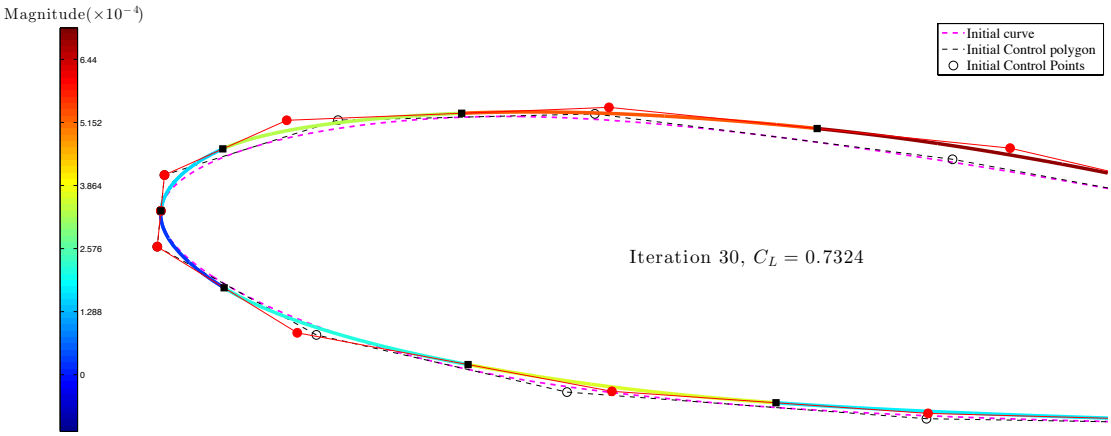
EIDÖGENSSISCHE TECHNISCHE HOCHSCHULE LAUSANNE
POLITECNICO FEDERALE DI LOSANNA
SWISS FEDERAL INSTITUTE OF TECHNOLOGY LAUSANNE

SCHOOL OF BASIC SCIENCES | MATHEMATICS INSTITUTE FOR ANALYSIS AND APPLICATIONS | Chair of Modeling & Scientific Computing

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

Shape Optimization of an Hydrofoil by Isogeometric Analysis

SEMESTER PROJECT (Fall 2013)
Department of Mathematics



Author:
Matthieu SIMEONI
matthieu.simeoni@epfl.ch

Under the supervision of:
Dr. Luca DEDE, CMCS, EPFL.
luca.dede@epfl.ch

HYDROS Multiplicateur d'énergies

CMCS

February 10, 2014

Abstract: In this work, we use Isogeometric Analysis as a framework for NURBS-based shape optimization procedure. We start by presenting geometrical representations by NURBS and some of their properties to design an hydrofoil. Then, we consider an irrotational flow around an hydrofoil and solve the Laplace problem in the stream function formulation. Finally, we perform the shape optimization of the hydrofoil by considering the stream function as the state problem and different objective functionals.

Table of Contents

Introduction

HYDROcontest 2014 is an international contest open to students of scientific and technology universities, organized and supported by Hydroptère Suisse SA in partnership with EPFL. The aim of the competition is to raise awareness of future engineers on the impact of maritime transport on the environment. In particular, the goal is to build and optimize a flying motorboat from a starter kit (see fig. 0.2) which allows fast transportation with reduced energy consumption. A competition among the motorboats designed by the participating schools will be held in July 2014 on the Geneva Lake.

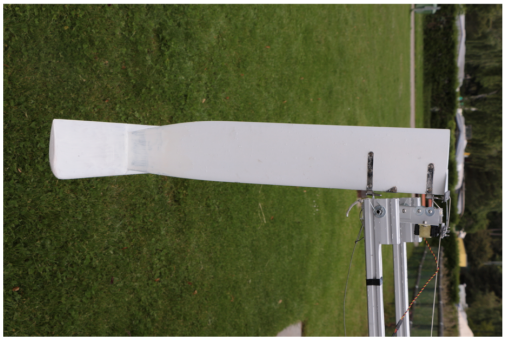
A team composed by students and nine laboratories of EPFL, under the coordination of the Transportation Center, is participating to the competition. Each laboratory contributes to the design, optimization, and realization of the motorboat for the competition.

In this context, this work aims at optimizing the shape of the hydrofoils of the boat (see fig. 0.2 (c) and (d)) for a better performance of the boat during the race. To this end, we propose a NURBS-based shape optimization procedure by using Isogeometric Analysis, which provides a unified framework for both design and analysis.

Figure 0.1: *Photography of the starting kit.* ©*Hydros*

(a) *The motorboat flying above the water*

(b) *Motorboat out of the water*



(c) *Rear hydrofoil*

(d) *Side hydrofoil*

Figure 0.2: *Photographies of the starting kit. ©Hydros*

Figure 0.3: *The EPFL students team*

CHAPTER

1 Geometric Design of the Domain and Mesh Generation by means of NURBS

In this chapter we introduce the Isogeometric Analysis philosophy and review some background concepts and basic properties of NURBS, that will play a key role in our study. This knowledge will help us in the design of our computational domain and in the mesh generation process.

Most of the results presented in this chapter are adapted from the Chapter 2 of Austin [1].

1 The Need for Isogeometric Analysis

Isogeometric analysis take birth in the wish of closing the increasing gap between the two worlds of engineering design and analysis. More specifically, **Isogeometric Analysis** lies at the intersection between **Computer-Aided Analysis (CAD)** and **Finite Element Analysis (FEA)**.

Definition 1.1 (*Computer-Aided Analysis*)

Computer-aided design (CAD) *is the use of various computer-based techniques (from the field of computational geometry) to assist in the creation, modification, analysis, or optimization of a design.*

Definition 1.2 (*Finite Element Analysis*)

Finite Element Analysis (FEA) *is a numerical method aiming to provide an approximate solution to any differential problem with prescribed boundary conditions.*
By breaking down the problem into smaller, more manageable (finite) elements, FEA seeks to reduce it to a linear system that can be solved using techniques from linear algebra.

Remark : Depending on the context, Finite Element Analysis can also be referred as Finite Element Method (FEM).

In the various industrial applications (aerospace, automotive, manufacturing, biomedical...), designers generate CAD files of the geometry of interest which then needs to be translated into an analysis-suitable geometry to be meshed and processed by FEA codes. However, while dealing with the same fundamental object, CAD and FEA have grew up separately, resulting in a lot of heterogeneity in the geometrical descriptions, complicating the interactions between the two domains (see fig. 1.2). As a result, the preparatory steps

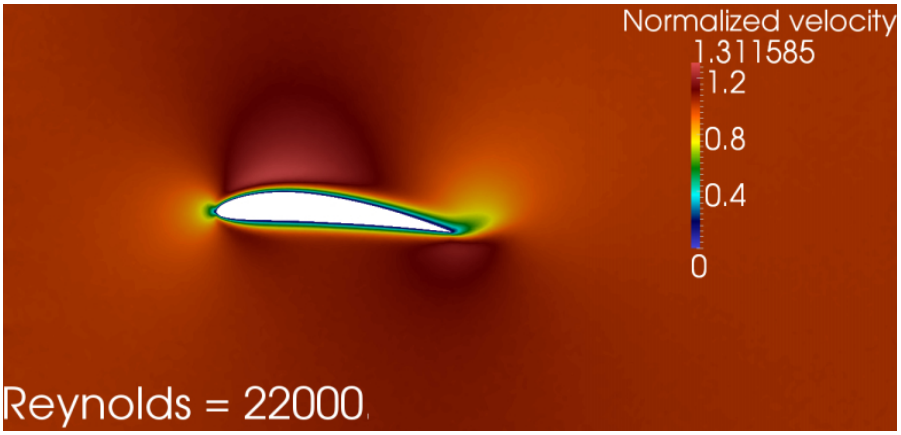


Figure 1.1: *Boundary layers are very sensitive to the precise geometry of the hydrodynamic configuration. In this example, the solution is affected by a poor mesh and appears to be unable to capture the boundary layer, resulting in an early detachment of the fluid flow from the surface. (Courtesy of Christian Kanesan.)*

to pass from CAD to FEA are now estimated to take over 80% of the overall analysis time. This issue is becoming even more problematic with the increasing complexity of engineering designs[1]. In addition to this tedious translation process, the finite element meshes generated out of it only are an approximation of the actual geometry that we view as exact. Unfortunately, this can lead in some applications (see fig. 1.1) to errors in analytical results. To address those issues, Hughes introduced in 2005 [6] a new approach referred to as **Isogeometric Analysis**. Since then, a number of additional papers have appeared. His idea was to rethink the entire process by focusing on a unique geometric model for both design and analysis, while maintaining compatibility with existing practices.

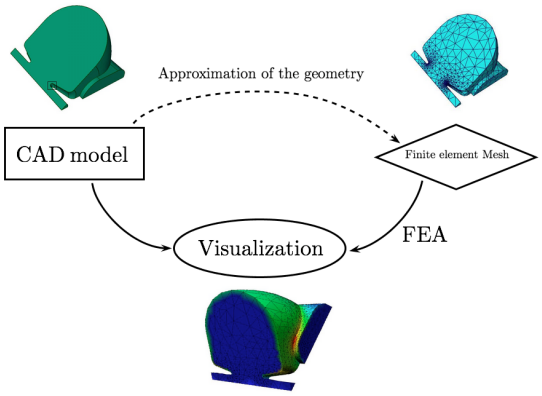


Figure 1.2: *Interactions between CAD model and finite element mesh during a classical finite element analysis.*

Among potential candidate computational geometry technologies that can be used to produce this geometric design, **NURBS (Non-Uniform Ratioanl B-Splines)** have been chosen. Those are the industry standard, and consequently widely used in engineering design. Thus there exist very efficient and numerically stable algorithms to generate NURBS objects. They are convenient for any free-form surface modeling and can exactly represent any conics sections. Finally, they possess very useful mathematical properties, that we will present in the following developments.

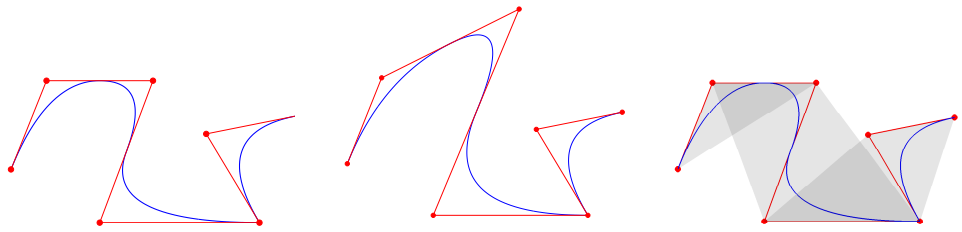
[1] for example, a typical automobile consists of about 3000 parts.

2 Key concepts of NURBS based representation

In classic FEA, we only have one notion of mesh and one notion of an element. In NURBS, things complicates a little as we have two notions of meshes: the **control mesh** and the **physical mesh**.

2.1 Control mesh

The **control mesh** is defined by **control points** $\mathbf{B} \in \mathbb{R}^n$. Depending on the dimension of the NURBS object, the control mesh can also be referred as: **control polygon** (NURBS curve), **control net** (NURBS surface) and **control lattice** (NURBS solid).
The control mesh is composed in general of n-linear elements [2], and thus looks like typical finite element meshes. It *does not* conform to the actual geometry of the considered object and behaves more like a *scaffold* controlling the geometry (fig. 1.3 gives some insight on this informal assertion).



(a) *Control mesh and control points (in red) with the corresponding NURBS curve (in blue).*

(b) *By moving a control point we understand better how the control mesh controls the geometry of the curve.*

(c) *NURBS obey the strong convex hull property: it is completely contained within the convex hull defined by its control points.*

Figure 1.3: ***Influence of the control mesh on the associated NURBS curve.***

2.2 Physical Mesh

The **physical mesh** is a *decomposition* of the actual geometry into *elements*. In NURBS there is two notions of elements: **patches** and **knot spans**. Both possess two representations, in the **physical space** Ω and in the **parent space** (or **parametric space**) $[0,1]^d$, with d the dimension of the NURBS object. Those two representations are linked through the NURBS mapping: each element in the physical space is the image of a corresponding element in the parameter space (see fig. 1.4).

Patches

The **patches** may be thought as *macro-elements*, helping to decompose complex designs in simplest subdomains, within which the material or physical properties are supposed to be uniform. In the parent space, they are lines, rectangles or cuboids depending one

[2] most often, bilinear quadrilateral for $n = 2$ and trilinear hexahedra for $n = 3$. However, control elements may be degenerated to more primitive shapes, such as triangles or tetrahedra.

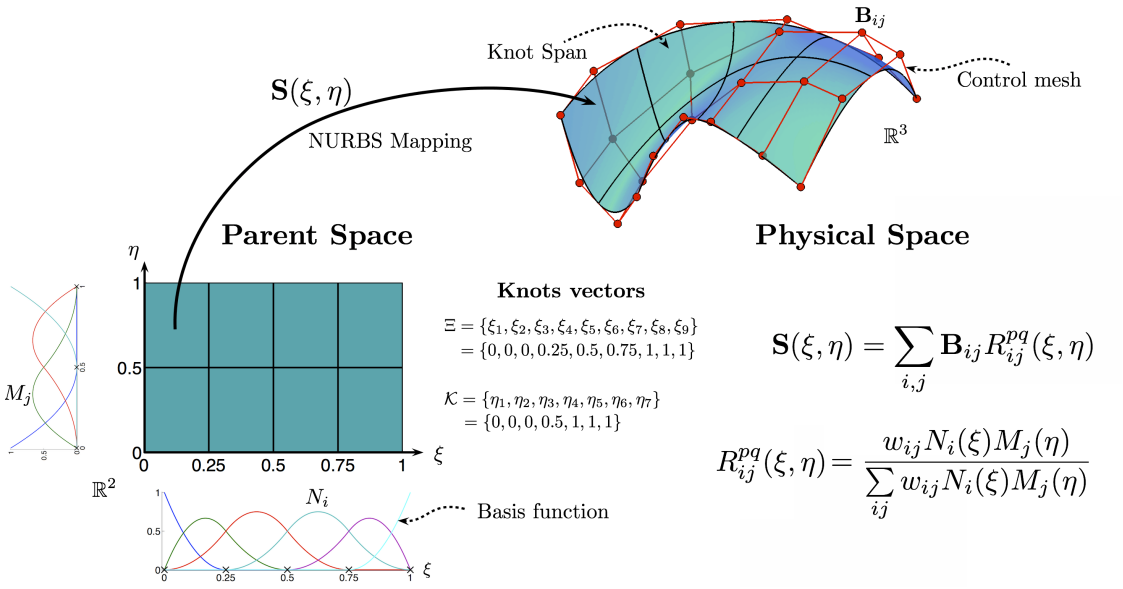


Figure 1.4: ***Schematic illustration of the different concepts involved in the NURBS construction of a one-patch surface.*** *The physical mesh (in blue) has two representations, in the physical space and in the parent space. Elements from the parent space are mapped to a corresponding element in the physical space through the NURBS mapping* $\mathbf{S}(\xi,\eta)$*. We also represented the basis functions* N_i *and* M_j *along the two parametric directions.*

the dimension of the NURBS object. Thus, complex topologies must be represented with more than one patch. Still, most of geometries from academic test cases can be designed from a single patch.

Knot spans

Each patch can be decomposed into **knot spans**, the smallest notion of an element. Knot spans are bounded by **knots**, which partition the parent space into elements.

Definition 2.1 (*Knot, Knot vector and Knot span*)

We consider a NURBS curve. Associated with it is a partition of the parametric space $[0,1]$*.This partition is stored into the* ***knot vector*** $\Xi = \{\xi_1,\xi_2,\ldots,\xi_{n+p+1}\}$*, with* $0 = \xi_1 \leq \xi_2 \leq \cdots \leq \xi_{n+p+1} = 1$*,* $\xi_i \in [0,1]$ *the* ***i-th knot****,* n *the number of* ***basis functions*** *and* p *the* ***degree*** *of the polynomials composing the basis function.*
Moreover, we call ***knot spans*** *the segments* $[\xi_i,\xi_{i+1}[$*, for* $i = 1,\ldots,n+p$*.*

Remark : The partition provided by the knot vector Ξ is said to be **uniform** if the knots are equally spaced, and **non-uniform** if not.

For NURBS surfaces, we define one knot vector per parametric direction, and then develop each of the knots along the other parametric direction. Then the knots are all the segments of the type $[\xi_i,\xi_{i+1}[\times\{\eta_j\}$ or $\{\xi_i\}\times[\eta_j,\eta_{j+1}[$, and knot spans are the rectangles

of the type $[\xi_i, \xi_{i+1}[\times[\eta_j, \eta_{j+1}[$ (see fig. 1.4 for an example). For higher dimensional NURBS objects, the construction is similar.
Together with the notion of knot, comes the notion of **multiplicity** m of a knot. In fact, knots values may be repeated, that is, more than one knot may take on the same value. Then the multiplicity m of a knot is defined as the number of time the value of this given knot has been repeated:

Definition 2.2 (*Multiplicity of a Knot*)

*Let $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$ be the knot vector of a NURBS object. Then, the **multiplicity** m_i of a knot $\xi_i \in [0, 1]$ is given by:*

$$m_i = \#\{\xi_j \in \Xi : \xi_j = \xi_i\}.$$

Remark : Sometimes we need to be able to discriminates among knots having multiplicities greater than one. To this end, we can represent the knots in the **index space** which uniquely identifies each knot with its index (see fig. 1.5).

