Generalizations of NURBS and their applications in CAGD, and isogeometric analysis

By

Alireza H. Taheri

A dissertation submitted in partial fulfillment of

the requirement of the degree of

Doctor of Philosophy

(Mechanical Engineering)

at the

UNIVERSITY OF WISCONSIN – MADISON

2021

Date of final oral examination: 12 Jan. 2021

The dissertation is approved by the following members of the Final Oral Committee:

Krishnan Suresh, Professor, Mechanical Engineering

Xiaoping Qian, Professor, Mechanical Engineering

Shiva Rudraraju, Assistant Professor, Mechanical Engineering

Corinne Henak, Assistant Professor, Biomedical Engineering

Dan Negrut, Professor, Mechanical Engineering

Generalizations of NURBS and their applications in CAGD, and isogeometric analysis

Abstract

Alireza H. Taheri

Under the Supervision of Professor Krishnan Suresh

At the University of Wisconsin – Madison

Non-Uniform Rational B-Splines (NURBS) are one of the most popular curve and surface representations today. Besides forming the foundations of computer aided geometric design (CAGD), they are also extensively used in a variety of engineering applications including isogeometric analysis (IGA), shape optimization, topology optimization, material modeling, reverse engineering, G-code generation, bio-engineering etc. The focus of this research is to improve the flexibility of NURBS, and particularly explore remedies for one of its long-lasting deficiencies, namely its inability to accurately capture discontinuities and steep local gradients in certain applications. This shortcoming reveals itself in many applications where NURBS is employed for the approximation of sharply varying fields or functions.

The proposed improvement is achieved by developing various generalizations of NURBS, referred to as GNURBS, obtained through either explicit or implicit decoupling of the weights associated with basis functions along different physical coordinates. It will be seen that this simple, yet unexplored basic idea significantly improves the capability of NURBS in aforementioned applications. The theoretical properties of these variations for both curves and surfaces are investigated, and sound mathematical proofs are provided. Further, it is shown that these representations can be used for improved construction of certain class of curves and surfaces in Computer Aided Geometric Design (CAGD). To better demonstrate the behavior and abilities of GNURBS in comparison to NURBS, two interactive MATLAB toolboxes for curves and surfaces have been developed and introduced.

Nevertheless, the main focus of this thesis is exploring the application of GNURBS in isogeometric analysis for improved solution of boundary value partial differential equations. This is achieved by devising a novel adaptivity technique in isogeometric analysis, referred to as adaptive *w*-refinement. A natural extension of IGA based on GNURBS is introduced where the weights of the basis functions in geometry and solution space are decoupled. Considering the additional unknown control weights in the solution function space as design variables, we develop an adaptive algorithm to find these unknowns by solving an unconstrained optimization problem. This procedure leads to the optimal rational function space associated with the problem under study, while preserving the underlying geometry as well as its parameterization.

We study the performance of this algorithm on elliptic problems with both smooth and rough solutions. Numerical results demonstrate significant improvement of accuracy as well as the convergence rate compared to classic NURBS-based IGA. Moreover, the proposed method enables the isogeometric method to solve problems, whose closed-form solutions lie in rational space, exactly, revealing a new crucial aspect of employing rational splines for analysis. The proposed adaptive *w*-refinement technique serves as a new powerful adaptive technique in IGA, and perhaps a competitive tool with hierarchical splines for alleviating the deficiencies of NURBS for analysis.

The proposed ideas in this thesis open doors to further research in a wide range of applications of NURBS. Further, these ideas inspire the possibility of developing additional generalizations of NURBS. These additional applications and alternative generalizations are briefly discussed in the last part of this thesis.

Acknowledgements

I would like to sincerely thank my family members and all my friends who truly supported me throughout my graduate program. I would also like to thank my advisor, my lab-mates, and all the committee members who helped me by sharing their insightful feedback on my research.

In particular, I would like to give special thanks to faculty members Prof. Hassani, Dr. Talebizadeh Sardari, Prof. Qian, Prof. Rudraraju, Dr. Calabro and Dr. Abolghasemi who unconditionally assisted me in various aspects of my research. It would have been impossible for me to resolve all the technical challenges I faced during this research without their kind help and support.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1. Historical background of NURBS

Non-Uniform Rational B-Splines (NURBS) are perhaps the most popular curve and surface representation method in Computer-Aided Design/Computer-Aided Manufacturing (CAD/CAM). They were first introduced in 1975 by Versprille [1] as rational extension of B-splines. NURBS form the backbone of CAD, and are considered the dominant technology for engineering design [2]; further, they have also been extensively used in several applications including isogeometric analysis (IGA) [3], NURBS-augmented finite element analysis [4], shape optimization [5,6], topology optimization [7,8], material modeling [9,10], reverse engineering [11], G-code generation [12] etc.

Recent generalizations of NURBS-based technology include T-splines [13,14] which constitute a superset of NURBS, and provide the local refinement properties by allowing for some unstructured-ness. An alternative generalization of NURBS, referred to as Generalized Hierarchical NURBS (H-NURBS), were introduced in 2008 by Chen et al. [15] by extending the idea of hierarchical B-splines to NURBS. Similar to T-splines, H-NURBS primarily bring the possibility of local refinement with tensor-product surfaces. A novel shape-adjustable generalized Bézier curve with multiple shape parameters has been recently proposed by Hu et al. [16], and its applications to surface modelling in engineering has been studied. Most recent class of splines which removes the limitations of T-splines are Unstructured-splines (U-splines) that have been developed by Thomas [17].

Other generalizations of NURBS have also been suggested in the literature, even though these representations have not gained popularity. For instance, Wang et al. [18] propose a generalized NURBS curve and surface representation with the primary advantage of representing smooth surfaces with genus zero using only one surface patch. This also provides a new method to exactly generate conic curves and revolution surfaces. Further, it simplifies modelling local features such as creases and ruled patches.

Historically, NURBS were primarily introduced to represent conical shapes precisely. This is the critical advantage of NURBS over other polynomial-based classes of splines, and the main reason

for its prevalence. This is achieved by the introduction of weights into the basis functions in a rational manner. The applications of this rational form, however, is not limited to precise construction of conics. According to the literature, there are other applications in CAGD where the weights have been employed as additional degrees of freedom for improved flexibility.

For instance, the weights can be employed as additional design variables for interactive shape design so that one can utilize both control point movement, and weight modification to attain local shape control [19]. Many studies suggest employing the weights as additional design variables in data-fitting for better accuracy [11,20]. Carlson [20] develops a non-linear least square fitting algorithm based on NURBS, and discusses multiple methods for solving this problem. His numerical results demonstrate significant improvement in the accuracy of approximation compared to B-splines, especially in the case of rapidly varying data. This is in fact one of the other main advantages of NURBS over B-splines. While smooth piecewise polynomials such as B-splines are poor in the approximation of rapidly varying data and discontinuities, employing rational functions is an effective tool for addressing this class of problems [20]. In order to avoid solving a non-linear optimization problem, Ma [11,21] develops a two-step linear algorithm for data approximation using NURBS.

1.2. Motivation for introducing Generalized NURBS (GNURBS)

Despite being an effective technique for improving the performance of NURBS, there is a wide range of applications where treating the weights as extra design variables is either impossible or can be problematic. For instance, Dimas and Briassoulis [22] point out that a bad choice of weights in approximation can lead to poor curve/surface parameterization. Piegl [23] mentions that "*improper application of the weights can result in a very bad parameterization, which can destroy subsequent surface constructions*". Further, there are numerous applications where employing the weights as additional design variables is essentially impossible. We will later discuss some of these applications in CAGD in Chapters 2 and 3, as well as in computational mechanics in Chapter 4.

The focus of this thesis is to develop new generalizations of NURBS to primarily address this shortcoming. These proposed generalizations improve the performance of NURBS, and provide an alternative concept for removing the deficiencies of NURBS. It will be shown that, unlike T-splines, these generalizations are only variations of classic NURBS, and do not constitute a new superset of NURBS, making it easy to integrate and deploy them in modern CAD/CAM systems.

1.3. Application of GNURBS in isogeometric analysis

Isogeometric analysis (IGA) was introduced by Hughes et al. [3] as an innovative numerical methodology for the solution of boundary value problems. In contrast to classic Finite Element Method (FEM), IGA is more tightly integrated with the geometry, and circumvents the requirement for a conventional mesh generation process, via direct communication with CAD models. Moreover, it has been shown that, when deployed for analysis, higher-order smooth spline bases commonly used in CAGD yield superior results in terms of accuracy and robustness compared to standard C^0 discretizations. This has been demonstrated in a variety of application areas such as structural, fluid, etc. [2].

While offering many well-known advantages over classic FEM, the ultimate success of the method in integrating design and analysis is mainly contingent upon the development of modern spline technologies which sufficiently meet the demands of both analysis and design. Extensive research has been conducted towards this, and immense progress has been made in various aspects. A major difficulty which has attracted significant attention is the inability of piecewise smooth tensor product splines in solving problems with irregularities such as sharp layers or singularities. As will be discussed further below, the same concept of providing the possibility of local *h*-refinement in splines has been commonly pursued for alleviating this issue by the community so far.

In the following sections, we provide an overview of these studies. We will later explore an alternative powerful technology that isogeometric analysis exclusively provides for addressing the above-mentioned problems. The proposed method provides the possibility of enrichment of function space without introducing additional basis functions. This novel adaptivity technique, which will be referred to as *adaptive w-refinement*, is established based on the proposed generalizations of NURBS.

1.3.1. Refinement techniques in isogeometric analysis

One of the most interesting aspects of using splines as the basis for analysis is the possibility of exploiting multiple elegant and efficient techniques that they provide for the enrichment of function space. Having an initial parameterization of computational model, a variety of refinement techniques can be used to improve the accuracy of approximation in IGA. One may classify these

techniques into two categories, namely, *function space refinement*, and *control net refinement* techniques described next.

Function space refinement techniques attempt to enrich the function space while preserving the underlying geometry and its parameterization unchanged. The existing techniques in this class are *h*, *p*, and *k* refinements. The details of these algorithms can be found in [2]. While *h* and *p* refinements are common to both classic FEM and IGA, *k*-refinement is exclusive to IGA. This additional possibility of refinement is one of the key advantages of IGA over classic FEM as it provides higher order continuity by performing degree elevation followed by knot insertion in a special manner. Finally, we note here that these algorithms may be employed in combination for improved performance, usually in an adaptive manner, as in *hp*-adaptive refinement [24], etc.

In contrast to function space refinement, the second class of *control net refinement* adaptively modifies an initial parameterization to improve the accuracy of approximation *without* enrichment of the function space while preserving the boundaries of the geometric domain. Strategies in this category are usually referred to as *r-refinement* [25,26]. This refinement is usually posed as an optimization problem which aims at minimization of an estimation of the error, obtained using a posteriori error estimation technique, by adaptively repositioning the *interior* control points; see e.g. [27]. For a review of these studies please see [25]. There are however many deficiencies in this method discussed below which makes it impractical, especially for large scale problems.

For instance, the above-mentioned procedure leads to solving a heavily constrained non-linear optimization problem since the parameterization must remain valid (bijective) throughout the optimization process. This involves imposing a large number of constraints to ensure the positivity of Jacobian over the whole domain [27]. Another major shortcoming of these methods is that they are only applicable to problems which have an *interior* region and do not apply for problems with an arbitrary geometry such as free-form shells or curved beams. Further, derivation of analytical sensitivities does not seem possible; hence, the existing studies, e.g. [27], rely on finite difference method which makes the algorithm prohibitively expensive. Finally, to the best of our knowledge, none of the existing studies in this class report a substantial improvement in the accuracy or rate of convergence.

1.3.2. The necessity for unstructured splines

Despite providing the above-mentioned effective refinement techniques, in contrast to the standard nodal basis commonly used in FEA, a multivariate tensor-product spline basis, such as NURBS, does not provide a natural possibility for local mesh refinement. In fact, this was soon known as a fundamental limitation of IGA since the possibility of adaptive local mesh refinement is critical in FEA, and is commonly used to resolve local features such as internal and boundary layers in advection dominated flows and stress concentrations in structures [28].

Eliminating this limitation by modifying the existing spline technologies or developing new variations of splines has perhaps been the most active area of research in IGA community over the last decade; see, for example, [28–35]. We do not intend to review these studies here; instead we refer to [36,37] for a comprehensive review. We suffice to mention here that the primary purpose of these studies is to open up the possibility of local *h*-refinement in splines by allowing for some unstructuredness. Various forms of T-splines [31,38–50], subdivision basis functions coupled with the truncation mechanism [51–53], LR-splines [54], (truncated) hierarchical B-splines [29,55,56], (R)PHT-splines [34,57], and most recently U-splines [17] are some of the popular technologies in this category. This concept has shown promising results for resolving local features and achieving an improved rate of convergence in problems with poor regularities that contain steep layers or singularities [28]. These techniques have also been incorporated with commercial FEM software such as Abaqus and LS-DYNA to support real engineering IGA applications [58].

In one of these studies, which is of particular interest to the current research, Atroshchenko et al. [59] suggest a generalization of IGA by weakening the tight coupling between geometry and solution space. This concept, which is referred to as Geometry-Independent Field approximaTion (GIFT) by its authors, allows for different spaces for the parameterization of the computational domain and approximation of the solution field. They argue that this method inherits the main advantage of IGA by preserving the original exact CAD representation of the geometry, such as NURBS, but allows for pairing it with an approximation space, such as T-splines, LR-splines, (truncated) hierarchical B-splines, or PHT-splines, which is more suitable/flexible for analysis [59]. In particular, it offers the advantage of adaptive local refinement without the need to reparametrize the domain, and therefore without losing the link with the CAD model. They study the performance of this method with different choices of geometry and field spaces and

demonstrate that, despite the failure of the standard patch test, the optimum convergence rate is achieved for non-nested spaces [59].

1.3.3. Adaptive w-refinement in GNURBS-based IGA

Despite being an effective technique for improving the performance of NURBS in a variety of applications in CAGD, considering the literature of IGA, the application of NURBS as well as other rational splines surprisingly seems to be merely limited to precise representation of conical shapes in this area. In this thesis, by taking inspiration from the concept of GIFT [59], we study the application of GNURBS for improved solution of boundary value problems using IGA. It will be seen that this proposed refinement technique provides an alternative powerful tool for removing the deficiencies of NURBS for analysis. Detailed discussion of the proposed refinement technique will be provided in Chapter 4.

1.4. Organization of the thesis

The remainder of this thesis is organized as follows: in Chapter 2, we introduce two different generalizations of NURBS for parametric curves, obtained via explicit or implicit decoupling of the weights along different physical coordinates, and develop their theoretical properties. We explore some of the applications of GNURBS curves in CAGD in this chapter, and compare their performance against classic NURBS. A developed interactive MATLAB toolbox for GNURBS curves, referred to as GNURBS-Lab will also be introduced here.

In Chapter 3, similar generalizations of NURBS are developed for bi-variate parametric surfaces. Their theoretical properties are discussed and a few applications of these representations in the context of CAGD are investigated. These generalizations will form the basis for developing adaptive *w*-refinement which will be discussed in the proceeding chapter. An extension of GNURBS-Lab for surfaces, named GNURBS3D-Lab, has also been developed which will be introduced briefly at the end of this chapter.

In Chapter 4, the application of GNURBS for improved solution of boundary value partial differential equations is investigated. This is achieved by developing a novel adaptive procedure in GNURBS-based IGA, referred to as adaptive *w*-refinement. We study the performance of this algorithm on elliptic problems with both smooth and rough solutions. Moreover, the performance of the proposed method in solving problems whose closed-form solutions lie in rational space is

investigated. It will be seen that the proposed adaptive *w*-refinement technique serves as a new powerful adaptive technique in IGA, and perhaps a competitive tool with hierarchical splines for alleviating the deficiencies of NURBS for analysis.

Finally, Chapter 5 summarizes the main findings of this thesis. Moreover, further potential applications of GNURBS as well as other possible generalizations of NURBS are elaborated in this chapter.

Chapter 2: Generalizations of NURBS curves[1]

In this chapter, we introduce two different generalizations of NURBS curves obtained by decoupling of the weights along physical coordinates in different manners. The theoretical properties of these representations in comparison to classic NURBS are thoroughly discussed. Further, the application of these generalizations for improved approximation of different class of functions is studied.

2.1. Generalized NURBS Curves: non-isoparametric form via explicit decoupling of the weights

We recall that the equation of a NURBS curve is parametrically defined as

$$\mathbf{C}(\xi)=\sum_{i=0}^{n} R_{i,p}(\xi)\mathbf{P}_i, \qquad a\le\xi\le b \tag{2.1}$$

where $\mathbf{P}_i$ are a set of $n+1$ control points and $R_{i,p}(\xi)$ are the corresponding rational basis functions associated with i^{th} control point defined as

$$R_{i,p}(\xi)=\frac{N_{i,p}(\xi)w_i}{\sum_{j=0}^{n} N_{j,p}(\xi)w_j} \tag{2.2}$$

where w_i are the weights associated with control points, and $N_{i,p}(\xi)$ are the B-spline basis functions of degree p, defined on a set of non-decreasing real numbers $\Xi=\{\xi_0,\xi_1,...,\xi_{n+p}\}$ called knot vector. $N_{i,p}(\xi)$ is recursively defined as:

$$\begin{aligned} N_{i,0}(\xi)&=\begin{cases}1 & \text{if } \xi_i\le\xi<\xi_{i+1}\\ 0 & \text{otherwise}\end{cases}\\ N_{i,p}(\xi)&=\frac{\xi-\xi_i}{\xi_{i+p}-\xi_i}N_{i,p-1}(\xi)+\frac{\xi_{i+p+1}-\xi}{\xi_{i+p+1}-\xi_{i+1}}N_{i+1,p-1}(\xi)\end{aligned} \tag{2.3}$$

[1] The presented materials in this chapter have been published in Engineering with Computers journal, ref. [78]

The NURBS curve in Eq. (2.1) is a vector equation which, assuming $\mathbf{P}_i = [x_i \;\; y_i \;\; z_i]^T$, could be written in the following expanded form in 3D space

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{n} R_{i,p}(\xi) \begin{Bmatrix} x_i \\ y_i \\ z_i \end{Bmatrix} \tag{2.4}$$

As observed in the above equation, NURBS curves are *isoparametric* representations where all the physical coordinates are constructed by linear combination of the same set of scalar basis functions in parametric space. This is the case for all the other popular CAGD representations, e.g. all different types of splines; and ensures significant properties such as affine invariance and convex hull which are of interest in geometric modelling.

We introduce here the concept of Generalized Non-Uniform Rational B-Splines (GNURBS) by the extension of the above equation as follows

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{n} \begin{Bmatrix} R^x_{i,p}(\xi)\, x_i \\ R^y_{i,p}(\xi)\, y_i \\ R^z_{i,p}(\xi)\, z_i \end{Bmatrix} \tag{2.5}$$

where $[R^x_{i,p}(\xi),\, R^y_{i,p}(\xi),\, R^z_{i,p}(\xi)]^T$ is now a vector set of basis functions which is defined as

$$\begin{bmatrix} R^x_{i,p}(\xi) \\ R^y_{i,p}(\xi) \\ R^z_{i,p}(\xi) \end{bmatrix} = \left[\frac{N_{i,p}(\xi) w_i^x}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^x}, \;\; \frac{N_{i,p}(\xi) w_i^y}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^y}, \;\; \frac{N_{i,p}(\xi) w_i^z}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^z} \right]^T \tag{2.6}$$

where $\left(w_i^x, w_i^y, w_i^z\right)$ is the set of coordinate-dependent weights associated with i^{th} control point. Denoting the vector set of basis functions in Eq. (2.6) by $\mathbf{R}_{i,p}(\xi) = [R^x_{i,p}(\xi),\, R^y_{i,p}(\xi),\, R^z_{i,p}(\xi)]^T$, the equation of a GNURBS curve can be written in the following compact form

$$\mathbf{C}(\xi) = \sum_{i=0}^{n} \mathbf{R}_{i,p}(\xi) \odot \mathbf{P}_i, \qquad a \le \xi \le b \tag{2.7}$$

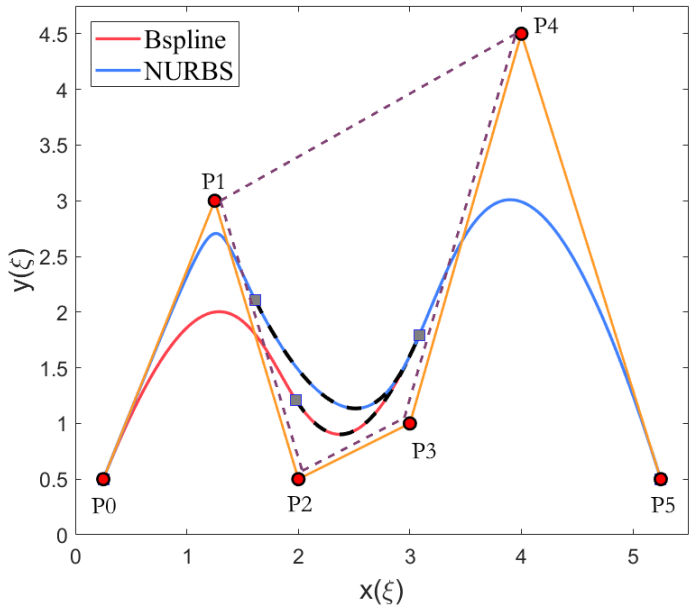
where $\odot$ denotes Hadamard (entry-wise) product of two vector variables.

Comparison of the above equation with that of classic NURBS shows that the main difference of the proposed generalized form is assigning independent weights to different physical coordinates of control points. As can be seen, the above leads to a *non-isoparametric* representation. This modification results in loss of properties such as strong convex hull and affine invariance. However, it will be established that GNURBS are only disguised forms of higher-order classic NURBS, i.e., all the properties of NURBS can be recovered through a suitable transformation, thus ensuring a strong theoretical foundation. In the following section, we develop the theory of GNURBS, and discuss how the properties of this non-isoparametric representation compare to those of NURBS.

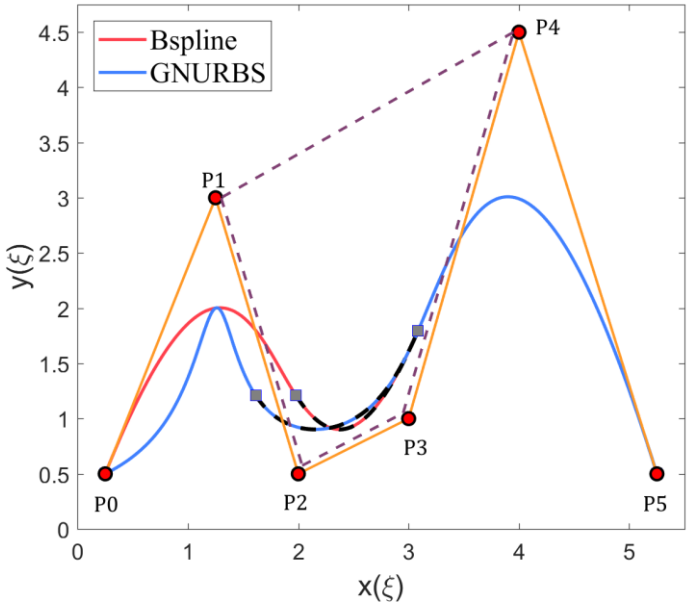
2.1.1. Theory and properties

It can be easily shown that many properties of NURBS curves elaborated in [19] such as end-points interpolation, continuity, etc. are similarly satisfied in GNURBS. However, when treated in the direct form, some of the NURBS properties will be modified or even violated. We first discuss these, and later show how a simple transformation can be applied to recover all NURBS properties.

1. Affine invariance: Due to coordinate-dependence of the basis functions in GNURBS, applying an affine transformation directly to the control points will not result in the same curve as applying the same transformation to the curve; hence, this property is not satisfied.
2. Strong convex hull: A GNURBS curve need not lie in the convex hull of its control points. We demonstrate this graphically in Figure 2.1 for a cubic curve ($p = 3$) constructed on the knot vector $\boldsymbol{\Xi} = \{\xi_0, \xi_1, ..., \xi_9\} = \{0,0,0,0,⅓,⅔,1,1,1,1\}$. Figure 2.1(a) shows a B-spline curve and a NURBS curve with $\{w_0, ..., w_5\} = \{1,5,1,1,1,1\}$ constructed using the same control polygon. As observed, by increasing w_1 the middle knot span $\xi \in [\xi_4, \xi_5)$ always lies within the convex hull of control points $\{\mathbf{P}_1, \mathbf{P}_2, \mathbf{P}_3, \mathbf{P}_4\}$. Figure 2.1(b) illustrates an example where the same knot span of a cubic GNURBS curve constructed with the same control polygon but a decoupled set of weights $\{w_0^x, ..., w_5^x\} = \{1,5,1,1,1,1\}$ and $\{w_0^y, ..., w_5^y\} = \{1,1,1,1,1,1\}$ exits the convex hull of its control points. However, we prove that it satisfies a weaker condition referred to as "*axis-aligned bounding box*" property described below.



(a)



(b)

Figure 2.1. (a) A NURBS knot-span lies inside the convex hull of its control points. (b) A GNURBS knot-span need not lie inside the convex hull of its control points.

The function spaces corresponding to Figure 2.1 are depicted in Figure 2.2. Observe that the function space associated with the NURBS curve in Figure 2.1(a) is identical for both x and y physical components, i.e. $R(\xi)$. Nevertheless, in the case of GNURBS curve shown in Figure 2.1(b), the x-coordinate is constructed using the rational set of basis

functions $R(\xi)$, while the *y*-coordinate is constructed using the set of B-spline basis functions $N(\xi)$.

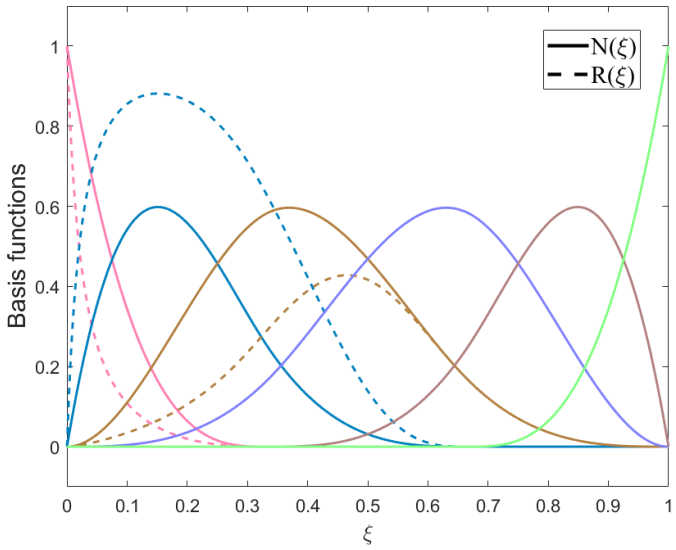


Figure 2.2. Cubic function spaces corresponding to Figure 2.1: B-spline function space $N(\xi)$, and NURBS function space $R(\xi)$ with $\{w_0, \ldots, w_5\}=\{1,5,1,1,1,1\}$.

3. Axis-aligned bounding box (AABB): Every GNURBS knot span lies within the *axis-aligned bounding box* of its corresponding control points. That is, if $\xi \in [\xi_i, \xi_{i+1})$, then $\mathbf{C}(\xi)$ lies within the bounding box of the control points $\{\mathbf{P}_{i-p}, ..., \mathbf{P}_i\}$.

Proof:

Eq. (2.5) can be easily written in the following form:

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{n} R^x_{i,p}(\xi) \begin{Bmatrix} x_i \\ 0 \\ 0 \end{Bmatrix} + \sum_{i=0}^{n} R^y_{i,p}(\xi) \begin{Bmatrix} 0 \\ y_i \\ 0 \end{Bmatrix} + \sum_{i=0}^{n} R^z_{i,p}(\xi) \begin{Bmatrix} 0 \\ 0 \\ z_i \end{Bmatrix} \tag{2.8}$$

Accordingly, Eq. (2.7) could be written as

$$\mathbf{C}(\xi) = \mathbf{C}_x(\xi) + \mathbf{C}_y(\xi) + \mathbf{C}_z(\xi), \qquad a \le \xi \le b \tag{2.9}$$

where $\mathbf{C}_x(\xi)$, $\mathbf{C}_y(\xi)$ and $\mathbf{C}_z(\xi)$ are simply classic NURBS curves. From a geometric standpoint, each of these curves is the projection of the original non-isoparametric curve onto the corresponding physical axis. The following figure shows a graphical representation of above equations for a 2D cubic curve constructed over the knot vector $\Xi = \{0,0,0,0,\frac{1}{3},\frac{2}{3},1,1,1,1\}$.

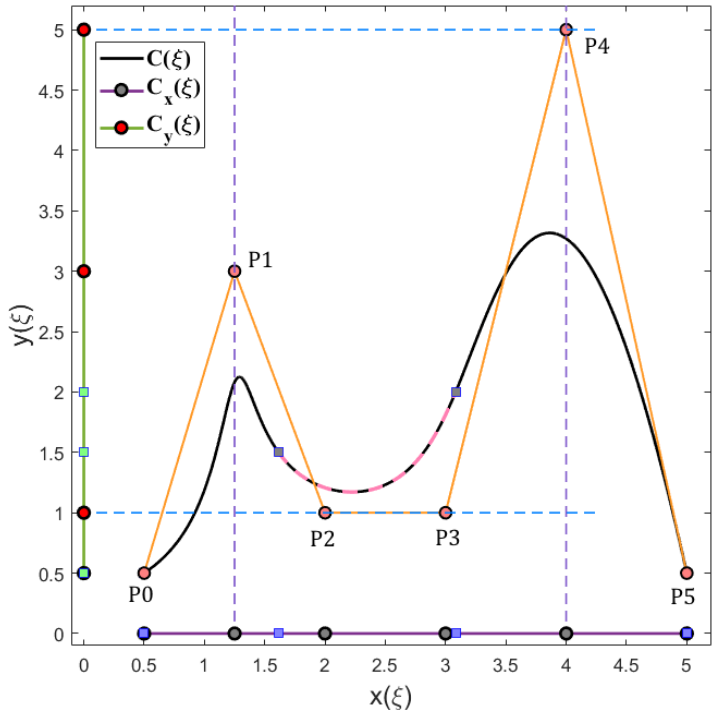


Figure 2.3. Graphical representation of the bounding box property of a 2D cubic GNURBS curve with $\{w^x_0, \dots, w^x_5\}=\{1,5,1,1,1,1\}$ and $\{w^y_0, \dots, w^y_5\}=\{1,1,1,1,1,1\}$.

4. Since each of these curves is a classic NURBS curve, they satisfy the convex hull property. Therefore, the middle knot span of the curve which is marked in Figure 2.3, must lie within the convex hulls of its corresponding control points on both projected curves. That is, if $\xi \in \left[\frac{1}{3}, \frac{2}{3}\right)$, then $\mathbf{C}_x(\xi)$ lies within the convex hull of the control points $\{x_1, \dots, x_4\}$ which is the region between the two vertical lines in Figure 2.3. Similarly, $\mathbf{C}_y(\xi)$ lies within the convex hull of the control points $\{y_1, \dots, y_4\}$ which is the area between the two horizontal lines in this figure. Consequently, $\mathbf{C}(\xi)$ is contained in the intersection of these two convex hulls, which is the rectangular area shown in Figure 2.3, referred to as the bounding box of $\{\mathbf{P}_1, \dots, \mathbf{P}_4\}$.

5. Local Modification: Similar to NURBS, one can show that, in GNURBS, if a control point $\mathbf{P}_i$ is moved, or if any of the weights $w_i^d (d = x, y, z)$ is changed, it affects only the curve segment over the interval $[\xi_i, \xi_{i+p+1})$. However, unlike NURBS, changing the weights will only affect the parameterization of the curve along the corresponding physical coordinate d, while the curve parameterization in the other directions will be preserved. This is, in

fact, the key difference between GNURBS and NURBS which increases control. Assuming $\xi \in [\xi_i, \xi_{i+p+1})$, if w_i^d is increased (decreased), the curve will move closer to (farther from) $\mathbf{P}_i$. Further, for a fixed ξ, a point on $\mathbf{C}(\xi)$ moves along a horizontal (vertical) straight line as a weight w_i^x (w_i^y) is modified; see Figure 2.1(b). This can be easily concluded from the proposed decomposition in Eq. (2.8) and the properties of classic NURBS curves.

6. Variation Diminishing Property: Due to loss of convex-hull property, this property is also not preserved in the direct form of GNURBS; that is, since the curve does not need to lie within the convex hull of its control points, there can be a plane (line in 2D) which intersects the curve multiple times without having any intersections with the control polygon.
7. NURBS Inclusion: If the weights in all directions are equal for each control point, then the GNURBS curve reduces to a NURBS curve.

Having discussed the properties of GNURBS in the direct form, we now develop a transformation of GNURBS into an equivalent NURBS of a higher order. Towards this end, we first review two lemmas on the multiplication of Bézier, as well as B-spline functions. The proofs of these lemmas can be found in [60].

Lemma 1:

Let $f_b(\xi)$ and $g_b(\xi)$ be two Bézier functions of degree p and q, respectively. Their product function $h_b(\xi)$ is a Bézier function of degree $p+q$ which can be computed as [61]

$$h_b(\xi) = f_b(\xi) g_b(\xi) = \sum_{k=0}^{p+q} B_{k,p+q}(\xi) h_k^b \tag{2.10}$$

where $B_{k,p+q}(\xi)$ denotes k^{th} Bézier basis function of degree $p+q$, and

$$h_k^b = \sum_{j=\max(0,k-q)}^{\min(p,k)} \frac{\binom{p}{j}\binom{q}{k-j}}{\binom{p+q}{k}} f_j g_{k-j} \tag{2.11}$$

End of Lemma 1

Lemma 2:

Let $f(\xi)$ and $g(\xi)$ be two univariate B-spline functions of degree *p* and *q*, respectively. Their product function $h(\xi)$ is a B-spline function of degree *p*+*q*, i.e.

$$h(\xi) = f(\xi)g(\xi) = \sum_{k=0}^{n_h} N_{k,p+q}(\xi) h_k \tag{2.12}$$

where h_k are the ordinates of the product B-spline function.

End of Lemma 2

Specific to Lemma 2, numerous algorithms have been proposed in the literature for evaluating the ordinates; see [62–65], for instance. In this paper, we will use a straightforward algorithm proposed by Piegl and Tiller [61] including three steps of

- Performing Bézier extraction
- Computation of the product of Bézier functions
- Recomposition of the Bézier product functions into B-spline form using knot removal.

The product of Bézier functions in the second step can be computed analytically employing Lemma 1. Further, one can construct the knot vector of $h(\xi)$ as described in [61]. A more advanced algorithm referred to as Sliding Windows Algorithm (SWA) recently proposed by Chen et al. could be found in [63].

The decomposition in Eq. (2.8) together with the above two Lemmas lead to the following interesting theorem on the equivalence of NURBS and GNURBS.

Theorem: Every GNURBS curve of degree *p* and dimension m can be transformed exactly into a NURBS curve of degree $m \times p$.

Proof. We provide the proof here for a 2D curve, however, it can easily be extended to any higher dimension. The proof relies on the lemma that the summation of two NURBS curves is a higher order NURBS curve [61]. We rewrite Eq. (2.8) for a 2D curve in the following form:

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \end{Bmatrix} = \frac{\sum_{i=0}^{n} N_{i,p}(\xi) w_i^x}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^x} \begin{Bmatrix} x_i \\ 0 \end{Bmatrix} + \frac{\sum_{i=0}^{n} N_{i,p}(\xi) w_i^y}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^y} \begin{Bmatrix} 0 \\ y_i \end{Bmatrix} \tag{2.13}$$

Extracting the common denominator leads to:

$$x(\xi)=\frac{\left(\sum_{i=0}^{n}N_{i,p}(\xi)w_i^x x_i\right)\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^y\right)}{\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^x\right)\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^y\right)}$$
$$y(\xi)=\frac{\left(\sum_{i=0}^{n}N_{i,p}(\xi)w_i^y y_i\right)\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^x\right)}{\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^y\right)\left(\sum_{j=0}^{n}N_{j,p}(\xi)w_j^x\right)} \tag{2.14}$$

As can be observed, evaluation of Eq. (2.14) involves performing the multiplication of univariate B-spline functions. According to Lemma 2, the product functions in Eq. (2.14) are B-spline functions of degree 2*p*. Therefore, we can obtain the equivalent higher order NURBS representation of Eq. (2.13) in the following form

$$\begin{Bmatrix}x(\xi)\\y(\xi)\end{Bmatrix}=\sum_{i=0}^{\hat{n}}R_{i,2p}\begin{Bmatrix}X_i\\Y_i\end{Bmatrix} \tag{2.15}$$

where

$$R_{i,2p}=\frac{N_{i,2p}(\xi)W_i}{\sum_{i=0}^{\hat{n}}N_{i,2p}(\xi)W_i} \tag{2.16}$$

in which (X_i, Y_i, W_i) are the coordinates and weights of the equivalent higher order NURBS curve, which can be obtained using the algorithm described in Lemma 2, and $\hat{n}+1$ is the number of control points.

End of proof

In the special case of Rational Bézier (R-Bézier) curves, one can obtain straightforward analytical expressions for the coefficients of the equivalent higher order R-Bézier curve in Eq. (2.15). For this case, Eqs. (2.15) and (2.16) can be written as

$$\begin{Bmatrix}x(\xi)\\y(\xi)\end{Bmatrix}=\sum_{i=0}^{2p}R_{i,2p}(\xi)\begin{Bmatrix}X_i\\Y_i\end{Bmatrix} \tag{2.17}$$

where

$$R_{i,2p} = \frac{B_{i,2p}(\xi) W_i}{\sum_{i=0}^{\hat{n}} B_{i,2p}(\xi) W_i} \tag{2.18}$$

Using relations (2.10) and (2.11) in Lemma 1, the weights and control points in these equations are obtained as

$$\begin{aligned} W_i &= \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} w_j^x w_{i-j}^y \\ X_i &= \frac{1}{W_i} \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} x_j w_j^x w_{i-j}^y \\ Y_i &= \frac{1}{W_i} \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} y_j w_j^y w_{i-j}^x \end{aligned} \tag{2.19}$$

where $\lambda_{ij} = \dfrac{\binom{n}{j}\binom{n}{i-j}}{\binom{2n}{i}}$.

Figure 2.4 shows a quadratic GNURBS curve, and its equivalent quartic NURBS curve obtained using the above theorem.

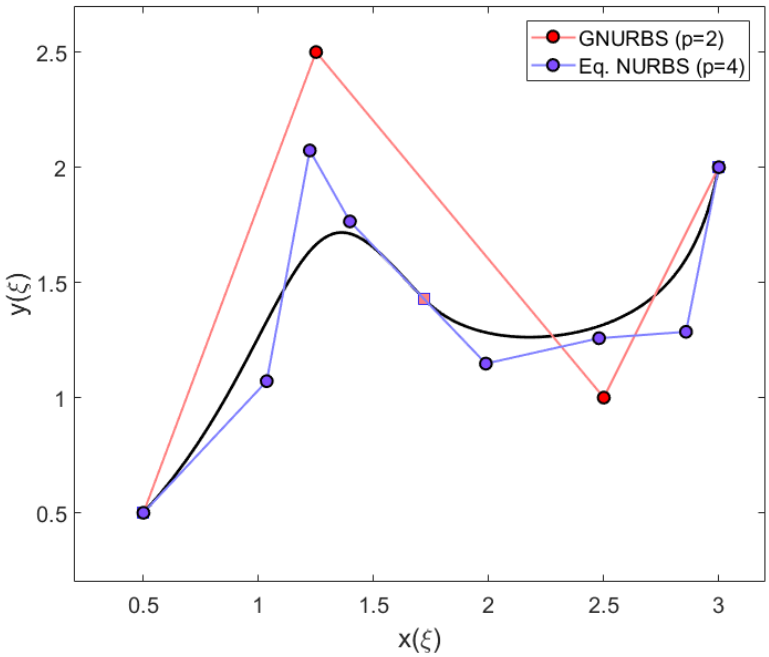


Figure 2.4. Equivalence of a 2D quadratic GNRUBS curve with $\{w^x_0, \ldots, w^x_3\}=\{1, 2.5, 1.5, 3\}$ and $\{w^y_0, \ldots, w^y_3\}=\{1, 1, 2.5, 2\}$, with a quartic NURBS curve with $\{w_0, \ldots, w_7\}=\{1.00, 1.75, 2.30, 3.19, 3.81, 4.04, 5.25, 6.00\}$.

It needs to be pointed out that, despite the apparent violation of some properties of NURBS, the above theorem establishes that GNURBS are merely disguised form of higher order classic NURBS, thereby inheriting all the properties of NURBS indirectly. For instance, as can be seen in Figure 2.4, the curve violates the global convex-hull of the original control polygon of GNURBS, however, it does lie within the convex-hull of the control polygon associated with its equivalent higher order classic NURBS.

2.1.2. Partial decoupling for 3D curves

One can easily extend the above theorem and formulation to 3D curves with independent weights along all three physical directions. However, a more practical case, which will be the emphasis for the rest of this chapter, is to perform partial decoupling of the weights. In particular, in 3D, one can use the same set of weights in *x* and *y* directions, denoted by w^{xy}, and a different set of weights in *z* direction w^{z}. Accordingly, Eq. (2.5) could be written as

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{n} \begin{Bmatrix} R_{i,p}^{xy}(\xi)\, x_i \\ R_{i,p}^{xy}(\xi)\, y_i \\ R_{i,p}^{z}(\xi)\, z_i \end{Bmatrix} \tag{2.20}$$

where

$$R_{i,p}^{xy}(\xi) = \frac{N_{i,p}(\xi)\, w_i^{xy}}{\sum_{j=0}^{n} N_{j,p}(\xi)\, w_j^{xy}} \tag{2.21}$$

Observe that owing to this decoupling of the in-plane and out-of-plane weights, unlike in classic NURBS, one can now freely manipulate the weights along *z* direction, for instance, without perturbing the geometry or parameterization of the underlying curve in *x*-*y* plane. For better insight, we provide a graphical visualization of designing a 3D curve with an in-plane circular shape in Figure 2.5.

