Representation and application of spline-based finite elements

Tatiana Kravetc

Department of Computer Science and Computational Engineering
Faculty of Engineering Science and Technology
UiT – The Arctic University of Norway

Preface

This thesis has been carried out at UiT – The Arctic University of Norway, campus Narvik, as a partial fulfillment of the requirements for the degree of Philosophiae Doctor (Ph.D.). The research has been conducted between 2016 and 2019 under the supervision of Prof., Dr.Ing. Børre Bang and co-supervision of Ph.D. Rune Dalmo.

The work presented in the thesis is a manuscript which resulted from individual work under appropriate supervision. The first chapter provides background information and objectives of the work. Then follows the main part which explains significant results. The concluding chapter gives some remarks and notes for future research.

Acknowledgments

Foremost, I would like to express my gratitude to my main supervisor Børre Bang for his endless patience, sharing of knowledge, support and guidance of my Ph.D. study and research. My co-supervisor, Rune Dalmo, has provided insights, valuable help, his expertise and deep involvement in both theoretical and practical aspects of this research.

Besides my supervisors, I would like to thank Lubomir Dechevsky. Without his immense contribution to the development of the theory of expo-rational B-splines this research would not have been realized.

I am grateful to Arne Lakså and Peter Zanaty for sharing their related scientific materials. My sincere thanks also goes to my other colleagues and friends for motivating discussions and providing constructive feedback during the period of this work. I also thank Ingrid Howes for proofreading.

Last but not least, I would like to thank my parents, Tatiana and Valerii, for believing in me and supporting me throughout my life.

Abstract

Isogeometric analysis, as a generalization of the finite element method, employs spline methods to achieve the same representation for both geometric modeling and analysis purpose. Being one of possible tools in application to the isogeometric analysis, blending techniques provide strict locality and smoothness between elements. Motivated by these features, this thesis is devoted to the design and implementation of this alternative type of finite elements.

This thesis combines topics in geometry, computer science and engineering. The research is mainly focused on the algorithmic aspects of the usage of the spline-based finite elements in the context of developing generalized methods for solving different model problems.

The ability for conversion between different representations is significant for the modeling purpose. Methods for conversion between local and global representations are presented.

Contents

List of Symbols

General	
A^{T}	matrix transpose
A^{-1}	matrix inverse
$D^{\alpha}B$	derivative of the function B
$H^1(\Omega)$	Hilbert space of square-integrable functions
$L^p(\Omega)$	Lebesgue space of functions
$W_p^k(\Omega)$	Sobolev space of functions
C^d	differentiability class
$\|\| \cdot \|\|$	L^2 norm
(x, y)	Cartesian coordinates
$(t)/(u, v)/(u_1, u_2, u_3)$	parameters on the one-/two-dimensional /triangular element
$(\sigma)/(\xi, \eta)$	local parameters on the one-/two-dimensional element
m	number of elements on the domain
n	number of basis functions
Φ	1D parametric domain
Ω	2D domain
Θ	2D parametric domain
$\partial\Omega$	domain boundary
`T`, `E`	connectivity and edge matrices
`l2b`	local-to-global mapping
$\mathbf{H} = \{H_i\}_{i=1}^n$	set of basis functions on the entire domain
H^e	set of basis functions on an element
H	one basis function on the entire domain
$\mathbf{c} = \{[c^x, c^y]_i\}_{i=1}^n$	control points of the tensor product surface
$C(u, v)$	tensor product surface
$\mathbf{E}^e$	extraction operator on an element

B-splines

p	B-spline degree
$\mathbf{N} = \{N_i\}_{i=1}^{n_N}$	set of B-spline basis functions
$\mathbf{P} = \{P_i\}_{i=1}^{n_N} = \{[p^x, p^y]_i]\}_{i=1}^{n_N}$	B-spline coefficients
$S(u, v)$	B-spline tensor product surface

Blending splines/surfaces

d	local geometry degree
ℓ	local patch
$\mathbf{W}_d = [b_{d,0}, \ldots, b_{d,d}]$	Bernstein polynomial basis of degree d
$\mathbf{B}$	set of symmetric ERBS basis functions
$\mathbf{G} = \{G_i\}_{i=1}^{n_G}$	set of combined expo-rational basis functions
$G^e(\xi, \eta)$	set of $(d+1)^2$ basis functions on the one corner of an element
$\mathbf{Q} = \{Q^k\}_{k=0}^{m} = \{[q^x, q^y]_i\}_{i=1}^{n_G}$	set of coefficients of the local geometry
$A(t)/A(u, v)$	blending spline/surface

Triangulations

K	triangular element
$\triangle$	Bézier triangle domain
(u, v, w)	barycentric coordinates on the local Bézier triangle
$W_d^K = [b_{0,0,d}^d, b_{0,1,d-1}^d, ..., b_{d,0,0}^d]$	Bernstein polynomial basis of degree d on the triangle K
$\beta_1, \beta_2, \beta_3$	expo-rational basis on triangulation
$G^K = \{G_i^K\}_{i=1}^{n_G^K}$	set of combined expo-rational basis functions on the triangle K
$Q^K = \{[q^x, q^y]_i\}_{i=1}^{n_G^K}$	set of coefficients of local triangles for the ERBS triangle on the domain K
$A(u_1, u_2, u_3)$	ERBS triangle

Finite elements

$\{\Omega^e\}_{e=1}^m$	set of finite elements
$\{\gamma_1, ..., \gamma_s\}$	set of edges belonging to the domain boundary
$\mathcal{A}$	stiffness matrix
$\mathcal{M}$	mass matrix
b	load (force) vector
$\mathcal{R}, r$	Robin matrix and Robin vector
J, $\lvert J\rvert$	Jacobi matrix, Jacobian
h	mesh width
Υ, Υ_g	test space and trial space
Υ_h, Ψ_h	finite element collections of trial solutions
υ, ψ	test functions

U, U_h	parametric representation of the domain and its finite element approximation
ϑ, ϑ_h	trial and finite element trial solutions
$\zeta = \{\zeta\}_{i=1}^{n_G}$	coefficients of the solution
g_D	Dirichlet boundary data
g_N	Neumann boundary data
f	external load (force)
κ	penalty constant
ε	error

Chapter 0

Introduction

0.1 Background

0.1.1 Historical notes

Discrete and continuous representation of the properties of the world initially appeared as separate and independent of each other. Subsequently, the unity of these opposites allowed scientists to produce significant progress in solving applied engineering problems.

For the explicit representation of continuum properties special mathematical concepts were developed. One of these concepts is the finite element method (FEM).

Eventually the discretization concept has been transformed to a representation of information in digital computers. To be able to solve continuous problems, mathematicians and engineers propose various discretization methods which involve approximation. Engineering problems often require a linkage between geometrical model and computational process.

Splines play an important role in geometric modeling and approximation theory. They are used in data fitting [4, 45], computer aided geometric design (CAGD) [44, 36, 43], computer graphics [47], etc. Schoenberg [87] has shown that splines have powerful approximation properties. Subsequently, spline techniques became popular for a broad scope of applications. Most of the graphic software built today are based on de Boor's [24, 21, 23, 22], Bézier's [2, 3] and de Casteljau's [25] concepts and algorithms. B-splines became a standard tool for approximation techniques, geometry processing and many other areas.

The finite element method is the most successful technique for numerical simulations in engineering and applied mathematics. Its practical usage in the computer program development has been exploited much later than the fundamental mathematical concept was established by Ritz [82], Rayleigh [81], and Bubnov [11, 50], Galerkin [49]. The classical Rayleigh-Ritz method represents a variational approach, by which a solution of the differential problem is approximated by a combination of admissible functions and coefficients. The Bubnow-Galerkin method approximates solutions of boundary value problems directly, without using the variational formulation.

As an extension of FEM, several methods were proposed. In the partition of unity finite element method (PUFEM) [61] and stable generalized finite element method (SGFEM) [58] the trial space of standard finite element method is augmented with non-polynomial shape functions with compact support. Spline-based methods for solving partial differential equations were proposed, for instance, in [80]. As an alternative to mesh-based finite elements, a mesh-free scheme based on web-splines was introduced by Höllig [62] with an extension to the isogeometric analysis in [55]. This technique has a number of algorithmic advantages, which follow from the use of a uniform grid and hierarchical bases.

Isogeometric analysis is a generalization of standard finite elements. It was introduced in [57] and described in detail in [17]. The main idea of the isogeometric concept is that the same basis functions are used for both geometry description and analysis [15]. Instead of standard polynomial elements, which typically give continuous but not smooth solution, the isogeometric representation is typically smooth. The main goal is to be geometrically exact no matter how coarse the discretization. The usage of smooth basis is efficient in many areas including turbulence, thin shell analysis, structural analysis and fluid mechanics. A primary similarity between these areas is the sensitivity to geometry, i.e. small geometric imperfections lead to significant inaccuracies in the computed characteristics.

Expo-rational B-splines were presented for the first time in [27, 70] as an alternative tool for CAGD. Besides ERBS, explored in [29, 71], generalized expo-rational B-splines [28] and logistic expo-rational B-splines [32] constitute a family of blending type spline constructions. This construction possesses C^∞ expo-rational functions with minimal local support as a basis. The main difference between ERBS and polynomial B-splines is that the spline coefficients are represented as local geometries instead of ordinary control points. Moreover, blending splines contribute flexible manipulation of the geometry by affine transformations of local functions. Application of blending splines in the setting of solving partial differential equation in general fits into the partition of unity method [98]. We utilize a simple version of expo-rational basis functions [69] in the current research. A short overview of ERBS, blending splines, tensor product surfaces and ERBS triangles is given in Section 0.1.2.

Adaptive spatial resolution of the solution field is essential in finite element analysis. There are a number of refinement algorithms developed for standard finite elements, such as a Delaunay refinement algorithm [84] and Rivara refinement algorithm [83] for triangulations, which are investigated, for example, in [39]. On tensor-product-based meshes hierarchical B-splines [54] are common in adaptive mesh refinement. A subdivision projection technique [8] facilitates their implementation.

In addition, approaches to local refinement of B-splines can be based on knot insertion [5, 16]. Most of these methods are applicable to ERBS blending spline construction [19] with an extension such that the local function for a new knot is expressed in terms of existing local curves. Moreover, in contrast to B-splines, knot insertion strategies in application to ERBS possess a local effect, i.e. inserting of a new knot affects only a few knot intervals.

In the isogeometric analysis framework, the mesh density can be adopted for obtaining an optimal solution [97]. Blending splines can also support this type of refinement by redistribution not only control points, but local surfaces. Furthermore, the complete expo-rational basis functions [69] support adjustment of extra parameters, which modify a shape of the basis function and, consequently, the density of the parameter lines.

In the present thesis we point our attention to the extraction operator. In general, extraction techniques are based on local representation of smooth basis functions in terms of C^0 polynomials, which provide an element structure for the efficient implementation of isogeometric analysis. In this context, Bézier extraction was introduced first. In this instance, Bézier extraction operator maps a piecewise Bernstein polynomial basis onto a non-uniform B-spline basis [6]. On volumes, Bézier extraction was examined in [96]. Such approach was later generalized to T-splines [7], hierarchical B-splines [85], hierarchical T-splines [40], LR-splines [35]. A concept of Lagrange extraction was introduced in [86].

The Lagrange extraction establishes a link between splines and nodal finite element basis functions.

0.1.2 Introduction to splines

Both spline curves and surfaces are usually represented as a linear combination of control points and basis functions. In the current work we mostly examine surfaces, while one-dimensional examples are more illustrative. Next we introduce notations and common formulations on the example of parametric surfaces, which can be easily reduced to 1D when needed. In addition, we consider some common properties of basis functions, particularly useful for implementing finite element algorithms.

Let $\Theta = [0,1] \times [0,1]$ be a parametric domain with two independent parameters (u, v). The parametric domain Θ is divided into m parts, where $m = m_u m_v$, are denoted as Θ^e, $e = 1, 2, ..., m$ and are called *elements*. The tensor product surface is a mapping $C : \Theta \subset \mathbb{R}^2 \to \mathbb{R}^3$.

The reason for parameterizing the domain is that it allows for a simple evaluation of piecewise function spaces on this domain globally. On each parametric element Θ^e we define a set of basis functions. The construction of such functions requires also continuity between neighboring elements. Any basis function is uniquely defined on the entire domain. Two sets of basis functions are defined to construct a tensor product surface: $\{H_i\}_{i=1}^{n_u}$ in the u direction and $\{H_j\}_{j=1}^{n_v}$ in the v direction.

To form a basis in the u direction, the functions H_i, $i = 1, ..., n_u$, satisfy the requirements:

- they constitute a partition of unity, i.e $\forall u$

$$\sum_{i=1}^{n_u} H_i(u) = 1;$$

- for any knot interval, the non-zero basis functions on this interval are linearly independent.

These properties above of univariate basis can be directly extended to bivariate basis. The general formula of a tensor product surface is

$$C(u, v) = \sum_{i=1}^{n_u} \sum_{j=1}^{n_v} c_{ij}\, H_i(u)\, H_j(v), \tag{0.1.1}$$

where c_{ij}, $i = 1, ..., n_u$, $j = 1, ..., n_v$, are control points, which are represented as a control net.

In order to generalize the computational approach for one- and two-dimensional problems, we reproduce the detailed formula (0.1.1) in a compact matrix representation

$$C = \mathbf{c}^{\mathrm{T}}\, \mathbf{H}. \tag{0.1.2}$$

Thus, the formula (0.1.2) can be used for both curves and surfaces. A set of coefficients $\mathbf{c}$ corresponds to an ordered set of basis functions $\mathbf{H}$. For surfaces, the vector of basis functions $\mathbf{H}$ has $[1 \times n_u n_v]$ elements, and each function depends on u and v.

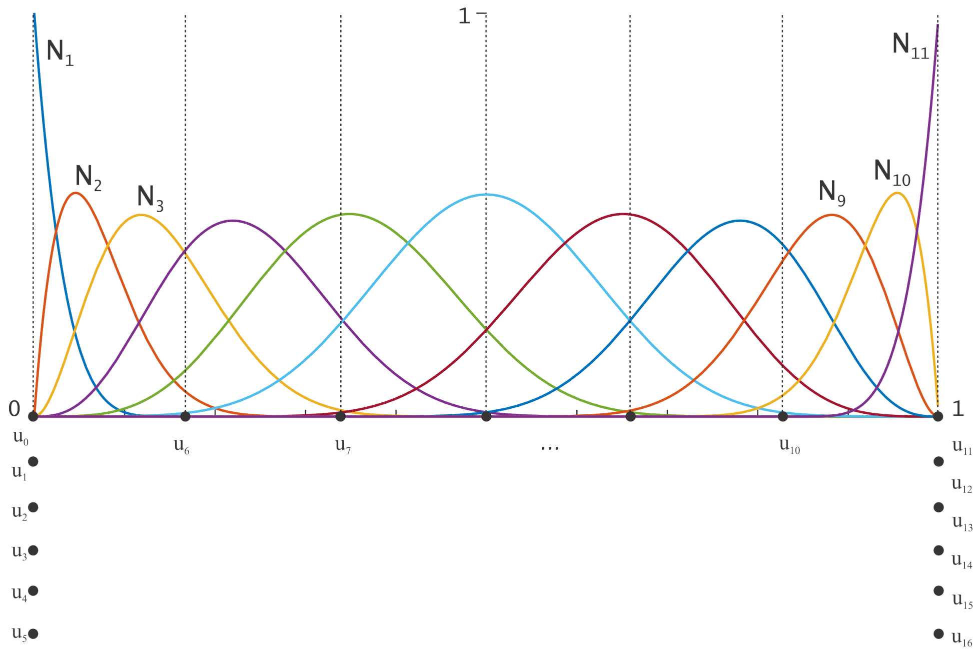


Figure 0.1.1: The B-spline basis functions $\mathbf{N}(u) = \{N_i(u)\}_{i=1}^{11}$ of degree five. The knot vector $\{u_j\}_{j=0}^{16}$ has six ending multiple knots on both sides.

B-splines

A B-spline curve of polynomial degree p is defined by a linear combination of n_u B-spline basis functions and n_u control points. We denote a set of B-spline basis functions as $\mathbf{N}(u) = \{N_i(u)\}_{i=1}^{n_u}$. The corresponding set of vector-valued control points is denoted as $\mathbf{P} = \{P_i\}_{i=1}^{n_u}$, where each point belongs to the real coordinate space of a certain dimension.

Let us define the knot vector $\{u_i\}_{i=0}^{n_u+p} = \{u_0, u_1, ..., u_{n_u}, ..., u_{n_u+p}\}$, where the first and the last $p+1$ knots are equal (i.e., $u_0 = u_1 = ... = u_p$ and $u_{n_u} = u_{n_u+1} = ... = u_{n_u+p}$), as shown in Figure 0.1.1. A recursive formulation of B-spline basis functions can be obtained by the de Boor's algorithm [24].

Definition 0.1.1. For the knot vector $\{u_j\}_{j=0}^{n_u+p}$, the B-spline basis functions are defined recursively starting with piecewise constant $(p = 0)$

$$N_{i,0}(u) = \begin{cases} 1 & \text{if } u_i \leq u < u_{i+1}, \\ 0 & \text{otherwise.} \end{cases}$$

For $p = 1, 2, 3, ...$, they are defined by

$$N_{i,p}(u) = \frac{u - u_i}{u_{i+p} - u_i} N_{i,p-1}(u) + \frac{u_{i+p+1} - u}{u_{i+p+1} - u_{i+1}} N_{i+1,p-1}(u). \tag{0.1.3}$$

On the element level we define a vector of basis functions $\mathbf{N}^e_{j,p}$, where e is an element index, j is a knot index, $j = p, ..., n_u + p$, p is a spline degree.

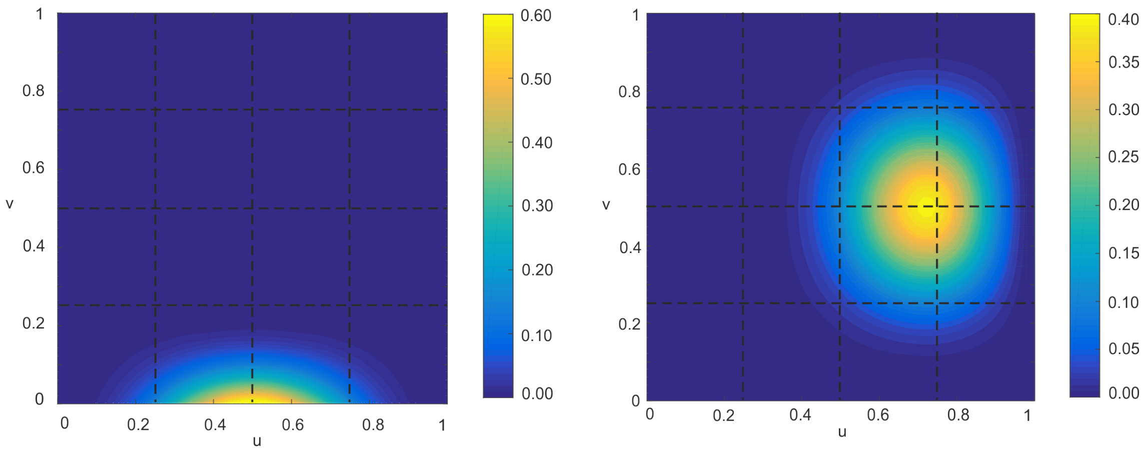


Figure 0.1.2: A contour plot of two bicubic B-spline basis functions: the left one on the boundary, the right one inside the parametric domain Θ.

Each B-spline basis function has a local support, i.e. is different from zero only on the interval $u \in [u_{j-p}, u_j)$. On the element level these functions are established by the multiplication of the factor matrices [88] up to degree p

$$\mathbf{N}^e_{j,p} = \begin{bmatrix} 1-\omega_{1,j}(u) & \omega_{1,j}(u) \end{bmatrix} \begin{bmatrix} 1-\omega_{2,j-1}(u) & \omega_{2,j-1}(u) & 0 \\ 0 & 1-\omega_{2,j}(u) & \omega_{2,j}(u) \end{bmatrix} \cdots$$
$$\cdots \begin{bmatrix} 1-\omega_{p,j-p+1}(u) & \omega_{p,j-p+1}(u) & \dots & 0 \\ 0 & 1-\omega_{p,j-p+2}(u) & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & \dots & 1-\omega_{p,j}(u) & \omega_{p,j}(u) \end{bmatrix} \qquad (0.1.4)$$

where

$$\omega_{\iota,j}(u) = \begin{cases} \dfrac{u-u_j}{u_{j+\iota}-u_j}, & \text{if } u_j \le u \le u_{j+\iota} \\ 0, & \text{otherwise.} \end{cases}$$

is a local/global translation and scaling function.

The simplest possibility [62] of obtaining the μ-variate basis functions is to form tensor product of uniform B-splines.

Definition 0.1.2. A set of μ-variate B-spline basis functions is defined as

$$\mathbf{N}_p(u_1, ..., u_\mu) = \prod_{\nu=1}^{\mu} \mathbf{N}_{p_\nu}(u_\nu), \qquad (0.1.5)$$

where p_ν is the degree in the ν^{th} variable, with the convention that $p_1 = ... = p_\mu$ unless explicitly stated otherwise.

By evaluating the B-spline basis in both directions u and v using the formula (0.1.3), we obtain the bivariate basis with $\mu = 2$ by formula (0.1.5). Let us agree that $p_u = p_v = p$. An example of the bivariate B-spline basis function is shown in Figure 0.1.2.

First-order partial derivatives of the bivariate B-spline basis are differences of lower degree B-splines basis functions. Because of the product structure of multivariate B-splines, all univariate identities and algorithms generalize easily [62]. For example, the derivative of $N_{i,j,p}(u,v)$ with respect to u is

$$D_u N_{i,j,p}(u,v) = p\left(\frac{N_{i,p-1}(u)}{u_{i+p}-u_i} - \frac{N_{i+1,p-1}(u)}{u_{i+1+p}-u_{i+1}}\right) N_{j,p}(v). \tag{0.1.6}$$

In matrix notations we represent (0.1.6) as $D_u\mathbf{N} = D_u\mathbf{N}_p^{\mathrm{T}}(u)\,\mathbf{N}_p(v)$. Similarly, the partial derivative of $\mathbf{N}$ with respect to v is $D_v\mathbf{N} = \mathbf{N}_p^{\mathrm{T}}(u)\,D_v\mathbf{N}_p(v)$.

In order to implement the generalized formula (0.1.2) we reconstruct the set of μ-variate B-spline basis functions from the element level, where the size of the vector of basis functions on the element is equal to $(p+1)^\mu$, to a set $\mathbf{N}$ of continuous basis functions on the entire domain, which size is equal to $\prod\limits_{\nu=1}^{\mu} n_\nu$, where n_ν is a number of basis functions in the ν direction, and each basis function is determined on the entire domain.

A thorough study of B-splines and spline methods can be found in [74]. In the following we refer to some basic properties of B-splines:

- The support of each B-spline basis function $N_{i,j,p}(u,v)$ is compact and contained in the subdomain $[u_i, u_{i+p+1}] \times [v_j, v_{j+p+1}]$.
- B-spline basis functions are positive on their local support.
- The B-spline basis of degree p is $(p-1)$-times continuously differentiable.
- The construction of B-splines produces piecewise polynomials.

Expo-rational B-splines

We now consider some of the theory of blending type spline constructions, which is relevant for this work. A comprehensive study of the expo-rational B-splines (ERBS) can be found in [29, 69].

Definition 0.1.3. The simple version of an expo-rational basis function associated with the strictly increasing knots t_{k-1}, t_k and t_{k+1} is defined as follows

$$B_k(t) = \begin{cases} \Gamma_{k-1}\displaystyle\int_{t_{k-1}}^{t} \phi_{k-1}(s)ds, \text{ if } t_{k-1} < t \le t_k, \\ \Gamma_k \displaystyle\int_{t}^{t_{k+1}} \phi_k(s)ds, \text{ if } t_k < t < t_{k+1}, \\ 0, \text{ otherwise} \end{cases}$$

where

$$\phi_k(t) = \exp\left(-\frac{\left(t - \dfrac{t_k+t_{k+1}}{2}\right)^2}{(t-t_k)(t_{k+1}-t)}\right),$$

and the scaling factor

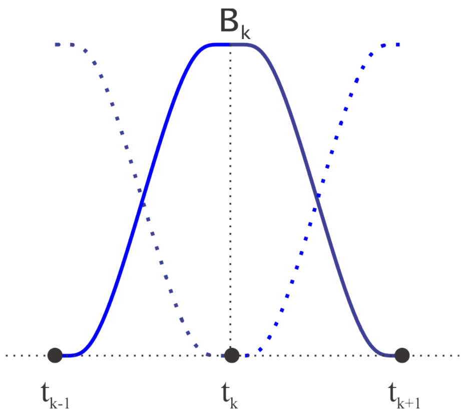


Figure 0.1.3: The expo-rational basis function B_k over the knot interval $[t_{k-1}, t_{k+1})$. The dotted curves show the halves of the corresponding basis functions B_{k-1} and B_{k+1}.

$$\Gamma_k = \frac{1}{\int_{t_k}^{t_{k+1}} \phi_k(t)dt}.$$

An example of the ERBS basis function over the knot interval $[t_{k-1}, t_{k+1})$ is shown in Figure 0.1.3. One can see that this basis is strictly local and symmetric.