As we will see, the multiplicities of knots values have important implications for the properties of the basis functions.
Finally, a standard in the CAD literature is to consider only **open knot vectors**:

Definition 2.3 (*Open Knot Vector*)

*A knot vector is said to be **open** if its first and last knot values appear $p + 1$ times, with p the degree of the basis functions.*

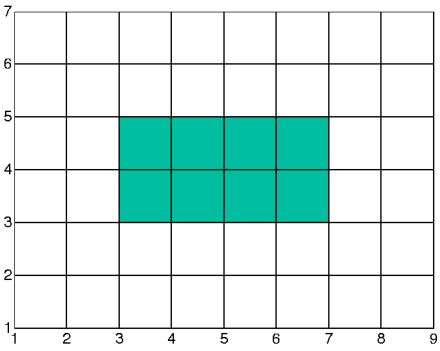


Figure 1.5: ***Index space of the example from fig. 1.4.*** *The colored part corresponds to the support of the parameteric space.*

We will see that basis functions associated with an open knot vector $\{\xi_1, \ldots, \xi_{n+p+1}\}$ are interpolatory at the end of the parameter space interval $[\xi_1, \xi_{n+p+1}]$. Another implication on considering open knot vectors is that the boundary of a NURBS object with d parametric dimensions is itself a NURBS object with $d - 1$ parametric dimension.

2.3 Basis functions

A NURBS object is properly defined once provided a knot vector, a set of control points and the associated **basis functions**. As NURBS are built from **B-splines**, we will present here as a starting point the simpler construction of the B-splines basis functions, that we will later extend into the NURBS setting. For a B-spline object, basis functions are defined recursively through the **Cox-de-Boor recursion formula**:

Definition 2.4 (*Basis functions*)

*Let $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+q+1}\}$ be a knot vector. Then, the **basis functions** $N_{i,p}$, with $i = 1, \ldots, n+q$ and p the degree of the basis functions, are defined by:*

- *for $p = 0$:*

$$N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

- *for $p = 1, 2, 3, \ldots, q$:*

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi). \tag{1.2}$$

Remark : In the case of repeated knots value , we will adopt the usual convention '$0/0 = 0$'.

The results of applying (1.1) and (1.2) to a uniform knot vector are presented in fig. 1.6. We can observe in fig. 1.6 that the basis functions are each identical but shifted relative to each other. This "homogeneous" pattern continues as increases the degree p, but is no more observable for non-uniform knots with repeated knots values (see fig. 1.7).
We also notice three important features of B-splines basis functions: the basis functions are all positive, their support increases as p increases, and each function of degree p has $p-1$ continuous derivatives across the knots. All of those important features of B-splines are summarized in the subsequent proposition :

Proposition 2.1

Let $\Xi = \{\xi_1, \ldots, \xi_{n+p+1}\}$ a knot vector and $\{N_{i,p}\}_{1 \leq i \leq n}$ basis functions of degree p as in definition 2.4. Then, we have:

(P1) ***Support:*** *The support of a basis function $N_{i,p}$ of degree p is $[\xi_i, \xi_{i+p+1}[$, or $p+1$ knot spans. Moreover, any given basis function shares is support with (including itself) at most $2p+1$ other basis functions.*

(P2) ***Nonnegativity:*** $N_{i,p}(\xi) \geq 0, \ \forall \xi \in [0,1], \ \forall i = 1, \ldots, n.$

(P3) ***Smoothness:*** *Any basis function of degree p has $p - m_i$ continuous derivatives at a knot ξ_i with multiplicity $m_i \leq p$. When $m_i = p$, then the basis functions are interpolatory at that knot.*

Remarks :
- B-splines curves constructed from B-splines basis functions will inherit the property (P1) through the property of **locality**: moving a single control point can affect the geometry of no more than $p+1$ elements of the curve.
- When $m_i = p+1$ (first and last knot for open knot vectors), we extend the property (P3) by defining a function $\mathcal{C}^{-1}$ as a discontinuous function (see fig. 1.8).

Proof : These three properties can be shown by induction on p, exploiting the recursive construction of the basis described in formula (1.2).

(P1)

- Base case ($p = 0$): The assertion is obvious for $p = 0$, as the functions $N_{i,0}$ simply are indicator function of the segments $[\xi_i, \xi_{i+1}[$. Each of these segments correspond to one knot span. Moreover, as they all are disjoint, any function $N_{i,0}$ shares indeed its support with $2 \times 0 + 1 = 1$ other functions including itself.

- Induction step: Suppose (P1) is true for basis functions $\{N_{i,p-1}\}_{1 \leq i \leq n}$ of degree $p - 1, p \geq 1$. Then, it must be true for basis functions of degree p. From the (1.6) we have:

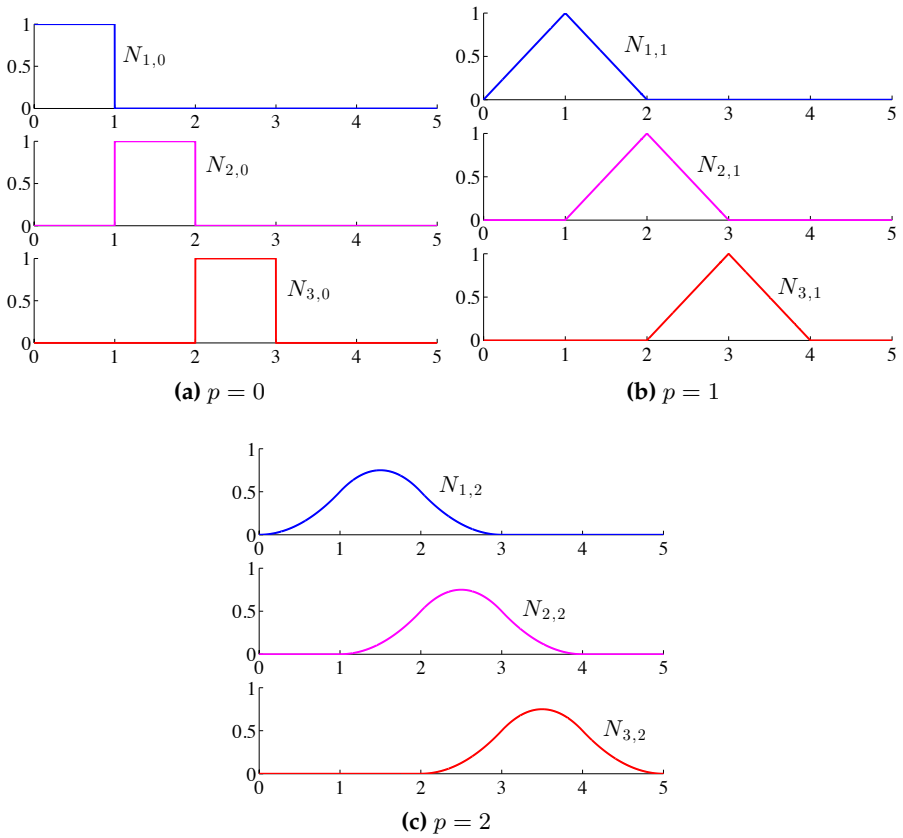
$$N_{i,p}(\xi) = \overbrace{\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i}}^{\mathcal{S}_1=[0,1]} \overbrace{N_{i,p-1}(\xi)}^{\mathcal{S}_2=[\xi_i,\xi_{i+p}[\ by\ (P1)} + \overbrace{\frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}}}^{\mathcal{S}_3=[0,1]} \overbrace{N_{i+1,p-1}(\xi)}^{\mathcal{S}_4=[\xi_{i+1},\xi_{i+p+1}[\ by\ (P1)} .$$

  Then, the total support $\mathcal{S}_{i,p}$ of the function $N_{i,p}$ is $\mathcal{S}_{i,p} = (\mathcal{S}_1 \cap \mathcal{S}_2) \cup (\mathcal{S}_3 \cap \mathcal{S}_4)) = [\xi_i, \xi_{i+p+1}[$, or $p + 1$ knot spans, for $i = 1, \ldots, n$. From the expression of the support, one can easily convinces itself that any given basis function shares is support with (including itself) at most $2p + 1$ other basis functions.

(P2)

- Base case (p=0): Obvious.

- Induction step: Suppose (P2) is true for basis functions $\{N_{i,p-1}\}_{1 \leq i \leq n}$ of degree $p - 1$.



(a) $p = 0$ **(b)** $p = 1$

(c) $p = 2$

Figure 1.6: *Basis functions of degree 0,1 and 2 for a uniform knot vector* $\Xi = \{0, 1, 2, 3, 4, \ldots\}$.

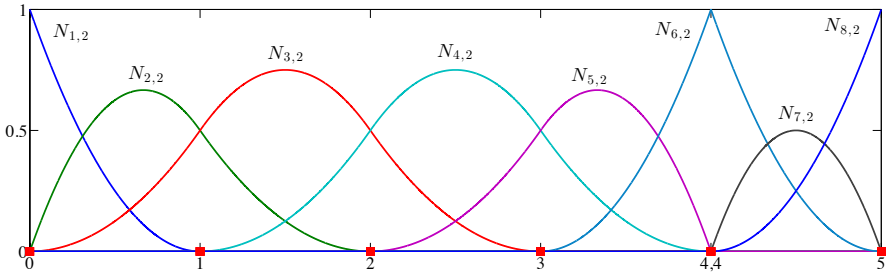


Figure 1.7: *Quadratic basis functions for open, non-uniform knot vector* $\Xi = \{0, 0, 0, 1, 2, 3, 4, 4, 5, 5, 5\}$.

Then, it must be true for basis functions of degree p. Again from (1.6), we have:

$$N_{i,p}(\xi) = \underbrace{\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi)}_{f_1(\xi)} + \underbrace{\frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi)}_{f_2(\xi)}.$$

By exploiting the induction hypothesis and the property (P1) proved previously, we can analyze the signs of the two terms $f_1(\xi)$ and $f_2(\xi)$ on $[0, 1]$:

ξ	0		ξ_i		ξ_{i+p}		1
$N_{i,p-1}(\xi)$		0	0	+	0	0	
$\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i}$		−	0	+		+	
$f_1(\xi)$		0	0	+	0	0	

and,

ξ	0		ξ_{i+1}		ξ_{i+p+1}		1
$N_{i+1,p-1}(\xi)$		0	0	+	0	0	
$\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}}$		+		+	0	−	
$f_2(\xi)$		0	0	+	0	0	

Thus, $N_{i,p}(\xi) \geq 0, \forall \xi \in [0, 1]$ as a sum of two nonnegative functions on [0,1].

(P3) The proof of (P3) is postponed as it requires the knowledge of additional results. ■

Before going back to the proof of the property (P3), we need to present the following result:

Proposition 2.2 (*Partition of unity*)

Let $\Xi = \{\xi_1, \ldots, \xi_{n+p+1}\}$ *an open knot vector and* $\{N_{i,p}\}_{1 \leq i \leq n}$ *basis functions of degree* p *as in definition 2.4. Then, the B-splines basis functions form a* ***partition of unity****, namely:*

$$\sum_{i=1}^{n} N_{i,p}(\xi) = 1, \quad \forall \xi \in [0, 1].$$

Proof : By induction on p:

- Base case (p=0): The case $p = 0$ is trivially verified, as the functions are simply indicator function of the segments $[\xi_i, \xi_{i+1}]$ which form a partition of the segment $[0, 1]$.
- Induction step: Suppose the property holds for $p - 1$. Then, from (1.2) we have:

$$\begin{aligned}
\sum_{i=1}^{n} N_{i,p}(\xi) =& \sum_{i=1}^{n} \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi), \\
=& \frac{\xi - \xi_1}{\xi_{1+p} - \xi_1} N_{1,p-1}(\xi) + \sum_{i=2}^{n} \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \cdots \\
& \cdots + \sum_{i=2}^{n-1} \frac{\xi_{i+p} - \xi}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi - \xi_{n+p+1}}{\xi_{n+p+1} - \xi_{n+1}} N_{n+1,p-1}(\xi), \\
=& \frac{\xi - \xi_1}{\xi_{1+p} - \xi_1} \underbrace{N_{1,p-1}(\xi)}_{=0,\ as\ \Xi\ open} + \frac{\xi - \xi_{n+p+1}}{\xi_{n+p+1} - \xi_{n+1}} \underbrace{N_{n+1,p-1}(\xi)}_{=0,\ as\ \Xi\ open} + \sum_{i=2}^{n-1} N_{i,p-1}(\xi), \\
=& \sum_{i=1}^{n} N_{i,p-1}(\xi) = 1,
\end{aligned}$$

  with one more time the convention '$0/0 = 0$'. ■

With this result in hand, we are ready to show (P3). Once again, we invoke induction. However, this time exploiting the formula (1.2) for the inductive step seems hopeless, so we introduce the subsequent lemma, which provides an analog recursive formula for the derivatives of a B-spline function.

Lemma 2.1 (*Derivatives of B-spline functions*)

For a given polynomial degree p and a knot vector Ξ, the derivative of the i^{th} basis function is given by:

$$\frac{d}{d\xi} N_{i,p}(\xi) = \frac{p}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) - \frac{p}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi), \quad \forall \xi \in [0, 1]. \tag{1.3}$$

Proof : This result can be shown by induction. ■

Finally, the next lemma provides us the base step of the induction proof:

Lemma 2.2

Suppose that no knot among $\xi_i, \xi_{i+1}, \ldots, \xi_{i+p}, \xi_{i+p+1}$ occurs more than $p \geq 1$ times. Then the B-spline function $N_{i,p}$ is continuous for all real number $\xi \in [0, 1]$.

Proof : The proof is by induction on the degree p.

- We start from $p = 1$. In this case, it's easy to see with an explicit representation that a linear B-spline is continuous for three distinct knots (see fig. 1.6).
- Induction step: We now assume that the lemma holds for B-splines of degree $p - 1$. To show that it is also true for p, we must split the in three different cases:

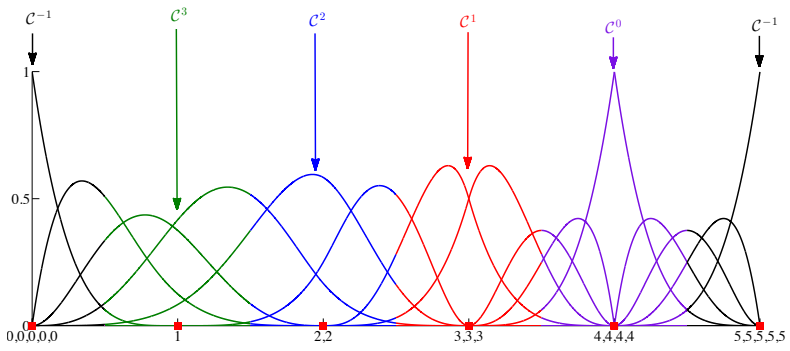


Figure 1.8: *Quartic ($p = 4$) basis functions for an open, non-uniform knot vector $\Xi = \{0, 0, 0, 0, 0, 1, 2, 2, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 5\}$. The smoothness across an interior knot is a direct result of the polynomial order p and the multiplicity of the knot m_i. When $m_i = p$, basis functions are interpolatory at that knot.*

1. $\xi_i = \xi_{i+1} = \ldots = \xi_{i+p-1}$:
   From (1.2) we have:

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi).$$

   Because of the repeated knots, the support of $N_{i,p-1}$ is $[\xi_i, \xi_{i+p}[= [\xi_{i+p-1}, \xi_{i+p}[$. Moreover, when $\xi = \xi_{i+p-1}$ the term $\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i}$ vanishes as $\xi_i = \xi_{i+p-1}$ and thus, for $\xi \in [0, \xi_{i+p-1}] \cup [\xi_{i+p}, 1]$ we have $N_{i,p}(\xi) = \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi)$. From the induction hypothesis, $N_{i+1,p-1}$ is continuous on $[0, 1]$ as no knot among $\xi_{i+1}, \ldots, \xi_{i+p+1}$ occurs more than $p - 1$ times. Then $N_{i,p}$ is continuous on $[0, \xi_{i+p-1}] \cup [\xi_{i+p}, 1]$, as a product of two continuous functions. Finally, for $\xi \in]\xi_{i+p-1}, \xi_{i+p}[$, $N_{i,p-1}$ and $N_{i+1,p-1}$ are simply polynomials of degree $p - 1$ and thus are smooth, which yields $N_{i,p}$ continuous on $[0, 1]$.
2. $\xi_{i+2} = \xi_{i+3} = \ldots = \xi_{i+p+1}$:
   The reasoning is analog to the previous case, except that this time the induction hypothesis applies for $N_{i,p-1}$.
3. $\xi_{i+1} = \xi_{i+3} = \ldots = \xi_{i+p}$:
   The third case is handled a little differently. In fact, in this case the induction hypothesis does not apply neither for $N_{i,p-1}$ nor $N_{i+1,p-1}$, so we need to show continuity directly. For $\xi = \xi_i$, $N_{i,p}(\xi_i) = 0$ because $\frac{\xi - \xi_i}{\xi_{i+p} - \xi_i}$ vanishes and ξ_i does not belong to the support of $N_{i+1,p-1}$. Therefore, $N_{i,p}$ is continuous at ξ_i. For $\xi = \xi_{i+p+1}$ we have once again $N_{i,p}(\xi_{i+p+1}) = 0$ and so continuity at this knot also. For $\xi = \xi_{i+1} = \ldots = \xi_{i+p}$ one needs to notice that the function $N_{i,p}$ is the only B-spline basis function that can go through this knot, as the support of B-spline basis functions is always $p + 1$ knot spans. Then, all other basis functions with ξ_{i+1} in its support will either be in the case 1 or 2, and thus be null at this knot. Therefore, the partition of unity property requires $N_{i,p}$ to be equal to one at this knot. Moreover, the function cannot be discontinuous at this point or the partition of unity property would be violated in the neighborhood of the knot. In fact all the other functions sharing this knot in their support are continuous and null at this knot, so they are continuously decreasing to 0 as ξ tends to ξ_{i+1}. Then,

to maintain the partition of unity property, $N_{i,p}$ must also increase continuously to one as ξ tends to ξ_{i+1}. Finally, $N_{i,p}$ is trivially continuous on $[0,1] - \{\xi_i, \xi_{i+1}, \xi_{i+p+1}\}$ as a sum of polynomials. ■

