CAD-consistent adaptive refinement using a NURBS-based Discontinuous Galerkin method

Régis Duvigneau

CAD-consistent adaptive refinement using a NURBS-based Discontinuous Galerkin method

R. Duvigneau

Université Côte d'Azur, Inria, CNRS, LJAD, 2004 route des lucioles, 06902 Sophia-Antipolis, France

Abstract

This study concerns the development of a new method combining high-order CAD-consistent grids and adaptive refinement / coarsening strategies for efficient analysis of compressible flows. The proposed approach allows to use geometrical data from Computer-Aided Design (CAD) without any approximation. Thus, the simulations are based on the exact geometry, even for the coarsest discretizations. Combining this property with a local refinement method allows to start computations using very coarse grids and then rely on dynamic adaption to construct suitable computational domains. The resulting approach facilitates interactions between CAD and Computational Fluid Dynamics (CFD) solvers and focuses the computational effort on the capture of physical phenomena, since geometry is exactly taken into account. The proposed methodology is based on a Discontinuous Galerkin (DG) method for compressible Navier-Stokes equations, modified to use Non-Uniform Rational B-Spline (NURBS) representations. Local refinement and coarsening are introduced using intrinsic properties of NURBS associated to a local error indicator. A verification of the accuracy of the method is achieved and a set of applications are presented, ranging from viscous subsonic to inviscid trans- and supersonic flow problems.

1 Introduction

Computation Fluid Dynamics (CFD) is now commonly employed in engineering design for several applications. This evolution is due to the increase of computational facilities, which has significantly reduced the computational burden related to CFD, and also to the maturity of underlying numerical methods. Nevertheless, the time necessary to set up CFD simulations is still significant because of the complexity of pre-processing tasks, like mesh generation from Computer-Aided Design (CAD) data. Despite the progress of CAD and grid

generation software, the automation of geometrical processing is limited and, as consequence, time-consuming human work is still necessary.

Besides, several recent studies have pointed out the limitations related to the use of piecewise linear grids and the necessity to use curved meshes to obtain the theoretical convergence rates. The use of a piecewise linear approximation of the geometry may also lead to spurious phenomena in the solution [1, 2, 3]. This is especially true in the context of high-order schemes based on Discontinuous Galerkin (DG) or Finite-Volume (FV) methods, which are more and more employed to solve complex problems, for instance in Large Eddy Simulation (LES). As shown by some authors, the use of a high-order geometry description is beneficial, not only in terms of solution accuracy, but also in terms of computational efficiency, because it avoids performing excessive mesh refinement to describe curved boundaries [4].

Therefore, the objective of the current work is to adress both issues: facilitate the transfer of geometrical data from CAD to analysis and use curved grids to maximize the accuracy and efficiency of CFD solvers. A similar idea is adopted in the so-called *isogeometric analysis* approach proposed ten years ago in the context of Finite-Element (FE) methods [5]. This approach obtained a significant success for elliptic problems, with main applications to structural mechanics [6, 7], but only a few investigations can be found in CFD [8, 9, 10, 11], mainly concerning incompressible viscous flows. Indeed, most CFD methods for convection-dominant problems rely on FV or DG formulations, which are more suited to hyperbolic conservation laws.

Accounting for curvilinear boundaries requires first to construct a curvilinear mesh. A possible way is to extend classical meshing techniques to produce curvilinear elements in the vicinity of boundaries [12, 13, 14, 15]. However, these approaches are often restricted to polynomial representations. Thus the link with CAD is not maintained because CAD data mostly rely on rational representations like Non-Uniform Rational B-Splines (NURBS). Alternatively, the CAD community has been very active in the last years in proposing new algorithms to construct high-order computational domains from CAD boundaries, either in a structured environment [16, 17, 18, 19, 20, 21], or in an unstructured triangular or tetrahedral context [22, 23, 24, 25, 26].

Once curved grids are generated, one has to adapt CFD solvers to this curvilinear geometry. The isoparametric approach [27] has been used for a long time in FE and DG contexts, but it is usually based on polynomials. Consequently, points at the boundary of the grid do not lie exactly on the CAD geometry, normals and derivatives are not preserved. More advanced approaches have been proposed recently, to fully integrate NURBS geometries in DG solvers, independently from the approximation space [2, 28]. The demonstrated results are convincing, in terms of accuracy and convergence, but the proposed methodology may suffer from the complexity of the underlying mappings used for spatial integration.

To fully integrate CAD data in a compressible flow solver environnment, we developed in [29] a CAD-consistent DG method, able to preserve exactly CAD geometries defined using NURBS. The high-accuracy obtained was demon-

strated on some academic test-cases. However, the application range was limited, because of the difficulty to generate NURBS-based grids, which are fully adapted to both geometry and solution. Therefore, an extension is presented in this work, which relies on adaptive refinement and coarsening techniques in the context of NURBS-based grids. With this adaptive approach, we obviously aim at improving the quality of the solution and the computational efficiency of the solver, but also at facilitating the initial grid generation. Indeed, we intend to intensively use local refinement procedures operating on a very coarse initial curved mesh, whose generation should be far easier than usually. The article is organized as follows: the underlying NURBS representations are described in section 2, while grid generation and refinement/corsening techniques are presented in section 3. The CAD-consistent DG method is described in section 4, followed by the adaptive strategy in section 5. Finally, a verification exercise is analysed in section 6 and some more complex applications are presented in section 7, ranging from a subsonic viscous flow to inviscid trans- and supersonic flows.

2 NURBS representations

NURBS curves (and surfaces) are now considered as standard to define geometries in CAD [30]. In particular, they allow to represent exactly a broad class of geometric curves commonly encountered in engineering, like conic sections. Moreover, they permit a very intuitive definition and modification of geometrical objects through the handling of *control points*. In the following sections, we describe the most relevant features of such representations and explain how to construct a computational grid based on NURBS, yielding a domain geometrically exact with respect to CAD.

2.1 Basis functions

NURBS functions [31] are rational extensions of B-Spline functions [32]. Both are defined using a so-called *knot* vector $\Xi = (\xi_1, \dots, \xi_l) \in \mathbb{R}^l$, which consists of nondecreasing l real numbers. This knot vector defines a discretization of the *parametric domain* $[\xi_1, \xi_l]$. Open knot vectors, i.e. knot vectors with first and last knots of multiplicity $p+1$, are usually used for representations of degree p to impose interpolation and tangency conditions at both extremities [32]. Therefore $\xi_1 = \ldots = \xi_{p+1}$ and $\xi_{n+1} = \ldots = \xi_{n+p+1}$. B-Spline basis functions $(N_i^p)_{i=1,\cdots,n}$ are defined recursively as [32]:

$$N_i^0(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$N_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_i^{p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1}^{p-1}(\xi). \tag{2}$$

Note that the quotient $0/0$ is assumed to be zero. According to this definition, B-Spline functions of degree 0 and 1 coincide with classical piecewise constant

and linear FE basis functions. However, B-Spline functions of higher degree differ from classical Lagrange polynomials. In particular, they exhibit a higher regularity [5]. We underline that this recursive definition is practically convenient to construct numerical schemes of arbitrary order and promotes the use of p-adaptive strategies. The degree of the functions p, the number of knots l and functions n are related by the relation $l = n+p+1$ [31]. A set of such basis functions is illustrated in Fig. (2a).

Multi-dimensional NURBS basis functions of degree p are defined by using tensor products of B-Spline functions and by associating a *weight* to each function. We present below the two-dimensional case in detail, but the cases corresponding to other dimensions can be derived in a straightforward way. Thus, the two-dimensional NURBS basis function of index $i_1 \times i_2$ is defined in the parameteric domain $[\xi_1, \xi_l] \times [\eta_1, \eta_l]$ (for the sake of simplicity we assume the number of knots and the degrees equal for the two parameters) by associating the weight $w_{i_1\, i_2}$ to the tensor product of B-Spline functions according to:

$$R^p_{i_1\, i_2}(\xi, \eta) = \frac{w_{i_1\, i_2}\; N^p_{i_1}(\xi)\; N^p_{i_2}(\eta)}{\sum_{j_1=1}^{n_1} \sum_{j_2=1}^{n_2} w_{j_1\, j_2} N^p_{j_1}(\xi) N^p_{j_2}(\eta)}. \tag{3}$$

2.2 NURBS patches

Once the NURBS basis is defined, planar NURBS surfaces, that will be considered then as building blocks of the computational domain and denoted as *patches*, can be easily defined by associating to the functions a set of geometrical degrees of freedom:

$$\mathbf{x}(\xi, \eta) = \sum_{i_1=1}^{n_1} \sum_{i_2=1}^{n_2} R^p_{i_1\, i_2}(\xi, \eta) \mathbf{X}_{i_1 i_2}, \tag{4}$$

where $\mathbf{x} = (x, y)$ are the cartesian coordinate values spanned by the patch. In this context, the coefficients associated to the basis functions $(\mathbf{X}_{i_1 i_2})_{i_1=1,\cdots,n_1\; i_2=1,\cdots,n_2}$ are denoted as *control points*. In the following, we simplify this expression by omitting the double index notation and the degree, which gives:

$$\mathbf{x}(\xi, \eta) = \sum_{i=1}^{n_1 \times n_2} R_i(\xi, \eta) \mathbf{X}_i. \tag{5}$$

This construction is illustrated in Fig. (1), for a cubic patch ($p = 3$) with a 4×4 control point lattice ($n_1 = n_2 = 4$), in the particular case $\xi_1 = \xi_2 = \xi_3 = \xi_4 = 0$ and $\xi_5 = \xi_6 = \xi_7 = \xi_8 = 1$ (same knot vector for η). An important property is the fact that any NURBS curve or surface can be considered as the projection of a B-Spline curve or surface defined in a space of higher dimension using its weights [32]. For instance, a NURBS curve lying in the plane (x, y), characterized by a set of control points $(\mathbf{X}_i)_{i=1,\cdots,n} = (x_i, y_i)_{i=1,\cdots,n}$ and weights $(\omega_i)_{i=1,\cdots,n}$, can be described as the projection of a B-Spline curve lying in a 3D space and defined by the control points $(x_i\omega_i, y_i\omega_i, \omega_i)_{i=1,\cdots,n}$. This

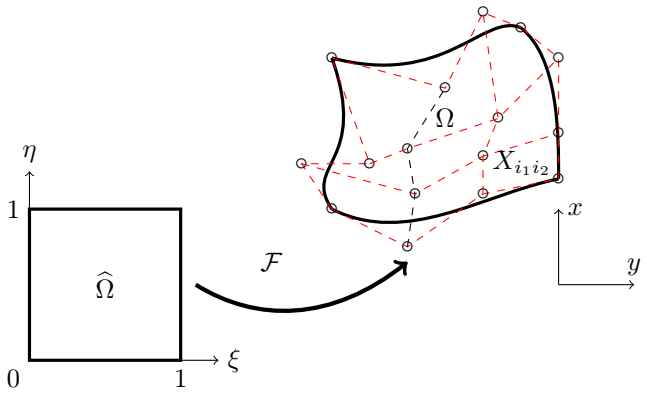


Figure 1: NURBS representation

representation of NURBS objects as B-Spline ones can be useful for practical manipulations.

