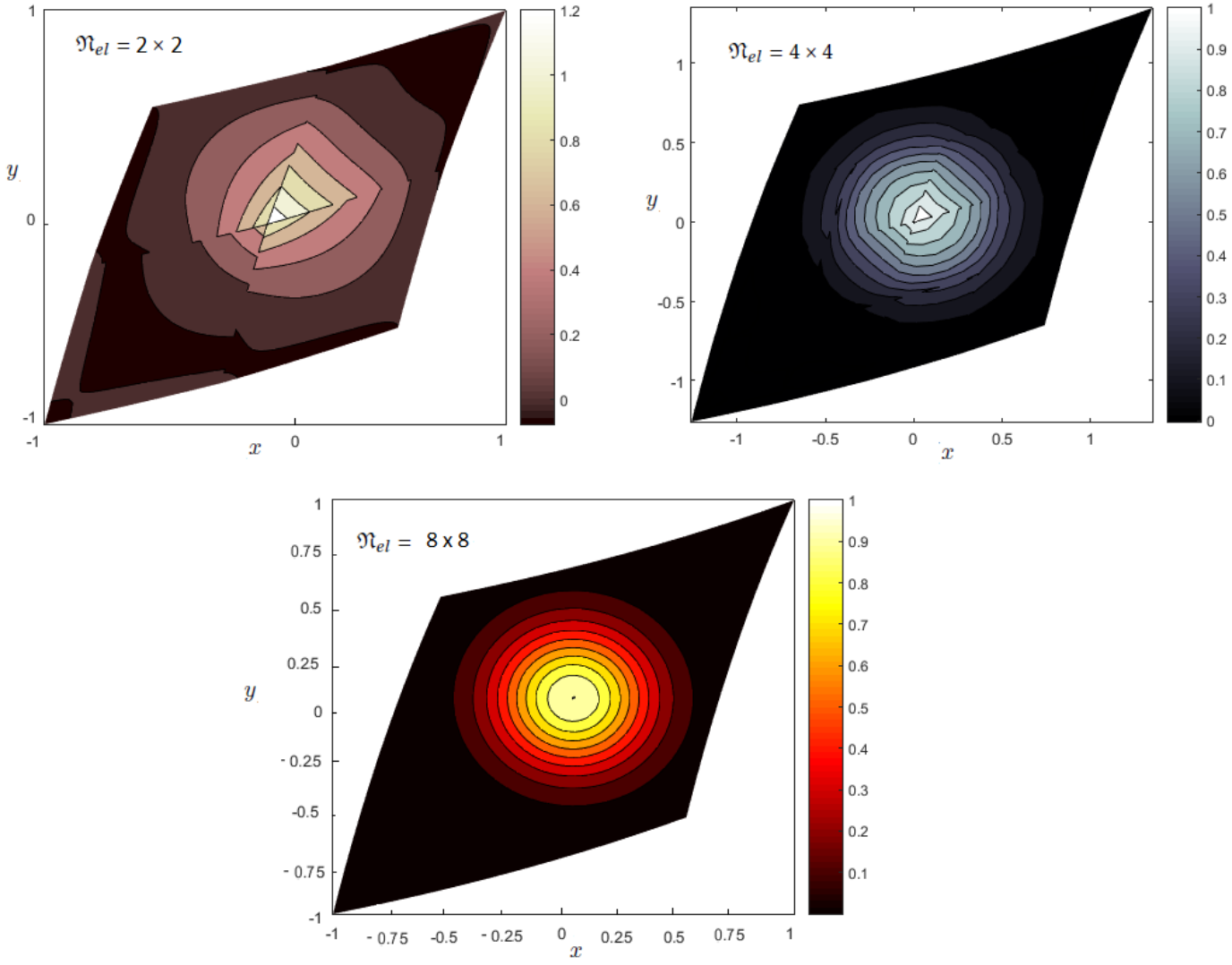


Figure 8.10: Contour plots of numerical results for bivariate quadratic Bernstein basis at $T = 0.05s$.

Figure 8.11 shows the numerical solutions from IGDG space discretization and explicit $RK4$ time integration for the linear, quadratic, cubic and quartic Bernstein cases. We can see the effect of the degree elevation of the bivariate Bernstein basis function on the accuracy.

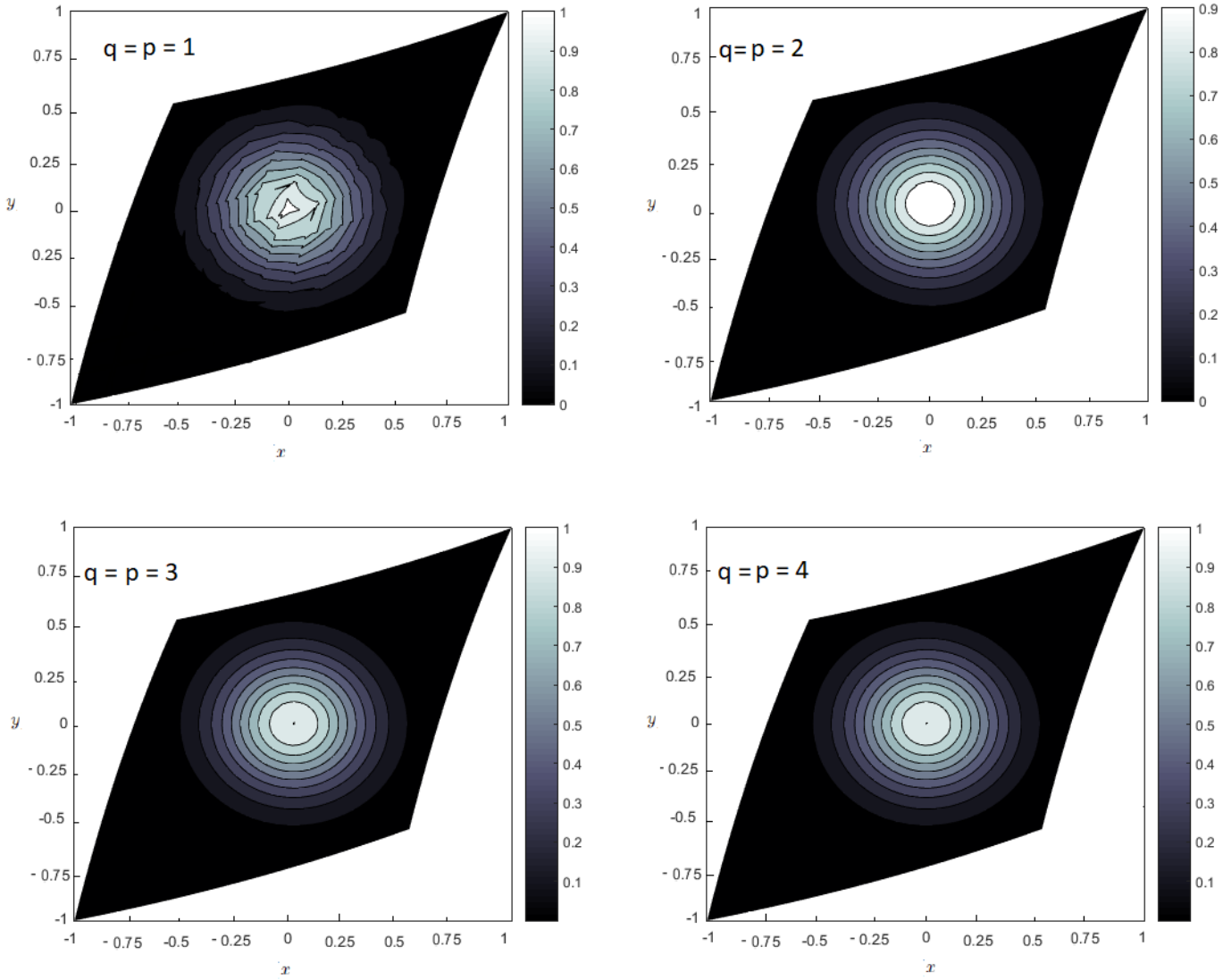


Figure 8.11: IGDG solutions for different bivariate degrees (p, q) for $\mathfrak{N}_{el} = 8 \times 8$.

In the following we investigate the numerical order of convergence in the the L^2-norm of the IGDG discretizations, the problem is solved for several different values of bivariate Bernstein polynomial degrees (p, p) and numbers of patches $\mathfrak{N}_{el}$. The L^2-norm of the error and convergence rate are shown in Table 8.5 for the linear and quadratic case and in Table 8.6 for the cubic and quartic Bernstein.

Mesh	L^2-error	rate
$\mathfrak{h}$	$2.176E-01$	–
$\frac{\mathfrak{h}}{2}$	$9.935E-02$	1.13
$\frac{\mathfrak{h}}{4}$	$2.792E-02$	1.83
$\frac{\mathfrak{h}}{8}$	$6.295E-03$	2.14

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.139E-01$	–
$\frac{\mathfrak{h}}{2}$	$2.206E-02$	2.36
$\frac{\mathfrak{h}}{4}$	$3.113E-03$	2.82
$\frac{\mathfrak{h}}{8}$	$3.657E-04$	3.08

Table 8.5: L^2-error for the $2D$ advection problem and convergence order for the IGDG method for the linear (left) and quadratic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

Mesh	L^2-error	rate
$\mathfrak{h}$	$4.243E-02$	–
$\frac{\mathfrak{h}}{2}$	$4.686E-03$	3.17
$\frac{\mathfrak{h}}{4}$	$3.0617E-04$	3.93
$\frac{\mathfrak{h}}{8}$	$1.705E-05$	4.16

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.611E-02$	–
$\frac{\mathfrak{h}}{2}$	$8.272E-04$	4.28
$\frac{\mathfrak{h}}{4}$	$2.382E-05$	5.11
$\frac{\mathfrak{h}}{8}$	$6.609E-07$	5.17

Table 8.6: L^2-error for the $2D$ advection problem vs. mesh parameter and convergence order for the IGDG method for the cubic (left) and quartic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

Again, as can be seen, we obtain that the L^2 convergence rate is approximately $p+1$.

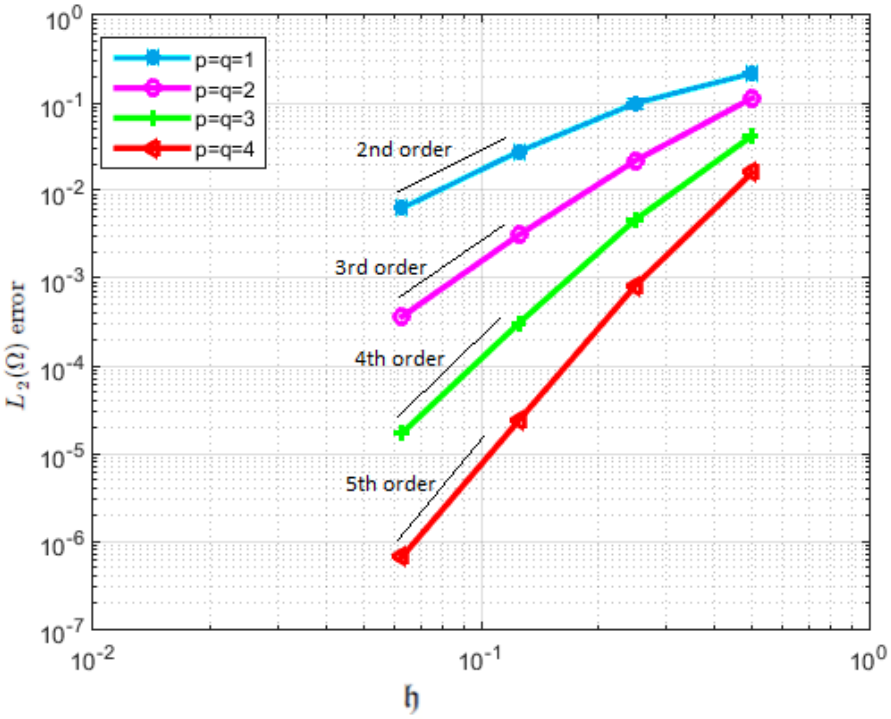


Figure 8.12: L^2-errors for the $2D$ advection problem using the IGDG method in conjunction with $RK4$.

Table 8.5 and table 8.6 show the quality of the numerical approximations in L^2 norm for various size of element ɦ and order of polynomial (p and q). The error of the numerical approximations is depicted in Figure 8.12. The convergence rates are 2, 3, 4 and 5 using bases of degrees $p = q = 1$, $p = q = 2$, $p = q = 3$ and $p = q = 4$, respectively.

8.6 Conclusion

As it has been pointed out previously, we can conclude that IGDG is a powerful tool. In this chapter, we have introduced a new analysis framework, called isogeometric discontinuous Galerkin (IGDG) method for bi-dimensional hyperbolic problem, which is based on Bézier extraction. We have confined our attention to the development of the RK-IGDG methods for bi-dimensional advection problem. The resulting IGDG method in conjunction with RK method is stable, high-order accurate, and highly parallelizable. The scheme can easily handle complicated geometries and boundary conditions. The flexibility of the method to handle different geometries and to work with different elements has been shown.
As consequence, this method can be easily formulated and implemented. Thus, the numerical behavior of the method is evaluated and it has shown an optimal convergence rate.

CHAPTER 9

2D ACOUSTIC WAVE EQUATIONS

The purpose of this chapter is to develope and analyze the new IGDG method, that uses the IGA discretization concept combined with the DG technique, for solving the first-order acoustic wave equation in $2D$, modelling sound propagation phenomena. The computational domain is divided into non-overlapping sub-domains, composed of B-spline patches. The DG approach was applied on element level, each element being a Bézier patch constructed from initial B-spline patches. The solution of the problem is approximated in every element without imposing any continuity requirements for the discrete solution on the interfaces. Basic tests of accuracy and stability are demonstrated, including optimal convergence rates with respect to L^2-norm .

9.1 Introduction and basic theory

Acoustic equations model acoustic wave propagation in a medium. Several application fields are covered by such a model, like elastic wave propagation in the ground, or sound propagation in the air [76]. The form itself is usually considered as a linear problem governed by the compressible linearized Euler equations, in order to describe mean flow effects on sound propagation, such as refraction.

The pressure-velocity formulation of the acoustic wave equations is expressed as a linear PDE system, in which the acoustic pressure and velocity interact with one another to propagate waves through materials. Let $\Omega \subset \mathbb{R}^2$ be a two dimensional domain with boundary $\partial\Omega$. On each point on $\partial\Omega$ we denote by $\vec{n}$ the outward normal vector. Let $T > 0$ be a fixed time.

For $(x, y, t) \in \Omega \times \big(0, T\big)$, the linear acoustic wave equation can be formulated as a first–order hyperbolic system in terms of the pressure field $\mathbf{p}$ and velocity field $\overrightarrow{\mathscr{U}} = \big(\mathbf{u}, \mathbf{v}\big)$:

$$\begin{cases} \frac{\partial \overrightarrow{\mathscr{U}}}{\partial t} + \nabla \mathbf{p} & = 0 \qquad (x, y, t) \in \Omega \times [0, T], \\ \frac{\partial \mathbf{p}}{\partial t} + \nabla \cdot \overrightarrow{\mathscr{U}} & = 0 \qquad (x, y, t) \in \Omega \times [0, T], \\ \overrightarrow{\mathscr{U}} \cdot \overrightarrow{n} & = 0 \qquad \text{on} \quad \partial\Omega \times [0, T]. \end{cases} \tag{9.1}$$

Here, $\mathbf{p}$ represent the pressure perturbation and $\overrightarrow{\mathscr{U}}$ the velocity perturbation with respect to a reference state at rest. Moreover, the equations have been adimensionalized by assuming that the sound speed is equal to unity.

At boundary, we assume perfect wall conditions, all waves being ideally reflected.

We supplement the system (9.1) with the initial condition:

$$\begin{cases} \mathbf{u}(x, y, 0) & = \mathbf{u}_0(x, y) \qquad (x, y) \in \Omega, \\ \mathbf{v}(x, y, 0) & = \mathbf{v}_0(x, y) \qquad (x, y) \in \Omega, \\ \mathbf{p}(x, y, 0) & = \mathbf{p}_0(x, y) \qquad (x, y) \in \Omega. \end{cases} \tag{9.2}$$

An equivalent formulation of the system (9.1) is the following:

$$\begin{cases} \frac{\partial \mathbf{u}}{\partial t} + \frac{\partial \mathbf{p}}{\partial x} & = 0 \qquad (x, y, t) \in \Omega \times [0, T], \qquad (1) \\ \frac{\partial \mathbf{v}}{\partial t} + \frac{\partial \mathbf{p}}{\partial y} & = 0 \qquad (x, y, t) \in \Omega \times [0, T], \qquad (2) \\ \frac{\partial \mathbf{p}}{\partial t} + \frac{\partial \mathbf{u}}{\partial x} + \frac{\partial \mathbf{v}}{\partial y} & = 0 \qquad (x, y, t) \in \Omega \times [0, T], \qquad (3) \\ \overrightarrow{\mathscr{U}} \cdot \overrightarrow{n} & = 0 \qquad \text{on} \quad \partial\Omega \times [0, T]. \end{cases} \tag{9.3}$$

9.2 IGDG approximation of the acoustic wave equations

9.2.1 Spatial discretization

This section introduces the weak formulation of the linear acoustic wave equations given above by equation (9.3) using IGDG method. As highlighted in the previous chapter, in DG method the basis functions are allowed to be discontinuous at the element boundaries, the integrals are performed element-wise, and the compling of the wave field across the elements is weakly imposed using fluxes. The first step is to derive the weak formulations of equation (9.3). This weak formulation is then discretized by introducing an approximation for the pressure and velocity in a finite-dimensional subspace to obtain a linear system of ordinary differential equations.