We are now ready to prove the property (P3):

Proof of (P3) : We fix a knot $\xi_i \in \Xi$ with multiplicity m_i. First we remark that (P3) is equivalent to the following assertion: *any basis function of degree $r + m_i$ has r continuous derivatives at ξ_i*. Then, we proceed by induction on r:

- Base case (r=0): For the base case the assertion simply states that when $mi = p$ any basis function is continuous at the knot ξ_i. This is true in vertu of lemma 2.2.
- Induction step: Suppose that the assertion is true for $r - 1$. Then, from lemma 2.1 we have:

$$\frac{d}{d\xi} N_{i,r+m_i}(\xi) = \frac{r+m_i}{\xi_{i+r+m_i} - \xi_i} N_{i,r-1+m_i}(\xi) - \frac{r+m_i}{\xi_{i+r+m_i+1} - \xi_{i+1}} N_{i+1,r-1+m_i}(\xi),$$

which has $r - 1$ continuous derivatives at ξ_i from the induction hypothesis. Therefore, $N_{i,r+m_i}$ has r continuous derivatives at ξ_i. ■

3 B-spline geometries

3.1 B-spline curves

B-splines curves in $\mathbb{R}^d$ are simply linear combinations of the B-spline basis functions. The vector-valued coefficients of this linear combination are the **control points** of the curve.

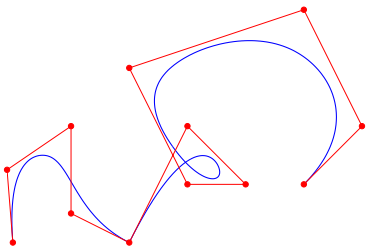
Definition 3.1 (*B-spline curve*)

*Given n basis functions $\{N_{i,p}\}_{i=1,\dots,n}$ and the associated control points $\{\mathbf{B}_i\}_{i=1,\dots,n}$, $\mathbf{B}_i \in \mathbb{R}^d$, a piecewise-polynomial **B-spline curve** is given by:*

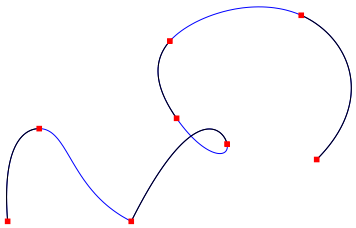
$$\mathbf{C}(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi)\mathbf{B}_i, \quad \forall \xi \in [0,1].$$

Remark : The function $\mathbf{C}(\cdot)$ is mapping elements from the parameter space to corresponding elements in the physical mesh.

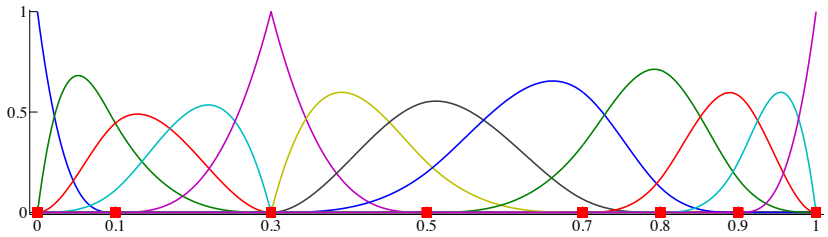
An example of a quartic ($p = 4$) B-spline curve can be found on fig. 1.9. This curve is built from quartic basis functions, with an open, non-uniform knot vector Ξ. The curve is interpolatory at the first and last control points, which is characteristic of a curve built from an open knot vector. The curve is also interpolatory at the fifth control point, where the multiplicity of the knot is equal to the degree of the basis functions. Except from these three knots, the curve is $\mathcal{C}^{p-1} = \mathcal{C}^3$ everywhere else. It is interesting to observe the difference between the control points and the images of the knots.



(a) *Curve (in blue) and control polygon (in red).*

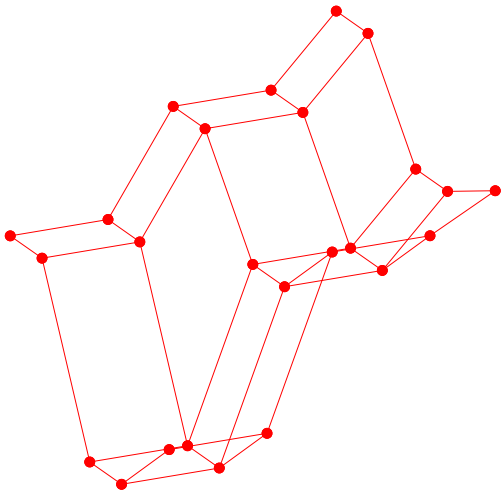


(b) *Curve and physical mesh (images of knots in red, mapping of knot spans in blue and black).*

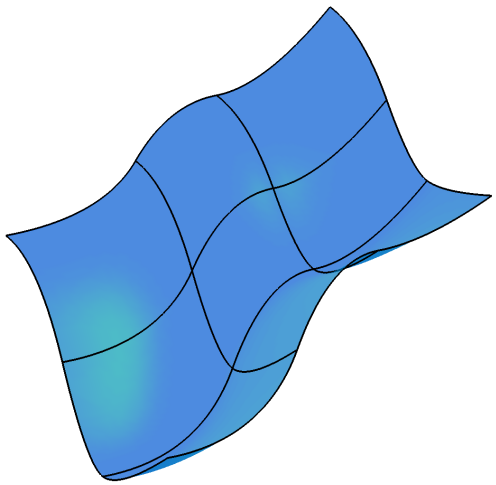


(c) *Basis functions*

Figure 1.9: *Example of a B-spline curve and its associated control polygon, physical mesh and quartic basis functions, for the open, non-uniform knot vector:* $\Xi = \{0, 0, 0, 0, 0.1, 0.3, 0.3, 0.3, 0.5, 0.7, 0.8, 0.9, 1, 1, 1, 1\}$



(a) *Control net (linear interpolation of control points)*



(b) *Physical mesh (mapping of knot spans in the physical space).*

Figure 1.10: *Example of a B-spline surface and its associated control net, and physical mesh, for the open, non-uniform knot vectors:* $\Xi = \{0, 0, 0, 1/3, 2/3, 1, 1, 1\}$ *and* $\mathcal{H} = \{0, 0, 0, 1/3, 2/3, 1, 1, 1\}$.

3.2 B-spline surfaces

To design a B-spline surface in $\mathbb{R}^d$, we first define two open knots vectors Ξ and $\mathcal{H}$: one for each parametric direction (see fig. 1.4). From each of these knot vectors, we build the associated univariate B-spline basis functions $\{N_{i,p}\}_{i=1,\dots,n}$, and $\{M_{j,q}\}_{j=1,\dots,m}$, with degree respectively p and q. Then, given a control net $\{\mathbf{B}_{i,j}\}_{i=1,\dots n}^{j=1,\dots m}$, with $\mathbf{B}_{ij} \in \mathbb{R}^d$, a B-spline surface is defined by a tensor product:

Definition 3.2 (*B-spline surface*)

*Given univariate basis functions $\{N_{i,p}\}_{i=1,\dots,n}$ and $\{M_{j,q}\}_{j=1,\dots,m}$ and a control net $\{\mathbf{B}_{i,j}\}_{i=1,\dots,n}^{j=1,\dots,m}$, with $\mathbf{B}_{i,j} \in \mathbb{R}^d$, a tensor-product **B-spline surface** is given by:*

$$\mathbf{S}(\xi,\eta) = \sum_{i=1}^{n}\sum_{j=1}^{m} N_{i,p}(\xi) M_{j,q}(\eta)\mathbf{B}_{ij}, \quad \forall(\xi,\eta) \in [0,1]^2.$$

Remark : From this definition, we can see that the edges (ξ or η fixed) of a B-spline surface will be a B-spline curve. This is an interesting result for design.

Figure 1.10 shows an example of a B-spline surface and its control net. Figure 1.11 shows the biquadratic multivariate basis functions of this NURBS surface. As an attempt to generalize the different properties of univariate B-spline basis functions previously derived, one can define **bivariate basis functions**: $\tilde{N}_{i,j;p,q}(\xi,\eta) := N_{i,p}(\xi)M_{j,q}(\eta)$. Then, these basis functions, and in consequence the B-spline surface built out of it, inherit many properties from the univariate case:

Proposition 3.1

*Let Ξ and $\mathcal{H}$ be two open knot vectors, with univariate basis functions respectively $\{N_{i,p}\}_{i=1,\dots,n}$ and $\{M_{j,q}\}_{j=1,\dots,m}$. Then, the **bivariate basis functions**: $\tilde{N}_{i,j;p,q} = N_{i,p}M_{j,q}$ have the following properties:*

(P1) Support: *The support of a bivariate basis function $\tilde{N}_{i,j;p,q}$ of degrees p and q is $[\xi_i, \xi_{i+p+1}[\times[\eta_j, \eta_{j+q+1}[$, or $(p+1)(q+1)$ knot spans.*

(P2) Nonnegativity: $\tilde{N}_{i,j;p,q}(\xi,\eta) \geq 0, \quad \forall(\xi,\eta) \in [0,1], \quad \forall i = 1,\dots,n, \quad \forall j = 1,\dots,m.$

(P3) Smoothness: *The number of continuous partial derivatives in a given parametric direction may be determined from the associated one-dimensional knot vector and polynomial order (see proposition 2.1,(P3)).*

(P4) Partition of unity: *The basis functions form a partition of unity:*

$$\sum_{i=1}^{n}\sum_{j=1}^{m} \tilde{N}_{i,j;p,q}(\xi,\eta) = \left(\sum_{i=1}^{n} N_{i,p}(\xi)\right)\left(\sum_{j=1}^{m} M_{j,q}(\xi)\right) = 1, \quad \forall(\xi,\eta) \in [0,1]^2.$$

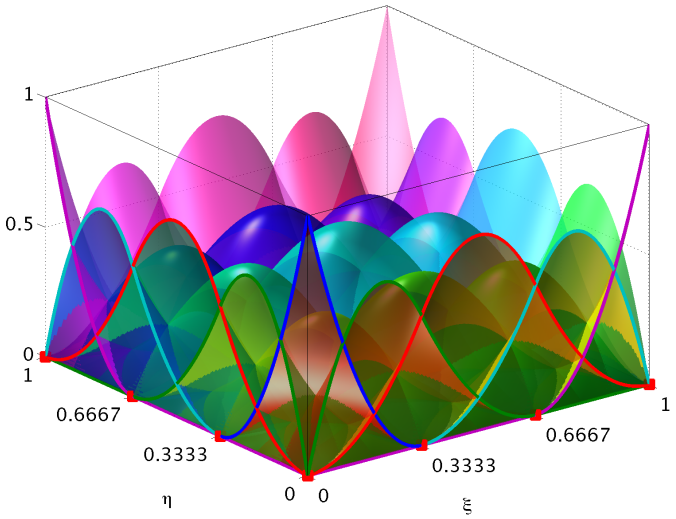
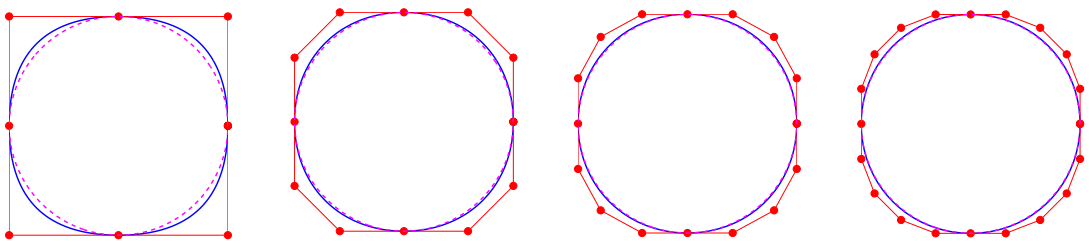


Figure 1.11: *Biquadratic ($p = 2, q = 2$) multivariate basis functions $\tilde{N}_{i,j;2,2}(\xi, \eta)$ for the B-spline surface from fig. 1.10*

4 Non-Uniform Rational B-Splines

Despite their flexibility and their mathematically convenient properties, B-splines appear unable to represent a wide array of objects, many of which being used in engineering design. This intrinsic limitation comes from the incapability of polynomials to exactly represent some curves, and in particular conic sections (such as circles, ellipses...). One could satisfy itself with a reasonable approximation of conics section with B-splines, but unfortunately this comes at a price ! On fig. 1.12 we see that a degree $p = 5$ B-spline curve is necessary to obtain a reasonable approximation of a circle.
To address this issue, we need to generalize B-splines into **Non-Uniform Rational B-splines (NURBS)** (for a good reference on NURBS, see [2]).
A NURBS entity in $\mathbb{R}^d$ is obtained by the **projective transformation** of a B-spline entity in $\mathbb{R}^{d+1}$. In fact, conic sections, such as circles or ellipses, can be *exactly* constructed by projective transformations of piecewise quadratic curves (see fig. 1.13, where a circle in



(a) $p = 2$ **(b)** $p = 3$ **(c)** $p = 4$ **(d)** $p = 5$

Figure 1.12: ***Approximation of a circle by a B-spline curve.*** *As the degree p of the basis functions increase, the approximation gets better and better. For $p = 5$, the approximation looks reasonable, but still the curve is not an exact circle.*

$\mathbb{R}^2$ is constructed from a piecewise quadratic Bspline curve in $\mathbb{R}^3$). The transformation we consider is projecting every point in the curve onto the $z = 1$ plane by a ray through the origin:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \begin{pmatrix} x/z \\ y/z \\ 1 \end{pmatrix}. \tag{1.4}$$

This transformation can bring finite points to infinity and points at infinity to finite range. In this context, we need to update notations, in order to avoid any confusion between the NURBS curve in $\mathbb{R}^d$ and the B-spline curve in $\mathbb{R}^{d+1}$. The B-spline $\mathbf{C}^w(\xi) \in \mathbb{R}^{d+1}$ is called the **projective curve**, and the associated control points $\mathbf{B}_i^w \in \mathbb{R}^{d+1}$ are called the **projective control points**. The terms **curve** and **control points** are reserved to the NURBS object $\mathbf{C}(\xi)$ and $\mathbf{B}_i$ respectively (see fig. 1.13).
We get the control points $\mathbf{B}_i$ of the NURBS curve by applying the projective transformation (1.4) to the projective control points of the projective B-spline curve (see fig. 1.13):

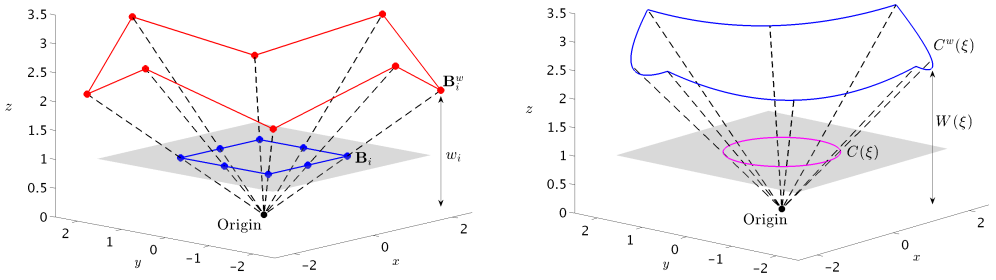
Definition 4.1 (*Control points of a NURBS curve*)

Given a projective B-spline curve $\mathbf{C}^w(\xi) \in \mathbb{R}^{d+1}$ and its associated projective control points $\mathbf{B}_i^w \in \mathbb{R}^{d+1}$, we define the control points $\mathbf{B}_i \in \mathbb{R}^d$ of the NURBS curve $\mathbf{C}(\xi) \in \mathbb{R}^d$ by:

$$\begin{aligned} (B_i)_j &= (\mathbf{B}_i^w)_j / w_i, \quad j = 1, \dots d, \\ w_i &= (\mathbf{B}_i^w)_{d+1}, \end{aligned}$$

*where $(\mathbf{B}_i)_j$ refers to the j^{th} component of the vector $\mathbf{B}_i$ and w_i is called the i^{th} **weight**, positive in most of applications.*

To obtain the NURBS curve, we repeat the operation and apply the projective transfor-



(a) *Control points (in blue) and projective control points (in red).*

(b) *Curve (in magenta) and projective curve (in blue).*

Figure 1.13: *A circle in $\mathbb{R}^2$, obtained by applying the projective transformation* (1.4) *to a piecewise quadratic B-spline in $\mathbb{R}^3$. *(a) Projective transformation of the projective control points $\mathbf{B}_i^w$ yields the control points of the NURBS object $\mathbf{B}_i$. (b) Projective transformation of the B-spline curve $\mathbf{C}^w(\xi)$ yields the NURBS curve $\mathbf{C}(\xi)$.*

mation to each point of the projective B-spline curve. To do so, we need to introduce the **weighting function**:

Definition 4.2 (*Weighting Function*)

*Given a projective B-spline curve $\mathbf{C}^w(\xi) \in \mathbb{R}^{d+1}$ with its associated B-spline basis functions $N_{i,p}(\xi)$, we define the **weighting function** by:*

$$W(\xi) = \sum_{i=1}^{n} N_{i,p}(\xi) w_i, \quad \forall \xi \in [0,1],$$

with w_i as in definition 4.1.