The derivative of an expo-rational basis function is

$$DB_k(t) = \begin{cases} \Gamma_{k-1}\phi_{k-1}(t), \text{ if } t_{k-1} < t \leq t_k, \\ -\Gamma_k\phi_k(t), \text{ if } t_k < t < t_{k+1}, \\ 0, \text{ otherwise} \end{cases} \tag{0.1.7}$$

Some of the important properties [32] of ERBS are

- providing C^∞-smooth partition of unity on $\mathbb{R}$;
- minimal support of the basis functions;
- vanishing derivatives at the knots.

A blending curve is a linear combination of local curves and corresponding basis functions. Bézier curves of a certain degree d are one type of local curve used.

Definition 0.1.4. The general formula for an expo-rational B-spline curve [29] over the knot vector $\{t_k\}_{k=0}^{m+1}$ is

$$A(t) = \sum_{k=1}^{m+1} \ell_k B_k(t), \tag{0.1.8}$$

where the coefficients ℓ_k are the local functions, and $B_k(t)$ are the expo-rational basis functions, defined by (0.1.9).

The first derivative of the blending curve (0.1.8) is

$$DA(t) = \sum_{k=1}^{m+1} (D\ell_k \, B_k(t) + \ell_k \, DB_k(t)).$$

The local functions ℓ_k are scaled and translated to the interval $[t_{k-1}, t_{k+1}]$. For this, we introduce a local/global affine mapping $\omega_k(t)$, which scales the support of local curves ℓ_k to the interval $[t_{k-1}, t_{k+1}]$.

$$\omega_k(t) = \begin{cases} \dfrac{t - t_{k-1}}{t_{k+1} - t_{k-1}}, & t_{k-1} < t \le t_{k+1}, \\ 0, & \text{otherwise}. \end{cases} \tag{0.1.9}$$

In the following we employ Bézier curves and surfaces as local geometry. The local Bézier curve ℓ_k, $k = 1, ..., m+1$ is defined as

$$\ell_k(t) = \sum_{\iota=0}^{d} q_\iota^k b_{d,\iota}(\omega_k(t)), \tag{0.1.10}$$

where q_ι^k are the control points of the k^{th} local curve, and $b_{d,\iota}$ are the Bernstein polynomials of degree d, which are obtained by the following general formula

$$b_{d,\iota}(t) = \binom{d}{\iota} t^\iota (1-t)^{d-\iota} = \frac{d!}{\iota!(d-\iota)!} t^\iota (1-t)^{d-\iota}. \tag{0.1.11}$$

The derivatives of the d^{th} degree Bernstein polynomials are polynomials of degree $d-1$ and are given by

$$Db_{d,\iota} = d(b_{d-1,\iota-1} - b_{d-1,\iota}). \tag{0.1.12}$$

An ERBS tensor product surface resembles the usual formula (0.1.1), except that the coefficients are not points, but surfaces. The ERBS tensor product surface is a blending of local patches. This surface with $(m_u + 1) \times (m_v + 1)$ local patches can be divided into $m_u \times m_v$ parts, where each of them is a blending part of four local patches. These parts are hereinafter referred to as "elements".

We consider Bézier surfaces of the bivariate degree d as local patches. The basis for each patch can be evaluated by the tensor product of two Bernstein polynomial bases in the corresponding directions of the local parameters. The Bézier surface can be constructed as a linear combination of $(d+1)^2$ control points and the same number of basis functions. Each local patch is defined on the appropriate subdomain $[u_{k-1}, u_{k+1}] \times [v_{l-1}, v_{l+1}]$, where the corresponding intervals are formed from the knot vectors $\{u_k\}_{k=0}^{m_u+1}$ and $\{v_l\}_{l=0}^{m_v+1}$.

ERBS basis on triangulations

Construction of triangular ERBS patches was presented in [1]. A comprehensive study of barycentric coordinates and Bézier triangles can be found in [68, 42]. A generalization of barycentric coordinates and their applications are presented in [46]. We provide some relevant definitions concerning this topic.

Suppose $\triangle$ is a nondegenerate triangle (with nonzero area) in $\mathbb{R}^2$ with vertices

$$q_a = (x_a, y_a), \qquad a = 1, 2, 3.$$

Definition 0.1.5. Every point $q = (x, y) \in \mathbb{R}^2$ has a unique representation in the form

$$q = uq_1 + vq_2 + wq_3,$$

with

$$u + v + w = 1.$$

The parameters u, v, w are called *the barycentric coordinates* of the point q relative to the triangle $\triangle$.

We now introduce Bernstein basis polynomials of degree d relative to the triangle $\triangle$.

Definition 0.1.6. Let $\triangle$ be a fixed triangle with barycentric coordinates u, v, w for each point $q = (x, y) \in \mathbb{R}^2$. Given nonnegative integers i, j, k, summing up to d, let

$$b^d_{i,j,k} = \frac{d!}{i!j!k!} u^i v^j w^k.$$

The polynomials $b^d_{i,j,k}(u, v, w)$ are called the Bernstein basis polynomials of degree d relative to $\triangle$.

To define a directional derivative of the basis function $b^d_{i,j,k}$ we first introduce a vector in the barycentric coordinates. The vector $\tilde{q}$ is defined by a subtraction of two points $\tilde{q} = q_2 - q_1$ and has the barycentric coordinates $(\tilde{u}, \tilde{v}, \tilde{w})$, where $\tilde{u} + \tilde{v} + \tilde{w} = 0$.

Definition 0.1.7. Suppose $\tilde{q}$ is a vector with barycentric coordinates $(\tilde{u}, \tilde{v}, \tilde{w})$. Then for any integer i, j, k, where $i + j + k = d$

$$D_{\tilde{q}} b^d_{i,j,k}(u, v, w) = d \left(\tilde{u}\, b^{d-1}_{i-1,j,k} + \tilde{v}\, b^{d-1}_{i,j-1,k} + \tilde{w}\, b^{d-1}_{i,j,k-1} \right).$$

The function $D_{\tilde{q}} b^d_{i,j,k}$ is called the directional derivative of the basis function $b^d_{i,j,k}$ in the direction $\tilde{q}$.

The set of Bernstein basis polynomials forms a basis for the Bézier triangle construction.

Definition 0.1.8. Let $q_{i,j,k} \in \mathbb{R}^3$, $i + j + k = d$, be the coefficients of the Bézier triangle and polynomials $b^d_{i,j,k}$ form the Bernstein basis of degree d. Then the surface

$$\ell(u, v, w) = \sum_{i+j+k=d} q_{i,j,k}\, b^d_{i,j,k}(u, v, w), \qquad u + v + w = 1,$$

is called the Bézier triangular surface.

Note that the number of coefficients is equivalent to the number of basis functions and it is equal to $\binom{d+2}{2}$. Also note that there is a specific order of control points for Bézier triangles, and, consequently, for ERBS triangles. Coefficients have an associated set of domain points. For example, for $d = 1$, the domain points coincide with the vertices q_1, q_2, q_3 of the triangle $\triangle$. An ordering for the coefficients and their corresponding domain points is established in [68].

An ERBS triangle is a surface that blends three Bézier triangles of the degree d via expo-rational basis functions. We define a simple version of the underlying basic expo-rational basis function over the formal parameter u_1.

Definition 0.1.9. The underlying basic expo-rational basis function in barycentric coordinates is defined by $B(u_1)$, $u_1 \in (0, 1]$, as follows

$$B(u_1) = \begin{cases} \Gamma \int\limits_0^{u_1} \phi(s)ds, \text{ if } 0 < u_1 \leq 1, \\ 0, \text{ otherwise} \end{cases}$$

where

$$\phi(u_1) = \exp\left(-\frac{\left(u_1 - \frac{1}{2}\right)^2}{u_1(1-u_1)} \right),$$

and the scaling factor

$$\Gamma = \frac{1}{\int\limits_0^1 \phi(u_1)du_1}.$$

A set of such expo-rational functions forms a basis for the blending type surface construction.

Definition 0.1.10. For any point $\nu = (u_1, u_2, u_3)$, satisfying the convexity property, see Definition 0.1.5, a set of expo-rational basis functions in barycentric coordinates is defined as follows

$$\beta_i(\nu) = \frac{B(u_i)}{B(u_1) + B(u_2) + B(u_3)} \qquad \text{for } i = 1, 2, 3,$$

where $B(u_i)$ are as defined by Definition 0.1.9.

Figure 0.1.4 shows a set of expo-rational basis functions on a triangle.

As for the Bernstein basis functions, we define derivatives of expo-rational basis functions in specific directions $\tilde{\nu} = \nu_2 - \nu_1$. Thus, the partial derivatives are necessary to

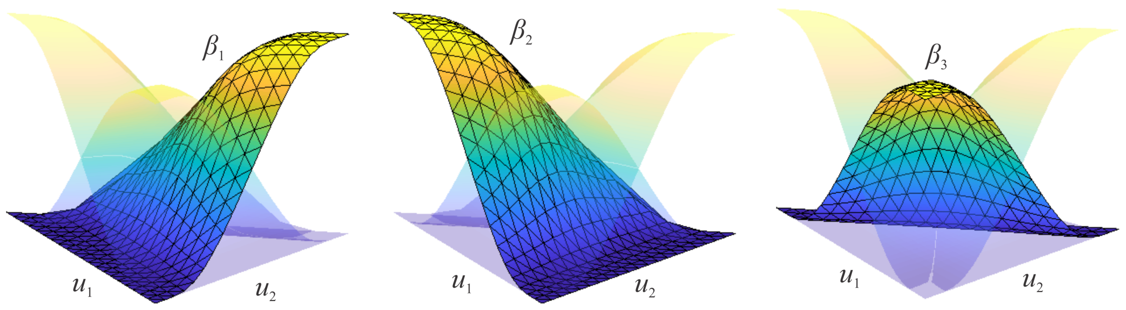


Figure 0.1.4: Expo-rational basis functions in barycentric coordinates.

compute as components. There are two types of formulas for partial derivatives: when we find $D_{u_i}\beta_i(\nu)$

$$D_{u_i}\beta_i(\nu) = DB(u_i)\frac{\sum\limits_{\iota=1}^{3} B(u_\iota) - B(u_i)}{\left(\sum\limits_{\iota=1}^{3} B(u_\iota)\right)^2}, \tag{0.1.13}$$

and when we find $D_{u_j}\beta_i(\nu)$, $j \neq i$

$$D_{u_j}\beta_i(\nu) = DB(u_j)\frac{-B(u_i)}{\left(\sum\limits_{\iota=1}^{3} B(u_\iota)\right)^2}. \tag{0.1.14}$$

Definition 0.1.11. For a given vector $\tilde{\nu} = \nu_2 - \nu_1$ with barycentric coordinates $(\tilde{u}_1, \tilde{u}_2, \tilde{u}_3)$, $\tilde{u}_1 + \tilde{u}_2 + \tilde{u}_3 = 0$, the directional derivatives for the expo-rational basis functions are

$$D_{\tilde{\nu}}\beta_i(\nu) = \sum_{\iota=1}^{3} \tilde{u}_\iota D_{u_\iota}\beta_i(\nu),$$

where $D_{u_\iota}\beta_i(\nu)$ are partial derivatives of the i^{th} expo-rational basis function, found by formulas (0.1.13) or (0.1.14).

A construction of an ERBS triangle is based on a linear combination of three Bézier triangles of degree d and the set of expo-rational basis functions in barycentric coordinates.

Definition 0.1.12. For a set of local Bézier triangles $\ell_i(u_1, u_2, u_3)$, $i = 1, 2, 3$, and corresponding expo-rational basis functions $\beta_i(u_1, u_2, u_3)$, the general formula for the ERBS triangle is

$$A(u_1, u_2, u_3) = \sum_{i=1}^{3} \ell_i(u_1, u_2, u_3)\, \beta_i(u_1, u_2, u_3), \tag{0.1.15}$$

where $u_1 + u_2 + u_3 = 1$ and $u_1, u_2, u_3 \geq 0$.

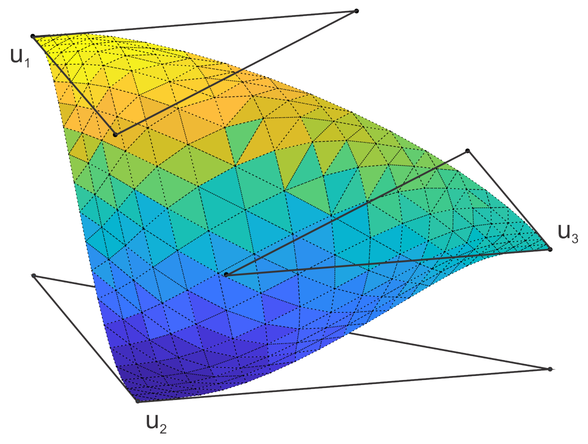


Figure 0.1.5: An example of the ERBS triangle with local Bézier triangles of the first degree.

Figure 0.1.5 shows an example of ERBS triangle with local Bézier triangles of degree 1. This construction is very flexible and can be fitted to a geometry of a relatively high degree of smoothness. The local approximation error of the Hermite interpolation for sufficiently smooth functions is $O(h^{d+1})$, where h is the longest edge of the triangle, and d is the degree of local triangles [1].

Local approximation

The ERBS construction possesses a Hermite interpolation property which follows from the vanishing derivative property. The definition of piecewise Hermite interpolating surfaces at the nodes resembles (0.1.1)

$$F_h = \sum_{j=1}^{n_v} \sum_{i=1}^{n_u} \ell_{ij}\, B_i(u) B_j(v),$$

where the local functions ℓ_{ij} are the Bézier surfaces of degree d.

The order of approximation over the whole support of the C^∞-smooth ERBS is the minimum of the order of the local functions.

All C^d-smooth interpolating blending surfaces F_h of $F \in W_\infty^{d+1}$ on a domain Θ satisfies the following error estimation from the Bramble-Hilbert lemma [9, 14, 62]: there exists a constant independent on the grid width h and F such that

$$||F - F_h||_{W_\infty^\alpha(\Theta)} \leq \mathrm{const}(\Theta, d)|h|^{d+1-\alpha}||F||_{W_\infty^{d+1}(\Theta)}$$

for $0 \leq \alpha \leq d+1$.

Some model examples of local approximation of univariate curves using ERBS are studied in [32].

0.1.3 Basic finite element concept

A comprehensive study of partial differential equations (PDE) can be found in [41, 91]. Fundamentals of the finite element analysis (FEA) are detailed in [94, 75]. A general concept of finite element method (FEM) for the solution of a boundary value problem (BVP) is considered in [100, 90, 56, 10, 13]. A practical approaches to FEM and its implementation are detailed in [73, 64, 79]. The next practical problems were examined: heat conduction [38], vibration of the membrane [52, 76], elasticity problem [64], Helmholtz equation [53, 78], Navier-Stokes equation [72, 12], and some unsolved problems, observed by Zienkiewicz [99].

The main sequence of steps in solving the boundary value problem by the finite element method is:

1. Strong formulation of the problem.
2. Variational statement of the problem.
3. Approximate solution of variational equations through the use of finite element functions.

Strong form of the problem

Let $\Omega \subset \mathbb{R}^2$ be a real connected domain with boundary $\partial\Omega$ and closure $\bar{\Omega} = \Omega \cup \partial\Omega$.

As an example, suppose we want to solve a Poisson's equation of the form

$$-\nabla \cdot (a\,\nabla\vartheta) = f, \quad \text{in } \Omega, \tag{0.1.16}$$

which involves imposing boundary conditions on the function ϑ. With the aim of generalizing the approach we consider the Robin boundary conditions

$$-a\frac{\partial\vartheta}{\partial\overline{n}} = \kappa(\vartheta - g_D) - g_N, \quad \text{on } \partial\Omega, \tag{0.1.17}$$

where $\dfrac{\partial}{\partial\overline{n}}$ is differentiation in the outward normal direction to $\partial\Omega$, g_D, g_N are given functions, κ is a specific constant.

If κ is zero, then we obtain the Neumann boundary condition

$$a\frac{\partial\vartheta}{\partial\overline{n}} = g_N, \quad \text{on } \partial\Omega.$$

A large κ leads to the Dirichlet boundary condition

$$\vartheta = g_D, \quad \text{on } \partial\Omega.$$

Variational form of the problem

To define the variational formulation of the problem, we need to characterize two classes of functions. The first one consists of *trial solutions.* The derivatives of the trial solutions are required to be square integrable. That is, if ϑ is a trial solution, then

$$\int_\Omega |\nabla\vartheta|^2\, d\Omega < \infty.$$

Thus the collection of trial solutions, denoted by Υ_g, consists of all functions which have square-integrable derivatives and vanishes on $\partial\Omega$, i.e. belongs to a Hilbert space $H^1(\Omega)$. This is written as follows

$$\Upsilon_g = \{\vartheta : \vartheta \in H^1(\Omega),\ \vartheta|_{\partial\Omega} = g_D\}.$$

The second class consists of *test functions.* This collection is very similar to the first one except that it is homogeneous on the boundary. This class of functions is denoted by Υ and defined by

$$\Upsilon = \{v : v \in H^1(\Omega),\ v|_{\partial\Omega} = 0\}.$$

Multiplying (0.1.16) with a test function $v \in \Upsilon$ and integrating by Green's formula we have

$$\begin{aligned}\int\limits_{\Omega} f v \, d\Omega = -\int\limits_{\Omega} \nabla \cdot (a\nabla\vartheta) v \, d\Omega = \\ = \int\limits_{\Omega} a\nabla\vartheta \cdot \nabla v \, d\Omega - \int\limits_{\partial\Omega} a\frac{\partial\vartheta}{\partial\overline{n}} v \, d(\partial\Omega) = \\ = \int\limits_{\Omega} a\nabla\vartheta \cdot \nabla v \, d\Omega + \int\limits_{\partial\Omega} (\kappa(\vartheta - g_D) - g_N) v \, d(\partial\Omega).\end{aligned}$$

By collecting the terms we get the following variational formulation: find $\vartheta \in \Upsilon_g$ such that $v \in \Upsilon$ and

$$\int\limits_{\Omega} a\nabla\vartheta \cdot \nabla v \, d\Omega + \int\limits_{\partial\Omega} \kappa\vartheta v \, d(\partial\Omega) = \int\limits_{\Omega} f v \, d\Omega + \int\limits_{\partial\Omega} (\kappa g_D + g_N) v \, d(\partial\Omega). \tag{0.1.18}$$

Spatial discretization

In order to discretize the variational formulation in space, we first construct finite-dimensional approximations of Υ_g and Υ. These classes of functions are denoted by $\Upsilon_{g,h} \subset \Upsilon_g$ and $\Upsilon_h \subset \Upsilon$. Let Υ_h consists of all linear combinations of basis functions $H_i : \bar{\Omega} \to \mathbb{R}$, $i = 1, 2, ..., n$.

Replacing the trial and test spaces by the corresponding finite-dimensional approximations in the variational formulation (0.1.18) we obtain the following finite element method: find $\vartheta_h \in \Upsilon_{g,h}$ such that

$$\int\limits_{\Omega} a\nabla\vartheta_h \cdot \nabla H_i \, d\Omega + \int\limits_{\partial\Omega} \kappa \, \vartheta_h \, H_i \, d(\partial\Omega) = \int\limits_{\Omega} f \, H_i \, d\Omega + \int\limits_{\partial\Omega} (\kappa g_D + g_N) \, H_i \, d(\partial\Omega). \tag{0.1.19}$$

We seek for a discrete solution ϑ_h to the variational formulation as a linear combination of basis functions H_j, $j = 1, 2, ..., n$,

$$\vartheta_h = \sum_{j=1}^{n} \zeta_j H_j. \tag{0.1.20}$$

Substituting the linear combination (0.1.20) into the discrete variational formulation (0.1.19), we define a system of n PDEs for the n coefficients ζ_j, $j = 1, 2, ..., n$. In matrix form we write this as

$$(\mathcal{A} + \mathcal{R})\zeta = b + r, \tag{0.1.21}$$

where the entries of the $n \times n$ stiffness matrix A and the $n \times 1$ force (load) vector b are defined by

$$\mathcal{A}_{ij} = \int\limits_{\Omega} a\,\nabla H_i \cdot \nabla H_j \, d\Omega, \qquad i, j = 1, 2, ..., n, \tag{0.1.22}$$

$$b_i = \int\limits_{\Omega} f\, H_i \, d\Omega, \qquad i = 1, 2, ..., n, \tag{0.1.23}$$

We assemble the $n \times n$ boundary matrix R and the $n \times 1$ boundary vector r containing the integrals originating from the Robin boundary condition (0.1.17). These entries are given by

$$\mathcal{R}_{ij} = \int\limits_{\partial\Omega} \kappa\, H_i H_j \, d(\partial\Omega), \qquad i, j = 1, 2, ..., n \tag{0.1.24}$$

and

$$r_i = \int\limits_{\partial\Omega} (\kappa g_D + g_N) H_i \, d(\partial\Omega), \qquad i = 1, 2, ..., n. \tag{0.1.25}$$

The approach described above can be modified or extended depending on the problem. For example, in the case of a time-dependent problem, time discretization needs to be provided. In some specific cases, considered in the present work, *the mass matrix* $\mathcal{M}$ appears. It has the following general form

$$\mathcal{M}_{ij} = \int\limits_{\Omega} H_i H_j \, d\Omega, \qquad i, j = 1, 2, ..., n. \tag{0.1.26}$$

Algorithm

We now describe a basic algorithm for the finite element method. The algorithm implies solving the discretized variational formulation of some PDE, consisting of combination of mass, stiffness matrices, force vector, and corresponding boundary matrices and vectors. Any type of basis functions can be used on the parameterized domain. On the triangulated domain we use local basis functions, i.e. each basis function has its support only on a set of triangles belonging to one node.

Given a domain $\Omega \subset \mathbb{R}^2$. If this domain can be parameterized and represented as a tensor product surface, one can choose rectangular elements. Otherwise, the domain can be triangulated. The domain is divided into m finite elements Ω^e, $e = 1, ..., m$. A set of elements is called *a mesh.*

Since one basis function covers a small number of elements, an element of each integral (0.1.22)-(0.1.26) with subscript "ij" for matrices and "i" for vectors can be represented as a local element matrix or vector with superscript "e". From this it follows that global matrices and vectors can be constructed by summing the contributions of element matrices and vectors. The same approach applies to the boundary matrix $\mathcal{R}$ and vector r, where many of their entries are zero except boundary elements.

A transformation of the integrals (0.1.22)-(0.1.26) into a parametric domain is considered in the main part of the present thesis in relation to specific smooth basis functions. Curvilinear elements are called *isoparametric.* Construction of C^0 two- and three-dimensional triangular and rectangular isoparametric elements, coordinate transformations and numerical integrations are detailed in [100].

Let us assume that we seek the discretized solution ϑ_h, which can be represented in terms of basis functions $\mathbf{H}$ and coefficients ζ, as formula (0.1.20) shows. We find the coefficients ζ by solving the linear matrix equation $\boldsymbol{\Xi}_\Omega(\mathcal{A}, \mathcal{M}, b, \mathcal{R}, r)$ (for instance (0.1.21)) obtained by the discretization of the variational formulation of the problem.

Note that spaces, which describe the geometry of the domain and the solution are independent of each other, although they are approximated using the same basis functions.

