Shape sensitivity analysis in aerodynamics using an isogeometric Discontinuous Galerkin method

Maxime Stauffert, Régis Duvigneau

Shape sensitivity analysis in aerodynamics using an isogeometric Discontinuous Galerkin method

M. Stauffert* R. Duvigneau†

Abstract

The sensitivity equation method aims at estimating the derivative of the solution of partial differential equations with respect to a parameter of interest. The objective of this work is to investigate the ability of an isogeometric Discontinuous Galerkin (DG) method to evaluate accurately sensitivities with respect to shape parameters originating from Computer-Aided Design (CAD), in the context of compressible aerodynamics. The isogeometric DG method relies on Non-Uniform Rational B-Spline representations, which allow to define a high-order numerical scheme for Euler equations, fully consistent with CAD geometries. We detail how this formulation can be exploited to construct an efficient and accurate approach to evaluate shape sensitivities. A particular attention is paid to the treatment of boundary conditions for sensitivities, which are more tedious in the case of geometrical parameters. The proposed methodology is first verified on a test-case with analytical solution and then applied to a more demanding problem, that concerns the flow around an airfoil with its camber as shape parameter.

1 Introduction

Sensitivity Analysis (SA) refers to the study of how a model output is influenced by a change of an input parameter. It is therefore a classical topic in engineering, in the perspective of optimization of the performance of a complex system, quantification of the output uncertainty according to the knowledge of input statistics, etc. This task can be achieved in several ways, which depend on the nature of the model of interest, the amplitude of the parameter perturbation or its stochastic / deterministic origin [15]. In this work, we focus on models governed by Partial Differential Equations (PDEs), and more specifically on non-linear hyperbolic systems of conservation laws. Such models are employed in tranport problems, like in fluid mechanics, wave propagation or even crowd dynamics. In this context, the current study concerns the estimation of the derivative of a PDE solution with respect to an input parameter. Thus, SA is considered here as a local approach and only small pertubations of the parameter are under consideration. Obviously, we make the assumption that the PDE solution is regular enough to be differentiated. Note that it is not always the case with non-linear hyperbolic systems, as studied in [1, 12]. Two different approaches can be carried out to estimate the derivative of a PDE model with respect to an input parameter: the adjoint and the sensitivity equation methods. In the former, a single functional output can be differentiated with respect to several input parameters with a computational cost independent from the number of parameters. In the latter, the whole solution field can be differentiated with respect to a single parameter, for a similar cost. As consequence, the adjoint approach is typically suited to optimization involving a large number of design parameters. On the contrary, the sensitivity approach is suited to the propagation of uncertainty of an input parameter to the whole solution field. Both can be performed at the continuous level (*differentiate-then-discretize*) or at the discrete level (*discretize-then-differentiate*). The adjoint method has been studied in depth in several works, from the pioneer developements using the continuous [26, 16] or discrete [2, 23] approach, and

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

applied to optimization in several disciplines [5, 17, 18]. However, less attention has been paid to the sensitivity equation method, although it has been pointed out that this approach can be powerful for uncertainty propagation [30, 31] or low-cost evaluation of neighboring solutions [10]. The current work focuses on this methodology.

The Sensitivity Equation Method (SEM) consists in solving a new system of PDEs, for each parameter of interest, obtained by differentiating the original PDE model. After resolution of the resulting system, one obtains as solution the derivative of the solution fields with respect to the parameter. As mentionned above, there are two ways to derive a numerical scheme for the sensitivity equations, the *discretize-then-differentiate* approach and the *differentiate-then-discretize* one. The first approach is referred as the Discrete SEM (DSEM) [20] and can be carried out using Automatic Differentiation (AD) software [23, 27], which can generate the computational code automatically and thus facilitate the implementation. However, the main issue concerning the DSEM is related to the fact that the mesh is involved in the *discretize* step and, consequently, some mesh derivative terms appear in the *differentiate* step. Alternatively, we prefer to use the Continuous SEM (CSEM) [3], for which the *differentiate* step is achieved on the continuous equations, thus no mesh derivatives need to be computed and the approach can be considered as more flexible. For a deeper comparison between DSEM and CSEM, one refers to [4] and [19].

Nevertheless, a difficulty arises in the CSEM when the parameter of interest controls the geometry of the computational domain. In this case, the boundary conditions for the sensitivity are not known exactly but depend on the gradients of the solution at the boundary, which can be tedious to estimate accurately and significantly impacts the accuracy of the sensitivity fields. To remedy to this situation, some specific techniques have been introduced, such as a local projection reconstruction [33, 24] or a least square reconstruction in the neighborhood of the boundary [9]. Alternatively, a Lagrangian formulation can also be adopted, but its implementation may be rather complex in practice [6]. Note that a non-intrusive implementation for a shape parameter is proposed in [21].

The objective of the current work is to investigate the use of the CSEM for shape parameters in the context of an isogeometric Discontinuous Galerkin (DG) framework. IsoGeometric Analysis (IGA) has been introduced in [14] as a way to solve PDE systems consistently with Computer-Aided Design (CAD) representations. The core idea of IGA is to use the same Non-Uniform Rational B-Spline (NURBS) basis to represent both the geometry and the solution, in a standard Finite-Element Method (FEM). Indeed, NURBS are widely used in CAD systems to define complex geometries [25]. As consequence, IGA allows a seamless integration of CAD and FEM. Recently, IGA has been extended to DG methods, for a better numerical treatment of hyperbolic systems of conservation laws [7, 22, 28, 32]. The current study is conducted in this framework. As shown in previous works [7, 8], the NURBS-based DG scheme allows to solve hyperbolic systems with a high accuracy and is fully consistent with CAD representations. Therefore, we intend to evaluate its potentiality for sensitivity analysis including shape parameters derived from CAD representations. The content of this paper is organized as follows: we first present the principle of the CSEM, with a particular emphasis on the treatment of shape parameters. Then, we summarize the main properties of NURBS representations and describe the NURBS-based DG scheme for compressible Euler equations. In a third part, we derive the sensitivity equations. Finally, we demonstrate the capabilities of the proposed approach for a problem with an analytical solution first, and for a more demanding application then, which involves the flow around an airfoil with its camber as shape parameter.

2 Principle of continuous sensitivity analysis

2.1 General formulation

For the sake of genericity, we consider first that the governing equations can be expressed as an hyperbolic system of conservation laws, that writes in the physical domain Ω as:

$$\partial_t \mathbf{U} + \nabla \cdot \mathbf{F}(\mathbf{U}) = \mathbf{0}, \tag{1}$$

where $\mathbf{U}\colon (\mathbf{x},t) \in \Omega \times \mathbb{R}^+ \mapsto \mathbf{U}(\mathbf{x},t) \in \mathbb{R}^n$ is the vector of state variables at any point and any time and $\mathbf{F}\colon \mathbf{U} \in \mathbb{R}^n \mapsto \mathbf{F}(\mathbf{U}) \in (\mathbb{R}^n)^d$ the corresponding physical flux. The integers d and n are respectively the dimension of Ω (which means $\Omega \subset \mathbb{R}^d$) and the number of state variables (or equations). We assume the system to be hyperbolic, thus the Jacobian matrix of the flux $\partial_{\mathbf{U}}\mathbf{F}$ is symmetric positive definite. Finally, this system is associated with boundary conditions, which can be of Dirichlet type: $\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{D}},t) = \mathbf{U}_\mathrm{D}$ on the boundary Γ_D, or Neumann type: $\partial_{\mathbf{n}}\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{N}},t) = \delta\mathbf{U}_\mathrm{N}$ on Γ_N, with $\partial\Omega =: \Gamma = \Gamma_\mathrm{D} \cup \Gamma_\mathrm{N}$.

We consider a parameter of the problem $a \in \mathbb{R}$, which impacts the solution. As consequence, the state variables depend now on a, which is denoted $\mathbf{U}(x,t;a)$. The objective of continuous sensitivity analysis is to estimate the derivative of the solution with respect to the parameter a. This quantity is referred as the sensitivity of $\mathbf{U}$ and denoted:

$$\mathbf{U}' = \partial_a \mathbf{U}. \tag{2}$$

We assume obviously that the state vector is regular enough with respect to a, so that its sensitivity exists. Theoretical fundations can be found in [1]. We differentiate now the state equation (1) with respect to a in order to obtain a new system governing the sensitivity. All the above functions $\mathbf{U}$, $\mathbf{F}$, $\mathbf{U}_\mathrm{D}$ and $\mathbf{U}_\mathrm{N}$ depend *a priori* on a. Moreover, in the case of smooth solutions, one can commute ∂_a with ∂_t and ∇. Note that this commutation is usually not possible in presence of discontinuities in the solution. For a study of sensitivity analysis in the context of discontinuous solutions, one can refer to [12]. Finally, the differentiation of the state equation (1) with respect to a yields therefore:

$$\partial_a\left(\partial_t\mathbf{U}\right) + \partial_a\left(\nabla\cdot\mathbf{F}\left(\mathbf{U}\right)\right) = \partial_t\left(\partial_a\mathbf{U}\right) + \nabla\cdot\left(\partial_a\left[\mathbf{F}\left(\mathbf{U}\right)\right]\right) = \mathbf{0},$$

or, after simplification:

$$\partial_t\mathbf{U}' + \nabla\cdot\left(\mathbf{F}'\left(\mathbf{U},\mathbf{U}'\right)\right) = \mathbf{0}, \tag{3}$$

where $\mathbf{F}'$ is the sensitivity flux, i.e. the derivative of the physical flux with respect to a. One can notice that the sensitivity equation (3) has the same conservative form as the state equation (1). Moreover, the sensitivity flux can be expressed as:

$$\mathbf{F}' = \partial_a\left[\mathbf{F}\left(\mathbf{U}\right)\right] = \partial_{\mathbf{U}}\mathbf{F}\cdot\partial_a\mathbf{U} = \partial_{\mathbf{U}}\mathbf{F}\cdot\mathbf{U}'.$$

As consequence, the sensitivity flux is linear with respect to $\mathbf{U}'$ and its Jacobian is identical to the Jacobian of the physical flux:

$$\partial_{\mathbf{U}'}\mathbf{F}' = \partial_{\mathbf{U}}\mathbf{F}$$

This property is important because it allows to establish that the sensitivity equation is (linearly) hyperbolic, with the same characteristic speeds as the state equation. This system can be solved independently from the state equation, after evaluation of the solution $\mathbf{U}$. However, as will be detailed later, it may be convenient to gather the two systems into a global conservation law for a concatenated variable:

$$\partial_t\mathbf{W} + \nabla\cdot\mathbf{G}\left(\mathbf{W}\right) = \mathbf{0}, \tag{4}$$

with:

$$\mathbf{W} = \begin{pmatrix}\mathbf{U}\\ \partial_a\mathbf{U}\end{pmatrix} = \begin{pmatrix}\mathbf{U}\\ \mathbf{U}'\end{pmatrix} \qquad \text{and} \qquad \mathbf{G}\left(\mathbf{W}\right) = \begin{pmatrix}\mathbf{F}(\mathbf{U})\\ \partial_a\left[\mathbf{F}(\mathbf{U})\right]\end{pmatrix} = \begin{pmatrix}\mathbf{F}(\mathbf{U})\\ \mathbf{F}'(\mathbf{U},\mathbf{U}')\end{pmatrix}.$$

The boundary conditions for the sensitivity can now be derived, by differentiating the boundary conditions for the state variables. However, two different cases should be considered, depending if the parameter a controls the geometry of the domain, i.e. $\Omega(a)$ (shape parameter), or not (value parameter).