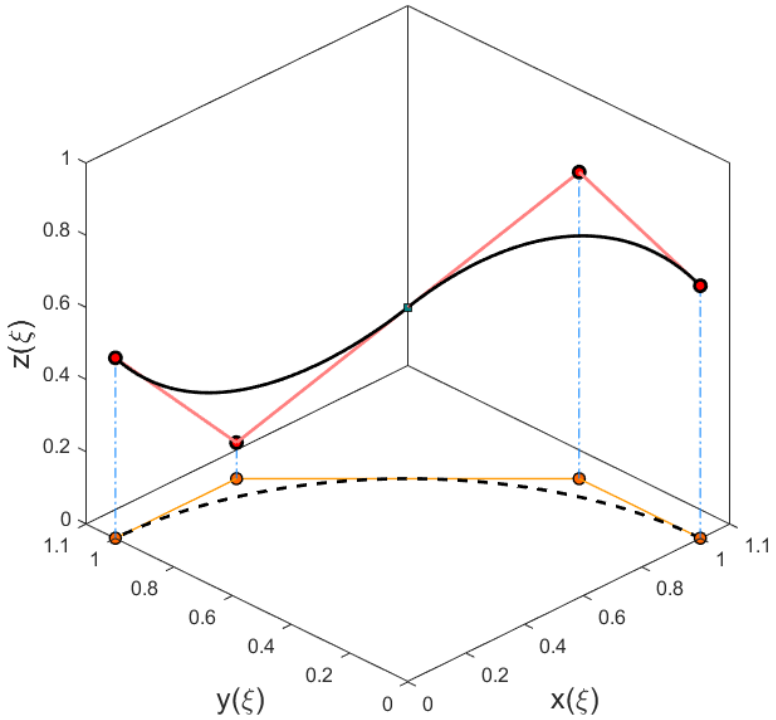


Figure 2.5. A 3D GNURBS curve with an underlying precise circular arc: $\{w^{xy}_0, \dots, w^{xy}_3\}=\{1, 0.8536, 0.8536, 1\}$ and $\{w^{z}_0, \dots, w^{z}_3\}=\{1, 1, 1, 1\}$.

As can be clearly seen in Figure 2.5, treating the independent set of out of plane weights can provide better flexibility and control. As a simple example, one can use this representation as an intermediate interactive shape design tool, and finally convert it to a higher order classic NURBS, if desired, to recover affine invariance and other properties. In this study, we will focus on demonstrating superior approximation abilities of this representation in certain applications where a height function, field or set of data points need to be approximated over an underlying 2D curve.

To derive the equivalent higher order NURBS representation of (2.20), we rewrite this equation in the following form

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \frac{\sum_{i=0}^{n} N_{i,p}(\xi) w_i^{xy}}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^{xy}} \begin{Bmatrix} x_i \\ y_i \\ 0 \end{Bmatrix} + \frac{\sum_{i=0}^{n} N_{i,p}(\xi) w_i^{z}}{\sum_{j=0}^{n} N_{j,p}(\xi) w_j^{z}} \begin{Bmatrix} 0 \\ 0 \\ z_i \end{Bmatrix} \tag{2.22}$$

Following a very similar procedure as for 2D curves, we can easily derive the expressions for the equivalent higher order NURBS curve to the generalized form in Eq. (2.20) as

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{\hat{n}} R_{i,2p} \begin{Bmatrix} X_i \\ Y_i \\ Z_i \end{Bmatrix} \tag{2.23}$$

where

$$R_{i,2p} = \frac{N_{i,2p}(\xi) W_i}{\sum_{i=0}^{\hat{n}} N_{i,2p}(\xi) W_i} \tag{2.24}$$

(X_i, Y_i, Z_i, W_i) in these equations can be obtained using a similar algorithm as for 2D curves in the following form

$$W_i = \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} w_j^{xy} w_{i-j}^{z} \tag{2.25}$$

and

$$\begin{aligned} X_i &= \frac{1}{W_i} \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} x_j w_j^{xy} w_{i-j}^{z} \\ Y_i &= \frac{1}{W_i} \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} y_j w_j^{xy} w_{i-j}^{z} \\ Z_i &= \frac{1}{W_i} \sum_{j=\max(0,i-n)}^{\min(n,i)} \lambda_{ij} z_j w_j^{z} w_{i-j}^{xy} \end{aligned} \tag{2.26}$$

where $\lambda_{ij} = \dfrac{\binom{n}{j}\binom{n}{i-j}}{\binom{2n}{i}}$.

It should be noted here that the properties of classic NURBS which are lost in this proposed generalization are not critical or even of interest in many applications of NURBS. Nevertheless, in some applications, these properties can be crucial. In order to make GNURBS applicable to such applications, we develop an alternative variation of NURBS which can be directly derived from the generalization proposed above.

2.2. Generalized NURBS curves: isoparametric form via implicit decoupling of the weights

Note that the equivalent higher order NURBS representation in Eq. (2.15) or (2.23) itself provides another variation of NURBS which can be directly employed as another alternative to NURBS with better flexibility in some applications.

In order to clarify how these equations provide additional flexibility than classic NURBS, we first derive a more generic form of these equations via an alternative approach using an extension of order elevation technique.

Assume a 2D R-Bézier curve of degree *p* is given as follows

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \end{Bmatrix} = \frac{\sum_{i=0}^{p} B_{i,p}(\xi) w_i^{xy}}{\sum_{j=0}^{p} B_{j,p}(\xi) w_j^{xy}} \begin{Bmatrix} x_i \\ y_i \end{Bmatrix} \tag{2.27}$$

In order to elevate the degree of this curve by *q*, we can simply multiply both numerator and denominator of this equation by any arbitrary expression in the following form

$$f(\xi) = \sum_{i=0}^{q} B_{i,q}(\xi) w_i^z \tag{2.28}$$

Recalling Lemma 1, we can obtain the higher order R-Bézier curve with *q* degree elevations as

$$\begin{Bmatrix} x(\xi) \\ y(\xi) \end{Bmatrix} = \sum_{i=0}^{\hat{n}} R_{i,p+q} \begin{Bmatrix} X_i \\ Y_i \end{Bmatrix} \tag{2.29}$$

where

$$R_{i,p+q} = \frac{B_{i,p+q}(\xi) W_i}{\sum_{i=0}^{\hat{n}} B_{i,p+q}(\xi) W_i} \tag{2.30}$$

in which $\hat{n} = p + q$ and (X_i, Y_i, W_i) can be obtained using Eqs. (2.31) and (2.32)

$$W_i = \sum_{j=\max(0,i-q)}^{\min(p,i)} \lambda_{ij} w_j^{xy} w_{i-j}^z \tag{2.31}$$

$$X_i = \frac{1}{W_i} \sum_{j=\max(0,i-q)}^{\min(p,i)} \lambda_{ij} x_j w_j^{xy} w_{i-j}^{z}$$
$$Y_i = \frac{1}{W_i} \sum_{j=\max(0,i-q)}^{\min(p,i)} \lambda_{ij} y_j w_j^{xy} w_{i-j}^{z} \tag{2.32}$$

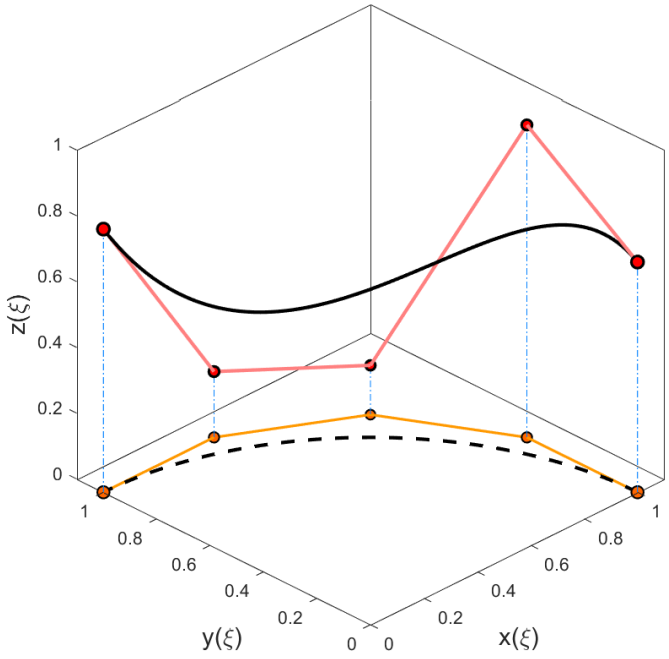
where $\lambda_{ij} = \dfrac{\binom{p}{j}\binom{q}{i-j}}{\binom{p+q}{i}}$.

Observe that this procedure can be seen as a trivial extension of the classic order elevation techniques in the literature [19,66]. In fact, one can simply recover the common order elevation algorithm by assigning $w_i^z = 1, \forall i$ in Eq. (2.28). We will refer to this procedure as *generalized order elevation* hereafter. Now suppose we intend to add another dimension to the representation in Eq. (2.29) in an isoparametric manner. Again, this extra dimension can be viewed as the height function of a parametric curve in 2D, or may represent a field or set of data points which needs to be approximated over a 2D curve. For this purpose, we extend Eq. (2.29) as

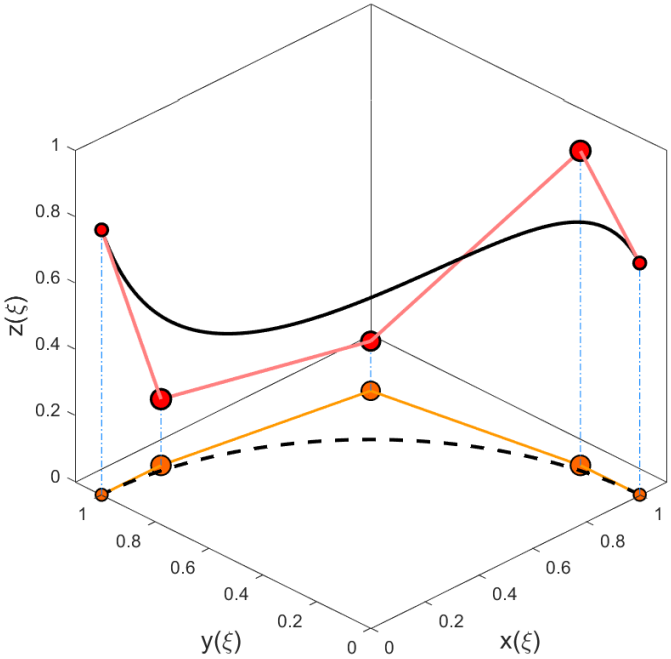
$$\begin{Bmatrix} x(\xi) \\ y(\xi) \\ z(\xi) \end{Bmatrix} = \sum_{i=0}^{\hat{n}} R_{i,p+q} \begin{Bmatrix} X_i \\ Y_i \\ Z_i \end{Bmatrix} \tag{2.33}$$

It is interesting to notice that, although Eq. (2.33) apparently seems to be a classic R-Bézier curve, it provides additional flexibility. Observe that in the above procedure, w_i^z are arbitrary variables which can be freely chosen without perturbing the geometry or parameterization of the underlying curve in *x-y* plane.

In order to better demonstrate the effect of these weights on the behavior of GNURBS curves, we generate a 3D quartic GR-Bézier curve by performing the above process with $q = 2$ on a quadratic R-Bézier circular arc and assigning the heights of control points as shown in Figure 2.6.



(a)



(b)

Figure 2.6. A 3D isoparametric GNURBS curve with (a) $\{w^z_0, w^z_1, w^z_2\}=\{1, 1, 1\}$, and (b) $\{w^z_0, w^z_1, w^z_2\}=\{1, 2.5, 1\}$.

The obtained results with $\left\{w_1^z, w_2^z, w_3^z\right\} = \{1,1,1\}$ (classic order elevation) and $\left\{w_1^z, w_2^z, w_3^z\right\} = \{1, 2.5, 1\}$ are represented in Figure 2.6(a) and (b), respectively. As observed, the heights of control

points in both cases are identical. For more clarity, the size of control points is plotted proportional to their weights. Further, the corresponding sets of basis functions are plotted in Figure 2.7.

Comparing Figure 2.6(a) and (b), it can be noticed that by increasing w_2^z, the weights of the three interior control points are increased which results in out of plane deformation of the curve as depicted in Figure 2.6(b). However, as this figure shows, this leads to automatic in-plane re-arrangement of control points in such a manner that the in-plane geometry of the curve (as well as its parameterization) remains unchanged.

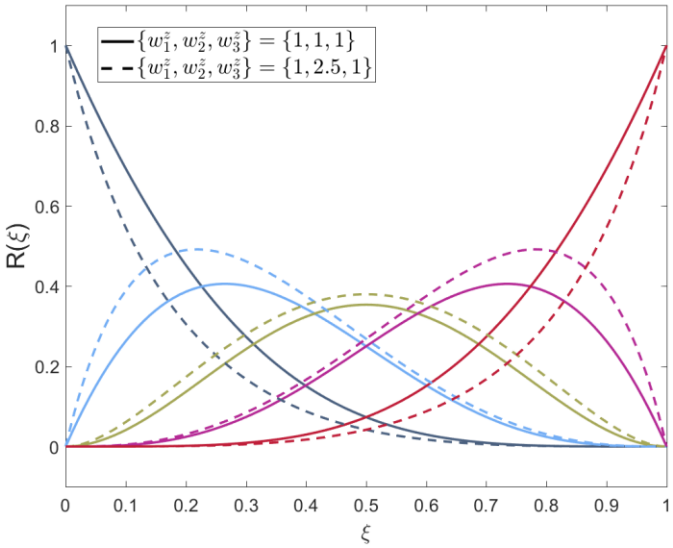


Figure 2.7. The function spaces corresponding to GNURBS curves in Figure 2.6.

The above algorithm can be extended to NURBS in a straightforward manner using a similar three step algorithm explained in Lemma 2. That is, Eq. (2.33) also holds true for NURBS with the rational basis functions defined as

$$R_{i,p+q} = \frac{N_{i,p+q}(\xi) W_i}{\sum_{i=0}^{\hat{n}} N_{i,p+q}(\xi) W_i} \tag{2.34}$$

We here note that while the variables w_i^z in Eq. (2.33) or (2.34) can be directly treated as design variables for improved flexibility, the physical meaning and local support of the weights in this variation are lost. Hence, it might not be suitable for being used as an interactive shape design tool. However, as will be shown in the next section, it can still be effectively employed as an enhanced tool for approximation purposes where the decision on the optimal values of the weights is made by a numerical algorithm.

2.3. Applications

The proposed generalizations of NURBS in Eqs. (2.20) and (2.33) provide alternative tools to NURBS which can be useful in certain applications such as IGA. In this section, we investigate function approximations as a simple application in the context of CAGD. Hereafter, we will persistently refer to Eq. (2.20) as the first generalization of NURBS or non-isoparametric GNURBS, while we will refer to Eq. (2.33) as the second generalization of NURBS or isoparametric GNURBS.

Both these variations primarily provide the common and significant possibility of treating the out-of-plane weights as additional design variables, without perturbing the underlying geometry or its parameterization. However, the difference between them should be clear since the first form is obtained via explicit decoupling of the weights along different physical coordinates resulting in a non-isoparametric representation with the properties elaborated in Section 2.1.1, while the second variation is obtained by implicit decoupling of the weights within the isoparametric set of basis functions; thereby preserving the properties of NURBS. As discussed above, the generation of these implicitly decoupled set of weights in the second variation requires order elevation a priori.

Finally, we emphasize that although these new representations finally lie in the NURBS space, obtaining their results in certain class of applications by directly making use of NURBS does not seem possible.

2.3.1. Approximation over curved domains

There are various applications where the data or a function needs to be approximated over a parametric curved domain. For instance, there are numerous studies in the literature for the approximation of scattered data or functions on curved surfaces; see [67,68] for a rigorous review. A similar problem arises in other applications such as modelling helical curves and surfaces [69–71], treating the non-homogenous essential boundary conditions in IGA [72–75] etc. In all these applications the limitation of preserving the underlying parameterization applies. Therefore, employing the weights as additional design variables is disallowed. In this section, we investigate the performance of GNURBS versus NURBS in this class of problems for two cases of approximating a smooth function as well as a rapidly varying one.

2.3.2. Least-square minimization using NURBS and GNURBS

Suppose an in-plane circular arc is given in the following parametric form

$$\mathbf{C}(\xi) = r\left(\cos(\frac{\pi}{2}\xi), \sin(\frac{\pi}{2}\xi)\right) \qquad 0 \le \xi \le 1 \tag{2.35}$$

where r is the radius of the circular arc. Eq. (2.35) can be precisely constructed using NURBS. Now, assume a height function $z(\xi)$ needs to be approximated over this arc with minimum error. This can be easily posed as a least-square approximation problem leading to optimal accuracy in *L_2-norm*. Assuming $\{\xi_s \to (\overline{x}_s, \overline{y}_s, \overline{z}_s) : s \in \mathcal{S}\}$ is the set of *ns* collocation points, the error function f to be minimized is defined as

$$f = \frac{1}{2}\sum_{s\in\mathcal{S}} \left\|\hat{z}(\xi_s) - \overline{z}_s\right\|^2 = \frac{1}{2}\sum_{s\in\mathcal{S}} \left\| \sum_{L\in\mathcal{L}^s} R_L(\xi_s) z_L - \overline{z}_s \right\|^2 \tag{2.36}$$

where $\hat{z}(\xi)$ is the approximated NURBS function, ξ_s are the corresponding collocation points in the parametric space, $\mathcal{L}^s$ is the set of indices of non-zero basis functions at ξ_s and $\overline{z}_s = z(\xi_s)$.

In the case of NURBS, the only unknowns to consider are *control variables* z_L and the problem leads to a linear least square problem in the following matrix form

$$\sum_{s\in\mathcal{S}} \begin{pmatrix} R_0(\xi_s)R_0(\xi_s) & \cdots & R_0(\xi_s)R_n(\xi_s) \\ \vdots & \ddots & \vdots \\ R_n(\xi_s)R_0(\xi_s) & \cdots & R_n(\xi_s)R_n(\xi_s) \end{pmatrix} \begin{pmatrix} z_0 \\ \vdots \\ z_n \end{pmatrix} = \sum_{s\in\mathcal{S}} z(\xi_s) \begin{pmatrix} R_0(\xi_s) \\ \vdots \\ R_n(\xi_s) \end{pmatrix} \tag{2.37}$$

which can be solved for the $n+1$ unknowns $\boldsymbol{\lambda} = \{z_0, ..., z_n\}$ by proper choice of collocation points.

To improve the accuracy of approximation, invoking the proposed variations of NURBS, we can treat the out-of-plane weights w_i^z as extra design variables without perturbing the geometry or parameterization of the underlying precise circular arc. We may refer to these variables as *control weights* hereafter. With the first generalization in Eq. (2.20), the vector of design variables becomes $\boldsymbol{\lambda} = \{z_0, ..., z_n, w_0^z, ..., w_n^z\}$, where the positivity constraints on control weights ($w_i^z > 0,\ \forall i$) are often desired to be satisfied for numerical stability. Considering the new set of design variables, Eq. (2.37) now becomes a non-linear least-square problem which can be solved using any of the existing solvers such as Levenberg-Marquardt.

To avoid solving a non-linear problem, one can alternatively employ a two-step algorithm developed by Ma [11,21], which leads to two separate linear systems of equations; a homogenous system which yields the optimal control weights and a non-homogenous one that yields the corresponding optimal control variables. The development of this algorithm for GNURBS is provided next.

Employing the concept of homogeneous coordinates, the third component of GNURBS curve in Eq. (2.20) can be written in the following matrix form

$$z(\xi) = \frac{\mathbf{N}^T(\xi)\mathbf{z}^w}{\mathbf{N}^T(\xi)\mathbf{w}^z} \tag{2.38}$$

where the vector variables are defined as

$$\begin{aligned} &\mathbf{N} = [N_0(\xi), N_1(\xi), ..., N_n(\xi)]^T \\ &\mathbf{z}^w = [z_0^w, z_1^w, ..., z_n^w]^T = [z_0 w_0^z, z_1 w_1^z, ..., z_n w_n^z]^T \\ &\mathbf{w}^z = [w_0^z, w_1^z, ..., w_n^z]^T \end{aligned} \tag{2.39}$$

We may refer to z_i^w in this equation as weighted control variables. Also, we have dropped the subscript *p* in denoting the B-spline basis functions, for brevity. Eq. (2.38) can be written at the collocation points in the following form

$$\mathbf{N}^T(\xi_s)\mathbf{z}^w = z(\xi_s)\mathbf{N}^T(\xi_s)\mathbf{w}^z \qquad \forall s \in \mathcal{S} \tag{2.40}$$

Denoting the set of data points and B-spline basis functions in the matrix forms of Eqs. (2.41) and (2.42), respectively

$$\bar{Z} = diag\{\bar{z}_1, ..., \bar{z}_{ns}\} \tag{2.41}$$

$$\tilde{N} = \begin{bmatrix} N_0(\xi_1) & N_1(\xi_1) & \cdots & N_n(\xi_1) \\ N_0(\xi_2) & N_1(\xi_2) & \cdots & N_n(\xi_2) \\ \vdots & \vdots & & \vdots \\ N_0(\xi_{ns}) & N_1(\xi_{ns}) & \cdots & N_n(\xi_{ns}) \end{bmatrix}_{ns\times(n+1)} \tag{2.42}$$

Eq. (2.40) can be written in the following compact form

$$\tilde{N}\,\mathbf{z}^w = \bar{Z}\,\tilde{N}\,\mathbf{w}^z \tag{2.43}$$

which can be re-written as

$$\tilde{\mathrm{A}} \begin{bmatrix} \mathbf{z}^w \\ \mathbf{w}^z \end{bmatrix} = [0]_{2n\times 1} \tag{2.44}$$

where

$$\tilde{\mathrm{A}} = \begin{bmatrix} \tilde{N} & -\bar{Z}\,\tilde{N} \end{bmatrix}_{ns\times 2n} \tag{2.45}$$

Eq. (2.44) is an over-determined system of equations and now represents a linear least-square problem. Multiplying the sides of this equation by $\tilde{\mathrm{A}}^T$ yields

$$\begin{bmatrix} \tilde{N}^T\tilde{N} & -\tilde{N}^T\bar{Z}\,\tilde{N} \\ -\tilde{N}^T\bar{Z}\,\tilde{N} & \tilde{N}^T\bar{Z}^2\tilde{N} \end{bmatrix} \begin{bmatrix} \mathbf{z}^w \\ \mathbf{w}^z \end{bmatrix} = [0]_{2n\times 1} \tag{2.46}$$

It is possible to separate the control weights from the control variables by eliminating the lower left element of Eq. (2.46), which yields

$$\begin{bmatrix} \tilde{N}^T\tilde{N} & -\tilde{N}^T\bar{Z}\,\tilde{N} \\ 0 & M \end{bmatrix} \begin{bmatrix} \mathbf{z}^w \\ \mathbf{w}^z \end{bmatrix} = [0]_{2n\times 1} \tag{2.47}$$

where

$$M = \tilde{N}^T\bar{Z}^2\tilde{N} - (\tilde{N}^T\bar{Z}\,\tilde{N})(\tilde{N}^T\tilde{N})^{-1}(\tilde{N}^T\bar{Z}\,\tilde{N}) \tag{2.48}$$

According to Eq. (2.47), the control weights are now decoupled from the control variables and can be obtained via solving the following homogeneous system of equations

$$M\,\mathbf{w}^z = [0]_{n\times 1} \tag{2.49}$$

Further details on different algorithms for solving Eq. (2.49) and extracting the optimal real or positive weights can be found in [21]. Once the unknown weights are found, the optimal control variables can be subsequently obtained via solving Eq. (2.43).

With the second generalization in Eq. (2.33), however, the development of a linear algorithm does not seem easily possible. Therefore, a non-linear least square algorithm needs to be used to find the optimal set of design variables. Further, since the derivation of analytical Jacobian matrix becomes complicated in case of having internal knots, we limit our study to GR-Bézier. The vector of design variables for this simplified case becomes $\boldsymbol{\lambda} = \left\{ Z_0, ..., Z_n, w_0^z, ..., w_q^z \right\}$ where $n = p + q$.

The imposition of the least square problem is quite straightforward; hence, we do not present it here. The derivation of Jacobian matrix components with respect to control weights, however, is non-trivial and requires evaluating the sensitivities using the following expressions

$$\frac{\partial W_i}{\partial w_k^z} = \begin{cases} \dfrac{\binom{p}{i-k}\binom{q}{k}}{\binom{p+q}{i}} w_{i-k}^{xy}, & if \quad (i-p) \le k \le i \\ 0 & Otherwise \end{cases} \tag{2.50}$$

The initial conditions for solving the least square problem are specified as follows

$$\boldsymbol{\lambda}_0 = \left\{ \underbrace{0,0,...,0}_{n+1}, \underbrace{1,1,...,1}_{q+1} \right\} \tag{2.51}$$

As previously discussed, by changing w_i^z during the optimization process, the in-plane coordinates of control points also vary at each iteration. However, since the in-plane geometry and parameterization are always fixed, one may only re-evaluate and update these coordinates after the termination of the optimization process according to the obtained optimal set of isoparametric basis functions. It is important to note that *this algorithm yields the combination of optimal weights and the corresponding arrangement of control points which results in the best approximation over a given parameterization*. To our knowledge, no such investigation has been reported in the literature thus far.

In the next section, we approximate various height functions over the circular arc in Eq. (2.35) modelled precisely with NURBS. In all cases, the interpolating end control points are prescribed to lie on the height function. Further, we employ 100 uniformly distributed sample points in the parametric space for setting up the least square problem. The numerical implementations are performed in MATLAB. Finally, the relative *L_2-norms* of the error are calculated using the following relation

$$error = \frac{\left(\int \left(\hat{z}(\xi) - z(\xi) \right)^2 d\Gamma \right)^{1/2}}{\int z(\xi) d\Gamma} \tag{2.52}$$

where the numerical integrations are calculated using Gaussian quadrature.

2.3.3. A smooth function: helix modelling

As the first numerical example, we consider approximating a smooth height function as

$$z(\xi) = b\varphi = b\left(\frac{\pi}{2}\xi\right) \tag{2.53}$$

over the parametric curve in Eq. (2.35). In the above equation, φ is the center angle of the circular arc in *x*-*y* plane and *b* is a constant. Eq. (2.53) together with (2.35) represent a segment of a helical curve, shown in Figure 2.8 for $b = 1$, and is a classic problem in geometric modelling. We here demonstrate how the proposed variations of NURBS can be useful for improved modelling of such type of problems.

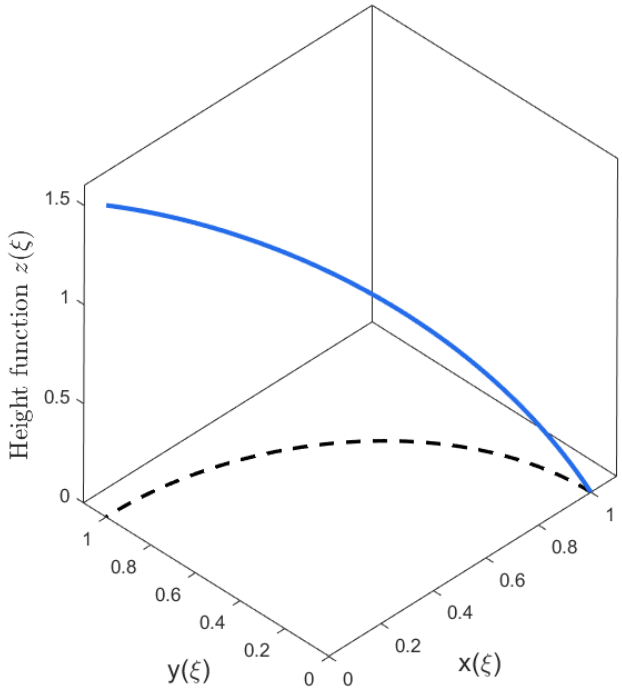


Figure 2.8. A smooth helical curve.

Helical curves and surfaces do not have an exact representation in terms of polynomials or rational polynomials [76]. A high accuracy of approximation by NURBS using the minimal number of control points is of interest, and will make the helix more convenient to use in current CAD/CAM systems [70]. There is a large number of studies in the literature addressing this problem using R-Bézier, NURBS or other parametric representations; see e.g. [69–71,77] for a review of these studies. Having examined these studies, it can be found that there are several considerations for a suitable approximation of helix such as the accuracy of normal angle, curvature, torsion and height, besides meeting certain geometric conditions at the end points of each segment [70]. However, we only focus here on approximating the height function with maximum accuracy, for simplicity. Further, it is desirable that the fitting curve precisely lies on the cylinder surface of the helix [71].

Since this is a geometric modelling problem, the properties of NURBS are important to be preserved for this particular application. Therefore, it is an ideal candidate for employing the second variation, i.e. isoparametric GR-Bézier, as the obtained optimal design is directly in the NURBS space. The obtained results using the above-discussed algorithm for different degrees of basis functions are presented in Table 2.1 for comparison.

Table 2.1. Error of approximating the helix height function using R-Bézier versus GR-Bézier in relative L^2-norm.

Curve type	Degree $(n = p + q)$	No. of control variables	No. of control weights	Error	Error ratio
R-Bézier	2	3	0	2.41E-2	1.0
2nd GR-Bézier			0	2.41E-2	
R-Bézier	3	4	0	1.50E-4	1.0
2nd GR-Bézier			2	1.50E-4	
R-Bézier	4	5	0	1.50E-4	121.9
2nd GR-Bézier			3	1.23E-6	
R-Bézier	5	6	0	2.30E-6	209.1
2nd GR-Bézier			4	1.10E-8	

As the table shows, the accuracy of approximation by GR-Bézier over R-Bézier increasingly improves by elevating the degree, as a larger number of control weights are added to the design space. In case of $p = 3$, however, no improvement in the accuracy is gained. This implies that the optimal values of the control weights for this case are equal to 1; that is, cubic R-Bézier obtained via order elevation is coincidentally optimal for the approximation of this height function.

The initial and optimal sets of basis functions for approximation with different degrees are represented in Figure 2.9. As can be observed in this figure, in both cases, the optimal sets of basis functions are only slightly different than the initial ones; however, this small deviation results in dramatic improvement of the accuracy of approximation as reported in Table 2.1.

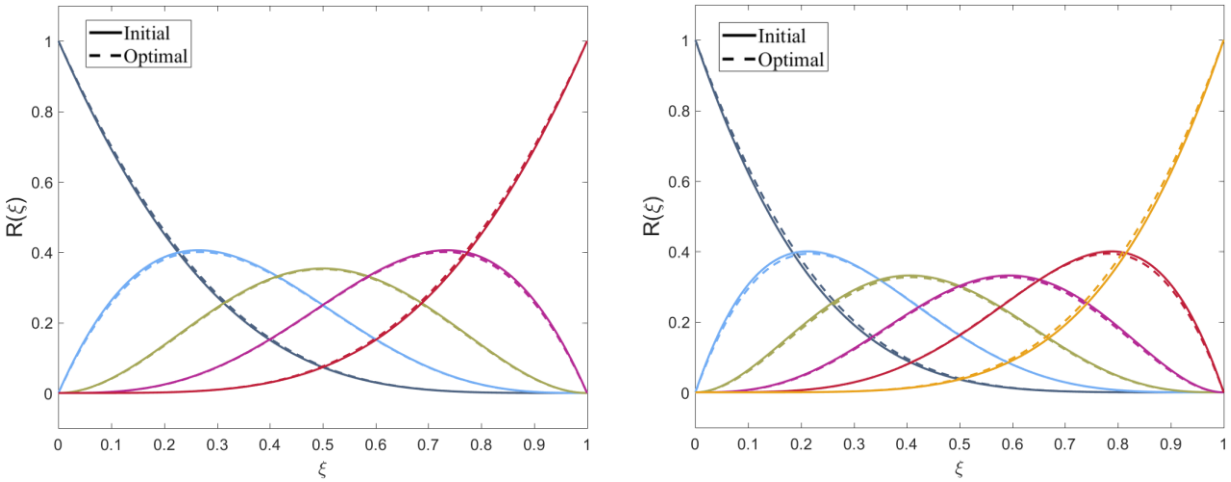
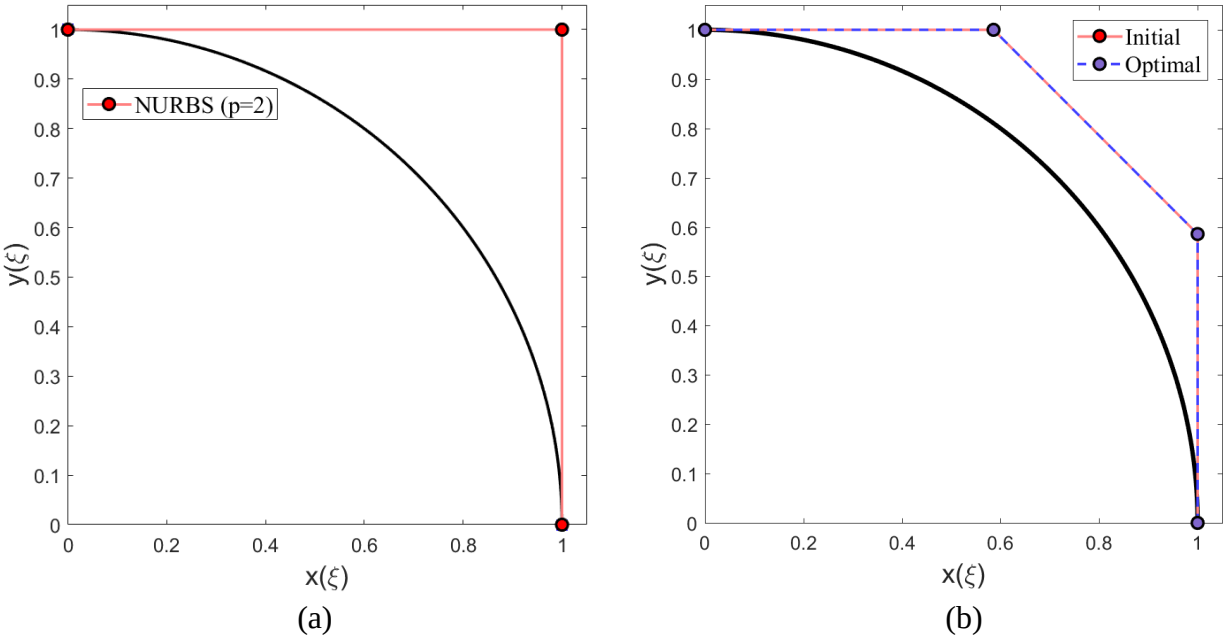


Figure 2.9. Initial and optimal basis functions for approximating the helix height function using 2nd GR-Bézier with degree (a) $n = 4$ and (b) $n = 5$.

We remind that in the case of isoparametric generalization (2nd GR-Bézier), the basis functions are identical along all physical coordinates. As previously explained, this leads to automatic re-arrangement of the in-plane coordinates of control points, depicted in Figure 2.10, in such a manner that the in-plane geometry and its parameterization remain unchanged.



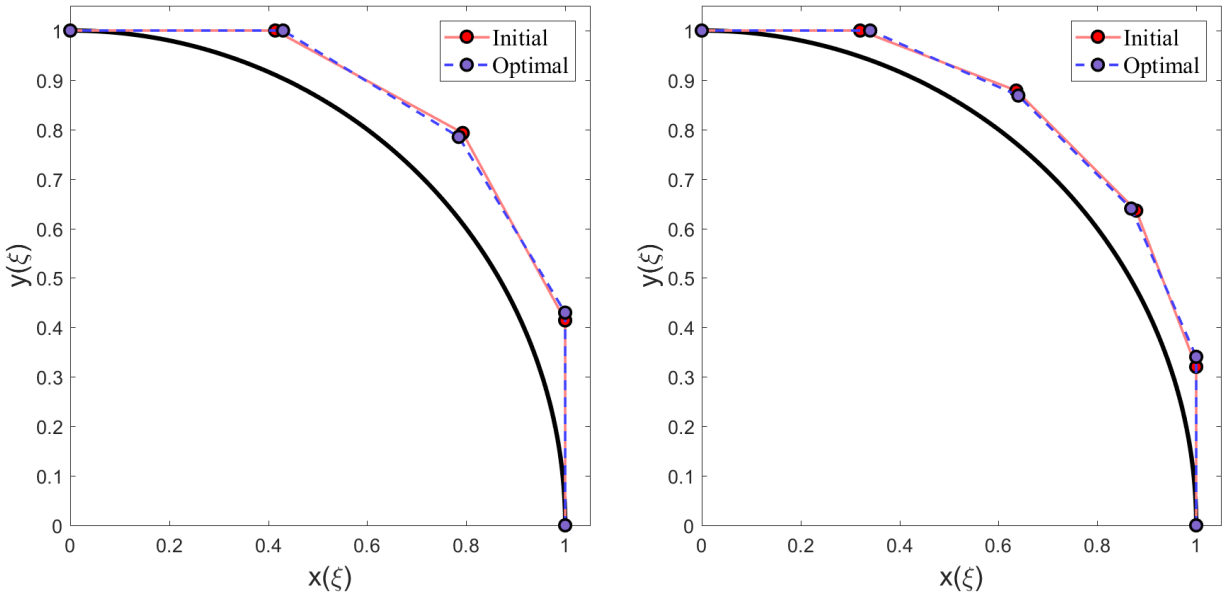


Figure 2.10. Initial and optimal control polygons for approximating the helix height function with (a) R-Bézier of degree $n = 2$, and 2nd GR-Bézier of degree (b) $n = 3$ (c) $n = 4$ and (d) $n = 5$.

We also investigate the performance of GNURBS compared to NURBS with respect to refining the knot sequence. For this experiment, we use the first variation (non-isoparametric), for simplicity and as it provides better flexibility.

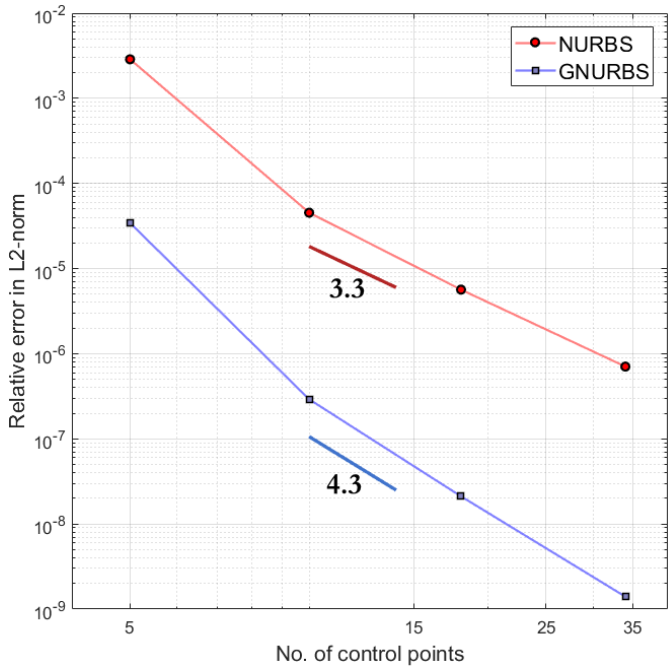


Figure 2.11. Convergence rate of 1st GNURBS versus NURBS for approximating the helix height function.

The obtained results for $p=2$ are represented in Figure 2.11. As the figure shows, by including the control weights to the design space, the convergence rate is improved from 3.3 to 4.3, resulting in dramatic improvement in the accuracy especially when larger numbers of control points are employed. However, as previously mentioned, in the case of GNURBS there is an extra computational cost for obtaining the optimal weights via solving an additional homogenous system of equations.

2.3.4. A rapidly varying function

As the second example, we investigate the performance of the proposed variations of NURBS in capturing rapidly varying functions. We consider the problem of approximating a rapidly varying function as in Eq. (2.54) over the same circular arc

$$z(\xi)=\varphi\left(1+e^{-\alpha(\varphi-0.5)^2}+e^{-\alpha(\varphi-0.8)^2}\right),\ \varphi=(\frac{\pi}{2})\xi \tag{2.54}$$

which is plotted in Figure 2.12 for $\alpha=20$.

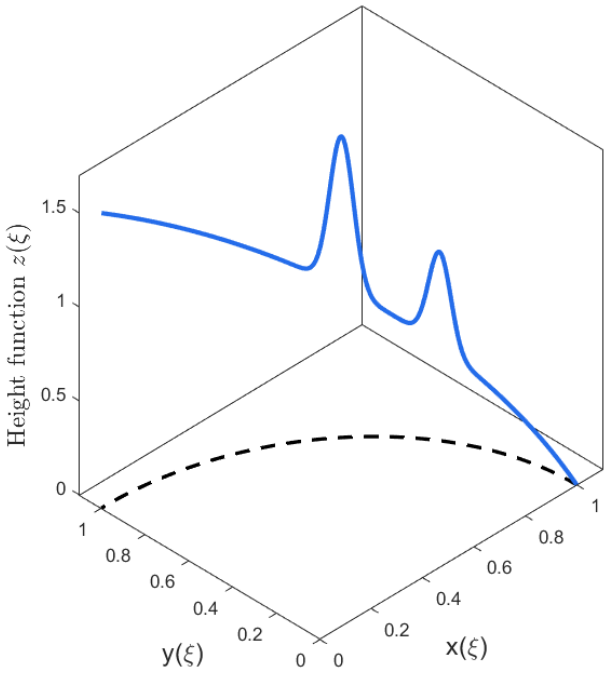


Figure 2.12. A rapidly varying function over a circular arc.