Remarks:
- In $\mathbb{R}^3$, the weighting function may be interpreted as the height $z(\xi) = W(\xi)$ of the B-spline projective curve for each $\xi \in [0,1]$ (see fig. 1.13).
- Figure 1.14 (c) shows the effect of weights on the weighting function.

Then, the NURBS curve $\mathbf{C}(\xi)$ may be defined as:

$$(\mathbf{C}(\xi))_j = \frac{(\mathbf{C}^w(\xi))_j}{W(\xi)}, \quad \forall \xi \in [0,1].$$

However, this definition is not convenient, and we would like to manipulate NURBS curves in the exact same fashion as B-spline curves: given a knot vector and control points, we would like to define basis functions so that the NURBS curve is the linear combination of these basis functions and the control points. To this end, we propose the following definition for NURBS basis functions:

Definition 4.3 (*NURBS basis functions*)

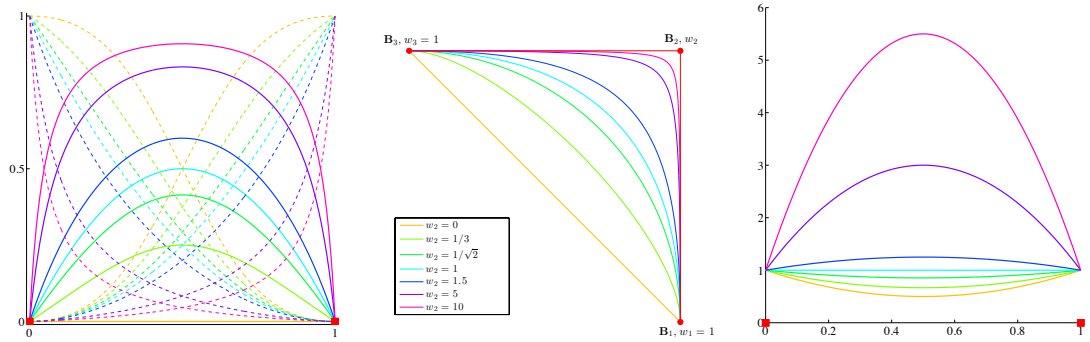
*Given a knot vector Ξ, the B-spline basis functions $\{N_{i,p}\}_{i=1,\dots,n}$ built out of it, and weights $\{w_i\}_{i=1,\dots,n}$, we define the **NURBS basis functions** by:*

$$R_i^p(\xi) = \frac{N_{i,p}(\xi) w_i}{W(\xi)} = \frac{N_{i,p}(\xi) w_i}{\sum_{j=1}^{n} N_{j,p}(\xi) w_j}, \quad \forall \xi \in [0,1].$$

Remarks:
- One can show that the NURBS basis functions R_i^p still verify all the properties of B-spline basis functions listed in proposition 2.1 and proposition 2.2 (support, nonnegativity, smoothness, partition of unity).
- Derivatives of the basis function can be computed by applying the quotient rule:

$$\frac{d}{d\xi} R_i^p(\xi) = w_i \frac{W(\xi) N'_{i,p}(\xi) - W'(\xi) N_{i,p}(\xi)}{W(\xi)^2}.$$

- The basis functions R_i^p are clearly piecewise rational functions, which explains the appellation of NURBS: Non-uniform *Rational* B-splines.



(a) *NURBS basis functions.* **(b)** *NURBS curve* **(c)** *Weighting function*

Figure 1.14: *Evolution of basis functions, NURBS curve and weighting function for different values of* $w_2 = \{0, 1/3, 1/\sqrt{2}, 1, 1.5, 5, 10\}$*, weight asocialted to* $\mathbf{B}_2$*, and an open knot vector* $\Xi = \{0, 0, 0, 1, 1, 1\}$*. The weights* w_1 *and* w_3 *for* $\mathbf{B}_1$ *and* $\mathbf{B}_2$ *are fixed to 1. We can see that the bigger is the weight, the more the curve is "attracted" by the control point* $\mathbf{B}_2$*. Moreover, we can check graphically on (a) that the partition of unity property is verified for all weights* w_2*.*

- Figure 1.14 (a) shows the effect of weights on the basis functions.

Then, a NURBS curve is naturally defined by:

Definition 4.4 (*NURBS curve*)

Given n *basis functions* $\{R_i^p\}_{i=1\dots,n}$ *and the associated control points* $\{\mathbf{B}_i\}_{i=1,\dots,n} \in \mathbb{R}^d$*, a* ***NURBS curve*** *is given by:*

$$\mathbf{C}(\xi) = \sum_{i=1}^{n} R_i^p(\xi)\mathbf{B}_i, \quad \forall \xi \in [0, 1]. \tag{1.5}$$

Remarks :

- the form of (1.5) is identical to that of B-splines.
- in this formulation, the control points $\mathbf{B}_i$ can be chosen independently from their associated weights, which loose any geometric interpretability.
- we see clearly that if all the weights are equal, then $R_i^p(\xi) = N_i^p(\xi)$ and thus B-splines are a special case of NURBS.
- Figure 1.14 (b) shows the effect of weights on the NURBS curve.

Finally, NURBS surfaces are defined analogously in terms of the rational basis functions:

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

5 Geometrical Properties of NURBS

NURBS and B-spline objects inherit many useful geometrical properties from the basis functions used to build them. These properties will help in the later design process of the computational domain. For the sake of simplicity, we will present and demonstrate these properties for a NURBS curve $\mathbf{C}(\xi) \in \mathbb{R}^d$. However, these properties are also true for NURBS surfaces or more general NURBS objects.

Proposition 5.1 (*Affine covariance*)

*Let $\mathbf{C}(\xi) \in \mathbb{R}^d$ be a NURBS curve, with control points $\{\mathbf{B}_i\}_{i=1,\dots n}$ and basis functions $\{R_i^p(\xi)\}_{i=1,\dots n}$. Let $\Phi : \mathbb{R}^d \to \mathbb{R}^d; \mathbf{x} \mapsto A\mathbf{x} + \mathbf{b}$ be an **affine transformation**, with $A \in \mathbb{R}^{d\times d}$ a matrix, and $\mathbf{b} \in \mathbb{R}^d$ a vector. Finally, let $\tilde{\mathbf{C}}(\xi)$ be the NURBS curve with control points $\{\Phi(\mathbf{B}_i)\}_{i=1,\dots n}$ and basis functions $\{R_i^p(\xi)\}_{i=1,\dots n}$. Then, we have:*

$$\Phi(\mathbf{C}(\xi)) = \tilde{\mathbf{C}}(\xi), \quad \forall \xi \in [0,1].$$

Remark : The above result can be restated as follows: *applying an affine transformation to a NURBS curve is the same as applying the affine transformation to the control points of this NURBS curve.*

Proof : As Φ is an affine transformation we have trivially:

$$\Phi(\mathbf{C}(\xi)) = \Phi(\sum_{i=1}^{n} R_i^p(\xi)\mathbf{B}_i) = \sum_{i=1}^{n} R_i^p(\xi)\Phi(\mathbf{B}_i) = \tilde{\mathbf{C}}(\xi), \quad \forall \xi \in [0,1].$$

■

The next property is very interesting for the shape optimization procedure, as it tells us how exactly is a NURBS curve affected by the displacement of a single control point.

Proposition 5.2 (*Property of Locality*)

Let $\mathbf{C}(\xi) \in \mathbb{R}^d$ be a NURBS curve, with control points $\{\mathbf{B}_i\}_{i=1,\dots n}$ and basis functions $\{R_i^p(\xi)\}_{i=1,\dots n}$. Then, moving a single control point $\mathbf{B}_j$ can affect the geometry of no more than $p+1$ elements of the curve, with p the degree of the basis functions.

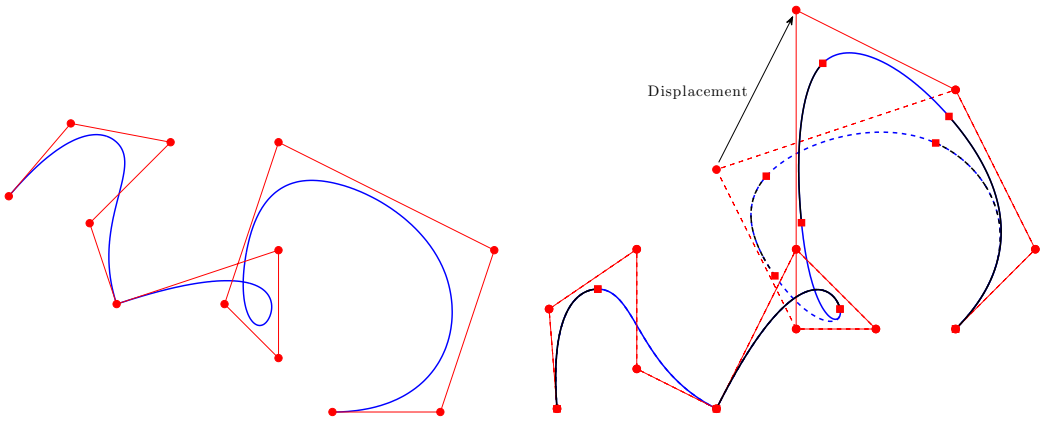
Proof : This is a direct consequence of the compact support of the basis functions (see proposition 2.1 (P1)). ■

Finally, we conclude this section with a very convenient property: the **strong convex hull** property. This property states that a NURBS curve of degree p is entirely contained in a certain convex hull, the p**-convex hull**, defined as follows:

Definition 5.1 (*p-Convex Hull*)

*Let $\{\mathbf{B}_i\}_{i=1,\dots n}$ be a set of points in $\mathbb{R}^d$, an $p \in \mathbb{R}$. Then, the p**-convex hull** is defined by :*

$$\bigcup_{j=1}^{n-p-1} \left\{ \sum_{i=0}^{p+1} \alpha_i \mathbf{B}_{j+i} \,|\alpha_i \geq 0, \forall i = 0,\dots,p+1; \sum_{i=0}^{p+1} \alpha_i = 1 \right\}.$$



(a) *Rotating the curve is the same as rotating the control points*

(b) *Moving a single control point can affect no more than $p + 1 = 4$ elements.*

Figure 1.15: *Illustration of the affine covariance property and the property of locality with the example of the NURBS curve from fig. 1.9*

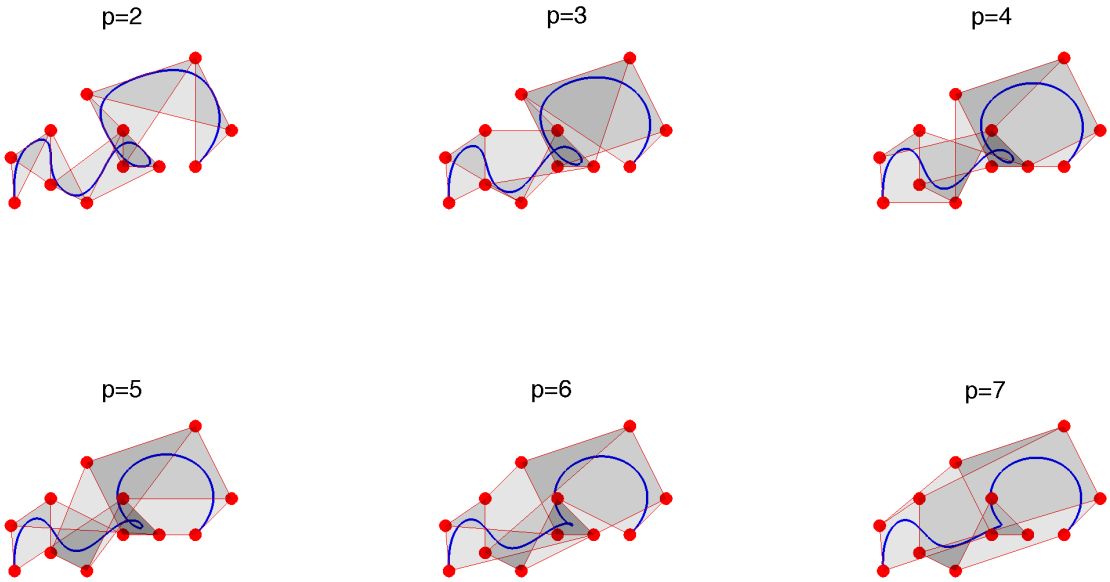


Figure 1.16: ***Strong convex hull property.*** *A NURBS curve of degree p is entirely contained within the convex hull defined by its control points. This convex hull is defined as the union of all the convex hulls formed by $p + 1$ consecutive control points. Note that as the degree p increases, the curve becomes smoother and the effect of each control point diminishes. This behavior is known as the* ***variation diminishing property****.*

Remark : The p-convex hull is the union of all the convex hulls formed by $p+1$ consecutive points.

With this definition in hand, we can state the strong convex hull property see fig. 1.16 for an example):

Proposition 5.3 (*Strong Convex Hull Property*)

Let $\mathbf{C}(\xi) \in \mathbb{R}^d$ be a NURBS curve, with control points $\{\mathbf{B}_i\}_{i=1,\dots n}$ and basis functions $\{R_i^p(\xi)\}_{i=1,\dots n}$.
Then, the curve $\mathbf{C}(\xi)$ is entirely contained in the p-convex hull formed by the control points, with p the degree of the basis functions.

Proof : Let fix $\xi \in [0,1]$. Then, $\mathbf{C}(\xi) = \sum_{i=1}^{n} R_i^p(\xi)\mathbf{B}_i$. Let denote ξ_k the knot such that $\xi \in [\xi_k, \xi_{k+1}[$ (such a knot exists as the knot vector Ξ forms a partition of the segment $[0,1]$. Then, from the support of the basis functions, we know that some basis functions will be null at ξ. More precisely, we have:

$$\mathbf{C}(\xi) = \sum_{i=k-p-1}^{k} R_i^p(\xi)\mathbf{B}_i = \sum_{i=0}^{p+1} R_{j+i}^p(\xi)\mathbf{B}_{j+i},$$

with $j = k-p-1$. Finally, from the nonnegativity and the partition of unity property of NURBS basis functions we have that:

$$\mathbf{C}(\xi) = \sum_{i=0}^{p+1} R_{j+i}^p(\xi)\mathbf{B}_{j+i} \in \left\{ \sum_{i=0}^{p+1} \alpha_i \mathbf{B}_{j+i} \,|\alpha_i \geq 0, \forall i = 0,\dots,p+1;\ \sum_{i=0}^{p+1} \alpha_i = 1 \right\},$$

and thus $\mathbf{C}(\xi)$ belongs to the p-convex hull for every ξ. ∎

6 Geometric representation of the computational domain

6.1 Basic geometry

We present here the basic geometry that we will use as an initial configuration for our shape optimization procedure. We wish to design both the hydrofoil and the computational domain around it by means of a NURBS surface. The design we will propose is constituted of a single patch, as the geometry is relatively simple.
First, we need to choose a parametrization for the boundaries of the physical domain. Let Ξ and $\mathcal{H}$ be the two open knots for the two parametric direction. Then, the boundaries of the parameter space are matched to the ones of the physical space as follows (see fig. 1.17):

- **Boundary 1:** $[\xi_1, \xi_{n+p+1}] \times \{\eta_1\}$. This boundary of the parameter space will be mapped to the surface of the hydrofoil into the physical space.
- **Boundary 2:** $\{\xi_{n+p+1}\} \times [\eta_1, \eta_{m+q+1}]$. This boundary of the parameter space will be glued in the physical space with the image of the boundary 4 to close the domain.
- **Boundary 3:** $[\xi_1, \xi_{n+p+1}] \times \{\eta_{m+q+1}\}$. This boundary of the parameter space will be mapped to the external boundaries of the physical domain.