2.2 Boundary conditions for value parameters

If the parameter a only influences the value of the boundary condition, i.e. $\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{D}}, t; a) = \mathbf{U}_\mathrm{D}$ on Γ_D or $\partial_\mathbf{n}\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{N}}, t; a) = \delta\mathbf{U}_\mathrm{N}$ on Γ_N, one simply obtains by differentiation:

$$\mathbf{U}'(\mathbf{x}_{\Gamma_\mathrm{D}}, t; a) = \partial_a\left(\mathbf{U}_\mathrm{D}\right),$$

for the sensitivity Dirichlet condition and:

$$\partial_\mathbf{n}\mathbf{U}'(\mathbf{x}_{\Gamma_\mathrm{N}}, t; a) = \partial_a\left(\delta\mathbf{U}_\mathrm{N}\right),$$

for the sensitivity Neumann condition.

2.3 Boundary conditions for shape parameters

If the geometry of the domain depends on a, the derivation of the boundary conditions for the sensitivity is more complex, because it should account for the spatial dependency of the boundary condition [9]. The Dirichlet boundary condition for the state variables thus writes $\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{D}}(a), t; a) = \mathbf{U}_\mathrm{D}$, with $\mathbf{x}_{\Gamma_\mathrm{D}}(a)$ the coordinates of the boundary curve parameterized by a. Its differentiation yields:

$$\mathbf{U}'(\mathbf{x}_{\Gamma_\mathrm{D}}, t; a) + \nabla\mathbf{U}\cdot\partial_a\mathbf{x}_{\Gamma_\mathrm{D}} = D_a\left(\mathbf{U}_\mathrm{D}\right),$$

where $D_a\left(\mathbf{U}_\mathrm{D}\right)$ represents the total derivative of $\mathbf{U}_\mathrm{D}$ with respect to a. The Dirichlet condition for the sensitivity is then:

$$\mathbf{U}'(\mathbf{x}_{\Gamma_\mathrm{D}}, t; a) = D_a\left(\mathbf{U}_\mathrm{D}\right) - \nabla\mathbf{U}\cdot\partial_a\mathbf{x}_{\Gamma_\mathrm{D}}. \tag{5}$$

The Neumann condition is even more tedious because it implies the normal derivative of the state variables, which depends on the geometry of the boundary:

$$\partial_\mathbf{n}\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{N}}(a), t; a) = \nabla\mathbf{U}(\mathbf{x}_{\Gamma_\mathrm{N}}(a), t; a)\cdot\mathbf{n}(a) = \delta\mathbf{U}_\mathrm{N}.$$

The differentiation of this condition yields finally:

$$\partial_\mathbf{n}\mathbf{U}'(\mathbf{x}_{\Gamma_\mathrm{N}}, t; a) = D_a\left(\delta\mathbf{U}_\mathrm{N}\right) - \left(\nabla^2\mathbf{U}\cdot\partial_a\mathbf{x}_{\Gamma_\mathrm{N}}\right)\cdot\mathbf{n} - \nabla\mathbf{U}\cdot\partial_a\mathbf{n}(a). \tag{6}$$

It is worth noticing that the additional terms make the boundary conditions for the sensitivity dependent on the derivatives of the solution. They reflect that any change in the parameter a modifies the location of the boundary condition. In the fortcoming sections, we describe how this general approach can be implemented in the context of an isogeometric DG method applied to compressible flow equations.

3 NURBS-based Discontinuous Galerkin method

3.1 Compressible Euler equations

We consider now as conservation laws the compressible Euler equations, which can be expressed for the 2D case as:

$$\partial_t\begin{pmatrix}\rho\\ \rho u\\ \rho v\\ E\end{pmatrix} + \nabla\cdot\begin{pmatrix}\rho u & \rho v\\ \rho u^2+p & \rho uv\\ \rho uv & \rho v^2+p\\ u(E+p) & v(E+p)\end{pmatrix} = \begin{pmatrix}0\\0\\0\\0\end{pmatrix}, \tag{7}$$

where ρ is the density, $\mathbf{u} = (u, v)$ the velocity vector and E the total energy per unit of volume. The pressure p is given by the perfect gas law:

$$p = (\gamma-1)\left(E - \frac{1}{2}\rho(u^2+v^2)\right), \tag{8}$$

with $\gamma = 1.4$. A classical slip condition is prescribed at the wall boundaries, which consists in imposing in a weak sense $\mathbf{u}\cdot\mathbf{n} = un_x + vn_y = 0$. At inlet or outlet boundaries, state variables are weakly imposed, according to the local flow direction.

3.2 Discontinuous Galerkin discretization

For any state variable $w \in \{\rho, \rho u, \rho v, E\}$, a weak formulation is obtained by multiplying equation (7) by a test function φ and integrating over the physical domain Ω:

$$\int_\Omega (\partial_t w + \nabla \cdot \mathbf{f})\, \varphi \, \mathrm{d}\Omega = 0, \tag{9}$$

where $\mathbf{f}$ is the physical flux associated to w. By integrating by parts, one obtains classically:

$$\int_\Omega (\partial_t w)\, \varphi \, \mathrm{d}\Omega - \int_\Omega \mathbf{f} \cdot \nabla\varphi \, \mathrm{d}\Omega + \int_\Gamma (\mathbf{f} \cdot \mathbf{n})\, \varphi \, \mathrm{d}\Gamma = 0. \tag{10}$$

The domain is then splitted in a set of elements $\Omega = \cup_{j=1}^{N} \Omega_j$, for which a local functional basis of size $p+1$ is defined:

$$\mathcal{P}^j = \{\mathbf{x} \in \Omega_j \mapsto P_{i,j}(\mathbf{x}) \in \mathbb{R}, i \in \{0, \dots, p\}\}.$$

The components of the solution $\mathbf{U}$ are finally decomposed into a finite space of piecewise-defined functions:

$$w(\mathbf{x}, t) = \sum_{i=0}^{p} w_{i,j}(t) P_{i,j}(\mathbf{x}) \qquad \forall \mathbf{x} \in \Omega_j \quad \forall t > 0,$$

where $w_{i,j}$ are the time dependent coefficients to be solved. If one chooses φ such that $\varphi|_{\Omega_j} = P_{k,j}$ and $\varphi|_{\Omega \setminus \Omega_j} = 0$, then equation (10) yields:

$$\sum_{i=0}^{p} \partial_t w_{i,j} \int_{\Omega_j} P_{i,j} P_{k,j} \, \mathrm{d}\Omega_j = \int_{\Omega_j} \mathbf{f} \cdot \nabla P_{k,j} \, \mathrm{d}\Omega_j - \int_{\Gamma_j} (\mathbf{f}^\star \cdot \mathbf{n})\, P_{k,j} \, \mathrm{d}\Gamma_j. \tag{11}$$

The left-hand side is the temporal term multiplied by the so-called mass matrix, whereas the right-hand side corresponds to the volumic and surfacic integrals. The last term exhibits a numerical flux function $\mathbf{f}^\star$ instead of the physical flux $\mathbf{f}$, due to the fact that the solution is not uniquely defined at element interfaces. In the present work, the Harten-Lax-van Leer (HLL) Riemann solver [29] is employed to estimate this quantity. All integrals are computed numerically using Gauss-Lagrange quadratures. Finally, the time-integration is carried out using the explicit forth-order Runge-Kutta method. All details can be found in [7, 8]. While most DG methods rely on piecewise polynomial functions, the present work adopts bases defined from CAD for a consistent shape description, as explained in the next section.

3.3 NURBS basis

In most modern CAD software, shapes are defined thanks to NURBS representations, which are the rational extensions of B-Splines [25]. These functions are polynomials of degree p defined recursively, with an initialization as piecewise constant functions:

$$N_i^0(\xi) = \begin{cases} 1, & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0, & \text{otherwise,} \end{cases} \tag{12}$$

and by the following recursive formula:

$$N_i^p(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_i^{p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1}^{p-1}(\xi), \tag{13}$$

where $\Xi = \{\xi_0, \dots, \xi_{n+p}\}$ is called the knot vector, defined in the parametric domain. Multivariate NURBS functions are then obtained by using tensor products of B-Spline functions multiplied by weight coefficients. For example, uni- and bi-variate NURBS functions are defined as:

$$R_i^p(\xi) = \frac{\omega_i N_i^p(\xi)}{\sum_{j=0}^{n-1} \omega_j N_j^p(\xi)} \qquad R_{i_1,i_2}^p(\xi, \eta) = \frac{\omega_{i_1,i_2} N_{i_1}^p(\xi) N_{i_2}^p(\eta)}{\sum_{j_1=0}^{n-1} \sum_{j_2=0}^{n-1} \omega_{j_1,j_2} N_{j_1}^p(\xi) N_{j_2}^p(\eta)}.$$

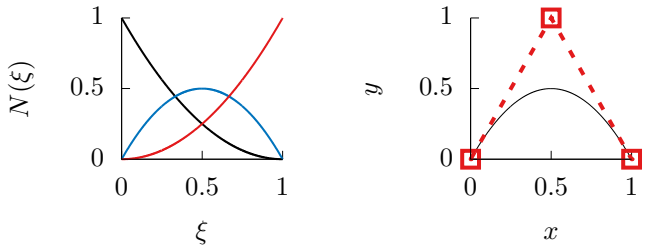


Figure 1: B-spline basis on left, curve with control points on right.

Note that the specific case $n = p + 1$ yields rational Bézier representations.