Algorithm 1 Basic Finite Element Method Algorithm

```
1: Given functions $f$ and $a$, a constant $\kappa$, and functions $g_N$ and $g_D$, which describe
   Neumann and Dirichlet boundary conditions, respectively. These functions are defined
   on the domain $\Omega$.
2: Create a mesh with $m$ rectangular or triangular elements.
3: Define the corresponding space of $n$ continuous basis functions $\mathbf{H} = \{H_i\}_{i=1}^n$.
4: Approximate the domain $\Omega$ by using the basis $\mathbf{H}$. Any approximation method can be
   used.
5: Allocate space for global $n \times n$ mass, stiffness and Robin matrices $\mathcal{M}$, $\mathcal{A}$ and $\mathcal{R}$, and
   $n \times 1$ force and Robin vectors $b$ and $r$, and define them to zero.
6: for $e = 1, 2, ..., m$ do
7:     Compute the element mass and stiffness matrices $\mathcal{M}^e$ and $\mathcal{A}^e$, and the element
       force vector $b^e$ with entries
       $\mathcal{M}^e = \int\limits_\Omega (\mathbf{H}^e)^{\mathrm{T}} \mathbf{H}^e \, d\Omega, \ \mathcal{A}^e = \int\limits_\Omega a (\nabla \mathbf{H}^e)^{\mathrm{T}} \nabla \mathbf{H}^e \, d\Omega,$
       $b^e = \int\limits_\Omega f \, \mathbf{H}^e \, d\Omega.$
8:     Compute the $n \times n$ Robin matrix $\mathcal{R}$, and the $n \times 1$ Robin vector $r$ with entries
       $\mathcal{R}^e = \int\limits_{\partial\Omega} \kappa \, (\mathbf{H}^e)^{\mathrm{T}} \mathbf{H}^e \, d(\partial\Omega), \ r^e = \int\limits_{\partial\Omega} (\kappa g_D + g_N) \mathbf{H}^e \, d(\partial\Omega).$
9: end for
10: Assemble the global matrices $\mathcal{M}$, $\mathcal{A}$, $\mathcal{R}$ and the global vectors $b$, $r$ of the element
    matrices.
11: Solve the linear system
        $\zeta = \boldsymbol{\Xi}_\Omega(\mathcal{A}, \mathcal{M}, b, \mathcal{R}, r),$
    which represents a discretized variational formulation.
12: Approximate the solution
        $\vartheta_h = \zeta^{\mathrm{T}} \, \mathbf{H}.$
```

0.1.4 ERBS as a basis for analysis

An analysis framework based on ERBS consists of the following items and features:

1. A mesh for a tensor product blending surface is defined by the product of knot vectors. Knot intervals subdivide the domain into elements.

2. The support of each basis function consists of four elements, where the corresponding local surface is defined.

3. The control points associated with the basis functions define the geometry. The same basis functions are used for representing the solution of the problem of interest (for example, elasticity problem, heat conduction, etc.).

4. A set of local surfaces provides an additional level of abstraction between control points and elements. Local surfaces interpolate finite elements while being constructed by control points. The finite elements are independent of each other and smoothly connected at the same time.

5. Adjustable mesh refinement can be achieved by knot insertion. New local surfaces are expressed in terms of existing local surfaces.

6. On a triangular mesh basis functions are defined separately, but they have continuous structure over the mesh.

7. ERBS triangles allow us to build a complex shape domain on a coarse initial discretization. Smooth boundary and flexible parameterization can be obtained.

8. Despite the use of standard FEM algorithms, which are common for any type of smooth basis, the solution obtained by employing the ERBS basis approximation preserves the properties of blending surfaces, such as Hermite interpolatory property, which can be used in the further analysis.

0.2 Objectives and overview

The purpose of the following scientific work is to examine the behavior of ERBS-based finite elements applied to various model problems.

In general, the usage of blending type spline construction in the finite element context does not impose any restrictions on the type of solvable problems. For the purposes of clarify, we specify the problems such as the solution of the PDE is approximated by the surface embedded in $\mathbb{R}^3$, i.e. the domain belongs to $\mathbb{R}^2$, while the target characteristic is mapped onto the third spatial coordinate.

We demonstrate the advantages of the ERBS finite elements compared to B-spline-based elements and standard polynomial finite elements. Standard triangle and rectangular finite elements can be obtained for C^0 continuity [100]. Continuity of the gradient is more difficult to achieve. However, the B-spline basis of degree p is $(p-1)$-times continuously differentiable with discontinuities of the p^{th} derivative at the knot points [62]. The finite elements based on blending splines, in their turn, combine advantages of both these approaches. Local patches facilitate the element-level localty, while their blending provides smoothness. The local surfaces contain positions and derivatives, which vanish at the knot points. This property allows for local Hermite interpolation, and the approximation order of the blending surface agrees with approximation order of the local Bernstein polynomials.

There follows an overview and a short description of the main objectives and contributions.

0.2.1 ERBS extraction

1. *Combined expo-rational basis.* A construction of the combined expo-rational basis allows us to construct blending splines and blending tensor product surfaces as a linear combination of this type of basis and vector-valued coefficients. This basis is a mixture of ERBS basis functions and Bernstein polynomial basis. The combined expo-rational basis aggregates global smoothness and interpolatory property. Since functions included in the basis are symmetric, we suggest a simplified algorithm for construction of the basis, which allows us to be more computationally efficient.

2. *Extraction operator.* ERBS extraction to B-splines formulates locally both blending spline and B-spline representations. This local representation provides an opportunity to compare the resulting approximations based on different types of spline constructions. Extraction operator is based on basis decomposition and follows from the linear independence of the basis. This operator allows for conversion B-spline control points to the local geometry of blending spline construction and vice versa.

0.2.2 Expo-rational finite elements

3. *Tensor product finite elements.* ERBS finite elements provide an additional level of abstraction. While coefficients of the standard finite element coincide with nodal points, and the coefficients of the B-spline tensor product surface affect each element of the corresponding basis support, local surfaces of the blending surface preserve both local manipulation and smoothness of the global surface. Intrinsic properties of the domain can be changed by manipulating the local surfaces.

4. *ERBS triangles as finite elements.* In contrast to tensor product finite elements, ERBS triangles can be connected in an arbitrarily way. The basis functions are constructed separately for each triangular element. Hence, the local triangles have very flexible constructive opportunities. The use of local triangles simplifies manipulation with domain parameterization, with the aim to satisfy the given intrinsic properties. An optimal position of the coefficients is a different algorithmic problem that goes beyond our current research purpose. For instance, smooth constructions on triangulated domains based on conformal mapping [37] were investigated in [31]. However, ERBS triangles basically have C^0 continuity due to the lack of overlapping local triangles between elements.

0.2.3 Numerical experiments

5. *Regression analysis.* The capabilities of local curves are shown on an example of treating data that is possibly noise contaminated which consists of more or less well defined stages. Considered method changes the representation of the raw data to a form that accommodates both local approximation and adjustable criteria for identifying shifts in trends. Blending splines makes it possible to keep the original approximation and gives a gradual refinement that can be used to balance accuracy and computational effort.

6. *L^2-projection.* This method is used for the domain initialization. In particular, it has been chosen to demonstrate capabilities of the one-dimensional extraction operator. All possible conversions between bases and corresponding spline coefficients are compared.

7. *Heat equation.* Features of the extraction operator are demonstrated on the example of time-dependent heat conduction. We demonstrate the conversion from a B-spline surface to a blending surface construction and vice versa, and compare them with an exact solution. In addition, the approximation capabilities of the blending tensor product surfaces are demonstrated on the example of the non-smooth surface approximation.

8. *Poisson's equation.* This example demonstrates the ERBS-based finite element method on a curvilinear domain. We solve a Poisson's equation with inhomogeneous boundary conditions and non-constant load, and compare several results, constructed on different mesh sizes, with an exact solution.

9. *Eigenvalue problem.* A very coarse mesh can be used to construct a geometrically exact domain. We solve an eigenvalue problem on a circular membrane using ERBS triangles as finite elements to confirm this statement. We also show how the local triangles of the first and second degree handle a complex shape of the solution.

0.2.4 Dissemination

10. Peer-reviewed publications.

    (a) T. Kravetc, B. Bang, R. Dalmo. Regression analysis using a blending type spline construction. In: *Mathematical Methods for Curves and Surfaces: 9th International Conference, MMCS 2016, Tønsberg, Norway*, June 23-28, 2016, Revised Selected Papers. Springer Publishing Company 2017. ISBN 978-3-319-67885-6. p. 145-161.

    (b) T. Kravetc, R. Dalmo. Finite element application of ERBS extraction. *In review for the Journal of Computational and Applied Mathematics*, 2019.

    (c) T. Kravetc. Finite element method application of ERBS triangles. *NIK: Norsk Informatikkonferanse 2019*, ISSN 1892-0721.

    (d) T. Martinsen, T. Kravetc. A model to estimate the economic benefit of a stationary battery energy storage at an EV charging station. (*To appear*).

11. Conference presentations.

    (a) *Mathematical Methods for Curves and Surfaces: 9th International Conference, MMCS 2016 Tønsberg, Norway, June 23-28, 2016.* Tatiana Kravetc: "Regression analysis using a blending type spline construction".

    (b) *Curves and Surfaces 2018, Arcachon, France.* Tatiana Kravetc: "Geometrical representation of a neural network using a blending type spline construction".

    (c) *NIK: Norsk Informatikkonferanse 2019, Narvik, Norway, November 25-27, 2019.* Tatiana Kravetc: "Finite element method application of ERBS triangles".

0.3 Organization of the thesis

The present thesis is directed to implementation of the framework solving the partial differential equations in variational form by using ERBS finite elements as a main tool. This research can be considered as a basis for developing teaching materials and as practical notes to develop finite element method in isogeometric context, i.e. using smooth basis functions.

The description of ERBS-based finite element method involves a combination of finite element analysis, spline concept and approximation theory, united by the isogeometric analysis. This composition may lead to some ambiguity of terms. Therefore, most relevant definitions and notations are given in the introductory part above.

The thesis is divided into three main parts. In the first part we define different ways to construct a combined expo-rational basis: on the entire domain, and on the element level; this basis is a primary tool for further research. Next, we introduce an ERBS extraction technique, which allows us to convert the B-spline control points to local geometry of blending spline/surface and vice versa.

The second part deals with ERBS-based finite elements. There are two types of elements: rectangular and triangular. The first type is based on tensor product surfaces. The second type is obtained by a concept of ERBS triangles, which blend a set of Bézier triangles. We consider features of these types of elements, construction of specific domains, and isoparametric transformation, which allows us to compute characteristics of the finite element problems.

In the third part we focus on the numerical experiments. These examples illustrate and confirm our propositions from the previous two parts. We show methods for implementing different model problems, compare different approaches and exact solutions.

Part I

ERBS extraction

Chapter 1

Combined expo-rational basis

1.1 Univariate basis

Let $\Phi = [0, 1]$ be a univariate parametric domain with a knot vector $\{t_k\}_{k=0}^{m+1}$. We define the global expo-rational basis (0.1.9) on this domain and a set of the Bernstein polynomial basis functions of the corresponding Bézier local curves (0.1.10). We denote the set of Bernstein polynomials of degree d, defined on each two knot intervals $[t_{k-1}, t_{k+1}]$, $k = 1, ..., m$, as

$$W_d = \begin{bmatrix} b_{d,0}(\omega_k(t)) & b_{d,1}(\omega_k(t)) & ... & b_{d,d}(\omega_k(t)) \end{bmatrix}, \tag{1.1.1}$$

where $b_{d,\iota}$, $\iota = 0, ..., d$ are defined by formula (0.1.11) and $\omega_k(t)$ is a local/global mapping, defined as (0.1.9).

One can combine the ERBS basis functions over the entire domain and the corresponding Bernstein basis functions on the local curve domains. Substituting (0.1.10) into (0.1.8) one can express the ERBS curve as

$$A(t) = \sum_{k=1}^{m+1} \sum_{\iota=0}^{d} q_{k,\iota}\, b_{d,\iota}(\omega_k(t))\, B_k(t).$$

To be able to apply the IGA approach to blending splines, we separate control points $q_{k,\iota}$ from basis functions. Thus, we merge Bernstein polynomials and expo-rational basis, and introduce *the combined expo-rational basis*

$$G_i(t) = b_{d,\iota}(\omega_k(t))\, B_k, \quad a = 0, ..., d, \; k = 1, ..., m,$$

where an index i is determined by $(d + 1)(k - 1) + a$. A similar approach to the basis restructuring was considered for the one dimensional case in [67], where it was compared to polynomial basis. The ERBS-generated basis is strictly local and C^∞-smooth.

We proceed by assembling a set of univariate combined expo-rational basis functions $\mathbf{G} = \mathbf{G}(t) = \{G_i(t)\}_{i=0}^{m(d+1)}$. The basis functions G_i are strictly local, i.e., they are different from zero only on the associated domains of the local curves. An example of the combined expo-rational basis evaluation is shown in Figure 1.1.1.

One can evaluate the basis $\mathbf{G}$ on an element level in matrix form. The main idea of constructing the combined basis is that the local Bernstein and underlying expo-rational basis functions are blended on the element level, instead of the level of the local curve domains. Consider a set of elements $\Phi^e = [t_e, t_{e+1}]$, $e = 0, ..., m$. Since the expo-rational basis function is symmetric, we denote an increasing part of this function as B and a

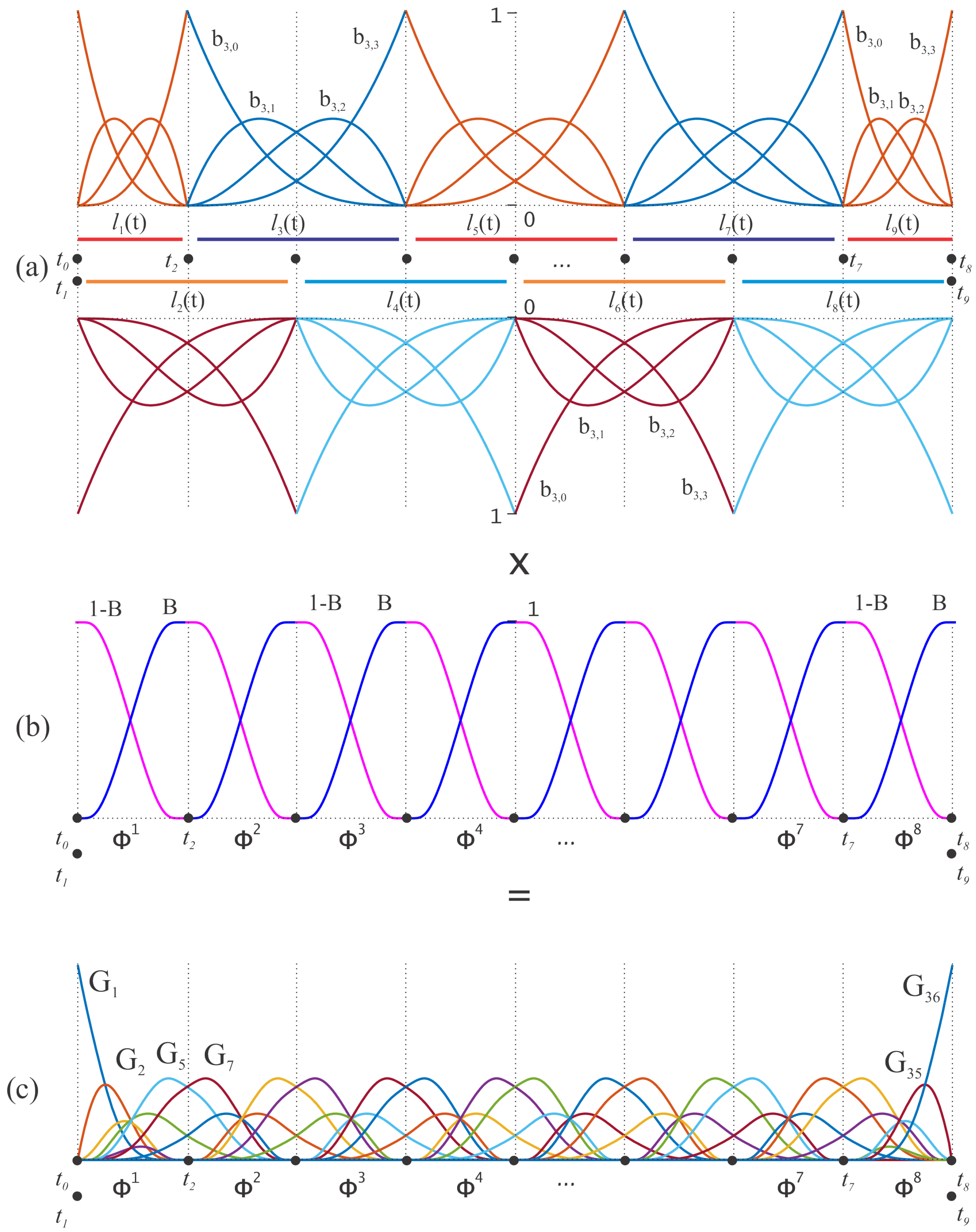


Figure 1.1.1: An example of constructing the combined expo-rational basis. (a) The local Bernstein basis of degree three. The corresponding Bézier local curves are shown as bold straight lines on appropriate local domains. (b) The expo-rational basis functions on the local elements Φ^e, $e = 1, ..., 8$. (c) A combination of the Bernstein polynomials and expo-rational basis functions on the entire domain Φ.

decreasing part as $1-B$ on each element. The set of Bernstein polynomial is also symmetric and defined on two knot intervals. Thus, let $W_d^e = \begin{bmatrix} b_{d,0}^e & b_{d,1}^e & \dots & b_{d,d}^e \end{bmatrix}$ be a set of Bernstein polynomials, recall (1.1.1), but we consider only a part which belongs to the element Φ^e. Hence, on each element Φ^e we obtain $2(d+1)$ local basis functions

$$\mathbf{G}^e = (B^e)^{\mathrm{T}} W_d^e = \begin{bmatrix} 1-B \\ B \end{bmatrix} \begin{bmatrix} b_{d,0}^e & b_{d,1}^e & \dots & b_{d,d}^e \end{bmatrix}. \tag{1.1.2}$$

We find the derivative of the expo-rational basis function and Bernstein polynomials using formulas (0.1.7) and (0.1.12), respectively. Thus, the derivative of the combined expo-rational basis can be evaluated as follows

$$\begin{aligned} D\mathbf{G}^e = (DB^e)^{\mathrm{T}} W_d^e + (B^e)^{\mathrm{T}} DW_d^e = \\ = \begin{bmatrix} -DB \\ DB \end{bmatrix} \begin{bmatrix} b_{d,0}^e & b_{d,1}^e & \dots & b_{d,d}^e \end{bmatrix} + \\ + \begin{bmatrix} 1-B \\ B \end{bmatrix} \begin{bmatrix} Db_{d,0}^e & Db_{d,1}^e & \dots & Db_{d,d}^e \end{bmatrix}. \end{aligned} \tag{1.1.3}$$

Finally, we interpret the ERBS curve as a linear combination of univariate combined expo-rational basis $\mathbf{G}$ and an ordered set of coefficients $\mathbf{Q}$ of local Bézier curves. Thus, the ERBS curve (0.1.8) can be evaluated as

$$A(t) = \mathbf{Q}^{\mathrm{T}}\mathbf{G}. \tag{1.1.4}$$

1.2 Bivariate basis

Let $\Theta = [0,1] \times [0,1]$ be a parametric domain with two parameters u and v, divided into $m = m_u m_v$ elements by the knot vectors.

To obtain the bivariate combined expo-rational basis we start with the u direction. We first define a set of the global expo-rational basis functions (0.1.9) and local Bernstein polynomial bases (0.1.11) on each two knot spans $[u_{k-1}, u_{k+1}]$, $k = 1, ..., m_u$. For each knot span we blend the bases by formula (1.1.2).

The derivative of the combined expo-rational basis in u direction can be obtained in accordance with (1.1.3).

We then define both basis and its derivative in the v direction. By using the tensor product formula (0.1.5) we finally obtain the bivariate combined expo-rational basis $\mathbf{G}(u,v) = \mathbf{G}(u)^{\mathrm{T}}\,\mathbf{G}(v)$. The number of basis functions on the entire domain is equal to $m(d+1)^2$.

The partial derivatives of the basis $\mathbf{G}$ with respect to u and v are

$$D_u\mathbf{G} = D_u\mathbf{G}(u)^{\mathrm{T}}\,\mathbf{G}(v),$$

$$D_v\mathbf{G} = \mathbf{G}(u)^{\mathrm{T}}\,D_v\mathbf{G}(v).$$

An example of the bivariate combined expo-rational basis is shown in Figure 1.2.1. One can see that the support of each basis function is contained in the subdomain $[u_{k-1}, u_{k+1}] \times [v_{l-1}, v_{l+1}]$.

A compact matrix formula, which is in general written as (0.1.2), for the tensor product surface takes the following form

$$A(u,v) = \mathbf{Q}^{\mathrm{T}}\mathbf{G}(u,v). \tag{1.2.1}$$

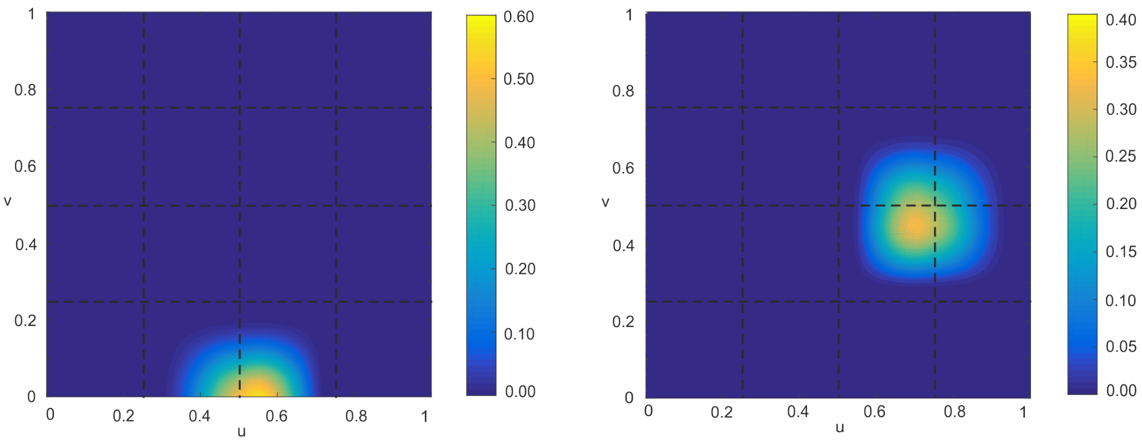


Figure 1.2.1: A contour plot of two bivariate combined expo-rational basis functions: the left one on the boundary, the right one inside the domain Θ. The local surfaces are the Bézier surfaces of degree 1.

1.3 Element level evaluation

Since the combined expo-rational basis is symmetric on an element level, one can improve its evaluation by generalization of the basis formulation.

Let us assume that we consider two knot vectors $u = \{u_k\}_{k=0}^{m_u+1}$ and $v = \{v_l\}_{l=0}^{m_v+1}$. We now focus on one interval $\Phi^k = [u_{k-1}, u_k]$ in the u direction. Introduce a local parameter $\xi \in [0,1]$ on this interval. In the following we use notations G, W_d and B for basis functions without an index, bearing in mind that they are defined only on the interval Φ^k. The notation $B(\xi)$ denotes an increasing part of the expo-rational basis function and $B(1-\xi)$ denotes a decreasing part. The local geometry is symmetrically overlapped over the interval. Figure 1.3.1 illustrates the process of building combined expo-rational basis functions on the interval Φ^k.

The Bernstein polynomial basis of degree d can be evaluated in the matrix formulation [26] as

$$W_d(\xi) = \underbrace{\begin{bmatrix} 1-\xi & \xi \end{bmatrix} \begin{bmatrix} 1-\xi & \xi & 0 \\ 0 & 1-\xi & \xi \end{bmatrix} \dots}_{d \text{ matrices}} = T_1(\xi)\, T_2(\xi) \dots T_d(\xi). \tag{1.3.1}$$

Here, the matrix T_a, $a = 1, \dots, d$ is called *a factor matrix.* The result of multiplication (1.3.1) is a $d+1$ vector W_d of Bernstein polynomials of degree d.

The first derivative of the Bernstein polynomial basis can be established by using the same factor matrices $T_1, \dots, T_d$

$$DW_d(\xi) = d\, T_1\, T_2 \dots DT_d,$$

where the derivative DT_d of the matrix T_d is a matrix of constants.