The transformation of the *parametric domain* $\widehat{\Omega} = [\xi_1, \xi_l] \times [\eta_1, \eta_l]$ to the *physical domain* Ω can now be introduced:

$$\mathcal{F} : \widehat{\Omega} \to \Omega, \quad (\xi, \eta) \mapsto \mathbf{x}(\xi, \eta) = \sum_{i=1}^{n_1 \times n_2} R_i(\xi, \eta) \mathbf{X}_i. \tag{6}$$

This transformation is in general nonlinear. We make the asumption that it is injective and defines a map between the physical domain and the parametric one.

This type of representation was used originally by T. Hughes and co-authors for the foundation of the *isogeometric analysis* paradigm, which is in short a Finite-Element (FE) approach relying on NURBS representations. The knot intervals in the parametric domain are considered as reference elements used in a Galerkin method. More sophisticated representations, such as *T-Splines* [16] and *LBR-Splines* [33] were proposed latter to achieve efficient local refinement. Nevertheless, in the context of a DG formulation, NURBS patches cannot be employed directly and an additional ingredient must be introduced to generate discontinuous solutions at element interfaces. This topic is addressed in section 3.

2.3 Knot insertion and Bézier extraction

An important property of NURBS representations is the capability to insert a new knot, and thus a new basis function, without altering the geometry [32]. It can be considered as a local h-refinement procedure [5]. The one-dimensional case is described below, the extension to higher dimension is straightforward thanks to the tensor product structure.

Any NURBS curve defined by the knot vector $\Xi = (\xi_1, \ldots, \xi_l) \in \mathbb{R}^l$, n basis functions and control points, as described above, can be identically represented using the knot vector $(\xi_1, \ldots, \xi_q, \bar{\xi}, \xi_{q+1} \ldots, \xi_l) \in \mathbb{R}^{l+1}$, that includes an additional knot $\bar{\xi}$ inserted between ξ_q and ξ_{q+1}. If we consider first the case of a B-Spline curve, the new representation can be written as:

$$x(\xi) = \sum_{i=1}^{n} N_i^p(\xi) X_i = \sum_{i=1}^{n+1} N_i^p(\xi) \bar{X}_i, \tag{7}$$

with a new set of control points defined as:

$$\bar{X}_i = (1 - \alpha_i) X_{i-1} + \alpha_i X_i \quad \alpha_i = \begin{cases} 1 & \text{if } i \leq q - p \\ \frac{\bar{\xi} - \xi_i}{\xi_{i+p} - \xi_i} & \text{if } q - p + 1 \leq i \leq q \\ 0 & \text{if } i \geq q + 1 \end{cases} \tag{8}$$

The extension to NURBS curves relies on the projection property already mentioned [31]. The extension for 2D or 3D patches simply consists in applying the above algorithm for each direction independently.

It is important to underline now a particular case: if a new knot is inserted at an existing knot location, the regularity of the curve is decreased. More generally, the curve at the knot ξ_q has the regularity C^{p-r}, where r is the multiplicity of the knot q [32]. Therefore, if one inserts p knots at an existing knot location, the geometry of the curve is preserved but the curve is now divided in two independent parts. If this multiple-knot insertion is achieved for all inner knots, the curve is decomposed into a set of independent Bézier curves. This procedure is named *Bézier extraction* and will be used to define an approximation space suitable to DG formulation, as explained below.

3 Generation and refinement of CAD-consistent domains

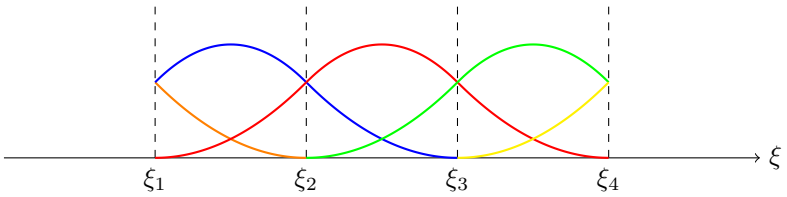
3.1 Initial NURBS-based domain

We consider now an arbitrary physical domain $\Omega \subset \mathbb{R}^2$ with the boundary $\partial\Omega$, that will be used as computational domain. We make the asumption that the geometry of this boundary is defined thanks to NURBS representations and, consequently, that we can define a set of NURBS patches that span the physical domain exactly. Note that the practical construction of such patches is far from being straightforward. It is analogous to building structured meshes, or multi-block structured meshes, but in the context of NURBS representations. In particular, specific techniques should be employed to ensure injectivity [17, 34]. The description of general methods to construct such domains is beyond the scope of the current study, but the reader can refer to [17, 25, 22, 23, 18] to have some examples of admissible methods. The construction of a NURBS-based computational domain is certainly the main issue that isogeometric analysis faces currently. However, the approach proposed in this study aims at alleviating

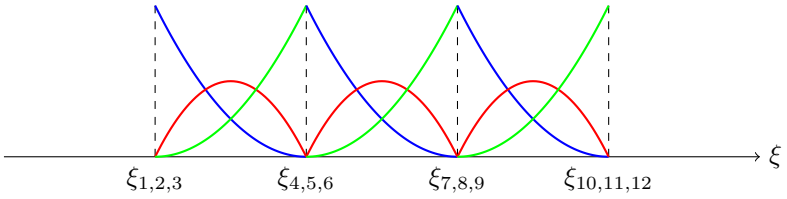
the difficulty by intensive use of local refinement techniques, starting from an extremely coarse initial computational domain, composed of a few elements only.

As explained above, we suppose therfore that the computational domain can be defined by a set of NURBS patches, that exactly matches the boundary of the physical domain. However, the underlying NURBS functions are not suitable to DG methods, since each support of function covers several knot intervals, as shown in Fig. (2a), which does not allow to define solution jumps at the interface between elements. Therefore, the patches have to be transformed into a set of fully disconnected elements, and associated basis functions, which can exhibit discontinuities at the interfaces. Obviously, this should be done without altering the geometry. Hopefully, this task can be easily achieved thanks to the *Bézier extraction procedure*, described in section 2.3. Indeed, as explained above, if p knots are inserted at existing inner knots of a patch of size $n_1 \times n_2$, the original NURBS patch is divided to a set of $(n_1-p)\times(n_2-p)$ *rational Bézier elements* of degree p. The geometry of the physical domain is unchanged, but each element is now defined according to its own basis of size $(p+1)\times(p+1)$, which enables the generation of discontinuities at the interfaces.

This procedure is illustrated in Fig. (2) for a one-dimensional case. Five NURBS functions of degree $p = 2$, which are non-zero on three selected knot intervals, are plotted in Fig. (2a). Their support is composed of $p + 1 = 3$ knot intervals. The knot insertion procedure applied twice for each existing inner knot yields nine quadratic rational Bézier functions, each of them with a support restricted to one knot interval, as illustrated by Fig. (2b).



(a) Five NURBS functions defined on three knot intervals.



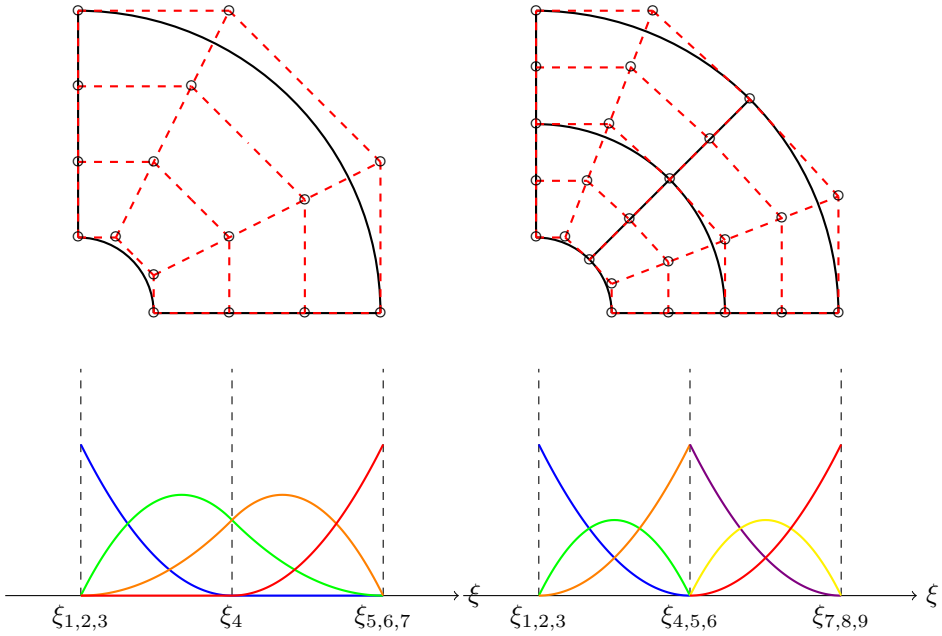
(b) Nine rational Bézier functions obtained by multiple knot insertions.