Curves, surfaces and volumes can then be introduced by associating coefficients $\{\mathbf{B}_0, \dots, \mathbf{B}_{n-1}\}$, referred as control points, to the basis functions. For instance, NURBS curves and surfaces are defined as:

$$C(\xi) = \sum_{i=0}^{n-1} R_i^p(\xi)\mathbf{B}_i \qquad S(\xi,\eta) = \sum_{i_1=0}^{n-1}\sum_{i_2=0}^{n-1} R_{i_1,i_2}^p(\xi,\eta)\mathbf{B}_{i_1,i_2}. \tag{14}$$

This representation is illustrated by a simple example: $p = 2$, $n = 3$, $\Xi = \{0,0,0,1,1,1\}$ and $\mathcal{B} = \left\{ \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0.5 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix} \right\}$. B-Spline functions are constructed as follows $\forall \xi \in [0,1]$:

$$\begin{array}{lllll} N_0^0(\xi) = 0, & N_1^0(\xi) = 0, & N_2^0(\xi) = 1, & N_3^0(\xi) = 0, & N_4^0(\xi) = 0, \\ N_0^1(\xi) = 0, & N_1^1(\xi) = (1-\xi), & N_2^1(\xi) = \xi, & N_3^1(\xi) = 0, & \\ N_0^2(\xi) = (1-\xi)^2, & N_1^2(\xi) = 2(1-\xi)\xi, & N_2^2(\xi) = \xi^2, & & \end{array}$$

and the corresponding curve writes (we suppose here unitary weights):

$$C(\xi) = (1-\xi)^2 \begin{pmatrix} 0 \\ 0 \end{pmatrix} + 2(1-\xi)\xi \begin{pmatrix} 0.5 \\ 1 \end{pmatrix} + \xi^2 \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \xi \\ 2(1-\xi)\xi \end{pmatrix}.$$

In Figure (1), the B-spline basis is shown on the left and the curve on the right.

NURBS representations have several interesting properties, that can be found in [25]. Among them, we underline the knot insertion property, which is a key ingredient to apply these bases to a DG discretization. According to this property, one can insert a knot $\widetilde{\xi} \in [\xi_k, \xi_{k+1})$ for any k, without altering the geometry of a curve:

$$C(\xi) = \sum_{i=0}^{n-1} N_i^p(\xi)\mathbf{B}_i = \sum_{i=0}^{n} \widetilde{N}_i^p(\xi)\widetilde{\mathbf{B}}_i,$$

where the B-spline basis $\widetilde{N}_i^p$, $i \in \{0, \dots, n\}$ corresponds to the knot vector:

$$\widetilde{\Xi} = \left\{ \xi_0, \dots, \xi_k, \widetilde{\xi}, \xi_{k+1}, \dots, \xi_{n+p} \right\},$$

and $\widetilde{\mathbf{B}}_i = \alpha_i \mathbf{B}_i + (1-\alpha_i)\mathbf{B}_{i-1}$ with:

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

Moreover, it can be shown [25] that the curve exhibits a regularity $\mathcal{C}^{(p-m)}$ at any knot of multiplicity m. Thus, by multiple knot insertion, one can generate a discontinuous B-spline basis, without

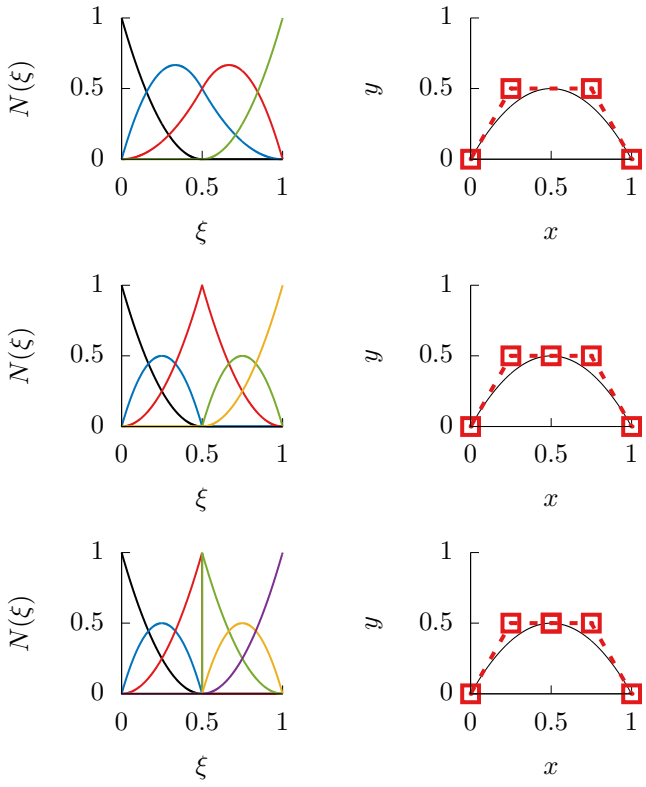


Figure 2: B-spline basis on left, curve with control points on right. Knot 0.5 respectively added 1 time on top, 2 times in the middle and 3 times on bottom.

changing the geometry of the curve C. This can be easily generalized to multi-variate NURBS. The resulting procedure is named Bézier extraction and allows to split any NURBS curve, surface or volume into a set of corresponding rational Bézier curves, surfaces or volumes without modifying the geometry. This property is illustrated on the simple example shown above, by inserting 3 times the knot 0.5, in Figure (2).

3.4 Isogeometric Discontinuous Galerkin method

This approach simply consists in employing NURBS representations, described in section 3.3, in the framework of the general DG discretization presented in section 3.2. This choice is motivated by the will to achieve CAD-consistent flow simulations. As shown in previous works [7, 8], the resulting method exhibits a higher accuracy with respect to a classical DG approach, based on piecewise linear grids, in presence of curved boundaries.

The approach is described here for two-dimensional problems, but it could be extended to other dimensions without difficulties. The core idea of the method is to replace the polynomial functions (like Lagrange basis) and the corresponding geometrical elements (like triangles) traditionally used in DG, by NURBS basis functions and NURBS surfaces. However, NURBS surfaces employed in CAD to represent shapes are usually too large to be considered directly as elements in a DG framework. Therefore, the proposed approach consists firstly in constructing a CAD-consistent grid, by splitting the NURBS surfaces that define originally the computational domain into a set of smaller rational Bézier surfaces, that will be considered as elements in the DG discretization. We underline again that this transformation modifies only the basis employed to describe the geometry, and not the geometry itself. From practical point of view, this phase relies on the Bézier extraction procedure explained previously, all the details being provided in [7].

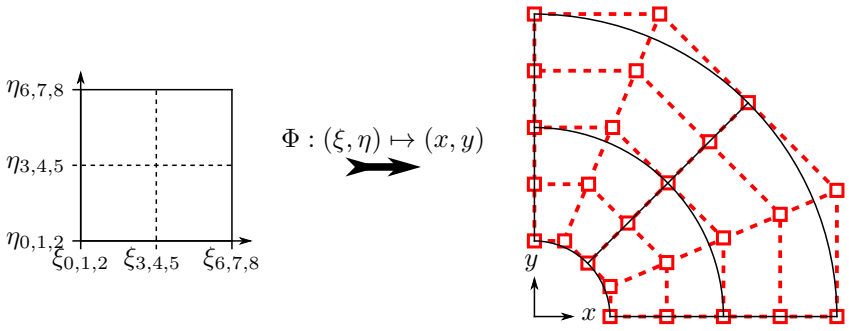


Figure 3: Parametric domain on left, physical domain using NURBS on right.

This is illustrated in Figure (3), for a geometry corresponding to a quarter of cylinder. It is originally defined using a single bi-quadratic NURBS surface. Thanks to the Bézier extraction procedure, it is splitted into four rational Bézier surfaces without altering the geometry. The figure depicts the splitted parametric domain and the rational Bézier elements with their control points.

As a result, the computational domain is represented by a set of rational Bézier surfaces, each of them being defined by its own rational Bézier basis, which enables to generate discontinuous solutions at element interfaces. The second phase consists thus in writing the general DG formulation in equation (11) for a rational Bézier surface $\hat{\Omega}_j$ and its corresponding rational Bézier basis (for clarity, we omit the subscript j for the functions):

$$\sum_{i_1=0}^{p}\sum_{i_2=0}^{p} \partial_t w_{i,j} \int_{\hat{\Omega}_j} R^p_{i_1,i_2} R^p_{k_1,k_2} \,\mathrm{d}\hat{\Omega}_j = \int_{\hat{\Omega}_j} \mathbf{f}\cdot\nabla R^p_{k_1,k_2} \,\mathrm{d}\hat{\Omega}_j - \int_{\hat{\Gamma}_j} (\mathbf{f}^\star\cdot\mathbf{n})\, R^p_{k_1,k_2} \,\mathrm{d}\hat{\Gamma}_j. \tag{15}$$

Note that these integrations are achieved in the parametric domain, as all quantities are now defined in this space. All other numerical ingredients implied in the DG method are maintained, only the bases being modified. We underline that the resulting approach can be referred as isogeometric, because the geometry as well as the solution are defined using the same CAD-consistent basis. In the following section, we explain how this approach can be employed for CAD-consistent sensitivity analysis.

4 Shape sensitivity analysis for compressible flows

4.1 Sensitivity equations

In the following, we still denote by $'$ the derivative with respect to the parameter a. If one differentiates the Euler system (7), one obtains:

$$\begin{cases} \partial_t \rho' + \partial_x(\rho u)' & + \partial_y(\rho v)' & = 0, \\ \partial_t(\rho u)' + \partial_x\left[(\rho u)'u + (\rho u)u' + p'\right] & + \partial_y\left[(\rho u)'v + (\rho u)v'\right] & = 0, \\ \partial_t(\rho v)' + \partial_x\left[(\rho v)'u + (\rho v)u'\right] & + \partial_y\left[(\rho v)'v + (\rho v)v' + p'\right] & = 0, \\ \partial_t E' + \partial_x\left[u'\left(E+p\right) + u\left(E'+p'\right)\right] & + \partial_y\left[v'\left(E+p\right) + v\left(E'+p'\right)\right] & = 0, \end{cases} \tag{16}$$

with:

$$p' = (\gamma-1)\left[E' - \frac{1}{2}\left((\rho u)'u + (\rho u)u' + (\rho v)'v + (\rho v)v'\right)\right].$$

The conservative form established in equation (4) can therefore be retrieved by using the following global variables and fluxes:

$$\mathbf{W} = \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ E \\ \rho' \\ (\rho u)' \\ (\rho v)' \\ E' \end{pmatrix} \qquad \mathbf{G}(\mathbf{W}) = \begin{pmatrix} \rho u & \rho v \\ \rho u^2 + p & \rho uv \\ \rho uv & \rho v^2 + p \\ u(E+p) & v(E+p) \\ (\rho u)' & (\rho v)' \\ (\rho u)'u + (\rho u)u' + p' & (\rho u)'v + (\rho u)v' \\ (\rho v)'u + (\rho v)u' & (\rho v)'v + (\rho v)v' + p' \\ u'(E+p) + u(E'+p') & v'(E+p) + v(E'+p') \end{pmatrix}. \tag{17}$$

Thus, sensitivity equations can be easily solved, simultaneously to the state equations, by using the DG discretization given in equation (15). The computations are achieved component by component, with the same explicit time-marching procedure. As the state and the sensitivity equations have the same flux Jacobian, the same stability conditions hold. We underline that this global formalism facilitates significantly the extension of an existing solver to sensitivity analysis.