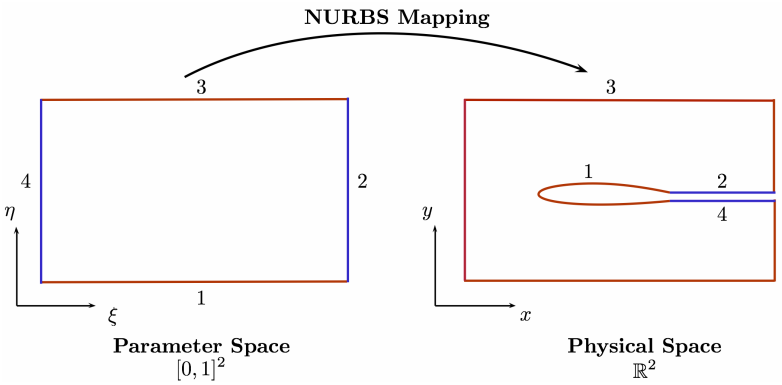


Figure 1.17: *Correspondence between boundaries of the parameter space and the physical space.*

- **Boundary 4:** $\{\xi_1\} \times [\eta_1, \eta_{m+q+1}]$. This boundary of the parameter space will be glued in the physical space with the image of the boundary 2 to close the domain.

Given this parametrization, we choose bi-quadratic ($p = 2, q = 2$) univariate basis functions for each parametric direction ξ and η. In general, it is recommended to work with low degree basis functions, as it helps in the design process. In fact, as we can observe in fig. 1.16, the lower is the degree p of the basis functions, the tighter is the p-convex hull around the NURBS curve. Therefore, for low degrees, the control polygon really behaves like a "skeleton" of the curve, which makes the choice of control points for a given curve easier. Moreover, for the special case of bi-quadratic basis functions ($p = 2$), the link between the control polygon and the curve is even stronger as we can show that the curve is tangent to each segment of the polygon.

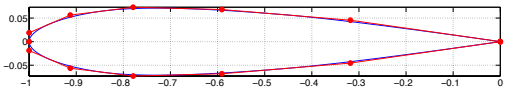
As we have seen previously, edges of a NURBS surface are NURBS curves. Therefore, we can design the hydrofoil independently from the rest of the surface. The design we propose here (see fig. 1.18) is inspired from the one of a symmetric NACA0012 foil. The open knot vector Ξ we consider is:

$$\Xi = \{0, 0, 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1, 1\}.$$

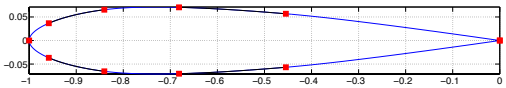
The knots 0 and 1 are superposed and correspond to the rear of the hydrofoil. The knot 0.5 corresponds to the front of the hydrofoil (see fig. 1.18). This knot has been repeated because of the impossibility to exactly represent with NURBS arcs greater than $90°$. Therefore, the front of the hydrofoil is composed of two conic sections glued together. Despite this repeated knot, the curve is still globally $\mathcal{C}^1$, since three wisely chosen control points have been aligned (see fig. 1.18). Notice also that the length of the chord [3] of the hydrofoil is one, which is standard in foil design.

We are now ready to design the entire domain. The open knots vectors for the two

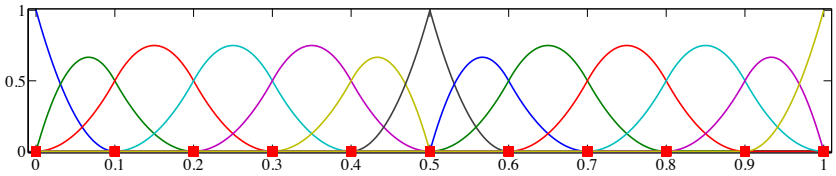
[3] line linking the front to the rear



(a) *NURBS curve with its control polygon*

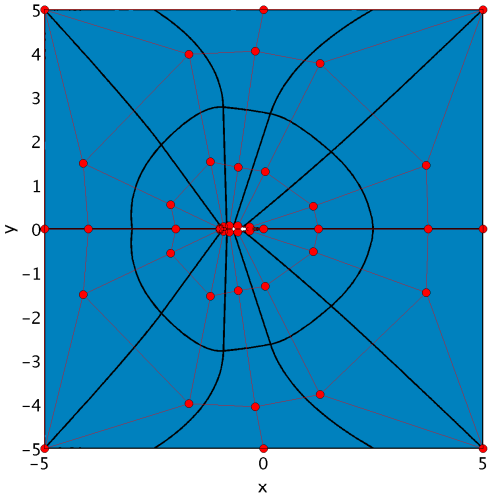


(b) *Elements of the NURBS curve*

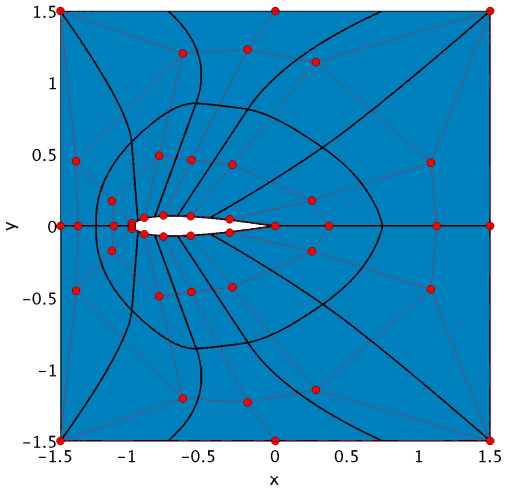


(c) *Basis functions*

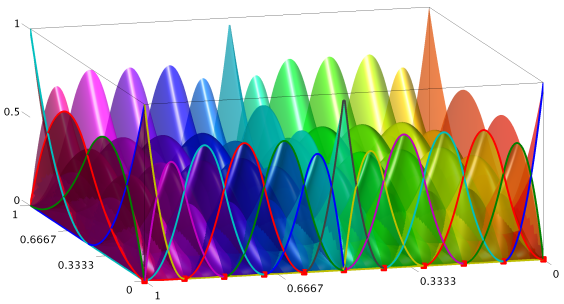
Figure 1.18: *NURBS curve design of the hydrofoil.*



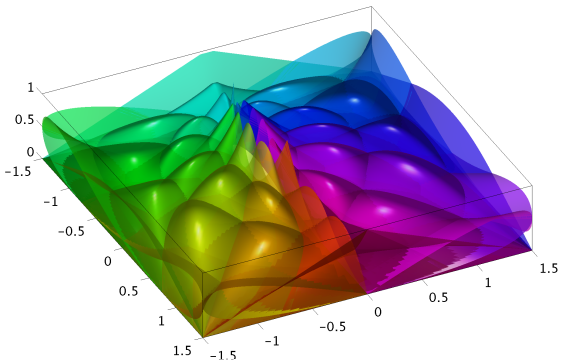
(a) *NURBS surface design of the domain used for computation.*



(b) *NURBS surface design of a reduced domain for visualization purpose.*



(c) *Multivariate basis functions of the NURBS surface*



(d) *Multivariate basis functions plotted in a reduced physical domain*

Figure 1.19: *NURBS surface design of the domain.*

parametric directions ξ and η are respectively:

$$\Xi = \{0, 0, 0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.5, 0.6, 0.7, 0.8, 0.9, 1, 1, 1\},$$
$$\mathcal{H} = \{0, 0, 0, 0.5, 1, 1, 1\}.$$

The squared shape of the domain has been obtained by superposing some control points in the corners (see fig. 1.19). To avoid boundary effects for the later simulations, the domain has been designed wide enough.

6.2 Enrichment of the NURBS space: (mesh) h-refinement

We now need to refine the physical mesh for the computation. This process, that can be really time consuming in classic FEA[4], is almost straightforward in NURBS setting, as the mesh is already representing exactly the actual geometry of the problem. There exist different ways in which NURBS basis functions may be enriched while leaving the underlying geometry and its parametrization intact. The three main mechanisms are :

- **Knot Insertion:** Inserts knots without changing the NURBS object geometrically or parametrically.
- **Order Elevation:** Raises the polynomial order of the basis functions used to represent the geometry, without changing the NURBS object geometrically or parametrically. During order elevation, the multiplicity of each knot value is increased by one, but no new knots are added.
- **k-refinement (higher order *and* higher continuity):** Elevates the order of the original basis functions to q and then inserts a unique knot value $\bar{\xi}$. The basis will have $q - 1$ continuous derivatives at $\bar{\xi}$.

For our purposes, we only need to investigate the first mechanism of knot insertion. Therefore, we do not present in details the two other mechanisms (an interested reader can find a complete description in [1]).

Definition 6.1 (*Knot Insertion*)

*Given a knot vector $\Xi = \{\xi_1, \ldots, \xi_{n+p+1}\}$, we introduce an **extended knot vector**, $\bar{\Xi} = \{\bar{\xi}_1 = \xi_1, \bar{\xi}_2, \ldots, \bar{\xi}_{n+m+p+1} = \xi_{n+p+1}\}$, such that $\Xi \subset \bar{\Xi}$. The new $n + m$ basis functions are built from this new knot vector $\bar{\Xi}$. The new $n + m$ control points , $\bar{\mathcal{B}} = (\bar{\mathbf{B}}_1, \ldots, \bar{\mathbf{B}}_{n+m})^T$, are formed from linear combinations of the original control points, $\mathcal{B} = (\mathbf{B}_1, \ldots, \mathbf{B}_n)^T$, by*

$$\bar{\mathcal{B}} = \mathbf{T}^p \mathcal{B},$$

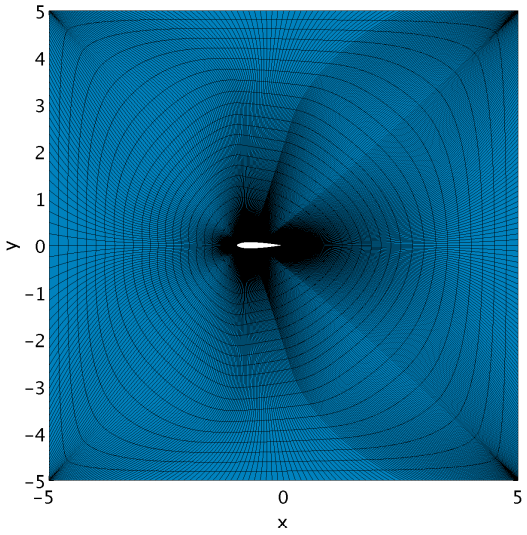
where, for $q = 0, 1, \ldots, p-1$:

$$T_{ij}^0 = \begin{cases} 1 & \bar{\xi}_i \in [\xi_j, \xi_{j+1}[, \\ 0 & \textit{otherwise,} \end{cases} \quad \textit{and} \quad T_{ij}^{q+1} = \frac{\bar{\xi}_{i+q} - \xi_j}{\xi_{j+q} - \xi_j} T_{ij}^q + \frac{\xi_{j+q+1} - \bar{\xi}_{i+q}}{\xi_{j+q+1} - \xi_{j+1}} T_{ij+1}^q.$$

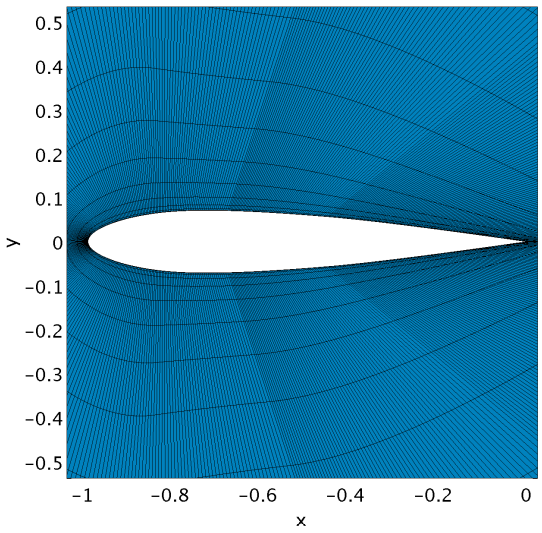
[4] in fact, really often, we need to investigate adaptive meshes which are enriched accordingly to the geometry of the problem.

Remark : One can show that with this procedure, neither the geometry nor the parametrization of the NURBS object is changed. Thus, this procedure respects the desired properties of knot insertion.

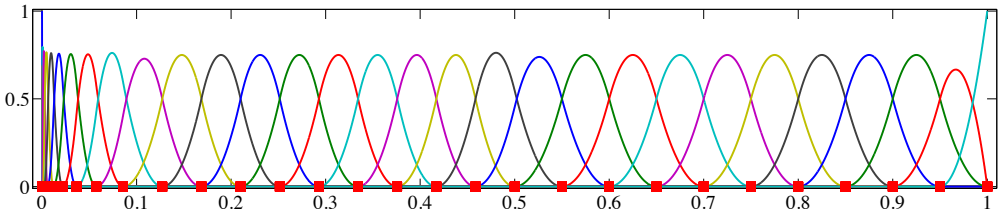
In our case, to save some computational time, we investigate a non-homogeneous enrichment of the physical mesh by knot insertion: we increase the enrichment around the hydrofoil, which is the region of interest, and at the tailing edge, where the elements are very distorted, due to the sharp geometry. During this process, we take care not to repeat existing knots, to avoid a reduction of continuity of the basis. The resulting refined physical mesh of the domain is depicted on fig. 1.20.



(a) *Refined mesh of the domain.*



(b) *Zoom on the refined mesh around the hydrofoil.*



(c) *Refined univariate basis functions in the direction η.*

Figure 1.20: *Refinement of the computational domain by knot insertion. The parametrization of the curve is unchanged through the process.*

CHAPTER

2 Irrotational flow around the hydrofoil

We consider a steady, irrotational 2D flow around the hydrofoil. To compute the velocity of the flow, we solve numerically the Laplace problem for the stream function. Finally, we compute the lift coefficient, as will be the goal of the shape optimization process.

1 Laplace problem

1.1 The stream function

We consider a steady and irrotational two-dimensional flow with velocity $\mathbf{v} = (u, v)$. We assume that we can associate to this flow a **stream function** defined by:

Definition 1.1 (*Stream function*)

*Let $\mathbf{v} = (u, v) \in \mathbb{R}^2$ be a velocity field defined in a domain Ω. Then , a **stream function** for $\mathbf{v}$ is a scalar function $\psi(x, y) \in \mathcal{C}^2(\Omega)$ such that:*

$$\begin{cases} u = \frac{\partial \psi}{\partial y}, & \text{in } \Omega, \\ v = -\frac{\partial \psi}{\partial x}, & \text{in } \Omega. \end{cases} \tag{2.1}$$

In vectorial notation, (2.1) *can be re-written as:*

$$\mathbf{v} = \nabla \times \mathbf{\Psi}, \tag{2.2}$$

*with $\nabla\times$ the **curl** operator, $\mathbf{\Psi} = (0, 0, \psi)^T$ and $\mathbf{v}$ is identified with the three-dimensional vector $(u, v, 0)^T$.*

Remarks :
- The solution ψ of (2.1) or (2.2) is defined up to a constant. Therefore, there is not a unique stream function for a given velocity field.
- In all that follow, we will identify ψ with its vectorial representation $\mathbf{\Psi}$ and the designation stream function will be used equivalently to designate one or the other depending on the context.

Flows that can be described by a stream function have useful mathematical properties, that simplify the study of many fluid problems. In particular, the continuity equation is always verified for such a flow:

Proposition 1.1 (*Continuity Equation*)

Let $\mathbf{v}$ be a velocity flow and ψ the associated stream function such that (2.1) *holds. Then the **continuity equation** is verified:*

$$\nabla \cdot (\rho \mathbf{v}) = 0 \quad \text{in } \Omega,$$

*with $\nabla\cdot$ the **divergence** operator, and ρ the **density** of the fluid, assumed to be constant.*

Remark : As ρ is constant, then the continuity equation becomes: $\nabla \cdot \mathbf{v} = 0$ in Ω. Then the flow is said to be **incompressible**.

Proof : Exploiting the definition of ψ we have:

$$\nabla \cdot (\rho \mathbf{v}) = \rho \left(\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} \right) = \rho \left(\frac{\partial^2 \psi}{\partial y \partial x} - \frac{\partial^2 \psi}{\partial x \partial y} \right) = 0 \quad \text{in } \Omega.$$

In fact, $\psi \in \mathcal{C}^2(\Omega)$ so from the Schwarz's theorem we have $\frac{\partial^2 \psi}{\partial y \partial x} - \frac{\partial^2 \psi}{\partial x \partial y} = 0$. ■

► Interpretation of the stream function

Volume flow interpretation:
We consider a flow ψ with velocity $\mathbf{v}$, and two points A and B (see fig. 2.1 such that $\vec{AB} = (dx, dy)$, sufficiently close to each other so that the velocity is constant in this region. Then, the flow per unit thickness $d\psi$ passing through the section AB can be computed by:

$$d\psi = \mathbf{v} \cdot \mathbf{n} dl,$$

with $\mathbf{n}$ the vector orthogonal to $\vec{AB}$ and $dl = \|\vec{AB}\|$.
This can be re-written:

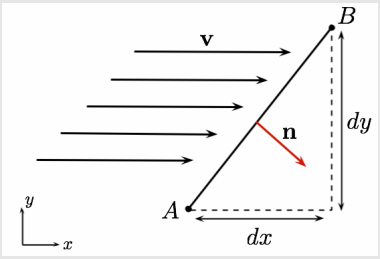


Figure 2.1: *Volume flow through a segment AB.*

$$d\psi = \begin{pmatrix} u \\ v \end{pmatrix} \cdot \begin{pmatrix} dy \\ -dx \end{pmatrix} = -v\, dx + u\, dy = \frac{\partial \psi}{\partial x}\, dx + \frac{\partial \psi}{\partial y}\, dy,$$

which yields: $u = \frac{\partial \psi}{\partial y}$, $v = -\frac{\partial \psi}{\partial x}$.