Figure 2: Procedure to transform a NURBS basis to a set of DG-compliant rational Bézier basis.

A two-dimensional illustration is provided by Fig. (3). One considers the domain delimited by two quarters of cylinder, which can be described as a

NURBS patch of degree $p = 2$ using a lattice of 4×4 control points, as depicted in Fig. (3a). The knot vector for this example is [0 0 0 1 2 2 2]. The basis functions span the whole parametric intervals, which prevent from generating discontinuities at interface ξ_4 (and η_4). As explained above, by inserting two additional knots at inner knot ξ_4 (and η_4), the domain is splitted in four rational Bézier elements, each of them being defined according to its own basis functions (see Fig. (3b)), yielding a DG-compliant computational domain whose geometry matches exactly the original CAD representation.

As summary, the proposed approach to generate an initial CAD-consistent DG-compliant domain is composed of two steps: firstly, generate a set of NURBS patches matching CAD boundaries; secondly, apply Bézier extraction to transform the patches into DG-compliant rational Bézier elements. This approach will be illustrated on several test-cases in the application section.



(a) One NURBS patch (b) Four rational Bézier elements

Figure 3: 2D illustration of Bézier extraction.

3.2 Local refinement

In the original isogeometric analysis paradigm proposed by T. Hughes, local refinement necessitated to introduce sophisticated approaches to avoid the propagation of refinement due to the underlying tensorial representation of NURBS patches, while maintaining a high-order regularity [35]. In the present approach,

local refinement is much simpler because the computational domain is composed of rational Bézier elements which are already disconnected. Therefore, it just consists in inserting $p+1$ new knots in a given element to split it into two new ones. The price to pay is obviously the loss of regularity at the new interface, which is however natural in a DG framework. The refinement of each element can be achieved in an isotropic or anisotropic manner, as illustrated by Fig. (4), by inserting the knots in one direction ξ or η only, or in both directions simultaneously. The procedure is illustrated for a real computational domain in Fig. (5), in the case the refinement is ruled explicitly by the user. The domain is initially defined by a single cubic NURBS patch, according to the CAD geometry of an airfoil. This domain is splitted into 8×6 rational Bézier elements (see Fig. (5a)), yielding a first very coarse DG-compliant grid, that matches exactly the airfoil. One refines then the area located around the airfoil by applying successive refinement in user-defined boxes (see Fig. (5b)). This approach generates a suitable computational domain that preserves the initial CAD geometry, but it supposes to know *a priori* where refinement should be achieved. This is not an easy task in general, especially for complex time-dependent problems. Therefore a fully automated local adaption procedure is developed, in the framework of the DG discretization described below.

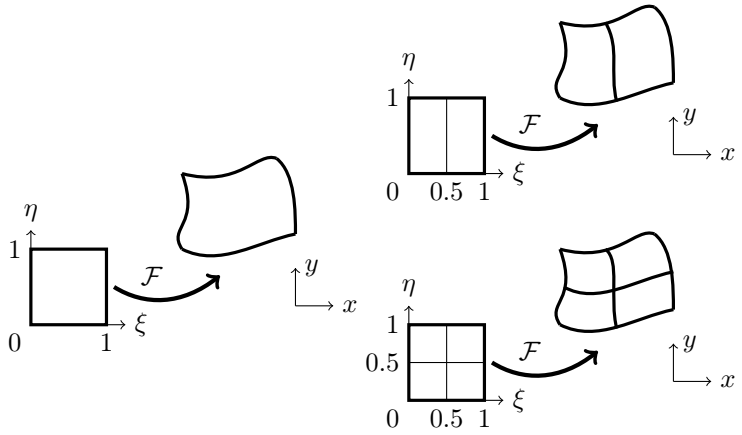
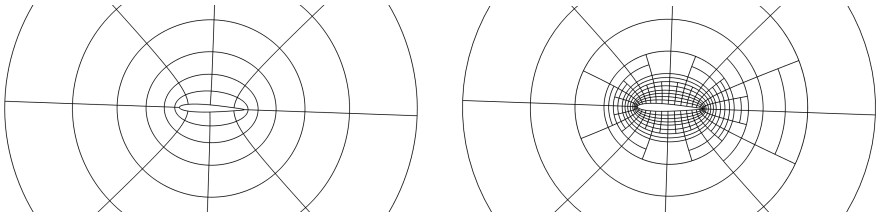


Figure 4: Isotropic or anisotropic local refinement for a single element.

4 Discontinuous Galerkin method

We describe in this section how a standard DG method can be modified to account for the proposed rational Bézier elements. The discretization is first detailed. Then, the issue of shock capturing is considered in the context of these high-order curved elements.



(a) Initial CAD-consistent grid. (b) Refined grid.

Figure 5: Illustration of the local refinement procedure.

4.1 Flow model

We consider the two-dimensional compressible Navier-Stokes equations, that can be written in the conservative form as follows:

$$\frac{\partial \mathbf{W}}{\partial t} + \nabla \cdot (\vec{F} - \vec{G}) = 0, \tag{9}$$

where $\mathbf{W}$ represents the conservative flow variables $(\rho, \rho u, \rho v, \rho e)$, with ρ the density, $\vec{U} = (u, v)$ the velocity vector and e the total energy per unit of mass. We denote c the sound speed. $\vec{F} = (\mathbf{F}_x(\mathbf{W}), \mathbf{F}_y(\mathbf{W}))$ is the vector of the inviscid fluxes, whereas $\vec{G} = (\mathbf{G}_x(\mathbf{W}, \mathbf{S}), \mathbf{G}_y(\mathbf{W}, \mathbf{S}))$ is the vector of the viscous fluxes, with $\mathbf{S} = \nabla \mathbf{W}$ the gradient fields. The pressure p is obtained from the perfect gas state equation:

$$p = \rho(\gamma - 1)(e - \frac{1}{2}\|\overrightarrow{U}\|^2) = \rho(\gamma - 1)e_i, \tag{10}$$

where γ is the ratio of the specific heat coefficients and e_i the internal energy. The inviscid fluxes are given by:

$$\mathbf{F}_x(\mathbf{W}) = \begin{pmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho u(e + \frac{p}{\rho}) \end{pmatrix} \quad \mathbf{F}_y(\mathbf{W}) = \begin{pmatrix} \rho v \\ \rho vu \\ \rho v^2 + p \\ \rho v(e + \frac{p}{\rho}) \end{pmatrix}. \tag{11}$$

The viscous fluxes write as:

$$\mathbf{G}_x(\mathbf{W}) = \begin{pmatrix} 0 \\ \tau_{xx} \\ \tau_{yx} \\ u\tau_{xx} + v\tau_{yx} - q_x \end{pmatrix} \quad \mathbf{G}_y(\mathbf{W}) = \begin{pmatrix} 0 \\ \tau_{xy} \\ \tau_{yy} \\ u\tau_{xy} + v\tau_{yy} - q_y \end{pmatrix}, \tag{12}$$

where the symmetric viscous stress tensor $\bar{\bar{\tau}}$ for Newtonian fluids is:

$$\tau_{xx} = \frac{2}{3}\mu\left(2\frac{\partial u}{\partial x} - \frac{\partial v}{\partial y}\right) \quad \tau_{xy} = \mu\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right) \quad \tau_{yy} = \frac{2}{3}\mu\left(2\frac{\partial v}{\partial y} - \frac{\partial u}{\partial x}\right). \tag{13}$$

The heat flux $\vec{q}$ is based on Fourier's law:

$$(q_x, q_y) = -\gamma \frac{\mu}{Pr} \left(\frac{\partial e_i}{\partial x}, \frac{\partial e_i}{\partial y} \right), \tag{14}$$

where Pr is the Prandtl number.

4.2 Discretization

To apply a DG method [36], one first writes a weak formulation of the problem by multiplying Eq. (9) by an arbitrary test function φ and by integrating over an element Ω_j. In the context of the CAD-consistent computational domain presented above, Ω_j is defined as a rational Bézier element. Thus, the weak formulation writes:

$$\int_{\Omega_j} \frac{\partial \mathbf{W}}{\partial t} \varphi \, d\Omega + \int_{\Omega_j} \nabla \cdot (\vec{F}(\mathbf{W}) - \vec{G}(\mathbf{W}, \mathbf{S})) \, \varphi \, d\Omega = 0. \tag{15}$$

After integration by parts, one obtains classically:

$$\int_{\Omega_j} \frac{\partial \mathbf{W}}{\partial t} \varphi \, d\Omega - \int_{\Omega_j} (\vec{F}(\mathbf{W}) - \vec{G}(\mathbf{W}, \mathbf{S})) \cdot \vec{\nabla} \varphi \, d\Omega + \int_{\partial\Omega_j} (\vec{F}(\mathbf{W}) - \vec{G}(\mathbf{W}, \mathbf{S})) \cdot \vec{n} \, \varphi \, d\Gamma = 0. \tag{16}$$