4.2 Implementation of sensitivity boundary conditions

In the context of the DG method employed here, the boundary conditions are weakly imposed, by prescribing a suitable flux on boundary interfaces. We detail below the case of wall and inlet / outlet boundary conditions for the state and sensitivity variables. In what follows, we suppose that the sensitivity parameter controls the geometry of the wall boundary. For all cases, the normal physical flux on the boundary writes:

$$\mathbf{F} \cdot \mathbf{n} = \begin{pmatrix} \rho(\mathbf{u} \cdot \mathbf{n}) \\ (\rho u)(\mathbf{u} \cdot \mathbf{n}) + p n_x \\ (\rho v)(\mathbf{u} \cdot \mathbf{n}) + p n_y \\ (\mathbf{u} \cdot \mathbf{n})(E + p) \end{pmatrix}, \tag{18}$$

with $\mathbf{n} = (n_x, n_y)$ the unitary normal vector pointing outwards. Now, if one evaluates the scalar product of the physical sensitivity flux defined in equation (16) with the normal vector on the boundary, one obtains:

$$\mathbf{F}' \cdot \mathbf{n} = \begin{pmatrix} [\rho' \mathbf{u} + \rho \mathbf{u}'] \cdot \mathbf{n} \\ \left[(\rho u)' \mathbf{u} + (\rho u)\mathbf{u}'\right] \cdot \mathbf{n} + p' n_x \\ \left[(\rho v)' \mathbf{u} + (\rho v)\mathbf{u}'\right] \cdot \mathbf{n} + p' n_y \\ \left[\mathbf{u}'(E+p) + \mathbf{u}(E+p)'\right] \cdot \mathbf{n} \end{pmatrix}. \tag{19}$$

These generic expressions are then adapted to the different boundary conditions. For the wall condition $\mathbf{u} \cdot \mathbf{n} = 0$, the boundary flux for the state variables becomes:

$$\mathbf{F} \cdot \mathbf{n}|_{\text{wall}} = \begin{pmatrix} 0 \\ p n_x \\ p n_y \\ 0 \end{pmatrix}, \tag{20}$$

The differentiation of the condition $\mathbf{u} \cdot \mathbf{n} = 0$ yields the wall boundary condition for the sensitivity variables:

$$(\mathbf{u} \cdot \mathbf{n})' = (\nabla \mathbf{u} \cdot \mathbf{x}') \cdot \mathbf{n} + \mathbf{u}' \cdot \mathbf{n} + \mathbf{u} \cdot \mathbf{n}' = 0,$$

where $\mathbf{x}' = \partial_a \mathbf{x}$ represents the derivative of the boundary coordinates with respect to the shape parameter. In the context of the isogeometric DG, this implies the derivative of the boundary curve:

$$\mathbf{x}' = \partial_a C(\xi) = \sum_{i=0}^{n-1} R_i^p(\xi) \partial_a \mathbf{B}_i, \tag{21}$$

where $\partial_a \mathbf{B}_i$ is the derivative of the control point of index i with respect to the parameter a. This quantity is obviously case dependent and its estimation will be detailed in forthcoming application sections. As explained in section 2.3, the gradient of the velocity appears during the differentiation because the condition is written at the boundary location, which depends on the parameter a. Accounting for both wall conditions, the normal sensitivity flux at wall boundary finally writes:

$$\mathbf{F}' \cdot \mathbf{n}|_{\text{wall}} = -\begin{pmatrix} \rho\left[(\nabla \mathbf{u} \cdot \mathbf{x}') \cdot \mathbf{n} + \mathbf{u} \cdot \mathbf{n}'\right] \\ (\rho u)\left[(\nabla \mathbf{u} \cdot \mathbf{x}') \cdot \mathbf{n} + \mathbf{u} \cdot \mathbf{n}'\right] - p' n_x \\ (\rho v)\left[(\nabla \mathbf{u} \cdot \mathbf{x}') \cdot \mathbf{n} + \mathbf{u} \cdot \mathbf{n}'\right] - p' n_y \\ \left[(\nabla \mathbf{u} \cdot \mathbf{x}') \cdot \mathbf{n} + \mathbf{u} \cdot \mathbf{n}'\right](E + p) \end{pmatrix}. \tag{22}$$

Regarding the inlet / outlet conditions, a similar approach is carried out to derive sensitivity boundary conditions. It is however simpler, because we make the assumption that the geometry of the boundary does not depend on a. Several formulations exist for inlet / outlet conditions in the context of DG discretizations. In this work, we impose at the boundary the physical flux corresponding to so-called boundary state variables $\mathbf{U}_{\text{bnd}}$, which are computed according to the interior $\mathbf{U}_{\text{in}}$ and the user-defined exterior $\mathbf{U}_{\text{out}}$ states. Formally, it writes:

$$\mathbf{F} \cdot \mathbf{n}|_{\text{io}} = \mathbf{F}(\mathbf{U}_{\text{bnd}}) \cdot \mathbf{n}, \tag{23}$$

with:

$$\mathbf{U}_{\text{bnd}} = \Phi(\mathbf{U}_{\text{in}}, \mathbf{U}_{\text{out}}), \tag{24}$$

The operator Φ defines how the boundary states result from a mix of interior and exterior quantities. The sensitivity flux at inlet / outlet boundary is obtained by adopting a similar approach:

$$\mathbf{F}' \cdot \mathbf{n}|_{\text{io}} = \mathbf{F}'(\mathbf{U}_{\text{bnd}}, \mathbf{U}'_{\text{bnd}}) \cdot \mathbf{n}, \tag{25}$$

where $\mathbf{U}'_{\text{bnd}}$ is the sensitivity of the boundary state variables:

$$\mathbf{U}'_{\text{bnd}} = \partial_{\mathbf{U}_{\text{in}}} \Phi(\mathbf{U}_{\text{in}}, \mathbf{U}_{\text{out}}) \mathbf{U}'_{\text{in}} + \partial_{\mathbf{U}_{\text{out}}} \Phi(\mathbf{U}_{\text{in}}, \mathbf{U}_{\text{out}}) \mathbf{U}'_{\text{out}}. \tag{26}$$

To be more practical, we consider below a simple case: at inlet, one imposes the density ρ_{out} and velocity vector $\mathbf{u}_{\text{out}}$, whereas the pressure is obtained from the interior domain p_{in}. The boundary state variables at inlet are then:

$$\mathbf{U}_{\text{inlet}} = \begin{pmatrix} \rho_{\text{out}} \\ \rho_{\text{out}} u_{\text{out}} \\ \rho_{\text{out}} v_{\text{out}} \\ \frac{p_{\text{in}}}{\gamma - 1} + \frac{1}{2}\rho_{\text{out}}(u_{\text{out}}^2 + v_{\text{out}}^2) \end{pmatrix}. \tag{27}$$

According to equations (23) and (18), the boundary flux at inlet is:

$$\mathbf{F} \cdot \mathbf{n}|_{\text{inlet}} = \begin{pmatrix} \rho_{\text{out}} \left(\mathbf{u}_{\text{out}} \cdot \mathbf{n}\right) \\ \rho_{\text{out}} u_{\text{out}} \left(\mathbf{u}_{\text{out}} \cdot \mathbf{n}\right) + p_{\text{in}} n_x \\ \rho_{\text{out}} v_{\text{out}} \left(\mathbf{u}_{\text{out}} \cdot \mathbf{n}\right) + p_{\text{in}} n_y \\ \left(\mathbf{u}_{\text{out}} \cdot \mathbf{n}\right) \left(\frac{\gamma}{\gamma - 1} p_{\text{in}} + \frac{1}{2}\rho_{\text{out}}(u_{\text{out}}^2 + v_{\text{out}}^2)\right) \end{pmatrix}. \tag{28}$$

If the exterior states do not depend on a, the term $\mathbf{U}_{\text{out}}{}'$ vanishes in equation (26) and the sensitivity of the boundary states is simply:

$$\mathbf{U}'_{\text{inlet}} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ \frac{p'_{\text{in}}}{\gamma - 1} \end{pmatrix}. \tag{29}$$

The sensitivity flux is then derived according to equation (25) and (19):

$$\mathbf{F}' \cdot \mathbf{n}|_{\text{inlet}} = \begin{pmatrix} 0 \\ p_{\text{in}}{}' n_x \\ p_{\text{in}}{}' n_y \\ \left(\mathbf{u}_{\text{out}} \cdot \mathbf{n}\right) \frac{\gamma}{\gamma - 1} p'_{\text{in}} \end{pmatrix}. \tag{30}$$

Other types of inlet / outlet conditions can be treated with the same approach, for instance when one uses Riemann invariants to estimate boundary state variables. In the case the geometry of the inlet / outlet boundary depends on a, additional terms including the solution gradients appear in equation (26), expressing the fact that the location of the boundary condition depends on a, as already explained for the wall condition.

5 Applications

Two test-cases are considered to assess the proposed approach. First, the flow between two cylinders is studied, with a shape parameter defined as the radius of the inner cylinder. This problem allows a rigorous verification of the results because the solution for the flow and the sensitivity is known analytically. Then, one considers the flow around an airfoil, which is originally symmetric, while a shape parameter controlling its camber is introduced. Contrary to the previous case, no analytical solution is provided for this problem. The results will therefore be validated by comparison of neighbouring solutions, obtained by linear extrapolation based on sensitivities and full non-linear resolution.

5.1 Flow between two cylinders with radius change

This problem, which has a steady analytical solution for the Euler system, is described in [13]. The flow is confined between two concentric cylinders of radii $r_0 < r_1$. The exact solution is given by:

$$\begin{cases} \rho = \rho_0, \\ \rho u = -\rho_0 \sin(\theta) u_\theta, \\ \rho v = \rho_0 \cos(\theta) u_\theta, \\ E = \dfrac{p}{\gamma - 1} + \dfrac{\rho_0}{2} u_\theta^2, \end{cases}$$

where

$$u_\theta(r) = \frac{0.2}{\left(\frac{r_1}{r_0} - \frac{r_0}{r_1}\right)} \left(\frac{r_1}{r} - \frac{r}{r_1}\right),$$

$$p(r) = \rho_0 \left[1 + \frac{0.2^2}{\left(\frac{r_1}{r_0} - \frac{r_0}{r_1}\right)^2} \left(\frac{r^2}{2r_1^2} - 2\ln(r) - \frac{r_1^2}{2r^2}\right)\right],$$

and the polar coordinates are given by $r = \sqrt{x^2 + y^2}$ and $\theta = \arctan\left(\frac{y}{x}\right)$.