Streamline interpretation:
We consider a line along which ψ is some constant ψ_1. Then, along such line we have:

$$d\psi = \frac{\partial \psi}{\partial x}\, dx + \frac{\partial \psi}{\partial y}\, dy = -v\, dx + u\, dy = 0,$$

which yields:

$$\frac{dy}{dx} = \frac{v}{u}.$$

We recognize here the equation for a **streamline** (curves that are instantaneously tangent to the velocity vector of the flow). Thus, equipotentials for ψ are streamlines of the flow.

1.2 Strong formulation

We now derivate a differential equation verified by ψ. From the null vorticity assumption we have:

$$\nabla \times \mathbf{v} = \nabla \times (\nabla \times \mathbf{\Psi}) = \mathbf{0} \quad \text{in } \Omega. \tag{2.3}$$

From a property of the vectorial Laplacian operator the following equality holds:

$$\Delta \mathbf{\Psi} = -\underbrace{\nabla \times (\nabla \times \mathbf{\Psi})}_{=0 \text{ (2.3)}} + \nabla(\nabla \cdot \mathbf{\Psi}). \tag{2.4}$$

We have $\mathbf{\Psi} = (0, 0, \psi)$ which yields $\nabla \cdot \mathbf{\Psi} = \frac{\partial \psi}{\partial z} = 0$, as ψ only depends on x and y. Then, (2.4) simplifies in:

$$\Delta \mathbf{\Psi} = 0 \quad \text{in } \Omega,$$

which is equivalent in 2D to:

$$\Delta \psi = 0 \quad \text{in } \Omega. \tag{2.5}$$

We now need to specify the boundary conditions to ensure existence and unicity of the physical solution. The computational domain Ω and its boundaries $\Gamma_{in}, \Gamma_{out}, \Gamma_{top}, \Gamma_{bot}$, and Γ_a are depicted on fig. 2.2.
First, we wish the flow to be only in x-direction (i.e. $v = 0$) at the inlet and the outlet of the computational domain. This gives us the following Neumann conditions:

$$\frac{\partial \psi}{\partial x} = 0, \text{ on } \Gamma_N = \Gamma_{in} \cup \Gamma_{out}.$$

The top and bottom boundaries of the domain are interpreted as streamlines of the flow, since far away from the hydrofoil. This yields the Dirichlet conditions:

$$\psi = \psi_{top} \text{ on } \Gamma_{top}, \quad \psi = \psi_{bot} \text{ on } \Gamma_{bot}.$$

The values of ψ_{top} and ψ_{bot} can be set by looking at the volume flow rate $\dot{V}$ through Γ_{in}:

$$\dot{V} = \int_{\Gamma_{in}} \mathbf{v} \cdot \mathbf{e}_x \, dy = \int_{y_{bot}}^{y_{top}} u \, dy = \int_{y_{bot}}^{y_{top}} \frac{\partial \psi}{\partial y} \, dy = \psi_{top} - \psi_{bot}.$$

This volume flow rate can also be computed as follows:

$$\dot{V} = \int_{\Gamma_{in}} \mathbf{v} \cdot \mathbf{e}_x \, dy = \int_{y_{bot}}^{y_{top}} u \, dy = u_0(y_{top} - y_{bot}),$$

with $u_0 = u(y_0)$, for some $y_0 \in [y_{bot}, y_{top}]$ (mean value theorem). Therefore, we have:

$$\psi_{top} - \psi_{bot} = u_0(y_{top} - y_{bot}).$$

Since ψ is defined up to a constant, we can choose:

$$\psi_{top} = u_0 \, y_{top}, \quad \psi_{bot} = u_0 \, y_{bot}.$$

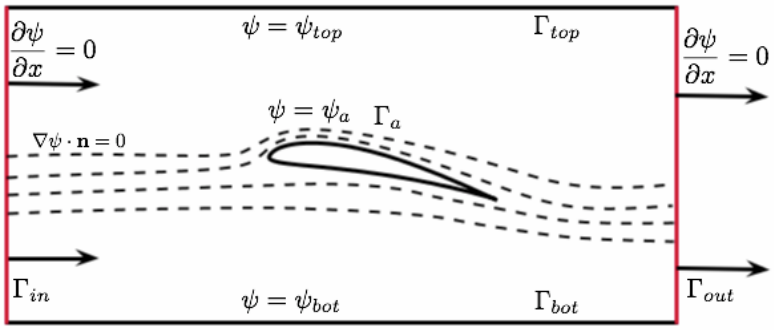


Figure 2.2: ***Schematic illustration of the boundary value problem.*** *The boundaries Γ_N with Neumann boundary conditions are depicted in red and the boundaries Γ_D with Dirichlet boundary conditions in black. Streamlines are depicted in dashed lines.*

Finally, the hydrofoil surface Γ_a will also be a streamline which gives us the last Dirichlet condition:

$$\psi = \psi_a \text{ on } \Gamma_a.$$

However, the stream function on the hydrofoil surface Γ_a is not known a priori, and thus ψ_a is also an unknown of the problem.

To find the value of ψ_a, we need to add one constraint : the so-called **Kutta condition**. The Kutta condition requires the flow to leave the trailing edge smoothly. This condition can be enforced by requiring the flow to be aligned along the bisector of the trailing edge, at the proximity of it. The geometry of the trailing edge is shown on fig. 2.3. For the flow to be aligned along $\mathbf{n}_{te}$, ψ must be locally constant along this direction (as equipotentials of ψ are streamlines of the flow). Therefore, the Kutta condition becomes:

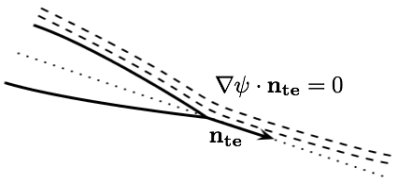


Figure 2.3: ***Geometry of the tailing edge.*** *The Kutta condition requires the flow to be aligned along the bisector of the trailing edge $\mathbf{n}_{TE}$.*

$$\nabla\psi \cdot \mathbf{n}_{te} = 0 \text{ at the trailing edge.}$$

Thus, the strong formulation of the Laplace problem verified by the stream function ψ is:

$$\text{Find } \psi : \Omega \to \mathbb{R},\ \psi_a \in \mathbb{R}, \text{ s.t.} \quad \begin{cases} -\Delta\psi = 0, & \text{in } \Omega, \\ \frac{\partial \psi}{\partial x} = 0, & \text{on } \Gamma_N = \Gamma_{in} \cup \Gamma_{out}, \\ \psi = \psi_{top} & \text{on } \Gamma_{top}, \\ \psi = \psi_{bot} & \text{on } \Gamma_{bot} \\ \psi = \psi_a & \text{on } \Gamma_a, \\ \nabla\psi \cdot \mathbf{n}_{te} = 0 & \text{at the trailing edge.} \end{cases} \tag{2.6}$$

1.3 Weak formulation

Let v be a test function in $H^1_{\Gamma_D} = \{v \in H^1(\Omega) \,:\, v = 0 \text{ on } \Gamma_D\}$, with $H^1(\Omega)$ an Hilbert space, and $\Gamma_D = \Gamma_{top} \cup \Gamma_{bot} \cup \Gamma_a$. Then, we multiply (2.5) by v and integrate both sides on the domain Ω. From the generalized integration by parts formula we obtain:

$$\begin{aligned}
-\int_\Omega v\,\Delta\psi\,d\Omega &= \int_\Omega \nabla v\cdot\nabla\psi\,d\Omega - \int_{\Gamma_N} v\nabla\psi\cdot\mathbf{n}_N\,d\Gamma_N - \underbrace{\int_{\Gamma_D} v\nabla\psi\cdot\mathbf{n}_D\,d\Gamma_D}_{=0\,as\,v\in H^1_{\Gamma_D}},\\
&= \int_\Omega \nabla v\cdot\nabla\psi\,d\Omega - \underbrace{\int_{\Gamma_{in}} v\nabla\psi\cdot\mathbf{e}_x\,d\Gamma_{in}}_{=0\,as\,\frac{\partial\psi}{\partial x}=0} - \underbrace{\int_{\Gamma_{out}} v\nabla\psi\cdot\mathbf{e}_x\,d\Gamma_{out}}_{=0\,as\,\frac{\partial\psi}{\partial x}=0},\\
&= \int_\Omega \nabla v\cdot\nabla\psi\,d\Omega.
\end{aligned}$$

Moreover, weak formulation of the Kutta condition is given by:

$$\int_\Omega w_{te}\nabla\psi\cdot\mathbf{n}_{te}\,d\Omega = 0,$$

with w_{te} a suitable weighting function (typically a relaxation of a Dirac function at the tailing edge node).
The Dirichlet conditions are non homogeneous on Γ_D, thus we introduce the **lifting function** $R_{\psi_a} \in H^1(\Omega)$ such that $R_{\psi_a} = \psi_{top}$ on Γ_{top}, $R_{\psi_a} = \psi_{bot}$ on Γ_{bot} and $R_{\psi_a} = \psi_a$ on Γ_a. Then, the function $\mathring{\psi} \in H^1_{\Gamma_D}$ such that:

$$\mathring{\psi} = \psi - R_{\psi_a},$$

is solution to the homogeneous boundary problem.
Therefore, the weak formulation of (2.6) is:

$$\begin{aligned}
&\text{Find } \mathring{\psi}\in H^1_{\Gamma_D}, \psi_a \in \mathbb{R}, \text{ s.t.:}\\
&\quad\begin{cases}
\int_\Omega \nabla v\cdot\nabla\mathring{\psi}\,d\Omega + \int_\Omega \nabla v\cdot\nabla R_{\psi_a}\,d\Omega = 0, & \forall v\in H^1_{\Gamma_D},\\
\mu\int_\Omega w_{te}\nabla\mathring{\psi}\cdot\mathbf{n}_{te}\,d\Omega + \mu\int_\Omega w_{te}\nabla R_{\psi_a}\cdot\mathbf{n}_{te}\,d\Omega = 0, & \forall\mu\in\mathbb{R}.
\end{cases}
\end{aligned} \tag{2.7}$$

For the sake of simplicity, we will write ψ instead of $\mathring{\psi}$ in all that follow.

Analysis of the weak problem

The problem (2.7) is in a non standard weak form, whose complete analysis would require a generalization of the Lax-Milgram lemma to problems in mixed form (see [3]). This is not in the scope of this report, and we will only provide here the analysis of a simplified version of (2.7), where ψ_a is assumed known. The problem we propose to analyze is then:

Given $\psi_a \in \mathbb{R}$ such that: $\mu \int_\Omega w_{te} \nabla\psi \cdot \mathbf{n}_{te}\, d\Omega = -\mu \int_\Omega w_{te} \nabla R_{\psi_a} \cdot \mathbf{n}_{te}\, d\Omega,$ holds $\quad \forall \mu \in \mathbb{R},$

$$\text{Find } \psi \in V, \text{ such that: } \quad a(\psi, v) = F_{\psi_a}(v), \forall v \in V, \tag{2.8}$$

with $V = H^1_{\Gamma_D}$, $a \in \mathcal{L}(V \times V, \mathbb{R})$ a bilinear and continuous form and $F_{\psi_a} \in \mathcal{L}(V)$ a linear and continuous functional on V. More precisely, we have:

$$a(\psi, v) = \int_\Omega \nabla v \cdot \nabla\psi \, d\Omega, \quad \forall v \in V$$
$$F_{\psi_a}(v) = -\int_\Omega \nabla v \cdot \nabla R_{\psi_a}\, d\Omega, \quad \forall v \in V.$$

Existence and unicity of the solution $\psi \in H^1_{\Gamma_D}$ to the problem 2.8 is guaranteed by the Lax-Milgram lemma which holds in this case. In fact, we have:

- **Continuity of** $a(\cdot,\cdot)$: From the Cauchy-Schwarz inequality, we have:

  $$|a(u,v)| = \left| \int_\Omega \nabla v \cdot \nabla\psi \, d\Omega \right| \leq \|\nabla v\|_{L_2} \|\nabla\psi\|_{L_2} = |v|_{H^1_{\Gamma_D}} |\psi|_{H^1_{\Gamma_D}} = \|v\|_V \|\psi\|_V, \ \forall \psi, v \in V.$$

  In fact, the seminorm $|v|_{H^1_{\Gamma_D}} = \|\nabla v\|_{L_2}$ is a norm on $V = H^1_{\Gamma_D}$.

- **Coercivity of** $a(\cdot,\cdot)$: Always from the Cauchy-Schwarz inequality we have:

  $$a(v,v) = \|v\|^2_V, \ \forall v \in V.$$

- **Continuity of** $F_{\psi_a}(\cdot)$**:** Always from the Cauchy-Schwarz inequality:

  $$|F_{\psi_a}(v)| = \left| \int_\Omega \nabla v \cdot \nabla R_{\psi_a}\, d\Omega \right| \leq \|\nabla R_{\psi_a}\|_{L_2} |v|_{H^1_{\Gamma_D}} = \eta \|v\|_V.$$

2 Numerical approximation of the solution

2.1 The Galerkin Method in NURBS setting

We propose to solve the problem (2.7) using the Galerkin method with NURBS basis functions. Let $\{V_h\}_{h \in \mathbb{R}}$ be a family of finite dimensional linear subspaces such that:

$$V_h \subset V, \quad \text{and } \dim(V_h) = N_h.$$

Then, the main difference with classic Galerkin method is in the choice of V_h. In the context of NURBS, we choose V_h such that:

$$V_h = \{v_h \in \mathcal{R}_h^{pq} \ : \ v_h = 0 \text{ on } \Gamma_D\},$$

with $\mathcal{R}_h^{pq}$ the finite dimensional linear space spanned by the bivariate NURBS basis functions $\mathcal{B} = \{\tilde{R}_{ij}^{pq}\}_{i=1,\dots,n}^{j=1,\dots,m}$ used to represent the physical mesh of the domain. This space clearly is a subspace of $V = H^1_{\Gamma_D}$ and as $\mathcal{B}$ forms a basis of $\mathcal{R}_h^{pq}$, we have that $N_h = n \times m$.
Care need to be taken in the fact that the functions $\tilde{R}_{ij}^{pq}$ are seen as taking values in the physical space Ω rather than in the parameter space (see fig. 1.19 (d) for an example). They are defined as:

$$\tilde{R}_{ij}^{pq} : \Omega \to \mathbb{R}; \quad (\tilde{\xi}, \tilde{\eta}) := \mathbf{S}(\xi, \eta) \mapsto R_{ij}^{pq}(\xi, \eta), \quad \forall i = 1, \dots, n \text{ and } \forall j = 1, \dots, m.$$

In classic finite element methods, h corresponds to the average size of an element of the grid. Here, h is the **characteristic mesh size** (typically, h is the average area of an element of the physical space). The idea of the Galerkin method is to solve the approximate problem (or Galerkin problem), restriction of the weak problem (2.7) to V_h:

$$\text{Find } \psi_h \in V_h, \psi_a \in \mathbb{R}, \text{ s.t.:} \quad \begin{cases} \int_\Omega \nabla v_h \cdot \nabla \psi_h \, d\Omega + \int_\Omega \nabla v_h \cdot \nabla R_{\psi_a} \, d\Omega = 0, & \forall v_h \in V_h, \\ \mu \int_\Omega w_{te} \nabla \psi_h \cdot \mathbf{n}_{te} \, d\Omega + \mu \int_\Omega w_{te} \nabla R_{\psi_a} \cdot \mathbf{n}_{te} \, d\Omega = 0, & \forall \mu \in \mathbb{R}. \end{cases} \tag{2.9}$$

Exploiting the vectorial structure of the spaceV_h, each element v_h of V_h can be rewritten as a linear combination of the basis functions $\tilde{R}_{ij}^{pq} \in \mathcal{B}$.
In particular, we can choose the lifting function R_{ψ_a} to be:

$$\begin{aligned} R_{\psi_a} &= \left(\sum_{i \in \Gamma_a} \tilde{R}_{i,1}^{pq}(\xi, \eta_1) \right) \psi_a + \left(\sum_{i \in \Gamma_{top}} \tilde{R}_{i,m}^{pq}(\xi, \eta_m) \right) \psi_{top} + \left(\sum_{i \in \Gamma_{bot}} \tilde{R}_{i,m}^{pq}(\xi, \eta_m) \right) \psi_{bot}, \\ &:= R_{\Gamma_a} \psi_a + R_{\Gamma_{top}} \psi_{top} + R_{\Gamma_{bot}} \psi_{top}. \end{aligned}$$