Since the solution is *a priori* discontinuous at the interfaces, the normal fluxes are evaluated via numerical flux functions $F_n^\star(\mathbf{W}^+, \mathbf{W}^-)$ and $G_n^\star(\mathbf{W}^+, \mathbf{W}^-, \mathbf{S}^+, \mathbf{S}^-)$, defined according to the values of the solution that prevail at each side of the interface. Several flux functions are classically used in DG methods [36, 37]. In this work, the HLLC flux is employed [38, 39] for the inviscid terms, whereas the LDG flux [40] is adopted for the viscous terms. The latter requires the evaluation of the gradient fields:

$$\mathbf{S} = \nabla \mathbf{W} = (\frac{\partial \rho}{\partial x}, \frac{\partial \rho}{\partial y}, \frac{\partial \rho u}{\partial x}, \frac{\partial \rho u}{\partial y}, \frac{\partial \rho v}{\partial x}, \frac{\partial \rho v}{\partial y}, \frac{\partial \rho e}{\partial x}, \frac{\partial \rho e}{\partial y}). \tag{17}$$

By multiplying this definition by a test function and integrating by parts, one obtains a DG formulation for the gradient fields:

$$\int_{\Omega_j} \mathbf{S} \, \varphi \, d\Omega - \int_{\Omega_j} \vec{H}(\mathbf{W}) \cdot \vec{\nabla} \varphi \, d\Omega + \int_{\partial\Omega_j} \vec{H}(\mathbf{W}) \cdot \vec{n} \, \varphi \, d\Gamma = 0, \tag{18}$$

where $\vec{H} = (\mathbf{H}_x(\mathbf{W}), \mathbf{H}_y(\mathbf{W}))$ is the vector of the gradient fluxes, expressed as:

$$\mathbf{H}_x(\mathbf{W}) = \begin{pmatrix} \rho \\ 0 \\ \rho u \\ 0 \\ \rho v \\ 0 \\ \rho e \\ 0 \end{pmatrix} \quad \mathbf{H}_y(\mathbf{W}) = \begin{pmatrix} 0 \\ \rho \\ 0 \\ \rho u \\ 0 \\ \rho v \\ 0 \\ \rho e \end{pmatrix}. \tag{19}$$

One introduces $H_n^\star(\mathbf{W}^+, \mathbf{W}^-)$, the corresponding numerical flux computed at element interfaces according to the LDG method.

Finally, one has to choose a basis for the representation of the solution on each element Ω_j, denoted $\mathbf{W}|^j$. Any basis defined on Ω_j could be selected, but it has been found convenient to employ again the rational Bézier basis of degree p used to describe the geometry of the element. This choice was already tested in [29] with satisfactory results. It was compared to a possible alternate choice consisting of Lagrange polynomials to define the solution associated to a NURBS-based geometry [28]. The accuracy of the two approaches was found very similar. Therefore, for the sake of simplicity, we choose to use a unique basis for both geometry and solution fields. In this sense, the proposed approach can be qualified as isogeometric because the same CAD-consistent representation is employed for the geometry and the solution. Therefore, conservative variables are expressed as:

$$\mathbf{W}|^j(\xi,\eta) = \sum_{i=1}^{(p+1)^2} R_i(\xi,\eta)\mathbf{W}_i. \tag{20}$$

The same basis is also employed for the test functions. Finally, by substitution and expressing integrals in the parametric domain, one obtains:

$$\begin{aligned}\sum_{i=1}^{(p+1)^2} \frac{\partial \mathbf{W}_i}{\partial t} \int_{\widehat{\Omega}_j} R_i\, R_k\, |J_\Omega|\, d\widehat{\Omega} = \int_{\widehat{\Omega}_j} (\vec{F}(\mathbf{W}) - \vec{G}(\mathbf{W},\mathbf{S})) \cdot \nabla \vec{R}_k\, |J_\Omega|\, d\widehat{\Omega} \\ - \int_{\partial\widehat{\Omega}_j} (F_n^\star(\mathbf{W}^+,\mathbf{W}^-) - G_n^\star(\mathbf{W}^+,\mathbf{W}^-,\mathbf{S}^+,\mathbf{S}^-))\, R_k\, |J_\Gamma|\, d\widehat{\Gamma}\end{aligned} \tag{21}$$

$$\begin{aligned}\sum_{i=1}^{(p+1)^2} \mathbf{S}_i \int_{\widehat{\Omega}_j} R_i\, R_k\, |J_\Omega|\, d\widehat{\Omega} = \int_{\widehat{\Omega}_j} \vec{H}(\mathbf{W}) \cdot \nabla \vec{R}_k\, |J_\Omega|\, d\widehat{\Omega} \\ - \int_{\partial\widehat{\Omega}_j} H_n^\star(\mathbf{W}^+,\mathbf{W}^-)\, R_k\, |J_\Gamma|\, d\widehat{\Gamma}.\end{aligned} \tag{22}$$

$|J_\Omega|$ represents the determinant of the Jacobian matrix of the geometric transformation $\mathcal{F}$, defined according to Eq. (6) and $|J_\Gamma|$ is the lineal counterpart. Obviously, one identifies easily a mass matrix on the left-hand side of Eqs. (21-22), volumic residuals and interface fluxes on the right-hand side. The local mass matrix is inverted once and its inverse is stored, an explicit third-order four-step Strong Stability Preserving (SSP) Runge-Kutta method being used for time integration. We underline that the spatial integrals in Eq. (21-22) have to be evaluated carefully. Indeed, the non-linearities in the flow variables and in the geometrical transformation necessitate high-order quadratures, depending on the basis degree. For the present work, classical Gauss-Legendre quadrature rules have been applied, but involving a large number of evaluation points when required.

4.3 Initial and boundary conditions

For unsteady problems, the initial flow solution has to be expressed accordingly to the approximation space selected, i.e. in terms of discontinuous rational Bézier functions, as in Eq. (20). This task is achieved by solving a set of local least-squares fitting problems. If the initial solution is defined globally by $\mathbf{w}_0$, the initial solution in each element Ω_j is obtained by:

$$\mathbf{W}|_{t=0}^{j} = \text{argmin } \frac{1}{2} \int_{\Omega_j} \left(\sum_{i=1}^{(p+1)^2} R_i \mathbf{W}_i - \mathbf{w}_0 \right)^2 d\Omega. \tag{23}$$

The optimality condition with respect to the degree of freedom $\mathbf{W}_k$, expressed in the parametric domain, yields a linear system involving the local mass matrix:

$$\sum_{i=1}^{(p+1)^2} \mathbf{W}_i \int_{\widehat{\Omega}_j} R_i \, R_k \, |J_\Omega| \, d\widehat{\Omega} = \int_{\widehat{\Omega}_j} \mathbf{w}_0 \, R_k \, |J_\Omega| \, d\widehat{\Omega}. \tag{24}$$

Boundary conditions are imposed weakly via the normal fluxes at boundary, which can be computed using different approaches [41], either physical or numerical. For inlet and outlet boundaries, a numerical flux is evaluated using the interior state on one side and a reconstructed exterior state on the other side, computed from the free-stream conditions using Riemann invariants [39, 41]. For wall boundaries, the physical flux is directly imposed, involving only the pressure terms for the inviscid part and the shear stress and heat flux for the viscous part.

4.4 Shock capturing

It is necessary to stabilize oscillations appearing in the solution in presence of shocks, either by introducing a solution limitation procedure or by adding artifical viscosity. In this work, the sub-cell capturing approach proposed by Persson and Peraire [42] is extended to rational Bézier representations.

An artifical local viscosity μ_j, considered as constant over each element Ω_j, is introduced in the model. As shown in [43], a more regular viscosity field could be beneficial and will be envisaged in a future work. In the case of Navier-Stokes equations, this local viscosity is just added to the physical diffusion coefficient μ. In the case of Euler equations, it necessitates to add the viscous terms in the discretization. As shown in [42], the use of the physical viscous flux offers some advantages with respect to a more simple laplacian-based smoothing operator. More precisely, the artifical viscosity μ_j is defined by the following smooth function:

$$\mu_j = \begin{cases} 0 & \text{if } s_j < s_0 - \kappa \\ \frac{\mu_j^0}{2} \left(1 + \sin \frac{\pi (s_j - s_0)}{2\kappa} \right) & \text{if } s_0 - \kappa < s_j < s_0 + \kappa \\ \mu_j^0 & \text{if } s_j > s_0 + \kappa \end{cases} \tag{25}$$

where μ_j^0 is the maximum artificial viscosity admissible in the element Ω_j, s_j a sensor of the oscillations of the solution in the element and (s_0, κ) global numerical parameters that control the artificial viscosity distribution. As analysed in [42], μ_0 should be defined according to local length and velocity scales. To capture the shock in a single cell, the length scale should be h/k, where h is a measure of the element length and k the scheme order. In the context of the proposed rational Bézier representation, the min distance between element corners is chosen as element length, i.e. $h = \min(|\mathbf{X_{11}} - \mathbf{X_{1n_2}}|, |\mathbf{X_{11}} - \mathbf{X_{n_1 1}}|, |\mathbf{X_{n_1 1}} - \mathbf{X_{n_1 n_2}}|, |\mathbf{X_{1n_2}} - \mathbf{X_{n_1 n_2}}|)$, which is easy to estimate whatever the element shape. Regarding the velocity scale, the free-stream velocity u_∞ is chosen, yielding a maximum viscosity defined locally as:

$$\mu_j^0 = \frac{h}{p+1}\, u_\infty \tag{26}$$

The parameter s_0 is scaled according to a Fourrier analysis [42]: if the solution is smooth, the amplitude of high-frequencies should decay like $\mathcal{O}(1/k^2)$. Therefore, the switch s_0 is set as $\tilde{s}_0/(p+1)^2$, with $\tilde{s}_0$ a problem dependent parameter which will be discussed latter, for each application. κ controls the extent of the transition and is chosen equal to $\tilde{s}_0/2$. It remains to define the local sensor s_j. In the original method [42], the sensor is based on the $\mathcal{L}^2$-norm of the highest term in a modal expansion of the solution. In this work, we propose to define the sensor by quantifying the oscillations in the control point lattice that represents the solution in each element. More precisely, one evaluates the following quantities, that compare the total variation of control points in each direction to the variation observed between element extremities (we suppose here that $p > 1$):

$$s_\xi^{i_2} = \left(\sum_{i_1=1}^{p} |\mathbf{W}_{i_1+1\, i_2} - \mathbf{W}_{i_1 i_2}| \right) - |\mathbf{W}_{p+1\, i_2} - \mathbf{W}_{1 i_2}| \tag{27}$$

$$s_\eta^{i_1} = \left(\sum_{i_2=1}^{p} |\mathbf{W}_{i_1 i_2+1} - \mathbf{W}_{i_1 i_2}| \right) - |\mathbf{W}_{i_1\, p+1} - \mathbf{W}_{i_1 1}| \tag{28}$$

These quantities vanish if the solution is monotonic over the element. If not, it is a measure of the oscillatory nature of the solution in each parametric direction, thanks to the convex hull property of NURBS curves[31]. The sensor for a given element is finally defined as:

$$s_j = \frac{1}{2(p+1)} \left(\sum_{i_2=1}^{p+1} s_\xi^{i_2} + \sum_{i_1=1}^{p+1} s_\eta^{i_1} \right) \frac{1}{\overline{\overline{\mathbf{W}}}} \tag{29}$$

where $\overline{\overline{\mathbf{W}}}$ is the mean solution over the element. In the illustration presented below, this sensor is applied to the density field, but other choices could be envisaged, such as Mach number for instance.