Employing the first proposed variation of NURBS, we approximate the height function using different degrees of basis functions. The obtained results are presented in Table 2.2. All these

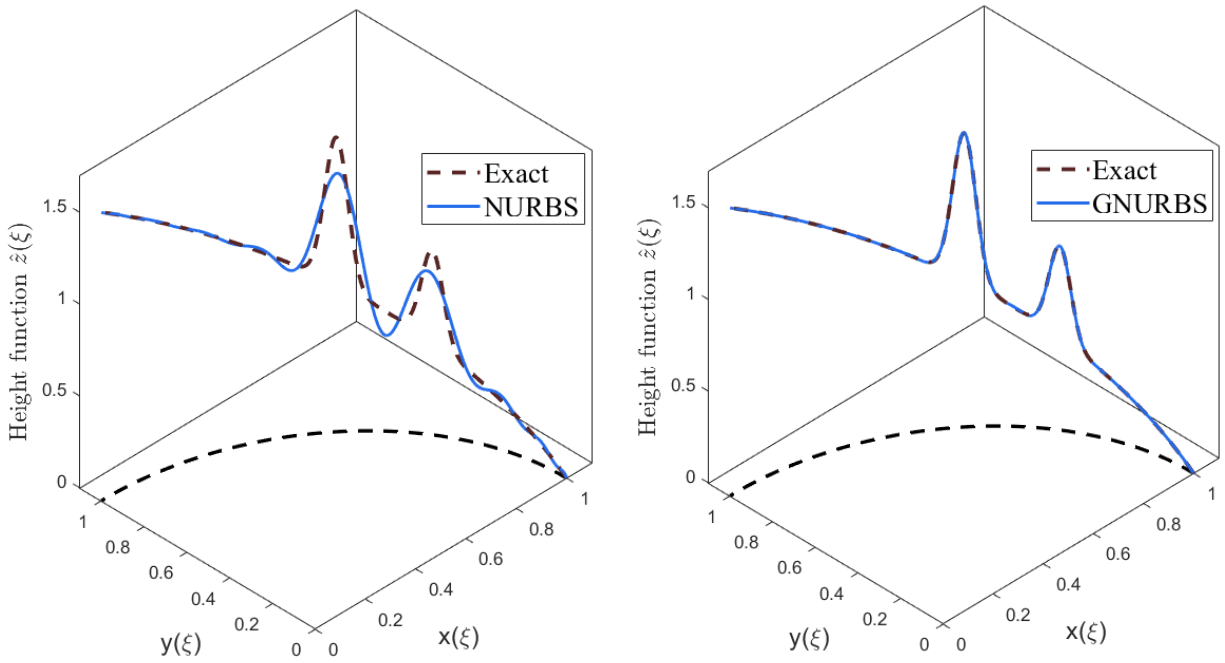
models are obtained by performing uniform knot insertion over an initial R-Bézier arc and therefore possess maximal continuity.

Table 2.2. Error of approximating the rapidly varying function in Eq. (2.54) using NURBS versus 1st GNURBS in relative L^2-norm.

Curve type	Degree (p)	No. of control variables	No. of control weights	Error	Error ratio
NURBS	2	18	0	6.86E-2	9.23
1st GNURBS			18	7.43E-3	
NURBS	3	19	0	5.35E-2	9.80
1st GNURBS			19	5.46E-3	
NURBS	4	20	0	6.27E-2	14.31
1st GNURBS			20	4.38E-3	
NURBS	5	21	0	5.48E-2	40.60
1st GNURBS			21	1.35E-3	

According to the table, the accuracy of approximation using NURBS does not change noticeably by elevating the degree. On the other hand, the obtained results with GNURBS persistently improve by elevating the degree, which reveals the superiority of approximation of GNURBS over NURBS in capturing rapidly varying fields.

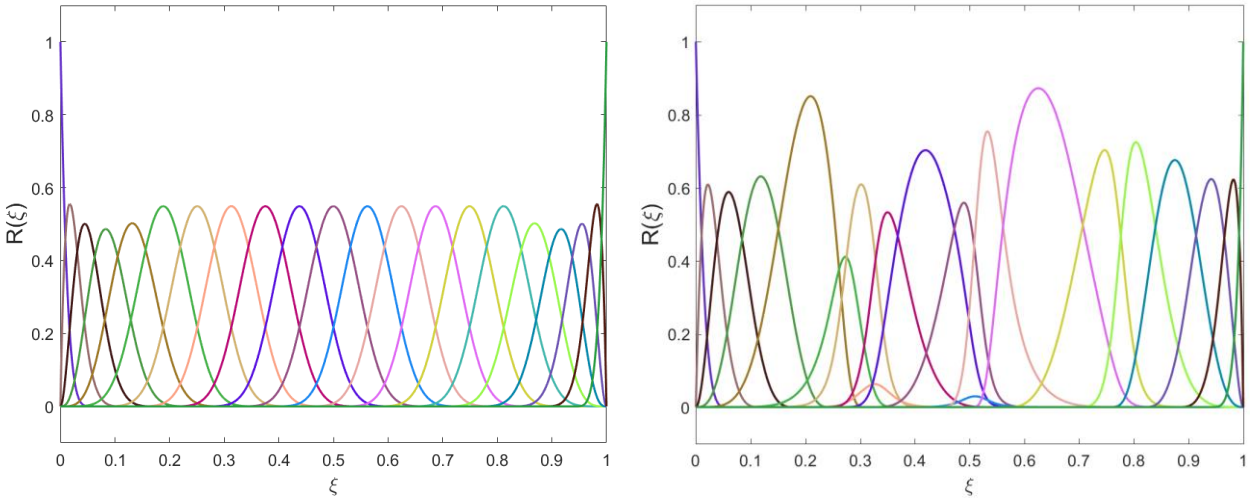
The approximation results for $p=5$ are plotted in Figure 2.13. The figure clearly shows the improvement of approximation in the case of GNURBS especially in the vicinity of existing sharp transitions in the field.



(a) (b)

Figure 2.13. Approximation of the rapidly varying function with quintic (a) NURBS and (b) 1st GNURBS.

Further, the corresponding basis functions are represented in Figure 2.14. It is interesting to note that, unlike the previous case of approximating a smooth function, there is a significant change between the initial and optimal basis functions here. As can be seen, this difference is more substantial for the basis functions effecting the behavior of the curve in the vicinity of existing sharp local gradients, implying that the corresponding weights tend to take the extreme values in these regions.



(a) (b)

Figure 2.14. (a) Initial, and (b) optimal sets of quintic basis functions associated with Figure 2.13.

2.4. **MATLAB Toolbox:** GNURBS Lab

In order to facilitate understanding the behavior of GNURBS and further abilities they provide, a comprehensive interactive MATLAB toolbox, *GNURBS Lab*, has been developed. This toolbox is developed via the extension of an existing NURBS toolbox in MATLAB, *Bspline Lab*, available as an opensource package under GNU license at github.com.

A snapshot of the *GNURBS Lab* environment is depicted in Figure 2.15, which demonstrates some of the available features in this software. The figure shows an example of designing a quadratic GNURBS curve with 5 control points constructed over a uniform knot-vector. Employing the provided tools, one can easily manipulate any defining parameter of the curve, including the locations of control points, knots or weight components, and observe the changes interactively in both the original GNURBS and its equivalent higher order counterpart, simultaneously.

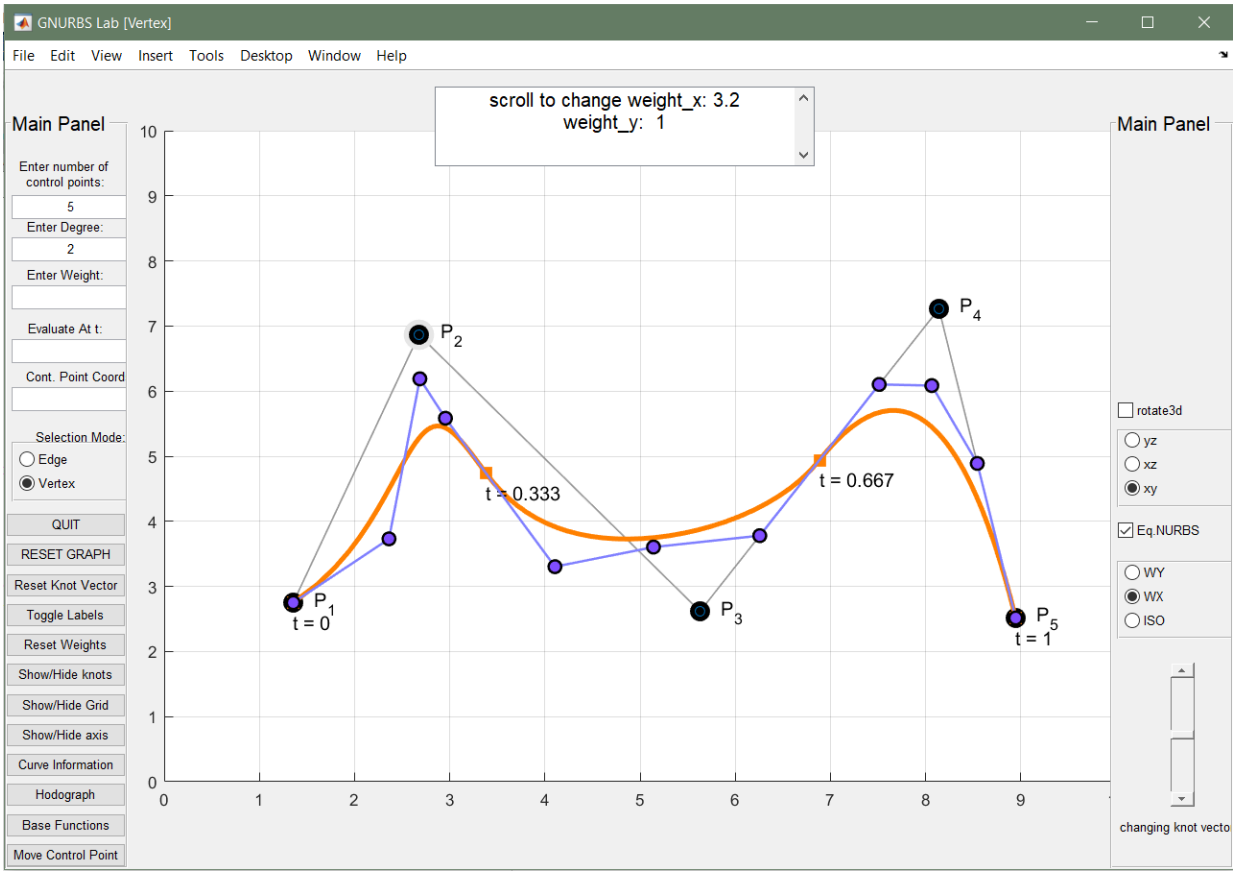


Figure 2.15. A snapshot of GNURBS lab.

The open-source toolbox is available at http://www.ersl.wisc.edu/software/GNURBS-Lab.zip Detailed instructions for using this toolbox is also available as an additional document *Manual.pdf* via the same link.

Chapter 3: Generalizations of NURBS surfaces[2]

In this chapter, we develop extensions of GNURBS for bi-variate parametric surfaces. We will demonstrate that GNURBS can be effectively used for improved approximation of certain class of surfaces such as helicoids, revolved surfaces and minimal surfaces. Further, these generalizations will establish the foundation for the proposed adaptivity technique in IGA which will be discussed in the next chapter.

3.1. Generalized NURBS surfaces: non-isoparametric form via explicit decoupling of the weights

We recall that the equation of a NURBS surface is defined in the following parametric form

$$\mathbf{S}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}R_{ij}^{p,q}(\xi,\eta)\mathbf{P}_{ij} \qquad \begin{matrix} a\le\xi\le b \\ c\le\eta\le d \end{matrix} \tag{3.1}$$

where $\mathbf{P}_{ij}=[x_{ij},y_{ij},z_{ij}]^T$ is a set of $(n_1+1)\times(n_2+1)$ control points and $R_{ij}^{p,q}(\xi,\eta)$ are the corresponding rational basis functions associated with $(i,j)^{\text{th}}$ control point defined as

$$R_{ij}^{p,q}(\xi,\eta)=\frac{N_{ij}^{p,q}(\xi,\eta)w_{ij}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)w_{kl}} \tag{3.2}$$

where w_{ij} are the weights associated with control points, and $N_{ij}^{p,q}(\xi,\eta)=N_{i,p}(\xi)N_{j,q}(\eta)$ are bivariate B-spline basis functions. $N_{i,p}(\xi)$ and $N_{j,q}(\eta)$ are the univariate B-spline basis functions of degree p and q defined on sets of non-decreasing real numbers $\Xi=\{\xi_0,\xi_1,...,\xi_{n_1+p}\}$ and $\mathbf{H}=\{\eta_0,\eta_1,...,\eta_{n_2+q}\}$, respectively, called knot vectors.

According to Eq. (3.1), NURBS surfaces are *isoparametric* representations where all the physical coordinates are constructed by linear combination of the same set of scalar basis functions in parametric space. This is the case for all the other popular CAGD representations such as different

[2] The presented materials in this chapter have been submitted for publication at Engineering with Computers journal.

types of splines; and ensures critical properties such as affine invariance and convex hull which are of interest in geometric modelling [78].

We extend here the concept of Generalized Non-Uniform Rational B-Splines (GNURBS) [78] to surfaces by modifying Eq. (3.1) as follows

$$\mathbf{S}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}\mathbf{R}_{ij}^{p,q}(\xi,\eta)\odot\mathbf{P}_{ij} \quad \begin{matrix} a\le\xi\le b \\ c\le\eta\le d \end{matrix} \tag{3.3}$$

where $\odot$ denotes Hadamard (entry-wise) product of two vector variables and $\mathbf{R}_{ij}(\xi,\eta)=[R_{ij}^{x}(\xi,\eta),\ R_{ij}^{y}(\xi,\eta),\ R_{ij}^{z}(\xi,\eta)]^{T}$ is now a vector set of basis functions. Note that superscripts p,q have been omitted for brevity. Denoting an arbitrary coordinate in physical space by $d\in\{x,y,z\}$, the corresponding basis function in direction *d* can be written as

$$R_{ij}^{d}(\xi,\eta)=\frac{N_{ij}^{p,q}(\xi,\eta)\,w_{ij}^{d}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)\,w_{kl}^{d}}. \tag{3.4}$$

In above equations, $\left(w_{ij}^{x},w_{ij}^{y},w_{ij}^{z}\right)$ represent the set of coordinate-dependent weights associated with $(i,j)^{\text{th}}$ control point.

Comparison of the above equation with that of classic NURBS in Eq. (3.1) shows that the main difference of the proposed generalized form is assigning independent weights to different physical coordinates of control points. As can be seen, the above leads to a *non-isoparametric* representation. This representation demonstrates different geometric properties compared to NURBS which are discussed in detail in the following section.

3.1.1. Theory and properties

It can be shown that due to coordinate-dependence of basis functions, a GNURBS surface need not satisfy properties such as strong convex hull and affine invariance. We here demonstrate that most of the theoretical properties which were discussed for GNURBS curves in the previous chapter can be extended for GNURBS surfaces.

1. Local modification effect

Similar to NURBS, one can show that, in GNURBS, if a control point $\mathbf{P}_{ij}$ is moved, or if any of the weights $w_{ij}^{d}(d = xy, z)$ is changed, it affects the surface shape only over the rectangle $[\xi_i, \xi_{i+p+1}) \times [\eta_j, \eta_{j+q+1})$. However, unlike NURBS, changing the weights will only affect the parameterization of the surface along the corresponding physical coordinate d, while the surface parameterization in the other directions will be preserved. This is, in fact, the key difference between GNURBS and NURBS which provides additional flexibility. In particular, assuming $(\xi, \eta) \in [\xi_i, \xi_{i+p+1}) \times [\eta_j, \eta_{j+q+1})$, if w_i^d is increased (decreased), the surface will move closer to (farther from) $\mathbf{P}_{ij}$. Further, for a fixed (ξ, η), a point on $\mathbf{S}(\xi, \eta)$ moves along a straight line along d towards $\mathbf{P}_{ij}$ as a weight w_{ij}^{d} is modified. This can be directly concluded from Eq. (3.3) and the properties of classic NURBS.

For better insight, we provide here a graphical representation of how this property differs in GNURBS compared to NURBS. For this purpose, we first generate a B-spline surface with linear in-plane parameterization using a net of 7×7 control points and quadratic basis functions in both parametric directions constructed over the knot vectors $\boldsymbol{\Xi} = \mathbf{H} = \{0, 0, 0, 0.2, 0.4, 0.6, 0.8, 1, 1, 1\}$. The employed net of control points is illustrated in Figure 3.1. As the figure shows, the heights of all control points are set to zero except for z_{44} which is raised to 1.

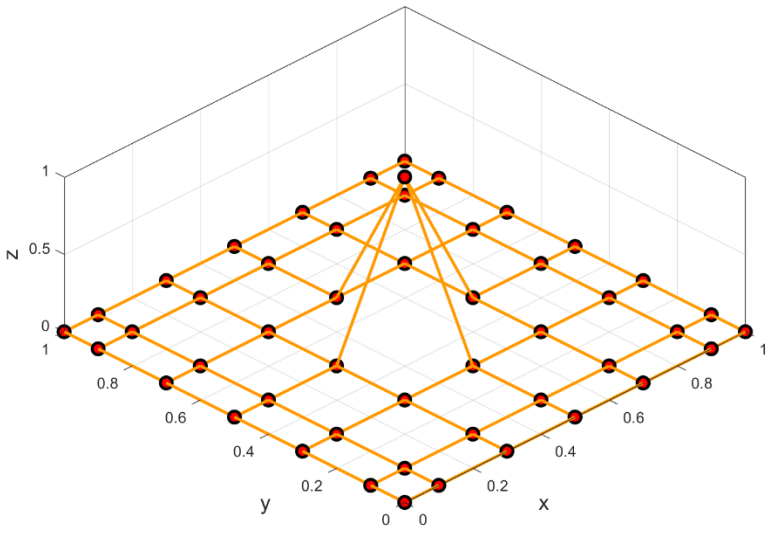


Figure 3.1. Employed control net for construction of different NURBS surfaces.

The B-spline surface obtained by using this control net is depicted in Figure 3.2.

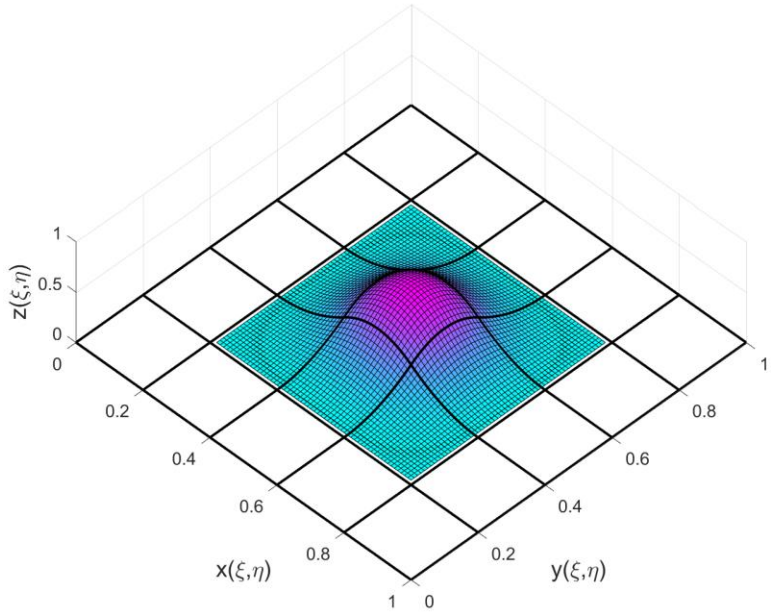


Figure 3.2. The B-spline surface in physical space.

Next, we increase w_{44} to 4 and plot the resulting NURBS surface in the physical space in Figure 3.3.

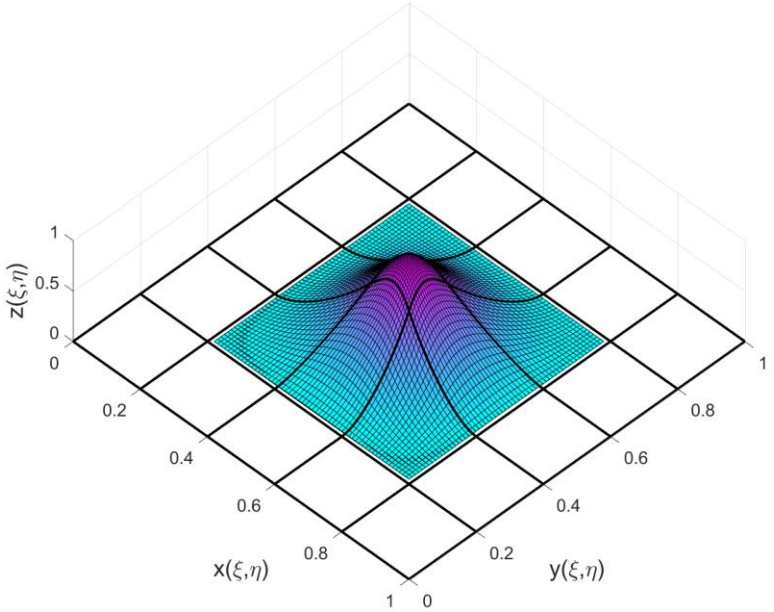


Figure 3.3. The NURBS surface with $w_{44} = 4$ in physical space.

Finally, using Eq. (3.3), we construct a GNURBS surface by only setting w_{44}^{z} to 4, and maintaining all other weights at 1. The resulting surface is shown in Figure 3.4.

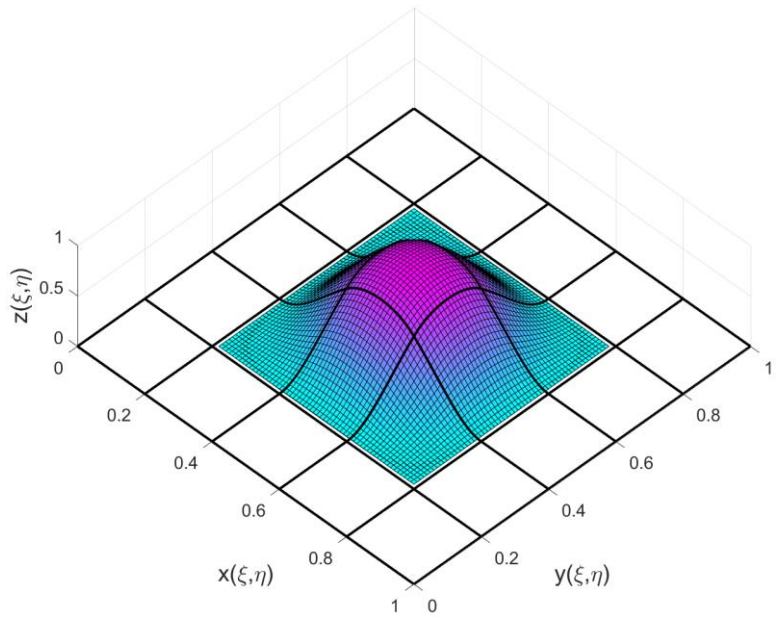


Figure 3.4. The GNURBS surface with $w^z_{44} = 4$ in physical space.

Note that the depicted GNURBS surface in Figure 3.4 is obtained by using two different sets of basis functions. The in-plane coordinates are obtained using the B-spline basis functions, while the out of plane coordinate is constructed using rational basis functions.

Comparing Figure 3.3 and Figure 3.4, one can clearly observe that modifying a weight in classic NURBS alters the parameterization of the surface in all physical directions, while in the case of GNURBS, the parameterization of the surface only changes in the direction of the varied directional weight (*z*-direction in Figure 3.4). It will be seen later that this property is critical for treating the weights as additional degrees of freedom in certain applications.

2. *Axis-aligned bounding box (AABB):*

Every GNURBS knot-element lies within the *axis-aligned bounding box* of its corresponding control points. That is, if $(\xi,\eta) \in [\xi_i, \xi_{i+1}) \times [\eta_j, \eta_{j+1})$, then $\mathbf{S}(\xi,\eta)$ lies within the bounding box of the control points $\mathbf{P}_{kl}$, $i-p \le k \le i$ and $j-q \le l \le j$.

Proof: Eq. (3.3) can be easily written in the following form:

$$\begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{Bmatrix} = \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} R^x_{ij}(\xi,\eta) \begin{Bmatrix} x_{ij} \\ 0 \\ 0 \end{Bmatrix} + \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} R^y_{ij}(\xi,\eta) \begin{Bmatrix} 0 \\ y_{ij} \\ 0 \end{Bmatrix} + \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} R^z_{ij}(\xi,\eta) \begin{Bmatrix} 0 \\ 0 \\ z_{ij} \end{Bmatrix} \qquad (3.5)$$

Accordingly, Eq. (3.5) could be written as

$$\mathbf{S}(\xi,\eta)=\mathbf{S}_x(\xi,\eta)+\mathbf{S}_y(\xi,\eta)+\mathbf{S}_z(\xi,\eta), \qquad \begin{cases} a\le\xi\le b \\ c\le\eta\le d \end{cases} \tag{3.6}$$

where $\mathbf{S}_x(\xi,\eta)$, $\mathbf{S}_y(\xi,\eta)$ and $\mathbf{S}_z(\xi,\eta)$ are simply classic NURBS surfaces. From a geometric standpoint, each of these surfaces is the projection of the original non-isoparametric surface onto the corresponding physical axes.

The following figure shows a graphical representation of the above equations for a quadratic × cubic GNURBS surface constructed over the knot vectors $\Xi=\{0,0,0,\frac{1}{3},\frac{2}{3},1,1,1\}$ and $\mathrm{H}=\{0,0,0,0,\frac{1}{3},\frac{2}{3},1,1,1,1\}$. Random weights in *z*-direction have been assigned to the control points and the control points are plotted proportional to these weights in size for better insight.

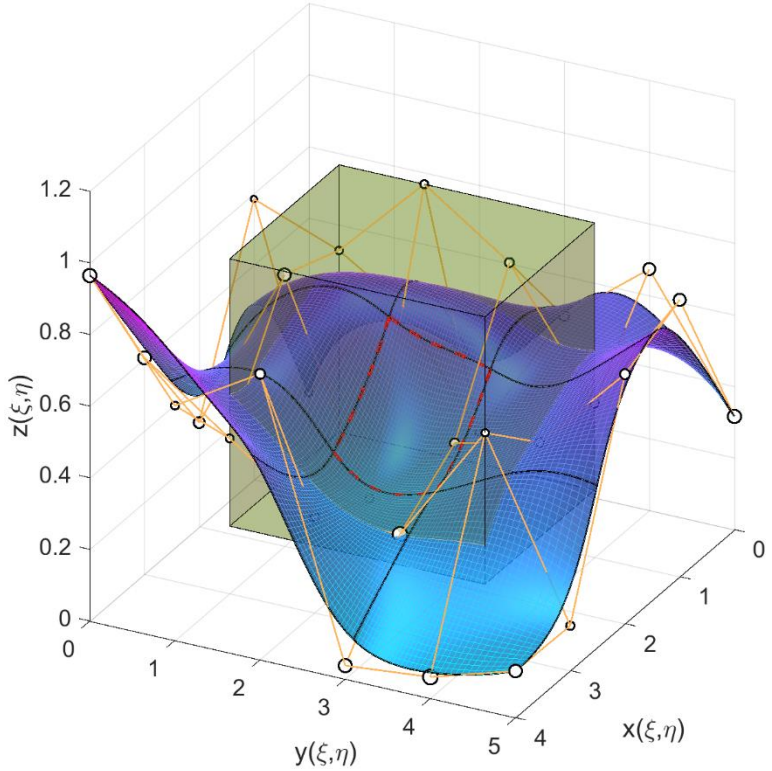


Figure 3.5. Geometric representation of the bounding box property for a GNURBS surface.

Since each of these projected surfaces is a classic NURBS surface, they satisfy the convex hull property. Therefore, the middle knot-element of the surface which is marked in Figure 3.5, must lie within the convex hulls of its corresponding control points on all three projected surfaces. That is, if $(\xi,\eta)\in\left[\frac{1}{3},\frac{2}{3}\right)\times\left[\frac{1}{3},\frac{2}{3}\right)$, then $\mathbf{S}_x(\xi,\eta)$ lies within the convex hull of the control points

$(x_{kl},0,0)$, $1\le k\le 3$ and $1\le l\le 4$ which is the space between the two planes parallel to *yz*-plane. Similarly, $\mathbf{S}_y(\xi,\eta)$ lies within the convex hull of the control points $(0,y_{kl},0)$, $1\le k\le 3$ and $1\le l\le 4$ which is the area between the two planes parallel to *xz*-plane, and $\mathbf{S}_z(\xi,\eta)$ lies within the convex hull of the control points $(0,0,z_{kl})$, $1\le k\le 3$ and $1\le l\le 4$ which is the area between the two planes parallel to *xy*-plane. Consequently, $\mathbf{S}(\xi,\eta)$ is contained in the intersection of these six planes, which is the highlighted box area shown in Figure 3.5, referred to as the axis-aligned bounding box of $\mathbf{P}_{kl}$, $1\le k\le 3$ and $1\le l\le 4$.

3.2. Non-isoparametric 3D GNURBS surfaces with partial decoupling of the weights

A more practical variation of GNURBS which will also later form the foundation for *w*-adaptivity in IGA, is obtained by partial decoupling of the weights. In particular, for 3D surfaces, one can use the same set of in-plane weights along *x* and *y* directions, denoted by w^{xy}, and a different set of out-of-plane weights in *z* direction w^z. Accordingly, Eq. (3.3) could be re-written in the following expanded form

$$\begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{Bmatrix} = \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} \begin{Bmatrix} R_{ij}^{xy}(\xi,\eta)\,x_{ij} \\ R_{ij}^{xy}(\xi,\eta)\,y_{ij} \\ R_{ij}^{z}(\xi,\eta)\,z_{ij} \end{Bmatrix} \quad \begin{matrix} a\le\xi\le b \\ c\le\eta\le d \end{matrix} \tag{3.7}$$

where

$$R_{ij}^{xy}(\xi,\eta) = \frac{N_{ij}^{p,q}(\xi,\eta)\,w_{ij}^{xy}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\,w_{kl}^{xy}} \tag{3.8}$$

Observe that owing to this decoupling of the in-plane and out-of-plane weights, unlike in classic NURBS, one can now freely manipulate the weights along *z* direction, for instance, without perturbing the geometry or parameterization of the underlying planer surface in *x*-*y* plane.

3.3. Equivalence with NURBS

Despite losing some properties of NURBS which might be of interest in certain applications, we state here a theorem which establishes that GNURBS are nothing but disguised forms of higher-

order classic NURBS. Therefore, all the properties of NURBS can be recovered through a suitable transformation and a strong theoretical foundation will be ensured.

Theorem 1. A 3D GNURBS surface of degree (p,q) with partially decoupled set of weights (w^{xy}, w^{z}), can be exactly transformed into a higher order NURBS surface of degree $(2p, 2q)$.

We first review two lemmas on the multiplication of Bézier, as well as B-spline bivariate functions. The proofs of these lemmas can be found in [62]

Lemma 1:

Let $f_b(\xi,\eta)$ and $g_b(\xi,\eta)$ be two $(p, q)^{\text{th}}$-degree bivariate Bézier functions defined as

$$\begin{aligned} f_b(\xi,\eta) &= \sum_{i=0}^{p}\sum_{j=0}^{q} B_{i,p}(\xi)B_{j,q}(\eta)\, f_{ij} \\ g_b(\xi,\eta) &= \sum_{i=0}^{p}\sum_{j=0}^{q} B_{i,p}(\xi)B_{j,q}(\eta) g_{ij} \end{aligned} \tag{3.9}$$

Their product function $h_b(\xi,\eta)$ is a Bézier function of degree $(2p, 2q)$ which can be computed as

$$h_b(\xi,\eta) = f_b(\xi,\eta)g_b(\xi,\eta) = \sum_{i=0}^{2p}\sum_{j=0}^{2q} B_{i,2p}(\xi)B_{j,2q}(\eta)H_{ij}^{b} \tag{3.10}$$

where the ordinates of the product Bézier function are

$$H_{ij}^{b} = \sum_{k=\max(0,i-p)}^{\min(p,i)} \sum_{l=\max(0,j-q)}^{\min(q,j)} \frac{\binom{p}{k}\binom{p}{i-k}}{\binom{2p}{i}} \frac{\binom{q}{l}\binom{q}{j-l}}{\binom{2q}{j}} f_{kl}\, g_{i-k,j-l} \tag{3.11}$$

Lemma 2:

Let $f(\xi,\eta)$ and $g(\xi,\eta)$ be two $(p, q)^{\text{th}}$-degree bivariate B-spline functions defined as

$$\begin{aligned} f(\xi,\eta) &= \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, f_{ij} \\ g(\xi,\eta) &= \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, g_{ij} \end{aligned} \tag{3.12}$$

Their product function $h(\xi,\eta)$ is a B-spline function of degree $(2p, 2q)$, that is

$$h(\xi,\eta) = f(\xi,\eta)g(\xi,\eta) = \sum_{i=0}^{n_1^h}\sum_{j=0}^{n_2^h} N_{ij}^{2p,2q}(\xi,\eta)H_{ij} \tag{3.13}$$

where H_{ij} are the ordinates of the product B-spline function.

Specific to this lemma, many algorithms have been proposed in the literature for evaluating the ordinates H_{ij}; see e.g. [62–65]. We will here use a straightforward algorithm proposed by Piegl and Tiller [61] including three steps of

- Performing Bézier extraction
- Computation of the product of Bézier functions
- Recomposition of the Bézier product functions into B-spline form using knot removal.

where the product of Bézier functions in the second step can be computed analytically employing Lemma 1. Further, the knot vector of $h(\xi)$ can be constructed as described in [61]. Alternatively, one may use a more advanced algorithm referred to as Sliding Windows Algorithm (SWA) recently proposed by Chen et al. [63].

Proof. The proof relies on the lemma that the summation of two NURBS surfaces is a higher order NURBS surface [61]. We rewrite Eq. (3.7) in Section 3.2 in the following form:

$$\begin{Bmatrix} \mathrm{x}(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{Bmatrix} = \frac{\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{xy}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\, w_{kl}^{xy}} \begin{Bmatrix} x_{ij} \\ y_{ij} \\ 0 \end{Bmatrix} + \frac{\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{z}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\, w_{kl}^{z}} \begin{Bmatrix} 0 \\ 0 \\ z_{ij} \end{Bmatrix} \tag{3.14}$$

Extracting the common denominator yields

$$\begin{aligned} x(\xi,\eta) &= \frac{\left(\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{xy} x_{ij}\right)\left(\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\, w_{kl}^{z}\right)}{D(\xi,\eta)} \\ y(\xi,\eta) &= \frac{\left(\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{xy} y_{ij}\right)\left(\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\, w_{kl}^{z}\right)}{D(\xi,\eta)} \\ z(\xi,\eta) &= \frac{\left(\sum_{i=0}^{n_1}\sum_{j=0}^{n_2} N_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{z} z_{ij}\right)\left(\sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta)\, w_{kl}^{xy}\right)}{D(\xi,\eta)} \end{aligned} \tag{3.15}$$

where

$$D(\xi,\eta)=\left(\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)\,w_{kl}^{xy}\right)\left(\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)\,w_{kl}^{z}\right) \tag{3.16}$$

As can be observed, evaluation of the above expressions involves performing the multiplication of bivariate B-spline functions. According to Lemma 2, all the product functions in Eq. (3.15) are B-spline functions of degree (2*p*, 2*q*). Therefore, we can obtain the equivalent higher order NURBS representation of Eq. (3.14) in the following form

$$\begin{Bmatrix}x(\xi,\eta)\\ y(\xi,\eta)\\ z(\xi,\eta)\end{Bmatrix}=\sum_{i=0}^{\hat{n}_1}\sum_{i=0}^{\hat{n}_2}R_{ij}^{2p,2q}(\xi,\eta)\begin{Bmatrix}X_i\\ Y_i\\ Z_i\end{Bmatrix} \tag{3.17}$$

where

$$R_{ij}^{2p,2q}(\xi,\eta)=\frac{N_{ij}^{2p,2q}(\xi,\eta)W_{ij}}{\sum_{k=0}^{\hat{n}_1}\sum_{l=0}^{\hat{n}_2}N_{kl}^{2p,2q}(\xi,\eta)W_{kl}} \tag{3.18}$$

in which $(X_{ij},Y_{ij},Z_{ij},W_{ij})$ are the coordinates and weights of the $(\hat{n}_1+1)\times(\hat{n}_2+1)$ control points of the equivalent higher order NURBS surface, which can be obtained using any of the mentioned algorithms followed by Lemma 2. □

In the special case of Rational Bézier (R-Bézier) surfaces, one can obtain straightforward analytical expressions for the coefficients of the equivalent higher order R-Bézier surface in Eq. (3.17). For this case, Eqs. (3.17) and (3.18) can be written as

$$\begin{Bmatrix}x(\xi,\eta)\\ y(\xi,\eta)\\ z(\xi,\eta)\end{Bmatrix}=\sum_{i=0}^{2p}\sum_{j=0}^{2q}R_{ij}^{2p,2q}(\xi,\eta)\begin{Bmatrix}X_{ij}\\ Y_{ij}\\ Z_{ij}\end{Bmatrix} \tag{3.19}$$

where

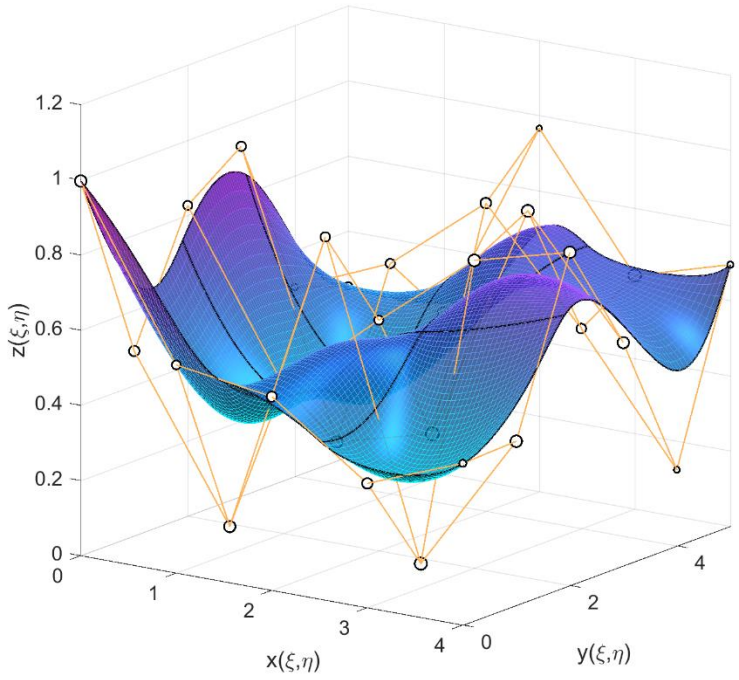
$$R_{ij}^{2p,2q}(\xi,\eta)=\frac{B_{ij}^{2p,2q}(\xi,\eta)W_{ij}}{\sum_{k=0}^{2p}\sum_{l=0}^{2q}B_{kl}^{2p,2q}(\xi,\eta)W_{kl}} \tag{3.20}$$

Using relations (3.10) and (3.11) in Lemma 1, the coordinates and weights of control points in these equations can be obtained as

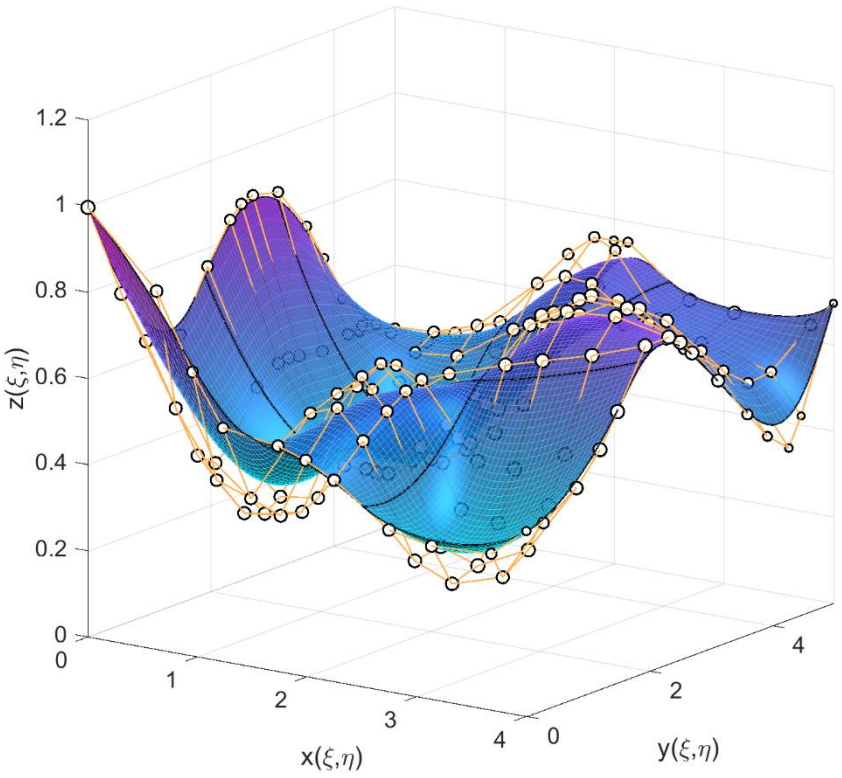
$$
\begin{aligned}
W_{ij} &= \sum_{k=\max(0,i-p)}^{\min(p,i)} \sum_{l=\max(0,j-q)}^{\min(q,j)} \lambda_{ik}^{p}\, \lambda_{jl}^{q}\, w_{i-k,j-l}^{xy}\, w_{kl}^{z} \\
X_{ij} &= \frac{1}{W_{ij}} \sum_{k=\max(0,i-p)}^{\min(p,i)} \sum_{l=\max(0,j-q)}^{\min(q,j)} \lambda_{ik}^{p}\, \lambda_{jl}^{q}\, w_{kl}^{xy} x_{kl} w_{i-k,j-l}^{z} \\
Y_{ij} &= \frac{1}{W_{ij}} \sum_{k=\max(0,i-p)}^{\min(p,i)} \sum_{l=\max(0,j-q)}^{\min(q,j)} \lambda_{ik}^{p}\, \lambda_{jl}^{q}\, w_{kl}^{xy} y_{kl} w_{i-k,j-l}^{z} \\
Z_{ij} &= \frac{1}{W_{ij}} \sum_{k=\max(0,i-p)}^{\min(p,i)} \sum_{l=\max(0,j-q)}^{\min(q,j)} \lambda_{ik}^{p}\, \lambda_{jl}^{q}\, w_{i-k,j-l}^{xy}\, z_{kl} w_{kl}^{z}
\end{aligned}
\tag{3.21}
$$

where $\lambda_{ij}^{n} = \dfrac{\binom{n}{j}\binom{n}{i-j}}{\binom{2n}{i}}$.

Figure 3.6(a) shows an example of a degree $(2,3)^{th}$ GNURBS surface with random directional weights assigned in *z*-direction. Its equivalent higher order NURBS surface obtained using the above theorem is depicted in Figure 3.6(b). Note that the size of control points in these figures are plotted proportional to their weights for better insight.