One defines the shape parameter as the inner radius $a = r_0$. The analytical solution for the sensitivity can easily be obtained by differentiating the solution for the flow with respect to r_0, yielding:

$$\begin{cases} \rho' = 0, \\ (\rho u)' = -\rho_0 \sin(\theta)\, \partial_{r_0} u_\theta(r), \\ (\rho v)' = \rho_0 \cos(\theta)\, \partial_{r_0} u_\theta(r), \\ E' = \dfrac{\partial_{r_0} p(r)}{\gamma - 1} + \rho_0\, u_\theta(r)\, \partial_{r_0} u_\theta(r), \end{cases}$$

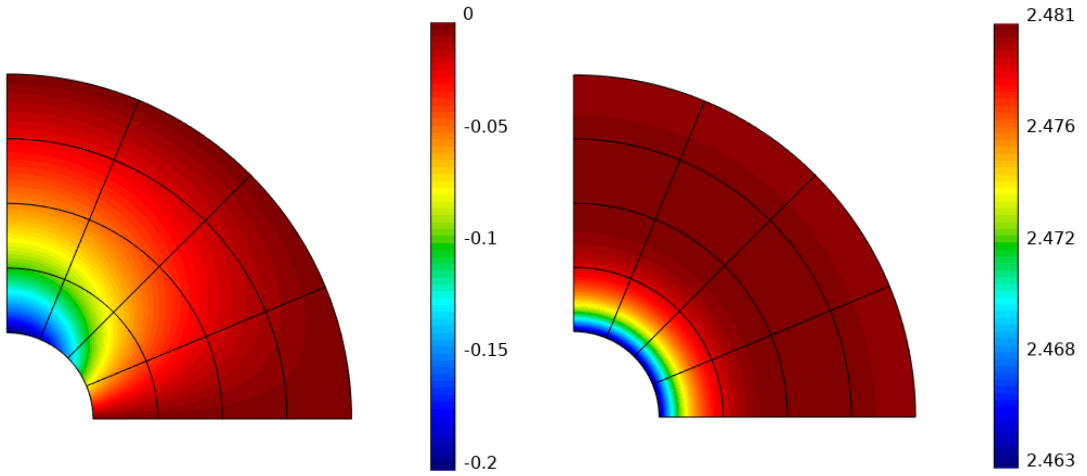
where:

$$\partial_{r_0} u_\theta(r) = u_\theta(r) \left(\frac{r_1}{r_0^2} + \frac{1}{r_1}\right)$$

$$\partial_{r_0} p(r) = \frac{2\,(p(r) - \rho_0)}{\left(\frac{r_1}{r_0} - \frac{r_0}{r_1}\right)} \left(\frac{r_1}{r_0^2} + \frac{1}{r_1}\right).$$

The geometry of the inner cylinder $\mathbf{x} = (r_0 \cos(\theta), r_0 \sin(\theta))$ depends linearly on the parameter r_0, therefore the sensitivity of the boundary coordinates is simply $\mathbf{x}' = \mathbf{x}/r_0$.

Due to the symmetries of the problem, only a quarter of the domain is simulated. As consequence, the computational domain can be represented exactly using a single quadratic NURBS surface. As explained in section 3.4 and illustrated in Figure (3), a CAD-consistent rational Bézier grid is generated by multiple knot insertion.



(a) Momentum-x. (b) Energy.

Figure 4: Flow solution for the cylinder test-case.

Wall boundary conditions are imposed on the two cylinders, whereas the flux corresponding to the exact solution is imposed at inlet and outlet boundaries. By splitting the computational domain in different numbers of rational Bézier surfaces, one generates a sequence of grids, from coarse to fine, which allows to carry out a rigorous assessment of the computations, for different basis degrees.

The solutions for the flow are provided in Figure (4), for a basis of degree five and a grid composed of four by four elements. The evolution of the error in L^2-norm is represented in Figures(5), (6) and (7), respectively for ρ, ρu and E, for degree p ranging from one to four. As can be seen, the numerical solution converges to the analytical one, with a convergence rate of value $p+1$. One can notice a significant improvement beween degree one and two, due to the fact that for degree two and above the geometry is exactly taken into account during the resolution, whereas for degree one the cylinder geometry is only approximated. This can also explain why the asymptotic convergence rate is more difficult to reach for the linear basis. Then, the accuracy of the sensitivity fields is similarly studied. The error for ρ', $(\rho u)'$ and E' are depicted in Figures (8), (9) and (10). One can notice that, contrary to the flow variables, a convergence rate of value p is only obtained for the sensitivity. This loss of accuracy is in fact due to the presence of the gradient term $\nabla \mathbf{u}$ in the boundary conditions. Indeed, this quantity is computed with an accuracy of order p, by differentiation of the numerical solution of order $p+1$. The sub-optimal convergence rate for the sensitivity was therefore expected. Thus, it is critical to use a high-accurate discretization method for the flow, in order to derive an accurate boundary condition for the sensitivity. This issue was studied in depth in [9, 10, 11], in the context of a finite-element method.

Finally, this test-case is an opportunity to quantify the benefit of using an isogeometric formulation, for the computation of both the flow and the shape sensitivity. In this perspective, we perform again the accuracy study, but using a sequence of (classical) piecewise linear grids generated by linearization of the grids previously used. The evolution of the error is plotted in Figures (11) and (12) for the first momentum component and its sensitivity. As can be observed, the error for the flow variables is deeply impacted by the geometrical error and the convergence rate is limited

to a value lower than two, whatever the basis employed. As already observed in [7], it is due to the presence of spurious rarefaction waves, which are generated at each boundary vertex because of the artificial change of wall slope. This illustrates how the geometry can be critical when using high-order schemes. This impact is larger for the sensitivity variables, because their boundary condition at the wall involves the velocity gradients. Then, the errors in the velocity fields at the boundary propagate to the sensitivity boundary conditions. Theses errors are even stronger when the basis degree increases. One can conclude that the computation of accurate shape sensitivity requires a high-order treatment of the geometry.

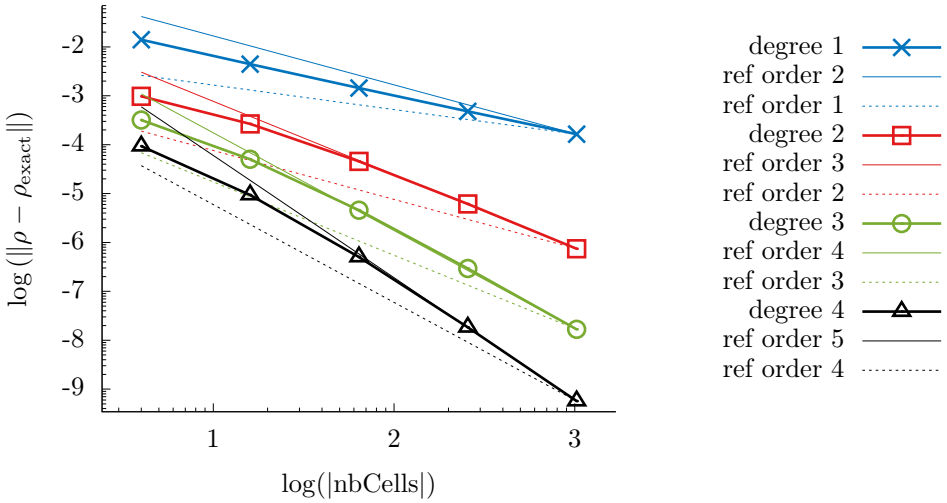


Figure 5: Errors for variable ρ between computed and exact solutions.

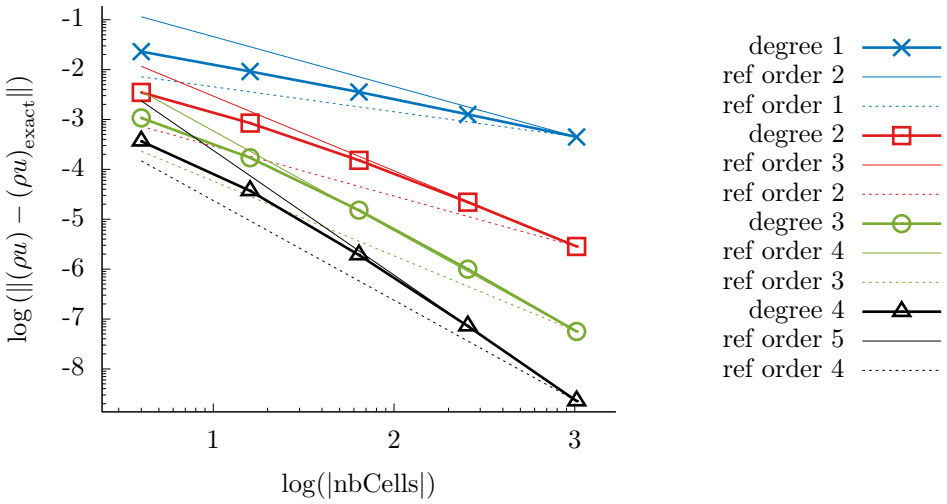


Figure 6: Errors for variable (ρu) between computed and exact solutions.

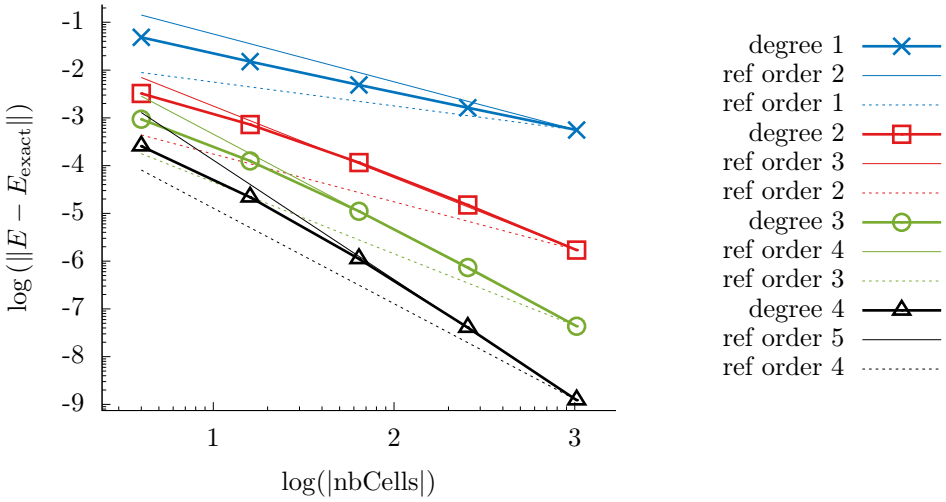


Figure 7: Errors for variable E between computed and exact solutions.