Using the partition of unity property of NURBS basis functions one can check that this function have the desired properties of a lifting function, namely: $R_{\psi_a} = \psi_{top}$ on Γ_{top}, $R_{\psi_a} = \psi_{bot}$ on Γ_{bot} and $R_{\psi_a} = \psi_a$ on Γ_a.
Then, if we store $\mathcal{B} = \{\tilde{R}_{ij}^{pq}\}_{i=1,\dots,n}^{j=1,\dots,m}$ in a vector of length $N_h = m \times n$, (2.9) can be transformed in a linear system:

$$\text{Find } \mathbf{\Psi_h} = (\psi_1, \dots, \psi_{N_h})^T \in \mathbb{R}^{N_h}, \psi_a \in \mathbb{R}, \text{ s.t.: } \begin{pmatrix} A & \mathbf{b} \\ \mathbf{c}^T & d \end{pmatrix} \begin{pmatrix} \mathbf{\Psi}_h \\ \psi_a \end{pmatrix} = \begin{pmatrix} \mathbf{f} \\ g \end{pmatrix}, \tag{2.10}$$

with:

- $A \in \mathbb{R}^{N_h \times N_h}$ a square matrix such that $A_{ij} := \int_\Omega \nabla \tilde{R}_i^{pq} \cdot \nabla \tilde{R}_j^{pq} \, d\Omega$,
- $\mathbf{b} \in \mathbb{R}^{N_h}$ a vector such that $\mathbf{b_i} := \int_\Omega \nabla \tilde{R}_i^{pq} \cdot \nabla R_{\Gamma_a} \, d\Omega$,
- $\mathbf{c} \in \mathbb{R}^{N_h}$ a vector such that $\mathbf{c}_i := \int_\Omega w_{te} \nabla \tilde{R}_i^{pq} \cdot \mathbf{n}_{te} \, d\Omega$,

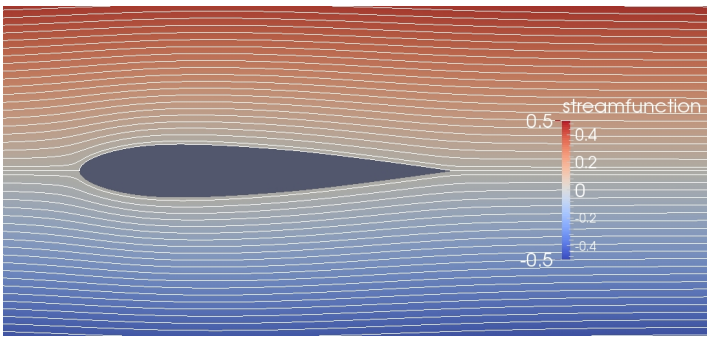
- $d \in \mathbb{R}$ a real number such that $d := \int_\Omega w_{te} \nabla R_{\Gamma_a} \cdot \mathbf{n}_{te}\, d\Omega$,
- $\mathbf{f} \in \mathbb{R}^{N_h}$ a vector such that $\mathbf{f}_i := -\int_\Omega \nabla \tilde{R}_i^{pq} \cdot (\psi_{top} \nabla R_{\Gamma_{top}} + \psi_{bot} \nabla R_{\Gamma_{bot}})\, d\Omega$,
- $g \in \mathbb{R}$ a real number such that $g = -\int_\Omega w_{te} (\psi_{top} \nabla R_{\Gamma_{top}} + \psi_{bot} \nabla R_{\Gamma_{bot}}) \cdot \mathbf{n}_{te}\, d\Omega$.

At the discrete level, we find that the problem (2.7) is well-posed, and we can solvethe linear system (2.10) to obtain $\mathbf{\Psi}_h = (\psi_1, \ldots, \psi_n)^T$ and ψ_a. Then, ψ_h is given by :

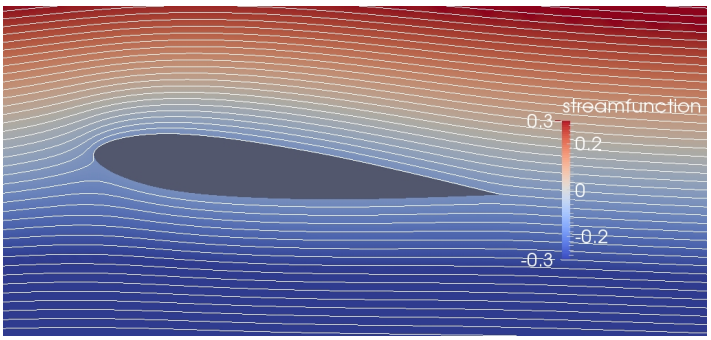
$$\psi_h = \sum_{i=1}^{N_h} \psi_i R_i^{pq}.$$

2.2 Numerical Simulation

We implemented and solved the weak problem (2.7) on Matlab using the GeoPDEs software, a suite of software tools for research on Isogeometric Analysis of PDEs. Then, we used ParaView for the visualization. Figure presents the results of simulations with different angles of attack. In both case, we observe that the hydrofoil surface is indeed



(a) *Angle of attack* $\theta = 0\ rad$



(b) *Angle of attack* $\theta = 0.1\ rad$

Figure 2.4: *Solution of the stream function and streamlines for a 2D irrotational flow. We used the GeoPDEs toolbox on Matlab for the computation, and the ParaView software for the visualization.*

a streamline and that the flow is aligned along the bisector of the trailing edge at the proximity of it : the Kutta condition is met.

2.3 Computation of the Lift

In this section, we are interested in the computation of the lift , and more precisely the lift coefficient.
In fact, we will use the lift coefficient to build the objective function of our shape optimization procedure. The **lift** is the component perpendicular to the incoming flow direction of the hydrodynamic force exerted by the flow on the hydrofoil. The component of this force in the tangential direction is called the **drag**. In the case of non-viscous flow, there is no drag, and therefore the modulus of the hydrodynamic force is equivalent to the lift.

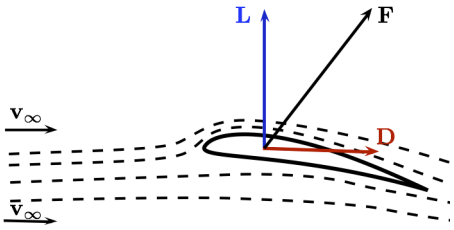


Figure 2.5: *Hydrodynamic force* $\mathbf{F}$ *on the hydrofoil surface. The lift is the vertical component while the drag is the horizontal one.*

Using the Bernoulli's law we can derive the lift which is given by :

$$L = \frac{1}{2}\rho\|\mathbf{v}_\infty\|^2 l \oint_{\Gamma_a} \left(1 - \frac{\|\mathbf{v}\|^2}{\|\mathbf{v}_\infty\|^2}\right) d\Gamma_a,$$

with ρ the density of the fluid, $\|\mathbf{v}_\infty\|^2$ the modulus of the velocity at Γ_{in} and l the characteristic length of the foil. For our purposes, it is more convenient to work with a dimensionless version of the lift, the **lift coefficient**, defined by :

$$C_L = \frac{L}{\frac{1}{2}\rho\|\mathbf{v}_\infty\|^2 l} = \oint_{\Gamma_a} \left(1 - \frac{\|\mathbf{v}\|^2}{\|\mathbf{v}_\infty\|^2}\right) d\Gamma_a = \int_0^1 \left(1 - \frac{\|\mathbf{v}(\mathbf{C}(\xi))\|^2}{\|\mathbf{v}_\infty\|^2}\right) \|\mathbf{C}'(\xi)\|\, d\xi, \quad (2.11)$$

with $\mathbf{C}(\xi)$ the NURBS parametrization of the surface[1] Γ of the hydrofoil . Finally, we also introduce another quantity, the **pressure coefficient**, defined by :

$$C_p := 1 - \frac{\|\mathbf{v}\|^2}{\|\mathbf{v}_\infty\|^2}, \quad \text{in } \Omega,$$

so that :

$$C_L = \oint_{\Gamma_a} C_p\, d\Gamma_a.$$

To compute C_L, we approximate (2.11) using the following mid-point rule scheme :

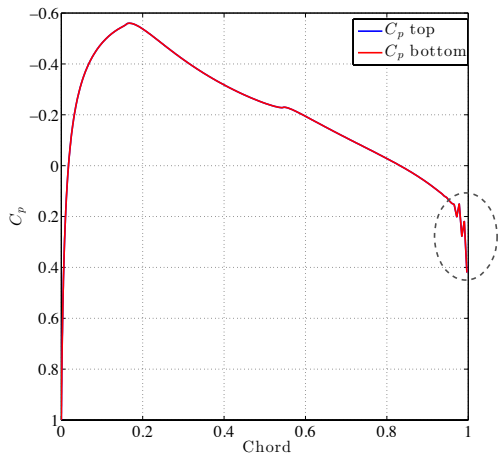
$$C_L = \sum_{i=1}^{N-1} \left(1 - \frac{\|\mathbf{v}_h(\mathbf{C}(\xi_i + h/2))\|^2}{\|\mathbf{v}_\infty\|^2}\right) \|\mathbf{C}'(\xi_i + h/2)\| h,$$

with $\{0 = \xi_1, \xi_2, \ldots, \xi_{N-1}, \xi_N = 1\}$ an uniform partition of the segment $[0, 1]$, $h = \xi_{i+1} - \xi_i$ and $\mathbf{v}_h = \nabla \times \psi_h$ with ψ_h computed with the Galerkin method.

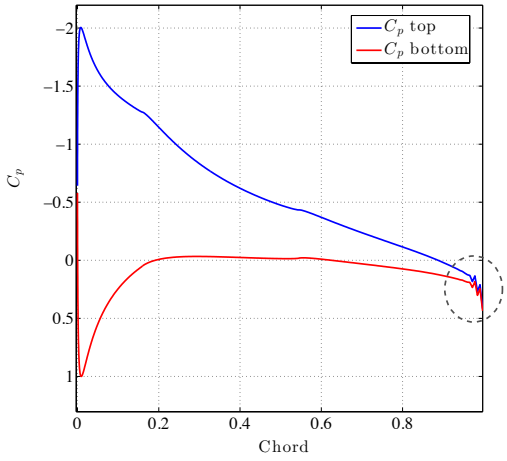
[1] in this case a curve as we are in 2D

Validation of the solver

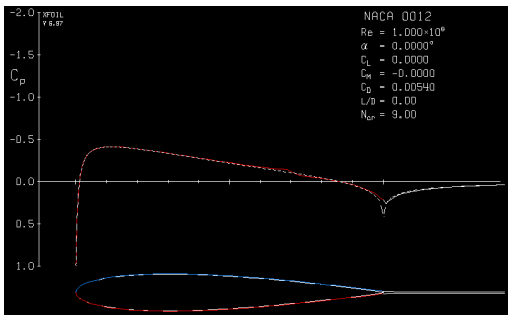
We perform now some qualitative tests on our solver to verify its correctness. We compare the plot of the pressure coefficient versus the chord of the foil with the same plot obtained by the software Xfoil for a NACA0012 foil (see fig. 2.6). We observe that the results obtained by both solvers are very similar, with some minor differences in the values of the pressure coefficient. However, our solver seems to present an abnormal behavior at the vicinity of the trailing edge, where the curve is oscillating. This is a numerical issue, due both to the sharpness of the geometry at this point and to the Kutta condition. These oscillations can be reduced by refining locally the physical grid around the tail, and are not intrinsic to the model. Finally, we observe that the curve is not perfectly smooth, presenting some peaks, since the model is only $\mathcal{C}^1$ continuous. Those peaks correspond to region of high gradient for the pressure coefficient, and can lead to a detachment of the flow from the surface of the foil. Therefore, such peaks have to be avoided, and our shape optimization procedure should try to increase the lift coefficient while smoothing the curve of the pressure coefficient versus the chord.



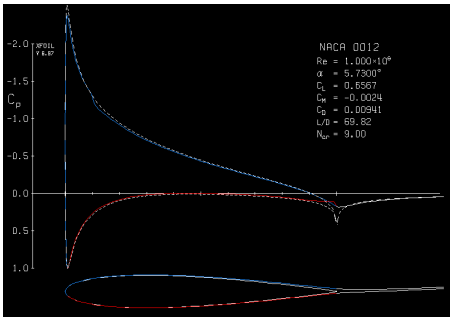
(a) *Angle of attack* $\theta = 0$ *rad*



(b) *Angle of attack* $\theta = 0.1$ *rad*



(c) *Angle of attack* $\theta = 0$ *rad (Xfoil)*



(d) *Angle of attack* $\theta = 0.1$ *rad (Xfoil)*

Figure 2.6: *Pressure coefficient* C_P *versus the chord of the foil for different angles of attack : (a) and (b) have been obtained with our solver, (c) and (d) have been obtained by the software Xfoil (©Mark Drela, MIT) for a NACA0012 foil.*

CHAPTER

3 Shape optimization

We present here a shape optimization procedure aiming to improve some features of the hydrofoil. We start by a theoretical description of the problem in terms of Lagrangian formulation, and then explain the methodology employed to solve the optimization problem numerically. Finally, we present the optimal shape and some of its characteristics.

1 Optimal control theory applied to shape optimization

The lift and pressure coefficients are greatly influenced by the geometry of the hydrofoil. Thus, one interesting problem would be to try to optimize the shape of the hydrofoil in order to modify these quantities in an interesting way for the navigation. Here, we present such a shape optimization procedure, with main objective being to maximize the lift coefficient, defined by :

$$C_L = \oint_\Gamma \left(1 - \frac{\|\mathbf{v}\|^2}{\|\mathbf{v}_\infty\|^2}\right) d\Gamma,$$

with $\mathbf{v} = \nabla \times \mathbf{\Psi}$, $\mathbf{\Psi} = (0, 0, \psi)$ and ψ the stream function, solution to the Laplace problem (2.6). For the sake of simplicity, we first consider modifying the geometry of the foil through the displacement of a single control point $\mathbf{B}_{1,j_0} = \mathbf{P}$ on the NURBS curve Γ_a. This control point $\mathbf{P}$ will be called the **control variable** of our optimal control problem. As changes in the control mesh will automatically induce a change in the solutions ψ and ψ_a of (2.6), we introduce ψ and ψ_a as **state variables** of the optimal control problem. The **cost function** of the optimal control problem is:

$$J(\mathbf{P}, \psi, \psi_a) = -C_L(\mathbf{P}, \psi, \psi_a) + \frac{\alpha}{2}\|\mathbf{P} - \mathbf{P_0}\|^2 + \frac{\beta}{2}\int_\epsilon^{1-\epsilon} \left(\frac{dC_P(x; \mathbf{P}, \psi, \psi_a)}{dx}\right)^2 dx, \quad (3.1)$$

with $\alpha, \beta \in \mathbb{R}_+$ two **penalty parameters**, $\epsilon = 0.1$, x the chord of the foil and $\mathbf{P}_0$ the position of $\mathbf{B}_{1,j_0}$ for some initial geometry. The penalty term $\frac{\alpha}{2}\|\mathbf{P} - \mathbf{P}_0\|^2$ is introduced to ensure the *well-posedness* of the problem. Roughly speaking, by minimizing the cost function J we expect $\mathbf{P}$ to be *relatively close* to $\mathbf{P}_0$, and thus avoid any degenerate shape for the curve Γ resulting from the minimization of the cost function. Moreover, adding this quadratic term improves the *coercivity*[1] of the functional, ensuring the existence of a

[1] the coercivity can be seen as a strong convexity notion

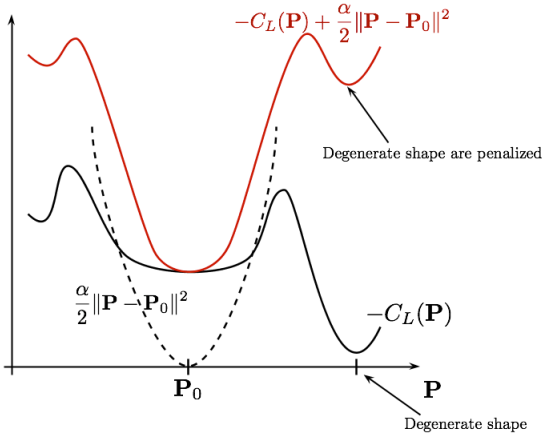


Figure 3.1: *Adding the quadratic penalty term* $\frac{\alpha}{2}\|\mathbf{P} - \mathbf{P}_0\|^2$ *improves the coercivity of the functional and excludes degenerate shape.*

local extremum and facilitating in the convergence of numerical methods (see fig. 3.1). The penalty term $\frac{\beta}{2}\int_\epsilon^{1-\epsilon}\left(\frac{dC_P(\mathbf{P},\psi,x)}{dx}\right)^2 dx$ is introduced to enforce the smoothness of the curve $C_P(x)$ (to avoid detachment of the flow, as discussed in section 2.3). We need now to enforce the state variables ψ and ψ_a to be solutions of the weak problem (2.7). To this end, we employ the **Lagrangian formalism**, and introduce the following **Lagrangian functional**:

$$\begin{aligned}\mathscr{L}(\mathbf{P},\psi,\psi_a,v,\lambda) =& J(\mathbf{P},\psi,\psi_a) + \int_{\Omega(\mathbf{P})} \nabla v\cdot\nabla\psi\, d\Omega(\mathbf{P}) + \int_{\Omega(\mathbf{P})} \nabla v\cdot\nabla R_{\psi_a}\, d\Omega(\mathbf{P}) + \cdots \\ &\cdots + \lambda\left(\int_{\Omega(\mathbf{P})} w_{te}\nabla\psi\cdot\mathbf{n}_{te}\, d\Omega(\mathbf{P}) + \int_{\Omega(\mathbf{P})} w_{te}\nabla R_{\psi_a}\cdot\mathbf{n}_{te}\, d\Omega(\mathbf{P})\right),\end{aligned} \tag{3.2}$$

with $V = H^1_{\Gamma_D}(\Omega)$, $V^* = \mathcal{L}(V)$ the **dual space**[2] of V , $R_{\psi_a} \in H^1(\Omega)$ the lifting function, $(\mathbf{P},\psi,\psi_a) \in \mathbb{R}^2 \times V \times \mathbb{R}$ and $v \in V^*$, $\lambda \in \mathbb{R}$ **lagrange multipliers**.