(a)



(b)

Figure 3.6. (a) A degree (2,3)[th] GNURBS surface with random weights assigned in *z*-direction, and (b) its equivalent (isoparametric) NURBS surface of degree (4,6).

3.4. Generalized NURBS surfaces: isoparametric form via implicit decoupling of the weights

It is interesting to note that the equivalent higher order NURBS representation in (3.17) itself provides another variation of NURBS which can be directly employed as another alternative to NURBS with better flexibility in many applications.

In order to clarify how this equation provides additional flexibility than classic NURBS, we first derive a more generic form of this equation via an alternative approach using an extension of order elevation technique. In this case, we limit our study to rational Bézier surfaces for simplicity.

Assume a 2D R-Bézier surface of degree (*p*,*q*) is given as follows

$$\begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \end{Bmatrix} = \sum_{i=0}^{p}\sum_{j=0}^{q} \frac{B_{ij}^{p,q}(\xi,\eta)\, w_{ij}^{xy}}{w^{xy}(\xi,\eta)} \begin{Bmatrix} x_{ij} \\ y_{ij} \end{Bmatrix} \tag{3.22}$$

In order to elevate the degree of this surface by (*r*, *s*), we can simply multiply both numerator and denominator of this equation by any arbitrary expression in the following form

$$w^z(\xi,\eta)=\sum_{i=0}^{r}\sum_{j=0}^{s}B_{ij}^{r,s}(\xi,\eta)\,w_{ij}^z \tag{3.23}$$

Recalling Lemma 1, we can obtain the higher order R-Bézier surface with (*r*, *s*) degree elevations as

$$\begin{Bmatrix}x(\xi,\eta)\\ y(\xi,\eta)\end{Bmatrix}=\sum_{i=0}^{\hat{p}}\sum_{j=0}^{\hat{q}}R_{ij}^{\hat{p},\hat{q}}\begin{Bmatrix}X_{ij}\\ Y_{ij}\end{Bmatrix} \tag{3.24}$$

where

$$R_{ij}^{\hat{p},\hat{q}}(\xi,\eta)=\frac{B_{ij}^{\hat{p},\hat{q}}(\xi,\eta)W_{ij}}{\sum_{k=0}^{\hat{p}}\sum_{l=0}^{\hat{q}}B_{kl}^{\hat{p},\hat{q}}(\xi,\eta)W_{kl}} \tag{3.25}$$

in which $\hat{p}=p+r,\hat{q}=q+s$ and $\left(X_{ij},Y_{ij},W_{ij}\right)$ can be obtained using the following relations

$$\begin{aligned}
W_{ij}&=\sum_{k=\max(0,i-r)}^{\min(p,i)}\sum_{l=\max(0,j-s)}^{\min(q,j)}\lambda_{ik}^{p,r}\,\lambda_{jl}^{q,s}\,w_{kl}^{xy}\,w_{i-k,j-l}^{z}\\
X_{ij}&=\frac{1}{W_{ij}}\sum_{k=\max(0,i-r)}^{\min(p,i)}\sum_{l=\max(0,j-s)}^{\min(q,j)}\lambda_{ik}^{p,r}\,\lambda_{jl}^{q,s}\,w_{kl}^{xy}x_{kl}w_{i-k,j-l}^{z}\\
Y_{ij}&=\frac{1}{W_{ij}}\sum_{k=\max(0,i-r)}^{\min(p,i)}\sum_{l=\max(0,j-s)}^{\min(q,j)}\lambda_{ik}^{p,r}\,\lambda_{jl}^{q,s}\,w_{kl}^{xy}y_{kl}w_{i-k,j-l}^{z}
\end{aligned} \tag{3.26}$$

where $\lambda_{ij}^{\alpha,\beta}=\dfrac{\binom{\alpha}{j}\binom{\beta}{i-j}}{\binom{\alpha+\beta}{i}}$.

Observe that this procedure can be seen as a trivial extension of the classic order elevation techniques in the literature [19,66]. In fact, one can simply recover the common order elevation algorithm by assigning $w_{ij}^z=1,\forall(i,j)$ in Eq. (3.23). We will refer to this procedure as *generalized order elevation* hereafter. Now assume we intend to add another dimension to the degree-elevated representation in Eq. (3.24) in an isoparametric manner. For this purpose, we extend this equation as

$$\begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{Bmatrix} = \sum_{i=0}^{\hat{p}} \sum_{j=0}^{\hat{q}} R_{ij}^{\hat{p},\hat{q}}(\xi,\eta) \begin{Bmatrix} X_{ij} \\ Y_{ij} \\ Z_{ij} \end{Bmatrix} \tag{3.27}$$

It is interesting to notice that, although Eq. (3.27) apparently seems to be a classic R-Bézier surface, it provides additional flexibility. Observe that in the above procedure, w_{ij}^{z} are arbitrary variables which can be freely chosen without perturbing the geometry or parameterization of the underlying surface in *x*-*y* plane. For better insight, we perform degree elevation on a circular annulus using the above procedure with different selections of w_{ij}^{z} and discuss how it differs from classic degree elevation technique.

For this purpose, we generate a 3D $(\hat{p},\hat{q}) = (3,2)$ isoparametric GR-Bézier surface by performing the above degree-elevation processes with $(r,s) = (1,1)$ on an initial quarter annulus modelled by a $(p,q) = (2,1)$ R-Bézier surface depicted in Fig. 6(a) and specifying the heights of control points of the degree-elevated surface as shown in Table 3.1.

Table 3.1. Assigned heights (z_{ij}) to the control points of the resulting degree-elevated isoparametric GR-Bézier surface.

	i=0	*i*=1	*i*=2	*i*=3
j=0	0	1	1	0
j=1	0	1	1	0
j=2	0	1	1	0

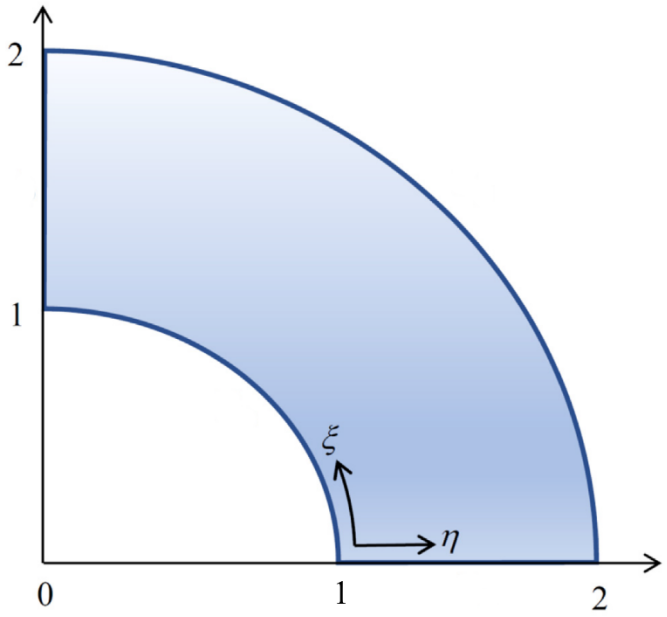


Figure 3.7. Configuration of the quarter annulus.

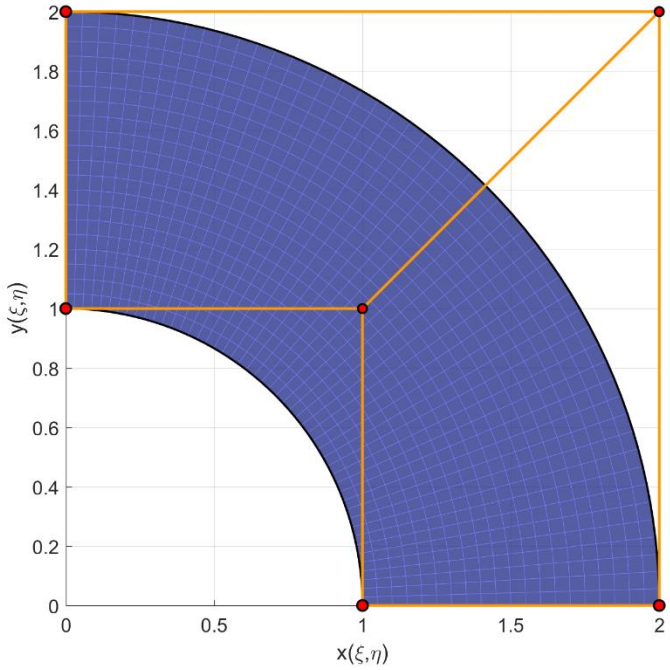
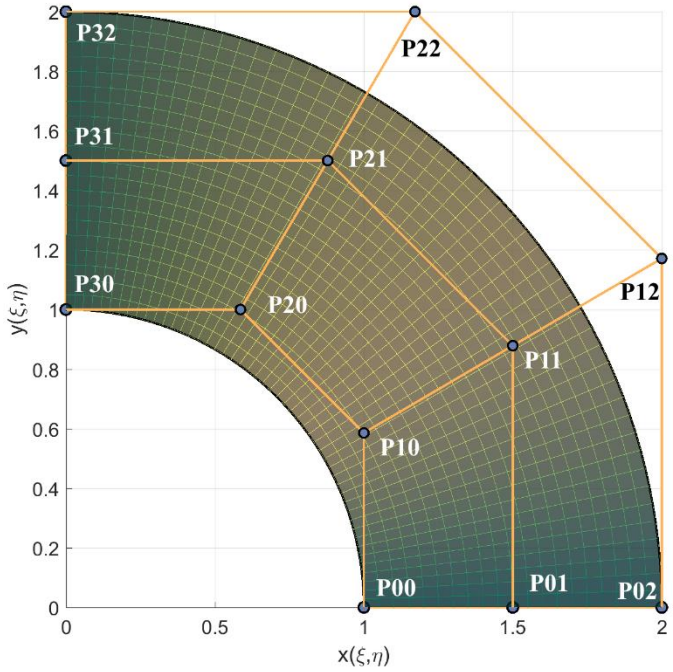


Figure 3.8. Exact representation of the quarter annulus with normal parameterization using a (p, q)=(2,1) rational Bézier surface.

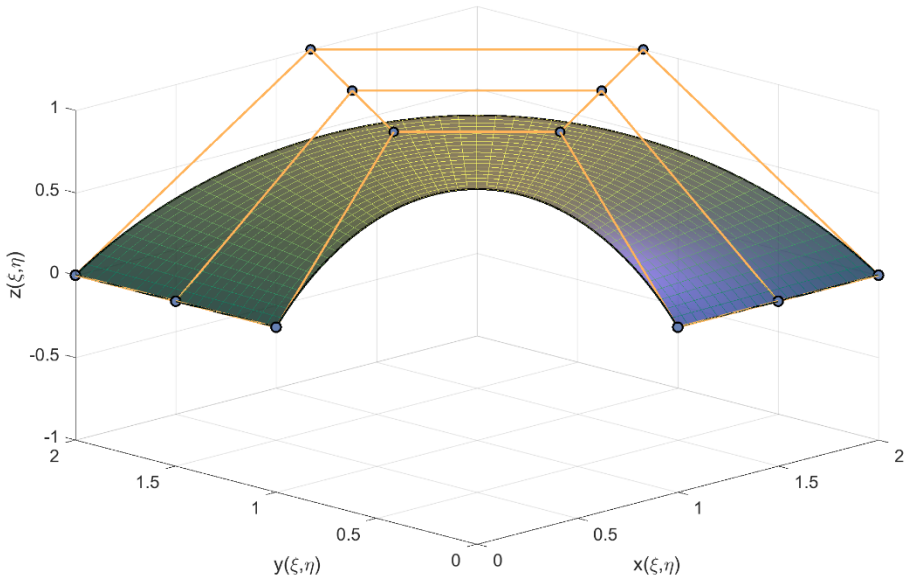
The obtained results for classic order elevation, that is, assuming unit values for all isoparametric control weights as in the following equation:

$$\begin{bmatrix} w_{11}^{z} & w_{12}^{z} \\ w_{21}^{z} & w_{22}^{z} \end{bmatrix} = \begin{bmatrix} 1.0 & 1.0 \\ 1.0 & 1.0 \end{bmatrix} \tag{3.28}$$

are shown in Figure 3.9.



(a)



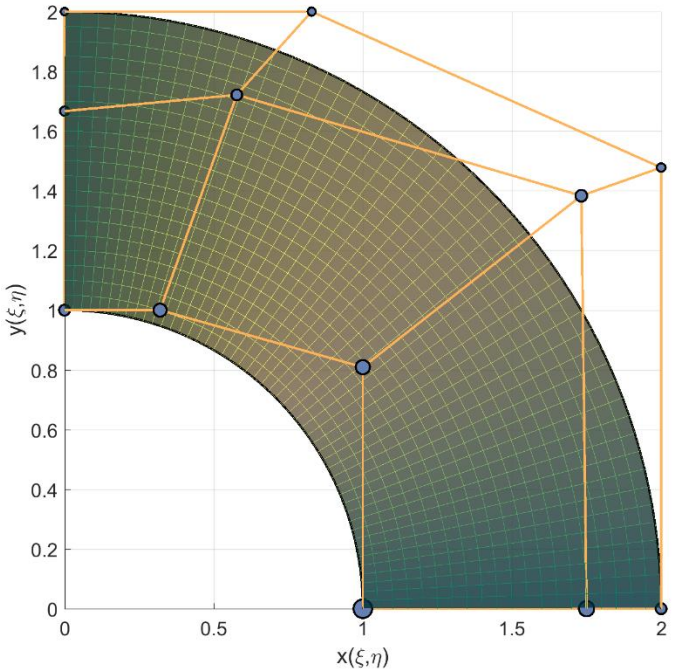
(b)

Figure 3.9. Classic degree-elevated R-Bézier representation of the quarter annulus with control variables of Table 3.1: (a) top view, (b) 3D view.

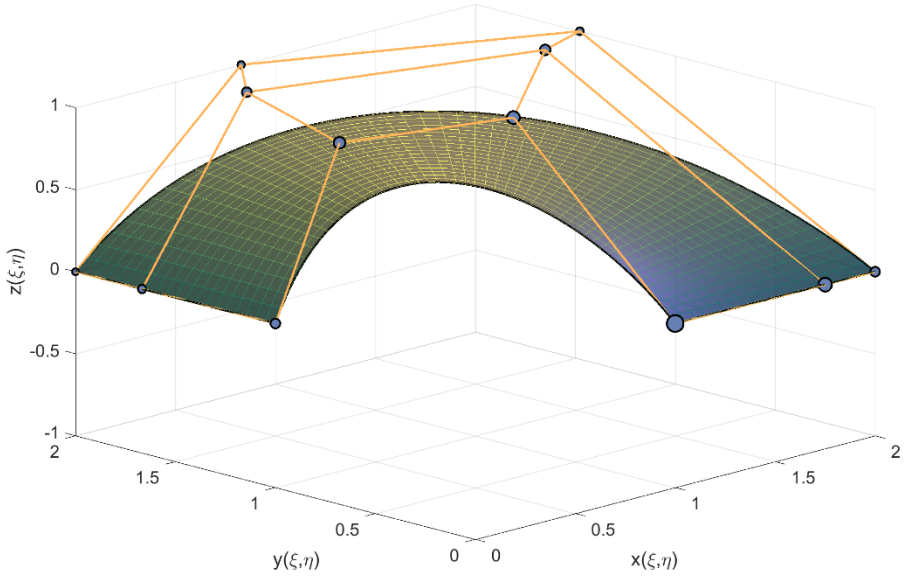
Moreover, the obtained results for generalized order elevation by assuming the following values for isoparametric control weights:

$$\begin{bmatrix} w_{11}^z & w_{12}^z \\ w_{21}^z & w_{22}^z \end{bmatrix} = \begin{bmatrix} 3.0 & 1.0 \\ 1.0 & 0.5 \end{bmatrix} \tag{3.29}$$

are depicted in Figure 3.10.



(a)



(b)

Figure 3.10. Generalized degree-elevated R-Bézier representation of the quarter annulus with control variables of Table 3.1: (a) top view, (b) 3D view.

As can be clearly seen in these figures, in both cases, the in-plane representation of the annular ring as well as its parameterization has remained unchanged. However, the out of plane deformation of the annular ring in the two cases are not identical.

While this variation of NURBS, which will be referred to as isoparametric GNURBS hereafter, similarly provides the same important possibility of treating the out of plane weights as additional degrees of freedom, it provides different advantages. In particular, unlike the first variation, it allows for introducing customized rationality for approximation, i.e. the number of coefficients to be considered as design variables in the denominator can be controlled here. Further, it directly lies in the NURBS space; hence, all the properties of NURBS are naturally satisfied.

The above algorithm can also be extended to NURBS in a straightforward manner using a similar three step algorithm elaborated above. That is, Eq. (3.27) also holds true for NURBS:

$$\begin{Bmatrix} x(\xi,\eta) \\ y(\xi,\eta) \\ z(\xi,\eta) \end{Bmatrix} = \sum_{i=0}^{\hat{n}_1} \sum_{j=0}^{\hat{n}_2} R_{ij}^{\hat{p},\hat{q}}(\xi,\eta) \begin{Bmatrix} X_{ij} \\ Y_{ij} \\ Z_{ij} \end{Bmatrix} \tag{3.30}$$

with the rational basis functions defined as

$$R_{ij}^{\hat{p},\hat{q}}(\xi,\eta) = \frac{N_{ij}^{\hat{p},\hat{q}}(\xi,\eta) W_{ij}}{\sum_{k=0}^{\hat{n}_1} \sum_{l=0}^{\hat{n}_2} N_{kl}^{\hat{p},\hat{q}}(\xi,\eta) W_{kl}} \tag{3.31}$$

The proposed generalizations of NURBS in Eqs. (3.7) and (3.30) can effectively improve the performance of NURBS in a wide area of applications. Exploring all these applications, however, is beyond the scope of this thesis. We limit our study here to a few classic examples in geometric modelling, that is the approximation of certain class of surfaces such as helical, revolved or minimal surfaces using GNURBS; and concisely point out some of their potential broader areas of applications. Finally, hereafter, we will persistently refer to Eq. (3.7) as the first generalization of NURBS or non-isoparametric GNURBS, while we will refer to Eq. (3.30) as the second generalization of NURBS or isoparametric GNURBS.

3.5. Least-square surface approximation using NURBS versus GNURBS

In this section, we demonstrate that the proposed generalizations of NURBS are able to provide superior approximation for certain class of surfaces compared to classic NURBS. We assume here

that a planar geometry with precise representation using NURBS, such as the annular ring in Figure 3.8, is given as:

$$\mathbf{S}_{xy}(\xi,\eta)=\begin{Bmatrix}x(\xi,\eta)\\ y(\xi,\eta)\end{Bmatrix},\quad \begin{matrix}0\le\xi\le 1\\ 0\le\eta\le 1\end{matrix} \tag{3.32}$$

Next, we assume that an analytical height function $z(\xi,\eta)$ is given and needs to be approximated with minimal error over the given planer surface. The problem can be posed as a least square approximation problem which leads to optimal accuracy in L_2-norm. Considering $\{(\xi_s,\eta_s)\to(x_s,y_s,z_s):s\in\mathcal{S}\}$ as a set of n_c chosen collocation points, the error function f to be minimized is defined as

$$f=\frac{1}{2}\sum_{s\in\mathcal{S}}\left\|\hat{z}(\xi_s,\eta_s)-\bar{z}_s\right\|^2=\frac{1}{2}\sum_{s\in\mathcal{S}}\left\|\sum_{L\in\mathcal{L}^s}R_L(\xi_s,\eta_s)z_L-\bar{z}_s\right\|^2 \tag{3.33}$$

where $\hat{z}(\xi,\eta)$ is the approximated NURBS function, $\mathcal{L}^s$ is the set of indices of non-zero basis functions at (ξ_s,η_s), $\bar{z}_s=z(\xi_s,\eta_s)$, and z_L are the unknown *control variables*. For simplicity, the global index L is used for numbering which is defined as $L=j(n_1+1)+i+1$ for the basis (i,j). In the following, we provide the detailed formulation of this problem using NURBS as well as its different proposed generalizations.

3.5.1. Linear least-square approximation using NURBS

In the case of NURBS, the only unknowns to consider are *control variables* z_L. Taking the partial derivatives of f with respect to the unknowns z_L, and setting them to zero yields

$$\frac{\partial f}{\partial z_k}=0,\qquad k=1,\ldots,n_T \tag{3.34}$$

$$\sum_{s\in\mathcal{S}}\sum_{L\in\mathcal{L}^s}R_k(\xi_s,\eta_s)R_L(\xi_s,\eta_s)z_L=\sum_{s\in\mathcal{S}}\bar{z}_sR_k(\xi_s,\eta_s),\qquad k=1,\ldots,n_T \tag{3.35}$$

where $n_T=(n_1+1)\times(n_2+1)$ denotes the total number of control points. Eq. (3.35) could be written in the matrix form

$$\sum_{s\in\mathcal{S}}\begin{pmatrix} R_1(\xi_s,\eta_s)R_1(\xi_s,\eta_s) & \cdots & R_1(\xi_s,\eta_s)R_{n_T}(\xi_s,\eta_s) \\ \vdots & \ddots & \vdots \\ R_{n_T}(\xi_s,\eta_s)R_1(\xi_s,\eta_s) & \cdots & R_{n_T}(\xi_s,\eta_s)R_{n_T}(\xi_s,\eta_s) \end{pmatrix}\begin{pmatrix} z_1 \\ \vdots \\ z_{n_T} \end{pmatrix} = \sum_{s\in\mathcal{S}}\overline{z}_s\begin{pmatrix} R_1(\xi_s,\eta_s) \\ \vdots \\ R_{n_T}(\xi_s,\eta_s) \end{pmatrix} \tag{3.36}$$

which represents a classic linear least square problem and can be easily solved for the n_T unknowns $\boldsymbol{\lambda} = \{z_1,...,z_{n_T}\}$ by proper choice of collocation points.

3.5.2. Non-linear least-square approximation using non-isoparametric GNURBS

In order to improve the accuracy of the above discussed NURBS-based approximation, we develop a non-linear least-square minimization algorithm using 1st GNURBS. Invoking the non-isoparametric GNURBS surface with partial decoupling of the weights in Section 3.2, we can treat the out of plane weights w_L^z as extra design variables without perturbing the geometry or parameterization of the underlying precise planar surface $\mathbf{S}_{xy}(\xi,\eta)$. We may refer to these variables as *control weights* hereafter.

The objective function to be minimized could still be written as (3.33). However, the vector of design variables now changes to $\boldsymbol{\lambda}' = \{z_1,...,z_{n_T}, w_1^z,...,w_{n_T}^z\}$. Moreover, the following bounding constraints on control weights are often desired to be satisfied for numerical stability.

$$w_k^z > 0, \qquad k = 1,...,n_T \tag{3.37}$$

Eq. (3.33) with the new vector of design variables $\boldsymbol{\lambda}'$ establishes a non-linear least-square optimization problem which could be solved using different existing algorithms. Some of these algorithms, such as Levenberg-Marquardt, do not allow for the imposition of bounding constraints on design variables. In this case, one can easily apply an exponential transformation to control weights to ensure their positivity without the imposition of bounding constraints as in [20]. We will use here the trust-region-reflective algorithm which is available in MATLAB and allows for the imposition of bounding constraints on design variables.

In order to solve the established problem, the Jacobian matrix is required. The Jacobian matrix $\mathbf{J}$ is composed of two parts

$$\mathbf{J} = [\mathbf{J}_z \ \ \mathbf{J}_w] \tag{3.38}$$

where $\mathbf{J}_z$ contains the partial derivatives of f with respect to z_k, while $\mathbf{J}_w$ includes the partial derivatives of f with respect to w_k^z. Differentiating with respect to z_k, $\mathbf{J}_z$ will be easily derived as

$$\mathbf{J}_z = \begin{bmatrix} R_1(\xi_1,\eta_1) & R_2(\xi_1,\eta_1) & \cdots & R_{n_T}(\xi_1,\eta_1) \\ R_1(\xi_2,\eta_2) & R_2(\xi_2,\eta_2) & \cdots & R_{n_T}(\xi_2,\eta_2) \\ \vdots & \vdots & \ddots & \vdots \\ R_1(\xi_{n_c},\eta_{n_c}) & R_2(\xi_{n_c},\eta_{n_c}) & \cdots & R_{n_T}(\xi_{n_c},\eta_{n_c}) \end{bmatrix} \tag{3.39}$$

The other component of the Jacobian matrix can be obtained as

$$\mathbf{J}_w = \begin{bmatrix} \dfrac{\partial \hat{z}(\xi_1,\eta_1)}{\partial w_1^z} & \cdots & \dfrac{\partial \hat{z}(\xi_1,\eta_1)}{\partial w_{n_T}^z} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial \hat{z}(\xi_{n_c},\eta_{n_c})}{\partial w_1^z} & \cdots & \dfrac{\partial \hat{z}(\xi_{n_c},\eta_{n_c})}{\partial w_{n_T}^z} \end{bmatrix} \tag{3.40}$$

In order to evaluate the partial derivatives with respect to weight design variables, we rewrite $\hat{z}(\xi,\eta)$ as

$$\hat{z}(\xi,\eta) = \frac{\mathcal{Z}(\xi,\eta)}{\mathcal{W}(\xi,\eta)} \tag{3.41}$$

where $\mathcal{Z}(\xi,\eta)$ and $\mathcal{W}(\xi,\eta)$ are

$$\mathcal{Z}(\xi,\eta) = \sum_{L=1}^{n_T} N_L(\xi,\eta)\, w_L^z z_L \tag{3.42}$$

$$\mathcal{W}(\xi,\eta) = \sum_{L=1}^{n_T} N_L(\xi,\eta)\, w_L^z \tag{3.43}$$

Using these definitions, we can obtain

$$\frac{\partial \hat{z}(\xi,\eta)}{\partial w_k^z} = \frac{N_k(\xi,\eta) w_k^z}{\mathcal{W}(\xi,\eta)}\left[z_k - \hat{z}(\xi,\eta)\right], \qquad k = 1,\ldots,n_T \tag{3.44}$$

Having the analytical Jacobian matrix components in Eqs. (3.39) and (3.40), we can now solve the established non-linear least-square optimization problem efficiently. We impose the initial conditions by setting all the control variables to zero and all the control weights to 1, i.e.

$$\boldsymbol{\lambda}_0' = \left\{ \underbrace{0,0,...,0}_{n_T}, \underbrace{1,1,...,1}_{n_T} \right\} \tag{3.45}$$

3.5.3. Non-linear least-square approximation using isoparametric GNURBS

Since the derivation of analytical Jacobian matrix with this generalization becomes complicated in case of having internal knots, we limit our derivation here to GR-Bézier. Invoking the isoparametric GR-Bézier representation in Eq. (3.27), we can again establish the approximation problem as a non-linear least square problem with the objective function defined in Eq. (3.33) but with the new set of design variables $\boldsymbol{\lambda}'' = \left\{ Z_1,...,Z_{n_T}, w_1^z,...,w_{n_d}^z \right\}$ where $n_d = (r+1)\times(s+1)$ is the total number of isoparametric control weights, and $n_T = (\hat{p}+1)\times(\hat{q}+1)$ is the total number of control points. The Jacobian matrix $\mathbf{J}$ can again be divided into two components as in (3.38) where $\mathbf{J}_z$ contains the partial derivatives of f with respect to Z_k, while $\mathbf{J}_w$ includes the partial derivatives of f with respect to w_l^z. Differentiating with respect to Z_k, $\mathbf{J}_z$ will be easily derived as

$$\mathbf{J}_z = \begin{bmatrix} R_1(\xi_1,\eta_1) & R_2(\xi_1,\eta_1) & \cdots & R_{n_T}(\xi_1,\eta_1) \\ R_1(\xi_2,\eta_2) & R_2(\xi_2,\eta_2) & \cdots & R_{n_T}(\xi_2,\eta_2) \\ \vdots & \vdots & \ddots & \vdots \\ R_1(\xi_{n_c},\eta_{n_c}) & R_2(\xi_{n_c},\eta_{n_c}) & \cdots & R_{n_T}(\xi_{n_c},\eta_{n_c}) \end{bmatrix} \tag{3.46}$$

Also, $\mathbf{J}_w$ can be obtained as

$$\mathbf{J}_w = \begin{bmatrix} \dfrac{\partial \hat{z}(\xi_1,\eta_1)}{\partial w_1^z} & \cdots & \dfrac{\partial \hat{z}(\xi_1,\eta_1)}{\partial w_{n_d}^z} \\ \vdots & \ddots & \vdots \\ \dfrac{\partial \hat{z}(\xi_{n_c},\eta_{n_c})}{\partial w_1^z} & \cdots & \dfrac{\partial \hat{z}(\xi_{n_c},\eta_{n_c})}{\partial w_{n_d}^z} \end{bmatrix} \tag{3.47}$$

In order to evaluate the partial derivatives of $\hat{z}(\xi)$ with respect to isoparametric control weights w_l^z, we rewrite $\hat{z}(\xi)$ as

$$\hat{z}(\xi) = \frac{\mathcal{Z}(\xi)}{\mathcal{W}(\xi)} \tag{3.48}$$

where $\mathcal{Z}(\xi)$ and $\mathcal{W}(\xi)$ are

$$\mathcal{Z}(\xi)=\sum_{L=1}^{n_T} B_L\left(\xi,\eta\right)W_L\, Z_L \tag{3.49}$$

$$\mathcal{W}(\xi)=\sum_{L=1}^{n_T} B_L\left(\xi,\eta\right)W_L \tag{3.50}$$

With these definitions, we can obtain the required derivatives as

$$\frac{\partial \hat{z}(\xi,\eta)}{\partial w_l^z}=\frac{1}{\mathcal{W}(\xi,\eta)}\left[\frac{\partial \mathcal{Z}(\xi,\eta)}{\partial w_l^z}-\hat{z}(\xi,\eta)\frac{\partial \mathcal{W}(\xi,\eta)}{\partial w_l^z}\right],\quad l=1,...,n_d \tag{3.51}$$

The derivatives in above equation can be evaluated using the following expressions

$$\frac{\partial \mathcal{Z}(\xi,\eta)}{\partial w_l^z}=\sum_{L=1}^{n_T} B_L(\xi,\eta)\frac{\partial W_L}{\partial w_l^z}Z_L \tag{3.52}$$

$$\frac{\partial \mathcal{W}(\xi,\eta)}{\partial w_l^z}=\sum_{L=1}^{n_T} B_L(\xi,\eta)\frac{\partial W_L}{\partial w_l^z} \tag{3.53}$$

where

$$\frac{\partial W_L}{\partial w_l^z}=\frac{\partial W_{ij}}{\partial w_{mn}^z}=\begin{cases}\lambda_{i,i-m}^{p,r}\,\lambda_{j,j-n}^{q,s}\,w_{i-m,j-n}^{xy}, & \text{if } (i-p)\le m\le i \;\&\; (j-q)\le n\le j\\ 0 & \text{otherwise}\end{cases} \tag{3.54}$$

in which

$$\lambda_{i,i-m}^{p,r}=\frac{\binom{p}{i-m}\binom{r}{m}}{\binom{p+r}{i}},\quad \lambda_{j,j-n}^{q,s}=\frac{\binom{q}{j-n}\binom{s}{n}}{\binom{q+s}{j}}.$$

Similar to previous case, we specify the initial conditions as follows

$$\boldsymbol{\lambda}_0''=\left\{\underbrace{0,0,...,0}_{n_T},\underbrace{1,1,...,1}_{n_d}\right\} \tag{3.55}$$

As previously discussed, by changing w_l^z during the optimization process, the in-plane coordinates of control points also vary at each iteration. However, since the in-plane geometry and parameterization are always fixed, one may only re-evaluate and update these coordinates after the termination of the optimization process according to the obtained optimal set of isoparametric

basis functions. It is important to note that this algorithm yields the combination of optimal weights and the corresponding arrangement of control points which result in the best approximation for a given in-plane parameterization. To our knowledge, no such investigation has been reported in the literature thus far.

3.6. Numerical examples

3.6.1. Test case 1: helicoid modelling

As the first numerical example, we consider the approximation of a partial helical surface with the following equation:

$$\mathbf{S}(\xi,\eta)=\big((\eta+1)\cos(\xi),(\eta+1)\sin(\xi),\xi\big),\quad \begin{matrix}0\le\xi\le\pi/2\\ 0\le\eta\le 1\end{matrix} \tag{3.56}$$

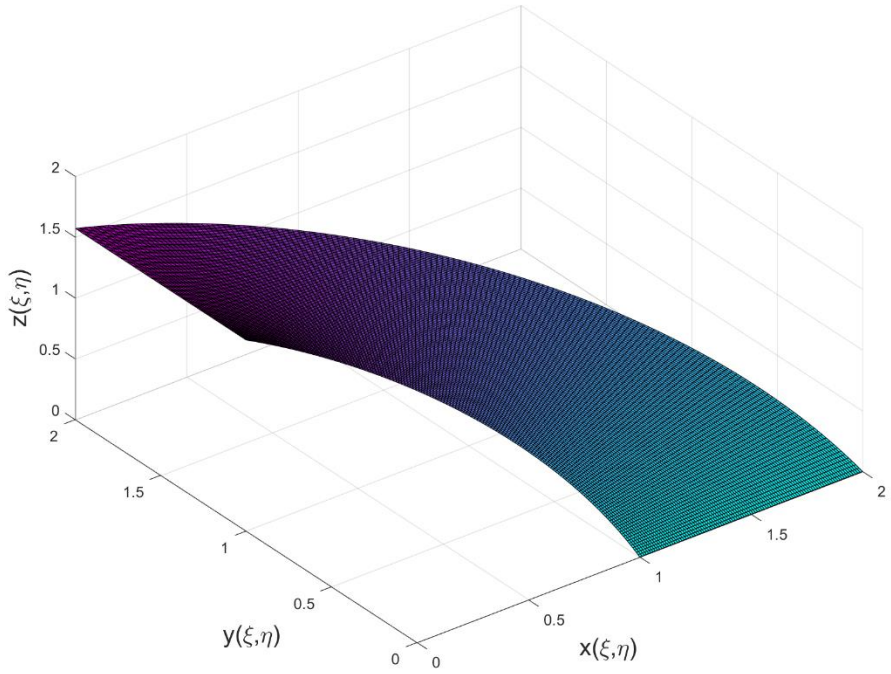


Figure 3.11. The helical surface in Eq. (3.56).

As observed, the in-plane parameterization of this surface is a quarter annulus with the configuration already shown in Figure 3.7. Since this is a geometric modelling problem where preserving the properties of NURBS are of interest, it is an ideal candidate for employing isoparametric GNURBS. Accordingly, following the procedure discussed above in Section 3.5.3, we try to approximate the given height function in Eq. (3.56) and compare the obtained results

with classic NURBS. The obtained results for different degrees of basis functions are presented in Table 3.2.

Table 3.2. Error of approximating the height function of helical surface in Eq. (3.56) using R-Bézier versus isoparametric GR-Bézier in relative L^2-norm.

Curve type	Degree $(\hat{p},\hat{q}) = (p+r, q+s)$	No. of control variables	No. of control weights	Error	Error ratio
R-Bézier	(2,1)	6	0	2.68E-2	1.0
2nd GR-Bézier			0	2.68E-2	
R-Bézier	(3,2)	12	0	1.28E-4	1.0
2nd GR-Bézier			4	1.28E-4	
R-Bézier	(4,3)	20	0	1.28E-4	109.4
2nd GR-Bézier			9	1.17E-6	
R-Bézier	(5,4)	30	0	2.22E-6	180.5
2nd GR-Bézier			16	1.23E-8	

According to this table, by including larger numbers of control weights, better improvement of accuracy is achieved. This reveals superior approximation of rational functions especially when higher degrees of basis functions are employed. The results, however, show no improvement in accuracy for the first level of degree elevation, i.e. $(r,s)=(1,1)$. This implies that the optimal values of control weights for this particular level of degree elevation are unity. In other words, classic order elevation results in optimal accuracy for the approximation of helical height function using this particular degree of basis functions.

3.6.2. Test case 2: Scherk minimal surface

As the second numerical experiment, we consider the construction of a minimal surface model referred to as Scherk minimal surface over a square domain. The equation of this minimal surface is given as

$$\mathbf{S}(\xi,\eta)=\left(\xi,\eta,\ln\left(\frac{\cos(\eta)}{\cos(\xi)}\right)\right), \quad \begin{matrix}-1.5\le\xi\le 1.5\\ -1.5\le\eta\le 1.5\end{matrix} \tag{3.57}$$

which is depicted in Figure 3.12.

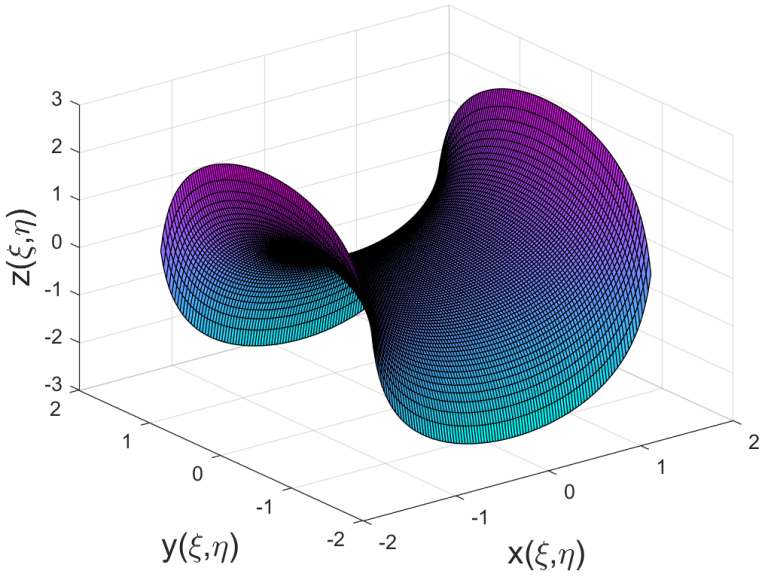


Figure 3.12. Scherk minimal surface.

As the figure shows, the surface features steep gradients near the boundaries. In this example, we use non-isoparametric GNURBS and compare its approximation properties with classic NURBS. The obtained results using various employed degrees of basis functions are shown in Table 3.3. As observed, the accuracy of approximation using 1st GNURBS in all cases is better than that of classic NURBS. Further, the increase in accuracy substantially improves when larger degrees of basis functions are used.

Table 3.3. Error of approximating the Scherk minimal surface in Eq. (3.57) using NURBS versus 1st GNURBS in relative L^2-norm.

Curve type	Degree (p,q)	No. of control variables	No. of control weights	Error	Error ratio
NURBS	(2, 2)	25	0	1.52E-1	18.76
1st GNURBS			25	8.11E-3	
NURBS	(3, 3)	36	0	9.40E-2	18.42
1st GNURBS			36	5.10E-3	
NURBS	(4, 4)	49	0	5.02E-2	142.21
1st GNURBS			49	3.53E-4	
NURBS	(5, 5)	64	0	3.59E-2	262.04
1st GNURBS			64	1.37E-4	

Moreover, for the case of quadratic basis functions, we also perform a convergence study where we persistently refine the knot sequence and compare the obtained accuracy of NURBS versus GNURBS. The obtained results are plotted in Figure 3.13. As the figure shows, the convergence rate of GNURBS is more than order faster than classic NURBS, resulting in substantial improvement of accuracy especially when larger numbers of control points are used.

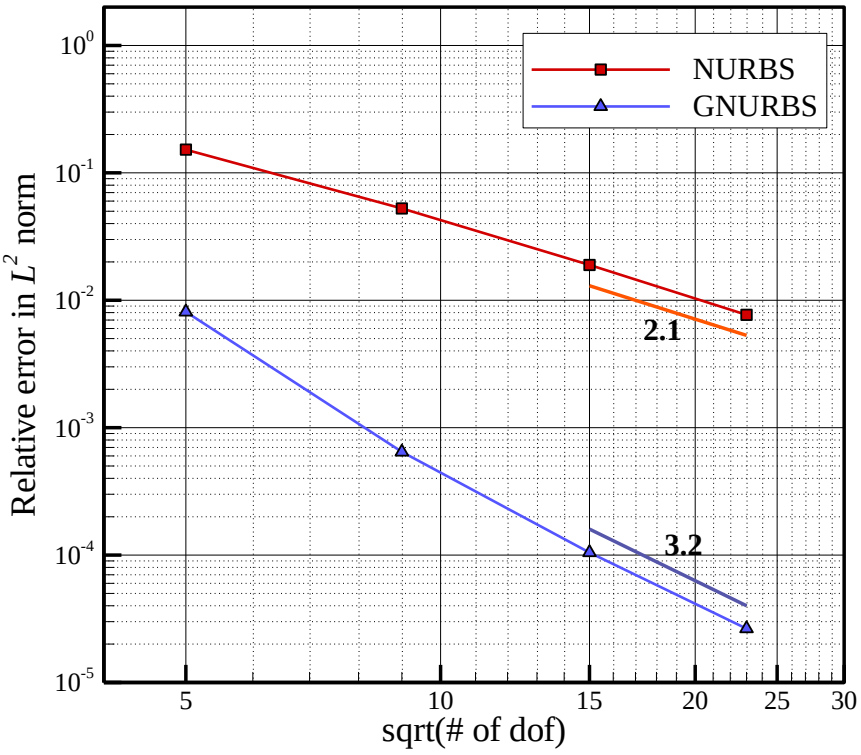


Figure 3.13. Convergence rate of quadratic NURBS versus GNURBS for the approximation of Scherk minimal surface.