In order to apply the IGA methodology, we highlight that the physical domain Ω is subdivided into B-spline patches $\mathfrak{D}^e$,

$$\mathscr{S}(\Omega) := \left\{\mathfrak{D}^e\right\}_{e=1}^{N_{pa}}.$$

We suppose that the computational domain Ω can be exactly represented by the union of non-overlapping patches $\mathfrak{D}^e$:

$$\overline{\Omega} = \bigcup \overline{\mathfrak{D}}^e \quad \text{with} \quad \overset{\circ}{\mathfrak{D}}{}^e \bigcap \overset{\circ}{\mathfrak{D}}{}^l = \emptyset \quad \forall \quad e \neq l.$$

Each patch $\mathfrak{D}^e$ defined by:

$$\mathfrak{D}^e = \left\{ X^e = (x^e, y^e) \in \mathbb{R}^2 \quad | \, X^e = \mathbb{T}(\xi,\eta) \quad \text{such} \quad \text{that} \quad (\xi,\eta) \in \widetilde{\mathfrak{D}}^e \right\},$$

where the transformation $\mathbb{T}$ is defined for all $(\xi,\eta) \in \widetilde{\mathfrak{D}}^e$ by:

$$\mathbb{T} : \widetilde{\mathfrak{D}}^e \quad \longmapsto \quad \mathfrak{D}^e$$

$$(\xi,\eta) \quad \longmapsto \quad \left(x^e(\xi,\eta), y^e(\xi,\eta)\right).$$

As in the previous chapter, a set of Bézier elements is obtained from the B-spline patches by multiple knot insertion. Within each element $\mathbb{D}^e$ we assume that the local solution is well approximated by two-dimensional Bernstein polynomials of degrees p and q:

$$\begin{aligned}
\begin{pmatrix} \mathbf{u}_h(x,y,t) \\ \mathbf{v}_h(x,y,t) \\ \mathbf{p}_h(x,y,t) \end{pmatrix}^e &= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \left(\Phi_{i,j}^{p,q}(x,y)\right)^e \begin{pmatrix} \mathbf{u}_{ij}(t) \\ \mathbf{v}_{ij}(t) \\ \mathbf{p}_{ij}(t) \end{pmatrix}^e \\
&= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \left(\mathfrak{B}_{i,j}^{p,q}(x,y)\right)^e \begin{pmatrix} \mathbf{u}_{ij}(t) \\ \mathbf{v}_{ij}(t) \\ \mathbf{p}_{ij}(t) \end{pmatrix}^e \\
&= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \left(\mathfrak{B}_{i,j}^{p,q}(\mathbb{T}(\xi,\eta))\right)^e \begin{pmatrix} \mathbf{u}_{ij}(t) \\ \mathbf{v}_{ij}(t) \\ \mathbf{p}_{ij}(t) \end{pmatrix}^e \\
&= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \left(B_{i,p}(\xi)\right)^e \left(B_{j,q}(\eta)\right)^e \begin{pmatrix} \mathbf{u}_{ij}(t) \\ \mathbf{v}_{ij}(t) \\ \mathbf{p}_{ij}(t) \end{pmatrix}^e
\end{aligned}$$

where $\mathbf{u}_{ij}^e : [0,T] \longmapsto \mathbb{R}$, $\mathbf{v}_{ij}^e : [0,T] \longmapsto \mathbb{R}$ and $\mathbf{p}_{ij}^e : [0,T] \longmapsto \mathbb{R}$, $\forall 1 \leq i \leq p+1$, $\forall 1 \leq j \leq q+1$ are local unknown coefficients.

The number of degree of freedom inside the element $\mathbb{D}^e$ is $(p+1)\times(q+1)$. $\left(\Phi_{i,j}^{p,q}\right)^e$ are the bivariate Bernstein polynomials of degree $p\times q$. The global solution can then be approximated by the direct sum of the local solutions:

$$\begin{pmatrix} \mathbf{u}(x,y,t) \\ \mathbf{v}(x,y,t) \\ \mathbf{p}(x,y,t) \end{pmatrix} \simeq \begin{pmatrix} \mathbf{u}_h(x,y,t) \\ \mathbf{v}_h(x,y,t) \\ \mathbf{p}_h(x,y,t) \end{pmatrix} = \bigoplus_{e=1}^{\mathfrak{N}_{el}} \begin{pmatrix} \mathbf{u}_h(x,y,t) \\ \mathbf{v}_h(x,y,t) \\ \mathbf{p}_h(x,y,t) \end{pmatrix}^e .$$

Before we proceed further, let us put $w_h^e \in \mathbb{V}_h^p$, the approximate solution will be sought for each $t \in [0,T]$ in the finite-dimensional space:

$$\mathbb{V}_h^p = \left\{ w := w\big(x(\xi,\eta), y(\xi,\eta)\big) \in L^2(\Omega), \quad w_{|_{\mathbb{D}^e}} \in span\{B^e\} \right\},$$

where B^e is the set of $(p+1)\times(q+1)$ Bernstein polynomials defined over $\widetilde{\mathbb{D}}^e$.
The starting point for a DG discretization is the weak formulation, which is obtained by multiplying each equations of the system (9.3) by a local arbitrary test function $w^e(x,y) \in \mathbb{V}^p$, and then integrating on each Bézier patch $\mathbb{D}^e$ separately. Note that, in this framework, no continuity on the state vectors $\mathbf{u}^e, \mathbf{v}^e, \mathbf{p}^e$ and the test function w^e is enforced along the interfaces between Bézier patches. The weak formulation of the acoustic wave equation is given for $e = 1,...,\mathfrak{N}_{el}$ for each equations of the system (9.3) by the following statement:

9.2.2 First variational equation

Applying a IGDG method to equation (1) of the system (9.4), the weak form of the problem can be written for each element $\mathbb{D}^e$ as:

$$\int_{\mathbb{D}^e} \Big(\partial_t \mathbf{u}^e(x,y,t) + \partial_x \mathbf{p}^e(x,y,t) \Big) w^e(x,y) d\mathbb{D}^e = 0. \tag{9.4}$$

By applying Green's formula, the weak formulation can be written as:

$$\frac{\partial}{\partial t} \int_{\mathbb{D}^e} \mathbf{u}^e(x,y,t)\, w^e(x,y) d\mathbb{D}^e + \int_{\Gamma^e} \begin{pmatrix} \mathbf{p}^e(x,y,t) \\ 0 \end{pmatrix} \cdot \overrightarrow{n}^e w^e(x,y) d\Gamma^e - \int_{\mathbb{D}^e} \begin{pmatrix} \mathbf{p}^e(x,y,t) \\ 0 \end{pmatrix} \cdot \nabla w^e(x,y) d\mathbb{D}^e = 0.$$

We denote: $\overrightarrow{e}_x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$ and $\overrightarrow{n}^e$ the outer unit normal to Γ^e of the element $\mathbb{D}^e$, so we get:

$$\frac{\partial}{\partial t} \int_{\mathbb{D}^e} \mathbf{u}^e(x,y,t)\, w^e(x,y) d\mathbb{D}^e = \int_{\mathbb{D}^e} \mathbf{p}^e(x,y,t) \overrightarrow{e}_x \cdot \nabla w^e(x,y) d\mathbb{D}^e - \int_{\Gamma^e} \mathbf{p}^e(x,y,t) \overrightarrow{e}_x \cdot \overrightarrow{n}^e w^e(x,y) d\Gamma^e.$$

By discretizing the problem on the Bézier basis associated with the element and using $\mathfrak{B}^e_{k,l}$ as test function, the problem is written:

$$\begin{aligned}\sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\partial_t \mathbf{u}^e_{i,j}(t)\Big(\int_{\mathbb{D}^e}\mathfrak{B}^e_{i,j}(x,y)\mathfrak{B}^e_{k,l}(x,y)d\mathbb{D}^e\Big) &= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\mathbf{p}^e_{i,j}(t)\Big(\int_{\mathbb{D}^e}\mathfrak{B}^e_{i,j}(x,y)\overrightarrow{e}_x\cdot\nabla\mathfrak{B}^e_{k,l}(x,y)d\mathbb{D}^e\Big)\\ &- \int_{\Gamma^e}\mathfrak{B}^e_{k,l}(x,y)\Big(\overrightarrow{e}_x\mathbf{p}^e\Big)\cdot\overrightarrow{n}^e d\Gamma^e\end{aligned}$$

$$\forall 1\le k\le p+1, \forall 1\le l\le q+1.$$

Therefore, the local problem takes the form of a linear system of size $(p+1)^2\times(q+1)^2$, which can be written in the following matrix form:

$$\mathbf{M}^e\partial_t\mathbf{u}^e = \mathbf{R}^e_x\mathbf{p}^e - F^e_x \qquad \forall t\in[0,T],\quad \forall 1\le e\le \mathfrak{N}_{el}. \tag{9.5}$$

9.2.3 Second variational equation

In the same way, the weak form of the equation (2) of the system (9.4) can be described for each element $\mathbb{D}^e$ as follows:

$$\int_{\mathbb{D}^e}\Big(\partial_t\mathbf{v}^e(x,y,t)+\partial_y\mathbf{p}^e(x,y,t)\Big)w^e(x,y)d\mathbb{D}^e = 0. \tag{9.6}$$

By applying Green's formula, the weak formulation can be written as:

$$\frac{\partial}{\partial t}\int_{\mathbb{D}^e}\mathbf{v}^e(x,y,t)w^e(x,y)d\mathbb{D}^e + \int_{\Gamma^e}\begin{pmatrix}0\\ \mathbf{p}^e(x,y,t)\end{pmatrix}\cdot\overrightarrow{n}^e w^e(x,y)d\Gamma^e - \int_{\mathbb{D}^e}\begin{pmatrix}0\\ \mathbf{p}^e(x,y,t)\end{pmatrix}\cdot\nabla w^e(x,y)d\mathbb{D}^e = 0.$$

We denote: $\overrightarrow{e}_y = \begin{pmatrix}0\\1\end{pmatrix}$, so we get:

$$\frac{\partial}{\partial t}\int_{\mathbb{D}^e}\mathbf{v}^e(x,y,t)w^e(x,y)d\mathbb{D}^e = \int_{\mathbb{D}^e}\mathbf{p}^e(x,y,t)\overrightarrow{e}_y\cdot\nabla w^e(x,y)d\mathbb{D}^e - \int_{\Gamma^e}\mathbf{p}^e(x,y,t)\overrightarrow{e}_y\cdot\overrightarrow{n}^e w^e(x,y)d\Gamma^e.$$

By discretizing the problem on the Bézier basis associated with the element and using $\mathfrak{B}^e_{k,l}$ as test function, the problem is written:

$$\begin{aligned}\sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\partial_t \mathbf{v}^e_{i,j}(t)\Big(\int_{\mathbb{D}^e}\mathfrak{B}^e_{i,j}(x,y)\mathfrak{B}^e_{k,l}(x,y)d\mathbb{D}^e\Big) &= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1}\mathbf{p}^e_{i,j}(t)\Big(\int_{\mathbb{D}^e}\mathfrak{B}^e_{i,j}(x,y)\overrightarrow{e}_y\cdot\nabla\mathfrak{B}^e_{k,l}(x,y)d\mathbb{D}^e\Big)\\ &- \int_{\Gamma^e}\mathfrak{B}^e_{k,l}(x,y)\Big(\overrightarrow{e}_y\mathbf{p}^e\Big)\cdot\overrightarrow{n}^e d\Gamma^e,\end{aligned}$$

$$\forall 1\le k\le p+1,\quad \forall 1\le l\le q+1.$$

Therefore, the local problem takes the form of a linear system of size $(p+1)^2 \times (q+1)^2$, which can be written in the following matrix form:

$$\mathbf{M}^e \partial_t \mathbf{v}^e = \mathbf{R}_y^e \mathbf{p}^e - F_y^e \qquad \forall\, t \in [0,T], \quad \forall 1 \le e \le \mathfrak{N}_{el}. \tag{9.7}$$

9.2.4 Third variational equation

In order to write the last variational equation of the system (9.4), we multiply the equation (3) with the test function w^e and integrate over $\mathbb{D}^e$:

$$\int_{\mathbb{D}^e} \Big(\partial_t \mathbf{p}^e(x,y,t) + \partial_x \mathbf{u}^e(x,y,t) + \partial_y \mathbf{v}^e(x,y,t)\Big) w^e(x,y)\, d\mathbb{D}^e = 0. \tag{9.8}$$

By applying Green's formula, the weak formulation can be written as:

$$\frac{\partial}{\partial t}\int_{\mathbb{D}^e} \mathbf{p}^e(x,y,t)\, w^e(x,y)\, d\mathbb{D}^e + \int_{\Gamma^e} \begin{pmatrix} \mathbf{u}^e(x,y,t) \\ \mathbf{v}^e(x,y,t) \end{pmatrix} \cdot \overrightarrow{n}^e w^e(x,y)\, d\mathbb{D}^e - \int_{\mathbb{D}^e} \begin{pmatrix} \mathbf{u}^e(x,y,t) \\ \mathbf{v}^e(x,y,t) \end{pmatrix} \cdot \nabla w^e(x,y)\, d\mathbb{D}^e = 0.$$

so we get:

$$\begin{aligned} \frac{\partial}{\partial t}\int_{\mathbb{D}^e} \mathbf{p}^e(x,y,t)\, w^e(x,y)\, d\mathbb{D}^e &= \int_{\mathbb{D}^e} \Big(\mathbf{u}^e(x,y,t)\,\overrightarrow{e}_x + \mathbf{v}^e(x,y,t)\,\overrightarrow{e}_y\Big) \cdot \nabla w^e(x,y)\, d\mathbb{D}^e \\ &- \int_{\Gamma^e} \Big(\mathbf{u}^e(x,y,t)\,\overrightarrow{e}_x + \mathbf{v}^e(x,y,t)\,\overrightarrow{e}_y\Big) \cdot \overrightarrow{n}^e w^e(x,y)\, d\Gamma^e. \end{aligned}$$

By discretizing the problem on the Bézier basis associated with the element and using $\mathfrak{B}_{k,l}^e$ as test function, the problem is written:

$$\begin{aligned} \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \partial_t \mathbf{p}_{i,j}^e(t) \Big(\int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(x,y)\mathfrak{B}_{k,l}^e(x,y)\, d\mathbb{D}^e\Big) &= \sum_{i=1}^{p+1}\sum_{j=1}^{q+1} \Big(\mathbf{u}_{i,j}^e(t)\,\overrightarrow{e}_x + \mathbf{v}_{i,j}^e(t)\,\overrightarrow{e}_y\Big)\Big(\int_{\mathbb{D}^e} \mathfrak{B}_{i,j}^e(x,y) \cdot \nabla \mathfrak{B}_{k,l}^e(x,y)\, d\mathbb{D}^e\Big) \\ &- \int_{\Gamma^e} \mathfrak{B}_{k,l}^e(x,y)\Big(\overrightarrow{e}_x \mathbf{u}^e + \overrightarrow{e}_y \mathbf{v}^e\Big) \cdot \overrightarrow{n}^e d\Gamma^e, \\ \forall 1 \le k \le p+1, \quad \forall 1 \le l \le q+1. \end{aligned}$$

Therefore, this local problem leads to a linear system of size $(p+1)^2 \times (q+1)^2$, which can be written in the following matrix form:

$$\mathbf{M}^e \partial_t \mathbf{p}^e = \mathbf{R}_x^e \mathbf{u}^e + \mathbf{R}_y^e \mathbf{p}^e - \big(F_x^e + F_y^e\big) \qquad \forall\, t \in [0,T] \quad \forall 1 \le e \le \mathfrak{N}_{el}. \tag{9.9}$$

9.3 Elementary linear system

The coefficients of the local mass and stiffness matrix for the element $\mathbb{D}^e$ are written as:

$$\begin{aligned}\mathbf{M}^e_{kl,ij} &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}(x,y)\mathfrak{B}^e_{k,l}(x,y)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}(\mathbb{T}(\xi,\eta))\mathfrak{B}^e_{k,l}(\mathbb{T}(\xi,\eta))d\mathbb{D}^e.\end{aligned}$$

By performing the integration in the local parametric space $\widetilde{\mathbb{D}}^e$, we get:

$$\mathbf{M}^e_{kl,ij} = \int_{\widetilde{\mathbb{D}}^e} \Big(\big(B_{i,p}(\xi)\big)^e\big(B_{j,q}(\eta)\big)^e\Big)\Big(\big(B_{k,p}(\xi)\big)^e\big(B_{l,q}(\eta)\big)^e\Big) \mid J^e(\xi,\eta) \mid d\widetilde{\mathbb{D}}^e,$$
$$\forall i,k = 1,...,p+1, \quad j,l = 1,...,q+1,$$

where $\mid J^e \mid$ is the local Jacobien determinant defined in the previous chapter by:

$$\mid J^e \mid = \Big(\big(\frac{\partial x}{\partial \xi}\big)\big(\frac{\partial y}{\partial \eta}\big)\Big)^e - \Big(\big(\frac{\partial x}{\partial \eta}\big)\big(\frac{\partial y}{\partial \xi}\big)\Big)^e.$$

Achieving the integration in the unit square $\widehat{\mathbb{D}}$, we get:

$$\mathbf{M}^e_{kl,ij} = \int_{\widehat{\mathbb{D}}} \Big(\big(B_{i,p}(\hat{\xi})\big)^e\big(B_{j,q}(\hat{\eta})\big)^e\Big)\Big(\big(B_{k,p}(\hat{\xi})\big)^e\big(B_{l,q}(\hat{\eta})\big)^e\Big) \mid \hat{J}^e(\xi,\eta) \mid\mid J^e(\xi,\eta) \mid d\widehat{\mathbb{D}},$$
$$\forall i,k = 1,...,p+1 \qquad j,l = 1,...,q+1.$$

The integrals are evaluated numerically via Gaussian quadrature.

$$\begin{aligned}\big(\mathfrak{R}^e_x\big)_{kl,ij} &= \int_{\mathbb{D}^e} \Big(\overrightarrow{e}_x \mathfrak{B}^e_{ij}(x,y)\Big)\cdot\nabla\mathfrak{B}^e_{kl}(x,y)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{ij}(x,y)\Big(\frac{\partial\mathfrak{B}^e_{kl}(x,y)}{\partial x}\Big)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}\big(\mathbb{T}(\xi,\eta)\big)\frac{\partial}{\partial x}\mathfrak{B}^e_{k,l}\big(\mathbb{T}(\xi,\eta)\big)d\mathbb{D}^e,\end{aligned}$$

$$\begin{aligned}\big(\mathfrak{R}^e_y\big)_{kl,ij} &= \int_{\mathbb{D}^e} \Big(\overrightarrow{e}_y \mathfrak{B}^e_{ij}(x,y)\Big)\cdot\nabla\mathfrak{B}^e_{kl}(x,y)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{ij}(x,y)\Big(\frac{\partial\mathfrak{B}^e_{kl}(x,y)}{\partial y}\Big)d\mathbb{D}^e \\ &= \int_{\mathbb{D}^e} \mathfrak{B}^e_{i,j}\big(\mathbb{T}(\xi,\eta)\big)\frac{\partial}{\partial y}\mathfrak{B}^e_{k,l}\big(\mathbb{T}(\xi,\eta)\big)d\mathbb{D}^e,\end{aligned}$$

$$\big(\mathfrak{R}^e\big)_{kl,ij} = \big(\mathfrak{R}^e_x\big)_{kl,ij} + \big(\mathfrak{R}^e_y\big)_{kl,ij}.$$

By expressing the integration in the local parametric space $\widetilde{\mathbb{D}}^e$, we get:

$$\left(\mathfrak{R}_x^e\right)_{kl,ij} = \int_{\widetilde{\mathbb{D}}^e} \Big(\big(B_{i,p}(\xi)\big)^e \big(B_{j,q}(\eta)\big)^e\Big)\Big(\big(\frac{\partial B^e_{k,p}(\xi)}{\partial \xi}\big)\big(B^e_{l,q}(\eta)\frac{\partial \xi}{\partial x}\big) + \big(B^e_{k,p}(\xi)\big)\big(\frac{\partial B^e_{l,q}(\eta)}{\partial \eta}\frac{\partial \eta}{\partial x}\big)\Big) \mid J^e(\xi,\eta)\mid d\widetilde{\mathbb{D}}^e,$$

and,

$$\left(\mathfrak{R}_y^e\right)_{kl,ij} = \int_{\widetilde{\mathbb{D}}^e} \Big(\big(B_{i,p}(\xi)\big)^e \big(B_{j,q}(\eta)\big)^e\Big)\Big(\big(\frac{\partial B^e_{k,p}(\xi)}{\partial \xi}\big)\big(B^e_{l,q}(\eta)\frac{\partial \xi}{\partial y}\big) + \big(B^e_{k,p}(\xi)\big)\big(\frac{\partial B^e_{l,q}(\eta)}{\partial \eta}\frac{\partial \eta}{\partial y}\big)\Big) \mid J^e(\xi,\eta)\mid d\widetilde{\mathbb{D}}^e.$$

By performing the integration in the reference square $\widehat{\mathbb{D}}$, we get:

$$\left(\mathfrak{R}_x^e\right)_{kl,ij} = \int_{\widehat{\mathbb{D}}} \Big(\big(B_{i,p}(\hat{\xi})\big)^e \big(B_{j,q}(\hat{\eta})\big)^e\Big)\Big(\big(\frac{\partial B^e_{k,p}(\hat{\xi})}{\partial \hat{\xi}}\big)\big(B^e_{l,q}(\hat{\eta})\frac{\partial \hat{\xi}}{\partial x}\big) + \big(B^e_{k,p}(\hat{\xi})\big)\big(\frac{\partial B^e_{l,q}(\hat{\eta})}{\partial \hat{\eta}}\frac{\partial \hat{\eta}}{\partial x}\big)\Big) \mid \hat{J}^e(\hat{\xi},\hat{\eta})\mid \mid J^e(\xi,\eta)\mid d\widehat{\mathbb{D}},$$

$$\left(\mathfrak{R}_y^e\right)_{kl,ij} = \int_{\widehat{\mathbb{D}}} \Big(\big(B_{i,p}(\hat{\xi})\big)^e \big(B_{j,q}(\hat{\eta})\big)^e\Big)\Big(\big(\frac{\partial B^e_{k,p}(\hat{\xi})}{\partial \hat{\xi}}\big)\big(B^e_{l,q}(\hat{\eta})\frac{\partial \hat{\xi}}{\partial y}\big) + \big(B^e_{k,p}(\hat{\xi})\big)\big(\frac{\partial B^e_{l,q}(\hat{\eta})}{\partial \hat{\eta}}\frac{\partial \hat{\eta}}{\partial y}\big)\Big) \mid \hat{J}^e(\hat{\xi},\hat{\eta})\mid \mid J^e(\xi,\eta)\mid d\widehat{\mathbb{D}}.$$

At the end, the computation is achieved using Gauss-Legendre quadrature rules.