Then, a theoretical resolution of this problem would imply making the dependency of (3.2) on the control and state variables explicit, in order to compute partial derivatives[3] of the Lagrangian and seek for critical points. However, this is not an easy task, and the calculations involved are tedious. Therefore, we chose not to pursue in this direction, and decided to solve the problem with the Matlab function `fmincon` which, even if less computationally efficient than using methods involving the computation of the derivatives, still provides really good results if properly constrained.

[2] space of all linear functionals on V

[3] using Fréchet derivatives for the state variable ψ and the lagrange multiplier v

2 Resolution of the optimal control problem

The `fmincon` function is a general optimization solver included in the Matlab's Optimization Toolbox which seeks the minimizer of a scalar function of multiple variables, within a region specified by linear constraints and bounds. Starting from an initial estimate, `fmincon` calculates at each iteration the derivatives and the Hessian matrix by the Broyden-Fletcher-Goldfarb-Shanno (BFGS) (see [7]) method, and use them to find a local minimum.
In our case, here are the steps of our algorithm, written in pseudo code (see also fig. 3.2) :

Algorithm 1: Shape Optimization with `fmincon`

Data: An initial configuration $\mathbf{P_0}$
Result: An optimal configuration $\mathbf{P_{opt}}$
$\mathbf{P}^{(0)} = \mathbf{P}_0$;
while $k \leq$ ***Max Iteration*** **do**
  Given the configuration $\mathbf{P}^{(k)}$, create the basic geometry ;
  Refine the physical mesh ;
  Solve the model to compute the objective functional $J^{(k)}$;
  Perform optimality test ;
  if *Optimal design* **then**
    $\mathbf{P}_{opt} = \mathbf{P}^{(k)}$;
    $J_{opt} = J^{(k)}$;
    return $(\mathbf{P}_{opt}, J_{opt})$;
  end
  else
    Update the design $\mathbf{P}^{(k+1)}$;
  end
end

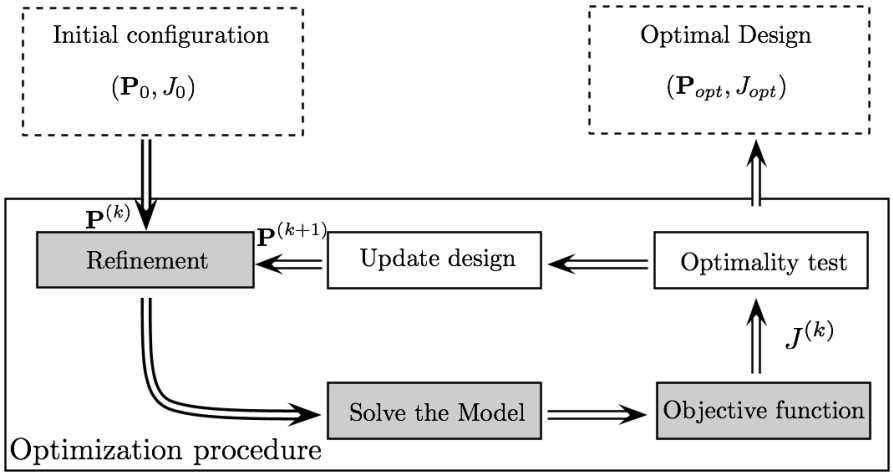


Figure 3.2: ***Scheme of the optimization procedure.*** *The grey boxes are "black boxes" for* `fmincon` *which sees them as a single step* $J^{(k)} = f(P^{(k)})$.

2.1 Setting up the shape optimization procedure

Control variables and initial configuration

We relax here the assumption of moving a single control points, and allow six control points to move in a constrained manner. Therefore, the control variables of our optimal control problem are : $\mathbf{B}_{1,3}, \mathbf{B}_{1,4}, \mathbf{B}_{1,5}, \mathbf{B}_{1,9}, \mathbf{B}_{1,10}$ and $\mathbf{B}_{1,11}$ (see fig. 3.3).

To help in the convergence of the algorithm, we constrain the control points in their displacements. First, we forbid them to cross each other to avoid self-intersecting shapes. Then, we bound them horizontally and vertically. Finally, we impose a minimum and maximum thickness for the foil. All of these constraints are linear constraints that are declared to `fmincon` function. We choose as an initial configuration the foil designed in section 6.1 with an angle of attack of $\theta = 0.1$ rad (see fig. 3.3).

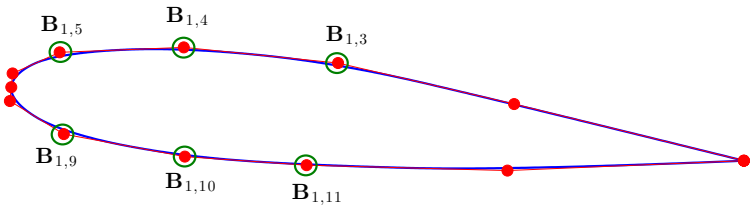


Figure 3.3: ***Initial configuration and control variables.*** *The control points encircled in green are allowed to move, while the other one are fixed. Displacements of the control points are constrained.*

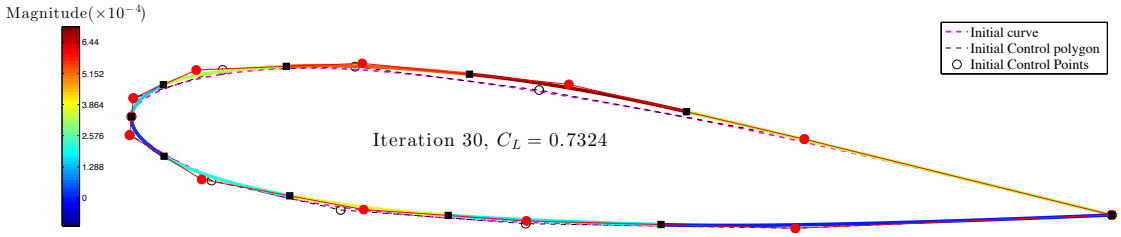
Choice of the penalty parameters

Minimizing the objective functional (3.1) is a competition between maximizing the lift coefficient and minimizing the penalty terms. Thus the choice of the penalty parameters α, β is crucial, as they determine the weight of each penalty. In our context, a high α is crucial, as we want to avoid any degenerate shapes. In comparaison, the smoothness of the curve $C_P(x)$ is less important. We tried to find the right balance between both penalties and finally chose the following values for α and β :

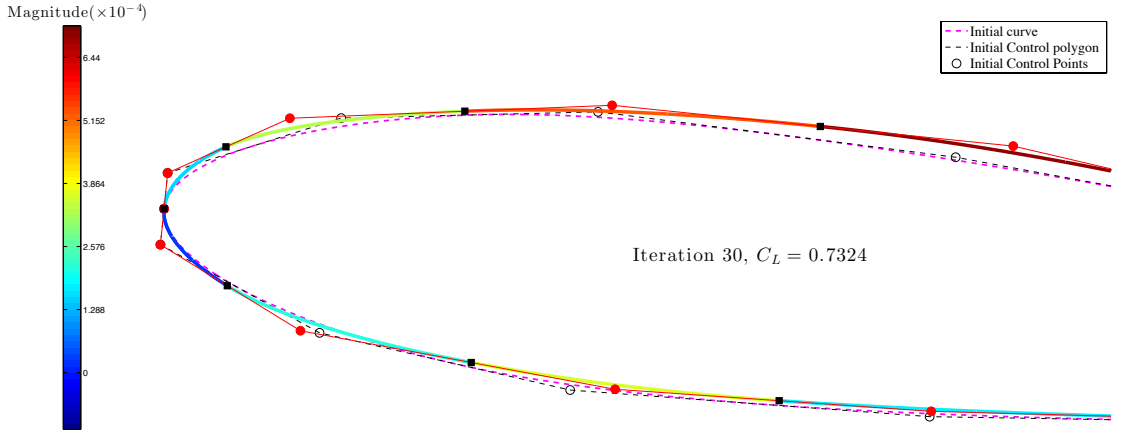
$$\alpha = 25 \times \frac{C_L(\mathbf{P}_0)}{\|\mathbf{P}_0\|^2}, \tag{3.3}$$

$$\beta = 0.01 \times \frac{C_L(\mathbf{P}_0)}{\int_\epsilon^{1-\epsilon} \left(\frac{dC_P(x;\mathbf{P}_0)}{dx}\right)^2 dx}. \tag{3.4}$$

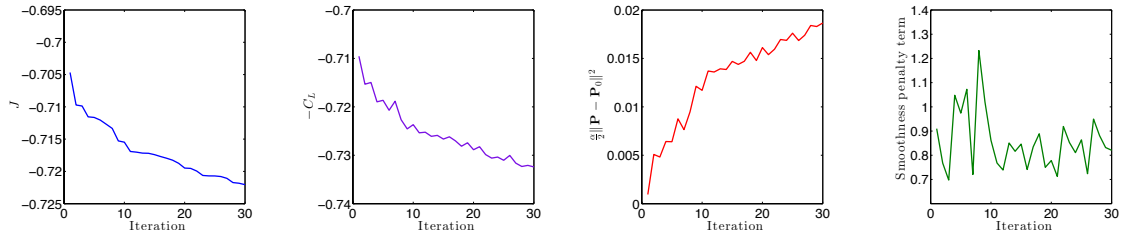
The terms $\frac{C_L(\mathbf{P}_0)}{\|\mathbf{P}_0\|^2}$ and $\frac{C_L(\mathbf{P}_0)}{\int_\epsilon^{1-\epsilon} \left(\frac{dC_P(x;\mathbf{P}_0)}{dx}\right)^2 dx}$ allow to express both penalty terms in the scale of the lift coefficient.



(a) *Optimal shape compared to the initial configuration. The elements have been colored according to the magnitude of their displacement with respect to the initial configuration.*



(b) *Optimal shape compared to the initial configuration (Zoom). The elements have been colored according to the magnitude of their displacement with respect to the initial configuration.*

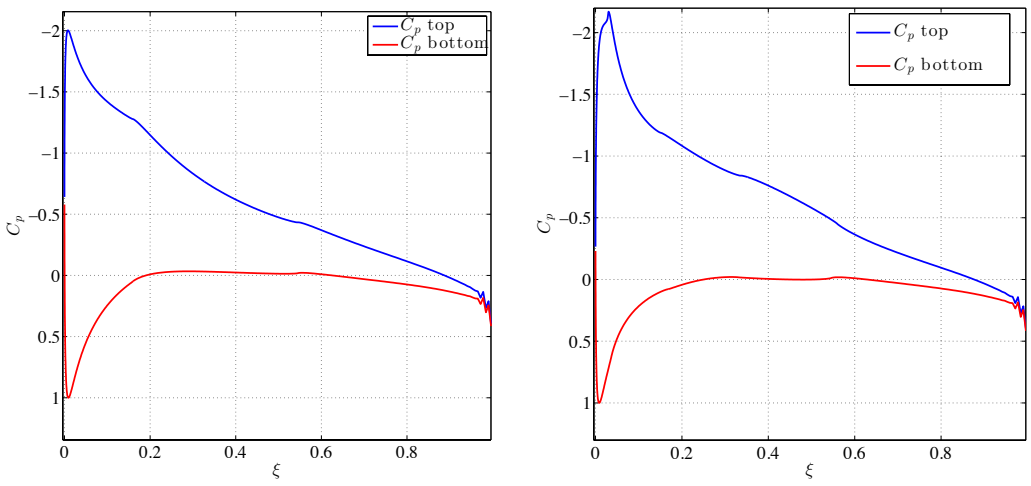


(c) *Evolution of the objective functional, the lift coefficient and the two penalty parameters through the iterations.*

Figure 3.4: *Results of the shape optimization procedure*

2.2 Results of the shape optimization procedure

Figure fig. 3.4 presents the results of the shape optimization procedure. The algorithm converged in 30 iterations, where it reached the optimality condition (the predicted change in the objective functional J was less than 10^{-3}). We observe that the optimal shape is globally close to the original one, which is not surprising, as our initial configuration was close to an industrial design, the NACA0012. We observe that the optimal foil is asymmetric : the top of the hydrofoil has been much more modified by the optimization procedure (see fig. 3.4 (a) and (b)) than the bottom of the foil. It is reassuring to see that the foil is really smooth and does not present any peak or abnormal thickness. On



(a) *Initial design* **(b)** *Optimal design*

Figure 3.5: *Pressure coefficient versus the chord of the foil. Comparaison for the initial and optimal configuration.*

fig. 3.4 (c) we can observe that the objective functional curve is decreasing as $-C_L$ but is smoothed out by both penalty terms. This is a result of the competition between the maximization of the lift coefficient and the minimization of the penalty parameters. The lift coefficient has increased significantly from 0.7 for the initial configuration to 0.733 for the optimal shape. Finally, we can observe on fig. 3.5 the curve of the pressure coefficient C_P versus the chord of the foil. We observe that even if the result is totally acceptable, there is no significant improvement in the smoothness of the curve in comparison to the initial design. We could therefore wonder on the necessity of the penalty term related to the smoothness of the C_p curve. To convince ourself about its utility, we repeated the shape optimization procedure with $\epsilon = 0.2$: we penalized the smoothess of the curve only on the interval $[0.2, 0.8]$. We observe that the curve fig. 3.6 is less smooth with a peak at 0.2., which legitimates the penalization.

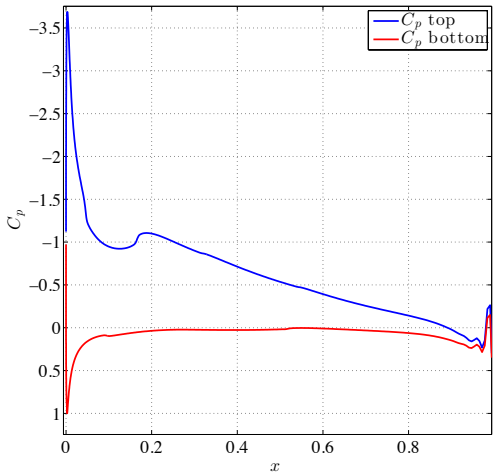


Figure 3.6: *Pressure coefficient versus the chord of the foil for an optimal solution with penalization on the smoothness only on the interval* $[0.2, 0.8]$.

Conclusion

This work provides a testbed for a NURBS-based shape optimization of hydrofoils. All along this study Isogeometric Analysis appeared to be particularly adapted to this kind of problem, as it conveniently encapsulated both the design and the analysis into a unified framework, allowing to work with the exact geometry of the hydrofoil. The shape optimization procedure has been tested in the simple case of a steady irrotational flow around the hydrofoil. Even if this is a quite unrealistic situation, it still remains a good approximation for most of the domain, and the results obtained can provide useful insights for the manufacturing of an hydrofoil. Of course this work would benefit to be extended in greater generality for Fluids Dynamics with fluids described by Naviers-Stokes model.
Finally, explicit computation of the descent direction in the shape optimization problem could be investigated in order to gain in efficiency in the shape optimization procedure.

✧✧✧

Acknowledgments

I wish to acknowledge Dr. Luca DEDE for having proposed me this project and followed me all along its realization. His advices and support have always been very helpful.

Bibliography

[1] J. A. Cottrell, T. J.R. Hughes, Y. Bazilevs *Isogeometric Analysis Toward Integration of CAD and FEA*, WILEY, 2009.

[2] L. Piegl, W. Tiller *The NURBS Book*, Springer, 1997.

[3] F. Brezzi, M. Fortin *Mixed and Hybrid Finite Element Methods*,Springer-Verlag,1995.

[4] A. Quarteroni *Numerical Models for Differential Problems*, Volume 2, Springer, MS&A, 2008.

[5] D. Darmofal *Computational Methods in Aerospace Engineering*, Lecture Notes, Course 16.901, MIT, 2005.

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

[7] J. Nocedal , S.J. Wright, *Numerical Optimization*, Springer, New York, 2006.