3.6.3. Test Case 3: Surface of revolution

As the final numerical study, we consider the problem of the approximation of a surface of revolution defined using Eq. (3.58), which is depicted in Figure 3.14.

$$\mathbf{S}(\xi,\eta)=\left((\eta+1)\cos(\xi),(\eta+1)\sin(\xi),e^{\eta^2}\right),\quad \begin{matrix}0\le\xi\le 2\pi\\ 0\le\eta\le 1\end{matrix} \tag{3.58}$$

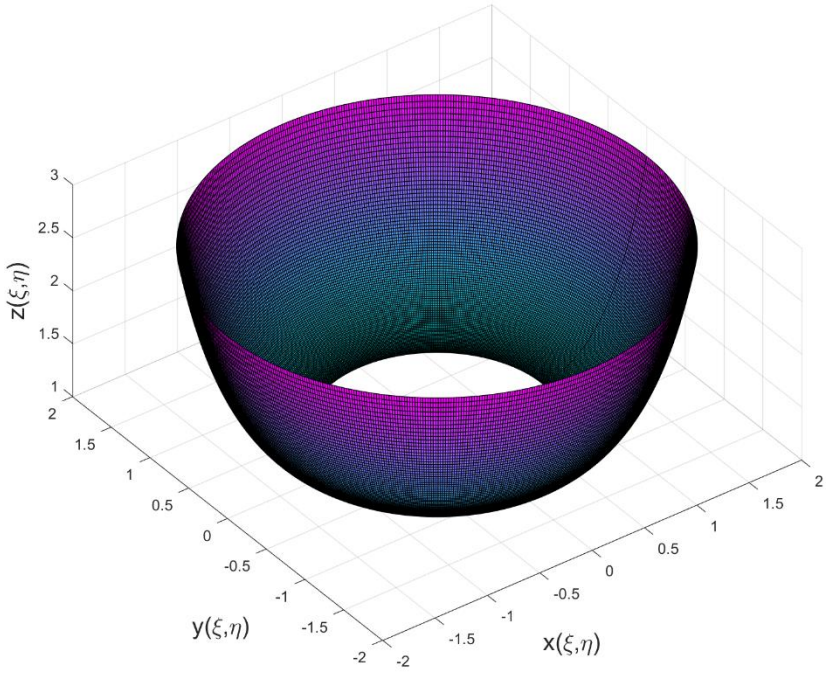


Figure 3.14. The surface of revolution in Eq. (3.58).

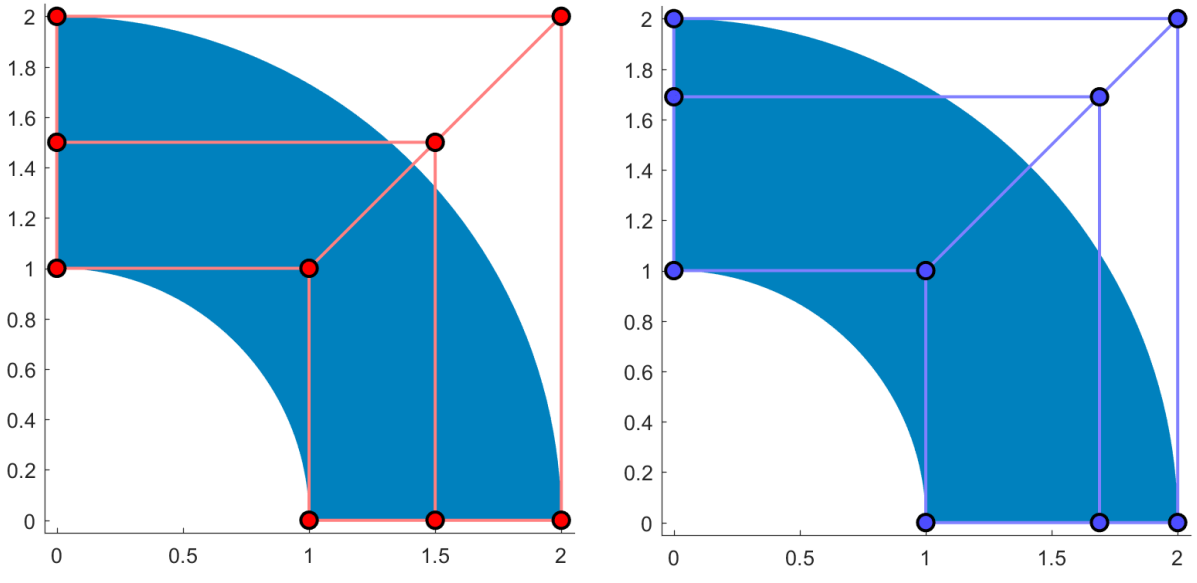
As observed, the surface has an exponential behavior along the radial direction. In this example, we demonstrate how employing the second proposed variation of NURBS could be useful for improved approximation of these type of surfaces using the same number of control points. For simplicity, we only consider modelling a quarter of the surface, i.e. $(0 \le \xi \le \pi/4)$. Similar to the first numerical example, we start with the initial model of degree $(p,q) = (2,1)$ in Figure 3.8. Since the height function here only varies along the radial direction, we only elevate the degree along this direction (η) and compare the obtained approximation results using Bézier (classic order elevation) with those of isoparametric GR-Bézier (optimal order elevation). The obtained results for $(r,s) = (0,0)$ up to $(r,s) = (0,3)$ are presented in Table 3.4.

According to this table, the accuracy of approximation by using isoparametric GR-Bézier is significantly higher than that of classic Bézier, especially when higher order elevations are applied. These results clearly show the superiority of rational functions for the approximation of this class of surfaces.

Table 3.4. Error of approximating the height function of the surface of revolution in Eq. (3.58) using R-Bézier versus isoparametric GR-Bézier in relative L^2-norm.

Case No.	Surface type	Degree $(\hat{p},\hat{q})$ $=(p+r,q+s)$	No. of control variables	No. of control weights	Error	Error ratio
1	R-Bézier	(2,1)	6	0	0.20E0	1.0
2	2nd GR-Bézier			0	0.20E0	
3	R-Bézier	(2,2)	12	0	3.42E-2	45.25
4	2nd GR-Bézier			4	7.55E-4	
5	R-Bézier	(2,3)	20	0	7.10E-3	43.58
6	2nd GR-Bézier			9	1.63E-4	
7	R-Bézier	(2,4)	30	0	1.12E-3	1.26E4
8	2nd GR-Bézier			16	8.90E-8	

Finally, the corresponding arrangements of control points for cases 3 to 8 are represented in Figure 3.15. As observed, the arrangements of control points in all cases only differ along the radial direction. This was expected to be the case, since in this example, order elevation has only been performed along the radial direction.



(a) (b)

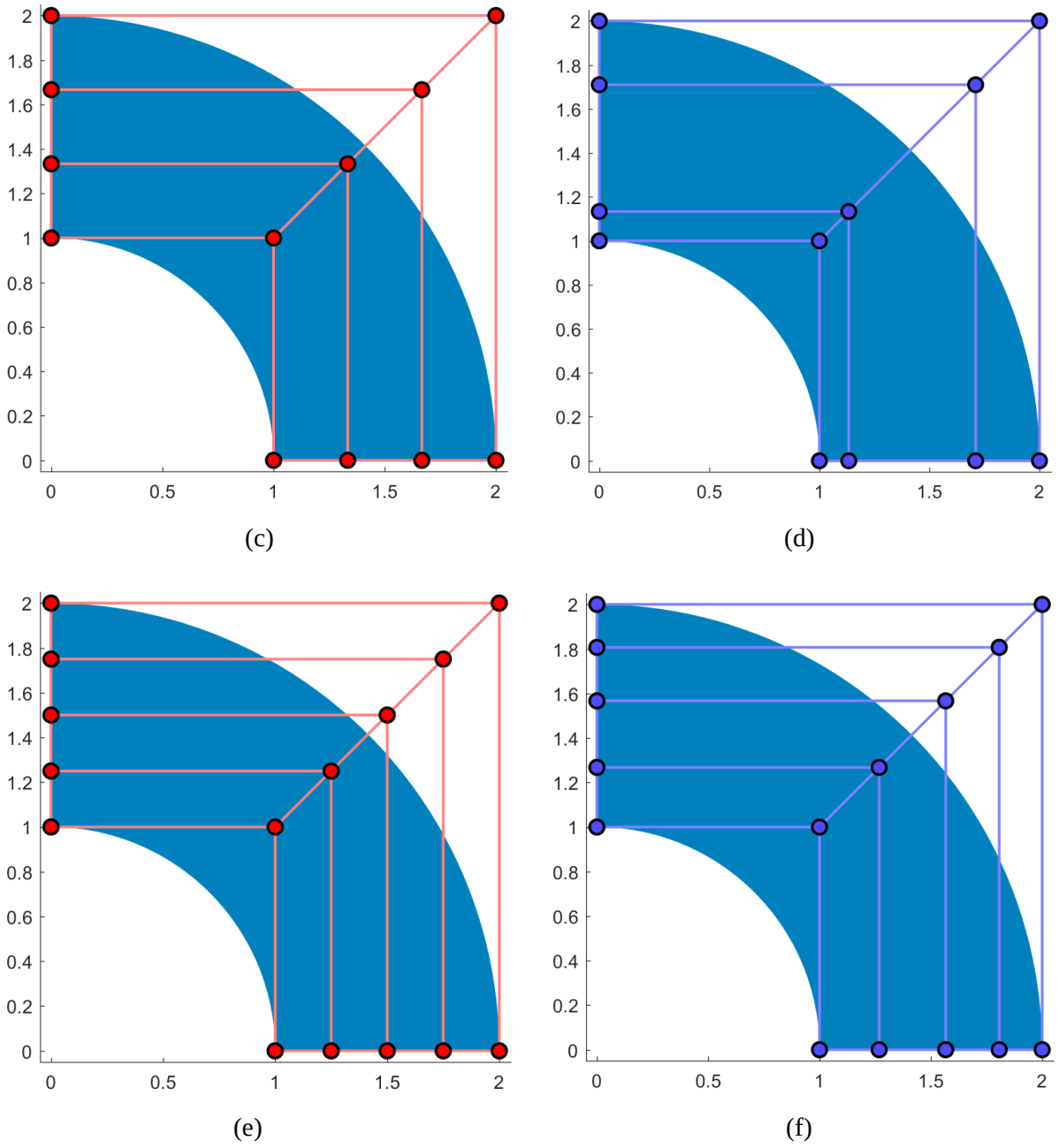


Figure 3.15. The resulting control net for the approximation of the surface of revolution in Eq. (3.58): (a) Case 3, (b) Case 4, (c) Case 5, (d) Case 6, (e) Case 7, and (f) Case 8.

3.7. Extensions and further applications

While, in this thesis, we limited our study to applying the proposed generalizations to NURBS, due to fundamental similarities between different variations of splines, similar generalizations

seem plausible to other rational forms of splines such as T-spline surfaces, Tri-angular Bézier surfaces etc.

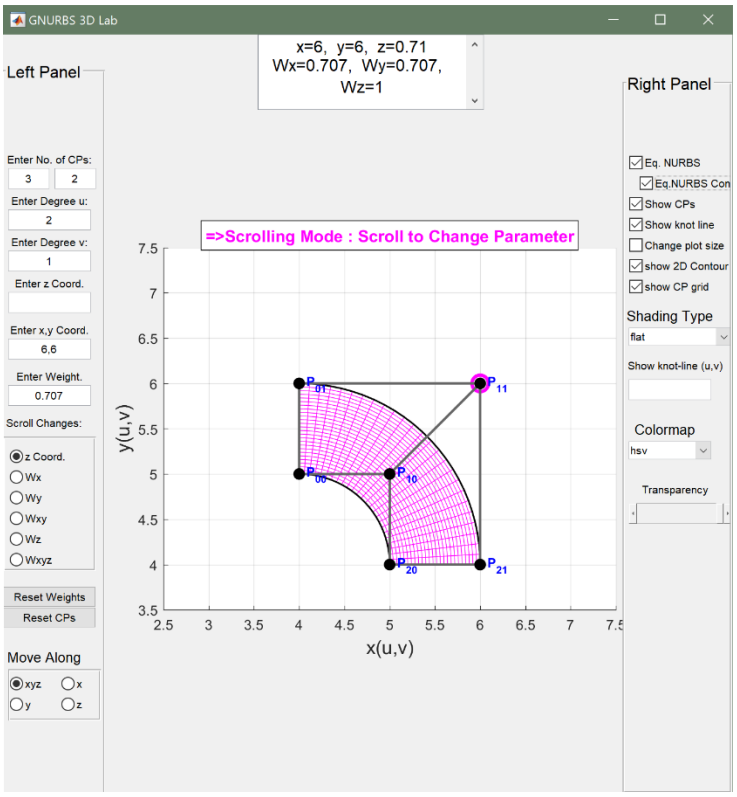
In addition to the discussed applications of GNURBS in CAGD, other applications of NURBS in this area can be found where employing the weights as additional design variables for better flexibility can be problematic or sometimes impossible. For instance, GNURBS may also help circumventing the difficulties of considering the weights as degrees of freedom in general surface fitting problems with arbitrary parameterization. As previously studied in [22,23], employing the weights as additional degrees of freedom in data approximation can deteriorate the surface parameterization, and lead to undesirable results, especially when approximating rapidly varying data. On the other hand, employing GNURBS, by including the *control weights* as design variables, one can create a good surface parameterization and preserve it during fitting without imposing any restrictions on the magnitude of variations of the weights.

Furthermore, NURBS have been extensively used in other disciplines such as computational mechanics for the optimization of different fields of interest over a given computational domain. Considering these studies, we can find out that in this class of applications, the parameterization of the design domain needs to remain fixed throughout the optimization process; see [8,79–87], for instance. Hence, they are only able to treat the out-of-plane coordinates of control points as design variables, as the variation of weights alters the underlying parameterization which is disallowed. However, owing to the proposed GNURBS representations with decoupled weights, one can now treat the control weights as additional design variables while setting up the optimization problem and still preserve the underlying geometry as well as its parameterization. As elaborated in this research, this can lead to significant improvement in the obtained accuracy in both cases of smooth as well as rapidly varying fields. Exploring some of these applications is the subject of our future studies.

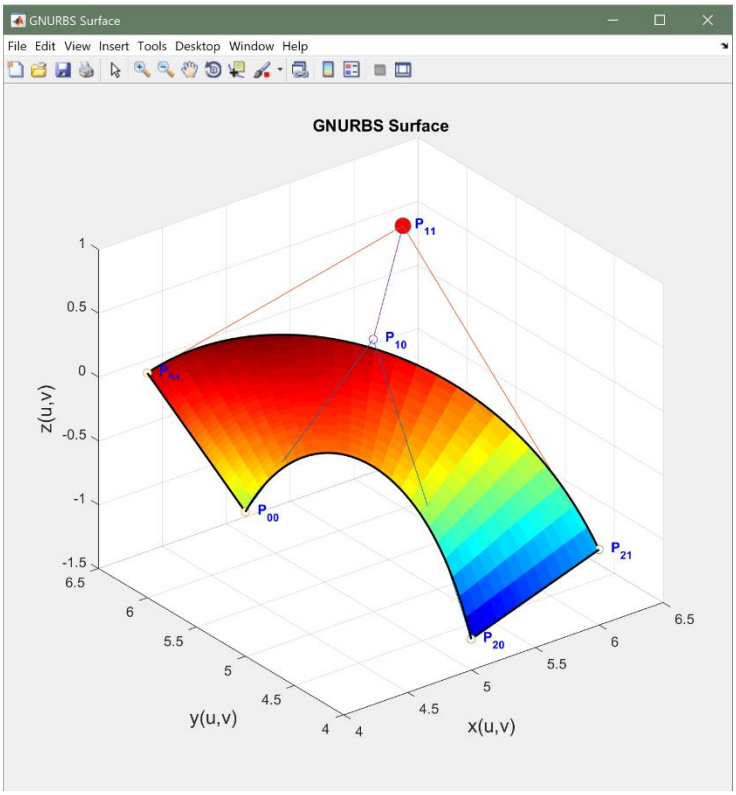
3.8. MATLAB toolbox: GNURBS3D Lab

In order to facilitate understanding the behavior of GNURBS surfaces and the additional abilities they serve, a comprehensive and fully interactive MATLAB toolbox, named *GNURBS3D Lab*, has been developed. This toolbox is developed via the extension of *GNURBS Lab*, a similar interactive MATLAB toolbox already developed for GNURBS curves [78]. Snapshots of different available

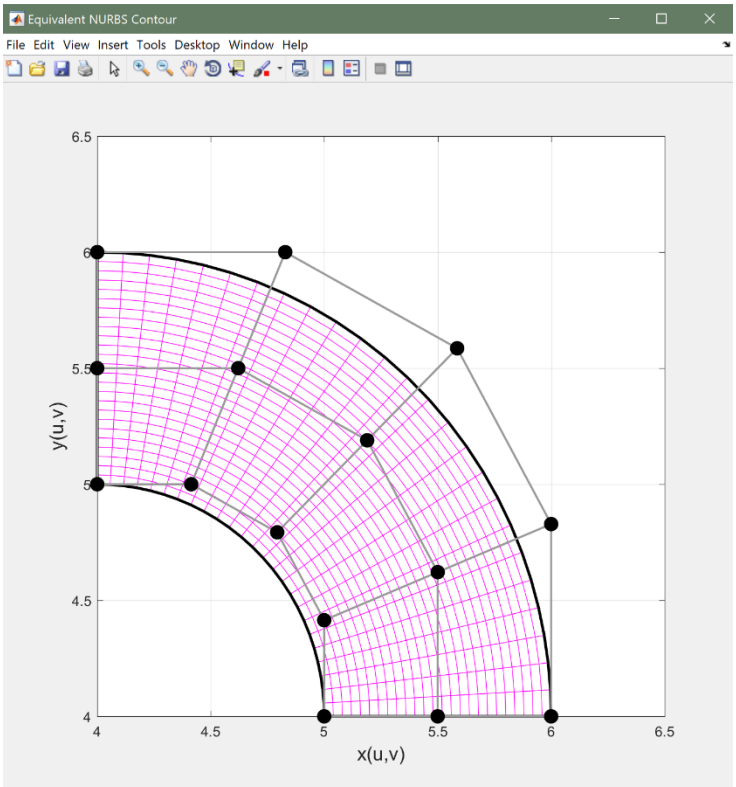
windows in *GNURBS3D Lab* are shown in Figure 3.16, which demonstrate the environment of the toolbox and numerous features that the software provides.



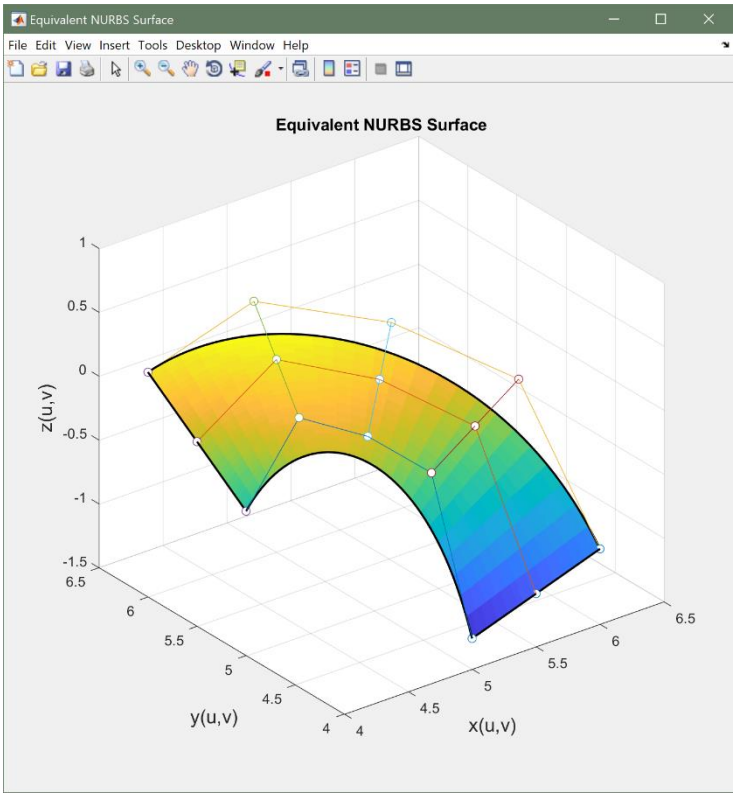
(a)


GNURBS Surface
File Edit View Insert Tools Desktop Window Help
GNURBS Surface
P11
P10
P01
P00
P21
P20
z(u,v)
y(u,v)
x(u,v)

(b)


Equivalent NURBS Contour
File Edit View Insert Tools Desktop Window Help
y(u,v)
x(u,v)

(c)



(d)

Figure 3.16. Snapshots of different windows of GNURBS3D-Lab: (a) Main window, (b) 3D surface plot window, (c) in-plane equivalent NURBS window, and (d) 3D equivalent NURBS window.

The figure shows an example of designing a 3D surface with an in-plane shape of a quarter annulus and a free-form out of plane shape using GR-Bézier. As demonstrated in Figure 3.16, the toolbox is enabled to evaluate the equivalent higher-order rational Bézier representations with the designed surface in 2D and 3D interactively. Employing the provided wide range of tools shown in Figure 3.16(a), one can easily manipulate any defining parameter of the surface, including the locations of control points, or a variety of weight components, and observe the changes interactively in all four windows shown in Figure 3.16, simultaneously.

The open-source toolbox is available at http://www.ersl.wisc.edu/software/GNURBS3D-Lab.zip Detailed instructions for using this toolbox are also provided in an additional document *GNURBS3D_Manual.pdf* accessible via the same link.

Chapter 4: Adaptive *w*-refinement in isogeometric analysis[3]

4.1. GNURBS-based IGA

In this chapter, we consider the application of GNURBS in a proposed adaptive method for improved solution of boundary value problems. While the proposed adaptivity technique is potentially applicable to arbitrary boundary value problems, in this research, we limit our study to single variable elliptic problems. In particular, we focus on steady reactive-diffusive transport problem over a 2D domain $\Omega \in \mathbb{R}^2$ with the boundary $\Gamma = \Gamma_D \bigcup \Gamma_N$, where Γ_D and $\Gamma_N = \Gamma \setminus \Gamma_D$ are the partitions of boundary where Dirichlet and Neumann boundary conditions are specified, respectively. We recall that the strong form of this PDE can be expressed as

$$\begin{aligned} &\mathcal{L}(u) \equiv -\nabla.(D\nabla u) + \sigma u = f \quad \text{in } \Omega, \\ &u = u_D \qquad \text{on } \Gamma_D, \\ &\mathbf{n}.D\nabla u = h \quad \text{on } \Gamma_N \end{aligned} \tag{4.1}$$

where $\mathcal{L}$ denotes the reaction-diffusion operator, D is the diffusion coefficient, σ is the reaction coefficient and f denotes the source term. Also, u_D and h are the specified Dirichlet and Neumann (normal diffusive flux) boundary conditions, and $\mathbf{n}$ denotes the unit outward normal along Γ.

We recall that in 2D NURBS-based IGA, the geometry is constructed by a planar NURBS surface

$$\mathbf{x}(\xi,\eta) = \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} R_{ij}(\xi,\eta)\mathbf{x}_{ij} = \mathbf{RP}, \qquad \begin{matrix} 0 \le \xi \le 1 \\ 0 \le \eta \le 1 \end{matrix} \tag{4.2}$$

where $\mathbf{x} = [x, y]^T$. Following the conventional *isoparametric* concept, the unknown field variable of the PDE is also approximated by using the same set of NURBS basis functions:

$$u(\xi,\eta) \approx u^h(\xi,\eta) = \sum_{i=0}^{n_1}\sum_{j=0}^{n_2} R_{ij}(\xi,\eta)u_{ij} = \mathbf{Ru} \tag{4.3}$$

[3] The presented materials in this chapter have been published in CMAME journal, ref. [112]

where $\mathbf{u}$ is the unknown vector of degrees of freedom, whose components are referred to as *control variables*, and $\mathbf{R}$ denotes the vector of basis functions. This *isoparametric* concept is commonly used in Finite Element Analysis and provides well-known benefits in certain applications which are discussed in [2]. For instance, it ensures the ability to represent all affine motions (i.e., rigid translations and rotations, uniform stretchings and shearings) exactly [2].

However, by invoking the first proposed generalization of NURBS in the previous chapter as well as the concept of GIFT [59], we introduce a natural extension of isogeometric analysis where the field variable is approximated using a set of NURBS basis functions with independent weights of the underlying geometry. Accordingly, Eqs. (4.2) and (4.3) can be written as (4.4) and (4.5)

$$\mathbf{x}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}R_{ij}^{G}(\xi,\eta)\mathbf{x}_{ij}=\mathbf{R}^{G}\mathbf{P} \tag{4.4}$$

$$u^{h}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}R_{ij}^{u}(\xi,\eta)u_{ij}=\mathbf{R}^{u}\mathbf{u} \tag{4.5}$$

where

$$R_{ij}^{G}(\xi,\eta)=\frac{N_{ij}^{p,q}(\xi,\eta)w_{ij}^{G}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)w_{kl}^{G}} \tag{4.6}$$

$$R_{ij}^{u}(\xi,\eta)=\frac{N_{ij}^{p,q}(\xi,\eta)w_{ij}^{u}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_{kl}^{p,q}(\xi,\eta)w_{kl}^{u}} \tag{4.7}$$

Observe that this decoupling of the weights in geometry and field variable space brings the critical possibility of treating the *control weights* w_{ij}^{u} as additional degrees of freedom without perturbing the exact underlying geometry or its parameterization. One can simply imagine that Eq. (4.4) together with Eq. (4.5) represent a GNURBS surface similar to Eq. (3.7), where the in-plane coordinates represents the exact planar geometry, while the out of plane coordinate can be viewed as the field primary variable, i.e. $w^{G}:=w^{xy}, w^{z}:=w^{u}$.

It is noted here that the above non-isoparametric approximation with additional unknown control weights leads to an unknown function space in which the optimal solution of the PDE is sought. This makes formulating the problem by directly applying the standard variational formulation, or the method of weighted residuals commonly used in FEM/IGA difficult. To circumvent this

difficulty, in the next section, we devise an elegant adaptivity technique which yields the optimal values of control weights via an iterative process.

Nevertheless, assuming that an initial guess of control weights is available, we shall proceed with the formulation of the problem to find the corresponding unknown control variables. Following a standard Galerkin formulation as described in [88], the equivalent discrete set of governing equation (4.1) can be obtained in the following form

$$\left(\mathbf{K}_d + \mathbf{K}_r\right)\mathbf{u} = \mathbf{f} \tag{4.8}$$

where $\mathbf{K}_d$ and $\mathbf{K}_r$ are the diffusion and reaction components of the global stiffness matrix, respectively, and $\mathbf{f}$ is the force vector. These expressions can be obtained as

$$\begin{aligned} \mathbf{K}_d &= \sum_P \left(\int_{\Omega_P} (\nabla \mathbf{R}^u)^T \left(D \nabla \mathbf{R}^u \right) \mathrm{d}\Omega \right) \\ \mathbf{K}_r &= \sum_P \left(\int_{\Omega_P} (\mathbf{R}^u)^T \mathbf{R}^u \, \mathrm{d}\Omega \right) \end{aligned} \tag{4.9}$$

and

$$\mathbf{f} = \sum_P \left(\int_{\Omega_P} \left(\mathbf{R}^u \right)^T f \, \mathrm{d}\Omega + \int_{\Gamma_{N,P}} \left(\mathbf{R}^u \right)^T h \, \mathrm{d}\Gamma \right) \tag{4.10}$$

where *P* denotes the index of a typical NURBS patch. Solving (4.8) will yield the unknown control variables such that the obtained solution is optimal in the energy sense.

4.2. Residual-based a posteriori error estimation

The proposed adaptivity algorithm mainly relies on a posteriori error estimator. A variety of a posteriori error estimation techniques have been proposed in the literature; we refer to [89] for a comprehensive review. One can hypothetically use any of these techniques within the framework of adaptive *w*-refinement. In this research, we employ one of the most common techniques which is the residual-based a posteriori error estimation. Defining the interior and boundary residual terms of the reaction-diffusion PDE in Eq. (4.1) as Eqs. (4.11) and (4.12), respectively

$$r = f + \nabla.\left(D \nabla u_h\right) - \sigma u_h \ \text{ in } K, \tag{4.11}$$

$$R = h - \mathbf{n}.D \nabla u_h \ \text{ on } \partial K \cap \Gamma_N, \tag{4.12}$$

this estimator establishes that the energy norm of error $\left\| e_h \right\|_{E(\Omega)}$ which is defined as

$$\left\|e_h\right\|_{E(\Omega)} = \left\|u-u^h\right\|_{E(\Omega)} = \left(\int_\Omega \left(\left[\nabla(u-u^h)\right]^T \nabla(u-u^h)\right) \mathrm{d}\Omega\right)^{1/2} \tag{4.13}$$

is restricted by the following upper bound [34]

$$\left\|e_h\right\|^2_{E(\Omega)} \le \eta_E^2 = C\sum_{K\in\psi}\left(\left\|r\right\|^2_{L^2(K)} h_K^2 + \left\|R\right\|^2_{L^2(\partial K)} h_K\right) \tag{4.14}$$

where C is an unknown constant, K is the index of a knot-element with diameter of a maximal inscribed circle h_K and ψ denotes the set of all knot-elements. The derivation of higher order derivatives in Eq. (4.11) is provided in Appendix A.

4.3. Formulation of adaptive *w*-refinement

The problem could be simply posed as an optimization problem, where the objective function is the estimated error in a desired norm and the design variables are all or a subset of control weights. The optimization problem can be expressed in the following mathematical form

$$\begin{aligned} &Find\ \boldsymbol{\chi} = \left\{w_1^u, w_2^u, ..., w_N^u\right\} \\ &\underset{\chi}{Minimize}\ E(\boldsymbol{\chi}) = \eta_E^2(\boldsymbol{\chi}) \\ &Subject\ to: \mathbf{K}(\boldsymbol{\chi})\mathbf{u} = \mathbf{f}(\boldsymbol{\chi}) \\ &\qquad w_{\min} \le w_L^u \le w_{\max},\ L = 1, ..., N \end{aligned} \tag{4.15}$$

where $\boldsymbol{\chi}$ represents the vector of design variables with N elements and $\mathbf{K} = \mathbf{K}_r + \mathbf{K}_d$. For simplicity, the global index L is used for numbering the design variables which is defined as $L = j(n_1+1)+i+1$ for the basis (i, j). Further, $w_{\min}$ and $w_{\max}$ denote the selected bounds on the design variables. We will later discuss how these bounds be chosen and how they may affect the performance of the algorithm.

It should also be mentioned here that the equilibrium constraint in Eq. (4.15) is intrinsically satisfied and need not be externally imposed. In addition, most unconstrained optimization algorithms allow for the imposition of the simple bounds on design variables as required in Eq. (4.15). Hence, this problem can be regarded as an unconstrained optimization problem in practice.

Moreover, we point out here that a variety of options exists for selecting the vector of design variables in Eq. (4.15). For instance, based on the type of differential operator $\mathcal{L}(u)$ in Eq. (4.1) and expected behavior of the solution, one can decide to include all or a selective subset of control

weights as design variables. Obviously, including a larger number of control weights in Eq. (4.15) is expected to result in better accuracy as well as an increased computational cost. In this study, we will only perform global adaptivity. Further, numerous subtleties need to be undertaken for proper and efficient setup of this optimization problem; we will cover these details in the remainder of this paper.

4.4. Sensitivity analysis

In order to efficiently solve the optimization problem in Eq. (4.15) using a gradient based algorithm, cost effective and accurate computation of sensitivities is critical. Fortunately, decoupling of control weights from the underlying geometric space provides the possibility of the derivation of these sensitivities analytically and in a very cost-effective fashion.

Differentiating the right-hand side of Eq. (4.14) with respect to an arbitrary design variable w_L^u yields the sensitivities of the objective function as

$$\frac{\partial\left(\eta_E^2\right)}{\partial w_L^u}=\sum_{K\in\psi}\left(\frac{\partial\left(\left\|r\right\|_{L^2(K)}^2\right)}{\partial w_L^u}h_K^2+\frac{\partial\left(\left\|R\right\|_{L^2(\partial K)}^2\right)}{\partial w_L^u}h_K\right) \tag{4.16}$$

where constant C is assumed to be unity. As observed, evaluation of the above expression requires finding the derivatives of the interior as well as boundary residual terms with respect to control weights. Using Eq. (4.11), the sensitivities of the interior residual term for a typical knot-element *K* becomes

$$\frac{\partial\left(\left\|r\right\|_{L^2(K)}^2\right)}{\partial w_L^u}=2\int_{\Omega_K} r\left(D\left(\frac{\partial^3 u^h}{\partial x^2\partial w_L^u}+\frac{\partial^3 u^h}{\partial y^2\partial w_L^u}\right)-\sigma\left(\frac{\partial u^h}{\partial w_L^u}\right)\right)\mathrm{d}\Omega \tag{4.17}$$

which includes the gradients of u^h as well as its first and second order spatial derivatives with respect to control weights. The derivation of these sensitivities is non-trivial and tedious. Hence, we have provided these derivations in Appendix B.

On the other hand, the sensitivities of the boundary residual term are obtained as

$$\frac{\partial(\left\|R\right\|_{L^2(\partial K)}^2)}{\partial w_L^u}=-2\int_{\Gamma_N} R\left(\frac{\partial^2 u^h}{\partial x\partial w_L^u},\frac{\partial^2 u^h}{\partial y\partial w_L^u}\right).\mathbf{n}\,\mathrm{d}\Gamma \tag{4.18}$$

Similarly, the detailed derivation of the required derivatives in Eq. (4.18) is provided in Appendix C. It is important to note that due to decoupling of control weights from the underlying geometry, the derivatives of h_K as well as $d\Omega$ and $d\Gamma$ with respect to design variables vanish. Moreover, this substantially simplifies sensitivity derivation of the spatial derivatives of the field variable in above equations. These simplifications result in substantial reduction of the sensitivity analysis cost.

4.5. Driving the adaptivity process with exact error

To verify the performance and effectiveness of the proposed *w*-adaptivity process, we will also run experiments on problems with existing closed-form solutions in which case the exact error can be calculated using Eq. (4.13) and be used instead of the estimated error $\eta_E(\chi)$ in Eq. (4.15) for driving the adaptivity process. The derivation of analytical sensitivities in this case is more straightforward as the sensitivities of the exact solution in Eq. (4.13) with respect to design variables vanish.

4.6. Treatment of boundary conditions

One of the subtleties for proper implementation of adaptive *w*-refinement is appropriate treatment of boundary conditions. We remind that since the control weights do not lie in the function space of the geometry, we can freely manipulate these *boundary control weights* for improved accuracy. In fact, this is critical for obtaining monotone distribution of error throughout the computational domain and subsequently achieving optimal convergence rates. We discuss the treatment of Neumann (natural) and Dirichlet (essential) conditions separately here.

4.6.1. Natural conditions

Treatment of natural boundary conditions is more straightforward as they naturally arise in the variational formulation. Therefore, one can freely include the boundary control weights associated with Neumann boundaries in the vector of design variables when setting up the optimization problem (4.15). The only important point here is that, at each iteration of the optimization process followed by updating the design variables, the force vector associated with these (non-zero) boundary conditions need to be reevaluated with the new set of boundary control weights using Eq. (4.10).

4.6.2. Essential conditions

On the other hand, the imposition of essential conditions during *w*-adaptivity is more intricate and needs precise attention. The strategy depends on whether these conditions are homogeneous or non-homogeneous. The homogeneous (or constant) Dirichlet conditions can be satisfied exactly by simple setting the respective boundary control variables to zero (or the given constant), irrespective of the values of their control weights. Note that in this case, variation of the corresponding boundary control weights does not affect the exact satisfaction of these conditions. Therefore, similar to natural boundary conditions, these boundary control weights can be freely included as design variables in Eq. (4.15), and their optimal values will be determined by the adaptivity algorithm.

In contrast, finding the optimal boundary control weights of non-homogeneous essential conditions needs special treatment. We remind here that due to the non-interpolatory behavior of spline basis functions, the imposition of non-homogeneous essential boundary conditions in IGA is, in general, non-trivial.

It has been found that direct imposition of these boundary conditions to control variables may lead to significant error and non-optimal rate of convergence [73–75]. Several strategies have been proposed for improved treatment of these conditions. These methods can be classified into two main types: ‘strong’ imposition by approximating the boundary profile in the NURBS space, and ‘weak’ imposition via variational methods.

Most common techniques for weakly imposition of these boundary conditions are Lagrange multiplier methods [72], Nitsche method [74,75,90,91], and penalty method [92]. On the other hand, strong approximation of boundary conditions includes least square fitting [93], collocation and transformation methods [73], quasi interpolation techniques [94], and coupling with Lagrange shape functions [95]. We refer to [92] for a rigorous review on these methods.

One can possibly generalize any of the above techniques based on GNURBS and incorporate it in adaptive *w*-refinement. In the following section, we develop such an extension for least-square fitting method and elaborate how this technique could be properly incorporated in the proposed adaptive framework.

4.6.2.1. Least-square minimization using GNURBS

Suppose a Dirichlet condition $u_D(x,y)$ over an arbitrary boundary of the domain Γ_D in Eq. (4.1) is specified. The problem can be simply posed as a curve fitting problem where a given height function u_D needs to be approximated with u_D^h defined in Eq. (4.19) over a fixed planar NURBS curve as in Eq. (4.20)

$$u_D(\rho) \approx u_D^h(\rho) = \sum_{i=0}^{n} R_{i,p}^{u}(\rho) u_i, \quad 0 \le \rho \le 1 \tag{4.19}$$

$$\mathbf{C}(\rho) = \begin{Bmatrix} x(\rho) \\ y(\rho) \end{Bmatrix} = \sum_{i=0}^{n} R_{i,p}^{xy}(\rho) \begin{Bmatrix} x_i \\ y_i \end{Bmatrix}, \quad 0 \le \rho \le 1 \tag{4.20}$$

where $\rho \in \{\xi, \eta\}$. This can be easily posed as a least-square approximation problem leading to optimal accuracy in *L^2-norm*. Assuming $\{\rho_s \to (x_s, y_s, u_{D,s}) : s \in \mathcal{S}\}$ is the set of N_s collocation points, the error function *E* to be minimized is defined as

$$E(\boldsymbol{\lambda}) = \frac{1}{2} \sum_{s \in \mathcal{S}} \left\| u_D^h(\rho_s) - u_D(\rho_s) \right\|^2 = \frac{1}{2} \sum_{s \in \mathcal{S}} \left\| \sum_{L \in \mathcal{L}^s} R_L(\rho_s) u_L - u_{D,s} \right\|^2 \tag{4.21}$$

where ρ_s are the corresponding collocation points in the parametric space, $\mathcal{L}^s$ is the set of indices of non-zero basis functions at ρ_s, and $u_{D,s} = u_D(\rho_s)$. Unlike classic NURBS, the vector of design variables $\boldsymbol{\lambda}$ here includes both boundary control variables as well as boundary control weights, i.e. $\boldsymbol{\lambda} = \{u_0, ..., u_n, w_0^u, ..., w_n^u\}$, where similar bounding-box constraints on control weights as in Eq. (4.15) are to be satisfied. With this set of design variables, Eq. (4.21) becomes a constrained non-linear least-square problem which can be solved using any of the existing solvers such as trust-region-reflective available in MATLAB. Alternatively, to avoid solving a non-linear problem, one can employ the two-step linear algorithm developed in Chapter 2 via the extension of the original algorithm for NURBS approximation by Ma [11,21]. As previously discussed, this algorithm leads to two separate linear systems of equations; a homogenous system which yields the optimal control weights and a non-homogenous one that yields the corresponding optimal control variables. As the reported numerical results in Chapter 2 suggest, by including the control weights, in both cases of smooth and rapidly varying functions, a dramatic improvement in the accuracy can be achieved.

Another significant advantage of including the boundary control weights as additional design variables for the approximation of Dirichlet conditions is the possibility of exact satisfaction of these conditions for a wider range of functions compared to NURBS. In particular, any Dirichlet conditions specified as a function of the form

$$u_D(x,y) = \frac{f(x,y)}{g(x,y)} \tag{4.22}$$

where f and g are arbitrary polynomials, can be exactly represented via solving Eq. (4.21) by appropriate choice of degrees of basis functions (p, q). We will provide a numerical example later and clarify why making direct use of NURBS-based IGA does not naturally allow for the exact imposition of the entire functions in this class.

4.6.2.2. Incorporation in adaptive w-refinement

We provide here further details on how to impose the obtained optimal boundary conditions from previous section on the problem. The process of imposing these conditions on Eq. (4.8) is quite similar to other classic ways of strongly imposing the Dirichlet conditions, apart from the subtle fact that, in the current case, one should first solve the problem in Eq. (4.21) and feed the obtained optimal boundary control weights to the basis functions used for defining the field variable function space in Eq. (4.5) prior to the construction of stiffness matrix in Eq. (4.8).

Moreover, strictly speaking, these boundary control weights (associated with non-homogeneous essential boundary) must be excluded from the design variables in Eq. (4.15). We emphasize here that, while using any of the *strong* methods, discussed in Section 4.6.2, for the imposition of non-homogeneous essential conditions, including these boundary control weights in Eq. (4.15) as design variables will result in erroneous results.

Finally, it needs to be mentioned that merely imposing the optimal boundary conditions obtained by the above-discussed algorithm before executing *w*-adaptivity may not result in improved accuracy, or may even deteriorate the solution compared to imposing the Dirichlet conditions using classic linear least-square fitting. This could be attributed to the fact that these boundary basis functions are shared with the interior ones; hence, while improving the accuracy on the boundary (e.g. along ξ), manipulating the control weights on the boundary may deteriorate the natural

balance between basis functions inward the domain (along η). However, this issue will be fully resolved after performing *w*-adaptivity.

4.7. Numerical integrations

Another significant aspect for effective implementation of *w*-refinement which should be carefully addressed is the employed quadrature rule. It is important to note that in order for *w*-adaptivity algorithm to perform effectively and lead to optimal solution, all the numerical integrations need to be performed with adequate accuracy. This includes integration of the stiffness matrix in (4.9), the load vector in (4.10), the estimated error in (4.14), as well as the sensitivity expressions in (4.17) and (4.18).

Several studies have been carried out for the development of efficient quadrature rules in tensor product splines; see e.g. [96–101]. Nevertheless, to our knowledge, all these studies ignore the variation of the weight function in the denominator and devise quadrature rules which satisfy the exactness condition for the non-rational basis. The argument is that "*often the weight appearing in the denominator of the NURBS basis functions changes slowly (compared to the polynomial numerator of the NURBS basis functions) because they are piecewise smooth functions on the initial coarse mesh where the geometry is exactly represented. Then, it is a common practice to select quadrature rules that give exact integration when the NURBS denominator is constant*." [101].