9.4 Numerical Lax–Friedrichs fluxes

The selection of the numerical flux is of utmost importance when formulating a DG method. The numerical flux must accurately couple the neighboring elements, while yielding a stable scheme. There are several flux methods. In our formulation, we only discuss the Lax-Friedrich flux [73], given by:

$$\widehat{f_n^e}(\mathbf{u}_l,\mathbf{u}_r) = \frac{1}{2}\big(f_n(\mathbf{u}_l^e) + f_n(\mathbf{u}_r^e)\big) - \frac{\rho}{2}\big(\mathbf{u}_r^e - \mathbf{u}_l^e\big), \tag{9.10}$$

where $\mathbf{u}_l$ and $\mathbf{u}_r$ are the left and right limits of the discontinuous solution $\mathbf{u}_h$. For the Lax-Friedrichs flux, ρ is taken as an upper bound for $\mid f'(\mathbf{u}) \mid$ in the scalar case, or for the absolute value of eigenvalues of the Jacobian for the system case. For the acoustic wave equation, ρ is the sound speed assumed $\rho = 1$.

Since the weak form of the DG method is written elementwise, the numerical flux between adjacent elements must be defined. For this purpose, it is possible to write:

$$
\begin{aligned}
\widehat{F^e} &= \sum_{k=1}^{4} \int_{\Gamma_k^e} \widehat{f_n^e} \mathfrak{B}_{k,l}^e(x,y) d\Gamma_k^e \\
&= \int_{[0,1]} \widehat{f_n^e|_{\Gamma_1^e}} \big(B_{k,p}(\xi)\big)^e \big(\underbrace{B_{l,q}(0)}_{=1}\big)^e \mid J^e(\xi,0) \mid\mid \widehat{J}^e(\widehat{\xi},0) \mid d\xi \\
&+ \int_{[0,1]} \widehat{f_n^e|_{\Gamma_2^e}} \big(\underbrace{B_{k,p}(1)}_{=1}\big)^e \big(B_{l,q}(\eta)\big)^e \mid J^e(1,\eta) \mid\mid \widehat{J}^e(1,\widehat{\eta}) \mid d\eta \\
&+ \int_{[0,1]} \widehat{f_n^e|_{\Gamma_3^e}} \big(B_{k,p}(\xi)\big)^e \big(\underbrace{B_{l,q}(1)}_{=1}\big)^e \mid J^e(\xi,1) \mid\mid \widehat{J}^e(\widehat{\xi},1) d\xi \\
&+ \int_{[0,1]} \widehat{f_n^e|_{\Gamma_4^e}} \big(\underbrace{B_{k,p}(0)}_{=1}\big)^e \big(B_{l,q}(\eta)\big)^e \mid J^e(0,\eta) \mid\mid \widehat{J}^e(0,\widehat{\eta}) d\eta.
\end{aligned}
$$

It is also necessary to define for the patch $\mathbb{D}^e$, the normal vectors on the four interfaces Γ_1^e, Γ_2^e, Γ_3^e and Γ_4^e given by:

$$
\begin{aligned}
\overrightarrow{n}^e_{|_{\Gamma_1^e}} &= \begin{pmatrix} \overrightarrow{n^e_{x|_{\Gamma_1^e}}} \\ \overrightarrow{n^e_{y|_{\Gamma_1^e}}} \end{pmatrix} = \begin{pmatrix} \frac{\partial y}{\partial \xi} \\ -\frac{\partial x}{\partial \xi} \end{pmatrix}^e, \\
\overrightarrow{n^e}_{|_{\Gamma_2^e}} &= \begin{pmatrix} \overrightarrow{n^e_{x|_{\Gamma_2^e}}} \\ \overrightarrow{n^e_{y|_{\Gamma_2^e}}} \end{pmatrix} = \begin{pmatrix} \frac{\partial y}{\partial \eta} \\ -\frac{\partial x}{\partial \eta} \end{pmatrix}^e, \\
\overrightarrow{n^e}_{|_{\Gamma_3^e}} &= \begin{pmatrix} \overrightarrow{n^e_{x|_{\Gamma_3^e}}} \\ \overrightarrow{n^e_{y|_{\Gamma_3^e}}} \end{pmatrix} = \begin{pmatrix} -\frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \xi} \end{pmatrix}^e, \\
\overrightarrow{n^e}_{|_{\Gamma_4^e}} &= \begin{pmatrix} \overrightarrow{n^e_{x|_{\Gamma_4^e}}} \\ \overrightarrow{n^e_{y|_{\Gamma_4^e}}} \end{pmatrix} = \begin{pmatrix} -\frac{\partial y}{\partial \eta} \\ \frac{\partial x}{\partial \eta} \end{pmatrix}^e.
\end{aligned}
$$

Obviously, these vectors are normalized for the computations.

9.5 Numerical results

We consider the example of an ideal acoustic resonator between two cylinders solved using the acoustic wave equation (9.3) over the physical domain Ω represented in Fig. 9.1, for which an analytical solution exists.

Two types of computational domains are considered: a curvilinear one constructed by least squares approximation and knot insertion (described in chapter 4) and a rectilinear one, that corresponds to classical grids with straight element interfaces.

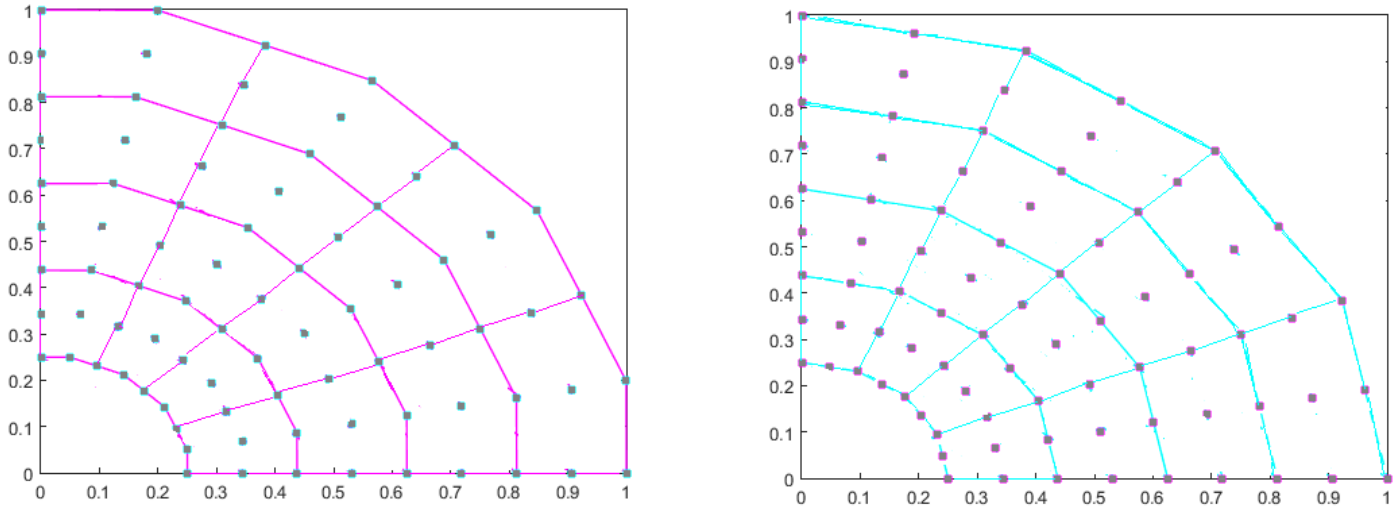


Figure 9.1: Control point lattice for quadratic rectilinear grid (on the right) and curvilinear grid (on the left).

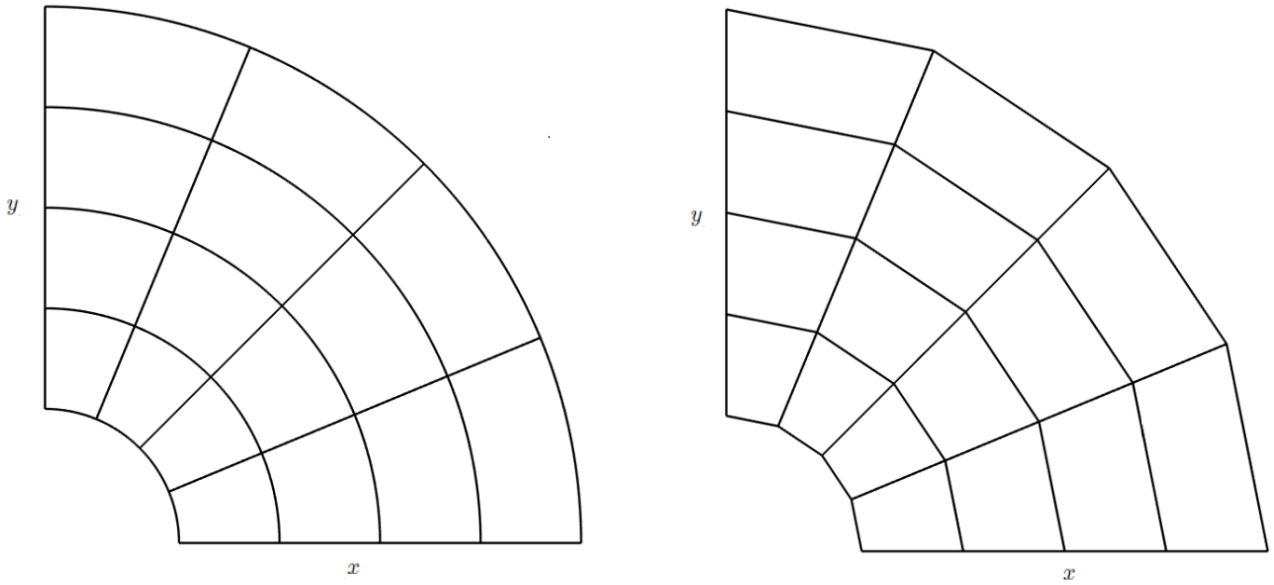


Figure 9.2: Rectilinear grid on the right and curvilinear grid on the left (4×4 elements).

The initial conditions are:

$$\begin{cases} \mathbf{p}_{|_{t=0}} = J_0\big(\alpha_1 + \frac{r-r_1}{r_2-r_1}(\alpha_2-\alpha_1)\big), \\ \vec{\mathscr{U}}_{|_{t=0}} = \big(\mathbf{u},\mathbf{v}\big)_{|_{t=0}} = (0,0). \end{cases}$$

This solution is provided where $\alpha_1 = 3.8317$, $\alpha_2 = 10.1735$ (α_1 and α_2: roots of $J_0' = J_1$) (see appendix C) , $r_1 = 0.25$, $r_2 = 1$ and J_0 the Bessel function of the first kind.
The exact solutions is given in cylindrical coordinates (r,θ) by:

$$\begin{cases} \mathbf{p}_{\text{ex}} = J_0\big(\alpha_1 + \frac{r-r_1}{r_2-r_1}(\alpha_2-\alpha_1)\big)\cos\big(\frac{\alpha_2-\alpha_1}{r_2-r_1}t\big), \\ \mathbf{u}_{\text{ex}} = J_1\big(\alpha_1 + \frac{r-r_1}{r_2-r_1}(\alpha_2-\alpha_1)\big)\sin\big(\frac{\alpha_2-\alpha_1}{r_2-r_1}t\big)\cos\big(\theta\big), \\ \mathbf{v}_{\text{ex}} = J_1\big(\alpha_1 + \frac{r-r_1}{r_2-r_1}(\alpha_2-\alpha_1)\big)\sin\big(\frac{\alpha_2-\alpha_1}{r_2-r_1}t\big)\sin\big(\theta\big). \end{cases}$$

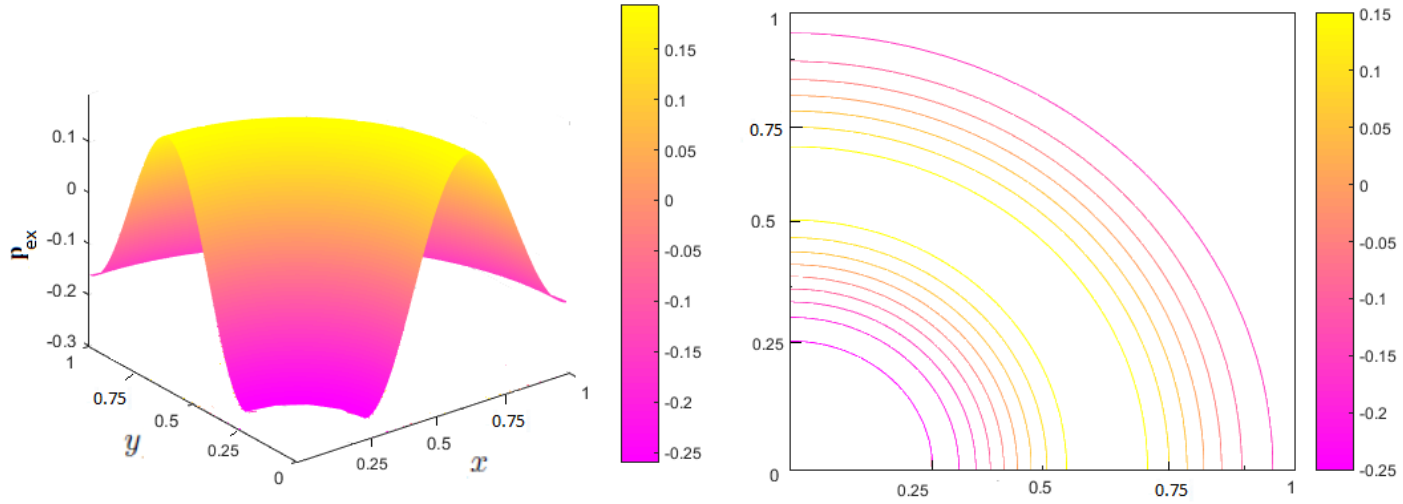


Figure 9.3: Plots and contour plots of the exact pressure $\mathbf{p}_{\text{ex}}$.

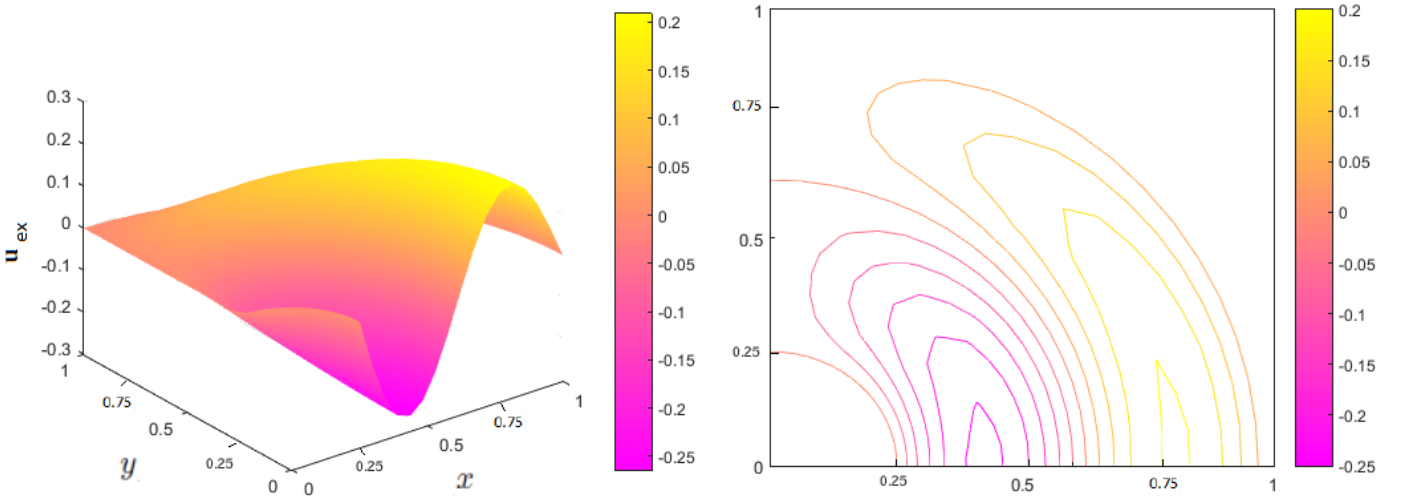


Figure 9.4: Plots and contour plots of $\mathbf{u}_{\text{ex}}$.

The results will be presented for rectilinear patches and curvilinear patches. The physical defined as a B-spline patch with $\mathfrak{N}_{el} = \mathfrak{N}^1_{el} \times \mathfrak{N}^2_{el} = 4 \times 4$ patches $\mathbb{D}^e$ which is plotted with the corresponding numerical solution for each choice of patches.

The bivariate Bernstein functions are taken to be of bi-degree (p, q). We focus for the case $p = q$ which will be specified in each example.