5 Adaptive refinement

As explained in section 3.2, a procedure based on successive user-defined refinements is very limited in terms of applications, because it relies on the intuition of the user. A common approach in engineering is to refine the domain in the vicinity of walls for an accurate approximation of the geometry and boundary layers, in the wakes to capture vortex shedding, possibly in regions where shocks could appear, etc. Obviously, this becomes tedious when unsteady flows are considered. This task is even harder when using high-order schemes, because the required resolution level is less intuitive.

The aim of the present work is to make adaption easier and more efficient. Accounting exactly for the CAD geometry, whatever the discretization level used for the resolution, is a first ingredient (*exact-geometry property*). It avoids refining locally the grid only to approximate the geometry and, thus, allows to focus refinement on the accuracy of the physical solution. The second ingredient is a fully automated, time-dependent, refinement and coarsening procedure, excluding user intuition to select areas to be refined. The joint use of both ingredients allows computations starting with an extremely coarse CAD-compliant domain, which is automatically and continuously adapted to the physical solution along time. The refinement procedure for the geometry has already been detailled in section 3.2. In the following sections, we describe the dynamic implementation of the computational domain, the refinement and coarsening criteria and the treatment of the solution.

5.1 Adaptive domain management

On the basis of the refinement procedure for rational Bézier elements described in section 3.2, a dynamic grid management is adopted using a tree-based approach. The root is composed of elements of level 0, defined by the user from CAD data (initial grid), as described in section 3.1. Then, the refinement of any element of level l yields four elements of level $l+1$, an isotropic splitting being chosen for the sake of simplicity. These four elements are denoted as brothers. Any element of level l has the knowledge of its corresponding element at level $n-1$, denoted as father, and corresponding elements at level $n+1$, denoted as sons, if they exist. Thus, the grid structure can be refined arbitrarily, with the constraint that two neighboring elements cannot have a difference of levels higher than one. This constraint is added to prevent an element from being faced to more than two refined elements in any direction, for mesh regularity reasons. A maximum refinement level l_{max} is imposed to avoid an infinite series of refinements, that may occur in case of shock adaption for instance. Obviously, all existing elements are not used at each time step for the resolution. Therefore, elements are flagged as active if they are currently used for integration, or inactive. A very similar management is adopted for faces, which are stored using a tree-based approach. The refinement of a given face yields two faces, denoted as daughters, whereas the original face is denoted as mother. Faces can also be flagged as active or inactive.

The use of such a locally refined computational domain has a very limited impact on the implementation of the DG method, thanks to the flexibility of the DG scheme to handle non-conforming grids. The only one modification concerns the flux integration at the interface between two elements of different levels, as illustrated in Fig. (6). In this case, the numerical quadrature is achieved by using the integration points of the smallest element. Thus, local refinement has no impact on the scheme accuracy, as will be verified later. Obviously, the parametric representation of two neighboring elements could be different, which should be tacken into account for the evaluation of left and right quantities.

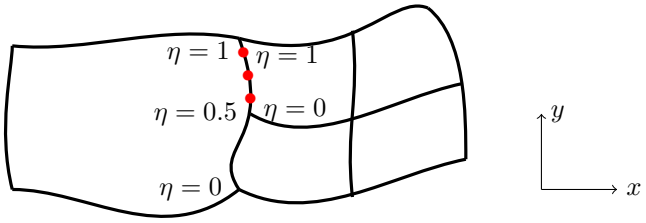


Figure 6: Flux quadrature in case of local refinement.

5.2 Refinement and coarsening criteria

Several refinement criteria can be found in the literature to drive the adaption procedure in the context of DG discretizations, see for instance [44, 45, 46]. In the present study, we choose an approach based on the measure of interface jumps, able to tackle a broad range of problems easily. More sophisticated approaches could be used as well and will be studied in forthcoming works. Nevertheless, as will be shown in application section, this indicator has the ability to identify regions exhibiting both discontinuities and under-resolved elements in case of smooth solutions. In practice, for each element Ω_j, one evaluates the solution jump at the interface between Ω_j and the neighboring elements:

$$\epsilon_j = \sum_{k \in \mathcal{N}_j} \int_{\Gamma_{jk}} \| \mathbf{W}|^j - \mathbf{W}|^k \| \, d\Gamma, \tag{30}$$

where $\mathcal{N}_j$ represents the set of elements around Ω_j and Γ_{jk} the interface between Ω_j and Ω_k. Then, the element Ω_j is flagged for refinement if the indicator ϵ_j exceeds the indicator average $\bar{\epsilon}$ by a user-defined ratio ϵ_{ref}:

$$\frac{\epsilon_j}{\bar{\epsilon}} > \epsilon_{ref}. \tag{31}$$

Similarly, the element is flagged for coarsening if ϵ_j is lower than $\bar{\epsilon}$, with a user-defined ratio ϵ_{coa}:

$$\frac{\epsilon_j}{\bar{\epsilon}} < \epsilon_{coa}. \tag{32}$$

From geometrical point of view, the refinement of a selected element is achieved using the knot insertion procedure described in section 2.3. The coarsening can be performed only if all the four brother elements are flagged for coarsening. In this case, no new computation is required, it just consists in changing their status as inactive, while their father becomes active again. Faces are also updated accordingly.

5.3 Solution refinement and coarsening

When the computational domain is refined or coarsened, the current solution has to be updated. In the case of refinement, we benefit from the fact that geometry and solution have the same rational Bézier representation. Therefore, both are splitted simultaneously and identically using the same knot insertion procedure described above. As consequence, no loss of information occurs during refinement.

In the case of coarsening, the solution should be approximated in the father element, from data located in the four sons elements. This is achieved by using a least-squares projection. Similarly as the approximation of the initial condition, see Eqs (23-24), one first computes a solution estimate $\mathbf{W}|^{f_{LS}}$ in the father element Ω_f from the solutions $(\mathbf{W}|^s)_{s=1,\dots,4}$ available in the son elements $(\Omega_s)_{s=1,\dots,4}$ by solving the following local linear system:

$$\sum_{i=1}^{(p+1)^2} \mathbf{W}_i|^{f_{LS}} \int_{\widehat{\Omega}_f} R_i\, R_k\, |J_\Omega|\, d\widehat{\Omega} = \sum_{s=1}^{4} \int_{\widehat{\Omega}_s} \mathbf{W}|^s\, R_k\, |J_\Omega|\, d\widehat{\Omega}. \tag{33}$$

Then, this approximation is shifted to enforce conservation, yielding a solution in the father element defined as:

$$\mathbf{W}|^f = \mathbf{W}|^{f_{LS}} + \frac{1}{\mathcal{A}_f}\left(\sum_{s=1}^{4} \int_{\widehat{\Omega}_s} \mathbf{W}|^s\, |J_\Omega|\, d\widehat{\Omega} - \int_{\widehat{\Omega}_f} \mathbf{W}|^{f_{LS}}\, |J_\Omega|\, d\widehat{\Omega}\right), \tag{34}$$

where $\mathcal{A}$ is the area of a given element. The integration of the above equation yields the conservation of the mean solution during coarsening, up to quadrature accuracy, which is a necessary condition. Contrary to the refinement process, coarsening leads however to a loss of information during the least-squares fitting procedure.