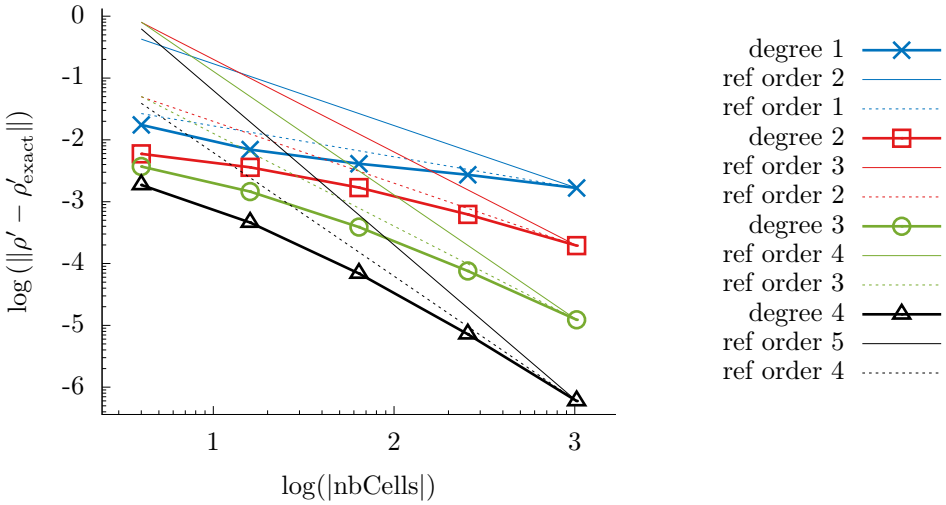


Figure 8: Errors for variable ρ' between computed and exact solutions.

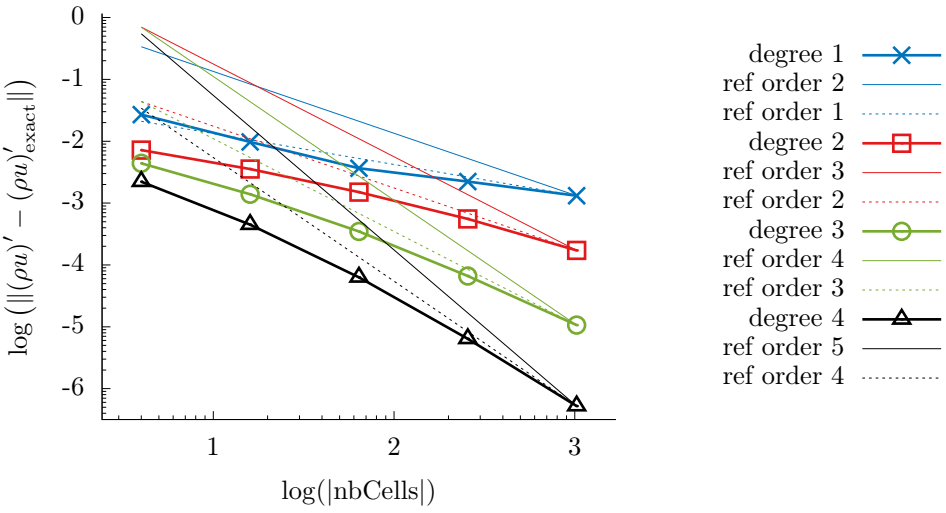


Figure 9: Errors for variable $(\rho u)'$ between computed and exact solutions.

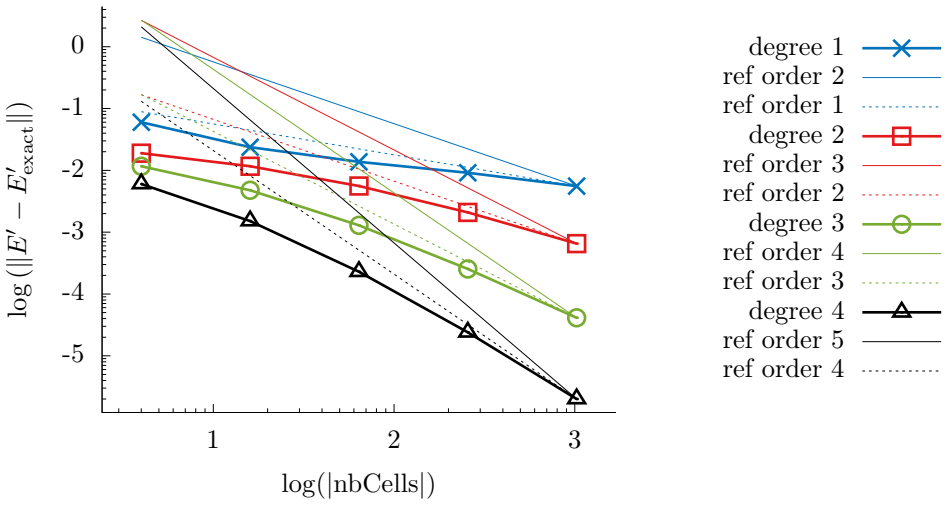


Figure 10: Errors for variable E' between computed and exact solutions.

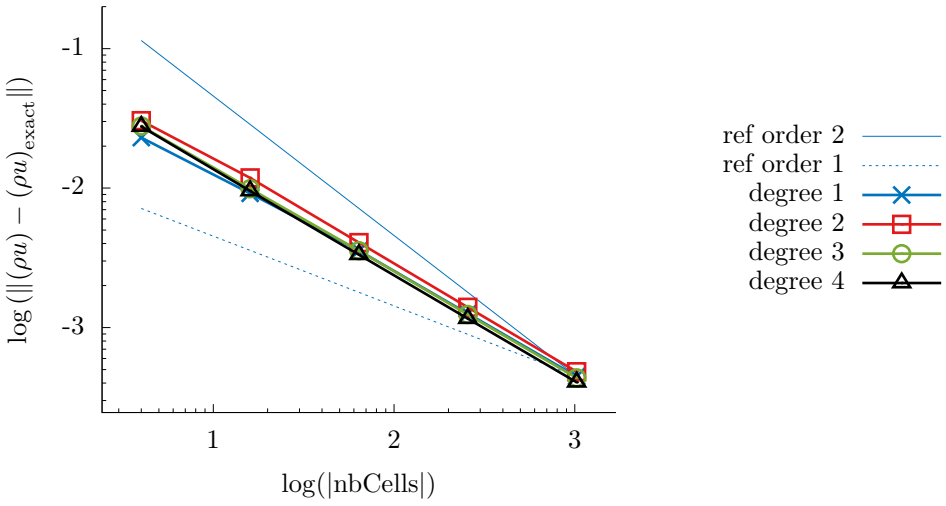


Figure 11: Errors for variable (ρu) between computed and exact solutions on linear meshes.

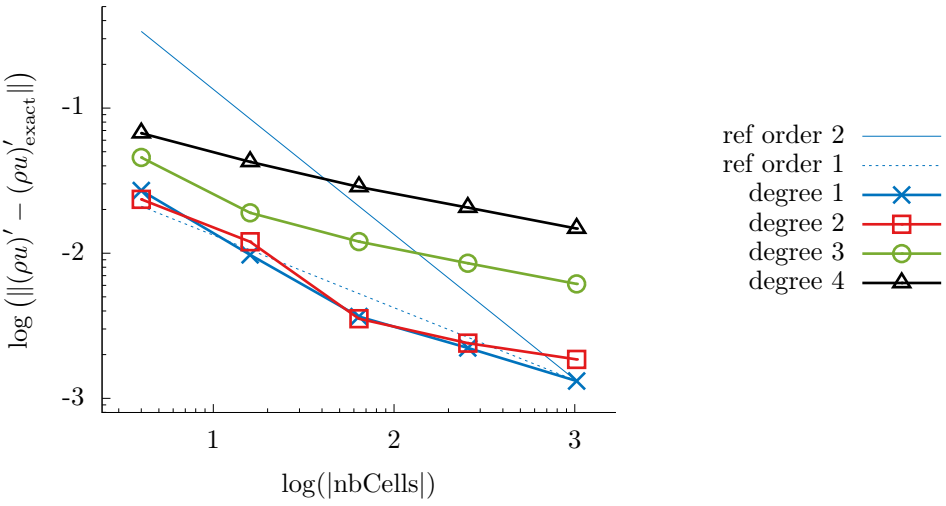


Figure 12: Errors for variable $(\rho u)'$ between computed and exact solutions on linear meshes.

5.2 Flow around an airfoil with camber change

We consider as second test-case the flow around an airfoil in subsonic regime. The objective is to study the ability of the proposed approach to evaluate the sensitivity of the flow with respect to the camber of the airfoil. The first step is to define the airfoil parametric shape and the camber parameter in this context. In this perspective, we adopt a NURBS representation that mimics the four-digit NACA airfoils, which are defined by setting the maximum camber m, the position p of this maximum and the maximum airfoil thickness t. We recall that, for all NACA airfoils, the coordinates of the upper (resp. lower) curve $(x_{\rm U}, y_{\rm U})$ (resp. $(x_{\rm L}, y_{\rm L})$) are given by:

$$x_{\rm U} = x - y_t \sin\theta, \qquad y_{\rm U} = y_c + y_t \cos\theta,$$
$$x_{\rm L} = x + y_t \sin\theta, \qquad y_{\rm L} = y_c - y_t \cos\theta,$$

where $x \in [0, c]$ is the position along the chord of length c, y_c the mean camber is defined as:

$$y_c = \begin{cases} \frac{m}{p^2}\left(2p\left(\frac{x}{c}\right) - \left(\frac{x}{c}\right)^2\right), & \text{if } 0 \le x \le pc, \\ \frac{m}{(1-p)^2}\left((1-2p) + 2p\left(\frac{x}{c}\right) - \left(\frac{x}{c}\right)^2\right), & \text{if } pc \le x \le c, \end{cases}$$

the angle θ equals $\arctan\left(\frac{{\rm d}y_c}{{\rm d}x}\right)$ and, finally, y_t is the half thickness:

$$y_t = 5t\left(0.2969\sqrt{\frac{x}{c}} - 0.1260\left(\frac{x}{c}\right) - 0.3516\left(\frac{x}{c}\right)^2 + 0.2843\left(\frac{x}{c}\right)^2 - 0.1036\left(\frac{x}{c}\right)^4\right).$$