Noting a symmetry of basis functions, we conclude that the general formula for any internal combined expo-rational basis function on the interval Φ^k can be obtained as follows

$$G(\xi) = B(\xi) W_d(\omega(u_k)\, \xi), \qquad \xi \in [0,1], \tag{1.3.2}$$

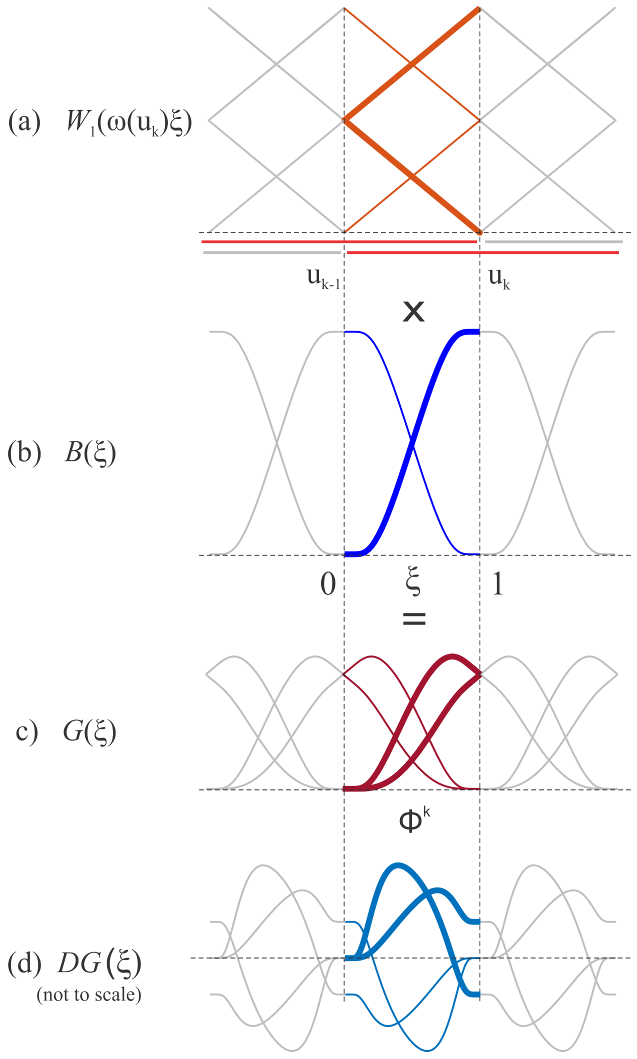


Figure 1.3.1: Evaluation of combined expo-rational basis functions on the interval Φ^k. (a) $W_1(\omega(u_k)\xi)$ is the Bernstein basis of degree 1, (b) $B(\xi)$ is the expo-rational basis, (c) $G(\xi)$ is their combination and (d) $DG(\xi)$ is the first derivative of $G(\xi)$. Bold curves identify the considered functions.

where the factor $\omega(u_k) = \dfrac{u_k - u_{k-1}}{u_{k+1} - u_{k-1}}$ is introduced for scaling the Bernstein polynomials over the considered interval.

The formula (1.3.2) gives $d+1$ combined expo-rational basis functions (indicated in Figure 1.3.1 with bold curves) defined on the interval Φ^k. Substituting $1-\xi$ into (1.3.2) instead of ξ we obtain a symmetric part of the basis on the element Φ^k. Then the complete set of basis functions on the element Φ^k is $\mathbf{G}^k = \begin{bmatrix} G(\xi) \\ G(1-\xi) \end{bmatrix}$.

The first derivative of the basis functions $G(\xi)$ should be scaled by the interval width $h = u_k - u_{k-1}$. The same applies to the blending basis $B(\xi)$. Thus, the derivative in the direction ξ can be obtained in the matrix form as follows

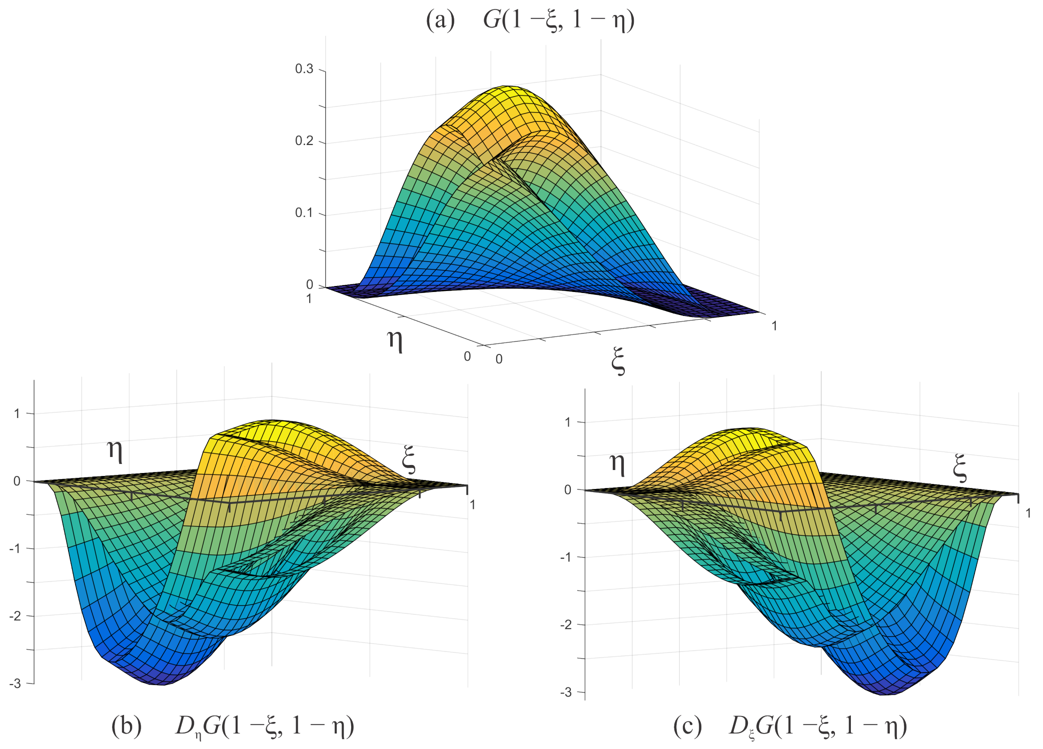


Figure 1.3.2: Local evaluation of bivariate combined expo-rational basis functions. (a) Combined expo-rational basis functions G, evaluated on a corner $(1-\xi, 1-\eta)$ of an element Θ^e. (b) Partial derivative $D_\eta G(1-\xi, 1-\eta)$ and (c) $D_\xi G(1-\xi, 1-\eta)$.

$$DG(\xi) = h^{-1}\, DB(\xi)\, W_d(\omega(u_k)\,\xi) + B(\xi)\, h^{-1}\, DW_d(\omega(u_k)\,\xi) =$$
$$= h^{-1}\begin{bmatrix} DB(\xi) & B(\xi)\end{bmatrix}\begin{bmatrix} W_d(\omega(u_k)\,\xi) \\ DW_d(\omega(u_k)\,\xi)\end{bmatrix}.$$

To define a bivariate basis, we introduce a new local parameter $\eta \in [0,1]$. Let the number of elements in the direction of the global parameter v be m_v. Finally, there are $m = m_u m_v$ elements on the entire parametric domain Θ. Then, $(d+1)^2$ bivariate basis functions, evaluated on one corner of the element Θ^e, shown on Figure 1.3.2, can be computed by the tensor product

$$G(\xi, \eta) = G(\xi)^{\mathrm{T}} G(\eta), \tag{1.3.3}$$

as well as their partial derivatives

$$D_\xi G(\xi, \eta) = DG(\xi)^{\mathrm{T}} G(\eta), \qquad D_\eta G(\xi, \eta) = G(\xi)^{\mathrm{T}} DG(\eta). \tag{1.3.4}$$

Figure 1.3.2 shows that by substituting $1-\xi$ and $1-\eta$ into formulas (1.3.3)-(1.3.4) we obtain the basis functions and their partial derivatives on the corner $(0,0)$ of the considered element. Other symmetric basis functions can be obtained by substituting, respectively, the values $(1-\xi, \eta)$, $(\eta, 1-\xi)$ and (ξ, η).

The computational technique considered above can be used for parallelization of the basis evaluation. On the uniform grid the combined expo-rational basis is point symmetric due to symmetry of both expo-rational basis function and local Bernstein polynomials which are defined on the support elements.

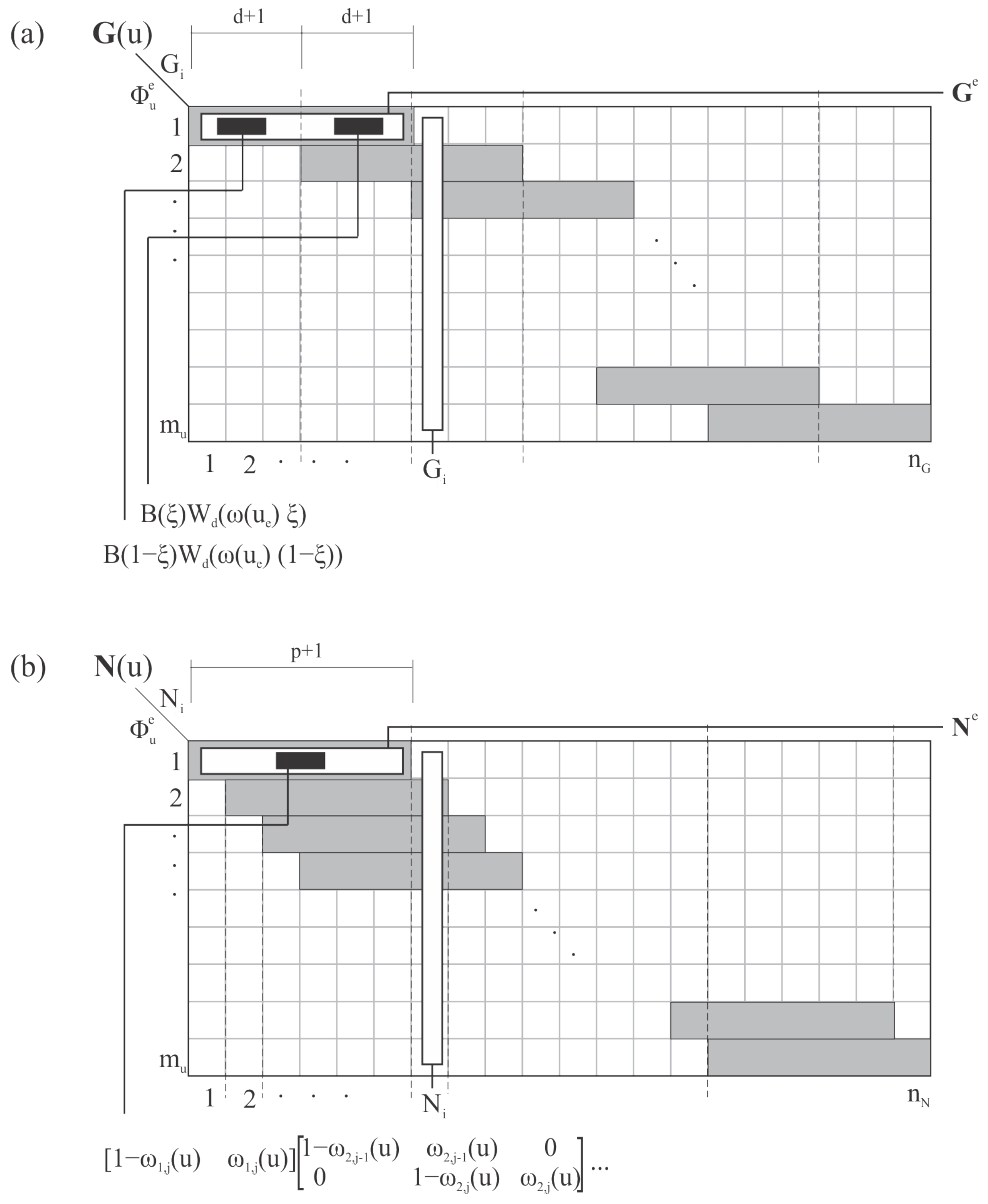


Figure 1.4.1: An example of the basis representation. (a) Combined expo-rational basis. (b) B-spline basis.

1.4 Basis representation as an array

Next, we present a method for representation of the basis in a computer program. On the one hand, each basis function is defined on the entire domain, although it has nonzero values on a small number of elements. On the other hand, each element contains parts of several basis functions. Both these representations are involved in the following array form of the basis. Figure 1.4.1 illustrates an example of the basis representation for the combined expo-rational basis $\mathbf{G}(u)$ and for the B-spline basis $\mathbf{N}(u)$. We chose for this example the basis evaluated in the u direction as a part of the bivariate basis. Evaluating the same basis in the v direction and employing the formula (0.1.5) we obtain the basis

$\mathbf{G}(u,v)$ or $\mathbf{N}(u,v)$, which can afterwards be reproduced in the same array form. It means that the presented form is common for both uni- and bivariate bases.

In accordance with Figure 1.4.1 the basis is represented as a two-dimensional array, where the index of each row corresponds to the index of the element. The rows are filled in such a way that piecewise global basis functions are obtained in each column. The global basis functions are shown in Figure 1.1.1(c) and denoted as G_i. Analogically for B-spline, which are shown in Figure 0.1.1 and denoted as N_i. These functions are piecewise and defined on the entire domain, but they are different from zero only on a small number of elements.

On each element several basis functions are defined. They are contained in the array denoted as $\mathbf{G}^e$ for the combined expo-rational basis and $\mathbf{N}^e$ for the B-spline basis. These arrays are the blocks of the final representation, and they are obtained specifically for different bases, namely by formulas (1.1.2) (or its symmetric analogue (1.3.2)) for $\mathbf{G}^e$ and (0.1.4) for $\mathbf{N}^e$. In the same way these blocks are assembled for the corresponding derivatives.

The structure of arrays Figure 1.4.1(a) and Figure 1.4.1(b) is similar, but the final form is different, and demonstrates some advantages of $\mathbf{G}$ compared to $\mathbf{N}$. Each nonzero cell in e^{th} row of the array $\mathbf{N}(u)$ is unique, while the array $\mathbf{G}(u)$ contains symmetric blocks of size $d+1$ in each row. Moreover, nonzero values in B-spline array are shifted by one cell for each row, while in the case of combined expo-rational basis the shift has length $d+1$ cells. It demonstrates the locality of the basis $\mathbf{G}$, i.e. each basis function covers only two elements, independently of the degree of the local geometry.

Chapter 2

Extraction operator

In this section we describe the decomposition of the B-spline basis into the combined expo-rational basis. For more details regarding extraction operators and conversions between basis functions and curve coefficients, we refer to [7, 34]. The extraction operator localizes the topological and global smoothness information to the element level.

The expo-rational extraction for B-splines determines the representation of the B-spline basis over each element in terms of a set of expo-rational basis functions. Since both bases are linearly independent and allow to form polynomial basis up to the same degree, it is possible to represent one basis by a linear combination of the basis functions of the another basis. The conversion is obtained in a discrete way, which yields a relative inaccuracy, since we convert polynomials into non-polynomial and strictly local functions. However, the accuracy can be controlled via adjusting the number of evaluating points.

We focus on a single element Φ^e with a parameter $\xi \in [0, 1]$. There is an important restriction imposed on the number of basis functions over the element Φ^e. The $p + 1$ B-spline basis functions with support over that element form a linearly independent and complete polynomial basis up to degree p. To represent the B-spline basis functions over that element by a linear combination of the combined expo-rational basis functions, the last functions should be linearly independent and have the same approximation power, i.e.

$$p + 1 \equiv 2(d + 1), \tag{2.0.1}$$

as shown in Figure 2.0.1. For example, the cubic local curves in the blending curve correspond to the seventh degree of the B-spline constructed on the uniform knot vector. The case of closed curves keeps such restriction in the same way.

Definition 2.0.1. For a localized B-spline basis function, $N^e_{j,p}(\xi)$, that satisfies the restriction (2.0.1), there exist coefficients, $\epsilon^e_{j,\delta}$, such that

$$N^e_{j,p}(\xi) = \sum_{\delta=1}^{2(d+1)} \epsilon^e_{j,\delta} G^e_\delta(\xi) \tag{2.0.2}$$

over the element Φ^e. In matrix form (2.0.2) can be written as

$$\mathbf{N}^e(\xi) = \mathbf{E}^e \, \mathbf{G}^e(\xi). \tag{2.0.3}$$

The matrix $\mathbf{E}^e$ maps the combined expo-rational basis to the B-spline basis over the element and it is called *the element extraction operator.*

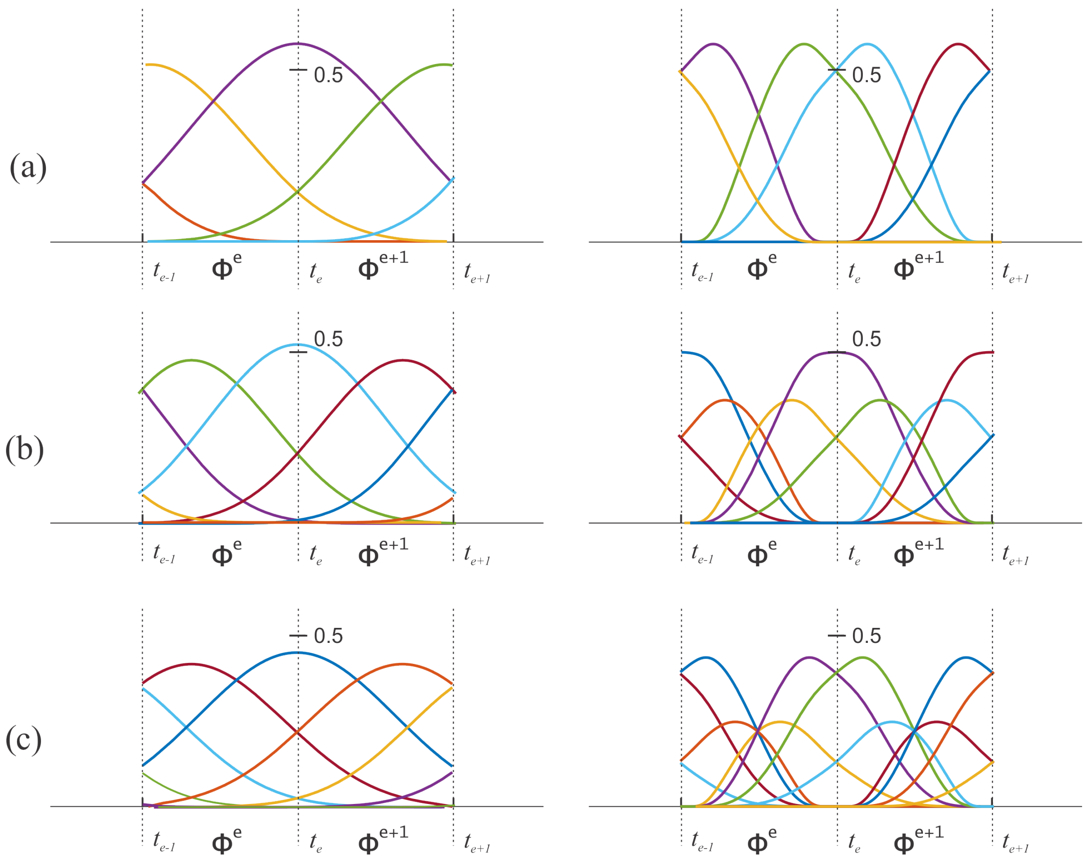


Figure 2.0.1: An example of localized bases of different degrees on the two elements Φ^e and Φ^{e+1} B-spline (left) and combined expo-rational (right). The number of basis functions over one element should be equal for both types of bases. (a) The cubic B-spline basis and the combined expo-rational basis with local Bézier curves of degree one. (b) Degrees five and two, respectively. (c) Degrees seven and three.

The reverse conversion, i.e. the mapping of the B-spline basis onto the expo-rational basis, can be obtained by the formula

$$\mathbf{G}^e(\xi) = (\mathbf{E}^e)^{-1}\mathbf{N}^e(\xi). \tag{2.0.4}$$

An illustration of basis conversion is shown in Figure 2.0.2. An objective of the extraction is to represent the ERBS curve as the B-spline curve and vice versa. The conversion is obtained locally, i.e. over the element Φ^e. It means that we get $p+1$ B-spline control points for each element after this conversion. Similarly, two local curves of corresponding degree are established after the conversion of the B-spline into ERBS control points, as shown in Figure 2.0.5.

We now insert the extraction (2.0.4) into the formula which defines the blending curve A (1.1.4) to evaluate the local curves and blend them to represent the B-spline curve S in terms of the combined expo-rational basis. For the curve segment defined over an element, we find

$$S^e(\xi) = (\mathbf{P}^e)^{\mathrm{T}}\mathbf{N}^e(\xi) = (\mathbf{P}^e)^{\mathrm{T}}\mathbf{E}^e\mathbf{G}^e(\xi) = (\mathbf{Q}^e)^{\mathrm{T}}\mathbf{G}^e(\xi) = A^e(\xi),$$

where the control points of the local curves of the blending spline that represent the B-spline curve are defined as

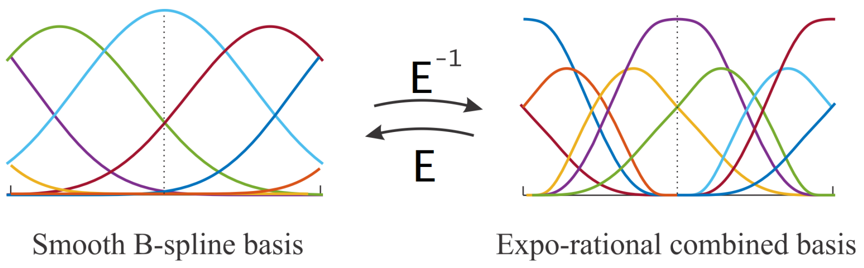


Figure 2.0.2: Illustration of the extraction operator for the conversion of B-spline to expo-rational basis functions and vice versa on an element level.

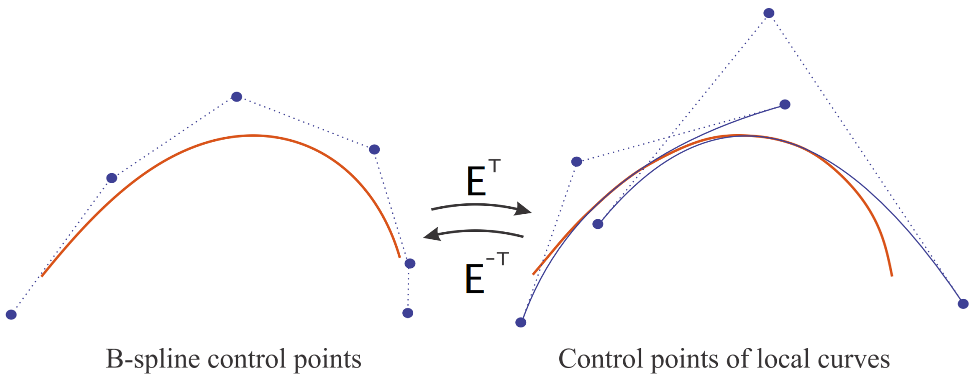


Figure 2.0.3: Illustration of the transpose of the extraction operator for the conversion of local B-spline control points to control points of local curves of an expo-rational spline curve and vice versa on an element level.

$$\mathbf{Q}^e = (\mathbf{E}^e)^{\mathrm{T}}\mathbf{P}^e. \tag{2.0.5}$$

Using the invertibility of the extraction operator, we can convert an array of the control points of the local curves into B-spline control points

$$\mathbf{P}^e = (\mathbf{E}^e)^{-\mathrm{T}}\mathbf{Q}^e. \tag{2.0.6}$$

Since the basis evaluation by tensor product keeps all the properties of the univariate basis, one can conclude that both bivariate B-spline and combined expo-rational bases are linearly independent. It allows us to represent one basis by a linear combination of the basis functions of another basis.

We organize the set of B-spline basis functions as a vector $\mathbf{N}$ with length $n_N = n_u n_v$, where n_u is a number of basis functions in the u direction, and n_v is the corresponding number in the v direction. The set of combined expo-rational basis functions $\mathbf{G}$ has a length $n_G = m_u m_v (d+1)^2$.

To provide conversion between two bases, we need to notice that the number of B-spline basis functions should be equivalent to the number of expo-rational combined basis functions *locally*, i.e., on each element. Since the bivariate basis is obtained as a tensor product of univariate bases, the restriction (2.0.1) remains the same.

We focus on a single element Θ^e of the parametric domain with two local parameters $\xi \in [0,1]$ and $\eta \in [0,1]$. Since the B-spline and combined expo-rational basis functions are linearly independent and supposing that the requirement (2.0.1) is fulfilled, we can represent the B-spline basis as a linear combination of the combined expo-rational basis functions. Thus, Definition 2.0.1 can be rewritten for the bivariate case as follows

Definition 2.0.2. Over the element Θ^e for a localized B-spline function $N^e_{\iota,p}(\xi,\eta)$, with index $\iota = 1, ..., (p+1)^2$ and degree p, there exist coefficients $\epsilon^e_{\iota,\delta}$ such that

$$N^e_{\iota,p}(\xi,\eta) = \sum_{\delta=1}^{4(d+1)^2} \epsilon^e_{\iota,\delta}\, G^e_\delta(\xi,\eta), \tag{2.0.7}$$

where G^e_δ, $\delta = 1, ..., 4(d+1)^2$, is a set of localized combined expo-rational functions.

In matrix form (2.0.7) can be written as

$$\mathbf{N}^e(\xi,\eta) = \mathbf{E}^e\, \mathbf{G}^e(\xi,\eta). \tag{2.0.8}$$

The matrix $\mathbf{E}^e$ is called *the bivariate element extraction operator.*

All one-dimensional extraction operator properties are retained for surfaces. Formulas (2.0.4), (2.0.5) and (2.0.6) are valid for the bivariate bases with two local parameters (ξ,η).