6 Verification

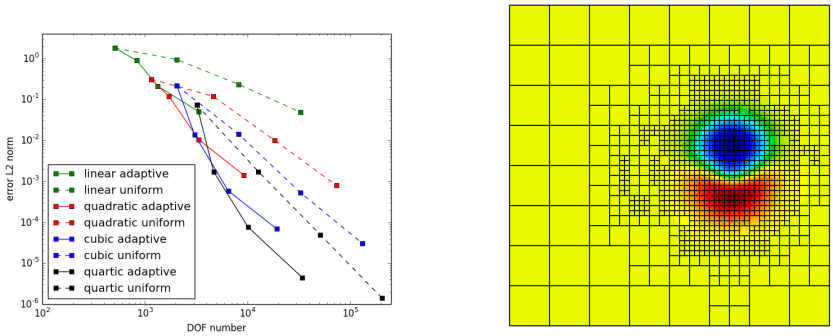
As first problem, we consider the transport of an isentropic vortex [36], whose analytical solution is provided by:

$$\begin{aligned} \rho &= \left(1 - \frac{\gamma - 1}{16\gamma\pi^2}\beta^2 e^{2(1-r^2)}\right)^{\frac{1}{\gamma-1}}, \\ u &= 1 - \beta e^{1-r^2}\ \frac{y - y_0}{2\pi}, \\ v &= \beta e^{1-r^2}\ \frac{x - t - x_0}{2\pi}, \\ p &= \rho^\gamma, \end{aligned} \tag{35}$$

with $r = \sqrt{(x - t - x_0)^2 + (y - y_0)^2}$, $x_0 = 5$, $y_0 = 0$ and $\beta = 5$. The physical domain Ω is the square $[0, 10] \times [0, 10]$. The analytical solution is used to define the initial condition and the exterior state for boundary fluxes. Error is computed at final time $t_{fin} = 2$. The assessment is first carried out using regular grids, for degrees ranging from one to four. Then, the proposed adaptive refinement and coarsening procedure is tested, using different maximum levels, for the same basis degrees. The parameters that drive the adaption are chosen as $\epsilon_{ref} = 1$ and $\epsilon_{coa} = 0.5$. The $\mathcal{L}^2$-norm of the error is computed by numerical quadrature, for the energy field. Figure (7a) shows the evolution of the error with respect to the number of degrees of freedom in log-log scale. Table (1) provides for some computations the number of degrees of freedom, the error and an estimate of the convergence rate. For the adaptive cases, the number of degrees of freedom corresponds to the final time and the maximum refinement level $l_{max} = 3$. The convergence rate is estimated from the slope of the error curve in Figure (7a). Globally, one observes that the convergence rate obtained using uniform grids is slightly above the optimal rate $p + 1$. When using the adaptive procedure, the number of degrees of freedom is drastically reduced, while the error is maintained at a similar level. The convergence rate is tedious to evaluate in the case of non-uniform adaptive grids, nevetheless its estimation seems to be close to the optimal one. Figure (7b) illustrates the solution obtained using three refinement levels.

	Uniform			Adaptive		
degree	DOF number	error	rate	DOF number	error	rate
1	32 768	4.8710^{-2}	2.2	3 318	5.1110^{-2}	3.1
2	73 728	7.8910^{-4}	3.6	9 150	1.4310^{-3}	3.8
3	131 072	3.0710^{-5}	4.1	19 209	6.9210^{-5}	3.9
4	204 800	1.4110^{-6}	5.1	34 160	4.4610^{-6}	4.7

Table 1: Convergence rates for the isentropic vortex case.



(a) Error assessment.

(b) Momentum-x field and adapted grid.

Figure 7: Isentropic vortex case.

7 Applications

7.1 Transonic inviscid flow in a nozzle

We consider as first application the inviscid flow in a nozzle at transonic regime. The geometry of the nozzle is defined using a single NURBS curve of degree four. Inlet and outlet are respectively located at $x = 0$ and $x = 4$. The maximum width of the nozzle is 2 and its minimum 1.6. The flow is supposed to be symmetric with respect to $y = 1$ and, therefore, only half of the geometry is represented as computational domain. As inlet, a density of value 1.4 and a velocity of value 0.65 are weakly imposed, while a pressure of value 1 is weakly prescribed at outlet. The solution is initialized using these values, yielding a reference Mach number $M_0 = 0.65$.

Since the geometry of the nozzle is described by a single NURBS curve, one could easily define an initial computational domain by a single NURBS patch. After Bézier extraction procedure, one obtains a set of 8×4 rational Bézier elements of degree $p = 4$ as initial grid, as illustrated in Fig. (8). The adaption procedure is achieved using refinement and coarsening criteria set to $\epsilon_{ref} = 2$ and $\epsilon_{coa} = 0.5$ respectively. The maximum refinement level is $l_{max} = 4$. The parameter controling the artificial viscosity switch is set to $\tilde{s}_0 = 10$. Some tests have shown that this parameters can vary significantly without impacting the results.

The time evolution of the energy field and the corresponding adapted grid are shown in Fig. (9). As observed, some waves are generated at initial time at the bump location, which are then propagated and reflected. The flow finally converges towards a solution characterized by a steady shock. The adaption procedure allows an accurate description of the wave system and a shap capture of the shock. One can underline that the exact-geometry property, permitted by the use of a NURBS-based grid, allows to employ a very coarse grid at the bump when the flow is regular. Clearly, it would not be possible to use such a coarse

mesh with classical piecewise linear elements. As illustrated in Fig. (10), the shock is located in a single element, as expected using the sub-cell capturing approach, and the artificial viscosity field is non-zero only in a very narrow region around the shock.

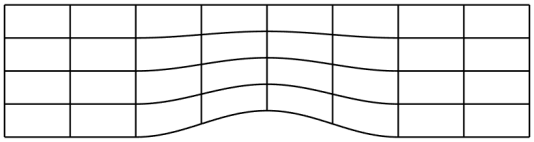
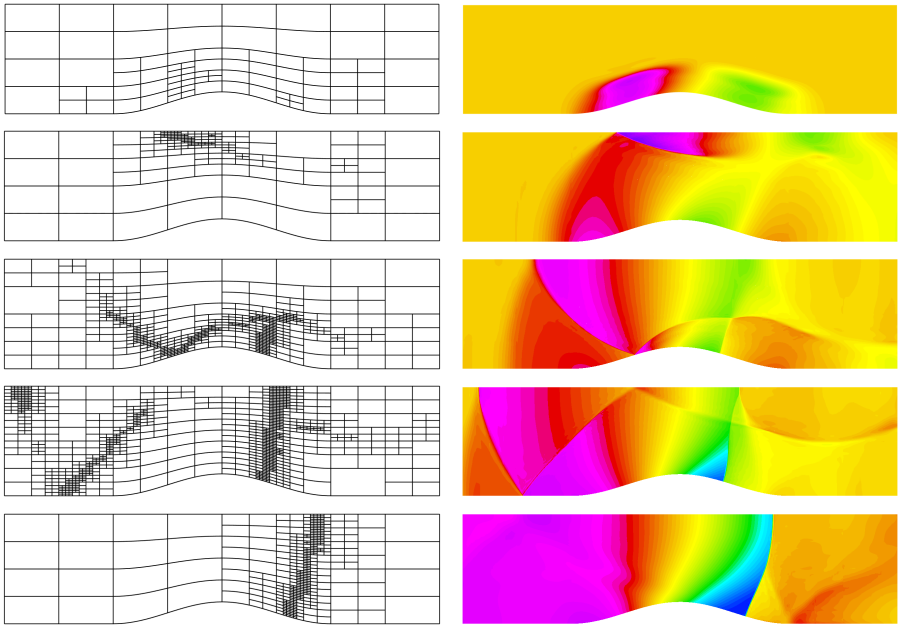


Figure 8: Initial grid for the nozzle test-case.

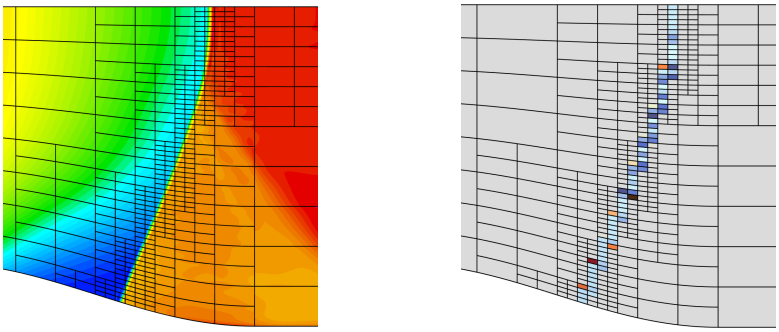


(a) Adapted grid (b) Energy field

Figure 9: Transonic flow test case at selected time steps.

7.2 Subsonic viscous flow around a cylinder

As second test-case, we consider the subsonic flow around a cylinder, with a Reynolds number $R_\infty = 200$ and a Mach number $M_\infty = 0.1$ as free-stream conditions. The computational domain is the rectangle $[-30, 40] \times [-30, 30]$, the cylinder radius being equal to one. The initial grid is composed of only 15 rational Bézier elements of degree $p = 3$, as shown in Fig. (11). It is constructed



(a) Energy field (b) Artificial viscosity field

Figure 10: Zoom around shock location at final time.

by defining first a set of four rational quadratic elements around the cylinder, as explained in section 3, which are then transformed into cubic elements by degree elevation [31], and finally connected to a set of surrounding cubic elements to cover the computational domain until the exterior boundary. This initial grid is extremely coarse, but very easy to define. We rely then on the CAD-consistent adaptive procedure to construct automatically a suitable computational grid. The adaption parameters are chosen as $\epsilon_{ref} = 2$ and $\epsilon_{coa} = 0.5$, and different maximum refinement levels are tested $l_{max} = 2$, $l_{max} = 4$ and $l_{max} = 6$. Note that a small rotation of 0.5 degree is carried out to favor the trigger of the vortex shedding process.

The results are plotted in Fig. (12), at non-dimensional time $t_{fin} = 350$, in terms of adapted grid and density field, with a zoom in the cylinder wake. As can be seen, the adaption algorithm targets for refinement both the vicinity of the cylinder and the wake. For $l_{max} = 2$, the discretization of the cylinder is composed of only 16 faces, which remains very coarse. Nevertheless, the vortex shedding process is captured. However, the description of the vortices in the wake is very poor and discontinuities of the solution at element interfaces are signficant. For $l_{max} = 4$, the flow description is far better. In particular, the density field in the vicinity of the cylinder looks smooth and vortices are now well represented, although some interface discontinuities can still be observed in the wake. One can notice that the refinement area follows correctly the oscillation of vortices in the wake. Finally, for $l_{max} = 6$, a very accurate and smooth representation of the density field is obtained everywhere, thanks to the refinement that fits perfectly the characteristics of the wake.