We are not using directly this definition because it does not match the NURBS representation used in this work. Nevertheless, we are considering as airfoil definition a very close approximation of the NACA 0012 airfoil ($m = 0$, $p = 0$, $t = 0.12c$), resulting from a least-squares fitting of the lower and upper NACA curves by two cubic NURBS curves. The Figure (13) shows a coarse grid around the airfoil constructed by using 16×4 cubic rational Bézier elements, obtained after splitting of the two fitted NURBS curves. The coordinates of the control points of the two NURBS curves defining the airfoil are denoted $\mathbf{x}_{\rm U}^{0012}$ and $\mathbf{x}_{\rm L}^{0012}$. To define the shape parameter controling the airfoil camber, we introduce a linear mapping between these two curves and the ones approximating the NACA 2412 airfoil ($m = 0.2c$, $p = 0.4c$, $t = 0.12c$), obtained with a similar least-squares fitting procedure and characterized by a camber of value $m = 0.2c$. The coordinates of the control points of the two NURBS curves defining this second airfoil are denoted $\mathbf{x}_{\rm U}^{2412}$ and $\mathbf{x}_{\rm L}^{2412}$. This allows to generate a set of NURBS-based airfoils of different camber values, whose control points have the coordinates $\mathbf{x}_{\rm U} = (1-\omega)\mathbf{x}_{\rm U}^{0012} + \omega\mathbf{x}_{\rm U}^{2412}$ and $\mathbf{x}_{\rm L} = (1-\omega)\mathbf{x}_{\rm L}^{0012} + \omega\mathbf{x}_{\rm L}^{2412}$. The weighting coefficient ω controls the global camber of the airfoil, as m for NACA airfoils, it is thus chosen as shape parameter for this study: $a = \omega$. As consequence, the sensitivities of the boundary coordinates are simply $\mathbf{x}'_{\rm U} = \mathbf{x}_{\rm U}^{2412} - \mathbf{x}_{\rm U}^{0012}$ and $\mathbf{x}'_{\rm L} = \mathbf{x}_{\rm L}^{2412} - \mathbf{x}_{\rm L}^{0012}$.

The flow and the sensitivity with respect to ω are computed for a subsonic regime correponding to a far-field Mach number $M_\infty = 0.2$. The mesh employed is similar to the one shown in Figure (13), but it is more extended spatially and includes a refined area around the airfoil. It counts 6818 cubic elements. Wall conditions are imposed at the airfoil curves, whereas inlet / outlet conditions are prescribed at the outer boundary, with exterior state defined as $\rho_{\rm out} = \gamma$, $u_{\rm out} = M_\infty$, $v_{\rm out} = 0$ and $p_{\rm out} = 1$.

In Figure (14) are plotted the density field around the airfoil and its sensitivity with respect to ω, for the symmetric airfoil configuration corresponding to $\omega = \omega_0 = 0$. Since the airfoil is symmetric and has no incidence with respect to the far-field flow, the density field is also symmetric. However, the sensitivity field is asymmetrical, because the sensitivity of the boundary coordinates involved in the boundary conditions is asymmetrical.

To assess these computations, we propose to use the sensitivity to extrapolate linearly the flow fields from the original symmetric airfoil to a sequence of airfoils with non-zero camber values. This is achieved in a straightforward way by combining the discrete flow variables $\mathbf{U}(\omega_0)$ and sensitivity variables $\mathbf{U}'(\omega_0)$ computed for the symmetric airfoil:

$$\mathbf{U}(\delta\omega)^{\rm lin} = \mathbf{U}(\omega_0) + \delta\omega\mathbf{U}'(\omega_0), \tag{31}$$

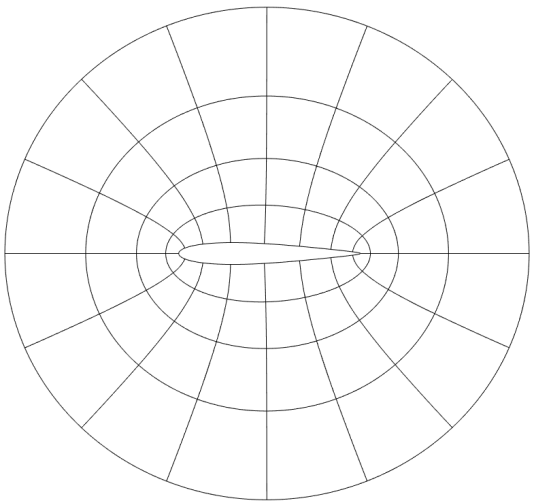
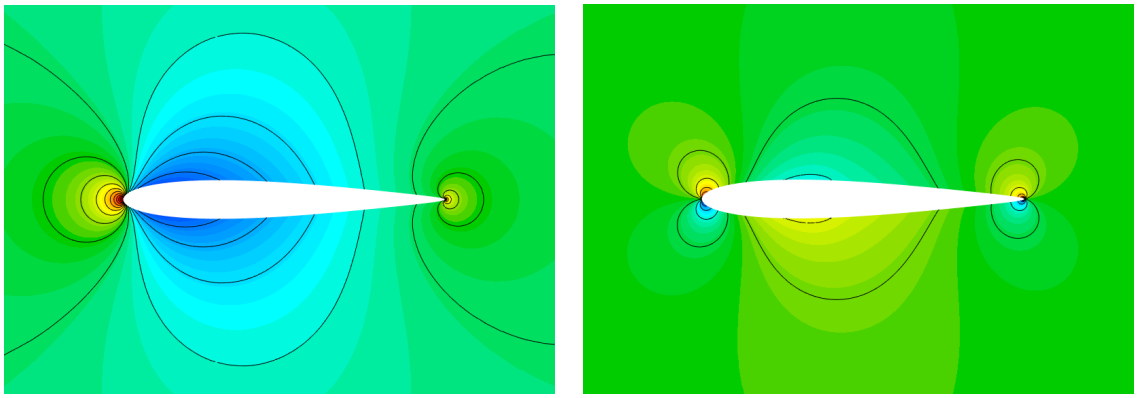


Figure 13: Example of cubic grid fitting the NACA 0012 airfoil.

where the parameter $\delta\omega$ represents the camber change and $\mathbf{U}^{\text{lin}}$ the predicted fields. According to this first-order Taylor expansion, the error in the extrapolated fields should be of order $\delta\omega^2$. Therefore, to validate the sensitivity computation, we solve the flow equations for a sequence of different camber values and plot the extrapolation error with respect to the perturbation $\delta\omega$ in logarithm scale, as shown in Figures 15, 16 and 17 for the variables ρ, (ρu) and E. As can be seen, a satisfactory agreement is obtained. It should be underlined that the theoretical order $\delta\omega^2$ is difficult to obtain in practice, because of the presence of discretization errors, which should be made negligeable with respect to the extrapolation error using a very fine grid and a high-accurate scheme. Moreover, the numerical evaluation of the error is tedious, because the two solutions (extrapolated one and resolved one) are achieved on different grids. Here, the grids corresponding to the airfoils with camber are generated using local deformations of the reference grid, thanks to the geometrical mapping described above. As consequence, the error curves presented in Figures 15, 16 and 17 are evaluated only in the areas of the computational domain that are identical for the two configurations. This may also explain the discrepancies between the error curves and the theoretical rates.



(a) Density around original arfoil.

(b) Density sensitivity around original airfoil.

Figure 14: Flow and sensitivity solution for original airfoil.

Finally, the potentiality of such linear extrapolations to estimate the flow for a neighboring shape at a low computational cost is demonstrated. A comparison of the density fields obtained by

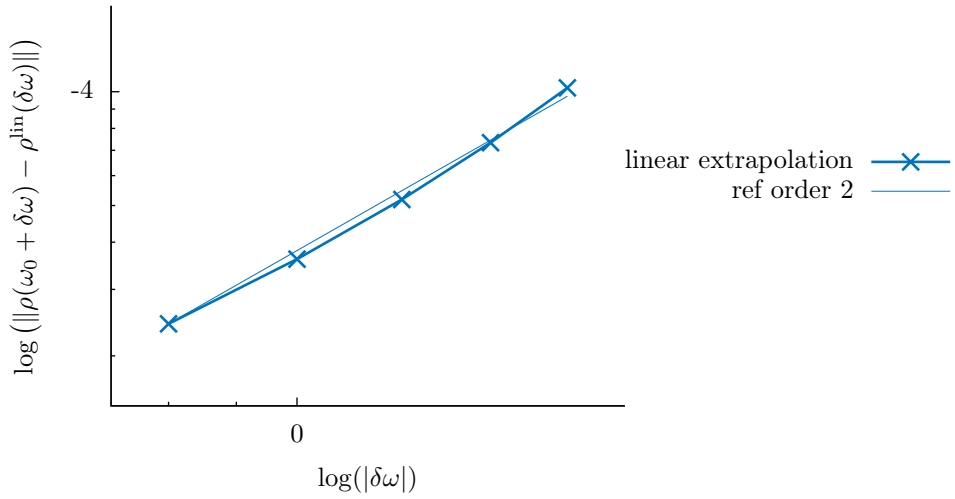


Figure 15: Extrapolation error for variable ρ.

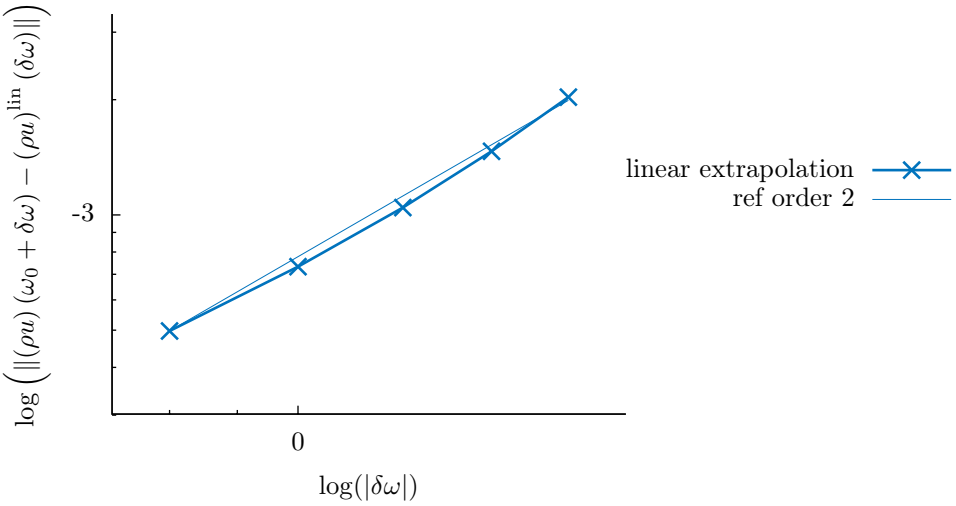


Figure 16: Extrapolation error for variable ρu.

extrapolation from the symmetric airfoil and full resolution using an airfoil with camber is shown in Figure 18, for a camber of value $\omega = 1$. Note that the extrapolated field is plotted for the original symmetric configuration, because it is evaluated on the corresponding computational domain. As consequence, it doe not make sens at some locations close to the airfoil boundary. The flow around the airfoil with camber is directly computed using a geometrical domain that fits the airfoil. As can be seen, the camber yields a significant change in the solution field, which is well predicted by the linear extrapolation. Although some discrepancies can be found, it shows that an accurate computation of the shape sensitivity allows to predict the characteristics of the flow for a quite important shape perturbation.