Figure 2.0.4 illustrates the bases, defined on the element Θ^e, that can be converted one to another and vice versa using the extraction operator $\mathbf{E}^e$. An element-level conversion from B-spline control net to an expo-rational tensor product surface $A^e(\xi,\eta)$, which blends Bézier local surfaces, is obtained by the following formula: for each element Θ^e, $e = 1, ..., m$,

$$A^e(\xi,\eta) = ((\mathbf{E}^e)^{\mathrm{T}}\, \mathbf{P}^e)^{\mathrm{T}}\, \mathbf{G}^e(\xi,\eta). \tag{2.0.9}$$

The inverse conversion is obtained by substituting (2.0.6) into (1.2.1)

$$S^e(\xi,\eta) = ((\mathbf{E}^e)^{-\mathrm{T}}\, \mathbf{Q}^e)^{\mathrm{T}}\, \mathbf{N}^e(\xi,\eta). \tag{2.0.10}$$

The conversions (2.0.9) and (2.0.10) are illustrated in Figure 2.0.5 for the fifth degree B-spline and the blending tensor product surface with local Bézier surfaces of degree 2. On the left hand side one can see the B-spline control net consisting of 6×6 points and the corresponding element of the tensor product surface. On the right hand side a similar surface element is represented as the blending of local geometry utilizing the expo-rational basis (note that only two of four local surfaces are shown). For such a local representation the number of control points for both kind of surfaces is equivalent, which allows for conversion between the two representations.

For completeness, consider Algorithm 2 for obtaining the extraction operator. We first assume that the bases are represented in a form such as described in Section 1.4. Thus, for each e^{th} element we have a block of parts of corresponding basis functions $\mathbf{G}^e$ and $\mathbf{N}^e$. Note that for successful mapping the size of these blocks should be equal for both bases, i.e. the requirement (2.0.1) is satisfied. We then evaluate the element level basis at equally spaced points in each parametric direction and solve the matrix equation (2.0.3) (or (2.0.8) for 2D case). Finally, we obtain a set of square matrices $\mathbf{E}^e$, $e = 1, ..., m$, which map the combined expo-rational basis onto the B-spline basis at the element level.

Algorithm 2 Extraction operator

```
1: for $e = 1, 2, ..., m$ do
2:     $\mathbf{E}(:,:,e) = \mathbf{N}^e(:,:,e) \cdot (\mathbf{G}^e(:,:,e))^{-1}$;
                                        ▷ here $() \cdot ()$ is a matrix multiplication
3: end for
```

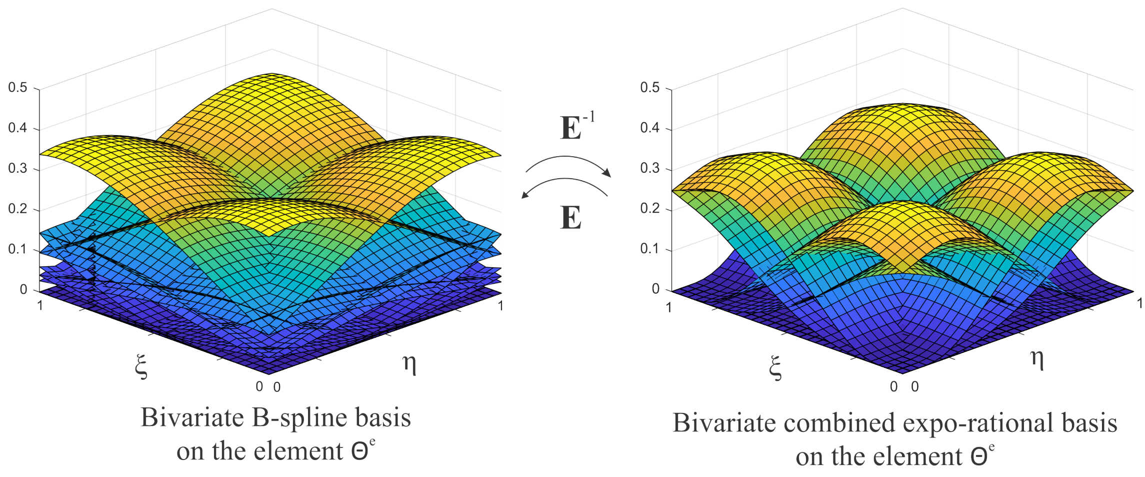


Figure 2.0.4: An extraction operator $\mathbf{E}$ converts the bivariate B-spline to expo-rational basis functions and vice versa on an element level.

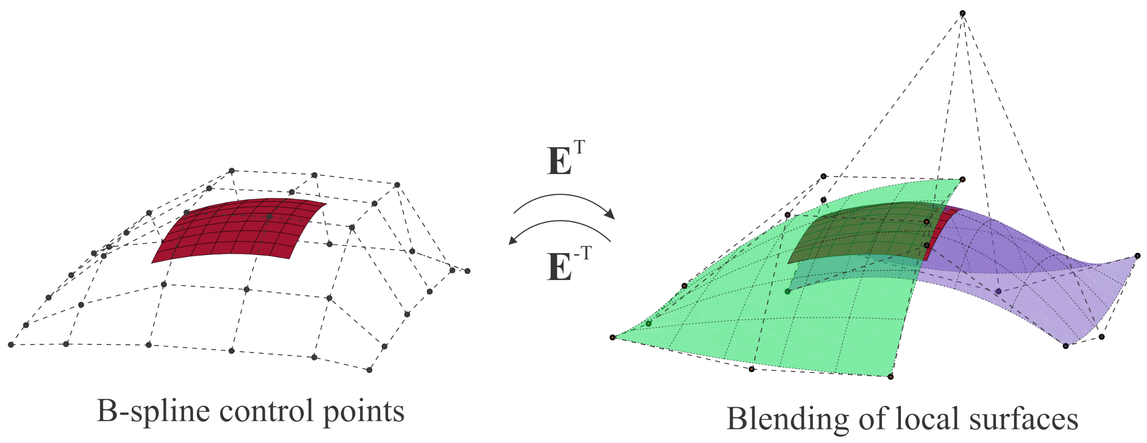


Figure 2.0.5: The transpose of the extraction operator converts the local B-spline control points to control points of local surfaces of the expo-rational tensor product surface and vice versa on an element level.

An extraction onto polynomial-based finite elements allows us to process the smooth interpretation in the same way as in a standard finite element computer program.

In contrast to ordinary polynomial finite elements, which are C^0 continuous between nodes, blending splines and surfaces preserve smoothness between nodes. The local properties are implied under the local geometry. This entails an opportunity of parallelizing the computational process.

The ERBS extraction operator provides an element data structure. Furthermore, a technique for localizing global basis information to an element is also provided.

The conversions between polynomial B-spline and combined expo-rational bases are applied to the L^2-projection technique in Chapter 6. FEM application of ERBS extraction is considered on an example of a time-dependent heat equation with non-smooth initial condition in Chapter 7.

Part II

Expo-rational finite elements

Chapter 3

Tensor product ERBS-based finite elements

3.1 Isoparametric mapping

The finite element concept is a technique for the spatial discretization of distributed parameter systems. The domain Ω of the system is partitioned into a set of subdomains. The subdomains are called *finite elements* and the set of finite elements is called *a mesh.* The reason for introducing a mesh is that it allows the construction of basis function spaces on the domain. For our specific case we consider a set of combined expo-rational functions $\mathbf{G}$ as a basis. A linear combination of these functions describes the behavior of the considered physical system.

The basis functions must be admissible and continuous [76]. Moreover, the basis functions need not be defined over the entire domain, but only over certain subdomains, and can be identically zero everywhere else. We refer to such a basis as *a local basis*, otherwise it is called *a global basis.*

In the following we consider two-dimensional elements in the global Cartesian coordinate system (x, y). Additionally, we introduce two local parameters $\xi, \eta \in [0, 1]$ on each element.

A small number of elements can represent a relatively complex form by distorting the simple elements. The curvilinear elements are called *isoparametric elements* and they retain the advantages of simple rectangular elements due to the mapping

$$x = x(\xi, \eta), \qquad y = y(\xi, \eta). \tag{3.1.1}$$

For isoparametric elements the coordinate transformation (3.1.1) is achieved by the linear combination of basis functions and control points. The ability to use the same functions both for constructing the domain and for analysis simplifies the calculations, as we show later. Figure 3.1.1 illustrates a mapping between parametric element space Θ^e and corresponding isoparametric element Ω^e in the Cartesian coordinate system.

Let m be a number of elements on the domain Ω and n_G be a number of basis functions G_i, $i = 1, 2, ..., n_G$ on the domain. The global stiffness and mass matrices ($\mathcal{A}$ and $\mathcal{M}$, respectively) have size $n_G \times n_G$, and the load vector b has size $n_G \times 1$. We now focus on one element Ω^e, where $e = 1, 2, ..., m$. Global matrices and vectors are constructed by summing the contributions of elemental matrices and vectors.

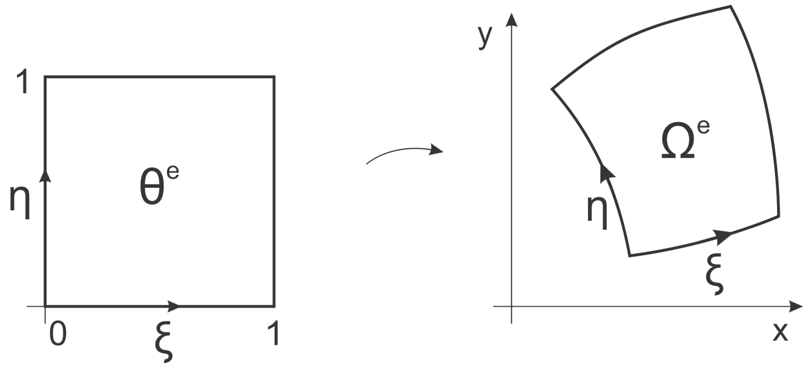


Figure 3.1.1: Two-dimensional mapping of the rectangular element.

The matrices defining element properties have to be found before we perform finite element method. The element matrix has the following form

$$\int_{\Omega^e} \mathbf{\Psi}\, d\Omega^e, \tag{3.1.2}$$

where an expression $\mathbf{\Psi}$ depends on the set of basis functions $\mathbf{G}^e$, defined on the considered element, or its derivatives with respect to global coordinates x and y. For simplicity we consider the following examples of the element stiffness matrix

$$\mathcal{A}^e = \int_{\Omega^e} \nabla(\mathbf{G}^e)^{\mathrm{T}} \nabla \mathbf{G}^e \, dxdy, \tag{3.1.3}$$

the element mass matrix

$$\mathcal{M}^e = \int_{\Omega^e} (\mathbf{G}^e)^{\mathrm{T}} \mathbf{G}^e \, dxdy, \tag{3.1.4}$$

and associated load vector

$$b^e = \int_{\Omega^e} f\, (\mathbf{G}^e)^{\mathrm{T}} \, dxdy. \tag{3.1.5}$$

In order to compute the integrals (3.1.3)-(3.1.5) we change the variables from the global Cartesian coordinate system to the local one defined on the element Θ^e. This requires the evaluation of the global basis functions, their derivatives, and the Jacobian from the physical space to the parent element at each point in the parent element.

Our question of interest is how to generate the mass and stiffness matrices for a given set of basis functions. The form of these matrices is different depending on the given basis, but the assembly algorithm is the same. An assembly subroutine for standard finite elements is detailed in [56].

Let us consider the form of the stiffness matrix. From (3.1.3) and locality of the basis functions it follows that the resulting matrix has a block diagonal form. Since the basis functions are zero outside a small number of neighbor elements, the tensor product of sets containing basis functions provides zero entries outside a band around the main diagonal. This structure of the matrix can easily be obtained if the basis is represented in a form described in Section 1.4. Due to such a representation, an appropriate location of terms $\mathcal{A}^e$, $\mathcal{M}^e$, b^e in the global matrix is established.

In the case of a stiffness matrix, constructed by analogy with formula (3.1.3) using the B-spline basis functions, it is the $(p+1)$-banded matrix, where p is the degree of the B-spline basis with n_N basis functions, defined on the entire domain. This matrix looks as follows

$$\mathcal{A}_{\text{B-spline}} =$$

$$= \begin{bmatrix} \mathcal{A}_{1,1} & \mathcal{A}_{2,1}^{\mathrm{T}} & \dots & \mathcal{A}_{p+1,1}^{\mathrm{T}} & 0 & \dots & 0 \\ \mathcal{A}_{2,1} & \mathcal{A}_{2,2} & \dots & \dots & \dots & \dots & 0 \\ \dots & \dots & \ddots & \dots & \dots & \dots & 0 \\ \mathcal{A}_{p+1,1} & \dots & \dots & \ddots & \dots & \dots & \mathcal{A}_{n_N,n_N-p}^{\mathrm{T}} \\ 0 & \dots & \dots & \dots & \ddots & \dots & \vdots \\ \vdots & \dots & \dots & \dots & \dots & \mathcal{A}_{n_N-1,n_N-1} & \mathcal{A}_{n_N,n_N-1}^{\mathrm{T}} \\ 0 & \dots & 0 & \mathcal{A}_{n_N,n_N-p} & \dots & \mathcal{A}_{n_N,n_N-1} & \mathcal{A}_{n_N,n_N} \end{bmatrix}. \quad (3.1.6)$$

Otherwise, if the matrix constructed using n_G combined expo-rational basis functions which are constructed using the Bernstein polynomials of degree d, it possesses the block tridiagonal form

$$\mathcal{A}_{\text{ERBS}} = \begin{bmatrix} \mathcal{A}_{1,1} & \mathcal{A}_{2,1}^{\mathrm{T}} & 0 & \dots & 0 \\ \mathcal{A}_{2,1} & \mathcal{A}_{2,2} & \mathcal{A}_{3,2}^{\mathrm{T}} & \dots & 0 \\ 0 & \mathcal{A}_{3,2} & \mathcal{A}_{3,3} & \dots & 0 \\ \vdots & \dots & \dots & \ddots & \mathcal{A}_{n_G/2,n_G/2-1}^{\mathrm{T}} \\ 0 & \dots & 0 & \mathcal{A}_{n_G/2,n_G/2-1} & \mathcal{A}_{n_G/2,n_G/2} \end{bmatrix}, \quad (3.1.7)$$

where the block has size $d+1 \times d+1$ and the same diagonal form.

The difference between the form of the matrices affects matrix inversion. The block diagonal matrices (3.1.7) are commonly used in the finite element method, see [76], their inverse have been studied by several authors, for instance, see [77]. General matrices, such as the $(p+1)$-banded matrix (3.1.6), are more complicated to invert. An algorithm to do so is described in [63].

The locality of the combined expo-rational basis gives us a fixed form of the resulting matrix, independently of the degree of local patches. It allows us to use the same computational routines for evaluation of each block of the matrix and thereby parallelize the process. This opportunity is clearly seen in contrast to the B-spline-based stiffness matrix, which form strictly depends on the spline degree.

3.2 Domain construction

To generate a mesh based on tensor product expo-rational elements, the mapping method is proposed. In the mesh generation process the model domain is subdivided into simple subregions, which are then mapped onto a regular grid to produce a mesh. Despite the fact that the manual subdivision into subregions can be quite difficult, especially for complex domains, the generation of elements from mappable subregions is much easier than other methods [100]. In addition, the elements generated by mapping methods usually have good shape and regular orientation.

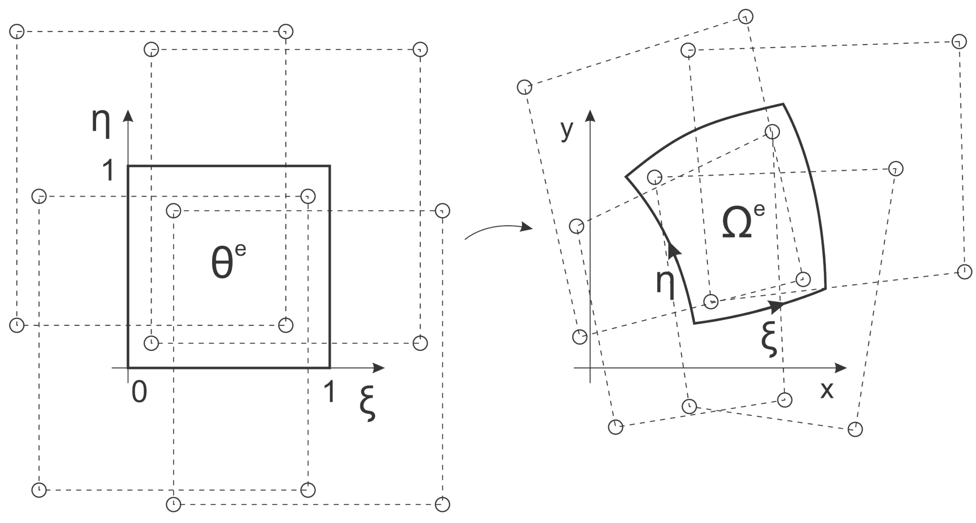


Figure 3.2.1: Two-dimensional mapping of the ERBS-based rectangular element. Control points are shown as empty circles. Dotted lines joining the control points represent local surfaces of degree 1.

Blending surfaces have a powerful tool for approximation. Local surfaces allow us to manipulate the blending surface locally and keep smoothness of the parameter lines. There is a rich variety of types of approximation methods for surfaces: Hermite interpolation, least-squares fitting, etc. Some of them in application to ERBS curves and surfaces are considered in [30]. In this research we study L^2-projection as a technique for approximating functions. L^2-projection gives a good on average approximation, as opposed to interpolation which is exact at the nodes. Moreover, in contrast to interpolation L^2-projection does not require the function we seek to approximate to be continuous or have well-defined node values [73].

Let $\Theta = [0,1] \times [0,1]$ be a parametric domain with parameters (u, v), and $\bar{\Omega} = \Omega \cup \partial\Omega$ be an initial real domain of a PDE problem defined in the Cartesian coordinate system (x, y).

A mapping $U_h : \Theta \to \bar{\Omega} \subset \mathbb{R}^2$, obtained by a linear combination of combined expo-rational basis functions and corresponding control points, is used as a base for the PDE problem. This mapping on an element level is shown in Figure 3.2.1. An overlapping of the local surfaces provides us smoothness of the domain.

Let $U \in \mathbb{R}^2$ be a surface we seek to approximate

$$U = \begin{cases} x(u, v), \\ y(u, v). \end{cases} \tag{3.2.1}$$

A suitable parameterization of the domain surface (3.2.1) should satisfy the requirement of boundary conformance. It means that the boundary of the parametric domain matches the boundary $\partial\Omega$ of the real domain.

We now formulate the L^2-projection method.

Definition 3.2.1. L^2-projection is a simple projection of an arbitrary function $U \in L^2(\Theta)$ into a finite element space $\Psi_h \subset L^2(\Theta)$, with $\Theta \subset \mathbb{R}^2$ a domain. Mathematically, it can be formulated as follows: find $U_h \in \Psi_h$ such that

$$\mathcal{L}(U_h) := \frac{1}{2}||U - U_h||^2_{L^2(\Theta)} \to \min.$$

The corresponding optimality condition reads

$$\int_\Theta (U - U_h)\psi_h \, d\Theta = 0, \qquad \forall\, \psi_h \in \Psi_h. \tag{3.2.2}$$

If (3.2.2) is satisfied for any choice of ψ as a basis function, then it is also satisfied for a linear combination of basis functions.

Since U_h belongs to Ψ_h it can be written as a linear combination

$$U_h = \sum_{j=1}^{n_G} q_j G_j = \mathbf{Q}^{\mathrm{T}}\, \mathbf{G} \tag{3.2.3}$$

with n_G unknown coefficients of the interpolant q_j to be determined. Thus, we discretized the interpolant U_h with the set of n_G basis functions $\mathbf{G} = \{G_j\}_{j=1}^{n_G}$.

Inserting (3.2.3) and a discretized test function $\psi_h = \mathbf{G}$ into (3.2.2) leads to the following system of equations

$$\int\limits_\Theta G_i\, U \; d\Theta = \int\limits_\Theta \left(\sum_{j=1}^{n_G} q_j G_j \right) G_i \; d\Theta = \sum_{j=1}^{n_G} q_j \int\limits_\Theta G_i G_j \; d\Theta, \; i = 1, ..., n_G. \tag{3.2.4}$$

We can write the system (3.2.4) in the matrix form as follows

$$\int\limits_\Theta \mathbf{G}^{\mathrm{T}} \mathbf{G}\, d\Theta\; \mathbf{Q} = \int\limits_\Theta \mathbf{G}^{\mathrm{T}} U \, d\Theta, \tag{3.2.5}$$

where $\mathbf{G} = \mathbf{G}(u, v)$ is the row vector of basis functions and $\mathbf{Q} \subset \mathbb{R}^2$ is the column vector of interpolant coefficients.

A linear combination of coefficients $\mathbf{Q}$, obtained by (3.2.5), and basis functions $\mathbf{G}$ gives an approximation U_h of the domain Ω. A mesh, which is a tensor product of two knot vectors in both u and v directions, corresponds to the mesh on the real domain Ω. The boundary $\partial\Omega$ can be found by setting limit values for the parameters u or v.

Blending spline type constructions are suitable for most mesh refinement techniques based on knot insertion [19]. New local patches for inserted elements are expressed in terms of the existing ones. In the isogeometric analysis computable error estimates are used to control iterative improvement of the solution accuracy. For example, in [97] the curvature-based adaptive mesh refinement is proposed for B-spline tensor product surfaces. An adaptive local refinement based on ERBS curves is investigated on the example of regression analysis in Chapter 5.

3.3 Coordinate transformation

3.3.1 Isoparametric elements

There are $4(d+1)^2$ control points belonging to one element Ω^e. These control points $\mathbf{Q}^e \subset \mathbb{R}^2$ represent four overlapped local surfaces. One local surface is common for neighbor elements. Such construction takes into account both interpolatory property and continuity of the surface over the knots.

The initial domain Ω is constructed as the tensor product surface. A solution of the finite element problem, that solves the partial differential equation over the discretized domain, is represented also as a tensor product surface. Assume that the finite element approximation of the solution ϑ_h^e over a given element has the form

$$\vartheta_h^e(\xi,\eta) = \sum_{i=1}^{4(d+1)^2} \zeta_i \, G_i(\xi,\eta) = \zeta^{\mathrm{T}} \, \mathbf{G}(\xi,\eta), \tag{3.3.1}$$

where $\mathbf{G}(\xi,\eta)$ is a set of combined expo-rational basis functions, constructed by using formula (1.3.3) for each corner of the parametric element Θ^e, and ζ is a vector of coefficients of the approximated solution.

On the real domain Ω, the mapping of (ξ,η) onto (x,y) is given by

$$x = \sum_{i=1}^{4(d+1)^2} G_i(\xi,\eta)\, x_i = \mathbf{G}(\xi,\eta)\,\mathbf{x}, \qquad y = \sum_{i=1}^{4(d+1)^2} G_i(\xi,\eta)\, y_i = \mathbf{G}(\xi,\eta)\,\mathbf{y}, \tag{3.3.2}$$

in which $\mathbf{x}$ and $\mathbf{y}$ are vectors with entries equal to the x- and y-components, respectively, of the coefficients $\mathbf{Q}^e$.

We are concerned only with transformations, where the dimensions of basis functions both for isoparametric mapping in formula (3.3.2) and approximation of the solution in (3.3.1) are equal.

The generalized approach to derive the stiffness and mass matrices for isoparametric elements is detailed in [100, 56] and [76]. We apply such an approach to expo-rational finite elements without loss of generalization.

From (3.1.4) it follows that the stiffness matrix involves the partial derivatives $\dfrac{dG}{dx}$ and $\dfrac{dG}{dy}$, as well as the differential element of area $dxdy$.

We first introduce *the Jacobi matrix*

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

Deriving J by using the basis functions G, which define coordinate transformation, we get

$$J = \begin{bmatrix} \sum\limits_i \dfrac{\partial G_i}{\partial \xi} x_i & \sum\limits_i \dfrac{\partial G_i}{\partial \xi} y_i \\ \sum\limits_i \dfrac{\partial G_i}{\partial \eta} x_i & \sum\limits_i \dfrac{\partial G_i}{\partial \eta} y_i \end{bmatrix} = \begin{bmatrix} \dfrac{\partial G_1}{\partial \xi} & \dfrac{\partial G_2}{\partial \xi} & \cdots \\ \dfrac{\partial G_1}{\partial \eta} & \dfrac{\partial G_2}{\partial \eta} & \cdots \end{bmatrix} \begin{bmatrix} x_1 & y_1 \\ x_2 & y_2 \\ \vdots & \vdots \end{bmatrix} = \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix} \mathbf{Q}^e, \tag{3.3.3}$$

where the partial derivatives of the basis functions can be found for each corner of the element by using expressions (1.3.4).