7.3 Supersonic inviscid flow around a cylinder in a duct

Finally, the supersonic flow around a cylinder in a duct is considered. The flow is supposed to be inviscid and inlet boundary conditions correspond to $M_\infty = 2$. A cylinder of diameter D is located between two walls forming a duct of height $H =$

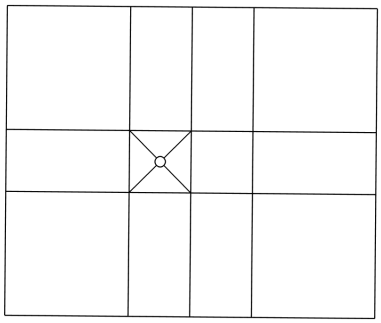
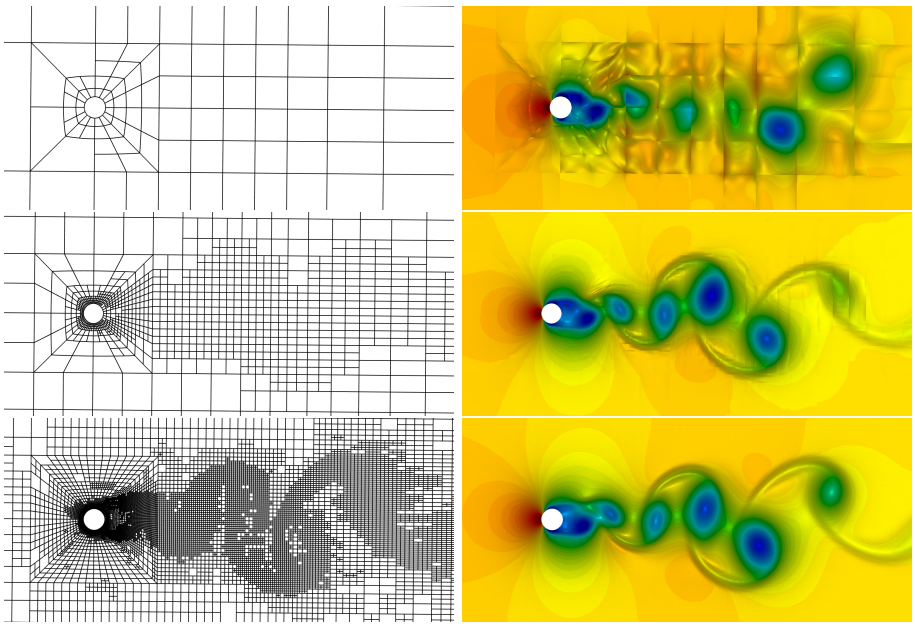


Figure 11: Initial grid for viscous cylinder test-case.



(a) Adapted grid (b) Density field

Figure 12: Subsonic viscous test case for $l_{max} = 2$ (top), $l_{max} = 4$ (middle) and $l_{max} = 6$ (bottom).

$8D$, as studied in [45]. An initial grid is constructed in the spirit of the one used in the previous section. It counts only six rational Bézier elements of degree $p = 3$, as depicted in Fig. (13). The flow is characterized by the generation of a bow shock at the front of the cylinder, which is reflected symmetrically on the duct walls. In the cylinder wake, some vortices are generated, which interact with the reflected shocks to form complex vortical structures, as shown in Fig. (14) for a fine mesh, at time $t_{fin} = 12.5\, D/c$. The objective of this test-case is therefore to

study the capability of the method to accurately capture, by local refinement, both shocks and vortices, starting from a very coarse discretization.

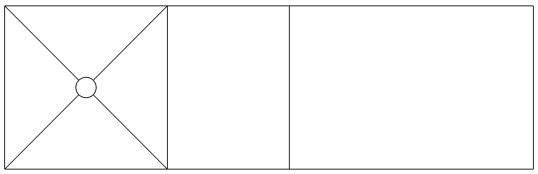


Figure 13: Initial grid for supersonic test-case.

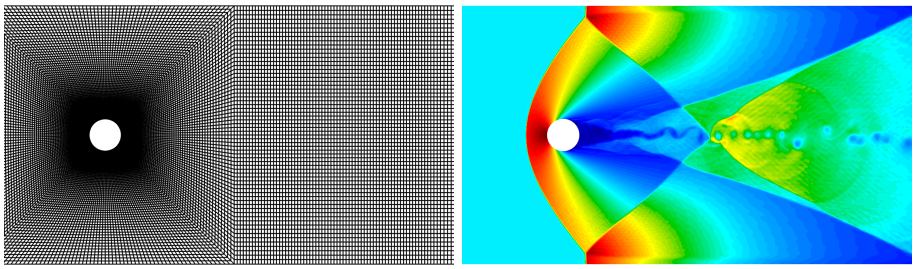
In this perspective, the maximum number of refinement levels is set to the value $l_{max} = 6$, with varying refinement and coarsening criteria. Additionally, a fine mesh is generated by applying the six refinement levels uniformly. Therefore, the smallest elements have the same size in the two cases. The grid parameters are summarized in table (2). The parameter controling the artificial viscosity switch is set to the value $\tilde{s}_0 = 5$. The flow solution and the grid are shown at time t_{fin} in Fig. (14) for the fine mesh, and in Fig. (15) for the adapted grids. For all cases with adaption, one can notice that the grid is quite coarse in the vicinity of the cylinder, which is not dommageable thanks to the exact-geometry property of the proposed approach. For the adapted grid number 1, characterized by large ϵ_{ref} and ϵ_{coa} values, the refinement is focused around the bow shock, which is accurately captured. However, the regions corresponding to the reflected shocks and the cylinder wake are below the refinement threshold for the last levels. As consequence, the grid remains too coarse in these regions to capture correctly these phenomena. For the adapted grid 2, a decrease of the refinement criteria yields a far better grid. In particular, the reflected shocks are well captured, as well as the interactions with the vortex shedding process. As can be seen in table (2), this improvement is obtained for a significant increase of the number of elements, which remains however far lower than that of the uniform fine grid. Nevertheless, some discrepencies with the reference results can still be observed in the region where shocks and vortices interact. They can be explained by two reasons: firstly, the generation of the vortices is initiated in two different ways ; secondly, the use of the adaption procedure seems to promote the generation of larger vortices, due to successive refinement / coarsening steps during time evolution. Finally, the use of the adapted grid 3 yields results very close to the reference ones. Indeed, using lower refinement criteria prevents coarsening in the wake region. As consequence, the transport of small vortices is better represented.

8 Conclusion

In this work, a CAD-consistent adaptive and refinement procedure was presented, in the framework of a DG method for compressible Euler and Navier-Stokes equations. In particular, we explained how intrinsic properties of NURBS

	ϵ_{ref}	ϵ_{coa}	DOF number (at t_{fin})
Fine grid	-	-	393 216
Adapted grid 1	0.70	2.00	22 896
Adapted grid 2	0.50	1.25	91 344
Adapted grid 3	0.25	1.00	164 880

Table 2: Refinement parameters for the supersonic test-case.

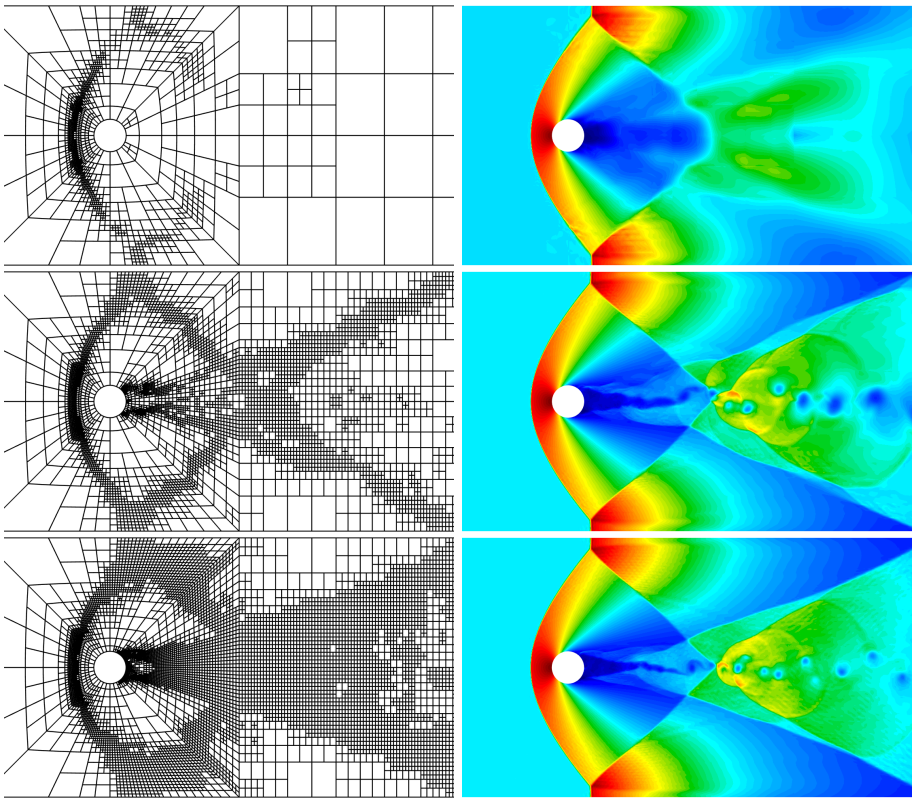


(a) Reference fine grid (b) Reference energy field

Figure 14: Reference results for the supersonic test-case.

representations can be used to generate a DG-compliant grid from CAD data, based on rational Bézier elements, and to define refinement and coarsening strategies. It should be underlined that the modifications of the DG method are limited and only concern the definition of the basis functions. If an existing DG code is written in a generic way, implementing the proposed approach may represent a small coding effort in practice. We also presented a modification of the sub-cell shock capturing method to account for the parametric representation employed.