Nonetheless, as will be seen later in numerical results, this assumption will not hold true in *w*-adaptive IGA, especially when the adaptivity is applied to problems with sharply varying solutions. In the current work, we will simply use a finer quadrature globally in such cases, to make sure all the integrations are calculated with adequate accuracy. Although this does not seem to be an efficient way for addressing this issue, we emphasize here that devising an efficient quadrature rule for these highly rational tensor product splines has not been a concern of us here and is beyond the scope of this study. This is in fact an interesting subject for further research in *w*-adaptive IGA.

On the other hand, another observation reported by Hughes et al. [3] is consistent with our experiments with *w*-adaptive IGA. To assess the validity of assuming NURBS as B-splines of the same polynomial order for deciding on the number of quadrature points, they perform tests in which they systematically increase the number of quadrature points. They report that "*for*

sufficiently fine meshes no differences in results were discernible. However, coarse meshes required more integration points due to large variations in the geometrical mapping. More research needs to be done to determine a robust strategy covering all situations." [3]

Similarly, our numerical experiments with *w*-adaptive IGA suggest that more quadrature points are required to achieve the same level of accuracy in integration when coarser meshes are used. However, in our case, this is mainly caused by large variations in the denominator of solution basis, that is

$$W^u(\xi,\eta) = \sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta) w_{kl}^u \tag{4.23}$$

rather than in geometrical mapping. Therefore, even in cases where the geometrical mapping is constant, a larger number of quadrature points will be required when *w*-adaptivity is performed on coarse meshes. The details of selection of quadrature rule will be included for all the numerical studies in this thesis.

4.8. Computer implementation aspects

The flowchart in Figure 4.1 summarizes the process of adaptive *w*-refinement. The implementation of this procedure is quite straightforward and can be easily included in an existing IGA package. Certain details, however, should be considered for the efficient implementation of this algorithm. For instance, at the initialization step, one can evaluate and pre-store all the entities constructed using the basis functions of the geometry as these entities are invariant during the optimization process. Further, if non-homogeneous boundary conditions exist, they need to be evaluated only once at this stage. On the other hand, the solution basis functions as well as all their resultants, such as natural boundary conditions, change at each iteration and need to be evaluated iteratively prior to re-estimation of the error. Finally, as shown in Figure 4.1, we have included a particular step for the modification of quadrature followed by updating the control weights in the flowchart of adaptivity for the sake of completeness, even though as discussed in previous section, it has not been implemented in this work and will be studied in a future research.

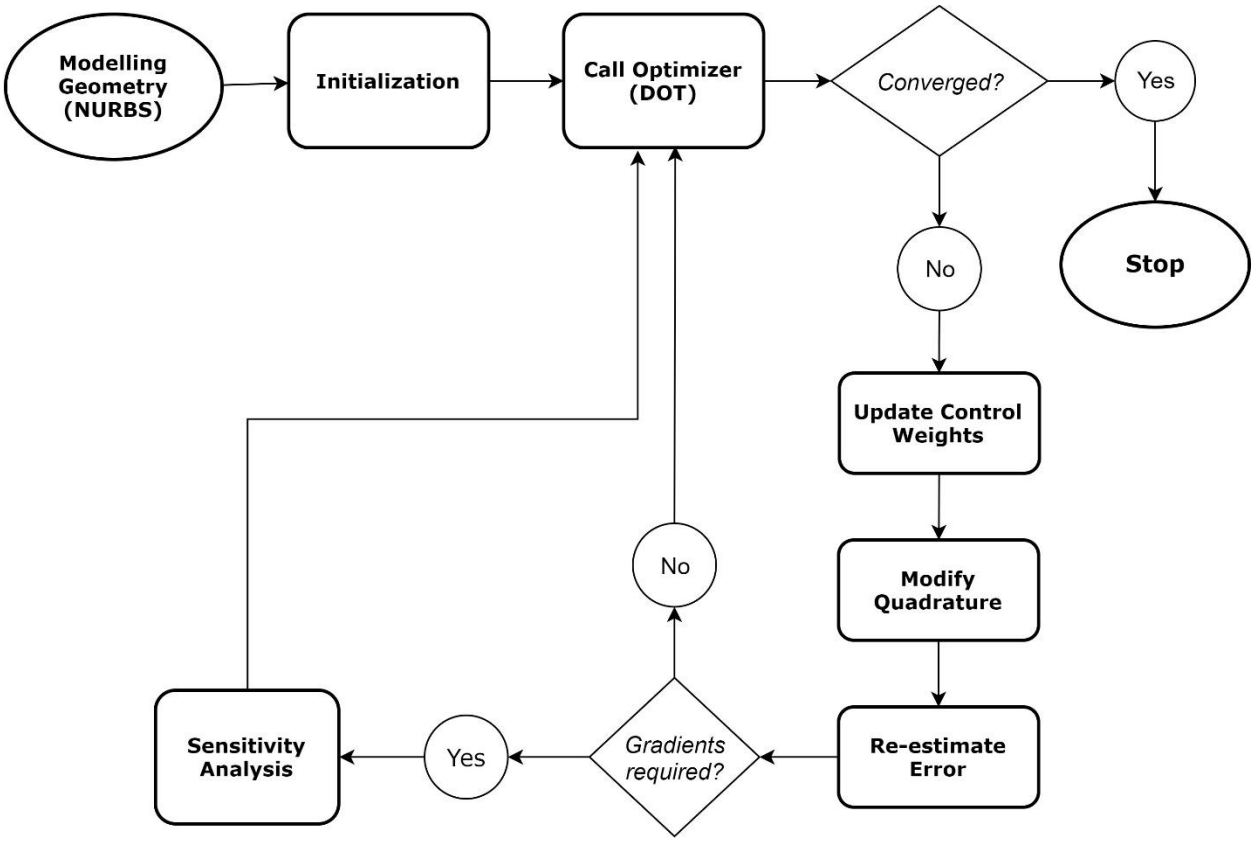


Figure 4.1. The procedure of adaptive *w*-refinement.

4.9. Analogy with other refinement techniques

Recalling the definition of function space refinement techniques in Chapter 1, we can see that the proposed method conforms to this definition as it adaptively enriches the function space and improves the accuracy without perturbing the underlying geometry or its parameterization. However, unlike *h* and *p* refinements, this procedure preserves the number of basis functions and only changes their contribution to the function space by adjusting their weights. It is important to mention that since the solution space is always simply discretized by NURBS basis functions, during this adaptive procedure, all the properties of basis functions such as their linear independence and partition of unity are naturally preserved. The proof of linear independence of NURBS basis functions can be found in [102]. Further, since no additional basis functions are introduced, the structure of stiffness matrix remains unchanged in terms of sparsity and bandwidth. The condition number, however, will change and needs to be monitored. In the following numerical experiments, we will study the effect of performing *w*-adaptivity on the conditioning of the stiffness matrix in different types of problems.

Another important aspect of this adaptive method is that, unlike many of existing methods for addressing problems with irregularities such as DPG [103,104], anisotropic NURBS approximation [105] etc., it does not require any prior knowledge of the solution behavior. The proposed method is in some sense quite similar to *r*-refinement as both methods attempt to minimize an estimation of the error by solving an optimization problem. However, unlike *r*-refinement, the design variables in proposed method are transferred to the solution space. This change of variable eliminates all the deficiencies of *r*-refinement discussed in Chapter 1 and makes it a competitive algorithm with existing adaptivity techniques.

4.10. Limitations

It is clear that performing *w*-refinement provides a trade-off between the achieved improvement in accuracy and the extra computational cost for solving the adaptivity problem illustrated in Figure 4.1. Of course, the algorithm is fruitful when this trade-off is reasonable, i.e. when the obtained accuracy dominates the additional computational cost. We note here that this trade-off depends on multiple factors such as the type of governing PDE, the behavior of the solution (number and orientation of existing layers, in particular), the accuracy of an initial guess for control weights, the employed degree of basis functions, the efficiency of the employed non-linear optimization solver etc. Detailed investigation of all these factors is beyond the scope of this study. However, considering the flowchart of *w*-refinement in Figure 4.1, we can identify the main sources of the additional computational cost as discussed below.

According to our numerical studies, the primary source of this additional cost is due to re-solving the governing PDE at each iteration followed by updating the control weights. Another major portion of the computational expense is due to re-evaluation of the objective function, i.e. estimated error in Eq. (4.14), as well as its sensitivities in Eqs. (4.16)-(4.18), repeatedly. We recall that the calculation of these expressions requires evaluating higher order derivatives of rational basis functions which are more complicated and expensive compared to polynomial basis functions. In the following numerical studies, we will show how these components of the algorithm contribute to the overall computational expense.

Consequently, we can see that the cost of adaptive *w*-refinement directly depends on the number of optimization iterations, where the cost of each iteration mainly relies on the number of quadrature points as well as the employed number of design variables. Therefore, the key factors

for improving efficiency are to reduce the number of optimization iterations, the number of quadrature points, as well as the number of considered design variables. We will later suggest some preliminary ideas for improving these factors in Chapter 5.

4.11. Numerical Experiments

To demonstrate the performance of the proposed adaptivity technique, in this section, we apply this method to a variety of problems with different solution behaviors and compare the obtained results with those of NURBS-based IGA.

The presented numerical results are obtained by the implementation of this adaptive algorithm in PGI Visual FORTRAN [106]. The conjugate gradient method with Incomplete Lower-Upper (ILU) preconditioner has been used for solving the system of equations. All the reported condition numbers of the stiffness matrix are measured in L^1-norm. The optimization problem is solved by the BFGS method available in the Design Optimization Tool (DOT) [107]. The initial guess, bounding constraints on design variables, and termination criteria are selected as follows.

Initial guess: It is well-known that in non-linear optimization problems, starting with a good initial guess can significantly improve the performance and efficiency of the algorithm. Not only can this make the algorithm converge faster, but also it increases the possibility of achieving the optimal results. Finding a suitable initial guess of control weights for a steady transport problem which is studied here, however, is non-trivial. In the following numerical experiments, unless stated otherwise, we will start the optimization process assuming unit values for all design variables, i.e. B-spline basis functions. Nevertheless, in certain cases, we will suggest effective ideas which can be used to obtain an improved initial guess.

Bounding constraints: Theoretically, selecting an infinitesimal positive value for $w_{\min}$ and an arbitrary larger value for $w_{\max}$ in Eq. (4.15) will cover the whole search space. However, these values affect the performance of the algorithm. In particular, selecting too small values for $w_{\min}$, if taken by design variables during adaptivity, will result in deterioration of the conditioning of the stiffness matrix which is not desired. Our experiments show that with the following suggested bounds on design variables, the algorithm works effectively

$$10^{-4} \le w_L^u \le 3.0,\ L = 1,..., N \tag{4.24}$$

Further research, however, is needed for finding the optimal bounds in different types of problems.

Termination criteria: As illustrated in Figure 4.1, the optimization process is terminated whenever a convergence criterion is satisfied. The employed design optimization tool (DOT) uses several criteria to make the decision when to stop. These include: if a maximum number of 100 iterations is reached; Kuhn–Tucker conditions are satisfied 'reasonably', defined as when all components of the gradient of objective are less than 1*e*-3; and the so-called diminishing returns criterion when either the relative or absolute change in the objective between two consecutive iterations is less than the specified tolerance 1*e*-6. Further details on these criteria can be found in [107].

4.11.1. Test Case 1- Poisson equation with a smooth solution

In the first numerical experiment, we investigate the performance of *w*-adaptivity on problems with smooth solutions. For this purpose, we consider the following governing Poisson equation with homogeneous boundary conditions on all edges

$$\begin{aligned} &-\Delta u = 2\pi^2 \sin(\pi x)\sin(\pi y), \qquad (x,y)\in[0,1]\otimes[0,1] \\ &u_D(x,0) = u_D(x,1) = u_D(0,y) = u_D(1,y) = 0 \end{aligned} \tag{4.25}$$

whose closed-form solution is given by:

$$u(x,y) = \sin(\pi x)\sin(\pi y) \tag{4.26}$$

which is illustrated in Figure 4.2.

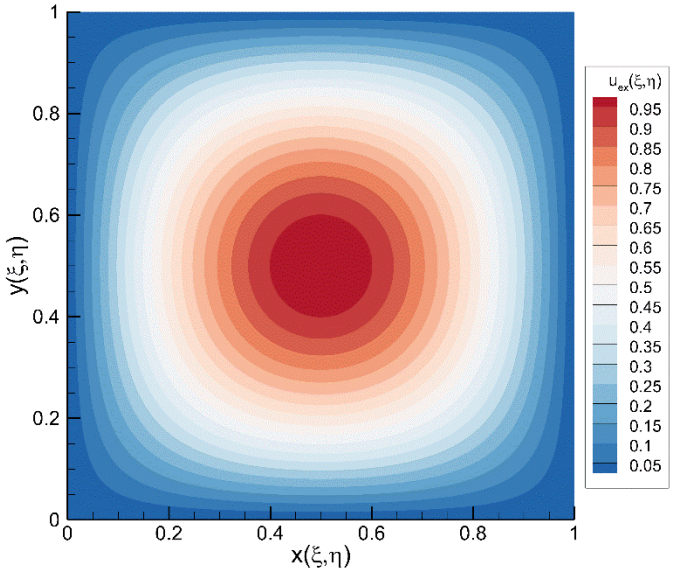
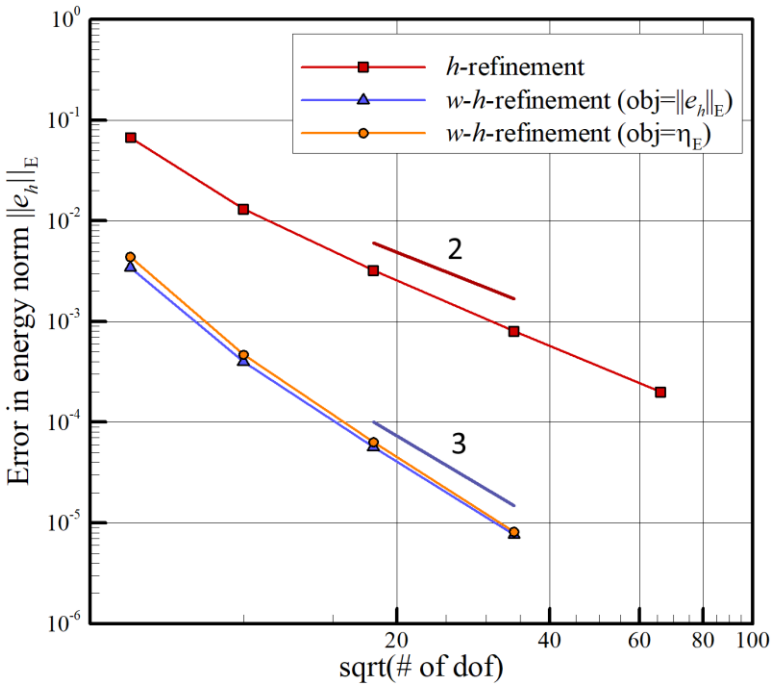
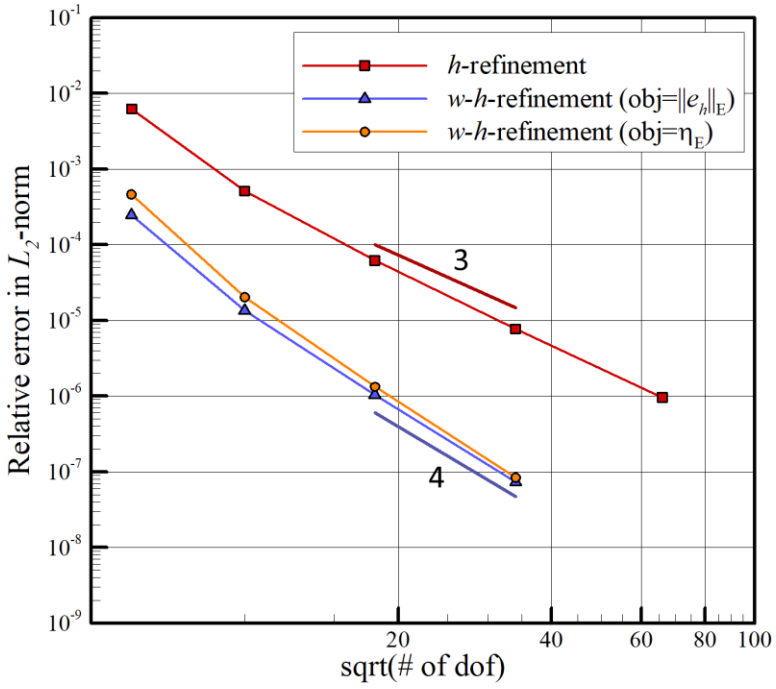


Figure 4.2. Exact solution of the Poisson equation in (4.26).

We perform a convergence study by starting with a mesh of 4×4 bi-quadratic elements with linear parameterization, and refining up to 64×64 elements. At each level of discretization, followed by *h*-refinement, we perform *w*-adaptivity and refer to this procedure as *w*-*h*-refinement. In all cases, we use a set of 4×4 quadrature points per element for integration. Further, to examine the reliability of the employed residual based a posteriori error estimator, we drive the adaptivity process with both the exact error in Eq. (4.13), as well as the estimator in Eq. (4.14). The obtained results are represented in Figure 4.3.



(a)



(b)

Figure 4.3. Convergence rates of *h*-refinement versus *w*-*h*-refinement in (a) energy norm, and (b) relative L^2-norm.

It is interesting to notice that, followed by performing *w*-adaptivity, the convergence rates have been improved by one order in both energy as well as L^2 norms, indefinitely. We note here that these are representative results of our studies with different degrees of basis functions on a variety of second order elliptic PDEs with a smooth solution. Our numerical results suggest the optimal rate of convergence of NURBS basis with optimal weights are $\mathcal{O}(p+2)$ in L^2 and $\mathcal{O}(p+1)$ in energy norm for these problems.

Moreover, the figure shows a good agreement between the results obtained with guiding the adaptivity process by the exact error as well as the estimated error. As can be seen, this agreement improves as the resolution of the mesh increases. This is expected since the performance of estimator is directly related to the accuracy of solving the PDE. That is, the more accurate the PDE is solved, a better estimation of the error is provided by the estimator.

The history of adaptivity process guided by the estimator is depicted in Figure 4.4 for the 4×4 mesh. The plotted results are obtained by assuming the unknown constant $C=1$ in Eq. (4.14). It is interesting to notice that by decreasing the objective function (η_E) at each iteration, the exact error $\|e_h\|_E$ also diminishes by a quite similar rate, which implies the reliability of the employed

residual based a posteriori estimator. Note that the offset between the estimated and exact error is unimportant and is caused by the assumption made for the unknown constant *C*.

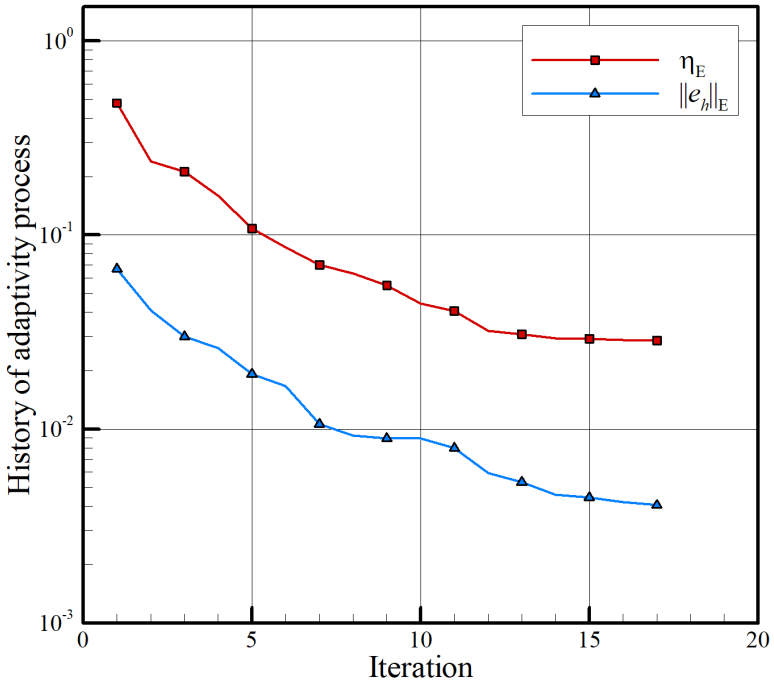
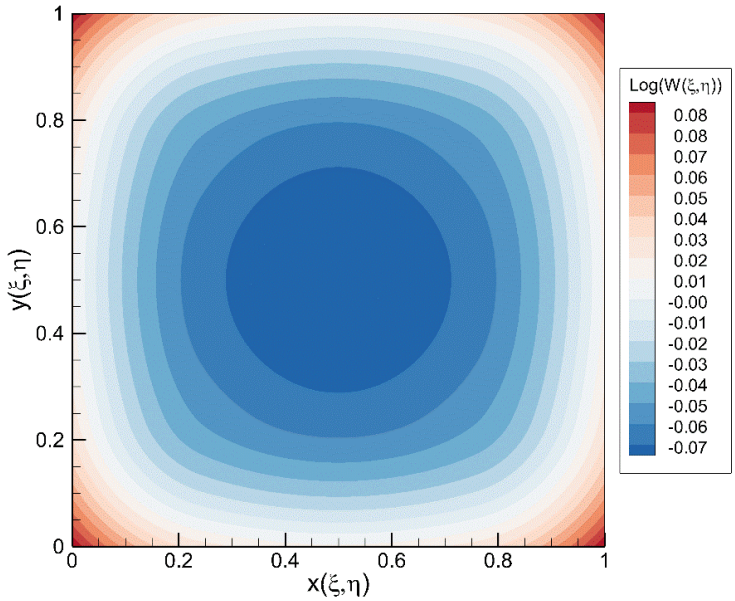
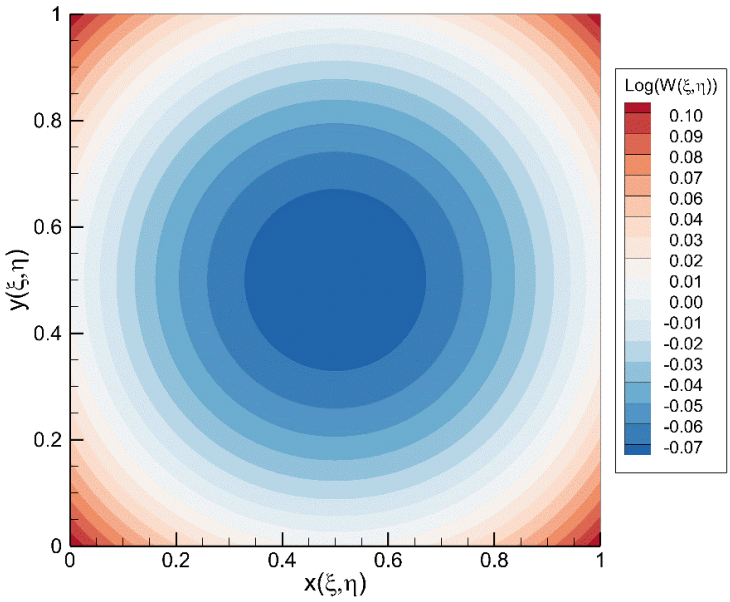


Figure 4.4. History of adaptivity process on the 4*4 mesh.

Furthermore, the optimal variation of the denominator of solution $W^u(\xi,\eta)$ is depicted in Figure 4.5(a) and (b), for the 4×4 and 32×32 meshes, respectively. The adaptivity process is driven by the estimator in both cases.



(a)



(b)

Figure 4.5. Optimal variation of $W^{u}(\xi, \eta)$ after performing *w*-adaptivity on the (a) 4*4 mesh, and (b) 32*32 mesh.

Comparing these figures, we can see that the optimal variation of denominator is converging towards a perfectly circular distribution, suggesting the possibility of existence of a closed-form expression. Moreover, we can see that in both cases the variation of the denominator is very small. This is expected to be always the case as long as the solution of the PDE is smooth.

The condition numbers of the stiffness matrix before and after adaptivity are presented in Table 4.1 for different mesh resolutions. As the table shows, no noticeable change in the condition number of the system of equations for any of the presented cases has occurred followed by performing *w*-adaptivity.

Table 4.1. Condition number of the stiffness matrix for different meshes.

Mesh	IGA	*w*-adaptive IGA	$\frac{w\text{-IGA}}{\text{IGA}}$
4×4	6.07E+00	6.48E+00	1.1
32×32	1.86E+02	1.86E+02	1.0

Finally, the relative amount of computational time consumed by different steps of the algorithm throughout the adaptivity process are shown in Table 4.2. The reported numbers are an average of the times calculated for different mesh resolutions.

Table 4.2. Relative computational times of different steps of *w*-refinement.

Basis	Objective & Sensitivities	Analysis	Other
30%	20%	45%	5%

According to this table, the largest computational time is consumed by the analysis step (construction of the stiffness matrix and solving the system of equations). The second time-consuming part is the evaluation of the basis functions together with their higher order spatial derivatives. Sensitivity analysis and evaluation of the objective function are also the next major time-consuming component of the algorithm.

4.11.2. Test Case 2- Reaction-diffusion equation with a rough solution

In this example, we examine the effectiveness of the proposed adaptive method on a problem with a rough solution. To this end, we consider a singularly perturbed reaction-diffusion equation with homogeneous essential boundary conditions on all edges

$$\begin{aligned} -\varepsilon^2 \Delta u + u &= f, \qquad (x,y) \in [0,1] \otimes [0,1] \\ u_D(x,0) = u_D(x,1) &= u_D(0,y) = u_D(1,y) = 0 \end{aligned} \tag{4.27}$$

where ε is the diffusion parameter, and f is determined by the exact solution

$$u(x,y) = \sin(\pi y)(1 - e^{-x/\varepsilon})(1-x) \tag{4.28}$$

which is depicted in Figure 4.6 for $\varepsilon = 0.01$.

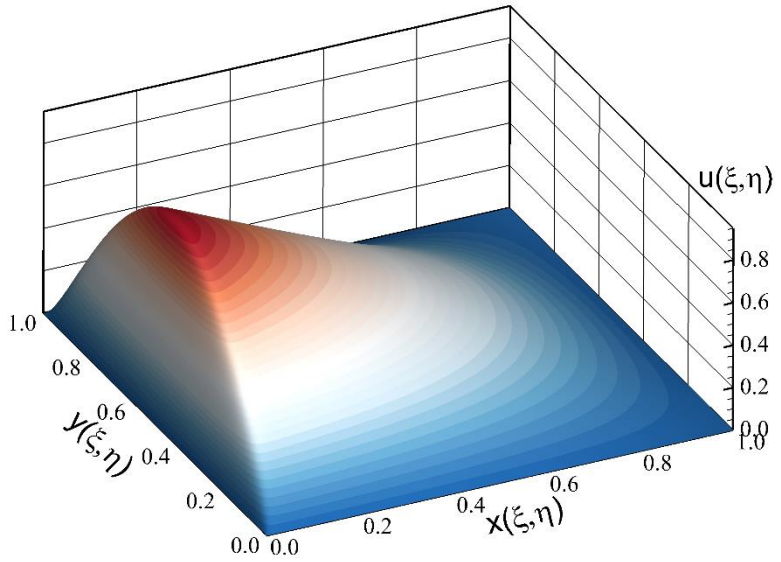
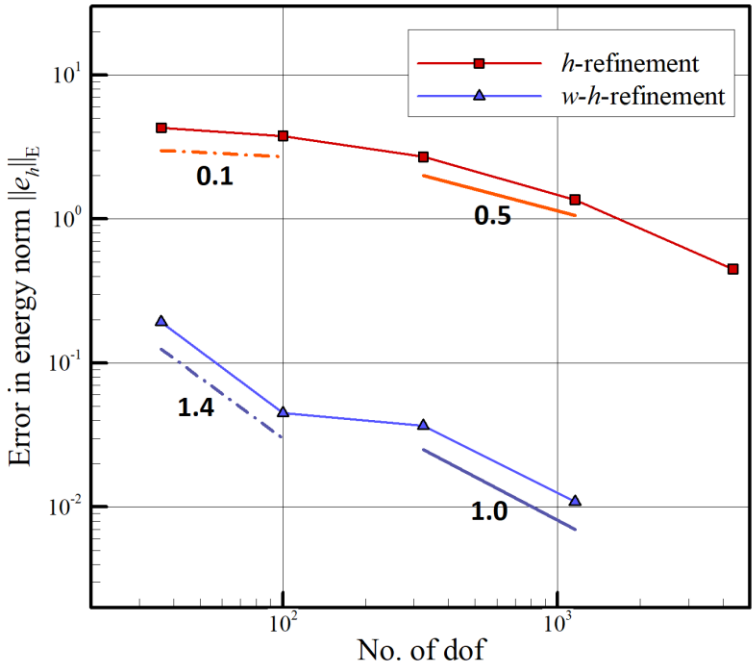


Figure 4.6. Exact solution of reaction-diffusion equation in Eq. (4.28) for ε = 0.01.

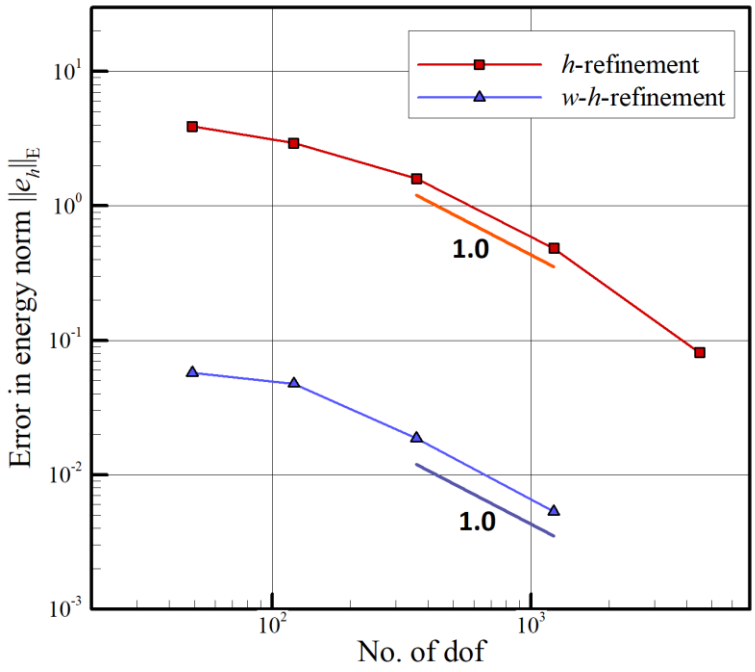
As the figure shows, the solution features a steep boundary layer of width ε near the left edge. Similar to the previous example, we study the convergence of *h* and *w*-*h*-refinement by performing *w*-adaptivity on different meshes of 4×4 to 32×32 elements. We employ sets of $15\times15, 12\times12, 10\times10$ and 8×8 quadrature points per knot-element for integration on the coarsest to finest mesh, respectively. Our experiments show no discernible change in results with using finer quadratures. However, as discussed earlier, a more systematic way is needed for deciding on the appropriate number of quadrature points.

In this example, we start the adaptivity process with an improved initial guess for control weights in the case of finer meshes, which is obtained by *h*-refining the optimal rational function space of the coarser mesh obtained by *w*-adaptivity. For this purpose, *h*-refinement is performed separately on the geometry and optimal solution function spaces of a coarser mesh. Our experiments indicate that this procedure leads to improved results and reduced number of iterations in problems with sharply varying solutions, such as the current example, where large variations in the denominator occur.

The obtained convergence rates are represented in Figure 4.7(a) and (b) for bi-quadratic and bi-cubic basis functions, respectively.



(a)



(b)

Figure 4.7. Convergence rates of *h*-refinement versus *w*-*h*-refinement in energy norm for (a) quadratic, and (b) cubic case.

As observed, in both cases a considerable improvement in the accuracy of approximation has been achieved. The improvement of the convergence rate, however, is not persistent and also differs for different degrees. Comparing Figure 4.7(a) and (b), we can see that the improvement in the convergence rate with quadratic basis is more evident, although this improvement is not persistent for all levels of refinement. It is worth noting that this non-uniform behavior of the convergence rate is common in other types of adaptive methods such as adaptive local *h*-refinement; see e.g. [34]. Further study is required to better perceive the effect of *w*-adaptivity on the convergence rate in problems with sharp layers. It is interesting to notice that in both cases, the obtained accuracy on a 4×4 mesh, after performing *w*-adaptivity, is better than that of a 64×64 mesh with B-spline functions. However, we reiterate that there is an additional computational cost for solving the optimization problem in *w*-adaptive IGA.

For better insight, the plots of approximated solution before and after adaptivity are depicted in Figure 4.8. Also, the corresponding distributions of error are represented in Figure 4.9.

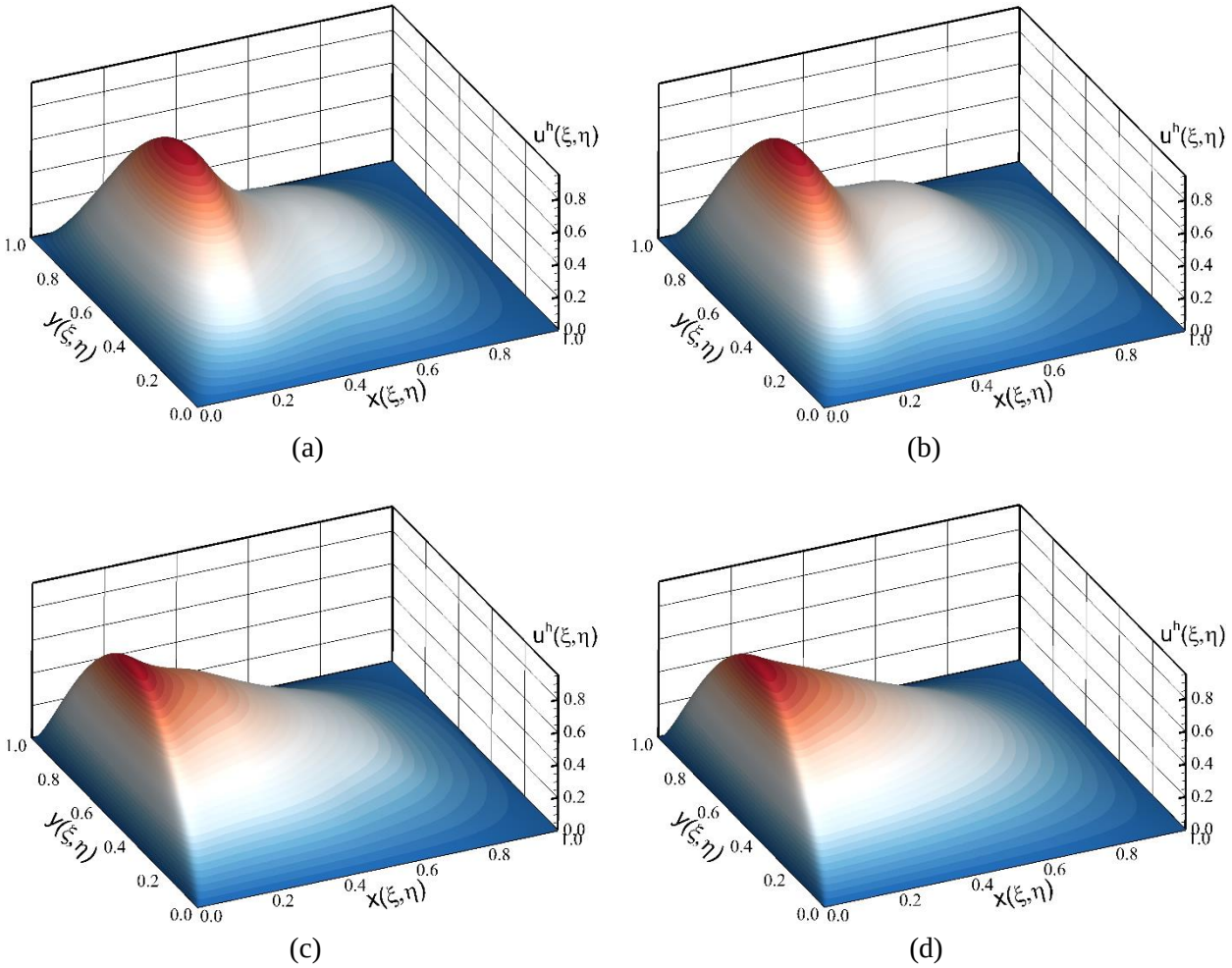
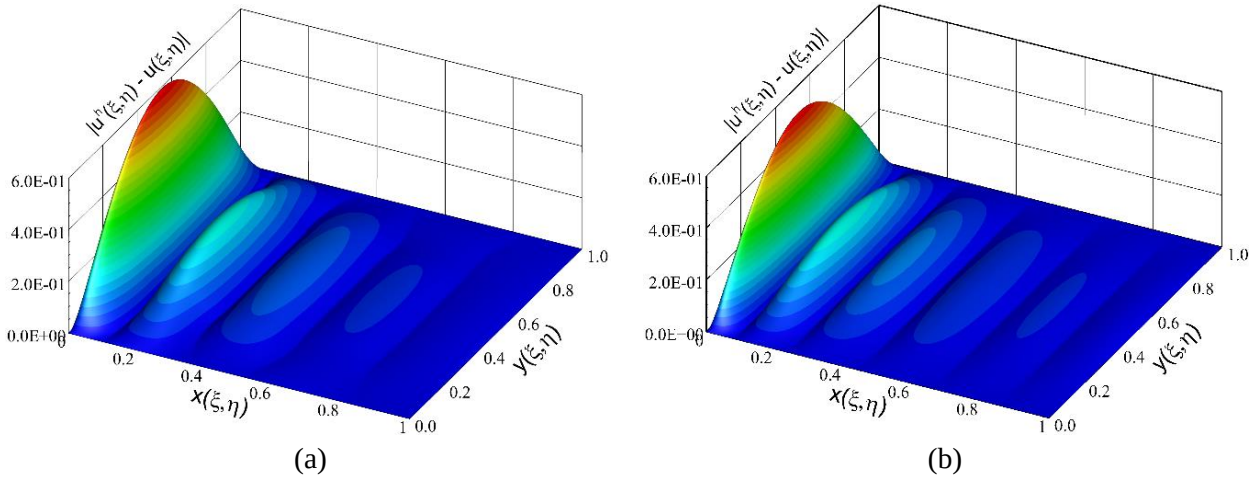


Figure 4.8. Approximate solution of reaction-diffusion problem with a 4*4 mesh using (a) quadratic IGA, (b) cubic IGA, (c) quadratic *w*-adaptive IGA, and (d) cubic *w*-adaptive IGA.



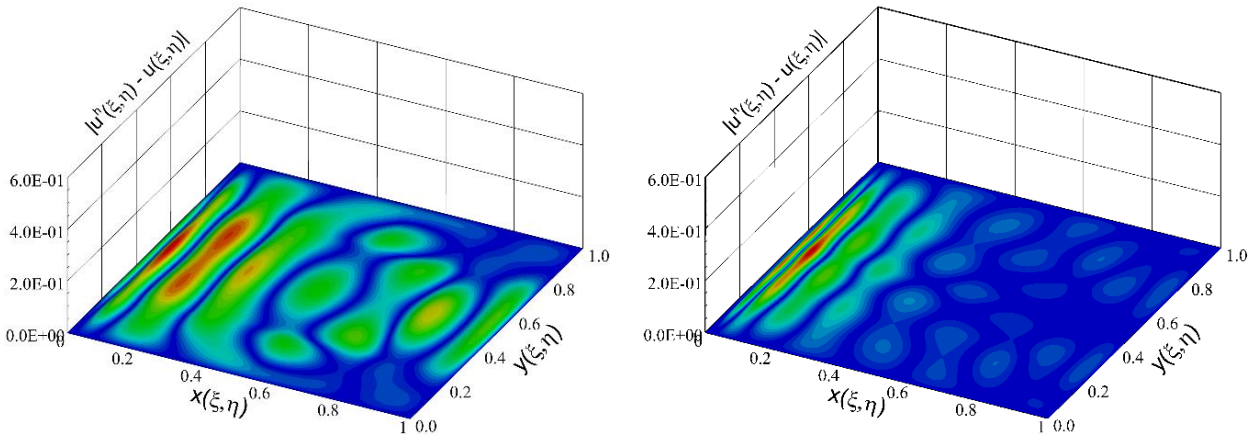
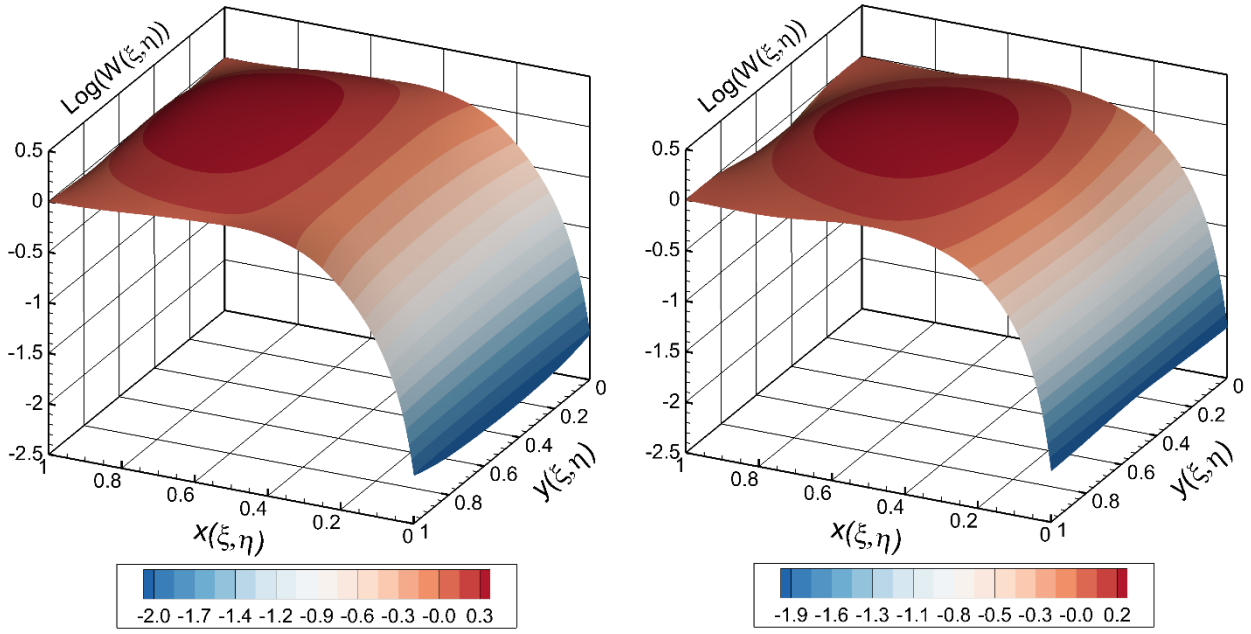


Figure 4.9. Error distribution of solving the reaction-diffusion problem with a 4*4 mesh using (a) quadratic IGA, (b) cubic IGA, (c) quadratic *w*-adaptive IGA, and (d) cubic *w*-adaptive IGA.