For the general transformation described by (3.3.2) we can write the isoparametric mapping of the differential element of area in the following matrix form

$$\begin{bmatrix} dx \\ dy \end{bmatrix} = J^{\mathrm{T}} \begin{bmatrix} d\xi \\ d\eta \end{bmatrix}. \tag{3.3.4}$$

Then, the differential element of area can be shown to transform according to [92]

$$dxdy = |J|\, d\xi d\eta, \tag{3.3.5}$$

where $|J|$ is *the Jacobian.*

By inverting the general transformation (3.3.2), we can write

$$\begin{bmatrix} d\xi \\ d\eta \end{bmatrix} = \bar{J}^{\mathrm{T}} \begin{bmatrix} dx \\ dy \end{bmatrix}, \tag{3.3.6}$$

where

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

Comparing (3.3.4) and (3.3.6), according to [76], we conclude that

$$\begin{bmatrix} \dfrac{\partial \xi}{\partial x} & \dfrac{\partial \eta}{\partial x} \\ \dfrac{\partial \xi}{\partial y} & \dfrac{\partial \eta}{\partial y} \end{bmatrix} = J^{-1} = \frac{1}{|J|} \begin{bmatrix} \dfrac{\partial y}{\partial \eta} & -\dfrac{\partial y}{\partial \xi} \\ -\dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \xi} \end{bmatrix}. \tag{3.3.7}$$

Next, we find the gradient of the basis functions. Determine their partial derivatives using the chain rule

$$\frac{\partial G}{\partial x} = \frac{\partial G}{\partial \xi}\frac{\partial \xi}{\partial x} + \frac{\partial G}{\partial \eta}\frac{\partial \eta}{\partial x}, \qquad \frac{\partial G}{\partial y} = \frac{\partial G}{\partial \xi}\frac{\partial \xi}{\partial y} + \frac{\partial G}{\partial \eta}\frac{\partial \eta}{\partial y}. \tag{3.3.8}$$

Expressions (3.3.8) in conjunction with (3.3.7) can be rewritten in the matrix form

$$\begin{bmatrix} \dfrac{\partial G}{\partial x} \\ \dfrac{\partial G}{\partial y} \end{bmatrix} = \frac{1}{|J|} \begin{bmatrix} \dfrac{\partial y}{\partial \eta} & -\dfrac{\partial y}{\partial \xi} \\ -\dfrac{\partial x}{\partial \eta} & \dfrac{\partial x}{\partial \xi} \end{bmatrix} \begin{bmatrix} \dfrac{\partial G}{\partial \xi} \\ \dfrac{\partial G}{\partial \eta} \end{bmatrix}.$$

We can now define the element matrix by computing an integral in the form (3.1.2). Since parameters (ξ, η) are normalized, one can write an integral (3.1.2) as

$$\int_0^1 \int_0^1 \mathbf{\Psi}(\xi, \eta)\, |J|\, d\xi d\eta. \tag{3.3.9}$$

An expanded formula for the isoparametric element stiffness matrix is obtained as follows

$$\mathcal{A}^e = \int\limits_{\Omega^e} \nabla \mathbf{G}^{\mathrm{T}} \nabla \mathbf{G}\, dxdy = \int\limits_{\Omega^e} \begin{bmatrix} \dfrac{\partial \mathbf{G}}{\partial x} & \dfrac{\partial \mathbf{G}}{\partial y} \end{bmatrix} \begin{bmatrix} \dfrac{\partial \mathbf{G}}{\partial x} \\ \dfrac{\partial \mathbf{G}}{\partial y} \end{bmatrix} dxdy =$$

$$= \int\limits_0^1 \int\limits_0^1 \left(J^{-1} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix} \right)^{\mathrm{T}} \left(J^{-1} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix} \right) |J|\, d\xi d\eta =$$

$$= \int\limits_0^1 \int\limits_0^1 \frac{1}{|J|} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} \left(\dfrac{\partial x}{\partial \eta}\right)^2 + \left(\dfrac{\partial y}{\partial \eta}\right)^2 & -\left(\dfrac{\partial x}{\partial \xi}\dfrac{\partial x}{\partial \eta} + \dfrac{\partial y}{\partial \xi}\dfrac{\partial y}{\partial \eta}\right) \\ -\left(\dfrac{\partial x}{\partial \xi}\dfrac{\partial x}{\partial \eta} + \dfrac{\partial y}{\partial \xi}\dfrac{\partial y}{\partial \eta}\right) & \left(\dfrac{\partial x}{\partial \xi}\right)^2 + \left(\dfrac{\partial y}{\partial \xi}\right)^2 \end{bmatrix} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix} d\xi d\eta =$$

$$= \int\limits_0^1 \int\limits_0^1 \frac{1}{|J|} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix}^{\mathrm{T}} \mathcal{B} \begin{bmatrix} D_\xi \mathbf{G}(\xi,\eta) \\ D_\eta \mathbf{G}(\xi,\eta) \end{bmatrix} d\xi d\eta. \quad (3.3.10)$$

By substituting (3.3.3) into formula (3.3.10) we obtain the value of the integral at any point on the element, for any element just by changing the coefficients $\mathbf{x}$ and $\mathbf{y}$. Thus, the computations become

$$|J| = \mathbf{x}^{\mathrm{T}} \left[D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G} - D_\eta \mathbf{G}^{\mathrm{T}} D_\xi \mathbf{G} \right] \mathbf{y}$$

and

$$\mathcal{B} = \begin{bmatrix} \mathbf{x}^{\mathrm{T}} D_\eta \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\eta \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y} & -\left(\mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y} \right) \\ -\left(\mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y} \right) & \mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\xi \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\xi \mathbf{G}\, \mathbf{y} \end{bmatrix}.$$

Moreover, the isoparametric element mass matrix can be found by inserting (3.3.3) and (3.3.5) into (3.1.4)

$$\mathcal{M}^e = \int\limits_0^1 \int\limits_0^1 \mathbf{G}^{\mathrm{T}} \mathbf{G}\, |J|\, d\xi d\eta.$$

The coordinate transformation of the element load vector b^e requires the external force f expressed in terms of local parameters ξ and η. It can be obtained by substituting the mapping $U_h : \Theta \to \Omega$ into the given function $f(x,y)$ as $f(U_h(\xi,\eta))$. Then, the formula (3.1.5) becomes

$$b^e = \int\limits_0^1 \int\limits_0^1 f(U_h(\xi,\eta))\, \mathbf{G}^{\mathrm{T}}\, |J|\, d\xi d\eta.$$

Thereafter, the integral of the form (3.3.9) can be approximately computed by a numerical integration formula. For example, in one dimension the Gaussian quadrature is optimal. Gaussian rules for integrals in several dimensions are constructed by employing one-dimensional Gaussian rules on each coordinate separately. The theory of numerical integration is detailed in [93].

Algorithm 3 describes an assembly process of the components of discretized variational formulation for two-dimensional PDE problems. The basis is represented in a form described in Section 1.4, i.e. each e^{th} row in the array represents a set of basis functions defined on this element. We introduce the local to global mapping ("`l2b`"), which is an array of indexes of nonzero values in $\mathbf{G}^e$ array. Thus, for tensor product surfaces we always get a matrix of the form (3.1.7).

Algorithm 3 Assembly of the Stiffness and Mass Matrix, and the Force Vector

```
 1: Let $m$ be the number of elements, $\mathbf{G}$ be the combined expo-rational basis consisting
    of $n$ functions, $D_u\mathbf{G}$ and $D_v\mathbf{G}$ be its derivatives. The mesh is described by the set of
    points $\mathbf{Q}$.
 2: Allocate memory for the $n_G \times n_G$ matrices $\mathcal{M}$, $\mathcal{A}$, and $n_G \times 1$ vector $b$, and initialize
    all their entires to zero.
 3: for $e = 1, 2, ..., m$ do
 4:     Compute the Jacobi matrix.
        $J = \begin{bmatrix} D_u\mathbf{G}^e_{\texttt{l2b}} \\ D_v\mathbf{G}^e_{\texttt{l2b}} \end{bmatrix} \mathbf{Q}_{\texttt{l2b}}.$
 5:     Define $\xi$, $\eta$ as local parameters on the $e^{\text{th}}$ element.
 6:     Compute the Jacobian and inverse Jacobi matrix.
        $|J| = \det(J(\xi,\eta)), \quad J^{-1} = \frac{1}{|J|} \begin{bmatrix} J[2,2] & -J[1,2] \\ -J[2,1] & J[1,1] \end{bmatrix}.$
 7:     Compute the gradients $\nabla\mathbf{G}$ on the $e^{\text{th}}$ element.
        $\nabla\mathbf{G} = \begin{bmatrix} D_x\mathbf{G} \\ D_y\mathbf{G} \end{bmatrix} = J^{-1} \begin{bmatrix} D_u\mathbf{G}^e_{\texttt{l2b}}(\xi,\eta) \\ D_v\mathbf{G}^e_{\texttt{l2b}}(\xi,\eta) \end{bmatrix}.$
 8:     Compute the local element matrices given by
        $\mathcal{M}^e = \int_0^1\int_0^1 (\mathbf{G}^e_{\texttt{l2b}}(\xi,\eta))^{\mathrm{T}}\, \mathbf{G}^e_{\texttt{l2b}}(\xi,\eta)\, |J|\, d\xi d\eta,$
        $\mathcal{A}^e = \int_0^1\int_0^1 \big((D_x\mathbf{G})^{\mathrm{T}} D_x\mathbf{G} + (D_y\mathbf{G})^{\mathrm{T}} D_y\mathbf{G}\big)\, |J|\, d\xi d\eta,$
        $b^e = \int_0^1\int_0^1 f\,(U_h(\xi,\eta))\, (\mathbf{G}^e_{\texttt{l2b}}(\xi,\eta))^{\mathrm{T}}\, |J|\, d\xi d\eta.$
                                                        ▷ $U_h$ is the domain approximation.
 9:     Set up the local to global mapping.
        $\mathcal{M}_{\texttt{l2b,l2b}} = \mathcal{M}_{\texttt{l2b,l2b}} + \mathcal{M}^e, \ \mathcal{A}_{\texttt{l2b,l2b}} = \mathcal{A}_{\texttt{l2b,l2b}} + \mathcal{A}^e,$
        $b_{\texttt{l2b}} = b_{\texttt{l2b}} + b^e.$
10: end for
```

3.3.2 Boundary conditions

To be more general in the implementation, we involve the Robin boundary conditions, which combine both Dirichlet and Neumann boundary conditions, as mentioned in Section 0.1.3.

For the boundary value problem, solving with respect to ϑ, the Robin boundary conditions are written as

$$-a\frac{\partial\vartheta}{\partial\overline{n}} = \kappa(\vartheta - g_D) - g_N, \quad \text{on } \partial\Omega, \tag{3.3.11}$$

where κ is a constant, g_D is a Dirichlet boundary condition and g_N is a Neumann boundary condition.

In the variational formulation, the boundary conditions (3.3.11) are assembled into the boundary matrix $\mathcal{R}$ and the boundary vector r with the entries (0.1.24) and (0.1.25).

Let us assume that the outer boundary consists of s edges $\partial\Omega = \{\gamma_1, \gamma_2, ..., \gamma_s\}$. Then the integrals (0.1.24), (0.1.25) become

$$\mathcal{R} = \sum_{\iota=1}^{s} \int\limits_{\gamma_\iota} \kappa\, \mathbf{G}^{\mathrm{T}}\mathbf{G}\, d\gamma_\iota \tag{3.3.12}$$

and

$$r = \sum_{\iota=1}^{s} \int\limits_{\gamma_\iota} (\kappa g_D + g_N)\mathbf{G}^{\mathrm{T}}\, d\gamma_\iota. \tag{3.3.13}$$

Boundary matrix (3.3.12) and vector (3.3.13) imply computations of curvilinear integrals along each boundary.

Since we required in Section 3.2 that the boundary of the parametric domain Θ conforms to the boundary of the real domain $\partial\Omega$, we represent the edges as a mapping of one of parameters (u, v), while the other one is fixed, onto (x, y).

$$\begin{aligned}
\gamma_1 &= \mathbf{G}(u, v = 0)^{\mathrm{T}}\, \mathbf{Q},\\
\gamma_2 &= \mathbf{G}(u = 1, v)^{\mathrm{T}}\, \mathbf{Q},\\
\gamma_3 &= \mathbf{G}(u, v = 1)^{\mathrm{T}}\, \mathbf{Q},\\
\gamma_4 &= \mathbf{G}(u = 0, v)^{\mathrm{T}}\, \mathbf{Q}.
\end{aligned}$$

Corresponding derivatives at the edges can be found in a similar way.

Then, the boundary matrix $\mathcal{R}$ (3.3.12) is obtained as

$$\mathcal{R} = \sum_{\iota=1}^{4} \int\limits_{0}^{1} \kappa\mathbf{G}|_{\gamma_\iota}^{\mathrm{T}}\mathbf{G}|_{\gamma_\iota}\, ||D\gamma_\iota||\, d\sigma, \tag{3.3.14}$$

where the basis functions are defined on the corresponding boundary (by fixing the corresponding parameter), and σ is a formal parameter, which is u or v depending on the boundary index.

The boundary vector r involves the inhomogeneous condition (3.3.11)

$$r = \sum_{\iota=1}^{4} \int_{0}^{1} (\kappa\, g_D + g_N)\, \mathbf{G}|_{\gamma_\iota}^{\mathrm{T}}\, ||D\gamma_\iota||\, d\sigma. \tag{3.3.15}$$

Algorithm 4 describes a method of obtaining the matrix $\mathcal{R}$ and vector r for solving PDE on the domain approximated by the tensor product surface such that the real domain boundaries conform to the boundaries of the parametric domain. A main benefit of this approach is that it is general for homogeneous, inhomogeneous and mixed boundary conditions. One can establish any combination of Dirichlet and Neumann boundary conditions at the stage of determining the variational formulation.

Algorithm 4 Assembling the Boundary Conditions

```
1: Let $\mathbf{G}(u,v)$ be the combined expo-rational basis, $D_u\mathbf{G}$ and $D_v\mathbf{G}$ be its derivatives.
   The mesh is described by the set of points $\mathbf{Q}$. The outer boundary $\partial\Omega$ consists of four
   edges $\gamma_1, ..., \gamma_4$.
2: Given the functions $g_{D_1}, ..., g_{D_4}$ and $g_{N_1}, ..., g_{N_4}$ and constants $\kappa_1, ..., \kappa_4$ for each bound-
   ary.
3: Allocate memory for the $n_G \times n_G$ matrix $\mathcal{R}$ and $n_G \times 1$ vector $r$, and initialize all their
   entires to zero.
4: Define $\sigma = [0,1]$ as the parameter on the boundary.
5: Find the derivatives of each boundary
       $D\gamma_1 = \begin{bmatrix} D\gamma_1^x & D\gamma_1^y \end{bmatrix} = D_u\mathbf{G}(u=\sigma, v=0)^{\mathrm{T}}\,\mathbf{Q},$
       $D\gamma_2 = \begin{bmatrix} D\gamma_2^x & D\gamma_2^y \end{bmatrix} = D_v\mathbf{G}(u=1, v=\sigma)^{\mathrm{T}}\,\mathbf{Q},$
       $D\gamma_3 = \begin{bmatrix} D\gamma_3^x & D\gamma_3^y \end{bmatrix} = D_u\mathbf{G}(u=\sigma, v=1)^{\mathrm{T}}\,\mathbf{Q},$
       $D\gamma_4 = \begin{bmatrix} D\gamma_4^x & D\gamma_4^y \end{bmatrix} = D_v\mathbf{G}(u=0, v=\sigma)^{\mathrm{T}}\,\mathbf{Q}.$
6: Find the basis on each boundary.
       $\mathbf{G}|_{\gamma_1} = \mathbf{G}(u=\sigma, v=0), \quad \mathbf{G}|_{\gamma_2} = \mathbf{G}(u=1, v=\sigma),$
       $\mathbf{G}|_{\gamma_3} = \mathbf{G}(u=\sigma, v=1), \quad \mathbf{G}|_{\gamma_4} = \mathbf{G}(u=0, v=\sigma).$
7: Compute the boundary matrix $\mathcal{R}$.
       $\mathcal{R} = \mathcal{R} + \sum_{\iota=1}^{4} \int_{0}^{1} \kappa_\iota\, \mathbf{G}|_{\gamma_\iota}^{\mathrm{T}} \mathbf{G}|_{\gamma_\iota} \sqrt{(D\gamma_\iota^x(\sigma))^2 + (D\gamma_\iota^y(\sigma))^2}\, d\sigma.$
8: Compute the boundary vector $r$.
       $r = r + \sum_{\iota=1}^{4} \int_{0}^{1} (\kappa_\iota\, g_{D_\iota}(\sigma) + g_{N_\iota}(\sigma))\, \mathbf{G}|_{\gamma_\iota}^{\mathrm{T}} \sqrt{(D\gamma_\iota^x(\sigma))^2 + (D\gamma_\iota^y(\sigma))^2}\, d\sigma.$
```

A model example implemented in Chapter 8 illustrates the approach to the domain discretization and subsequent coordinate transformation. The considered problem is a Poisson's equation with inhomogeneous boundary condition on a curvilinear domain.

Chapter 4

ERBS finite elements on triangulations

4.1 ERBS triangles

Triangulation is a common approach to the domain discretization. Triangulation is more general than the tensor product surfaces, due to fewer geometrical constraints. When approximating solutions by the finite element method, we should choose the finite element mesh in such a way that it must not only get an accurate approximate solution but also the edges of the outer elements must approximate well the boundary. Triangular elements are particularly suited to the task of filling domains with smooth boundaries, thus minimizing the difference between the initial domain and the finite element domain.

Triangular B-splines were firstly proposed in [18]. Despite their advantages and universality, triangular B-splines are quite difficult to control in IGA context. An accuracy improvement for this type of basis was presented in [60], where the improved triangular B-splines, named reproducing kernel triangular B-splines, deal with instability. Due to the construction of a kernel correction term the improved triangular B-splines satisfy the partition of unity condition.

In the presented thesis we explore an alternative type of elements, called ERBS triangles. These elements are based on expo-rational blending splines, which have been introduced in [29]. One of the distinctions between B-splines and expo-rational blending surfaces is that the second ones can be relatively easily evaluated on a triangular mesh. The reason for this is that the expo-rational basis is strictly local under the local geometry. The mesh constructed by ERBS triangles gives a very good approximation of the PDE solution with very coarse domain discretization, but geometrically exact on the boundary.

The following related works demonstrate utilization of blending spline surfaces on triangulations. An implementation of the spline blending surface approximation over a triangulated irregular network is shown in [20]. The first instances of expo-rational finite elements on triangulations was presented in [33] and their application can be found in [98].

To use the ERBS triangles in a finite element context, we need to introduce a basis for ERBS triangle that combines both Bernstein basis and expo-rational basis in barycentric coordinates. Locally, i.e. on the one ERBS triangle K, the number of combined basis functions is equal to $n_G^K = 3\binom{d+2}{2}$.

The Bernstein polynomial basis (see Definition 0.1.6), defined on the triangle K, can be written as a matrix W_d^K with ordered elements

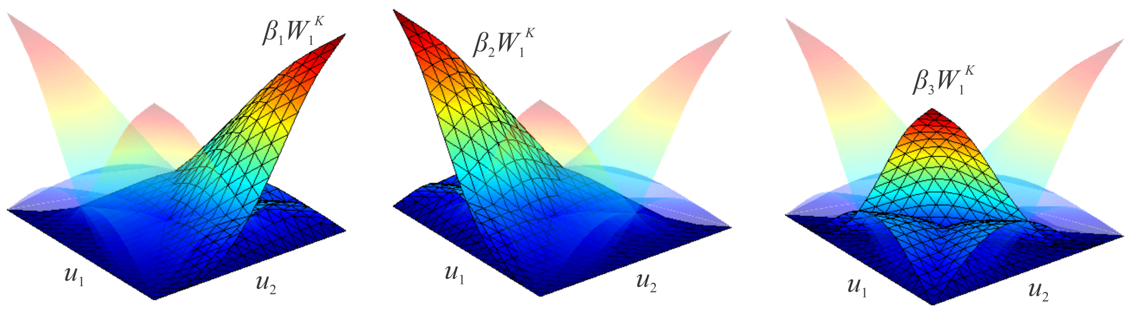


Figure 4.1.1: Combined expo-rational basis functions in barycentric coordinates, constructed with the first degree local triangles.

$$W_d^K = \begin{bmatrix} b^d_{0,0,d} & b^d_{0,1,d-1} & \cdots & b^d_{d,0,0} \end{bmatrix}.$$

The expo-rational basis $\beta_1, \beta_2, \beta_3$ is defined on the triangle K as shown in Definition 0.1.10. Then the combined basis $G^K = G^K(u_1, u_2, u_3)$, defined on the same triangle, is formulated as

$$G^K = \begin{bmatrix} \beta_1 W_d^K & \beta_2 W_d^K & \beta_3 W_d^K \end{bmatrix}. \tag{4.1.1}$$

Thus, the formula (0.1.15) can be rewritten in a compact matrix form as

$$A(u_1, u_2, u_3) = (Q^K)^{\mathrm{T}}\, G^K,$$

where Q^K is a set of corresponding coefficients of three local triangles. An example of these local triangles is demonstrated in Figure 0.1.5.

Figure 4.1.1 demonstrates a set of combined expo-rational basis functions G^K, constructed by using three first degree local triangles.

We now evaluate a set of partial derivatives of the combined expo-rational basis.

Definition 4.1.1. Let $D_{u_\iota} W_d^K$ be a set of directional derivatives in the direction u_ι of Bernstein basis functions of degree d, evaluated by Definition 0.1.7. Then, from the matrix formulation (4.1.1) of the combined expo-rational basis on the triangle, it follows that the partial derivatives of this basis can be formulated as

$$D_{u_\iota} G^K = \begin{bmatrix} D_{u_\iota}\beta_1\, W_d^K + \beta_1 D_{u_\iota}\, W_d^K \\ D_{u_\iota}\beta_2\, W_d^K + \beta_2\, D_{u_\iota} W_d^K \\ D_{u_\iota}\beta_3\, W_d^K + \beta_3\, D_{u_\iota} W_d^K \end{bmatrix} = \begin{bmatrix} D_{u_\iota}\beta_1 & \beta_1 \\ D_{u_\iota}\beta_2 & \beta_2 \\ D_{u_\iota}\beta_3 & \beta_3 \end{bmatrix} \begin{bmatrix} W_d^K \\ D_{u_\iota}\, W_d^K \end{bmatrix},$$

for each $\iota = 1, 2, 3$.

4.2 Domain construction

The union of triangular finite elements represents the finite element mesh. The expo-rational triangles allow construction of a very coarse mesh, while being exact to the boundary. Since local triangles provide an additional level of abstraction, the mesh generation can be based on the position and orientation of these triangles. An optimal position of

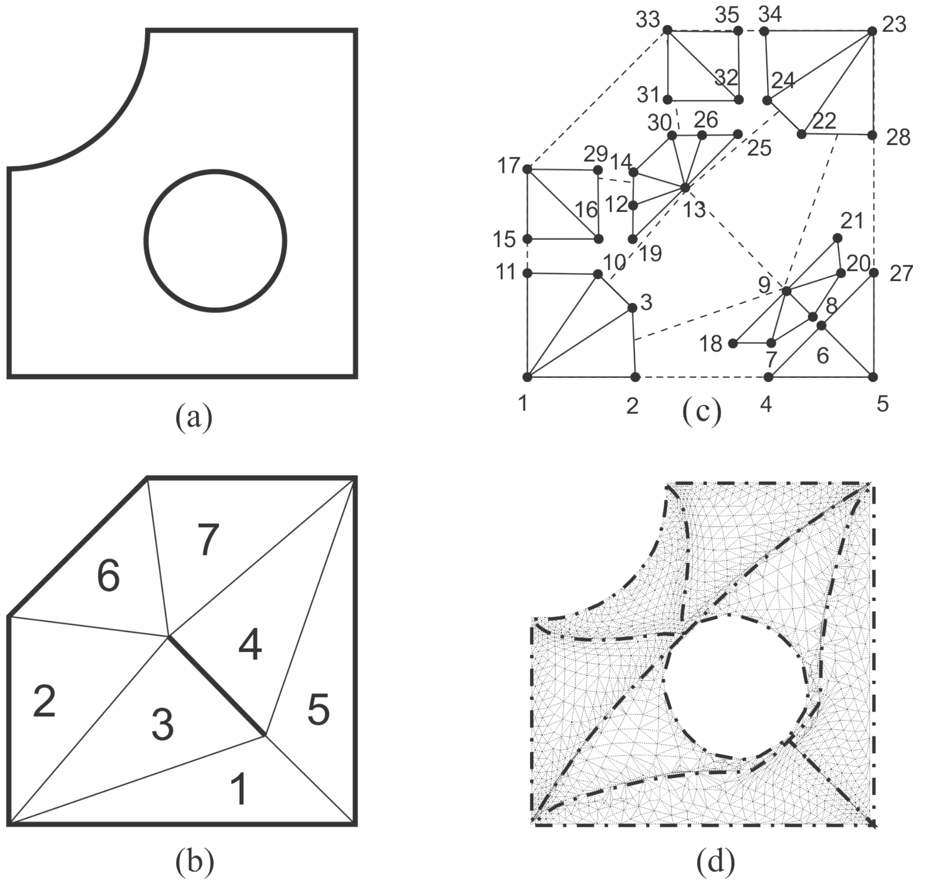


Figure 4.2.1: An example of the triangular mesh construction. (a) Original domain. (b) Coarse discretization into elements. (c) Local triangles and indexation of control points. (d) Final approximation of the domain.

the local triangles seems to be a non-trivial task and goes beyond our current research purpose. However, the outer boundary of the real domain can be approximated by using L^2-projection, formulated in Section 3.2, or any other approximation method. Internal points can be chosen with the aim to satisfy the intrinsic geometry properties.