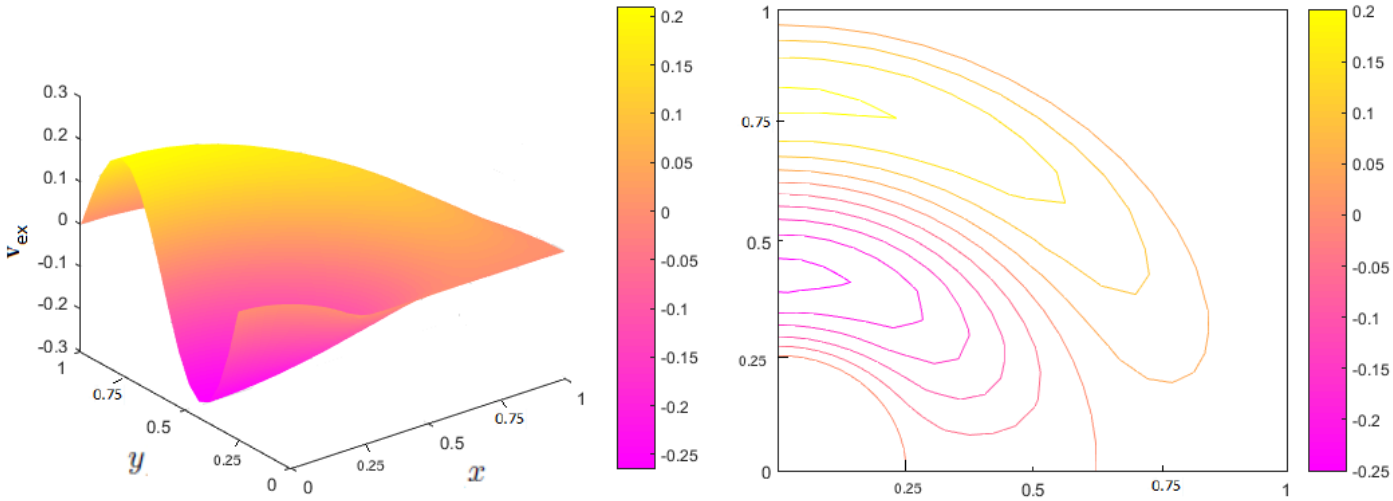


Figure 9.5: Plots and contour plots of $\mathbf{v}_{\text{analytic}}$.

9.5.1 Rectilinear grids

The first mesh type considered is a simple rectilinear grid. Figure 9.7 depicts the IGDG numerical solutions for the bivariate quadratic Bernstein case for the initial mesh with $\mathfrak{N}_{el} = 4 \times 4$ patches (as shown on the left in Fig. 9.6). We subsequently add a refinement (on the right of Fig. 9.6) of $(\frac{\mathfrak{h}_x}{2}, \frac{\mathfrak{h}_y}{2})$, the IGDG numerical solutions are shown in Fig. 9.8.

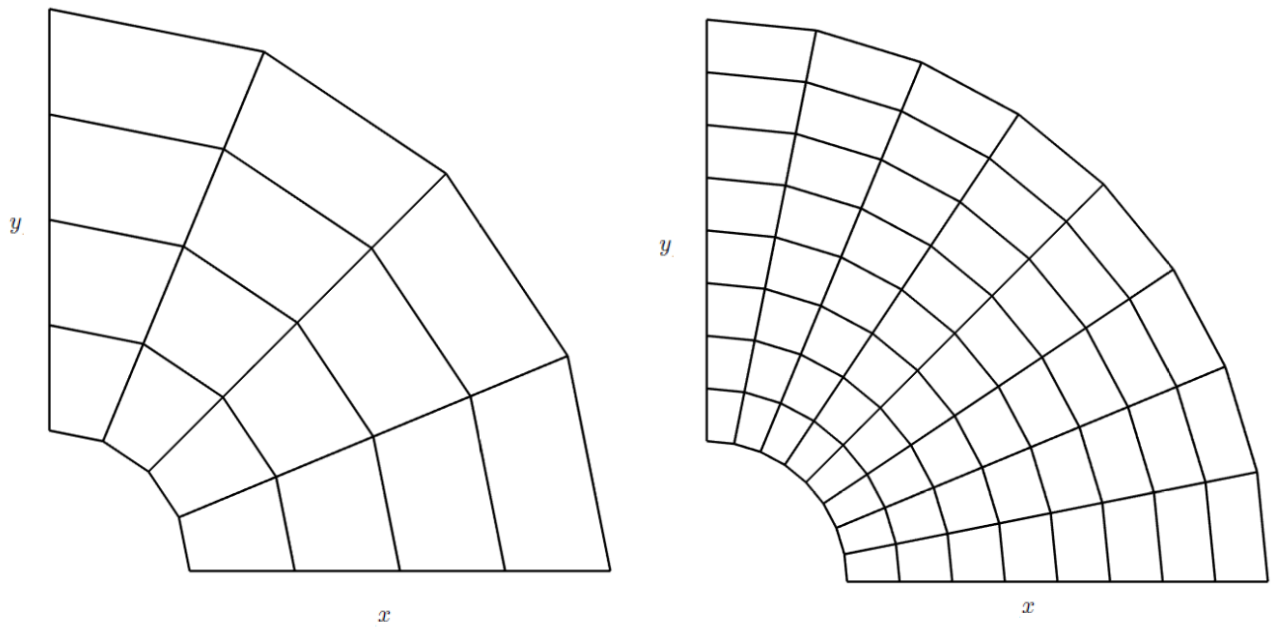


Figure 9.6: Rectilinear patches.

As can be seen on Figures 9.7 and 9.8, the use of a rectilinear grid strongly impacts the accuracy of the

solution near the boundaries of the domain. In particular, one can notice the presence of fictious variations of the solution at each boundary vertex, which are due to the fact that the boundary conditions are not prescribed at the true location and normals are not well approximated. When the rectilinear grid is refined, these effects are reduced but are still dommageable. In Figures 9.9 to 9.11, one can observe that increasing. The approximation order of the solution only does not provide any remedy to this issue.

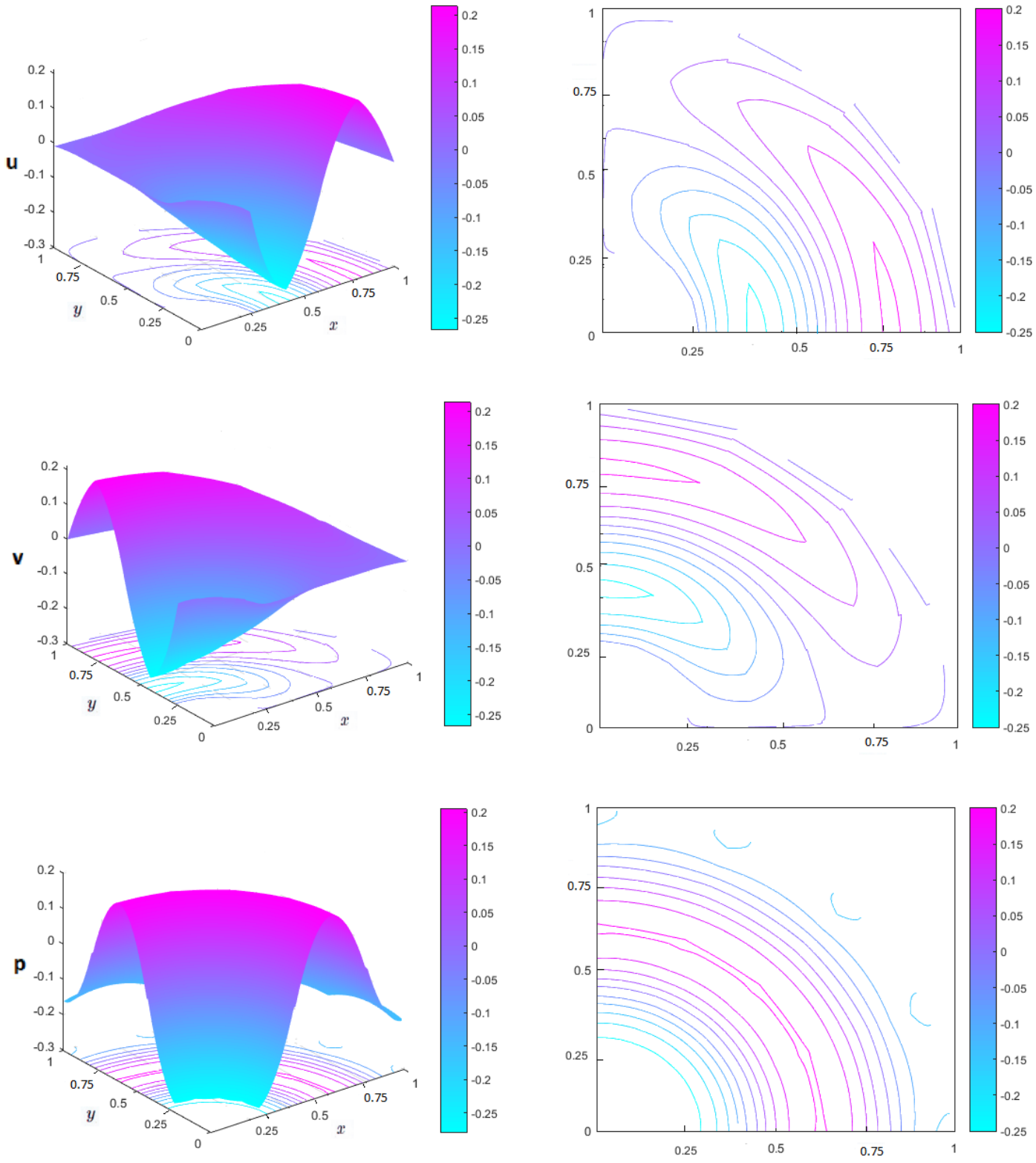


Figure 9.7: Plots and contour plots of numerical results for bivariate quadratic Bernstein basis with $\mathfrak{N}_{el} = 4 \times 4$ patches at $T = 0.1s$.

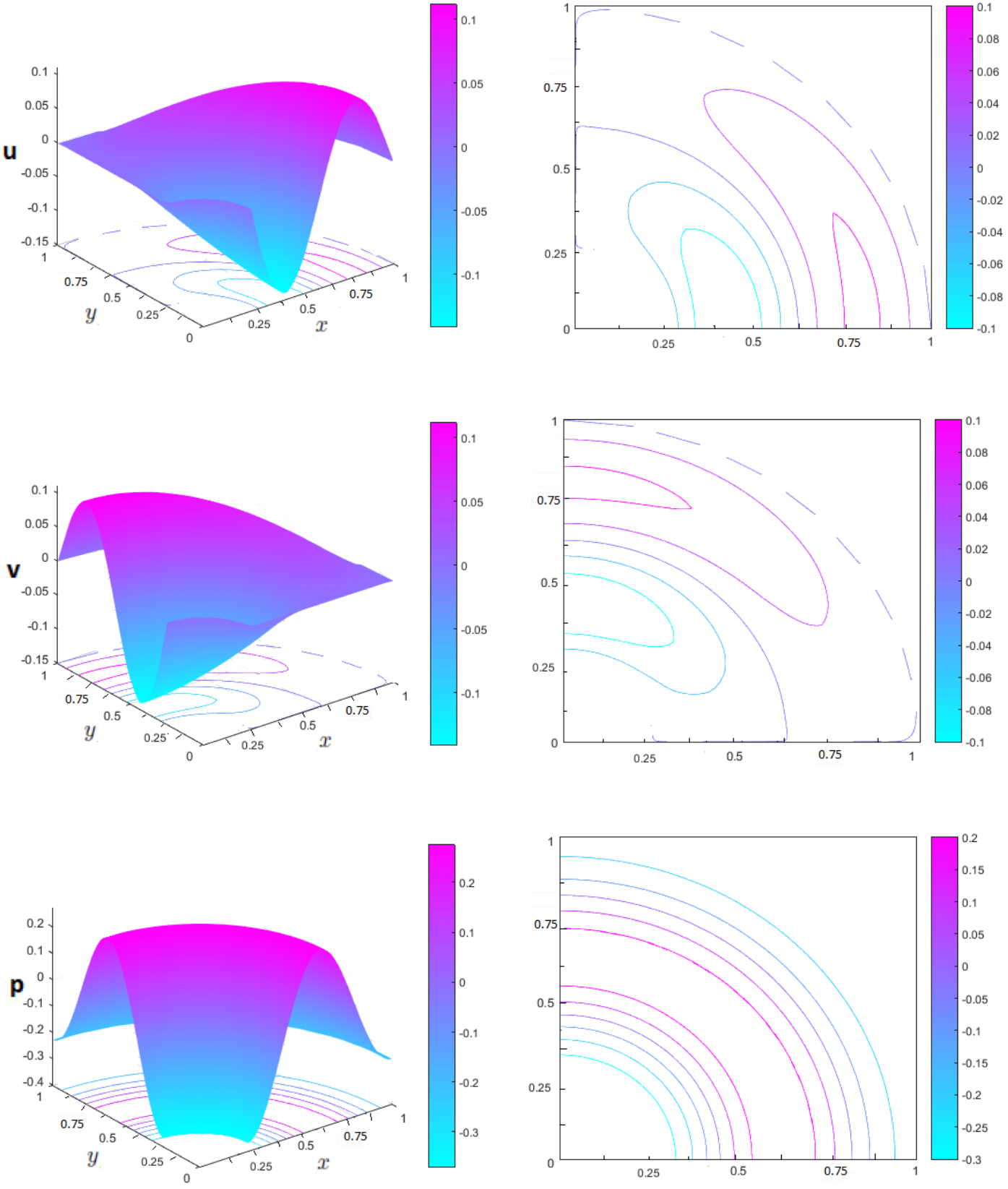


Figure 9.8: Plots and contour plots of numerical results for bivariate quadratic Bernstein basis $\mathfrak{N}_{el} = 8 \times 8$ patches at $T = 0.1s$.

Figure 9.9, Fig. 9.10 and Fig. 9.11 shows the numerical solutions from IGDG space discretization and explicit $RK4$ time integration for the quadratic, cubic and quatric Bernstein cases respectively. We can see the effect of the degree elevation of the bivariate Bernstein basis function on the accuracy.

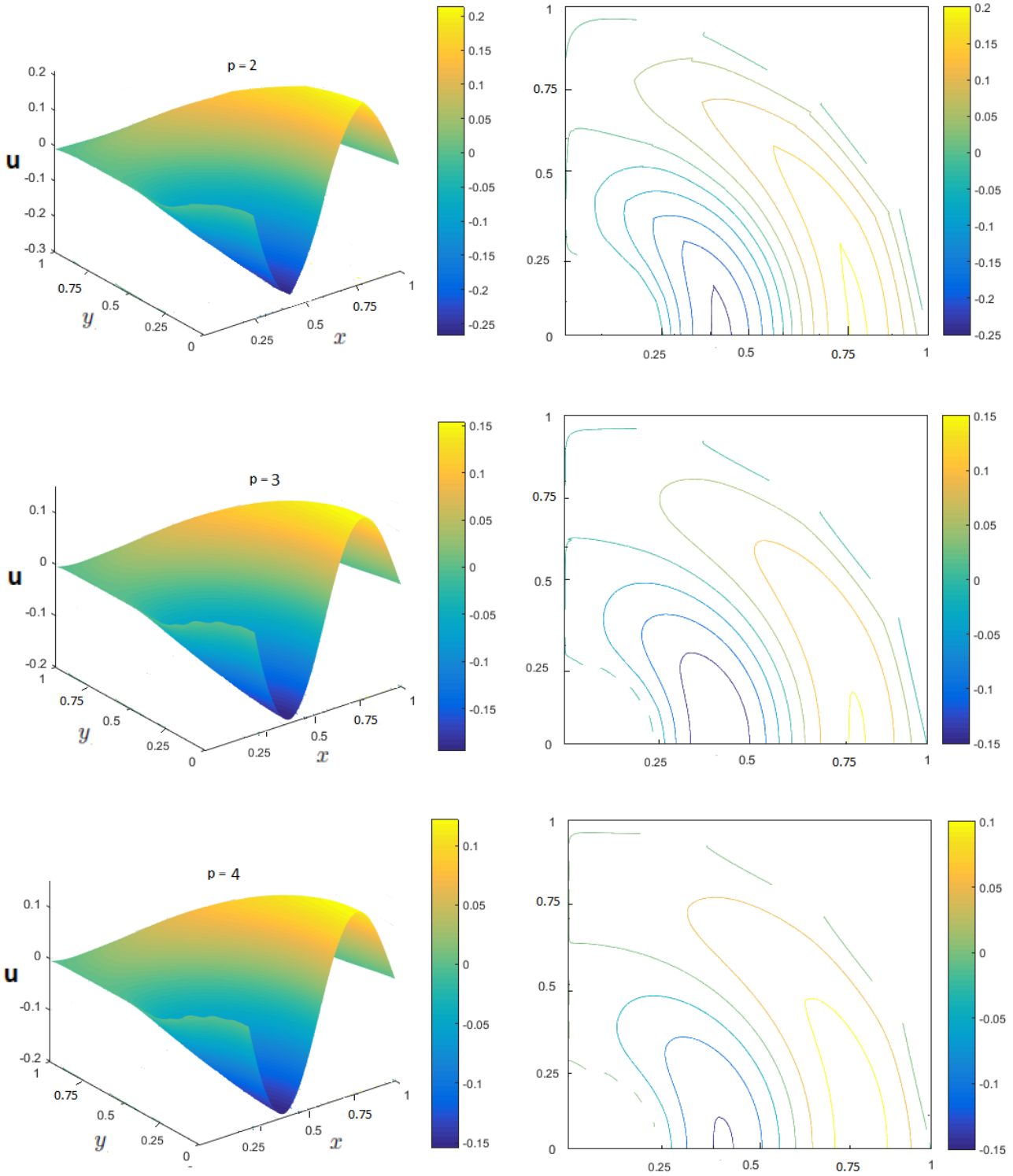


Figure 9.9: IGDG solution u for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