The accuracy of the method was verified for a problem with an analytical solution and optimal convergence rates were found. Finally, a set of flow problems were tested, to assess the capability of the method to capture accurately flow characteristics. As shown, the adaption strategy was able to consider a range of flow regimes, from subsonic to supersonic, and detect regions of interest like shocks and vortices, although the error indicator used was very simple. Obviously, the accuracy obtained depends on the number of refinement steps allowed, as well as the refinement / coarsening criteria used. It was reported that a good choice of these parameters may be tedious when different phenomena are simultaneously present, like shocks and vortices in the case of the supersonic flow around a cylinder. The use of curved geometries associated to refinement / coarsening capabilities was especially interesting. Indeed, it was possible to use very coarse initial grids and rely on the adaption procedure to define a suitable computational domain. Moreover, we observed that, in some cases, refinement was not necessary at wall boundaries, provided that the geometry was exactly represented by coarse curved elements.



(a) Adapted grids (b) Adapted solutions

Figure 15: Results with adaption for the supersonic test-case for different adaption parameters

Efforts should now be focused on the achievement of more complex problems, for instance three-dimensional or turbulent flows, which will require to develop more efficient tools to manage CAD geometries in practice. Coupled problems, like fluid-structure interactions, may also benefit from the developed methodologies.

References

[1] Bassi F, Rebay S. A high-order accurate discontinuous finite element method for the numerical solution of the compressible navier-stokes equations. *J. Comput. Physics* 1997; **131**(2):267–279.

[2] Silveira A, Moura R, Silva A, Ortega M. Higher-order surface treatment for discontinuous Galerkin methods with applications to aerodynamics. *Int. J. for Numerical Methods in Fluids* 2015; (79):323–342.

[3] Costa R, Clain S, Loubère R, Machado GJ. High-order accurate finite volume scheme on curved boundaries for the two-dimensional steady-state convection-diffusion equation with dirichlet condition. *Applied Mathematical Modelling* 2018; **54**.

[4] Sevilla R, Fernandez-Mendez S, Huerta A. NURBS-enhanced finite element method for euler equations. *Int. J. for Numerical Methods in Fluids* 2008; **57**(9).

[5] Hughes T, Cottrell J, Bazilevs Y. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 2005; (194):4135–4195.

[6] Bazilevs Y, de Veiga LB, Cottrell J, Hughes T, Sangalli G. Isogeometric analysis: approximation, stability and error estimates for refined meshes. *Mathematical Models and Methods in Applied Sciences* 2006; (6):1031–1090.

[7] Cottrell J, Hughes T, Bazilevs Y. *Isogeometric analysis : towards integration of CAD and FEA*. John Wiley & sons, 2009.

[8] Bazilevs Y, Michler C, Calo V, Hughes T. Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly-enforced boundary conditions on unstretched meshes. *Computer Methods in Applied Mechanics and Engineering* 2010; (199):780–790.

[9] Evans J, Hughes T. Isogeometric divergence-conforming B-Splines for the steady Navier–Stokes equations. *Mathematical Models and Methods in Applied Sciences* 2013; **23**(8).

[10] Nortoft P, Dokken T. Isogeometric analysis of Navier-Stokes flow using locally refinable B-Splines. *SAGA – Advances in ShApes, Geometry, and Algebra*, vol. 10, Springer (ed.), 2014; 299–318.

[11] Hosseini BS, Möller M, Turek S. Isogeometric analysis of the navier–stokes equations with taylor–hood b-spline elements. *Applied Mathematics and Computation* 2015; **267**(264–281).

[12] Persson P, Peraire J. Curved mesh generation and mesh refinement using lagrangian solid mechanics. *47th AIAA Aerospace Sciences Meeting*, 2009-949, 2009.

[13] George P, Borouchaki H. Construction of tetrahedral meshes of degree two. *Int. J. for Numerical Methods in Engineering* 2012; **90**(9):1156–1182.

[14] Abgrall R, Dobrzynski C, Froehly A. A method for computing curved meshes via the linear elasticity analogy, application to fluid dynamics problems. *Int. J. for Numerical Methods in Fluids* 2014; **76**:246–266.

[15] Geuzaine C, Johnen A, Lambrechts J, Remacle JF, Toulorge T. The generation of valid curvilinear meshes. *Notes on Numerical Fluid Mechanics and Multidisciplinary Design Volume* 2015; **128**(15–39).

[16] Bazilevs Y, Calo V, Cottrell J, Evans J, Hughes T, Lipton S, Scott M, Sederberg T. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering* 2010; **199**(5–8):229–263.

[17] Xu G, Mourrain B, Duvigneau R, Galligo A. Parametrization of computational domain in isogeometric analysis: methods and comparison. *Computer Methods in Applied Mechanics and Engineering* 2011; **200**(23-24).

[18] Xu G, Mourrain B, Duvigneau R, Galligo A. Analysis-suitable volume parameterization of multi-block computational domain in isogeometric analysis. *Computer Aided Design* 2013; **45**(2).

[19] Speleers H, Manni C. Optimizing domain parameterization in isogeometric analysis based on Powell–Sabin splines. *Journal of Computational and Applied Mathematics* 2015; **289**:68–86.

[20] Johannessen KA, Kvamsdal T, Dokken T. Isogeometric analysis using LR B-Splines. *Computer Methods in Applied Mechanics and Engineering* 2014; **269**:471–514.

[21] Aigner M, Heinrich C, Jüttler B, Pilgerstorfer E, Simeon B, Vuong AV. Swept volume parameterization for isogeometric analysis. *13th IMA International Conference on Mathematics of Surfaces*, 2009; 19–44.

[22] Engvall L, Evans J. Isogeometric triangular bernstein-bézier discretizations: automatic mesh generation and geometrically exact finite-element analysis. *Computer Methods in Applied Mechanics and Engineering* 2016; (304):378–407.

[23] Engvall L, Evans J. Isogeometric unstructured tetrahedral and mixed-element bernstein-bézier discretizations. *Computer Methods in Applied Mechanics and Engineering* 2017; (319):83–123.

[24] Jaxon N, Qian X. isogeometric analysis on triangulations. *Computer Aided Design* 2014; (46):45–57.

[25] Xia S, Wang X, Qian X. Continuity and convergence in rational triangular bézier spline based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering* 2015; (297):292–324.

[26] Xia S, Qian X. Isogeometric analysis with bézier tetrahedra. *Computer Methods in Applied Mechanics and Engineering* 2017; (316):782–816.

[27] Zienkiewicz OC, Morice P. *The finite element method in engineering science*. McGraw-Hill, 1971.

[28] Michoski C, Chan J, Engvall L, Evans J. Foundations of the blended isogeometric discontinuous Galerkin (BIDG) method. *Computer Methods in Applied Mechanics and Engineering* 2016; **305**:658–681.

[29] Duvigneau R. Isogeometric analysis for compressible flows using a Discontinuous Galerkin method. *Computer Methods in Applied Mechanics and Engineering* 2018; **333**(443–461).

[30] Farin G. *Curves and Surfaces for Computer-Aided Geometric Design*. Academic Press, 1989.

[31] Piegl L, Tiller W. *The NURBS book*. Springer-Verlag, 1995.

[32] De Boor C. *A Practical Guide to Splines*. Springer Verlag, 1978.

[33] Dokken T. Locally refined splines. *Non-Standard Numerical Methods for PDEs, Pavia*, 2010.

[34] Xu G, Mourrain B, Galligo A, Duvigneau R. Constructing analysis-suitable parameterization of computational domain from cad boundary by variational harmonic method. *J. Comput. Physics* November 2013; **252**.

[35] Cottrell J, Hughes T, Reali A. Studies of refinement and continuity in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering* 2007; (196):4160–4183.

[36] Hesthaven JS, Warburton T. *Nodal Discontinuous Galerkin Methods*. Springer, 2008.

[37] Cockburn B. *High-Order Methods for Computational Physics*, chap. Discontinuous Galerkin Methods for Convection-Dominated Problems. Springer, 1999.

[38] Batten P, Clarke N, Lambert C, Causon M. On the choice of wavespeeds for the hllc Riemann solver. *SIAM J. Sci. Comput.* November 1997; **18**(6):1553–1570.

[39] Toro E. *Riemann Solvers and Numerical Methods for Fluid Dynamics*. Springer Berlin Heidelberg, 1997.

[40] Cockburn B, Shu CW. The local discontinuous galerkin method for time-dependent convection-diffusion systems. *SIAM Journal of Num. An.* 1998; **35**(6):2440–2463.

[41] Mengaldo G, Grazia DD, Witherden F, Farrington A, Vincent P, Sherwin S, Peiro J. A guide to the implementation of boundary conditions in compact high-order methods for compressible aerodynamics. *7th AIAA Theoretical Fluid Mechanics Conference*, 2014.

[42] Persson PO, Peraire J. Sub-cell shock capturing for discontinuous galerkin methods. *AIAA Paper 2006-112*, 2006.

[43] Barter GE, Darmofal DL. Shock capturing with PDE-based artificial viscosity for DGFEM: Part i. formulation. *J. Comput. Physics* 2010; **229**(5):1810–1827.

[44] Leicht T, Hartmann R. Anisotropic mesh refinement for discontinuous galerkin methods in two-dimensional aerodynamic flow simulations. *Int. J. for Numerical Methods in Fluids* 2008; **56**(11):2111–2138.

[45] Papoutsakis A, Sazhin SS, Begg S, Danaila I, Luddens F. An efficient adaptive mesh refinement (AMR) algorithm for the discontinuous galerkin method: Applications for the computation of compressible two-phase flows. *J. Comput. Physics* 2018; **363**:399–427.

[46] Sun S, Wheeler MF. Anisotropic and dynamic mesh adaptation for discontinuous galerkin methods applied to reactive transport. *Computer Methods in Applied Mechanics and Engineering* 2006; **195**(25–28):3382–3405.