A control net based on ERBS triangles has a layered structure. Local triangles can be connected in a flexible way: to keep the mesh continuity or to provide holes in the domain. The number of coefficients and basis functions varies depending upon the point configuration, even if the number of local triangles and elements remains the same.

Figure 4.2.1 illustrates the mesh construction using ERBS triangles. A target domain, shown in Figure 4.2.1(a), has a circular hole. We first build a coarse triangulate mesh, shown in Figure 4.2.1(b), where third and fourth elements do not have a common edge. Then we allocate local triangles, as shown in Figure 4.2.1(c), and assign point indexes. When local triangles are blended, we obtain an approximation of the original domain, see Figure 4.2.1(d). Note that to get final approximation, local triangles are overlapped. This overlapping is not shown to make illustration more clear.

A standard way of representing a triangulation is to store it as *a point matrix* $\mathbf{Q}$, *a connectivity matrix* `T` and *an edge matrix* `E` containing the indexes of the local triangle coefficients, which are approximate to the boundary of the mesh. In contrast to a tensor product surface, where the real domain boundaries conform to the parametric domain boundaries, triangular domain requires a special representation of the boundary edges.

In the case when $\Omega \subset \mathbb{R}^2$ the point matrix $\mathbf{Q}$ is $2 \times n_G$, where each row contains x- or y-coordinate of the control points. The connectivity matrix $\mathtt{T}$ is $3\binom{d+2}{2} \times m$, where d is a degree of local triangles and m is a number of elements, contains the indexes of the control points constituting local triangles in counterclockwise order. The number of edges which belong to the boundary depends on the discretization. Suppose we have s edges making up the boundary, then the size of the matrix $\mathtt{E}$ is $2(d+1) \times s$, because one edge is approximated by two local triangles.

The matrices, involved in the variational formulation of PDE to solve, are constructed by summing the contributions of element matrices. A mapping from local to global indexation is obtained by using the connectivity matrix $\mathtt{T}$ for internal elements and the edge matrix $\mathtt{E}$ for the outer boundary. For example, for the mesh shown in Figure 4.2.1 the connectivities of the first three elements are

$$\mathtt{T} = \begin{bmatrix} 1 & 1 & 1 & \\ 2 & 10 & 3 & \\ 3 & 11 & 10 & \\ 4 & 19 & 7 & \\ 5 & 13 & 9 & \dots \\ 6 & 12 & 18 & \\ 7 & 15 & 19 & \\ 8 & 16 & 13 & \\ 9 & 17 & 12 & \end{bmatrix}.$$

Each column of $\mathtt{T}$ contains the indexes of the control points of the local triangles corresponding to the indexes of the global elements. For example, the first column corresponds to the first global triangle element having three local triangles of the first degree. Thus, the global triangle element is defined by three 3-tuples of vertices, whose indexes are the entries of the first column in the connectivity matrix $\mathtt{T}$. We note that it is important to keep counterclockwise direction of indexation for both control points of each local triangle and local triangles themselves. This approach provides surface orientation.

The mesh shown in Figure 4.2.1 has seven edges constituting the boundary. The edge matrix $\mathtt{E}$ for the first three edges has the following form

$$\mathtt{E} = \begin{bmatrix} 1 & 1 & 9 & \\ 2 & 11 & 18 & \\ 4 & 15 & 19 & \dots \\ 5 & 17 & 13 & \end{bmatrix}.$$

4.3 Coordinate transformation

4.3.1 Isoparametric elements

Since the considered elements are curvilinear, we should define coordinate transformation for them to compute the integrals constituting the element stiffness and mass matrices $\mathcal{A}$ and $\mathcal{M}$, and the element load vector b. Information regarding the appropriate location of terms $\mathcal{A}^e$, $\mathcal{M}^e$, b^e into the corresponding global matrices and vectors is stored in the connectivity matrix $\mathtt{T}$.

Let us define correspondence between barycentric curvilinear and Cartesian coordinates. We focus on one triangular element Ω^e. The mapping between the parametric triangle K and the curvilinear element Ω^e is a linear combination of control points

$Q^e = \{(q_i^x \quad q_i^y)\}_{i=1}^{n_G^K}$ of local Bézier triangles of degree d and combined expo-rational basis functions $G^e = \{G_i(u_1, u_2, u_3)\}_{i=1}^{n_G^K}$, where $n_G^K = 3\binom{d+2}{2}$

$$x = (G^e(u_1,u_2,u_3))^{\mathrm{T}} \begin{bmatrix} q_1^x \\ q_2^x \\ \dots \\ q_{n_G^K}^x \end{bmatrix}, \quad y = (G^e(u_1,u_2,u_3))^{\mathrm{T}} \begin{bmatrix} q_1^y \\ q_2^y \\ \dots \\ q_{n_G^K}^y \end{bmatrix}. \tag{4.3.1}$$

The relations (4.3.1) are valid for any local coordinate system. A slight complication with the barycentric coordinates is that they are not independent and the number of them is one more than in Cartesian coordinate system. To avoid this issue, we introduce new dependent formal variables

$$\begin{aligned} \xi &= u_1, \\ \eta &= u_2, \\ 1 - \xi - \eta &= u_3. \end{aligned} \tag{4.3.2}$$

For computation of the element matrices we need to provide two transformations. First, we express global derivatives of basis functions through local derivatives. Secondly, a differential element of area has to be represented in local coordinates and the integration limits should be correspondingly changed.

We can write partial derivatives with respect to new variables of the basis functions as

$$\begin{aligned} \frac{\partial G_i}{\partial \xi} &= \frac{\partial G_i}{\partial u_1}\frac{\partial u_1}{\partial \xi} + \frac{\partial G_i}{\partial u_2}\frac{\partial u_2}{\partial \xi} + \frac{\partial G_i}{\partial u_3}\frac{\partial u_3}{\partial \xi}, \\ \frac{\partial G_i}{\partial \eta} &= \frac{\partial G_i}{\partial u_1}\frac{\partial u_1}{\partial \eta} + \frac{\partial G_i}{\partial u_2}\frac{\partial u_2}{\partial \eta} + \frac{\partial G_i}{\partial u_3}\frac{\partial u_3}{\partial \eta}. \end{aligned} \tag{4.3.3}$$

Using (4.3.2) and (4.3.3), we get

$$\frac{\partial G_i}{\partial \xi} = \frac{\partial G_i}{\partial u_1} - \frac{\partial G_i}{\partial u_3}, \quad \frac{\partial G_i}{\partial \eta} = \frac{\partial G_i}{\partial u_2} - \frac{\partial G_i}{\partial u_3}. \tag{4.3.4}$$

Partial derivatives can be evaluated by the formula defined in Definition 4.1.1. Then, the transformation between local coordinates ξ, η and the corresponding global coordinates x, y can be written as

$$\begin{aligned} \frac{\partial G_i}{\partial \xi} &= \frac{\partial G_i}{\partial x}\frac{\partial x}{\partial \xi} + \frac{\partial G_i}{\partial y}\frac{\partial y}{\partial \xi}, \\ \frac{\partial G_i}{\partial \eta} &= \frac{\partial G_i}{\partial x}\frac{\partial x}{\partial \eta} + \frac{\partial G_i}{\partial y}\frac{\partial y}{\partial \eta}. \end{aligned}$$

Or, in matrix form

$$\begin{bmatrix} \dfrac{\partial G_i}{\partial \xi} \\ \dfrac{\partial G_i}{\partial \eta} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial x}{\partial \xi} & \dfrac{\partial y}{\partial \xi} \\ \dfrac{\partial x}{\partial \eta} & \dfrac{\partial y}{\partial \eta} \end{bmatrix} \begin{bmatrix} \dfrac{\partial G_i}{\partial x} \\ \dfrac{\partial G_i}{\partial y} \end{bmatrix} = J \begin{bmatrix} \dfrac{\partial G_i}{\partial x} \\ \dfrac{\partial G_i}{\partial y} \end{bmatrix},$$

where matrix J is the Jacobi matrix, which depends on local coordinates. The differential element of area is

$$dxdy = |J|\, d\xi d\eta. \tag{4.3.5}$$

By using (4.3.4) we find the global derivatives as

$$\begin{bmatrix} \dfrac{\partial G_i}{\partial x} \\ \dfrac{\partial G_i}{\partial y} \end{bmatrix} = J^{-1} \begin{bmatrix} \dfrac{\partial G_i}{\partial u_1} - \dfrac{\partial G_i}{\partial u_3} \\ \dfrac{\partial G_i}{\partial u_2} - \dfrac{\partial G_i}{\partial u_3} \end{bmatrix}. \tag{4.3.6}$$

Deriving J from the basis functions G_i, $i = 1, ..., n_G^K$, which define the coordinate mapping (4.3.1), we obtain

$$J = \begin{bmatrix} \sum \dfrac{\partial G_i}{\partial \xi} p_i^x & \sum \dfrac{\partial G_i}{\partial \xi} p_i^y \\ \sum \dfrac{\partial G_i}{\partial \eta} p_i^x & \sum \dfrac{\partial G_i}{\partial \eta} p_i^y \end{bmatrix} = \begin{bmatrix} \dfrac{\partial G_1}{\partial \xi} & \dfrac{\partial G_2}{\partial \xi} & \cdots \\ \dfrac{\partial G_1}{\partial \eta} & \dfrac{\partial G_2}{\partial \eta} & \cdots \end{bmatrix} Q^e =$$

$$= \begin{bmatrix} \dfrac{\partial G_1}{\partial u_1} - \dfrac{\partial G_1}{\partial u_3} & \dfrac{\partial G_2}{\partial u_1} - \dfrac{\partial G_2}{\partial u_3} & \cdots \\ \dfrac{\partial G_1}{\partial u_2} - \dfrac{\partial G_1}{\partial u_3} & \dfrac{\partial G_2}{\partial u_2} - \dfrac{\partial G_2}{\partial u_3} & \cdots \end{bmatrix} Q^e. \tag{4.3.7}$$

By analogy with (3.3.7) the inverse of the Jacobi matrix is computed as

$$J^{-1} = \frac{1}{|J|} \begin{bmatrix} \sum \dfrac{\partial G_i}{\partial \eta} q_i^y & -\sum \dfrac{\partial G_i}{\partial \xi} q_i^y \\ -\sum \dfrac{\partial G_i}{\partial \eta} q_i^x & \sum \dfrac{\partial G_i}{\partial \xi} q_i^x \end{bmatrix}. \tag{4.3.8}$$

Integration limits are changed to limits corresponding to a triangle. Finally, using formulas (4.3.5)-(4.3.8), the element stiffness and mass matrices are computed as follows

$$\mathcal{A}^e = \int_0^1 \int_0^{1-\eta} \begin{bmatrix} \dfrac{\partial G^e}{\partial x} & \dfrac{\partial G^e}{\partial y} \end{bmatrix} \begin{bmatrix} \dfrac{\partial G^e}{\partial x} \\ \dfrac{\partial G^e}{\partial y} \end{bmatrix} |J|\, d\xi d\eta,$$

$$\mathcal{M}^e = \int_0^1 \int_0^{1-\eta} (G^e)^{\mathrm{T}} G^e\, |J|\, d\xi d\eta.$$

Algorithm 5 describes an assembly process of the stiffness, mass matrices and the load vector for the two-dimensional PDE problem defined on the triangulated domain. The combined expo-rational basis G^K is defined on each triangle K by the formula (4.1.1) and the corresponding set of the partial derivatives is computed in accordance to Definition 4.1.1. A blending of the set of connected local Bézier triangles forms the domain Ω. The local to global mapping ("`l2b`") is an array of element indexes, which is contained in the connectivity matrix `T`.

Algorithm 5 Assembly of the Stiffness and Mass Matrix, and the Load Vector on Triangulations

```
1: Let $m$ be the number of triangular elements, $G^K(u_1, u_2, u_3)$ be the combined expo-
   rational basis defined on one triangle, $D_{u_\iota} G^K$, $\iota = 1, 2, 3$ be a set of its derivatives.
   The mesh is described by the set of points $\mathbf{Q} \in \mathbb{R}^2$ of the size $2 \times n_G$, the connectivity
   matrix T and the edge matrix E.
2: Allocate memory for the $n_G \times n_G$ matrices $\mathcal{M}$, $\mathcal{A}$, and $n_G \times 1$ vector $b$, and initialize
   all their entires to zero.
3: for $e = 1, 2, ..., m$ do
4:     Define the local to global mapping l2b as the $e^{\text{th}}$ column of the connectivity
   matrix T.
5:     Change the local parameters as $u_1 = \xi$, $u_2 = \eta$, $u_3 = 1 - \xi - \eta$ and compute the
   partial derivatives.
       $G^e = G^K(u_1 = \xi, u_2 = \eta, u_3 = 1 - \xi - \eta)$,
       $D_\xi G^e = D_{u_1} G^e - D_{u_3} G^e$,
       $D_\eta G^e = D_{u_2} G^e - D_{u_3} G^e$.
6:     Compute the Jacobi matrix.
       $J = \begin{bmatrix} D_\xi G^e \\ D_\eta G^e \end{bmatrix} \mathbf{Q}_{\texttt{l2b}}$.
7:     Compute the Jacobian and inverse Jacobi matrix.
       $|J| = \det(J(\xi, \eta))$,   $J^{-1} = \frac{1}{|J|} \begin{bmatrix} J[2,2] & -J[1,2] \\ -J[2,1] & J[1,1] \end{bmatrix}$.
8:     Compute the gradients $\nabla G^e$.
       $\nabla G^e = \begin{bmatrix} D_x G^e \\ D_y G^e \end{bmatrix} = J^{-1} \begin{bmatrix} D_\xi G^e \\ D_\eta G^e \end{bmatrix}$.
9:     Compute the local element matrices given by
       $\mathcal{M}^e = \int_0^1 \int_0^{1-\eta} (G^e)^{\mathrm{T}} G^e \, |J| \, d\xi d\eta$,
       $\mathcal{A}^e = \int_0^1 \int_0^{1-\eta} \left( (D_x G^e)^{\mathrm{T}} D_x G^e + (D_y G^e)^{\mathrm{T}} D_y G^e \right) |J| \, d\xi d\eta$,
       $b^e = \int_0^1 \int_0^{1-\eta} f\left(U_h(\xi, \eta)\right) (G^e)^{\mathrm{T}} \, |J| \, d\xi d\eta$.
       ▷ $U_h$ is a mapping between the $e^{\text{th}}$ triangle and the global Cartesian coordinates.
10:    Set up the local to global mapping.
       $\mathcal{M}_{\texttt{l2b,l2b}} = \mathcal{M}_{\texttt{l2b,l2b}} + \mathcal{M}^e$,  $\mathcal{A}_{\texttt{l2b,l2b}} = \mathcal{A}_{\texttt{l2b,l2b}} + \mathcal{A}^e$,
       $b_{\texttt{l2b}} = b_{\texttt{l2b}} + b^e$.
11: end for
```

4.3.2 Boundary conditions

A boundary $\partial\Omega$ consists of edges of curvilinear triangles constituting the boundary. For each of these triangles a linear combination of control points and appropriate basis functions approximates the boundary. Let the boundary $\partial\Omega$ is consisted of s edges γ_j, $j = 1, 2, ..., s$. We consider an element Ω^j such that the edge γ_j is a linear combination of basis functions defined along, for instance, u_2, and control points Q^j of the local triangles. Thus,

$$\gamma_j = \left(G^j(u_1, u_2 = 0, u_3)\right)^{\mathrm{T}} Q^j = \left(G^j(\xi, \eta = 0, 1 - \xi)\right)^{\mathrm{T}} Q^j.$$

Arrays of points Q^j belonging to each edge are located in the edge matrix `E`. Each column of this matrix contains indexes, which describe the local to global mapping ("`l2b`").

The derivative of the edge γ_j with respect to ξ is evaluated by involving the formula (4.3.4) as the linear combination of the partial derivative of the basis functions and the corresponding control points.

We transform the integral in the boundary matrix $\mathcal{R}$ in accordance with the formula for computing curvilinear integrals. The matrix $\mathcal{R}$ consists of the element matrix contributions $\mathcal{R}^j$ for each edge, $j = 1, ..., s$.

$$\mathcal{R}^j = \int_{\gamma_j} \kappa\, (G^j)^{\mathrm{T}} G^j \, d\gamma_j = \int_0^1 \kappa\, (G^j)^{\mathrm{T}} G^j \, ||D\gamma_j|| \, d\sigma,$$

where σ is a formal parameter, which is ξ or η depending on which edge belongs to the boundary. The boundary vector r can be found in a similar way.

Algorithm 6 describes a method of obtaining the matrix $\mathcal{R}$ and vector r for solving PDE on the triangulated domain. Let g_D and g_N be the Dirichlet and Neumann boundary conditions, respectively, which are defined on the boundary $\partial\Omega$. In general we assume that the boundary conditions can be mixed (as considered in Algorithm 4), but for simplicity let them be the same for the entire outer boundary. Also note that we define the basis on the boundary along the parameter u_2, which in general should depend on the parameter of the boundary edge. This formal parameter is denoted as $\sigma = [0, 1]$.

Algorithm 6 Assembling the Boundary Conditions for the Triangulated Domain

```
1: Let $G^K(u_1, u_2, u_3)$ be the combined expo-rational basis defined on one triangle $K$,
   $D_{u_\iota} G^K$, $\iota = 1, 2, 3$ be a set of its derivatives. The mesh is described by the set of
   points $\mathbf{Q}$ of the size $2 \times n_G$, the connectivity matrix T and the edge matrix E. The
   outer boundary $\partial\Omega$ consists of $s$ edges $\gamma_1, \gamma_2, ..., \gamma_s$.
2: Given the functions $g_D$ and $g_N$ and the constant $\kappa$.
3: Allocate memory for the $n_G \times n_G$ matrix $\mathcal{R}$ and $n_G \times 1$ vector $r$, and initialize all their
   entires to zero.
4: for $j = 1, 2, ..., s$ do
5:     Define the local to global mapping l2b as the jth column of the edge matrix E.
6:     Define $\sigma = [0, 1]$ as the parameter on the edge $\gamma_j$.
```

```
 7:    Find the basis on the edge $\gamma_j$.
       $G|_{\gamma_j} = G^K(u_1 = \sigma, u_2 = 0, u_3 = 1 - \sigma)$.
 8:    Find the derivative of the current edge.
       $D\gamma_j = \begin{bmatrix} D\gamma_j^x & D\gamma_j^y \end{bmatrix} = (D_{u_1} G|_{\gamma_j} - D_{u_3} G|_{\gamma_j})^{\mathrm{T}} \, \mathbf{Q}_{\mathtt{12b}}$.
 9:    Compute the local element boundary matrix $\mathcal{R}^j$.
       $\mathcal{R}^j = \int_0^1 \kappa \, G|_{\gamma_j}^{\mathrm{T}} G|_{\gamma_j} \sqrt{(D\gamma_j^x(\sigma))^2 + (D\gamma_j^y(\sigma))^2} \, d\sigma.$
10:    Compute the local element boundary vector $r$.
       $r^j = \int_0^1 (\kappa \, g_D(\sigma) + g_N(\sigma)) \, G|_{\gamma_j}^{\mathrm{T}} \sqrt{(D\gamma_\iota^x(\sigma))^2 + (D\gamma_\iota^y(\sigma))^2} \, d\sigma.$
11:    Set up the local to global mapping.
       $\mathcal{R}_{\mathtt{12b},\mathtt{12b}} = \mathcal{R}_{\mathtt{12b},\mathtt{12b}} + \mathcal{R}^j, \;\; r_{\mathtt{12b}} = r_{\mathtt{12b}} + r^j$.
12: end for
```

In Chapter 9 we solve an eigenvalue problem on a circular membrane with homogeneous Dirichlet boundary conditions. A circular domain is approximated by four ERBS triangles, two types of Bézier local triangles are considered. We demonstrate there how coarse discretization handles a complex solution shape.

Part III

Numerical experiments

Chapter 5

Regression analysis

This chapter is a modified version of [66].

5.1 Method overview

We seek to obtain a real-time continuous smooth approximation of noisy data, automatization of searching for non-stochastic deviation, and approximation with sufficient accuracy near sudden changes in the data. For more details regarding this research, we refer to [66].

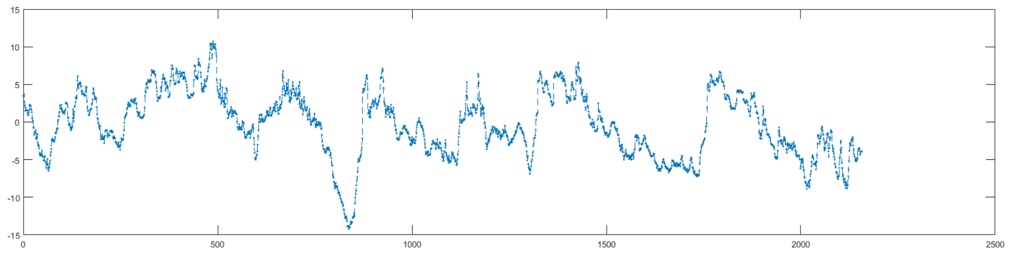
The main goal of the presented algorithm is searching for features of the data. In other words, the algorithm provides preparation for clusterization of the data. Based on that, we consider the use of data with the following definition: the 2- or 3-dimensional point set, which is the time dependence of some value, and can be divided into "stages". For example, change of temperature caused by the weather can be divided by times of day or seasons, depending on the length of the data set. In this example we consider open data from the Norwegian weather service yr.no at a specific location [89]. For the 2D data set we use the maximum temperature between 01.12.2016 and 01.03.2017, measured once per hour, as shown in Figure 5.1(a). For the 3D data set we use the data from the weather radar, as the pixel coordinates of the maximum amount of precipitation movement, as shown in Figure 5.1(b).

In order to choose our approximation method we first notice some important properties: the data set is collected throughout a long time and is has unpredictable dynamics. Additionally, the number of points increases with time, i.e., our approximation is based on the amount of points available at the current time step.

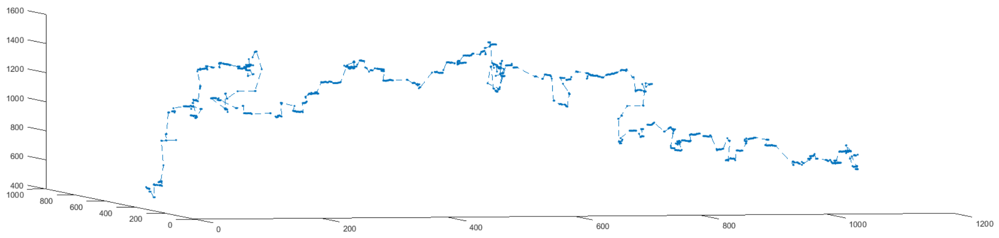
From existing methods for building regression models [4, 48, 95] we take the Multivariate Adaptive Regression Splines method (also known as MARS) [59] for comparison with our implemented method. The MARS method makes cubic B-spline approximation and provides optimization of knots positions and the number of knots. The maximum number of basis functions can be set as a parameter.

The approach is to recognize the changes between "stages" on the data action, when they occur, where the duration of one "stage" is unknown. Thus, we need to provide the approximation of each "stage". We can use polynomial approximation for this purpose because the dynamic within one "stage" does not change sharply. Then we blend these local polynomial curves together continuously and smoothly.

The considered algorithm combines several approximation methods. We initiate the approximation with a rough estimate and improve it to obtain clustered sets of points.



(a)



(b)

Figure 5.1.1: The data is taken from yr.no [89]. The dashed lines between the data points are generated to obtain a clearer view. Figure 5.1(a) shows time dependence of the temperature in Narvik during winter months, the timestep is one hour; figure 5.1(b) shows time dependence of the pixel's coordinate of the maximum amount of precipitation in Nordland taken from the weather radar.

The motivation is to keep the accuracy independent of the length of the point set, to provide stable real-time approximation, since the size of the data is unknown.

Consider the following short outline of the sequence of algorithms constituting the main algorithm proposed in the paper. Transition from one step to to the next occurs only if the step returned a value.