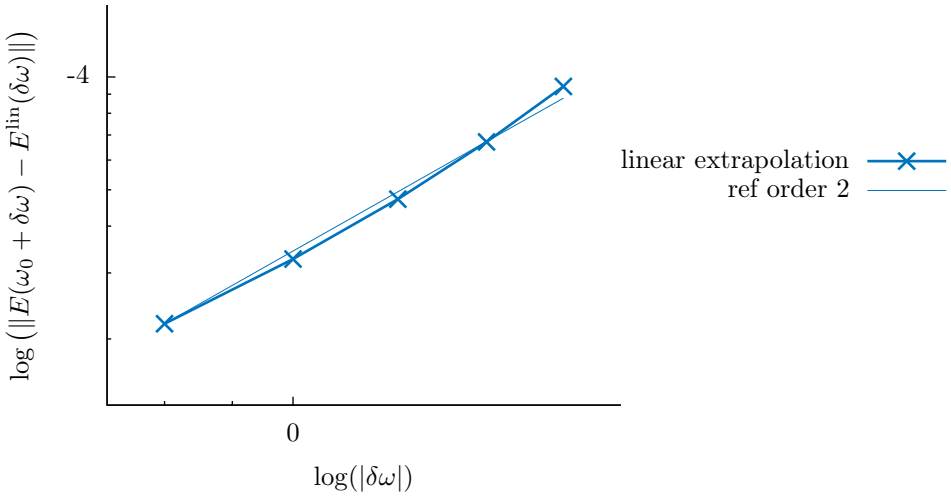
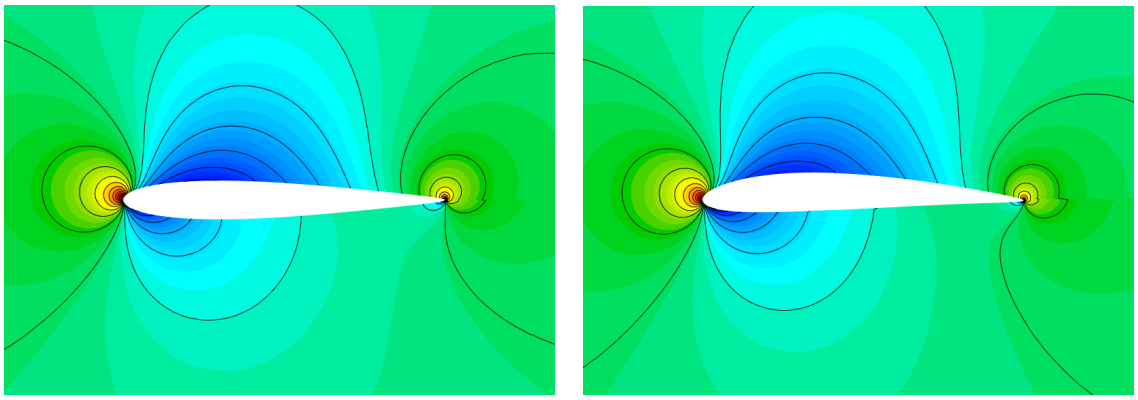


Figure 17: Extrapolation error for variable E.



(a) Extrapolation of the density field.

(b) Density around an airfoil with camber.

Figure 18: Comparison of extrapolated and solved density fields.

6 Conclusion

The objective of this work was to investigate the shape sensitivity analysis in the framework of an isogeometric DG method. After a summary of the NURBS-based discretization technique was provided, the extension to the resolution of the continuous sensitivity equations for the compressible Euler system was detailed. A particular emphasis was given to the description of the boundary conditions in the case of shape parameters. The resulting approach allows to solve in a compact form both the flow and sensitivity equations with a high-order CAD-consistent methodology.

A verification exercise was then carried out, for a test-case concerning the flow around two cylinders. It was demonstrated that a convergence rate of value $p + 1$ was obtained for the flow variables, whereas only a rate of value p was observed for sensitivity variables, due to the presence of flow gradients in the sensitivity boundary conditions. The critical impact of the use of high-order geometrical representations was reported. Then, a more complex problem was studied, regarding the flow around an airfoil with camber change. It was shown that accurate shape sensitivity computations allow to predict the flow change, even for a significant modification of the geometry.

Several issues remain to use such an approach for industrial applications. In particular, the difficulty to account for discontinuous solutions like shocks in the framework of the sensitivity equation method is well known. A possible approach was found in [12] for one-dimensional problems. Its extension to higher dimension is currently investigated.

References

[1] C. Bardos and O. Pironneau, *A formalism for the differentiation of conservation laws*, Compte rendu de l'Académie des Sciences, 335 (2002), pp. 839–845.

[2] O. Baysal and G. W. Burgreen, *Three-dimensional aerodynamic shape optimization using discrete sensitivity analysis*, AIAA Journal, 34 (1996), pp. 1761–1770.

[3] J. Borggaard and J. Burns, *A PDE sensitivity equation method for optimal aerodynamic design*, Journal of Computational Physics, 136 (1997), pp. 366–384.

[4] J. Borggaard and A. Verma, *On efficient solutions to the continuous sensitivity equation using automatic differentiation*, SIAM Journal on Scientific Computing, 22 (2000), pp. 39–62.

[5] A. Borzi and V. Schulz, *Computational Optimization of Systems Governed by Partial Differential Equations*, SIAM, 2012.

[6] L. Charlot, S. Etienne, and D. Pelletier, *A continuous lagrangian sensitivity equation method for incompressible flow*, Journal of Computational Physics, 231 (2012).

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

[8] R. Duvigneau, *CAD-consistent adaptive refinement using a NURBS-based Discontinuous Galerkin method*, International Journal for Numerical Methods in Fluids, (2020).

[9] R. Duvigneau and D. Pelletier, *On accurate boundary conditions for a shape sensitivity equation method*, Int. J. for Numerical Methods in Fluids, 50 (2006).

[10] R. Duvigneau and D. Pelletier, *A sensitivity equation method for fast evaluation of nearby flows and uncertainty analysis for shape parameters*, International Journal of Computational Fluid Dynamics, 20 (2006), pp. 497–512.

[11] R. Duvigneau, D. Pelletier, and J. Borggaard, *An improved continuous sensitivity equation method for optimal shape design in mixed convection*, Numerical Heat Transfer part B : Fundamentals, 50 (2006), pp. 1–24.

[12] C. Fiorini, C. Chalons, and R. Duvigneau, *A modified sensitivity equation method for the euler equations in presence of shocks*, Numerical Methods for Partial Differential Equations, (2019).

[13] J. S. Hesthaven and T. Warburton, *Nodal discontinuous Galerkin methods: algorithms, analysis, and applications*, Springer Science & Business Media, 2007.

[14] 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 (2005), pp. 4135–4195.

[15] B. Iooss and P. Lemaitre, *Uncertainty management in Simulation-Optimization of Complex Systems: Algorithms and Applications*, Springer, 2015, ch. A Review on Global Sensitivity Analysis Methods, pp. 101–122.

[16] A. Jameson, *Aerodynamic design via control theory*, Journal of Scientific Computing, 3 (1988), pp. 233–260.

[17] A. Jameson, L. Martinelli, and N. A. Pierce, *Optimum aerodynamic design using the Navier-Stokes equation*, Theorical and Computational Fluid Dynamics, 10 (1998), pp. 213–237.

[18] A. Keane and P. Nair, *Computational Approaches for Aerospace Design: The Pursuit of Excellence*, John-Wiley and Sons, 2005.

[19] M. Kleiber, T. D. Hien, H. Antúnez, and P. Kowalczyk, *Parameter sensitivity in nonlinear mechanics: Theory and finite element computations*, Wiley, 1997.

[20] V. Komkov, K. K. Choi, and E. J. Haug, *Design sensitivity analysis of structural systems*, vol. 177, Academic press, 1986.

[21] M. D. Kulkarni, R. A. Canfield, and M. J. Patil, *Nonintrusive continuum sensitivity analysis for fluid applications*, Journal of Computational Physics, 403 (2020), p. 109066.

[22] 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 (2016), pp. 658–681.

[23] B. Mohammadi and O. Pironneau, *Mesh adaptation and automatic differentiation in a cad-free framework for optimal shape*, International Journal for Numerical Methods in Engineering, 30 (1999), pp. 127–136.

[24] D. Pelletier, É. Turgeon, S. Etienne, and J. Borggaard, *Reliable sensitivity analysis via an adaptive sensitivity equation method*, in 3 rd AIAA Theoretical Fluid Mechanics Meeting, Saint Louis, MO, 2002.

[25] L. Piegl and W. Tiller, *The NURBS book*, Springer Science & Business Media, 2012.

[26] O. Pironneau, *Optimal shape design for elliptic systems*, Springer-Verlag, 1983.

[27] L. L. Sherman, A. C. Taylor III, L. L. Green, P. A. Newman, G. W. Hou, and V. M. Korivi, *First-and second-order aerodynamic sensitivity derivatives via automatic differentiation with incremental iterative methods*, Journal of Computational Physics, 129 (1996), pp. 307–331.

[28] A. Silveira, R. Moura, A. Silva, and M. Ortega, *Higher-order surface treatment for discontinuous Galerkin methods with applications to aerodynamics*, International Journal for Numerical Methods in Fluids, 79 (2015), pp. 323–342.

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

[30] É. Turgeon, D. Pelletier, and J. Borggaard, *Application of a sensitivity equation method to the $k-\epsilon$ model of turbulence*, in 15th AIAA Computational Fluid Dynamics Conference, Anaheim, CA, Jun. 2001.

[31] É. Turgeon, D. Pelletier, and J. Borggaard, *Sensitivity and uncertainty analysis for variable property flows*, in 39th AIAA Aerospace Sciences Meeting and Exhibit, Reno, NV, Jan. 2001.

[32] S. Yu, R. Feng, and T. Liu, *An isogeometric discontinuous galerkin method for euler equations*, Mathematical Methods and Models in Applied Sciences, 40 (2017).

[33] O. C. Zienkiewicz and J. Z. Zhu, *The superconvergent patch recovery and a posteriori error estimates. part 1: The recovery technique*, International Journal for Numerical Methods in Engineering, 33 (1992), pp. 1331–1364.