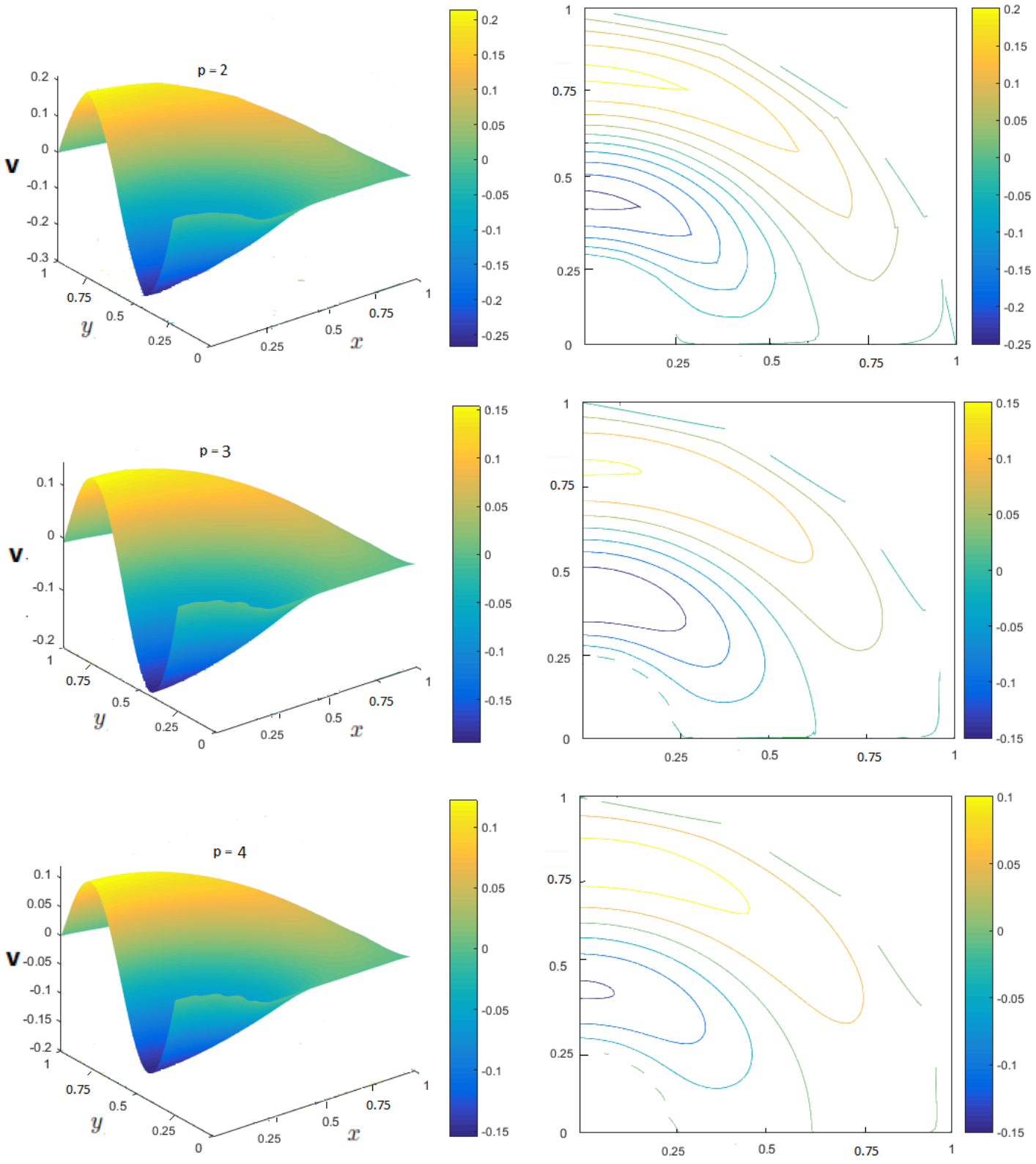


Figure 9.10: IGDG solution v for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

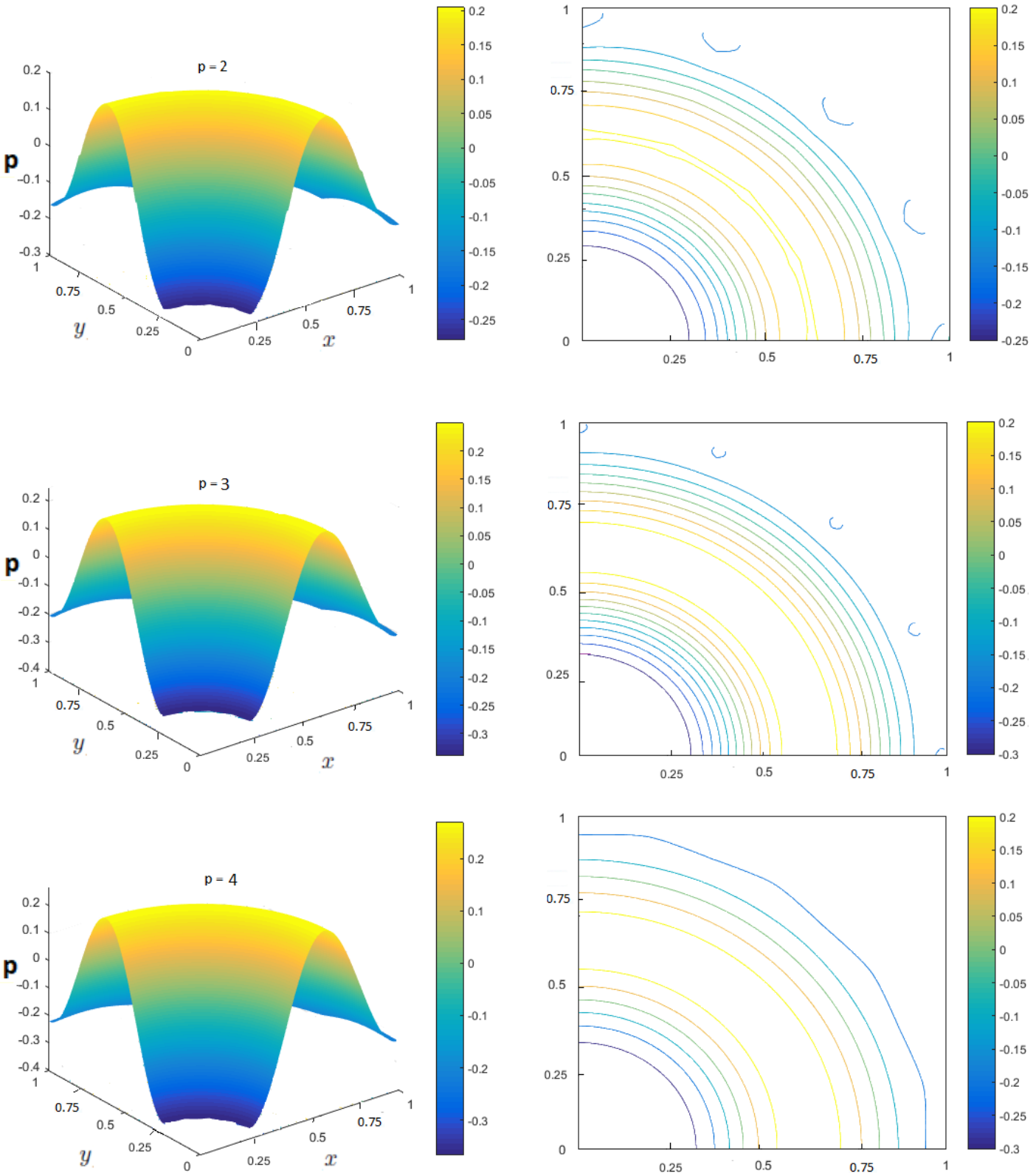


Figure 9.11: IGDG solution p for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

The L^2-error is now presented for rectilinear patches. Tables 9.1 and 9.2 summarize the convergence results in the L^2-norm for the bivariate quadratic, cubic and quartic Bernstein basis functions.

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.951E-01$	–
$\frac{\mathfrak{h}}{2}$	$5.241E-02$	1.89
$\frac{\mathfrak{h}}{4}$	$7.224E-03$	2.85
$\frac{\mathfrak{h}}{8}$	$1.928E-03$	1.90
$\frac{\mathfrak{h}}{16}$	$2.524E-04$	2.93

Table 9.1: L^2-error for the $2D$ acoustic problem and convergence order for the IGDG method for the quadratic Bernstein bases in conjunction with $RK4$ time discretisation.

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.324E-01$	–
$\frac{\mathfrak{h}}{2}$	$2.622E-02$	2.33
$\frac{\mathfrak{h}}{4}$	$6.244E-03$	2.07
$\frac{\mathfrak{h}}{8}$	$1.889E-03$	1.72
$\frac{\mathfrak{h}}{16}$	$2.480E-04$	2.92

Mesh	L^2-error	rate
$\mathfrak{h}$	$9.624E-02$	–
$\frac{\mathfrak{h}}{2}$	$2.579E-02$	1.78
$\frac{\mathfrak{h}}{4}$	$6.220E-03$	2.05
$\frac{\mathfrak{h}}{8}$	$1.890E-03$	1.71
$\frac{\mathfrak{h}}{16}$	$2.474E-04$	2.93

Table 9.2: L^2-error for the $2D$ acoustic problem and convergence order for the IGDG method for the cubic (left) and quartic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

The corresponding convergence data for uniform refinement $\mathfrak{h}$, $\frac{\mathfrak{h}}{2}$, $\frac{\mathfrak{h}}{4}$, $\frac{\mathfrak{h}}{8}$ and $\frac{\mathfrak{h}}{16}$ are shown in Figure 9.12.

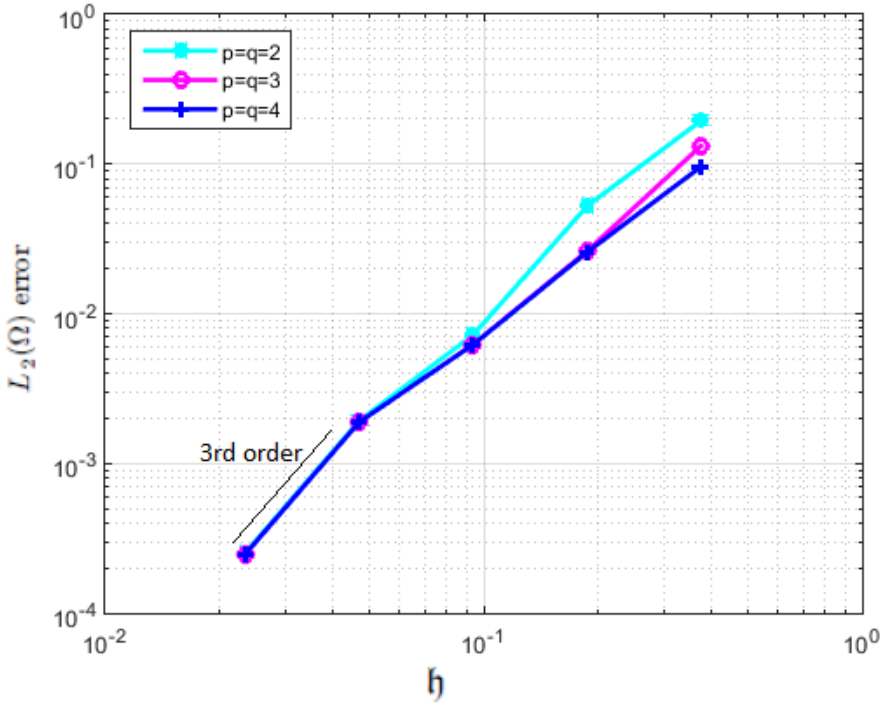


Figure 9.12: L^2-error for the $2D$ acoustic problem using the IGDG method in conjunction with $RK4$.

9.5.2 Curvilinear grids

We consider now the case of curvilinear grids. Figure 9.14 depicts the IGDG numerical solutions for the bivariate quadratic Bernstein case for the initial mesh with $\mathfrak{N}_{el} = 4 \times 4$ patches as shown on the left in Fig. 9.13. We subsequently add a reffinement of $(\frac{\mathfrak{h}_x}{2}, \frac{\mathfrak{h}_y}{2})$ as shown on the right in Fig. 9.13.

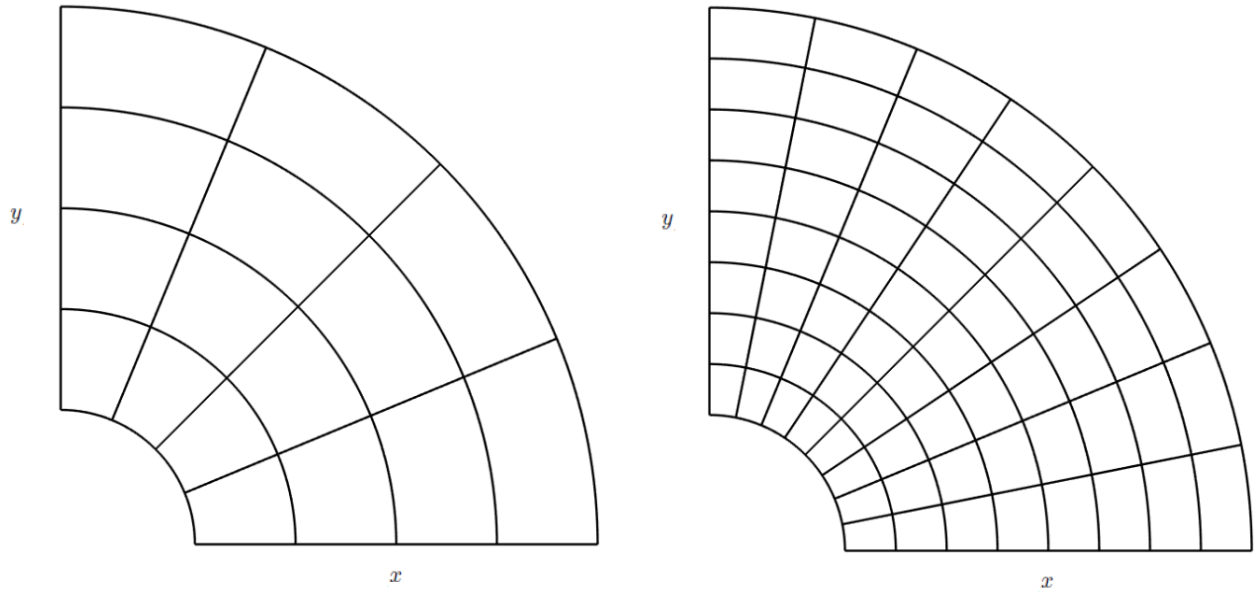


Figure 9.13: Curvilinear patches.

Contrary to the rectilinear case, the use of curvilinear grids yields a far better accuracy of the solution near the boundaries, as shown in figures 9.14 and 9.15. In particular, one can notice that the boundary isovalues are correctly captured, even on the coarsest grid. Figures 9.16 to 9.18 show that the increase of the approximation order of the solution yields a more accurate solution inside the domain while preserving the boundary accuracy. Note that, in this case, the circle is not exactly represented because B-spline representations are employed, instead of NURBS. Nevertheless, this does not seem to be critical in this example.

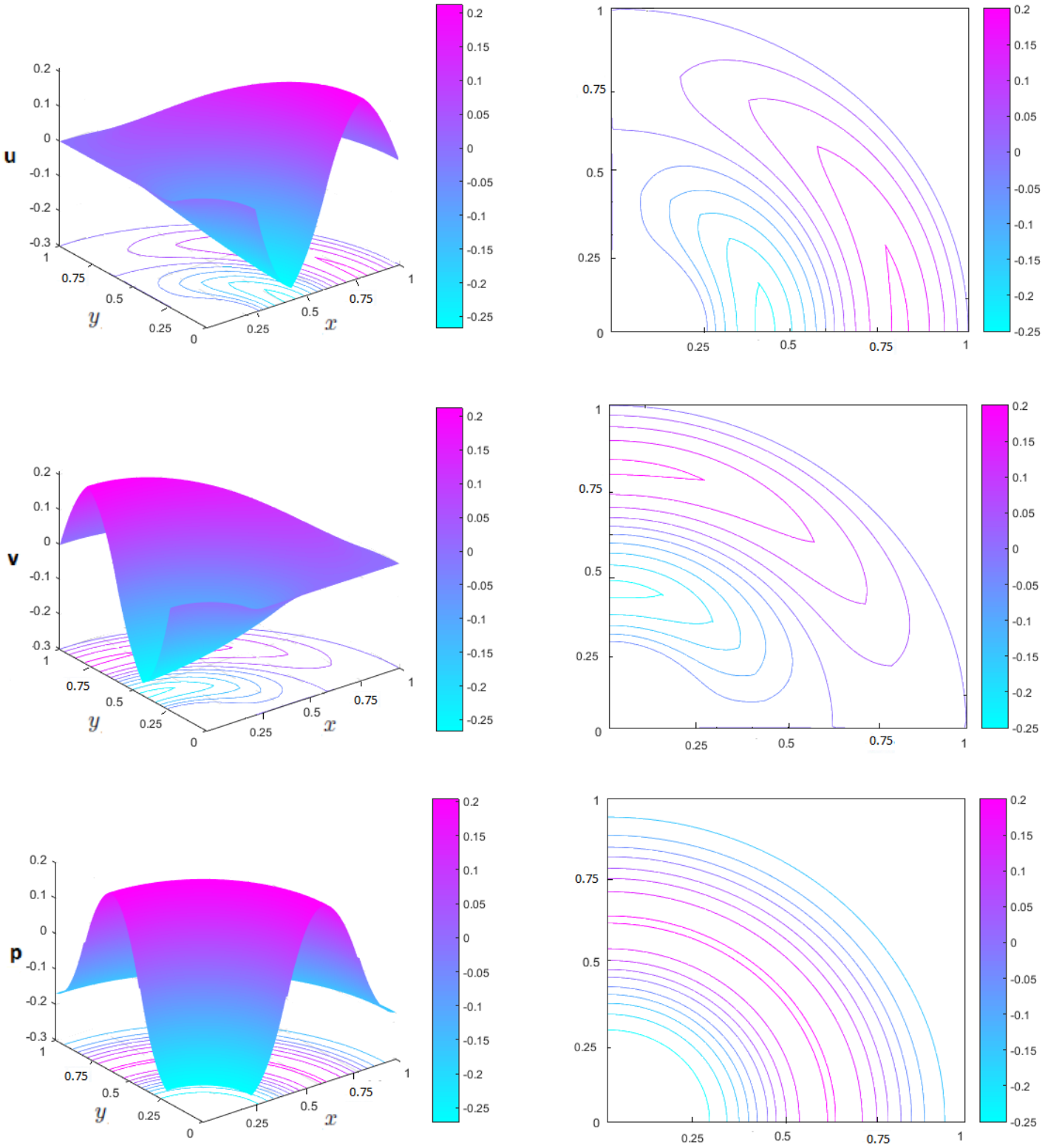


Figure 9.14: Plots and contour plots of numerical results for bivariate quadratic Bernstein basis with $\mathfrak{N}_{el} = 4 \times 4$ patches at $T = 0.1s$.