Having examined these figures, we can observe that after performing adaptivity, in both cases, the boundary layer has almost been completely resolved and a monotone distribution of error has been achieved.

The variations of denominator function $W^u(\xi,\eta)$ of the optimal solutions are demonstrated in Figure 4.10. It is interesting to notice that, unlike the previous example with a smooth solution, the variation of denominator in this case is very large. As the figure shows, the magnitude of this variation exceeds two orders of magnitude in the vicinity of the boundary layer. This experiment, in fact, reveals another significant aspect of using rational splines as a basis for analysis. While piecewise smooth polynomials are inherently poor for the approximation of fields with steep localized gradients, approximation with rational bases promises an effective tool for addressing this deficiency.



(a) (b)

Figure 4.10. Variation of solution denominator after *w*-adaptivity on the 4*4 mesh with (a) quadratic, and (b) cubic basis functions.

The condition numbers of the stiffness matrix before and after adaptivity are presented in Table 4.3 for different mesh resolutions. Comparing the condition numbers in Table 4.3 with those of the previous example in Table 4.1, we can observe moderate increase of the condition numbers in the current example.

Table 4.3. Condition number of the stiffness matrix for different meshes.

Mesh	Degree	IGA	*w*-adaptive IGA	$\frac{w\text{-IGA}}{\text{IGA}}$
4×4	2	5.65E+01	2.23E+02	3.9
	3	8.04E+02	1.79E+03	2.2
32×32	2	1.97E+01	3.14E+01	1.6
	3	1.77E+02	2.56E+02	1.4

Our experiments indicate that the increase of condition number is related to the variation of solution denominator, that is, larger variations of the denominator result in further increase of the condition number. Nevertheless, despite having orders of magnitude variation in the denominator of basis functions in the current case, according to Table 4.3, the condition numbers have not increased by any more than 4 times for any of the presented experiments. Finally, it is worth noting

that for more complex problems with meshes of practical scale, the increase of condition number can possibly be larger. Suitable preconditioners perhaps need to be developed and employed in such scenarios.

4.11.3. Test Case 3- Poisson equation with a closed-form solution in rational space

In this example, we attempt to reveal another crucial merit of *w*-adaptivity, which is its capability to solve problems whose exact solution lie in the rational space with machine precision, irrespective of how coarse the discretization is. Towards this goal, we consider the following Poisson equation over a quarter ring with Dirichlet conditions on all edges

$$\begin{aligned} -\Delta u &= f(r,\theta) \\ u &= u_D \quad \text{on}\,\Gamma_D \end{aligned} \tag{4.29}$$

where the source term f and boundary conditions u_D are specified by the closed-form solution (see Figure 4.11)

$$u(r,\theta) = \frac{\cos(\theta)}{r^2}. \tag{4.30}$$

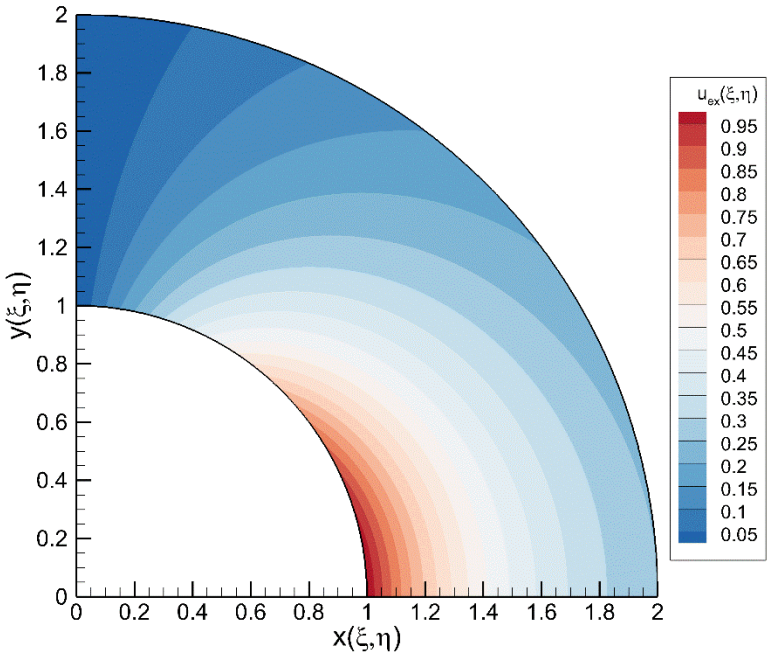


Figure 4.11. Exact solution of Poisson equation in Eq. (4.30).

The configuration of the problem is illustrated in Figure 4.12.

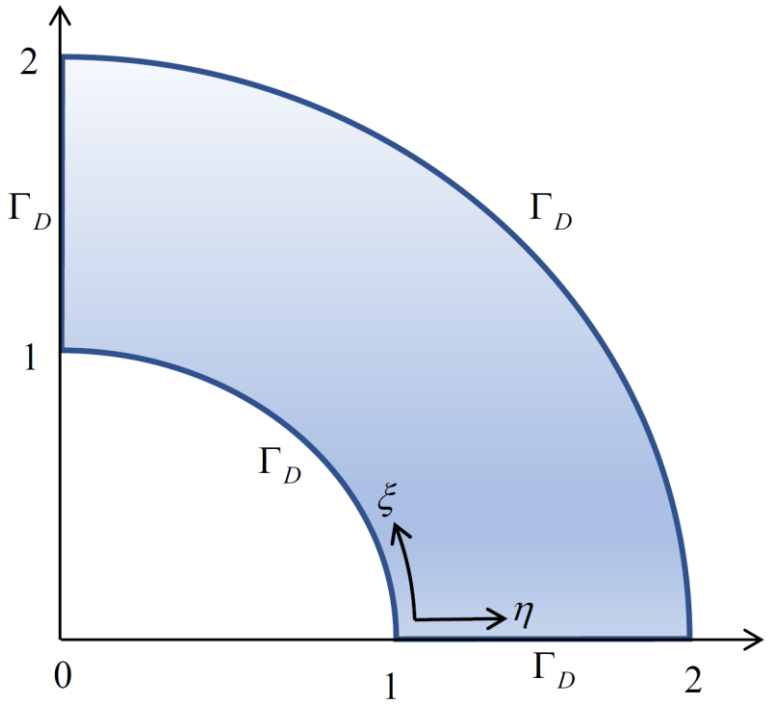


Figure 4.12. Configuration and boundary conditions of the quarter ring.

We construct a computational model with two bi-quadratic NURBS elements with normal parameterization as shown in Figure 4.13. The analytical values for the coordinates and weights of control points are provided in Table 4.4. Also, the knot-vectors are selected as $\boldsymbol{\Xi} = \{0,0,0,0.5,1,1,1\}$ and $\mathbf{H} = \{0,0,0,1,1,1\}$. It can be shown that the exact solution in Eq. (4.30) can be recovered by using the analytical values shown in Table 4.4 for control variables and control weights. According to this table, one can clearly see that the control weights required for recovering the exact solution in Eq. (4.30) have nothing to do with those of the underlying geometry. This is trivial as these control weights are determined by the governing PDE in Eq. (4.29).

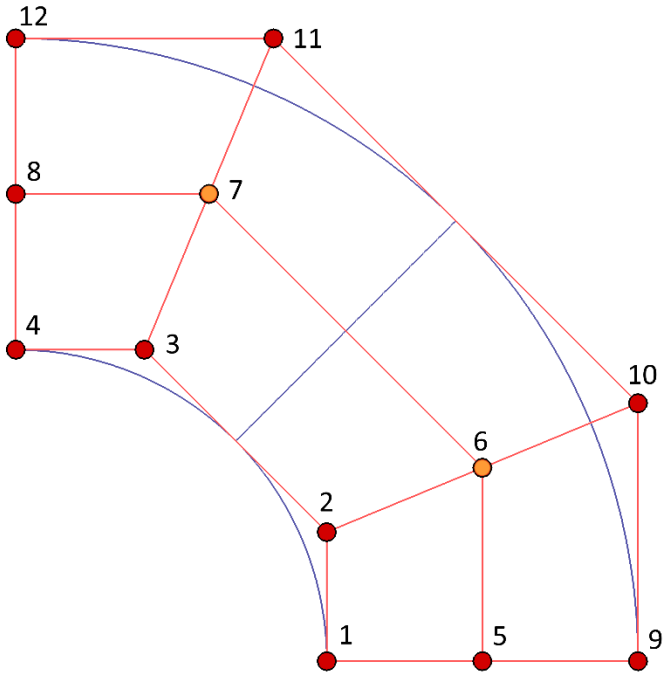


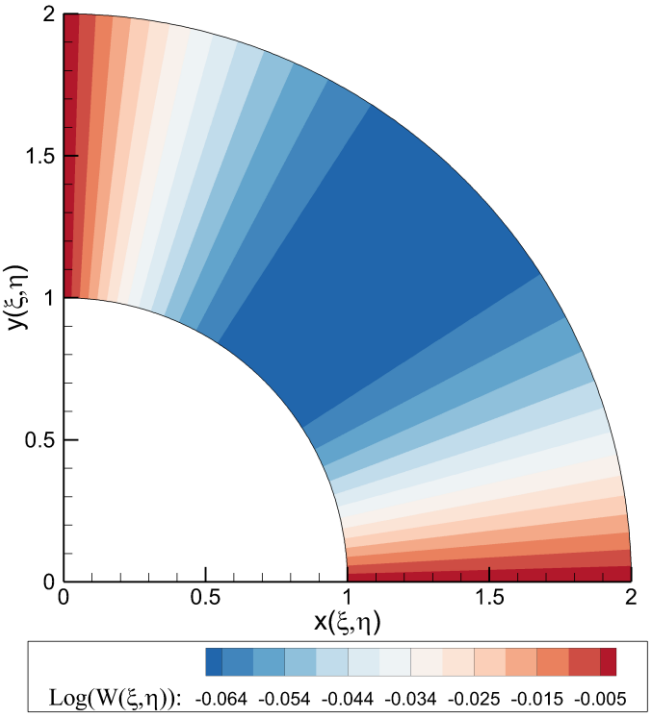
Figure 4.13. Control and physical mesh of the quarter ring with normal parameterization.

Table 4.4. Analytical values for exact modelling and solution of the problem.

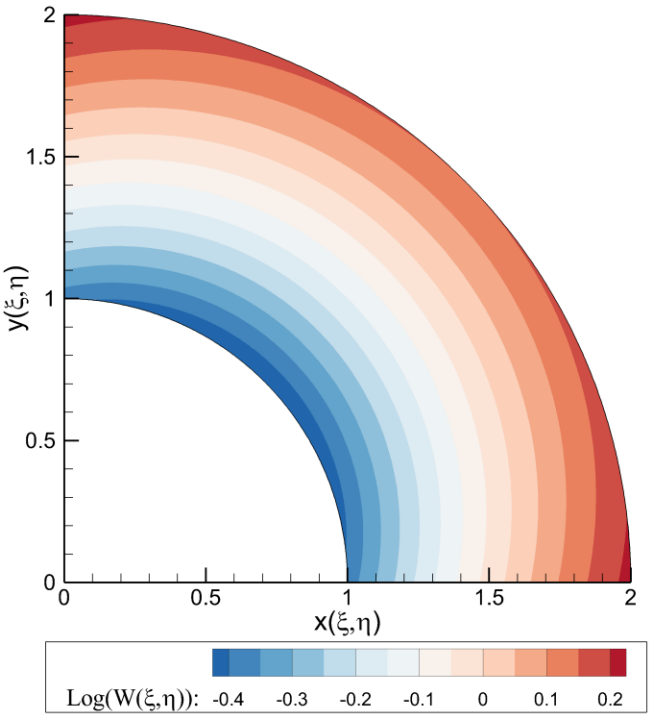
L	x_L	y_L	w_L^G	u_L	w_L^u
1	1.0	0.0	1.0	1.0	a
2	1.0	a	b^2	1.0	$0.5c$
3	a	1.0	b^2	a	$0.5c$
4	0.0	1.0	1.0	0.0	a
5	1.5	0.0	1.0	0.5	$2a$
6	1.5	$1.5a$	b^2	0.5	c
7	$1.5a$	1.5	b^2	$0.5a$	c
8	0.0	1.5	1.0	0.0	$2a$
9	2.0	0.0	1.0	0.25	$4a$
10	2.0	$2a$	b^2	0.25	$2c$
11	$2a$	2.0	b^2	$0.25a$	$2c$
12	0.0	2.0	1.0	0.0	$4a$

$$a = \tan(\frac{\pi}{8}), b = \cos(\frac{\pi}{8}), c = \cos(\frac{\pi}{4})$$

For better insight, the variation of denominator of the geometry as well as that of the field variable, obtained by the reported values in Table 4.4, are illustrated in Figure 4.14.



(a)



(b)

Figure 4.14. The variation of denominator of the (a) geometry, and (b) field variable.

Comparing these figures, we can see that there is no meaningful relationship between the variations of these two functions. Now, we attempt to solve the problem by using classic NURBS-based IGA, where the control weights are assigned identical values with the weights of the geometry, as well as the proposed *w*-adaptive IGA. To ensure that all integrations are calculated with machine precision, in both cases, we use a set of 15×15 quadrature points per knot-element.

In the case of IGA, we impose the essential boundary conditions by using classic least-square fitting. Note that the only remaining unknown control variables to be determined by analysis are (u_6, u_7). The obtained distribution of error of IGA solution is represented in Figure 4.15. As observed, the error is of order 10^{-2} over most regions of the domain. The energy norm of this error distribution is $\|e_h\|_E = 1.62e-1$. One may attempt to perform *h*-refinement to achieve a better accuracy with the expected optimal convergence rate of $\mathcal{O}(2)$ in energy norm, studied in test case 1.

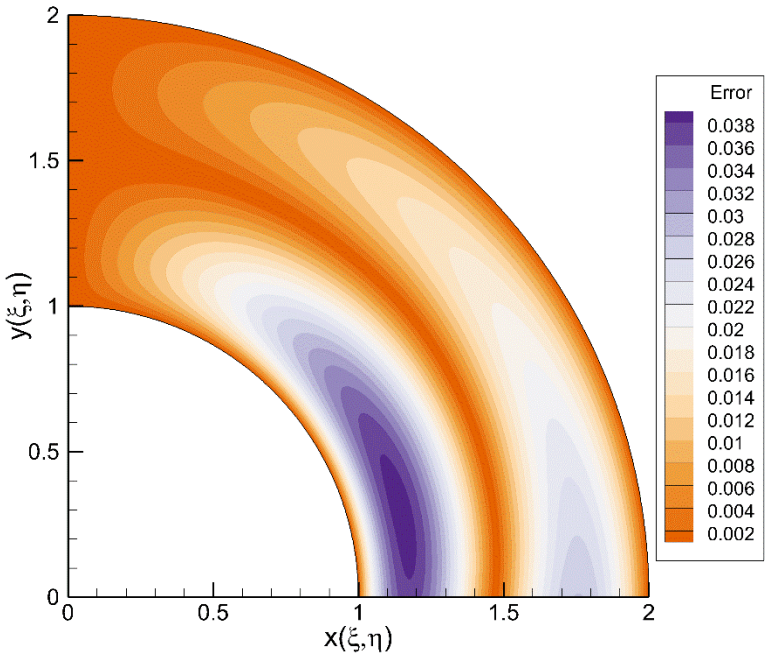


Figure 4.15. Error distribution of NURBS-based IGA solution.

Next, we study the performance of *w*-adaptivity for solving the same problem. In this case, the essential boundary conditions are imposed exactly by using the analytical values for boundary control variables and control weights, reported in Table 4.4, prior to performing the adaptivity. The only remaining unknowns to be determined by analysis here are (u_6, w_6^u) and (u_7, w_7^u). We

start the adaptivity process by the initial guess of assuming unit values for both unknown control weights, i.e. $\boldsymbol{\chi}_0 = \left\{ w_6^u, w_7^u \right\}_0 = \{1,1\}$. The initial distribution of error prior to performing the adaptivity is indicated in Figure 4.16. The energy norm of error in this case is $\|e_h\|_E = 1.05e-1$, which is slightly better than the result of previous case with IGA due to exact satisfaction of boundary conditions on all edges here.

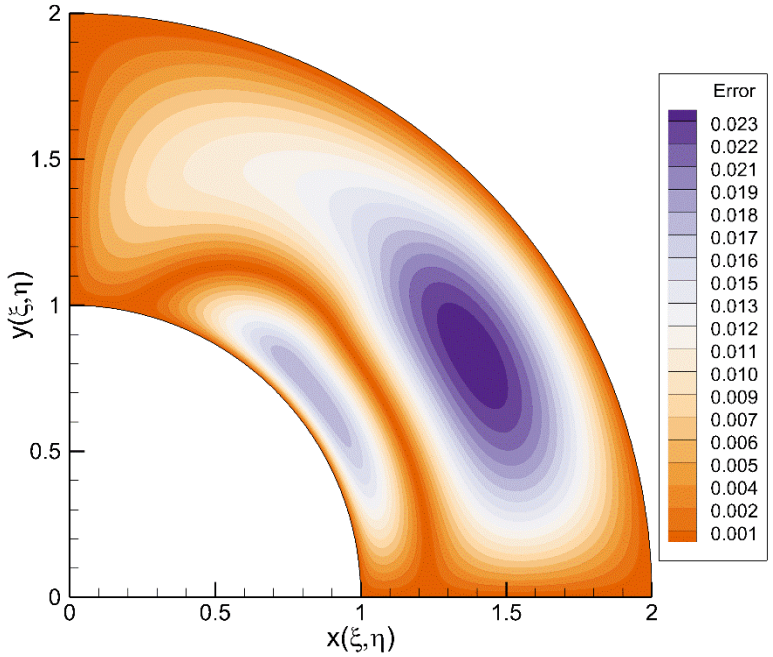


Figure 4.16. Error distribution of the initial solution before performing *w*-adaptivity.

Starting with this solution, we now execute the adaptivity process driven by the estimator. The history of estimated error as well as the exact error (in energy norm) during adaptivity are plotted in Figure 4.17. It is surprising to notice that although the adaptivity process has been guided by the estimator, the exact error has diminished to machine precision ($1.54e-13$) after 38 iterations, which implies the high reliability of the proposed adaptive framework.

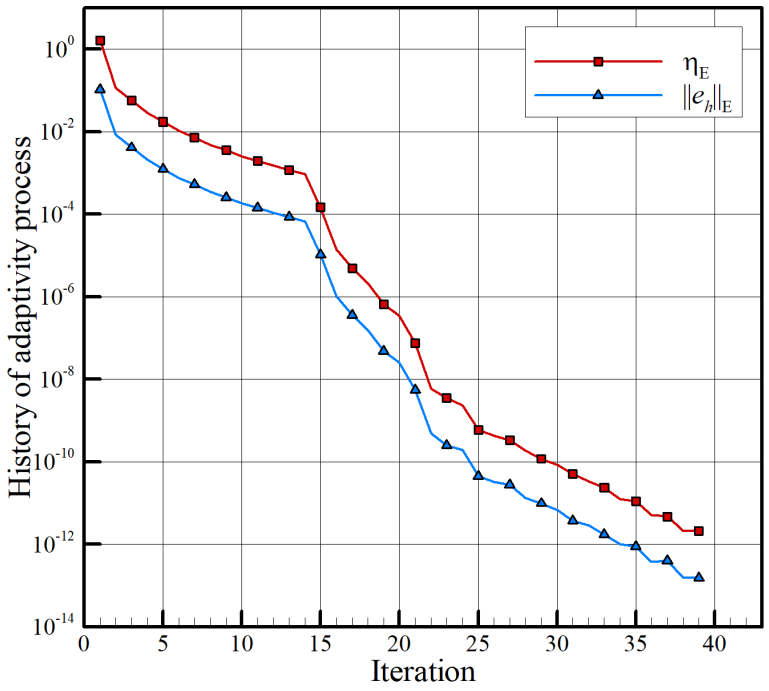


Figure 4.17. The histories of estimated and exact error during *w*-adaptivity.

The obtained optimal values of unknown control variables after adaptivity are presented in Table 4.5, alongside their analytical values. Considering this table, we can see that the obtained optimal values are in agreements with the analytical ones within 12 digits after the decimal place.

Table 4.5. The obtained optimal control weights by *w*-adaptivity together with the analytical values.

Design variable	Initial	Optimal	Analytical
w_6^u	1.0	0.707106781185937	0.707106781186548
w_7^u	1.0	0.707106781186656	0.707106781186548

Finally, it must be mentioned here that inspired by Theorem 1, it might seem tempting that one would be able to achieve this accuracy simply by performing the analysis using a higher order basis. We emphasize here that although Theorem 1 establishes that GNURBS can always be transformed to higher order classic NURBS, obtaining these results by directly making use of NURBS-based IGA using any order of basis functions is not possible. The reader can consult [78] for more details on this apparent inconsistency.

4.11.4. Test Case 4: Patch test

As the last numerical experiment, we study the satisfaction of patch test by the proposed *w*-adaptive isogeometric method. For this purpose, we consider the quarter annulus from previous example with a perturbed mesh as shown in Figure 4.18.

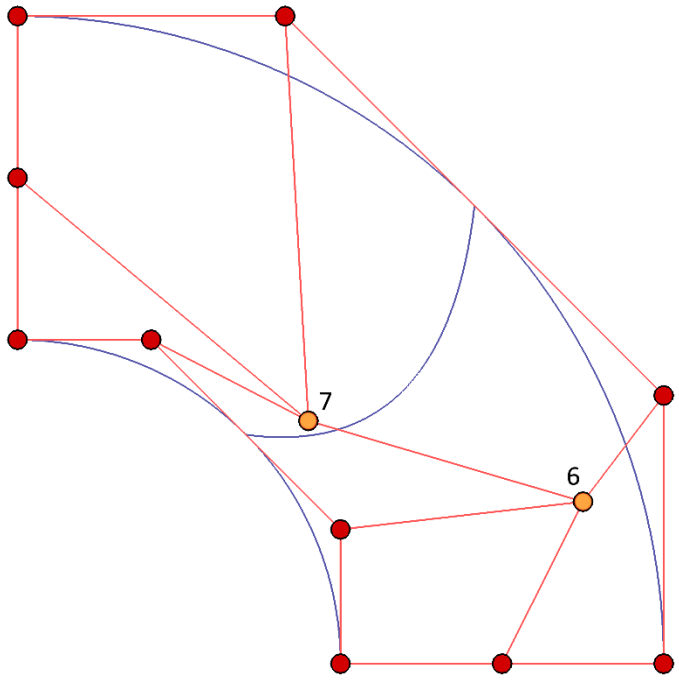


Figure 4.18. The quarter ring with a perturbed mesh.

We consider two cases of the standard patch test, as well as an extended patch test in rational space discussed below. In both cases, we use 12×12 quadrature points per knot-element for integration.

4.11.4.1. Standard patch test

We recall here that NURBS-based IGA satisfies the standard patch test as reported by Hughes et al. in [2,3]. This is in fact one of the key advantages of using an isoparametric basis. We investigate here if this test will also be satisfied by *w*-adaptive IGA. Towards this end, we consider a Poisson equation with Dirichlet boundary conditions on all edges

$$\begin{aligned} -\Delta u &= f \\ u &= u_D \quad \text{on}\,\Gamma_D \end{aligned} \tag{4.31}$$

where f and u_D are specified by the following exact solution

$$u(x,y) = 2x + y + 1. \tag{4.32}$$

First, we assign the boundary control weights the same values with those of the geometry. This ensures satisfaction of all essential conditions exactly. The boundary control variables are obtained by using linear least square fitting and enforced strongly.

Next, similar to previous example, we execute *w*-adaptivity with the initial guess of $\boldsymbol{\chi}_0 = \left\{ w_6^u, w_7^u \right\}_0 = \{1,1\}$ for interior control weights. The history of adaptivity process is illustrated in Figure 4.19. As observed, the error is diminished to machine precision ($3.26e-13$) after 31 iterations.

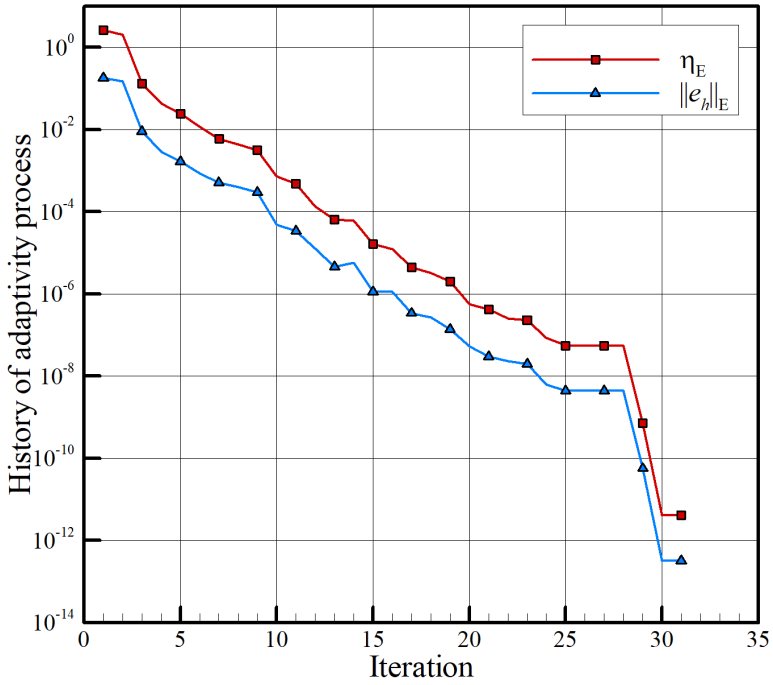


Figure 4.19. History of *w*-adaptivity for standard patch test.

Moreover, the obtained optimal values for control weights are presented in Table 4.6. According to this table, the obtained optimal values are in agreements with the analytical values, i.e. w_6^G, w_7^G, within 12 digits after the decimal place.

Table 4.6. The obtained optimal control weights by *w*-adaptivity as well as the analytical values for standard patch test.

Design variable	Initial	Optimal	Analytical
w_6^u	1.0	0.853553390593643	0.853553390593274
w_7^u	1.0	0.853553390592845	0.853553390593274

The results of this experiment are consistent with the assertions of Hughes et al. [3] regarding the fact that affine covariance, which is ensured by isoparametric concept, is an essential property for satisfying patch tests.

4.11.4.2. Rational patch test

As the final numerical experiment, we repeat the previous test on a Poisson problem with the following exact solution

$$u(x,y)=\frac{2x+y}{x+y+1} \tag{4.33}$$

Dirichlet boundary conditions specified by the exact solution are assumed on all edges. The exact boundary control weights and control variables are obtained by solving Eq. (4.21) and enforced strongly. Note that in this case, due to rational structure of the exact solution, the optimal control weights, unlike previous case, have nothing to do with those of the geometry. Similar to previous case, we conduct *w*-adaptivity with the initial guess of $\chi_0=\left\{w_6^u,w_7^u\right\}_0=\{1,1\}$ for interior control weights. The history of adaptivity process is depicted in Figure 4.20. As observed, the exact error is diminished to machine precision ($1.49e-14$) after 36 iterations.

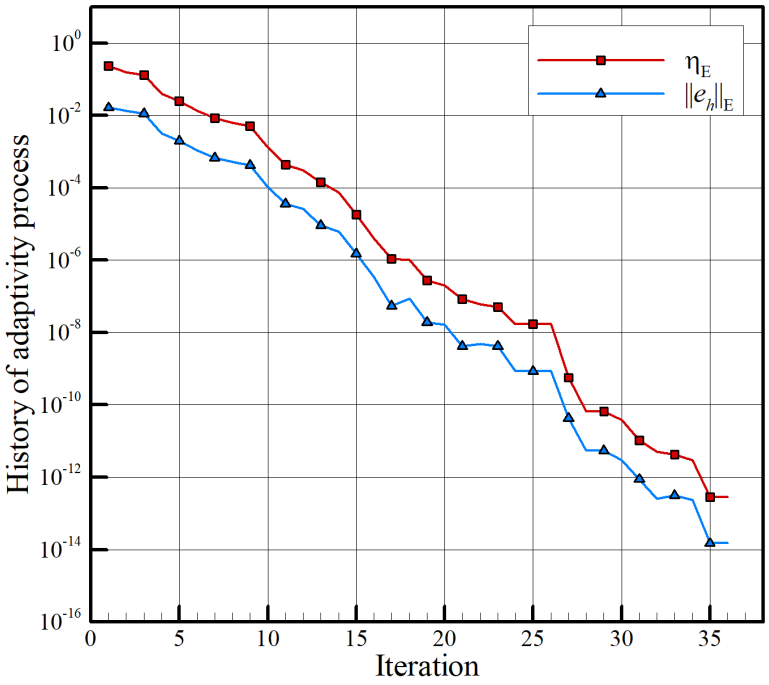


Figure 4.20. History of *w*-adaptivity for rational patch test.

This study reveals another significant fact about using rational splines for analysis, that is, over an arbitrary parameterization, *w*-adaptive IGA is able to reproduce the exact solution of a problem whose closed-form solution is a first order rational expression. The results of this experiment also lead to interesting questions for more studies on the correlation between isoparametric concept or affine-invariance and satisfaction of different patch tests. For instance, as mentioned earlier, Cottrell et al. [2] refer to affine-invariance, ensured by using an isoparametric basis, as an essential property for satisfying patch test. While this is consistent with our obtained results for the standard patch test, the devised experiment of rational patch test seems to be violating this necessity.

Chapter 5: Conclusions and future works

5.1. GNURBS in CAGD

We introduced two generalizations of NURBS, referred to as GNURBS, by decoupling of the weights associated with the control points along different physical coordinates. These generalizations were obtained via either explicit or implicit decoupling of the weights leading to non-isoparametric and isoparametric representations, respectively. As demonstrated, both these variations improve the flexibility of NURBS and circumvent its deficiencies by providing the possibility of treating the weights as additional design variables in special applications. It was proved that these representations are only variations of classic NURBS and do not constitute a new superset of NURBS. The superior approximation abilities of these variations for both smooth and rapidly varying functions were shown via simple examples in the context of CAGD in Chapters 2 and 3 for curves and surfaces, respectively. It was shown that GNURBS can be effectively used for improved construction of various types of curves and surfaces such as helical curves and surfaces, minimal surfaces and surfaces of revolution using the same number of control points.

Two comprehensive MATLAB toolboxes, named *GNURBS-Lab* and *GNURBS3D-Lab*, were developed and introduced for curves and surfaces, respectively, to demonstrate the behavior of GNURBS in a fully interactive manner. Overall, GNURBS was shown to provide a new powerful technology in CAGD with superior flexibility while including NURBS as a special case.

5.2. GNURBS as an enhanced tool for analysis

A novel adaptivity technique in isogeometric analysis, referred to as adaptive *w*-refinement, was introduced. The proposed adaptive method allows for the approximation of solution with optimal rational basis by treating the control weights as additional degrees of freedoms. It was shown that this procedure effectively alleviates the deficiencies of NURBS for analysis and leads to superior results at the expense of solving an unconstrained optimization problem. The performance of the proposed method on elliptic problems with smooth and sharp solutions was studied. It was observed that *w*-adaptive IGA results in one order faster convergence than classic IGA in the case of smooth problems, while significant improvement of accuracy is achieved in problems with

sharply varying solutions. Moreover, unlike classic IGA, the proposed adaptive method was demonstrated to be able to reproduce the exact solution of problems whose closed-form solutions lie in rational space, revealing a new critical aspect of using rational splines for analysis. Overall, the proposed adaptive *w*-refinement procedure provides a new effective technique for the enrichment of function space in isogeometric method and possibly a competitive tool with hierarchical splines.

5.3. Extensions and further applications

5.3.1. CAGD[4]

While, in this thesis, we limited our study to applying the proposed generalizations to NURBS curves and surfaces, they can be similarly applied to tri-variate volumes. Moreover, due to fundamental similarities between different variations of splines, these generalizations seem plausible to all other rational forms of splines such as T-splines, Tri-angular Béziers, etc.

In addition to the discussed applications in CAGD, there are other applications of NURBS where employing the weights as additional design variables for better flexibility can be problematic or sometimes impossible. For instance, while we mostly limited our numerical experiments to approximation over curved domains, GNURBS may also help circumventing the difficulties of considering the weights as degrees of freedom in general curve/surface fitting problems. As previously studied in [22,23], employing the weights as additional degrees of freedom in data approximation can deteriorate the surface parameterization, and lead to undesirable results. In this regard, existing studies suggest imposing bounding constraints on the variation of the weights explicitly or via regularization [11,20,21], to avoid this issue. However, this limits the obtained improvement in the accuracy of approximation, especially in the case of problems containing rapid variation in data or field where the weights tend to take extreme values.

On the other hand, employing the suggested variations of NURBS, one can create a good parameterization and preserve it while including the *control weights* as design variables for fitting the curve/surface to 3D data points, without imposing any limitations on the values of the weights.

---

[4] The materials in this section have been published in Engineering with Computers journal, ref. [78]

Further potential applications in CAD where GNURBS can be exploited with improved flexibility include NURBS-based metamodeling [108], which is of significant interest in engineering design.

5.3.2. Further improvements for analysis[5]

The present study opens doors to multiple new areas of research in IGA. While the numerical results are promising, there are many challenges which require further research to be addressed. We review a few of these aspects in this section in view of providing insights for interested readers.

Proof of convergence: While in this theis, we only numerically studied the rate of convergence of *w*-adaptive IGA, possible mathematical proofs of these results need to be developed. Specifically, in the case of second order elliptic PDEs with smooth solutions, the presented numerical results suggest that the NURBS basis of a particular degree with optimal weights has the same rate of convergence with the B-spline basis of one higher degree. To our knowledge, this has not been reported or proved in the literature. It also needs to be investigated if these results hold true for higher dimensions (3D) as well.

Adaptive Quadrature: One of the main aspects which needs to be more systematically addressed is devising efficient quadrature rules for integration over the arising highly-rational elements in *w*-adaptive IGA. In particular, two common strategies of devising a weighted quadrature rule [109], or employing an adaptive subdivision technique can be exploited for improved efficiency. Quadrature rules that restore variational consistency, or the so-called VCI domain integration methods which have been extended to IGA in [110], also provide an alternative promising technique for addressing this issue. An appropriate index, based on the total variation of denominator in Eq. (4.23) for instance, needs to be defined to identify elements where the quadrature needs to be improved.

Localization: Another aspect which can help improving the efficiency of the proposed adaptive method is the selection of the vector of design variables in Eq. (4.15). Although we only studied global *w*-adaptivity in this paper, as discussed earlier, this is unnecessary. For instance, it seems plausible to only consider the control weights of the elements in selective regions of the domain

[5] The materials in this section have been published in CMAME journal, ref. [112]

as design variables especially in the case of problems with irregularities such as test case 2. Robust and effective marking techniques, however, need to be developed and studied for any localization.

Stabilization: While the most promising scope of the proposed *w*-adaptive algorithm is perhaps on problems with sharp layers, instabilities may occur in many problems of this type with Galerkin approach [88]. Stabilization techniques, such as SUPG [111] need to be used in such scenarios. Incorporation of these techniques within the proposed *w*-adaptive framework, however, is non-trivial and needs further research.

Combination with other refinement techniques: Similar to other refinement techniques, the proposed *w*-refinement method can plausibly be combined with any of existing refinement methods for improved efficiency. Combination of this method with order elevation, in particular, requires employing an alternative generalization of NURBS, which is already introduced in [78] for performing optimal order elevation on parametric curves by the authors. In contrast to the introduced method in this paper, this combination will allow for introducing 'customized rationality' to the solution space which is expected to improve efficiency; please see [78] for more details.

Application to other rational splines: Finally, although we only studied here the application of the proposed *w*-adaptive method to GNURBS-based IGA, it seems plausible to all other variations of IGA based on rational forms of splines such as T-splines, U-splines etc.

5.3.3. Applications in structural optimization

Owing to the inherent properties of NURBS, they have been extensively used in structural optimization for the optimization of different fields of interest over a computational domain. For instance, Qian [79] employs B-spline basis for the representation of density field in FEM-based topology optimization as an intrinsic filtering technique. Within the framework of IGA, numerous studies have been performed where the same NURBS based parameterization of computational domain has also been used for the representation of different fields which need to be optimized over the domain in various applications such as size optimization of curved beams [83–85], topology optimization [8,80–82], optimization of material distribution in functionally graded materials (FGMs) [86,87] etc.

Having examined these studies, it can be noticed that in this class of applications, the parameterization of the design domain must remain fixed throughout the optimization process. Moreover, many of them require linear parameterization of the design domain and achieve this by placing the control points at their Greville abscissae, see e.g. [79,86]. Hence, they are only able to treat the out-of-plane coordinates of control points as design variables, as the variation of weights alters the underlying parameterization which is disallowed.

Owing to the proposed GNURBS representations with decoupled weights, one can now treat the out of plane weights as additional design variables while setting up the optimization problem and still preserve the underlying geometry as well as its parameterization unchanged. As the presented numerical results suggest, this idea can lead to significant improvement in the flexibility in both cases of smooth as well as rapidly varying fields. Exploring these applications is the subject of future studies.

5.4. Further generalizations

It is interesting to notice that in case of surfaces, an additional form of generalization is possible. This type of generalization is obtained by decoupling the weights with respect to parametric coordinates. This generalization could be applied to either classic NURBS surface, or either of the proposed non-isoparametric or isoparametric GNURBS surfaces. Any of these generalizations could potentially be useful in certain applications. We introduce here the following mathematical model as another possible generalization of NURBS surfaces

$$\mathbf{S}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}\mathcal{B}_{ij}^{p,q}(\xi,\eta)\mathbf{P}_{ij} \qquad \begin{matrix} 0\le\xi\le 1 \\ 0\le\eta\le 1 \end{matrix} \tag{5.1}$$

where

$$\mathcal{B}_{ij}^{p,q}(\xi,\eta)=\frac{N_i(\xi)N_j(\eta)w_{ij}^{\xi}w_{ij}^{\eta}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_k(\xi)N_l(\eta)w_{kj}^{\xi}w_{il}^{\eta}} \tag{5.2}$$

As observed, in this class of generalization, two different set of weights are associated with each basis, one of which is responsible for expanding the basis in ξ direction, while the other one expands the basis along η direction. It is important to note that the key obtained advantage in this formulation is that by changing the weights associated with a particular parametric direction, the parameterization of the surface in the other direction does not change. In other words, using this

formulation, one can create a parametric surface which is only rational in terms of one of the parametric coordinates while still remaining non-rational in terms of the other one.

One can naturally combine the above generalization with other generalizations, introduced earlier, by assigning different weights to each parametric as well as physical coordinate as follows

$$\mathbf{S}(\xi,\eta)=\sum_{i=0}^{n_1}\sum_{j=0}^{n_2}\boldsymbol{\mathcal{B}}_{ij}^{p,q}(\xi,\eta)\odot\mathbf{P}_{ij} \qquad \begin{matrix}0\le\xi\le1\\0\le\eta\le1\end{matrix} \tag{5.3}$$

in which $\boldsymbol{\mathcal{B}}_{ij}^{p,q}(\xi,\eta)=[\mathcal{B}_{ij}^{x}(\xi,\eta)\ \mathcal{B}_{ij}^{y}(\xi,\eta)\ \mathcal{B}_{ij}^{z}(\xi,\eta)]^T$ where superscripts *p*, *q* have been omitted for brevity in vector components. Denoting an arbitrary coordinate in physical space with $d\in\{x,y,\mathrm{z}\}$, the corresponding basis to *d*-coordinate could be written as

$$\mathcal{B}_{ij}^{d}(\xi,\eta)=\frac{N_{i,p}(\xi)N_{j,q}(\eta)w_{ij}^{d\xi}w_{ij}^{d\eta}}{\sum_{k=0}^{n_1}\sum_{l=0}^{n_2}N_k(\xi)N_l(\eta)w_{kj}^{d\xi}w_{il}^{d\eta}} \tag{5.4}$$

According to Eq. (5.3), a total of six independent weights has been assigned to each control point. Each of these weights are responsible to expand the surface in a certain parametric and physical direction.

Figure 5.1 shows a GNURBS bivariate basis function in the parametric space. As the figure shows, by increasing the weight associated with a particular parametric direction, the basis only expands in that direction while it remains *unchanged* in the other direction. To illustrate this fact better, we have plotted the neighboring basis functions in both ξ and η directions in Figure 5.2 and Figure 5.3, respectively. Comparing these figures, it can be clearly observed that the neighboring basis functions along ξ-direction have been contracted, while the ones along η-direction have remained unchanged as Bspline basis functions. The resulting GNURBS surface is depicted in Figure 5.4.

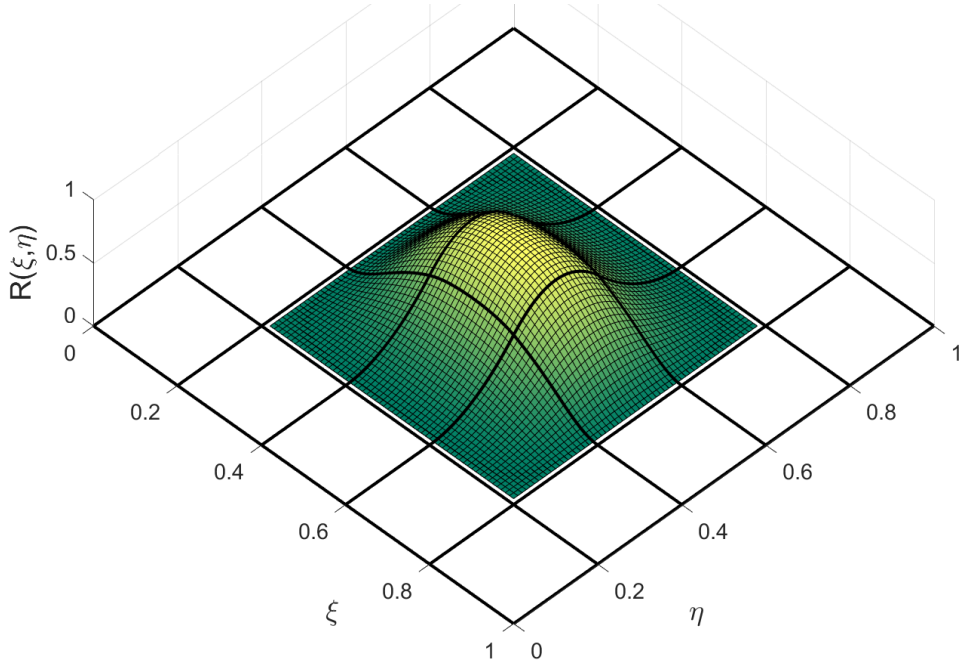


Figure 5.1. Bivariate GNURBS basis ß$_{4,4}$ (ξ,η) with w^{ξ}_{44} = 4 and w^{η}_{44} = 1 in parametric space.