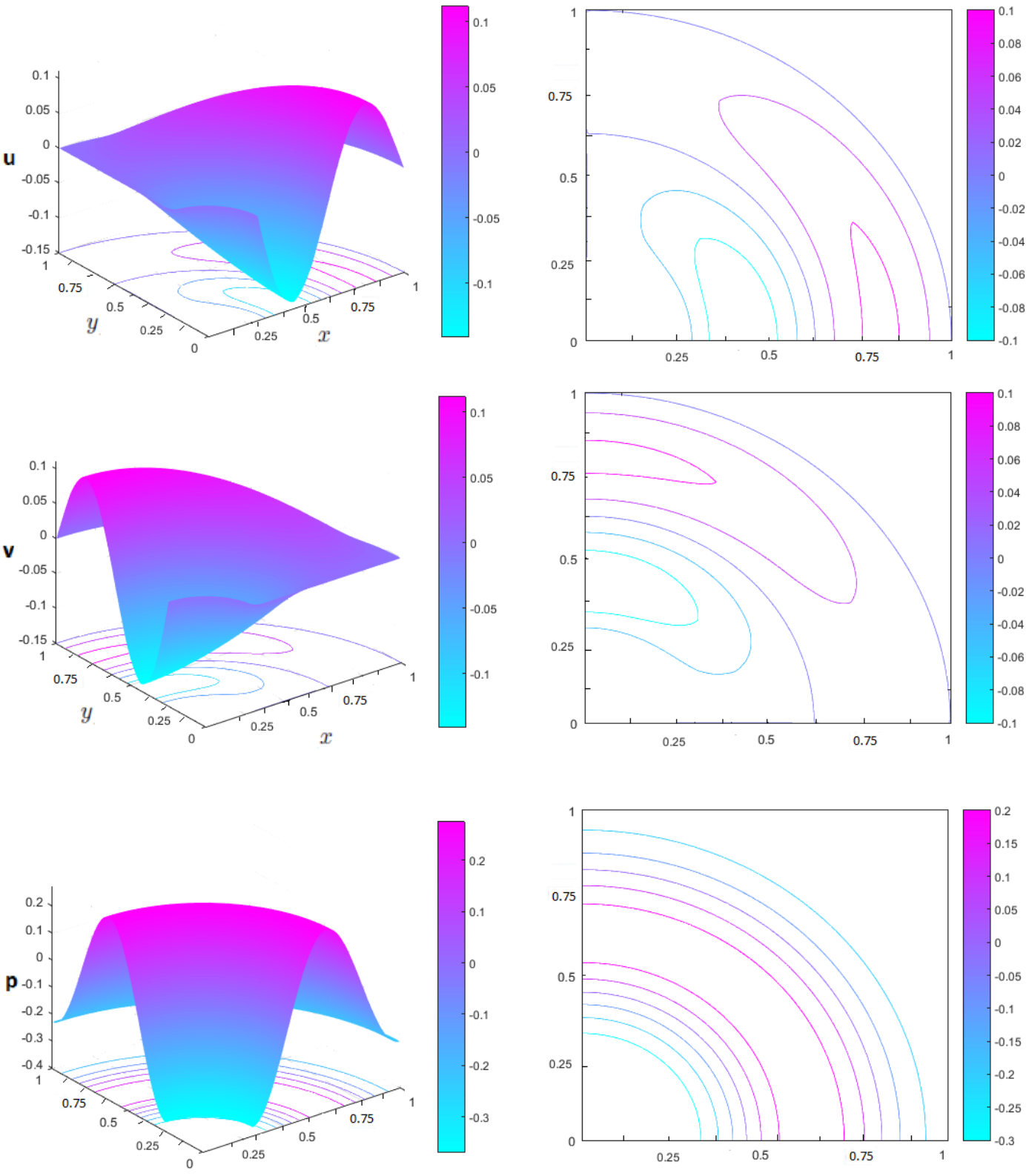


Figure 9.15: Plots and contour plots of numerical results for bivariate quadratic Bernstein basis $\mathfrak{N}_{el} = 8 \times 8$ patches at $T = 0.1s$.

Figure 9.16, Fig. 9.17 and Fig. 9.18 shows the numerical solutions from IGDG space discretization and explicit $RK4$ time integration for the quadratic, cubic and quatric Bernstein cases respectively. We can see the effect of the degree elevation of the bivariate Bernstein basis function on the accuracy.

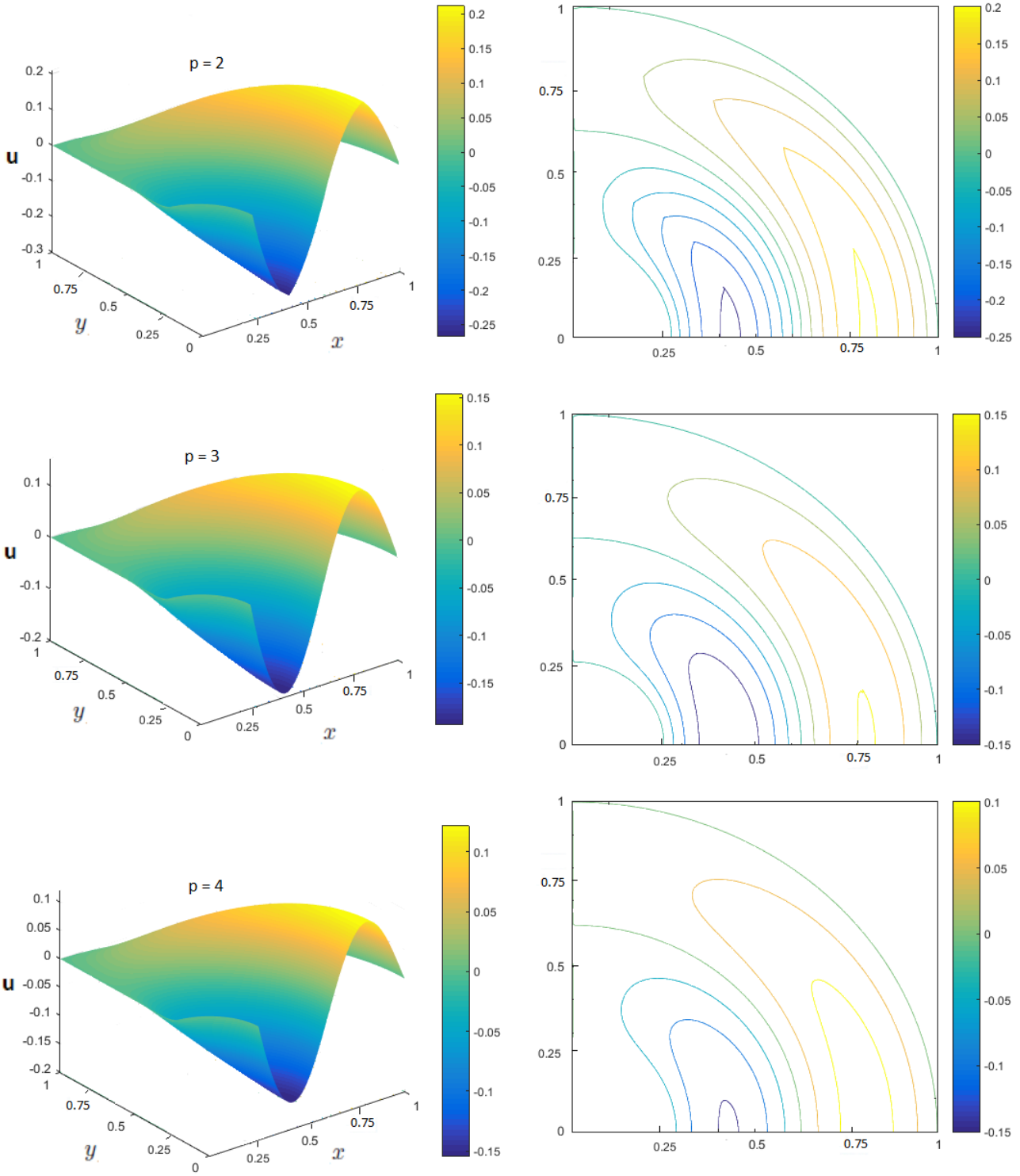


Figure 9.16: IGDG solution u for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

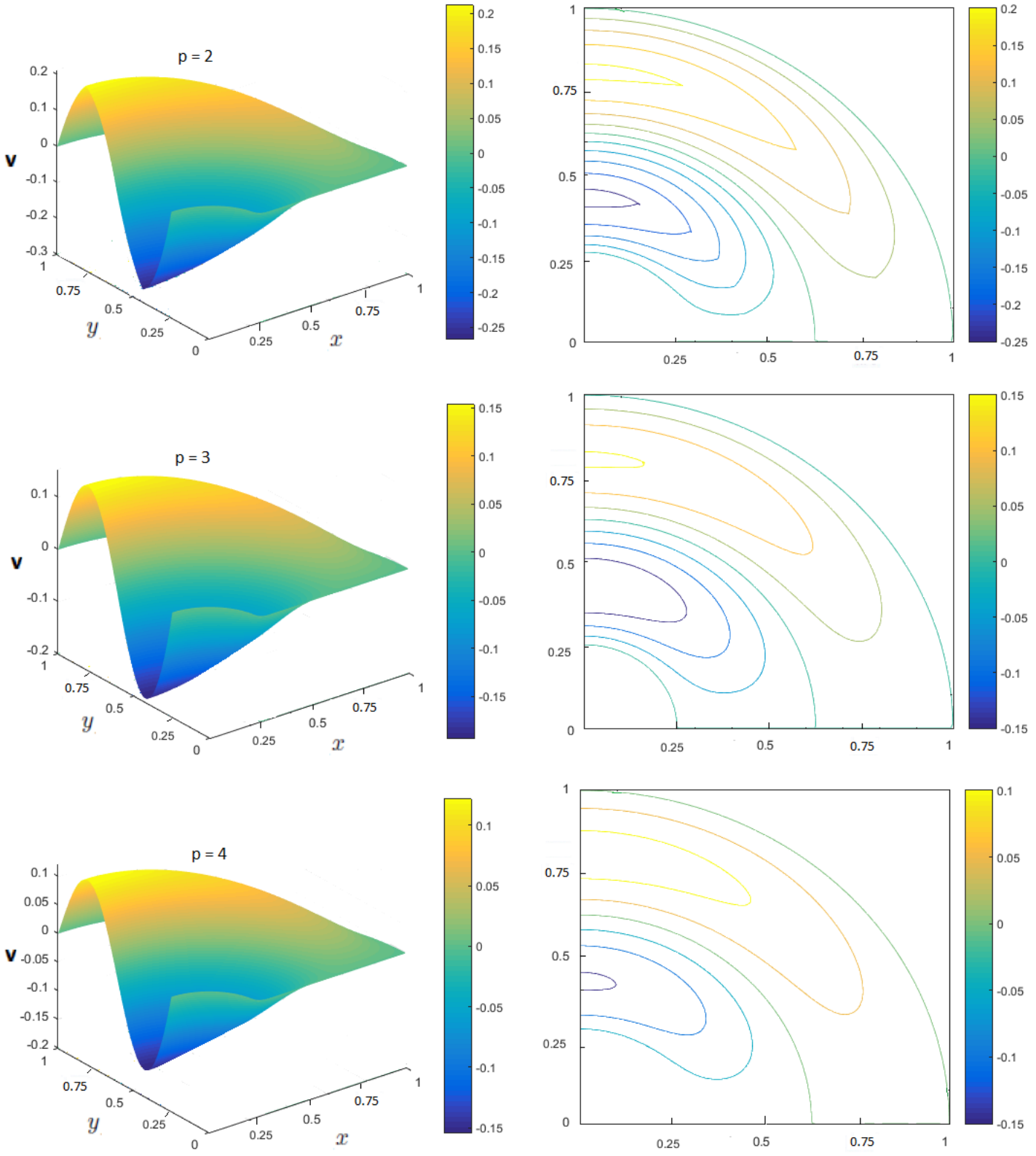


Figure 9.17: IGDG solution v for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

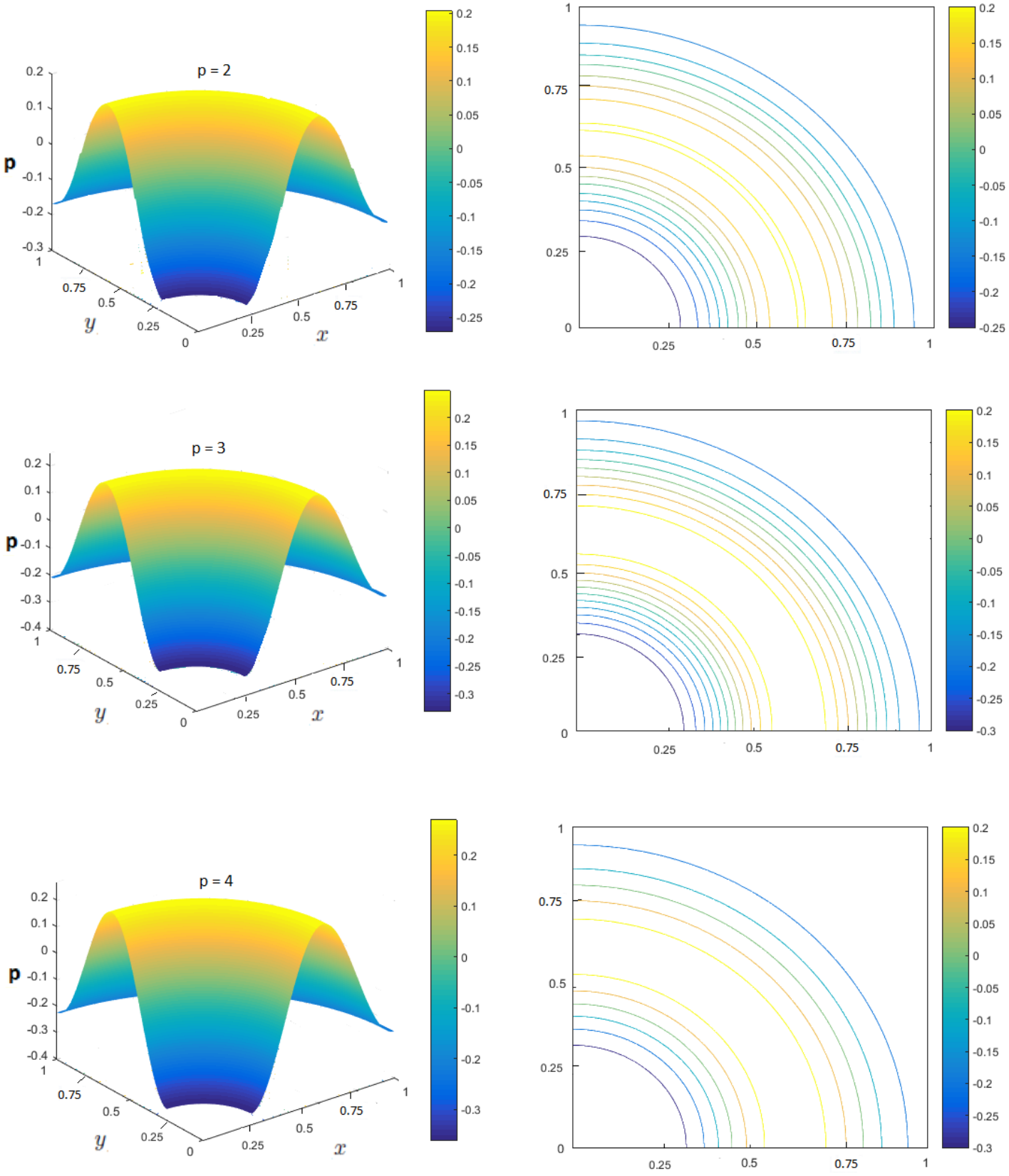


Figure 9.18: IGDG solution p for different degrees p for $\mathfrak{N}_{el} = 4 \times 4$ elements at $T = 0.1s$.

Tables 9.3 and 9.4 summarize the convergence results in the L^2-norm for the bivariate quadratic, cubic and quartic Bernstein basis functions.

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.983E-01$	–
$\frac{\mathfrak{h}}{2}$	$4.541E-02$	2.13
$\frac{\mathfrak{h}}{4}$	$3.436E-03$	3.72
$\frac{\mathfrak{h}}{8}$	$4.066E-04$	3.07
$\frac{\mathfrak{h}}{16}$	$5.084E-05$	2.99

Table 9.3: L^2-error for the $2D$ acoustic problem and convergence order for the IGDG method for the quadratic Bernstein bases in conjunction with $RK4$ time discretisation.

Mesh	L^2-error	rate
$\mathfrak{h}$	$9.907E-02$	–
$\frac{\mathfrak{h}}{2}$	$2.740E-03$	5.17
$\frac{\mathfrak{h}}{4}$	$3.173E-04$	3.11
$\frac{\mathfrak{h}}{8}$	$2.019E-05$	3.97
$\frac{\mathfrak{h}}{16}$	$1.310E-06$	3.94

Mesh	L^2-error	rate
$\mathfrak{h}$	$1.527E-02$	–
$\frac{\mathfrak{h}}{2}$	$1.046E-03$	3.86
$\frac{\mathfrak{h}}{4}$	$2.875E-05$	5.18
$\frac{\mathfrak{h}}{8}$	$9.704E-07$	4.88
$\frac{\mathfrak{h}}{16}$	$3.056E-08$	4.98

Table 9.4: L^2-error for the $2D$ acoustic problem and convergence order for the IGDG method for the cubic (left) and quartic (right) Bernstein bases in conjunction with $RK4$ time discretisation.

The corresponding convergence data for uniform and refinement $\mathfrak{h}$, $\frac{\mathfrak{h}}{2}$, $\frac{\mathfrak{h}}{4}$, $\frac{\mathfrak{h}}{8}$ and $\frac{\mathfrak{h}}{16}$ are shown in Fig. 9.19.

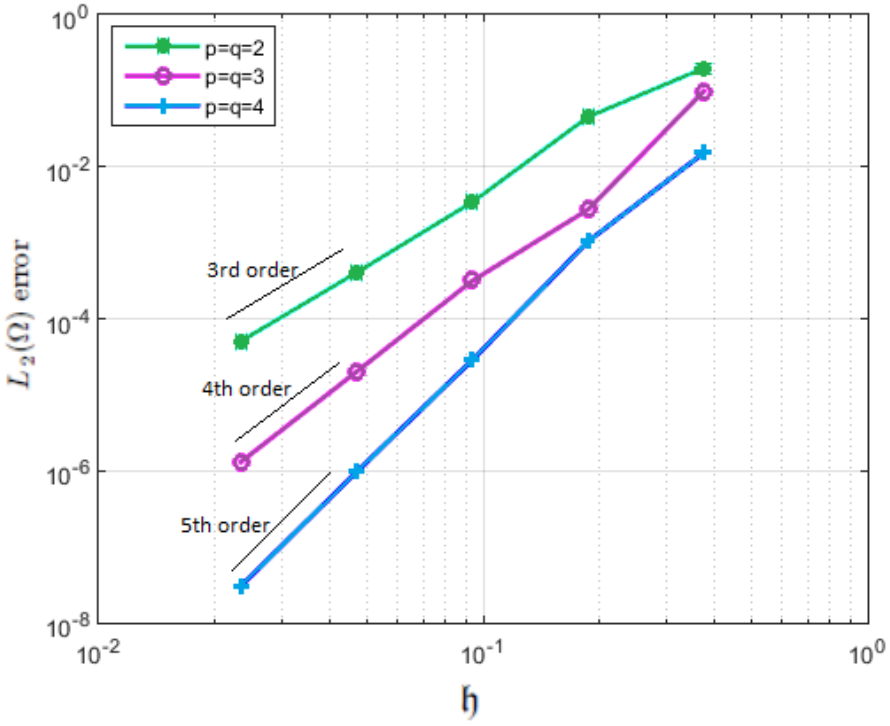


Figure 9.19: L^2-error for the $2D$ acoustic problem using the IGDG method in conjunction with $RK4$.