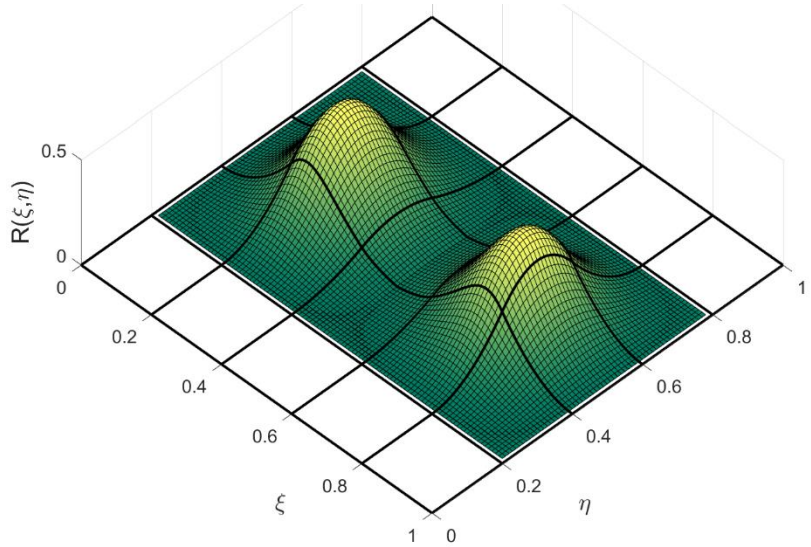


Figure 5.2. Bivariate neighboring GNURBS basis functions in ξ-direction: ß$_{3,4}$ (ξ,η) and ß$_{5,4}$ (ξ,η) with w^{ξ}_{44} = 4 and w^{η}_{44} = 1.

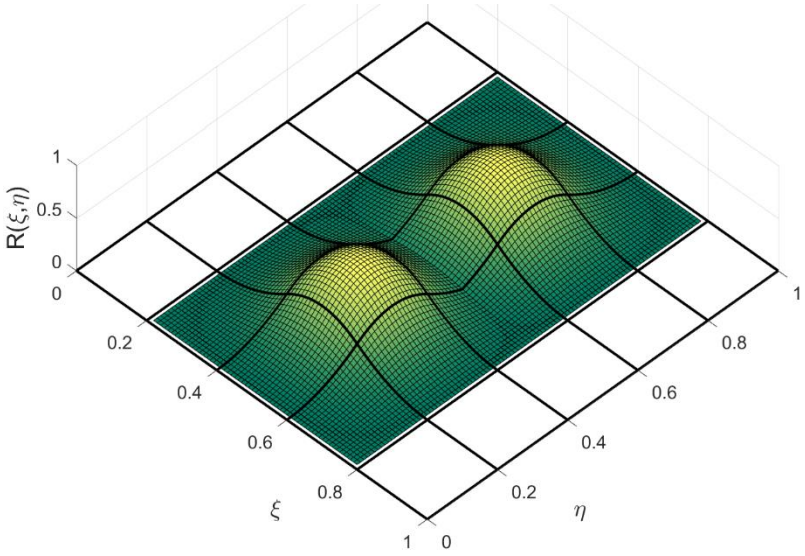


Figure 5.3. Bivariate neighboring GNURBS basis functions in η-direction: $ß_{4,3}(\xi,\eta)$ and $ß_{4,5}(\xi,\eta)$ with $w^{\xi}{}_{44} = 4$ and $w^{\eta}{}_{44} = 1$.

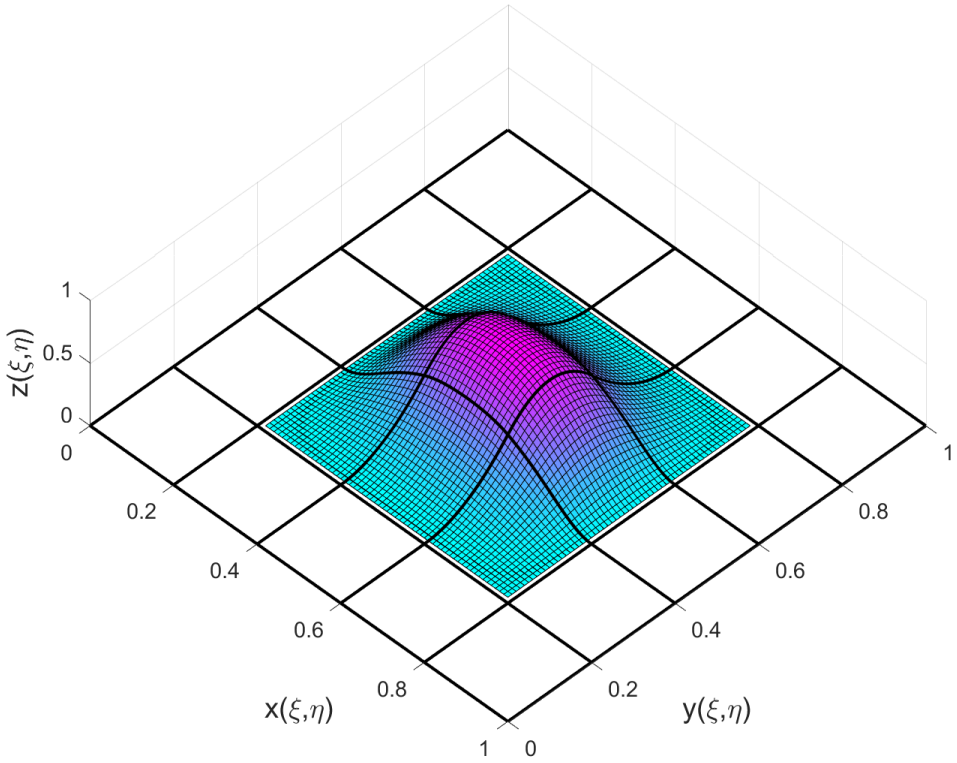


Figure 5.4. GNURBS surface with $w^{z\xi}{}_{44} = 4$ and $w^{z\eta}{}_{44} = 1$ in physical space over the nonzero area.

It is interesting to notice that this surface is rational in terms of ξ, while it is non-rational in terms of η. It can be shown that this property plays a crucial role in certain applications such as the treatment of nonhomogeneous essential boundary conditions in IGA etc, which will be the subject of future studies.

Finally, while we did not explore the theoretical properties of these additional generalizations in this thesis, successful application of these representations requires satisfaction of certain properties such as linear independence of basis functions and partition of unity. Exploring these theoretical properties is the subject of future studies.

Bibliography

[1] K.J. Versprille, Computer-aided Design Applications of the Rational B-spline Approximation Form, 1975.

[2] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric Analysis: Toward Integration of CAD and FEA, John Wiley & Sons, 2009. https://doi.org/10.1002/9780470749081.

[3] T.J.R. Hughes, J.A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Eng. 194 (2005) 4135–4195. https://doi.org/10.1016/j.cma.2004.10.008.

[4] B.P. Mishra, M. Barik, NURBS-augmented finite element method for stability analysis of arbitrary thin plates, Eng. Comput. 35 (2019) 351–362. https://doi.org/10.1007/s00366-018-0603-9.

[5] X. Qian, Full analytical sensitivities in NURBS based isogeometric shape optimization, Comput. Methods Appl. Mech. Eng. 199 (2010) 2059–2071. https://doi.org/10.1016/j.cma.2010.03.005.

[6] T. Takahashi, T. Yamamoto, Y. Shimba, H. Isakari, T. Matsumoto, A framework of shape optimisation based on the isogeometric boundary element method toward designing thin-silicon photovoltaic devices, Eng. Comput. 35 (2019) 423–449. https://doi.org/10.1007/s00366-018-0606-6.

[7] Q.X. Lieu, J. Lee, A multi-resolution approach for multi-material topology optimization based on isogeometric analysis, Comput. Methods Appl. Mech. Eng. 323 (2017) 272–302. https://doi.org/10.1016/j.cma.2017.05.009.

[8] A.H. Taheri, K. Suresh, An isogeometric approach to topology optimization of multi-material and functionally graded structures, Int. J. Numer. Methods Eng. 109 (2017) 668–696. https://doi.org/10.1002/nme.5303.

[9] M. Coelho, D. Roehl, K.-U. Bletzinger, Material Model Based on Response Surfaces of NURBS Applied to Isotropic and Orthotropic Materials, in: P.A. Muñoz-Rojas (Ed.), Comput. Model. Optim. Manuf. Simul. Adv. Eng. Mater., Springer International Publishing, Cham, 2016: pp. 353–373. https://doi.org/10.1007/978-3-319-04265-7_13.

[10] M. Coelho, D. Roehl, K.U. Bletzinger, Material model based on NURBS response surfaces, Appl. Math. Model. 51 (2017) 574–586. https://doi.org/10.1016/j.apm.2017.06.038.

[11] W. Ma, J.-P. Kruth, NURBS curve and surface fitting for reverse engineering, Int. J. Adv. Manuf. Technol. 14 (1998) 918–927. https://doi.org/10.1007/BF01179082.

[12] S.A. Kanna, A. Tovar, J.S. Wou, H. El-Mounayri, Optimized NURBS Based G-Code Part Program for High-Speed CNC Machining, in: ASME 2014 Int. Des. Eng. Tech. Conf. Comput. Inf. Eng. Conf., American Society of Mechanical Engineers, 2014.

[13] T. Sederberg, T.W., Cardon, D.L., Finnigan, G.T., North, N.S., Zheng, J., Lyche, T-spline simplification and local refinement, ACM Trans. Graph. 23 (2004) 276–283.

[14] A. Sederberg, T.N., Zhengs, J.M., Bakenov, A., Nasri, T-splines and T-NURCCSs, ACM Trans. Graph. 22 (2003) 477–484.

[15] W. Chen, Generalized Hierarchical NURBS for Interactive Shape Modification, in: VRCAI ’08 Proc. 7th ACM SIGGRAPH Int. Conf. Virtual-Reality Contin. Its Appl. Ind., 2008: pp. 1–4. https://doi.org/10.1145/1477862.1477894.

[16] G. Hu, J. Wu, X. Qin, A novel extension of the Bézier model and its applications to surface modeling, Adv. Eng. Softw. 125 (2018) 27–54. https://doi.org/10.1016/j.advengsoft.2018.09.002.

[17] D. Thomas, U-splines: splines over unstructured meshes, US20190130058A1, 2019. https://patents.google.com/patent/US20190130058A1/en.

[18] Q. Wang, W. Hua, G. Li, H. Bao, Generalized NURBS curves and surfaces, Proc. - Geom. Model. Process. 2004. (2004) 365–368. https://doi.org/10.1023/B:JMSC.0000008091.55395.ee.

[19] L. Piegl, W. Tiller, The NURBS Book, 1st ed., Springer-Verlag Berlin Heidelberg, 1995. https://doi.org/10.1007/978-3-642-97385-7.

[20] N. Carlson, NURBS Surface Fitting with Gauss-Newton, Lulea University of Technology, 2009.

[21] W. Ma, NURBS-based computer aided design modelling from measured points of physical models, Catholic University of Leuven, 1994.

[22] E. Dimas, D. Briassoulis, 3D geometric modelling based on NURBS: a review, Adv. Eng. Softw. 30 (1999) 741–751.

[23] L. Piegl, On NURBS: A Survey, IEEE Comput. Graph. Appl. (1991) 55–71.

[24] D. Schillinger, E. Rank, An unfitted hp-adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry, Comput. Methods Appl. Mech. Eng. 200 (2011) 3358–3380. https://doi.org/10.1016/j.cma.2011.08.002.

[25] U. Basappa, A. Rajagopal, J.N. Reddy, Adaptive Isogeometric Analysis Based on a Combined r-h Strategy, Int. J. Comput. Methods Eng. Sci. Mech. 2287 (2016). https://doi.org/10.1080/15502287.2016.1153171.

[26] A. Mirzakhani, B. Hassani, A. Ganjali, Adaptive analysis of three-dimensional structures using an isogeometric control net refinement approach, Acta Mech. 226 (2015) 3425–3449. https://doi.org/10.1007/s00707-015-1376-5.

[27] G. Xu, B. Mourrain, R. Duvigneau, A. Galligo, Parameterization of computational domain in isogeometric analysis : Methods and comparison, Comput. Methods Appl. Mech. Eng. 200 (2011) 2021–2031. https://doi.org/10.1016/j.cma.2011.03.005.

[28] D. Schillinger, L. Dedè, M.A. Scott, J.A. Evans, M.J. Borden, E. Rank, T.J.R. Hughes, An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces, Comput. Methods Appl. Mech. Eng. 249–252 (2012) 116–150. https://doi.org/10.1016/j.cma.2012.03.017.

[29] C. Giannelli, B. Jüttler, S.K. Kleiss, A. Mantzaflaris, B. Simeon, J. Špeh, THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis, Comput. Methods Appl. Mech. Eng. 299 (2016) 337–365. https://doi.org/10.1016/j.cma.2015.11.002.

[30] T. Kanduč, C. Giannelli, F. Pelosi, H. Speleers, Adaptive isogeometric analysis with hierarchical box splines, Comput. Methods Appl. Mech. Eng. 316 (2017) 817–838.

https://doi.org/10.1016/j.cma.2016.09.046.

[31] E.J. Evans, M.A. Scott, X. Li, D.C. Thomas, Hierarchical T-splines: Analysis-suitability, Bézier extraction, and application as an adaptive basis for isogeometric analysis, Comput. Methods Appl. Mech. Eng. 284 (2015) 1–20. https://doi.org/10.1016/j.cma.2014.05.019.

[32] D. Schillinger, J.A. Evans, A. Reali, M.A. Scott, T.J.R. Hughes, Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations, Comput. Methods Appl. Mech. Eng. 267 (2013) 170–232. https://doi.org/10.1016/j.cma.2013.07.017.

[33] A. Vuong, C. Giannelli, B. Jüttler, B. Simeon, A hierarchical approach to adaptive local refinement in isogeometric analysis, Comput. Methods Appl. Mech. Eng. 200 (2011) 3554–3567. https://doi.org/10.1016/j.cma.2011.09.004.

[34] P. Wang, J. Xu, J. Deng, F. Chen, Adaptive isogeometric analysis using rational PHT-splines, Comput. Aided Des. 43 (2011) 1438–1448. https://doi.org/10.1016/j.cad.2011.08.026.

[35] B.S. Michael R. Dörfel, Bert Jüttler, Adaptive isogeometric analysis by local h-refinement with T-splines, Comput. Methods Appl. Mech. Eng. 199 (2010) 264–275. https://doi.org/10.1016/j.cma.2008.07.012.

[36] T. Lyche, C. Manni, H. Speleers, eds., Splines and PDEs: From Approximation Theory to Numerical Linear Algebra, Springer, Cetraro, Italy, 2017. https://doi.org/10.1007/978-3-319-94911-6.

[37] X. Li, F.L. Chen, H.M. Kang, J.S. Deng, A survey on the local refinable splines, Sci. China Math. 59 (2016) 617–644. https://doi.org/10.1007/s11425-015-5063-8.

[38] Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, T.W. Sederberg, Isogeometric analysis using T-splines, Comput. Methods Appl. Mech. Eng. 199 (2010) 229–263. https://doi.org/10.1016/j.cma.2009.02.036.

[39] H. Casquero, X. Wei, D. Toshniwal, A. Li, T.J.R. Hughes, J. Kiendl, Y.J. Zhang, Seamless integration of design and Kirchhoff–Love shell analysis using analysis-suitable unstructured T-splines, Comput. Methods Appl. Mech. Eng. 360 (2020) 112765.

https://doi.org/10.1016/j.cma.2019.112765.

[40] L. Liu, H. Casquero, H. Gomez, Y.J. Zhang, Hybrid-degree weighted T-splines and their application in isogeometric analysis, Comput. Fluids. 141 (2016) 42–53. https://doi.org/10.1016/j.compfluid.2016.03.020.

[41] H. Casquero, L. Liu, Y. Zhang, A. Reali, H. Gomez, Isogeometric collocation using analysis-suitable T-splines of arbitrary degree, Comput. Methods Appl. Mech. Eng. 301 (2016) 164–186. https://doi.org/10.1016/j.cma.2015.12.014.

[42] H. Casquero, L. Liu, Y. Zhang, A. Reali, J. Kiendl, H. Gomez, Arbitrary-degree T-splines for isogeometric analysis of fully nonlinear Kirchhoff–Love shells, CAD Comput. Aided Des. 82 (2017) 140–153. https://doi.org/10.1016/j.cad.2016.08.009.

[43] M.A. Scott, X. Li, T.W. Sederberg, T.J.R. Hughes, Local refinement of analysis-suitable T-splines, Comput. Methods Appl. Mech. Eng. 213–216 (2012) 206–222. https://doi.org/10.1016/j.cma.2011.11.022.

[44] W. Wang, Y. Zhang, L. Liu, T.J.R. Hughes, Trivariate solid T-spline construction from boundary triangulations with arbitrary genus topology, CAD Comput. Aided Des. 45 (2013) 351–360. https://doi.org/10.1016/j.cad.2012.10.018.

[45] L. Liu, Y. Zhang, T.J.R. Hughes, M.A. Scott, T.W. Sederberg, Volumetric T-spline construction using Boolean operations, Eng. Comput. 30 (2013) 425–439. https://doi.org/10.1007/s00366-013-0346-6.

[46] Y. Zhang, W. Wang, T.J.R. Hughes, Conformal solid T-spline construction from boundary T-spline representations, Comput. Mech. 51 (2013) 1051–1059. https://doi.org/10.1007/s00466-012-0787-6.

[47] Y. Zhang, W. Wang, T.J.R. Hughes, Solid T-spline construction from boundary representations for genus-zero geometry, Comput. Methods Appl. Mech. Eng. 249–252 (2012) 185–197. https://doi.org/10.1016/j.cma.2012.01.014.

[48] W. Wang, Y. Zhang, G. Xu, T.J.R. Hughes, Converting an unstructured quadrilateral/hexahedral mesh to a rational T-spline, Comput. Mech. 50 (2012) 65–84. https://doi.org/10.1007/s00466-011-0674-6.

[49] W. Wang, Y. Zhang, M.A. Scott, T.J.R. Hughes, Converting an unstructured quadrilateral mesh to a standard T-spline surface, Comput. Mech. 48 (2011) 477–498. https://doi.org/10.1007/s00466-011-0598-1.

[50] X. Wei, Y. Zhang, L. Liu, T.J.R. Hughes, Truncated T-splines: Fundamentals and methods, Comput. Methods Appl. Mech. Eng. 316 (2017) 349–372. https://doi.org/10.1016/j.cma.2016.07.020.

[51] X. Wei, Y. Zhang, T.J.R. Hughes, M.A. Scott, Truncated hierarchical Catmull-Clark subdivision with local refinement, Comput. Methods Appl. Mech. Eng. 291 (2015) 1–20. https://doi.org/10.1016/j.cma.2015.03.019.

[52] X. Wei, Y.J. Zhang, T.J.R. Hughes, M.A. Scott, Extended Truncated Hierarchical Catmull-Clark Subdivision, Comput. Methods Appl. Mech. Eng. 299 (2016) 316–336. https://doi.org/10.1016/j.cma.2015.10.024.

[53] X. Li, X. Wei, Y.J. Zhang, Hybrid non-uniform recursive subdivision with improved convergence rates, Comput. Methods Appl. Mech. Eng. 352 (2019) 606–624. https://doi.org/10.1016/j.cma.2019.04.036.

[54] K.A. Johannessen, T. Kvamsdal, T. Dokken, Isogeometric analysis using LR B-splines, Comput. Methods Appl. Mech. Eng. 269 (2014) 471–514. https://doi.org/10.1016/j.cma.2013.09.014.

[55] X. Wei, Y.J. Zhang, T.J.R. Hughes, Truncated hierarchical tricubic C0 spline construction on unstructured hexahedral meshes for isogeometric analysis applications, Comput. Math. with Appl. 74 (2017) 2203–2220. https://doi.org/10.1016/j.camwa.2017.07.043.

[56] X. Wei, Y.J. Zhang, D. Toshniwal, H. Speleers, X. Li, C. Manni, J.A. Evans, T.J.R. Hughes, Blended B-spline construction on unstructured quadrilateral and hexahedral meshes with optimal convergence rates in isogeometric analysis, Comput. Methods Appl. Mech. Eng. 341 (2018) 609–639. https://doi.org/10.1016/j.cma.2018.07.013.

[57] A. Qarariyah, F. Deng, T. Yang, Y. Liu, J. Deng, Isogeometric analysis on implicit domains using weighted extended PHT-splines, J. Comput. Appl. Math. 350 (2019) 353–371. https://doi.org/10.1016/j.cam.2018.10.012.

[58] Y. Lai, Y.J. Zhang, L. Liu, X. Wei, E. Fang, J. Lua, Integrating CAD with Abaqus: A practical isogeometric analysis software platform for industrial applications, Comput. Math. with Appl. 74 (2017) 1648–1660. https://doi.org/10.1016/j.camwa.2017.03.032.

[59] E. Atroshchenko, S. Tomar, G. Xu, S.P.A. Bordas, Weakening the tight coupling between geometry and simulation in isogeometric analysis: From sub- and super-geometric analysis to Geometry-Independent Field approximaTion (GIFT), Int J Numer Methods Eng. 114 (2018) 1131–1159. https://doi.org/10.1002/nme.5778.

[60] A. Le Mehaute, L.L. Schumaker, C. Rabut, eds., Curves and Surfaces with Applications in CAGD, 1st ed., Vanderbilt University Press, 1997.

[61] L. Piegl, W. Tiller, Symbolic operators for NURBS, Comput. Aided Des. 29 (1997) 361–368. https://doi.org/10.1016/S0010-4485(96)00074-7.

[62] X. Che, G. Farin, Z. Gao, D. Hansford, The product of two B-spline functions, Adv. Mater. Res. 186 (2011) 445–448. https://doi.org/10.4028/www.scientific.net/AMR.186.445.

[63] X. Chen, R.F. Riesenfeld, E. Cohen, An algorithm for direct multiplication of B-splines, IEEE Trans. Autom. Sci. Eng. 6 (2009) 433–442. https://doi.org/10.1109/TASE.2009.2021327.

[64] E.T.Y. Lee, Computing a chain of blossoms, with application to products of splines, Comput. Aided Geom. Des. 11 (1994) 597–620.

[65] K. Mørken, Some identities for products and degree raising of splines, Constr. Approx. 7 (1991) 195–208. https://doi.org/10.1007/BF01888153.

[66] G. Farin, Curves and Surfaces for CAGD A Practical Guide, 5th ed., Morgan Kaufmann, 2001.

[67] P. Alfeld, M. Neamtu, L.L. Schumaker, Fitting scattered data on sphere-like surfaces using spherical splines, J. Comput. Appl. Math. 73 (1996) 5–43. https://doi.org/10.1016/0377-0427(96)00034-9.

[68] G.E. Fasshauer, L.L. Schumaker, Scattered Data Fitting on the sphere, Vanderbilt

University Press, Nashville, TN, 1998. https://doi.org/10.1016/j.powlec.2007.06.004.

[69] X. Pu, W. Liu, A subdivision scheme for approximating circular helix with NURBS curve, Proceeding 2009 IEEE 10th Int. Conf. Comput. Ind. Des. Concept. Des. E-Business, Creat. Des. Manuf. - CAID CD'2009. (2009) 620–624. https://doi.org/10.1109/CAIDCD.2009.5374879.

[70] X. Yang, High accuracy approximation of helices by quintic curves, Comput. Aided Geom. Des. 20 (2003) 303–317. https://doi.org/10.1016/S0167-8396(03)00074-8.

[71] I. Juhasz, Approximating the helix with rational cubic Bezier curves, Comput. Des. 27 (1995) 587–593.

[72] S. Shojaee, E. Izadpenah, A. Haeri, Imposition of Essential Boundary Conditions in Isogeometric Analysis Using the Lagrange Multiplier Method, Int. J. Optim. Civ. Eng. 2 (2012) 247–271. http://ijoce.iust.ac.ir/browse.php?a_code=A-10-1-65&slc_lang=en&sid=1.

[73] D. Wang, J. Xuan, An improved NURBS-based isogeometric analysis with enhanced treatment of essential boundary conditions, Comput. Methods Appl. Mech. Eng. 199 (2010) 2425–2436. https://doi.org/10.1016/j.cma.2010.03.032.

[74] A. Embar, J. Dolbow, I. Harari, Imposing Dirichlet boundary conditions with Nitsche's method and spline-based finite elements, Int. J. Numer. Methods Eng. 83 (2010) 877–898. https://doi.org/10.1002/nme.2863.

[75] T. Chen, R. Mo, Z.W. Gong, Imposing Essential Boundary Conditions in Isogeometric Analysis with Nitsche's Method, Appl. Mech. Mater. 121–126 (2011) 2779–2783. https://doi.org/10.4028/www.scientific.net/AMM.121-126.2779.

[76] H. Pottmann, S. Leopoldseder, M. Hofer, Approximation with Active B-spline Curves and Surfaces, in: Proc. 10th Pacific Conf. Comput. Graph. Appl., IEEE, 2002: pp. 8–25. https://doi.org/10.1109/PCCGA.2002.1167835.

[77] C. Erdönmez, N-Tuple Complex Helical Geometry Modeling Using Parametric Equations, Eng. Comput. 30 (2013) 715–726. https://doi.org/10.1007/s00366-013-0319-9.

[78] A.H. Taheri, S. Abolghasemi, K. Suresh, Generalizations of non-uniform rational B-splines via decoupling of the weights: theory, software and applications, Eng. Comput. 36 (2019) 1831–1848. https://doi.org/10.1007/s00366-019-00799-w.

[79] X. Qian, Topology optimization in B-spline space, Comput. Methods Appl. Mech. Eng. 265 (2013) 15–35. https://doi.org/10.1016/j.cma.2013.06.001.

[80] B. Hassani, M. Khanzadi, S.M. Tavakkoli, An isogeometrical approach to structural topology optimization by optimality criteria, Struct. Multidiscip. Optim. 45 (2012) 223–233. https://doi.org/10.1007/s00158-011-0680-5.

[81] Y. Wang, D.J. Benson, Isogeometric analysis for parameterized LSM-based structural topology optimization, Comput. Mech. 57 (2016) 19–35. https://doi.org/10.1007/s00466-015-1219-1.

[82] L. Dedè, M.M.J. Borden, T.J.R.T. Hughes, Isogeometric analysis for topology optimization with a phase field model, Arch. Comput. Methods Eng. 19 (2012) 427–465. https://doi.org/10.1007/s11831-012-9075-z.

[83] A.P. Nagy, M.M. Abdalla, Z. Gürdal, Isogeometric sizing and shape optimisation of beam structures, Comput. Methods Appl. Mech. Eng. 199 (2010) 1216–1230. https://doi.org/10.1016/j.cma.2009.12.010.

[84] A.P. Nagy, M.M. Abdalla, Z. Gürdal, Isogeometric sizing and shape optimisation of beam structures, Comput. Methods Appl. Mech. Eng. 199 (2010) 1216–1230. https://doi.org/10.1016/j.cma.2009.12.010.

[85] H. Liu, D. Yang, X. Wang, Y. Wang, C. Liu, Z.P. Wang, Smooth size design for the natural frequencies of curved Timoshenko beams using isogeometric analysis, Struct. Multidiscip. Optim. 59 (2019) 1143–1162. https://doi.org/10.1007/s00158-018-2119-8.

[86] Q.X. Lieu, J. Lee, Modeling and optimization of functionally graded plates under thermo-mechanical load using isogeometric analysis and adaptive hybrid evolutionary firefly algorithm, Compos. Struct. 179 (2017) 89–106. https://doi.org/10.1016/j.compstruct.2017.07.016.

[87] A.H. Taheri, B. Hassani, N.Z. Moghaddam, Thermo-elastic optimization of material

distribution of functionally graded structures by an isogeometrical approach, Int. J. Solids Struct. 51 (2014) 416–429. https://doi.org/10.1016/j.ijsolstr.2013.10.014.

[88] J. Donea, A. Huerta, Finite Element Methods for Flow Problems., John Wiley & Sons, 2003. https://doi.org/10.1002/0470013826.

[89] J.T.O. Mark Ainsworth, A Posteriori Error Estimation in Finite Element Analysis, Wiley-Interscience [John Wiley & Sons], New York, 2000.

[90] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, E. Rank, Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method, Int. J. Numer. Methods Eng. 95 (2013) 811–846. https://doi.org/10.1002/nme.4522.

[91] Y. Guo, M. Ruess, Weak Dirichlet boundary conditions for trimmed thin isogeometric shells, Comput. Math. with Appl. 70 (2015) 1425–1440. https://doi.org/10.1016/j.camwa.2015.06.012.

[92] S. Fernandez-Mendez, A. Huerta, Imposing essential boundary conditions in mesh-free methods, Comput. Methods Appl. Mech. Eng. 193 (2004) 1257–1275. https://doi.org/10.1016/j.cma.2003.12.019.

[93] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric analysis, John Wiley & Sons, 2009.

[94] P. Costantini, C. Manni, F. Pelosi, M.L. Sampoli, Quasi-interpolation in isogeometric analysis based on generalized B-splines, Comput. Aided Geom. Des. 27 (2010) 656–668. https://doi.org/10.1016/j.cagd.2010.07.004.

[95] J. Lu, G. Yang, J. Ge, Blending NURBS and Lagrangian representations in isogeometric analysis, Comput. Methods Appl. Mech. Eng. 257 (2013) 117–125. https://doi.org/10.1016/j.cma.2013.01.012.

[96] V. Calo, Q. Deng, V. Puzyrev, Quadrature blending for isogeometric analysis, Procedia Comput. Sci. 108 (2017) 798–807. https://doi.org/10.1016/j.procs.2017.05.143.

[97] R.R. Hiemstra, F. Calabrò, D. Schillinger, T.J.R. Hughes, Optimal and reduced quadrature

rules for tensor product and hierarchically refined splines in isogeometric analysis, 316 (2017) 966–1004. https://doi.org/10.1016/j.cma.2016.10.049.

[98] F. Calabrò, G. Sangalli, M. Tani, Fast formation of isogeometric Galerkin matrices by weighted quadrature, Comput. Methods Appl. Mech. Eng. 316 (2017) 606–622. https://doi.org/10.1016/j.cma.2016.09.013.

[99] M. Barton, V.M. Calo, Optimal quadrature rules for odd-degree spline spaces and their application to tensor-product-based isogeometric analysis, 305 (2016) 217–240. https://doi.org/10.1016/j.cma.2016.02.034.

[100] F. Auricchio, F. Calabrò, T.J.R. Hughes, A. Reali, G. Sangalli, A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis, Comput. Methods Appl. Mech. Eng. 249–252 (2012) 15–27. https://doi.org/10.1016/j.cma.2012.04.014.

[101] T.J.R. Hughes, A. Reali, G. Sangalli, Efficient quadrature for NURBS-based isogeometric analysis, Comput. Methods Appl. Mech. Eng. 199 (2010) 301–313. https://doi.org/10.1016/j.cma.2008.12.004.

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

[103] C. Carstensen, L. Demkowicz, J. Gopalakrishnan, Breaking spaces and forms for the DPG method and applications including Maxwell equations, Comput. Math. with Appl. 72 (2016) 494–522. https://doi.org/10.1016/j.camwa.2016.05.004.

[104] J. Chan, N. Heuer, T. Bui-Thanh, L. Demkowicz, A robust DPG method for convection-dominated diffusion problems II: Adjoint boundary conditions and mesh-dependent test norms, Comput. Math. with Appl. 67 (2014) 771–795. https://doi.org/10.1016/j.camwa.2013.06.010.

[105] L. Beirao da Veiga, D. Cho, G. Sangalli, Anisotropic NURBS approximation in isogeometric analysis, Comput. Methods Appl. Mech. Eng. 209–212 (2012) 1–11. https://doi.org/10.1016/j.cma.2011.10.016.

[106] PGI Visual FORTRAN Reference Guide, NVIDIA Corporation, Hillsboro, OR, 2018.

[107] Vanderplaats, DOT – Design Optimization Tools, Users Manual, Vanderplaats Research & Development, Inc., Colorado Springs, CO., 1999.

[108] C.J. Turner, R.H. Crawford, M.I. Campbell, Multidimensional sequential sampling for NURBs-based metamodel development, Eng. Comput. 23 (2007) 155–174. https://doi.org/10.1007/s00366-006-0051-9.

[109] W. Gautschi, The use of rational functions in numerical quadrature, J. Comput. Appl. Math. 133 (2001) 111–126. https://doi.org/10.1016/S0377-0427(00)00637-3.

[110] M. Hillman, J.S. Chen, Y. Bazilevs, Variationally consistent domain integration for isogeometric analysis, Comput. Methods Appl. Mech. Eng. 284 (2015) 521–540. https://doi.org/10.1016/j.cma.2014.10.004.

[111] A.N. Brooks, T.J.R. Hughes, Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations, Comput. Methods Appl. Mech. Engrg. 32 (1982) 199–259.

[112] A.H. Taheri, K. Suresh, Adaptive w-refinement : A new paradigm in isogeometric analysis, Comput. Methods Appl. Mech. Eng. 368 (2020). https://doi.org/10.1016/j.cma.2020.113180.

Appendix A: Derivation of the second order derivatives of basis functions and field variable

We recall the following relations for the transformation of the first and second order derivatives of an arbitrary variable $\mathcal{A}$ between physical and parametric coordinates:

$$\begin{bmatrix} \dfrac{\partial \mathcal{A}}{\partial x} \\ \dfrac{\partial \mathcal{A}}{\partial y} \end{bmatrix} = \left(J\right)^{-1} \begin{bmatrix} \dfrac{\partial \mathcal{A}}{\partial \xi} \\ \dfrac{\partial \mathcal{A}}{\partial \eta} \end{bmatrix} \tag{A.1}$$

where the Jacobian matrix J is

$$J = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \tag{A.2}$$

Also,

$$\begin{Bmatrix} \dfrac{\partial^2 \mathcal{A}}{\partial x^2} \\ \dfrac{\partial^2 \mathcal{A}}{\partial x \partial y} \\ \dfrac{\partial^2 \mathcal{A}}{\partial y^2} \end{Bmatrix} = \left(J^{II}\right)^{-1} \begin{Bmatrix} \dfrac{\partial^2 \mathcal{A}}{\partial \xi^2} - \dfrac{\partial \mathcal{A}}{\partial x} x_{,\xi\xi} - \dfrac{\partial \mathcal{A}}{\partial y} y_{,\xi\xi} \\ \dfrac{\partial^2 \mathcal{A}}{\partial \xi \partial \eta} - \dfrac{\partial \mathcal{A}}{\partial x} x_{,\xi\eta} - \dfrac{\partial \mathcal{A}}{\partial y} y_{,\xi\eta} \\ \dfrac{\partial^2 \mathcal{A}}{\partial \eta^2} - \dfrac{\partial \mathcal{A}}{\partial x} x_{,\eta\eta} - \dfrac{\partial \mathcal{A}}{\partial y} y_{,\eta\eta} \end{Bmatrix} \tag{A.3}$$

where

$$J^{II} = \begin{bmatrix} x_{,\xi}^2 & 2x_{,\xi}y_{,\xi} & y_{,\xi}^2 \\ x_{,\xi}x_{,\eta} & x_{,\xi}y_{,\eta} + x_{,\eta}y_{,\xi} & y_{,\xi}y_{,\eta} \\ x_{,\eta}^2 & 2x_{,\eta}y_{,\eta} & y_{,\eta}^2 \end{bmatrix}, \tag{A.4}$$

Note that Eqs. (A.1) and (A.3) can be directly used for the calculation of spatial derivatives of the basis functions as well as the field variable. We will later use the above expressions for sensitivity analysis.

Appendix B: Derivation of sensitivity expressions

For brevity, we first define the following simplified notations:

$$
\begin{aligned}
&\rho \in \{\xi, \eta\} \\
&W \equiv W^u(\xi,\eta) = \sum_{k=0}^{n_1}\sum_{l=0}^{n_2} N_{kl}^{p,q}(\xi,\eta) w_{kl}^u \\
&W_{,\rho} \equiv \frac{\partial W^u(\xi,\eta)}{\partial \rho} \\
&W_{,\rho\rho} \equiv \frac{\partial^2 W^u(\xi,\eta)}{\partial \rho^2} \\
&N_L \equiv N_{ij}(\xi,\eta) \\
&u^h \equiv u^h(\xi,\eta) \\
&w_L \equiv w_{ij}^u
\end{aligned}
\tag{B.1}
$$

Having the above expressions, we can find the required sensitivities in (4.17) and (4.18) as described below. The sensitivities of the field variable simplify to

$$
\frac{\partial u^h}{\partial w_L} = \frac{N_L}{W}\left(u^h - u_L\right) \tag{B.2}
$$

Further, the sensitivities of the first order spatial derivatives of the field variable can be derived as

$$
\begin{bmatrix} \dfrac{\partial u_{,x}^h}{\partial w_L} \\ \dfrac{\partial u_{,y}^h}{\partial w_L} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial^2 u^h}{\partial x \partial w_L} \\ \dfrac{\partial^2 u^h}{\partial y \partial w_L} \end{bmatrix} = (J)^{-1} \begin{bmatrix} \dfrac{\partial^2 u^h}{\partial \xi \partial w_L} \\ \dfrac{\partial^2 u^h}{\partial \eta \partial w_L} \end{bmatrix} \tag{B.3}
$$

where

$$
\frac{\partial^2 u^h}{\partial \rho \partial w_L} = \frac{1}{W}\left[\left(\frac{N_L}{W} W_{,\rho} - \frac{\partial N_L}{\partial \rho}\right)(u^h - u_L) - N_L \frac{\partial u^h}{\partial \rho}\right] \tag{B.4}
$$

By having the above relations, we can evaluate the sensitivities of the second order spatial derivatives of the field variable as follows

$$\left\{\begin{array}{c}\dfrac{\partial u^h_{,xx}}{\partial w_L}\\ \dfrac{\partial u^h_{,xy}}{\partial w_L}\\ \dfrac{\partial u^h_{,yy}}{\partial w_L}\end{array}\right\}=\left\{\begin{array}{c}\dfrac{\partial^3 u^h}{\partial x^2\partial w_L}\\ \dfrac{\partial^3 u^h}{\partial x\partial y\,\partial w_L}\\ \dfrac{\partial^3 u^h}{\partial y^2\partial w_L}\end{array}\right\}=(J^{II})^{-1}\left\{\begin{array}{l}\dfrac{\partial u^h_{,\xi\xi}}{\partial w_L}-\dfrac{\partial u^h_{,x}}{\partial w_L}x_{,\xi\xi}-\dfrac{\partial u^h_{,y}}{\partial w_L}y_{,\xi\xi}\\ \dfrac{\partial u^h_{,\xi\eta}}{\partial w_L}-\dfrac{\partial u^h_{,x}}{\partial w_L}x_{,\xi\eta}-\dfrac{\partial u^h_{,y}}{\partial w_L}y_{,\xi\eta}\\ \dfrac{\partial u^h_{,\eta\eta}}{\partial w_L}-\dfrac{\partial u^h_{,x}}{\partial w_L}x_{,\eta\eta}-\dfrac{\partial u^h_{,y}}{\partial w_L}y_{,\eta\eta}\end{array}\right\} \tag{B.5}$$

where the required sensitivities of the higher order parametric derivatives of the field variable can be computed using the following expressions

$$\frac{\partial u^h_{,\rho\rho}}{\partial w_L}=\frac{\partial^3 u^h}{\partial\rho^2\partial w_L}=\frac{1}{W^2}\left\{\begin{array}{l}\left[2\left(\dfrac{\partial N_L}{\partial\rho}-\dfrac{N_L}{W}W_{,\rho}\right)W_{,\rho}+N_LW_{,\rho\rho}-W\dfrac{\partial^2 N_L}{\partial\rho^2}\right]\left(u^h-u_L\right)\\ +2\left(N_LW_{,\rho}-W\dfrac{\partial N_L}{\partial\rho}\right)\dfrac{\partial u^h}{\partial\rho}\end{array}\right\}-\frac{N_L}{W}\left(\frac{\partial^2 u^h}{\partial\rho^2}\right) \tag{B.6}$$

and

$$\begin{aligned}\frac{\partial u^h_{,\xi\eta}}{\partial w_L}=\frac{\partial^3 u^h}{\partial\xi\,\partial\eta\,\partial w_L}=\frac{1}{W^2}&\left\{\begin{array}{l}\left[\left(\dfrac{\partial N_L}{\partial\eta}-\dfrac{2N_L}{W}W_{,\eta}\right)W_{,\xi}+N_LW_{,\xi\eta}+W_{,\eta}\dfrac{\partial N_L}{\partial\xi}-W\dfrac{\partial^2 N_L}{\partial\xi\partial\eta}\right]\left(u^h-u_L\right)+\\ \left[\left(N_LW_{,\xi}-W\dfrac{\partial N_L}{\partial\xi}\right)\dfrac{\partial u^h}{\partial\eta}-\left(N_LW_{,\eta}-W\dfrac{\partial N_L}{\partial\eta}\right)\dfrac{\partial u^h}{\partial\xi}\right]\end{array}\right\}\\ &-\frac{N_L}{W}\left(\frac{\partial^2 u^h}{\partial\xi\partial\eta}\right)\end{aligned} \tag{B.7}$$