The impact of the boundary curvature on the solution accuracy, already observed in the solution plots, is confirmed when the convergence rates are computed. For the isogeometric approach using curvilinear grids, quasi-optimal convergence rates close to order $p+1$ are obtained as the B-spline grids are refined. However, when using rectilinear grids, a far lower convergence rate is observed, close to second-order (globally), whatever the degree of the solution employed (see Figure 9.12). One can even notice that the accuracy is degraded as the degree of the solution increases, for a fixed number of degrees of freedom. The solver tends to a solution corresponding to the uncorrect boundary geometry (location and normals). Therefore, the use of an accurate geometry description, exact if possible, is mandatory for the efficiency of high-order discretization schemes

9.6 Conclusion

In this chapter we gave a description of the ongoing development of a IGDG method to solve the first-order acoustic wave equation in $2D$, expressed as a linear hyperbolic PDE system. As it has been pointed out previously, the computational domain is divided into non-overlapping sub-domains, composed of B-spline patches. The DG approach was applied on element level, each element being a Bézier patch constructed from initial B-spline patches. The solution of the problem is approximated in every element without any continuity requirements for the discrete solution on the interfaces. As a result, IGDG method can be easily formulated and implemented. Finally, the numerical behavior of the method is evaluated and it has shown an optimal convergence rate.

CONCLUSION

Dans le chapitre 7, nous avons décrit les différentes procédures de calcul pour la méthode de GD dans le cadre de l'AIG, spécialement pour le cas du problème hyperbolique unidimensionnel. Le présent chapitre et le chapitre précédent sont consacrés à l'extension au cas des problèmes hyperboliques bidimensionnels. Nous avons décrit le développement d'une méthode de GD dans le cadre AIG pour résoudre l'équation d'advection en $2D$ dans le chapitre précédent et le problème acoustiques de premier ordre en $2D$ dans le présent chapitre, qui est exprimé sous la forme d'un système EDP linéaire hyperbolique. Comme cela a été souligné précédemment, le domaine de calcul est divisé en sous-domaines, composés de patches B-splines. L'approche GD a été appliquée au niveau des éléments, chaque élément étant un patch de Bézier construit à partir de patches B-splines initiaux. La solution du problème est approximée dans chaque élément sans aucune exigence de continuité pour la solution discrète sur les interfaces. Par conséquent, la méthode de GD dans le cadre AIG peut être facilement formulée et mise en oeuvre, elle est capable de gérer facilement des géométries complexes. La flexibilité de la méthode pour gérer différentes géométries et pour travailler avec différents éléments a été montrée. Ainsi, le comportement numérique de la méthode est évalué et il a été montré un taux de convergence optimal selon la norme L^2.

Part V

General Conclusion & Perspectives

General Conclusion & Perspectives

Isogeometric Analysis can be considered as a Finite Element method that generalizes the set of basis functions from polynomials to B-splines or more generally to non-uniform-rational B-splines. As shown in the literature, this choice guarantees several advantages, from the exact parametrization of geometries defined via CAD to a higher accuracy per-degrees-of-freedom. It also allows for solution fields with higher smoothness. These reasons have made IGA a successful topic in recent years.

In this thesis we applied the IGA method to some hyperbolic problems. We considered standard Galerkin methods as well as stabilized methods, with a special emphasis on B-splines. IGA possesses a set of attractive features from the view point of accuracy and implementational convenience not present in standard FE discretizations. In fact, IGA allows exact representation of a wide class of geometries even on very coarse meshes. In particular, geometry domains having conic sections like circles, cylinders, spheres, ellipsoids, etc... can be represented exactly using NURBS. Refinements can be performed by subdivision of the grid (by inserting knots) or by elevation of the polynomial order of the basis functions in the same way as using the traditional finite element method.

Due to the fact that the aim of IGA is to generalize and improve upon classical FEA, we started this thesis by giving an introduction to IGA by revisiting the original analysis, i.e. FEA, in the context of hyperbolic PDE with SUPG stabilization.

Numerical examples for advection problem were given for both classical Lagrange and B-spline bases. It was shown that the IG method may be well suited to the hyperbolic problems if stabilization is correctly calibrated. The convergence rates were measured in the L^2-norm. The IG method was tested on different polynomial orders. It was found that, for a given polynomial order, high regularity gives lower error versus degrees of freedom compared to low regularity, without exceptions. Elsewhere, a possible drawback of the SUPG method is the sensitivity of the solution to the stabilization parameter, whose optimal value is not determined precisely by the available theory and is tedious to select in practice.

Therefore, we proposed in this thesis a new method which combines the IGA with the DG method, called IGDG method, for solving hyperbolic problems. The major reason for using DG methods in this thesis lies in their ability to provide stable numerical methods for hyperbolic problems, for which classical FEM is well known to perform poorly.

Our method takes advantage of both IGA and DG methods. In fact, DG methodology is adopted at Bézier patch level, i.e., we employ the traditional IGA within each Bézier patch, and employ the DG method across the patch interfaces to glue the multiple patches. Bézier patches, considered as elements, are constructed by transformation of the initial B-spline domain (Bézier extraction is a classical CAD technique).

We consider then scalar and system of conservation laws, as test cases. These test examples show that the resulting IGDG method in conjunction with RK method is stable, high-order accurate, and can easily handle curved geometries and boundary conditions. As consequence, this method can be easily formulated and implemented. Then, the numerical behavior of the method has been evaluated and it was shown an optimal convergence rate for $2D$ advection and acoustic problems. The higher accuracy obtained by using curvilinear elements has been demonstrated, in particular when coarse grids are employed. An accuracy up to order five has been reached.

The present method has shown to be very effective for linear hyperbolic problems and systems in one and two dimensions. Now, the proposed IGDG method should be extended to more complex non-linear conservation laws, like compressible Euler equations for instance. Efficient DG methods for such non-linear systems have been constructed recently, for classical rectilinear grids. Accounting for Bézier bases should not exhibit specific difficulties, except for the capture of solution discontinuities that may appear for non-linear conservation laws. In this perspective, one could extend the generalized limiters proposed by Cockburn and coauthors (see appendix) to local Bézier representations. Another necessary extension would concern the construction of the initial B-spline or NURBS computational domain. The geometries considered in this work are rather simple and are represented by only one B-spline patch, before Bézier extraction. To consider really complex geometries, a general framework to easily handle a set of B-spline patches should be proposed. In this perspective, we underline that the proposed approach, based on DG concepts, is more flexible that the original IGA method because no regularity constraint is necessary at the interface between patches. Finally, one should exploit the great flexibility offered by B-spline representations in terms of $p-$ and $h-$ refinements in order to propose automated local refinement strategies.

CONCLUSION GÉNÉRALE & PERSPECTIVES

L'analyse isogéométrique peut être considérée comme une méthode par éléments finis qui généralise l'ensemble des fonctions de base pôlynomiales aux B-splines ou plus généralement aux NURBS. Comme le montre la littérature, ce choix garantit plusieurs avantages de la paramétrisation exacte des géométries définies par CAO à une plus grande précision par degré de liberté. Il permet également des champs de solution réguliers. Ces raisons ont fait de l'AIG un sujet émergeant ces dernières années.

Dans cette thèse, nous avons appliqué la méthode de l'AIG à certains problèmes hyperboliques. Nous avons considéré les méthodes standards de Galerkin ainsi que les méthodes stabilisées, en mettant particulièrement l'accent sur les B-splines. L'AIG possède un ensemble de caractéristiques attractives du point de vue de la précision et de la commodité de mise en oeuvre qui ne sont pas présentes dans les discrétisations d'EF standards. En fait, l'AIG permet une représentation exacte d'une large classe de géométries, même sur des maillages très grossiers. En particulier, les domaines géométriques ayant des sections coniques comme les cercles, les cylindres, les sphères, les ellipsoides, etc. peuvent être représentés exactement à l'aide de NURBS. Les raffinements peuvent être effectués par subdivision de la grille (en insérant des noeuds) ou par élévation de l'ordre polynomial des fonctions de base de la même manière que la méthode des EF traditionnels.

Vu que l'objectif principal de l'AIG est de généraliser et d'améliorer l'AEF classique, nous avons commencé cette thèse en revisitant l'analyse originale, à savoir l'AEF, dans le contexte d'EDP hyperboliques avec la stabilisation SUPG.

Des exemples numériques de problèmes d'advection ont été donnés pour les bases classiques de Lagrange et B-splines. Il a été montré que la méthode d'AIG peut être bien adaptée aux problèmes hyperboliques si la stabilisation est correctement calibrée. Les taux de convergence ont été mesurés dans la norme standard L^2. La méthode d'AIG a été testée sur différents ordres polynomiaux. On a constaté que, pour un ordre polynomial donné, une régularité élevée donne une erreur plus faible par degré de liberté, par rapport à une faible régularité, sans exception.

Par ailleurs, un inconvénient possible de la méthode SUPG est la sensibilité de la solution au paramètre de stabilisation, dont la valeur optimale n'est pas déterminée précisément par la théorie disponible et qui est fastidieuse à sélectionner dans la pratique.

Par conséquent, nous avons proposé dans cette thèse une nouvelle méthode qui combine l'AIG avec la méthode de GD, appelée méthode IGDG, pour résoudre des problèmes hyperboliques. La principale raison de l'utilisation des méthodes de GD dans cette thèse réside dans leur capacité à fournir des méthodes numériques stables pour les problèmes hyperboliques, pour lesquels la MEF classique est bien connue pour ses performances médiocres.

Notre méthode tire parti des méthodes de l'AIG et de GD. En fait, la méthodologie de GD est adoptée au niveau des patches de Bézier, c'est-à-dire que nous utilisons l'AIG traditionnelle dans chaque patch de Bézier et la méthode de GD sur les interfaces de patch pour assembler les multiples patches. Les patches de Bézier, considérés comme des éléments, sont construits par transformation du domaine B-spline initial (l'extraction de Bézier est une technique classique de CAO).

Nous considérons alors des cas scalaires et des cas systèmes de lois de conservation, comme cas tests. Ces exemples de tests montrent que la méthode IGDG obtenue en conjonction avec la méthode de RK pour la discrétisation temporelle est stable, de haute précision, et peut facilement gérer les géométries courbes et les conditions aux limites. Par conséquent, cette méthode peut être facilement formulée et mise en oeuvre. Ainsi, le comportement numérique de la méthode a été évalué et on a montré un taux de convergence optimal pour l'advection $2D$ et les problèmes acoustiques. La plus grande précision obtenue en utilisant des éléments curvilignes a été démontrée, en particulier lorsque des grilles grossières sont utilisées. Une précision de l'ordre de cinq a été atteinte.

La méthode actuelle s'est avérée très efficace pour les problèmes hyperboliques linéaires et les systèmes à une et deux dimensions. Maintenant, la méthode IGDG proposée devrait être étendue à des lois de conservation non linéaires plus complexes, comme les équations d'Euler compressible, par exemple. Des méthodes de GD efficaces pour ces systèmes non linéaires ont été construites récemment pour les grilles rectilignes classiques. L'extension à des bases de Bézier ne devrait pas présenter de difficultés spécifiques, sauf pour la capture des discontinuités de solution pouvant apparaître pour les lois de conservation non linéaires. Dans cette perspective, on pourrait étendre les limiteurs généralisés proposés par Cockburn et ses coauteurs (voir annexe) aux représentations de Bézier locales. Une autre extension nécessaire concernerait la construction du domaine de calcul initial B-spline. Les géométries considérées dans ce travail sont plutôt simples et représentées par un seul patch B-spline, avant l'extraction de Bézier. Pour prendre en compte des géométries plus complexes, il convient de proposer un cadre général permettant de gérer facilement un ensemble de patches B-spline. Dans cette perspective, nous soulignons que l'approche proposée, basée sur les concepts GD, est plus flexible que la méthode d'AIG d'origine car aucune contrainte de régularité n'est nécessaire à l'interface entre les patches. Enfin, il convient d'exploiter la grande flexibilité offerte par les représentations B-splines en termes de raffinement $p-$ et $h-$ afin de proposer des stratégies de raffinement locales automatisées.

APPENDIX

A.1 Gaussian quadrature

Let $\Omega = [a,b] \subset \mathbb{R}$. A quadrature rule is defined by a set of nodes $(X^G(k))_{1\leq k\leq n_G} \in \Omega$ and a set of weights $(\omega^G(k))_{1\leq k\leq n_G} \in \mathbb{R}$. The fundamental result of Gaussian quadrature states that the optimal abscissas of the n_G-point Gaussian quadrature formulas are precisely the roots of the orthogonal polynomial for the same interval and weighting function. Gaussian quadrature is optimal because it integrates all polynomials up to degree $2n_G - 1$ exactly.

When $\Omega = [-1,1]$, the roots of the Legendre polynomials and their corresponding weights have been extensively tabulated, so we can simply use these tables without redoing the calculations.

n_G	Weights ω^G	Nodes X^G
1	2	0
2	$-\frac{1}{\sqrt{3}}, \quad \frac{1}{\sqrt{3}}$	$1, \quad 1$
3	$-\sqrt{\frac{3}{5}}, \quad 0, \quad \sqrt{\frac{3}{5}}$	$\frac{5}{9}, \quad \frac{8}{9}, \quad \frac{5}{9}$
4	$-\sqrt{\frac{(15+2\sqrt{30})}{35}}, \quad -\sqrt{\frac{(15-2\sqrt{30})}{35}}, \quad \sqrt{\frac{(15+2\sqrt{30})}{35}}, \quad -\sqrt{\frac{(15-2\sqrt{30})}{35}}$	$\frac{18-\sqrt{30}}{36}, \quad \frac{18+\sqrt{30}}{36}, \quad \frac{18+\sqrt{30}}{36}, \quad \frac{18-\sqrt{30}}{36}$

Table A.1: Gauss–Legendre nodes and coefficients

We can transform any given integral on the interval $[a,b]$ into an integral on the interval $[-1,1]$, simply use:

$$\int_a^b f(x)dx = \Big(\frac{b-a}{2}\Big)\int_{-1}^{1} f\Big(\frac{(b-a)y+(b+a)}{2}\Big)dy \approx \Big(\frac{b-a}{2}\Big)\sum_{k=1}^{n_G}\omega^G(k)f\Big(\frac{(b-a)X^G(k)+(b+a)}{2}\Big)$$

APPENDIX B

B.1 Runge-Kutta (RK) method:

The Runge-Kutta method is used to solve a system of ODEs given by:

$$\partial_t u(x,t) = \mathfrak{L}\big(t, u(x,t)\big). \tag{B.1}$$

The s-stage Runge-Kutta (RK) method for (B.1) is written in the form:

$$\begin{aligned}
\partial_t u^{(1)} &= \mathfrak{L}\big(t^{\check{n}}, u^{\check{n}}\big) \\
\partial_t u^{(i)} &= \mathfrak{L}\Big(t^{\check{n}} + c_i \Delta t, u^{\check{n}} + \Delta t \sum_{j=1}^{i-1} a_{ij} \partial_t h^{(j)}\Big) \qquad i = 2, ..., s
\end{aligned}$$

$$u^{\check{n}+1} = u^{\check{n}} + \Delta t \sum_{i=1}^{s} b_i \partial_t u^{(i)},$$

where $\Delta t = t^{\check{n}+1} - t^{\check{n}}$ is the time step, and $u^{\check{n}}$ and $u^{\check{n}+1}$ represent the values of u at time $t^{\check{n}}$ and $t^{\check{n}+1}$ respectively. The coefficients a_{ij}, b_i and c_i can be summarized in matrix/vector form by the Butcher tableau.
We have $a_{ij} = 0 \quad \forall\, j \geq i$. Moreover, the coefficients c_i and a_{ij} are connected by the condition:

$$c_i = \sum_{j=1}^{s} a_{ij} \qquad i = 1, ..., s.$$

B.2 1D slope limiting

If the solution of the problem exhibits discontinuities, the proposed scheme generates oscillations (Gibbs phenomenon). To overcome this difficulty, a particular treatment is required, such as filtering or limiting. In the present work, we envisage to apply the generalized limiting approach. The idea is to modify locally the solution, if a discontinuity is detected, in order to satisfy the following conditions:

i) Maintain the mass conservation principle in each element.

ii) Satisfy the Total Variation Diminishing in the Means (TVDM) property.

iii) Do not degrade accuracy of the method.

B.2.1 TVDM limiter

The proposed approach, relies on the minmod function defined as:

$$m(s_1,...,s_n) = \begin{cases} s\big(\min_{i=1,...,n} \mid s_i \mid\big) & \text{if} \quad s = sign(s_1) = ... = sign(s_n) \\ 0 & \text{otherwise} \end{cases} \tag{B.2}$$

Let $\overline{u}_j$ denote the mean value of the solution over the element Ω_j. Thanks to the partition of unity property of the Bézier representation, it can be simply evaluated by:

$$\overline{u}_j = \frac{1}{p+1} \sum_{i=1}^{p+1} u_j^{i\,(\check{n})}.$$

The minmod function is employed to extrapolate linearly the mean value $\overline{u}_j$ at the element extremities, according to a limited slope estimation, using:

$$u_j^l = \overline{u}_j - m\big(\overline{u}_j - u_j^{1\,(\check{n})}, \overline{u}_{j+1} - \overline{u}_j, \overline{u}_j - \overline{u}_{j-1}\big), \tag{B.3}$$

$$u_j^r = \overline{u}_j + m\big(u_j^{p+1\,(\check{n})} - \overline{u}_j, \overline{u}_{j+1} - \overline{u}_j, \overline{u}_j - \overline{u}_{j-1}\big). \tag{B.4}$$

If $u_j^l = u_j^{1\,(\check{n})}$ and $u_j^r = u_j^{p+1\,(\check{n})}$, i.e. the local slope (first argument of the minmod function) is lower than the slopes based on neighbors values (second and third arguments), and all slopes have the same sign, the solution remains unchanged, because this indicates a region where the solution is regular. On the contrary, a possible discontinuity is detected and the solution in the element Ω_j is replaced by a linear approximation with the limited slope:

$$u_{h|_{\Omega_j}}(x) = \overline{u}_j + \big(\frac{x - \overline{x}_j}{\mathfrak{h}/2}\big) \min(m_l, m_r),$$

where m_l and m_r denote the limited slopes computed in Eq.(B.3) and Eq.(B.4). Using a local Bernstein basis, such a linear representation can be obtained easily be setting a linear variation of the degrees of freedom:

$$u_j^{i\,(\check{n})} = \overline{u}_j + \big(\frac{2(i-1)}{p} - 1\big) \min(m_l, m_r) \qquad i = 1,...,p+1.$$

B.2.2 TVBM limiter

The TVDM generalized slope limiter described above yields a loss of accuracy in the vicinity of local extrema of the solution. To prevent this effect, one must construct a Total Variation Bounded in the Means (TVBM) limiter, instead of the TVDM one.
This can easily be achieved by introducing a modified minmod function (less restrictive):

$$\widetilde{m}(s_1,...,s_n) = \begin{cases} s_1 & \text{if } \mid s_1 \mid \leq C\mathfrak{h}^2, \\ m(s_1,...,s_n) & \text{otherwise,} \end{cases}$$

which is used in replacement of Eq. (B.2) to compute limited slopes. C should be an upper bound of the absolute value of the second order derivative of the solution at local extrema.

APPENDIX C

C.1 Bessel functions zeros

The first few roots $j_{v,k}$ of the Bessel functions $J_v(x)$ are given in the following table for small nonnegative integer values of v and k [68] [89]:

k	$J_0(x)$	$J_1(x)$	$J_2(x)$	$J_3(x)$	$J_4(x)$	$J_5(x)$
1	2.4048	3.8317	5.1356	6.3802	7.5883	8.7715
2	5.5201	7.0156	8.4172	9.7610	11.0647	12.3386
3	8.6537	10.1735	11.6198	13.0152	14.3725	15.7002
4	11.7915	13.3237	14.7960	16.2235	17.6160	18.9801
5	14.9309	16.4706	17.9598	19.4094	20.8269	22.2178

The first few roots $j'_{v,k}$ of the derivative of the Bessel function $J'_v(x)$ are given in the following table for small nonnegative integer values of v and k [89]:

k	$J'_0(x)$	$J'_1(x)$	$J'_2(x)$	$J'_3(x)$	$J'_4(x)$	$J'_5(x)$
1	3.8317	1.8412	3.0542	4.2012	5.3175	6.4156
2	7.0156	5.3314	6.7061	8.0152	9.2824	10.5199
3	10.1735	8.5363	9.9695	11.3459	12.6819	13.9872
4	13.3237	11.7060	13.1704	14.5858	15.9641	17.3128
5	16.4706	14.8636	16.3475	17.7887	19.1960	20.5755

BIBLIOGRAPHY

[1] J. E. Akin and T. E. Tezduyar. Calculation of the advective limit of the SUPG stabilization parameter for linear and higher-order elements. *Computer Methods in Applied Mechanics and Engineering*, 193(21-22):1909–1922, 2004.

[2] I. Akkerman, Y. Bazilevs, V. Calo, T. Hughes, and S. Hulshoff. The role of continuity in residual-based variational multiscale modeling of turbulence. *Computational Mechanics*, 41(3):371–378, 2008.

[3] D. Al-Akhrass. *Méthodes éléments finis mixtes robustes pour gérer l'incompressibilité en grandes déformations dans un cadre industriel.* PhD thesis, Ecole Nationale Supérieure des Mines de Saint-Etienne, 2014.

[4] A. B. H. Ali and A. Soulaimani. An unstructured finite elements method for solving the compressible RANS equations and the Spalart-Allmaras turbulence model. *Computer Methods in Applied Mechanics and Engineering*, 199(33-36):2261–2272, 2010.

[5] D. N. Arnold, F. Brezzi, B. Cockburn, and L. D. Marini. Unified analysis of discontinuous Galerkin methods for elliptic problems. *SIAM Journal on Numerical Analysis*, 39(5):1749–1779, 2002.

[6] C. Baiocchi, F. Brezzi, and L. P. Franca. Virtual bubbles and Galerkin-least-squares type methods (GLS). *Computer Methods in Applied Mechanics and Engineering*, 105(1):125–141, 1993.

[7] Y. Bazilevs, K. Takizawa, and T. E. Tezduyar. *Computational fluid-structure interaction: methods and applications.* John Wiley & Sons, 2013.

[8] P. Bézier. *Essai de définition numérique des courbes et des surfaces expérimentales: contribution à l'étude des propriétés des courbes et des surfaces paramétriques polynomiales à coefficients vectoriels.* PhD thesis, Université Pierre et Marie Curie (Paris *V I*), 1977.

[9] M. Billaud. *Eléments finis stabilisés pour des écoulements diphasiques compressible-incompressible.* PhD thesis, Université de Bordeaux 1, 2009.

[10] C. F. Borges and T. Pastva. Total least squares fitting of Bézier and B-spline curves to ordered data. *Computer Aided Geometric Design*, 19(4):275–289, 2002.

[11] S. Brenner and R. Scott. *The mathematical theory of finite element methods*, volume 15. Springer Science & Business Media, 2007.

[12] F. Brezzi. On the existence, uniqueness and approximation of saddle-point problems arising from lagrangian multipliers. *Revue Française d'Automatique, Informatique, Recherche Opérationnelle. Analyse Numérique*, 8:129–151, 1974.

[13] A. N. Brooks. *A Petrov-Galerkin finite element formulation for convection dominated flows*. PhD thesis, California Institute of Technology, 1981.

[14] A. N. Brooks and T. J. 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.

[15] H. Carbonel, A. Carlos, A. C. Galeão, and A. D. Loula. Numerical study of Petrov-Galerkin formulations for the shallow water wave equations. *Journal of the Brazilian Society of Mechanical Sciences*, 22(2):231–247, 2000.

[16] T. H. Chien. *Development of isogeometric finite element methods*. PhD thesis, Vietnam National University - Faculty of Mathematics and Computer Science Department of Mechanics, 2015.

[17] I. Christie, D. F. Griffiths, A. R. Mitchell, and O. C. Zienkiewicz. Finite element methods for second order differential equations with significant first derivatives. *International Journal for Numerical Methods in Engineering*, 10(6):1389–1396, 1976.

[18] B. Cockburn, S. Hou, and C.-W. Shu. The Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws. IV. The multidimensional case. *Mathematics of Computation*, 54(190):545–581, 1990.

[19] B. Cockburn, S.-Y. Lin, and C.-W. Shu. TVB Runge-Kutta local projection discontinuous Galerkin finite element method for conservation laws III: One-dimensional systems. *Journal of Computational Physics*, 84(1):90–113, 1989.

[20] B. Cockburn and C.-W. Shu. The Runge-Kutta local projection P^1-discontinuous-Galerkin finite element method for scalar conservation laws. *Mathematical Modelling and Numerical Analysis*, 25(3):337–361, 1991.

[21] B. Cockburn and C.-W. Shu. The Runge-Kutta discontinuous Galerkin method for conservation laws v: multidimensional systems. *Journal of Computational Physics*, 141(2):199–224, 1998.

[22] B. Cockburn and C.-W. Shu. Runge-Kutta discontinuous Galerkin methods for convection-dominated problems. *Journal of Scientific Computing*, 16(3):173–261, 2001.

[23] J. A. Cottrell, T. J. Hughes, and Y. Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

[24] J. A. Cottrell, A. Reali, Y. Bazilevs, and T. J. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195(41-43):5257–5296, 2006.

[25] G. Dattoli and A. Torre. Theory and applications of generalized Bessel functions. Aracne Rome, 1996.

[26] K. David. *Multi-patch Discontinuous Galerkin isogeometric analysis for porous media flow*. PhD thesis, 2017.

[27] C. De Boor. On calculating with B-splines. *Journal of Approximation Theory*, 6(1):50–62, 1972.

[28] D. A. Di Pietro and A. Ern. *Mathematical aspects of discontinuous Galerkin methods*, volume 69. Springer Science & Business Media, 2011.

[29] M. S. Floater. Derivatives of rational Bézier curves. *Computer Aided Geometric Design*, 9(3):161–174, 1992.

[30] L. P. Franca, S. L. Frey, and T. J. Hughes. Stabilized finite element methods: *I*. Application to the advective-diffusive model. *Computer Methods in Applied Mechanics and Engineering*, 95(2):253–276, 1992.

[31] L. P. Franca, G. Hauke, and A. Masud. Revisiting stabilized finite element methods for the advective-diffusive equation. *Computer Methods in Applied Mechanics and Engineering*, 195(13-16):1560–1572, 2006.

[32] T.-P. Fries and H. G. Matthies. A review of Petrov-Galerkin stabilization approaches and an extension to meshfree methods. *Technische Universitat Braunschweig, Brunswick*, 2004.

[33] A. Gdhami, R. Duvigneau, and M. Moakher. Méthode de Galerkin discontinue: Cas de l'analyse isogéométrique. In *TAM-TAM 2017-Tendances dans les Applications Mathématiques en Tunisie, Algérie et Maroc*, 2017.

[34] S. Gottlieb and C.-W. Shu. Total variation diminishing Runge-Kutta schemes. *Mathematics of Computation of the American Mathematical Society*, 67(221):73–85, 1998.

[35] D. Griffiths and J. Lorenz. An analysis of the Petrov—Galerkin finite element method. *Computer Methods in Applied Mechanics and Engineering*, 14(1):39–64, 1978.

[36] J.-L. Guermond and R. Pasquetti. A correction technique for the dispersive effects of mass lumping for transport problems. *Computer Methods in Applied Mechanics and Engineering*, 253:186–198, 2013.

[37] S. Hall, M. Eaton, and M. Williams. The application of isogeometric analysis to the neutron diffusion equation for a pincell problem with an analytic benchmark. *Annals of Nuclear Energy*, 49:160–169, 2012.

[38] J. Heinrich, P. Huyakorn, O. Zienkiewicz, and A. Mitchell. An "upwind" finite element scheme for two-dimensional convective transport equation. *International Journal for Numerical Methods in Engineering*, 11(1):131–143, 1977.

[39] J. S. Hesthaven and T. Warburton. *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*. Springer Science & Business Media, 2007.

[40] C. Hofer and U. Langer. Dual-primal isogeometric tearing and interconnecting solvers for large-scale systems of multipatch continuous Galerkin IgA equations. *ArXiv preprint arXiv:1511.07183*, 2015.

[41] T. Hughes and T. Tezduyar. Development of time-accurate finite element techniques for first order hyperbolic systems with emphasys on the compressible Euler equations. *Computer Methods in Applied Mechanics and Engineering*, 45(1-3):217–284, 1984.

[42] T. J. Hughes. A simple scheme for developing 'upwind' finite elements. *International Journal for Numerical Methods in Engineering*, 12(9):1359–1365, 1978.

[43] T. J. 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.

[44] T. J. Hughes, L. P. Franca, and M. Balestra. A new finite element formulation for computational fluid dynamics: v. Circumventing the Babuška-Brezzi condition: A stable Petrov-Galerkin formulation of the Stokes problem accommodating equal-order interpolations. *Computer Methods in Applied Mechanics and Engineering*, 59(1):85–99, 1986.

[45] T. J. Hughes and T. Tezduyar. Finite element methods for first-order hyperbolic systems with particular emphasis on the compressible Euler equations. *Computer Methods in Applied Mechanics and Engineering*, 45(1-3):217–284, 1984.

[46] N. Jaxon and X. Qian. Isogeometric analysis on triangulations. *Computer-Aided Design*, 46:45–57, 2014.

[47] 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.

[48] V. John and P. Knobloch. On spurious oscillations at layers diminishing (SOLD) methods for convection-diffusion equations: Part II-Analysis for P^1 and Q^1 finite elements. *Computer Methods in Applied Mechanics and Engineering*, 197(21-24):1997–2014, 2008.

[49] C. Johnson, U. Nävert, and J. Pitkäranta. Finite element methods for linear hyperbolic problems. *Computer Methods in Applied Mechanics and Engineering*, 45(1-3):285–312, 1984.

[50] R. Kelisky and T. Rivlin. Iterates of Bérnstein polynomials. *Pacific Journal of Mathematics*, 21(3):511–520, 1967.

[51] D. Kelly, S. Nakazawa, O. Zienkiewicz, and J. Heinrich. A note on upwinding and anisotropic balancing dissipation in finite element approximations to convective diffusion problems. *International Journal for Numerical Methods in Engineering*, 15(11):1705–1711, 1980.

[52] T. Krarup. A contribution to the mathematical foundation of physical geodesy. *Geod. Inst. Copenhagen, Medd., No. 44, 80 p.*, 44, 1969.

[53] P. Lancaster and K. Salkauskas. Surfaces generated by moving least squares methods. *Mathematics of Computation*, 37(155):141–158, 1981.

[54] U. Langer and I. Toulopoulos. Analysis of multipatch discontinuous Galerkin IgA approximations to elliptic boundary value problems. *Computing and Visualization in Science*, 17(5):217–233, 2015.

[55] B.-G. Lee and Y. Park. Approximate conversion of rational Bézier curves. *J. KSIAM*, 2:88–93, 1998.

[56] R. J. LeVeque. *Finite volume methods for hyperbolic problems*, volume 31. Cambridge University Press, 2002.

[57] K. J. Liew, A. Ramli, and A. A. Majid. B-spline surface fitting on scattered points. *Applied Mathematics & Information Sciences*, 10(1):273, 2016.

[58] J. Liou and T. E. Tezduyar. Finite element solution techniques for large-scale problems in computational fluid dynamics. Technical, Department of Mechanical Engineering University of Houston, Houston, TX 77004, 1987.

[59] H. Liu and K. Xu. A Runge-Kutta discontinuous Galerkin method for viscous flow equations. *Journal of Computational Physics*, 224(2):1223–1242, 2007.

[60] S. May. *Splines for damage and fracture in solids*. PhD thesis, University of Glasgow, 2016.

[61] C. Michoski, J. Chan, L. Engvall, and J. A. Evans. Foundations of the blended isogeometric discontinuous Galerkin (BIDG) method. *Computer Methods in Applied Mechanics and Engineering*, 305:658–681, 2016.

[62] S. J. Miller. The method of least squares. *Mathematics Department Brown University*, 8:1–7, 2006.

[63] A. Mizukami and T. J. Hughes. A Petrov-Galerkin finite element method for convection-dominated flows: an accurate upwinding technique for satisfying the maximum principle. *Computer Methods in Applied Mechanics and Engineering*, 50(2):181–193, 1985.

[64] K. W. Morton and D. F. Mayers. *Numerical solution of partial differential equations: an introduction*. Cambridge University Press, 2005.

[65] E. Nava-Yazdani and K. Polthier. De Casteljaus algorithm on manifolds. *Computer Aided Geometric Design*, 30(7):722–732, 2013.

[66] T. N. Nguyen. Isogeometric Finite Element Analysis based on Bézier Extraction of NURBS and T-Splines. Master's thesis, NTNU- Norwegian University of Science and Technology, 2011.

[67] K. Nguyen Tan. *Surfaces polyédriques et surfaces paramétriques: une reconstruction par approximation via les surfaces de subdivision*. PhD thesis, Aix-Marseille 2, 2010.

[68] J. Niedziela. Bessel functions and their applications. *University of Tennessee-Knoxville*, 2008.

[69] A. Owens, J. Welch, J. Kópházi, and M. D. Eaton. Discontinuous isogeometric analysis methods for the first-order form of the neutron transport equation with discrete ordinate (SN) angular discretisation. *Journal of Computational Physics*, 315:501–535, 2016.

[70] Y. Park and N. Lee. Application of degree reduction of polynomial Bézier curves to rational case. *Journal of Applied Mathematics and Computing*, 18(1-2):159, 2005.

[71] J. Proft and B. Rivière. Analytical and numerical study of diffusive fluxes for transport equations with near-degenerate coefficients. *University of Pittsburgh Report*, 2007.

[72] J. Proft and B. Riviere. Discontinuous Gelerkin methods for convection-diffusion equations for varying and vanishing diffusivity. *International Journal of Numerical Analysis & Modeling*, 6(4), 2009.

[73] J. Qiu, B. C. Khoo, and C.-W. Shu. A numerical study for the performance of the Runge-Kutta discontinuous Galerkin method based on different numerical fluxes. *Journal of Computational Physics*, 212(2):540–565, 2006.

[74] W. H. Reed and T. Hill. Triangular mesh methods for the neutron transport equation. Technical report, Los Alamos Scientific Lab., N. Mex.(USA), 1973.

[75] D. Régis. An introduction to isogeometric analysis with application to thermal conduction. Thème numérique 6957, INRIA, Juin 2009.

[76] S. W. Rienstra and A. Hirschberg. An introduction to acoustics. *Eindhoven University of Technology*, 18:19, 2003.

[77] S. H. M. Roth. *Bernstein-Bézier representations for facial surgery simulation*, volume 16. ETH Zurich, 2002.

[78] F. Shakib, T. J. Hughes, and Z. Johan. A new finite element formulation for computational fluid dynamics: x. the compressible Euler and Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 89(1-3):141–219, 1991.

[79] C.-W. Shu. Discontinuous Galerkin methods: general approach and stability. *Division of Applied Mathematics, Brown University*, 44, 2009.

[80] C.-W. Shu and S. Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes. *Journal of Computational Physics*, 77(2):439–471, 1988.

[81] E. Süli. Lecture notes on finite element methods for partial differential equations. *Mathematical Institute, University of Oxford*, 2012.

[82] T. Tezduyar and D. Ganjoo. Petrov-Galerkin formulations with weighting functions dependent upon spatial and temporal discretization: Applications to transient convection-diffusion problems. *Computer Methods in Applied Mechanics and Engineering*, 59(1):49–71, 1986.

[83] T. Tezduyar and T. Hughes. Finite element formulations for convection dominated flows with particular emphasis on the compressible Euler equations. In 21*st Aerospace Sciences Meeting*, page 125, 1983.

[84] T. E. Tezduyar. Computation of moving boundaries and interfaces and stabilization parameters. *International Journal for Numerical Methods in Fluids*, 43(5):555–575, 2003.

[85] T. E. Tezduyar, S. Mittal, S. Ray, and R. Shih. Incompressible flow computations with stabilized bilinear and linear equal-order-interpolation velocity-pressure elements. *Computer Methods in Applied Mechanics and Engineering*, 95(2):221–242, 1992.

[86] T. E. Tezduyar and Y. Osawa. Finite element stabilization parameters computed from element matrices and vectors. *Computer Methods in Applied Mechanics and Engineering*, 190(3-4):411–430, 2000.

[87] T. Toulorge. Efficient Runge-Kutta discontinuous Galerkin methods applied to aeroacoustics. *KU Leuven, Departement Werktuigkunde, Leuven*, 2012.

[88] L. N. Trefethen. *Finite difference and spectral methods for ordinary and partial differential equations*, volume 299. Cornell University-Department of Computer Science and Center for Applied Mathematics, 1996.

[89] G. N. Watson. *A treatise on the theory of Bessel functions*. Cambridge University Press, 1995.

[90] E. Wendland and H. Schulz. Numerical experiments on mass lumping for the advection-diffusion equation. *Revista Minerva*, 2(2):227–233, 2005.

[91] C. Wervaecke. *Simulation d'écoulements turbulents compressibles par une méthode d'éléments finis stabilisée*. PhD thesis, University of Bordeaux 1, 2010.

[92] G. Xu, B. Mourrain, R. Duvigneau, and A. Galligo. Parameterization of computational domain in isogeometric analysis: methods and comparison. *Computer Methods in Applied Mechanics and Engineering*, 200(23-24):2021–2031, 2011.

[93] Q. Xu and J. S. Hesthaven. Discontinuous Galerkin method for fractional convection-diffusion equations. *SIAM Journal on Numerical Analysis*, 52(1):405–423, 2014.

[94] B. Yazid. Etude des courbes de Bézier et des B-splines. Master's thesis, Université Ahmed Ben Bella d'Oran 1, Es Senia, 2011.

[95] F. Zhang, Y. Xu, and F. Chen. Discontinuous Galerkin methods for isogeometric analysis for elliptic equations on surfaces. *Communications in Mathematics and Statistics*, 2(3-4):431–461, 2014.

[96] L. Zhang, T. Gu, J. Zhao, S. Ji, M. Hu, and X. Li. An improved moving least squares method for curve and surface fitting. *Mathematical Problems in Engineering*, 2013.