(i) Statistical searching of a "stage" or, in other words, element of a knot vector;

(ii) Compute the $(i-1)^{\text{th}}$ local curve ℓ_{i-1}, defined by formula (0.1.10);

(iii) Blend ℓ_{i-1} together with ℓ_{i-2} by formula (0.1.8);

Extension of the knot vector:

(a) Find candidates for knot insertion from the knot intervals;

(b) Find the positions for new knots.

The sequence above is executed for each time step.

5.2 Statistical method for real-time approximation

Our data are noisy, so we can divide them into noise and a non-stochastic part. We assume that the noise tends towards a normal distribution. We construct probability distribution functions [51] for some initial number of points, and successively add points to the point set. The deviation from the normal distribution yields a new stage of activity. Such a

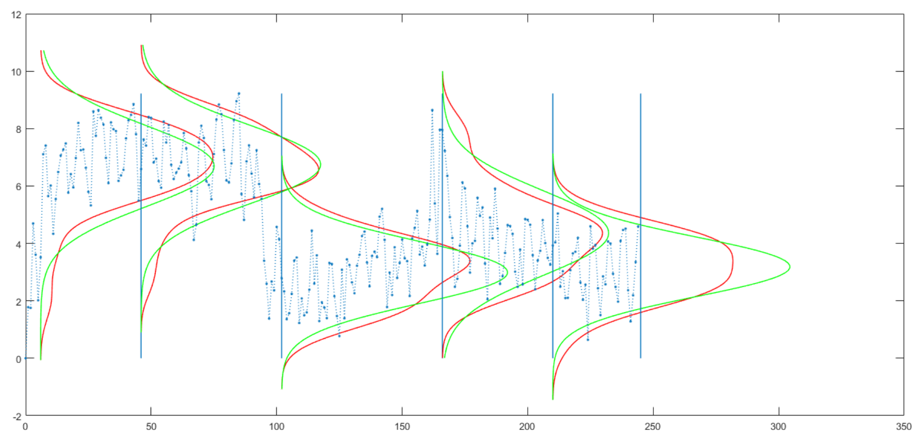


Figure 5.2.1: Illustration of statistical "stages", i.e., initial knots for 2D data (see Figure 5.1(a)). The blue lines show recognized knots, red lines show probability distribution functions of groups of data points separated by the blue lines, and the green curves are the normal distribution functions for those data points.

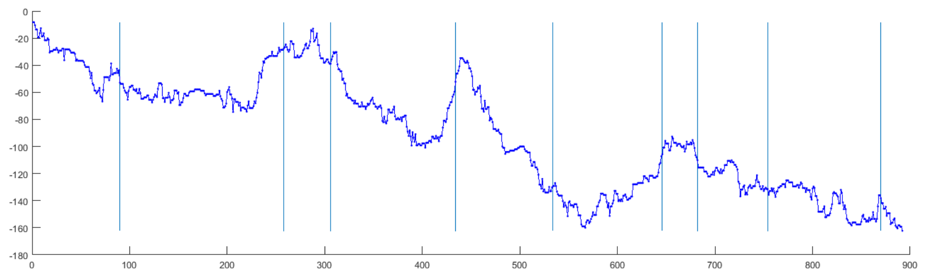


Figure 5.2.2: Illustration of searching for "stages" for 3D data (see Figure 5.1(b)). The distance between neighbor points in this figure is equal to the distance between neighbor points in 3D, and the direction corresponds to the projection on the plane determined by the t and y axes. The blue lines show the knots.

method facilitates the so-called *real-time*, that is, we run the algorithm while we receive new data.

We search for knots by comparing the normal distribution with the probability distribution function for an open ended stream of data.

A virtual example of this algorithm is shown in Figure 5.2.1. Here, the green curves are normal distributions, where the red curves are probability distribution functions (PDF) for all points between each pair of blue lines. The blue lines illustrate the *stages* array, which corresponds to the knot vector.

Let us imagine that the set of points is the random translation of one point. Then for each time step it makes a displacement with an average value of 1. The initial value for displacement is 0 for both axes. Since the PDF can be unstable for small number of points, we use a constant interval as a minimum distance between the knots.

For the $\mathbb{R}^3$ case we define translation of the point as a continuous displacement of the $\mathbb{R}^3 \to \mathbb{R}^2$ projection.

The distance between two points is

$$|\mathbf{d}_j| = |\mathbf{p}_{j+1} - \mathbf{p}_j|. \tag{5.2.1}$$

The initial displacement D_0 is zero, and accumulates the distance (5.2.1) between the first and the second point. Then we choose the direction D_{t+1} of translation as the sign of the difference between the y coordinates as

$$D_{t+1} = D_t + \operatorname{sign}(y_{t+1} - y_t)\sqrt{(x_{t+1} - x_t)^2 + (y_{t+1} - y_t)^2}. \tag{5.2.2}$$

By involving these steps we get the picture of a random walk, see Figure 5.2.2, to which we can apply the algorithm, considered above.

Figure 5.2.2 shows translation of the point as a projection from 3D to 2D, using formula (5.2.2), combined with an illustration of the searching for "stages".

We then construct local curves as Bézier curves of degree 3 and blend them together by using formula (0.1.8).

5.3 Adding knots

If the knot vector contains at least two knot intervals, we can improve the approximation of the data, i.e., increase the accuracy, by inserting new knots at certain positions.

Let $A(t)$ be a spline function. X is the discrete set of points, $\mathbf{x}_t \in X$, $\mathbf{x}_t = (t, y)$ for the 2D case and $\mathbf{x}_t = (t, x, y)$ for the 3D case, where t is the time variable. Thus, we have a point for each t and a continuous approximation of this set of points. The knot vector is denoted by $\{t_i\}_{i=0}^{n}$, where n is number of knots.

We introduce *a moving frame*, denoted by its tangent and normal vectors, $\bar{\eta}$ and $\bar{\xi}$ for the 2D case, and tangent, normal and binormal vectors $\bar{\eta}$, $\bar{\tau}$, $\bar{\xi}$, for the 3D case, respectively. Such a frame represents a moving local coordinate system.

Definition 5.3.1. The "scope" is the interior of the geometric boundary outlined by the pair of straight lines for the 2D case, or planes for the 3D case, which are defined via the knot interval. Each knot interval yields the top and the bottom borders of the "scope", which go through the points from the set $X^i \subset X$, $X^i = [\mathbf{x}_{t_i}, \mathbf{x}_{t_{i+1}}]$, which have the maximum distance from the local origin along the axis $\bar{\xi}$ of the moving frame, and are parallel to the axis $\bar{\eta}$, for the 2D case, or rectifying plane denoted by the axes $\bar{\eta}$ and $\bar{\tau}$, for the 3D case.

The process of inserting new knots consists of two steps: finding candidates for knot insertion from intervals in the knot vector, and defining the position for knot insertion.

a) The first step is based on the detection of points which are outside of the "scope". The "scope" should contain all of the points. Figure 5.3.1 describes a process for finding the indices of intervals where new knots should be inserted.

b) The second step is based on the properties of the blending spline curve. We seek to divide the knot interval by inserting a new knot at the position where we can measure the largest change in the point set. We define this to be the position with the highest speed, thus, we need to find local maximums of the first derivative of the curve. Figure 5.3.2 describe the process of inserting new knots.

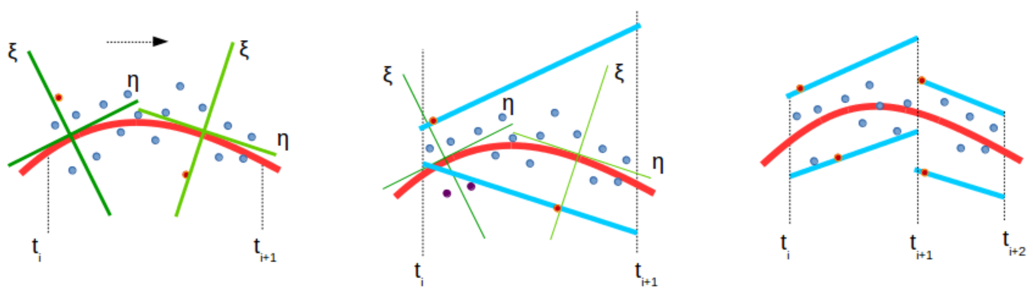


Figure 5.3.1: Finding candidates for knot insertion from the knot vector. (a) A moving frame on the curve. The blue lines in (b) are the tangent lines of the curve through the red points, corresponding to maximum and minimum distances along the $\bar{\xi}$ axis on the considered knot interval, which are used to obtain the "scope". Since there are points outside of the "scope", we insert a new knot and recompute the curve as shown in (c).

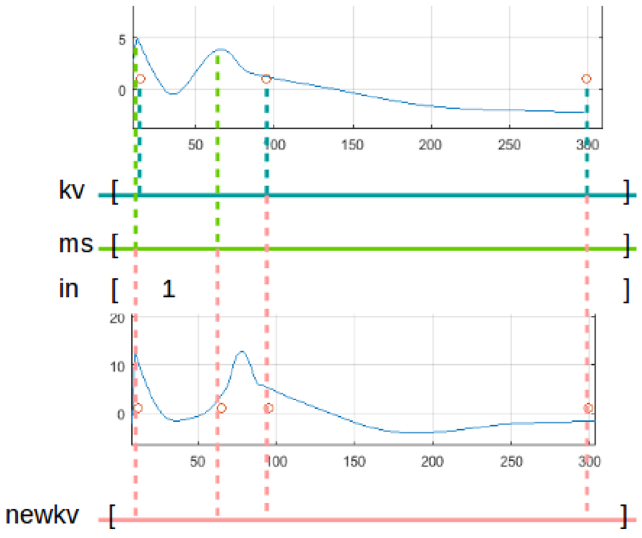


Figure 5.3.2: A process of inserting new knots. The blue curves show the derivative of the curve $A(t)$ before (top) and after (bottom) knot insertion. The red circles illustrate the positions of the knots. *kv* is the knot vector before knot insertion, *ms* is the array of positions on the time axis of the maximums of the curve first derivative, *in* is the array of indices of the knot intervals where we need to insert new knots, and *newkv* is the new knot vector, found via comparing the *kv*, *ms* and *in* arrays.

5.4 Results

Figures 5.4.1 and 5.4.3 show some steps of the proposed algorithm applied to representative examples in $\mathbb{R}^2$ and $\mathbb{R}^3$, respectively.

The approximation is a continuous smooth function which is generated by the automated algorithm. One can use intrinsic parameters of the resulting curve for further analysis. The settings of the algorithm are implicit: sensitivity to the detection of stages by changing the tolerance to the points which are outside of the "scope", and adjusting the degree of the local curves (in our case $d = 3$). We do not need to set the initial number of knots, or initial length of knot intervals, or number of iterations.

For comparison, we consider the MARS algorithm [59]. Note that the implementation of this algorithm is not real-time, so, assuming that we can realize a comparable approach, we simply run MARS for each time step.

By making a visual comparison of Figures 5.4.1 and 5.4.2 one can see the differences and similarities between the results of the two algorithms. We note that the length of the resulting knot vectors for both methods are equal, but the values are very different. For example, the accuracy of the approximation changes within the range $300 - 600$. Also one can compare the range $800 - 1200$ for both algorithms.

One cannot conclude which one is the better, based on the curves, since we do not have an original curve. However, we can discuss which method is more fit for our task and for our data.

The presented method provides flexible approximation of curves, independent of the complete data set. The curve is an affine combination of two and only two local functions on each knot interval. We do not need to keep all local curves or all "stages". This feature shows the distinction between local curves and control points. For comparison, with MARS, if we remove the first knots, we completely lose the connection with the "earlier" data points. But by using local curves, we keep the previous "stage" only as long as we need it.

Thus, we conclude that the presented algorithm is suitable for data possessing similar properties as our model data, whereas MARS is an established method with flexible settings for specific tasks.

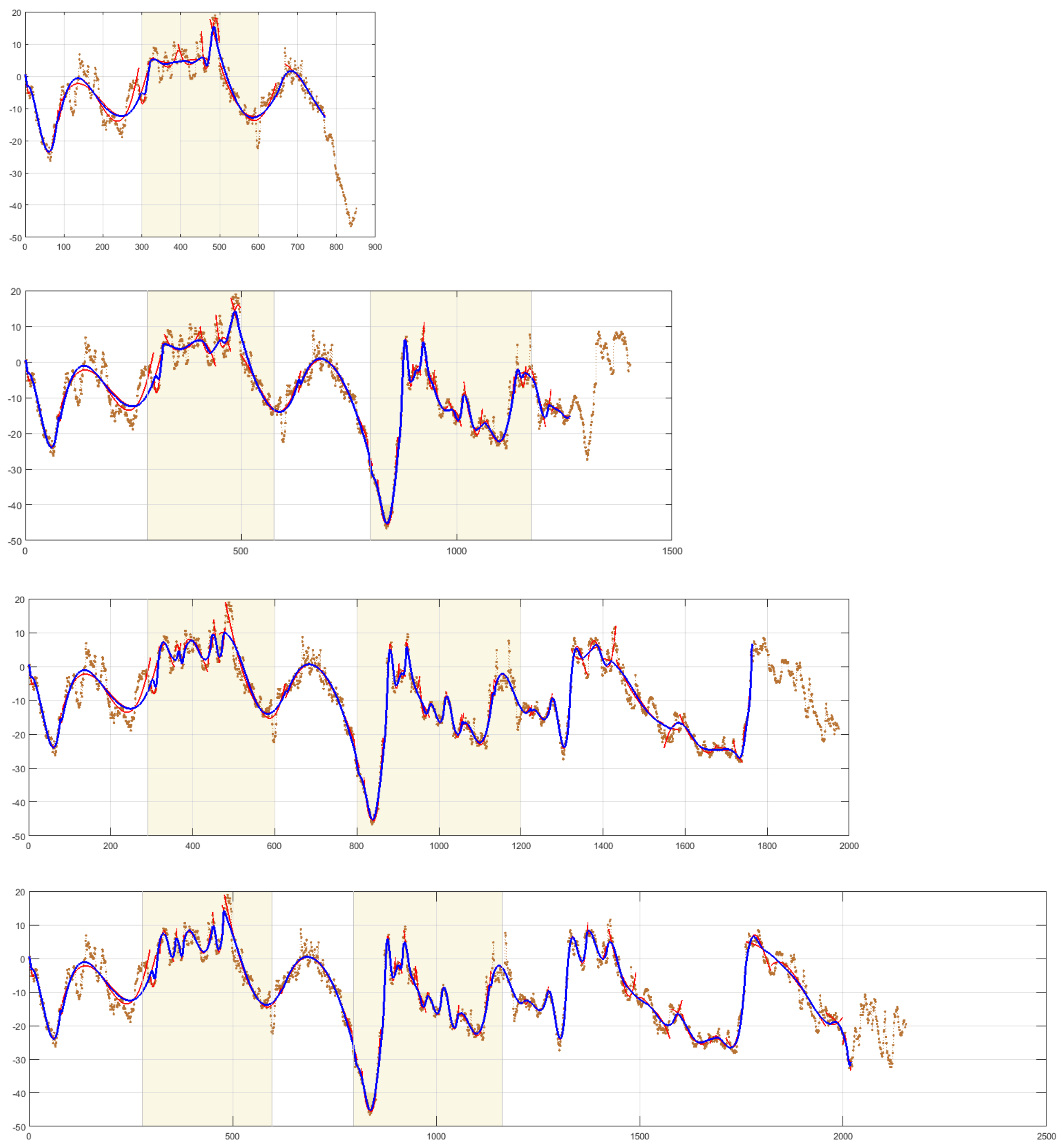


Figure 5.4.1: The process of approximation for 2D data (see Figure 5.1(a)). The red curves are local curves, the blue curve is the approximation curve.

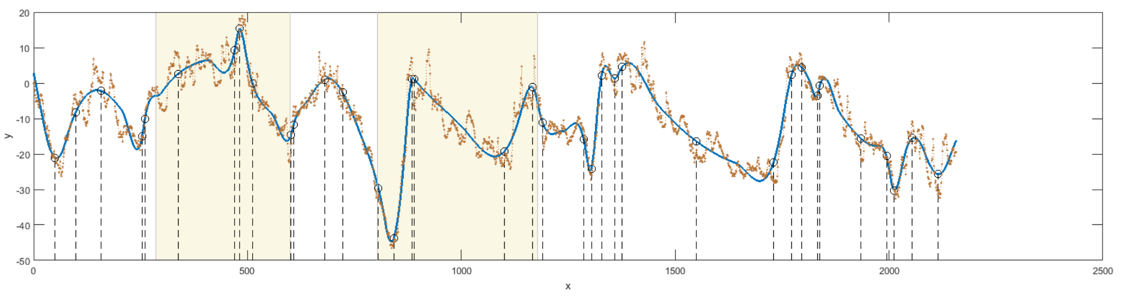


Figure 5.4.2: Result of the MARS [59] algorithm.The blue curve is the resulting curve, black circles and dashed lines show knots.

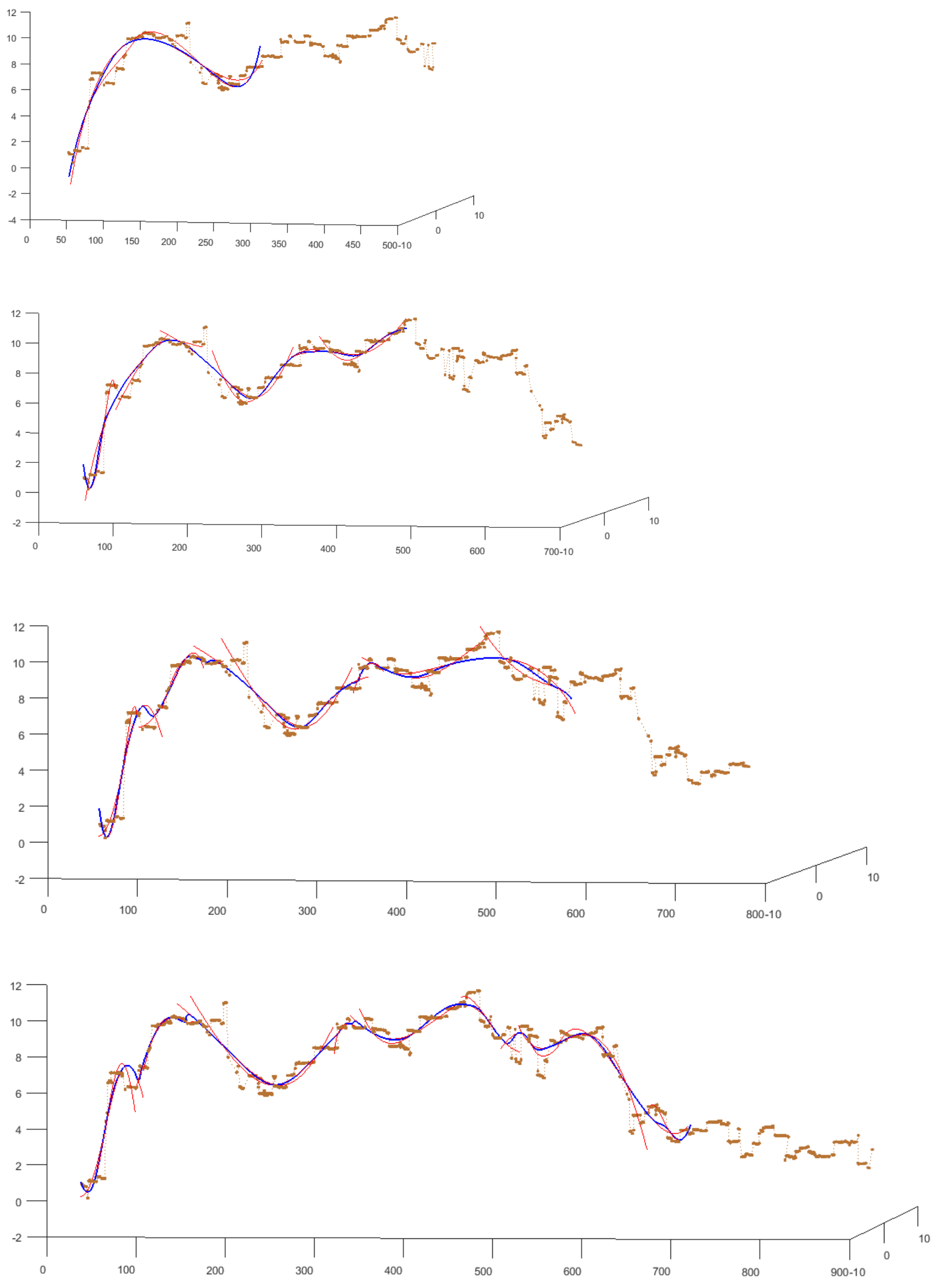


Figure 5.4.3: The process of approximation for 3D data (see Figure 5.1(b)). The red curves are local curves, the blue curve is the approximation curve.

Chapter 6

L^2-projection

6.1 Problem statement

In the following we apply the element-level projection technology for splines to formulate a target function approximated by one type of spline in terms of the different type of spline, maintaining accuracy. The basic idea is to convert the B-spline basis to the combined expo-rational basis via the expo-rational extraction operator, as a matrix named **E**. The extraction operator is a mapping of a smooth piecewise polynomial B-spline basis into a local expo-rational basis. This linear transformation allows us to represent the B-spline basis by the expo-rational basis and vice versa while keeping the degree of the complete polynomial basis locally. Moreover, the transpose of the extraction operator converts the B-spline coefficients into the control points of the local curves and vice versa. A process of constructing the expo-rational extraction operator for one-dimensional case is presented in Section 2.

Let $\Phi \subset \mathbb{R}$ be a domain and function $F \in L^2(\Phi)$ be a function to approximate. Then L^2-projection of an arbitrary function into a finite element space $\Psi_h \subset L^2(\Phi)$ is formulated as follows: find $F_h \in \Psi_h$ such that

$$\mathcal{L}(F_h) = \frac{1}{2}||F_h - F||^2_{L^2(\Phi)} \to \min.$$

The corresponding variational formulation is written as

$$\int_\Phi \mathbf{H}^\mathrm{T}\mathbf{H}\, dt\, \mathbf{c} = \int_\Phi \mathbf{H}^\mathrm{T} F\, dt, \tag{6.1.1}$$

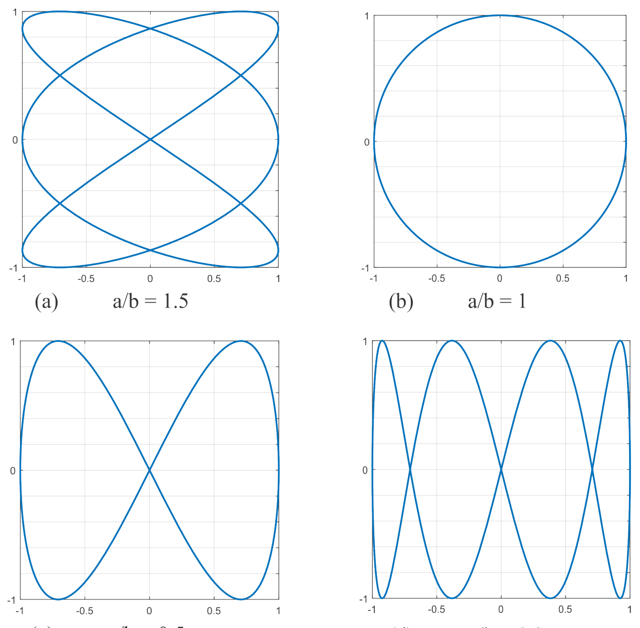
where **H** is the row vector of basis functions and **c** is the column vector of control points.

The local element version of the system (6.1.1) is the following

$$\int_{\Phi^e} (\mathbf{H}^e)^\mathrm{T}\mathbf{H}^e\; dt\; \mathbf{c}^e = \int_{\Phi^e} (\mathbf{H}^e)^\mathrm{T} F\; dt,$$

where $\mathbf{c}^e$ is the vector of control points and $\mathbf{H}^e$ is the vector of basis functions denoted over the element Φ^e only, $e = 1, ..., m$.

Modifications that we investigate in this example include (i) the conversion of the basis **H** before solving the system (6.1.1) and (ii) the conversion of the control points $\mathbf{c}^e$ after the computations. The corresponding computational formulas are presented in Table 6.1. Only the computation of the basis function subroutine needs to be modified to obtain these conversions for both B-spline and ERBS approximations. All other aspects of the



(a) a/b = 1.5 (b) a/b = 1

(c) a/b = 0.5 (d) a/b = 0.25

Figure 6.1.1: Some examples of Lissajous curves (6.1.2) with $\delta = 0$ and decreasing ratios $\frac{a}{b}$.

projection algorithm remain the same. The extraction process is automatic and can be applied to splines of any degree and any knot vector configuration.

As target functions for approximation we consider a family of Lissajous curves, the graph of the system of the parametric equations

$$F(t) = \begin{cases} \hat{A}\cos(at + \delta), \\ \hat{B}\sin(bt), \end{cases} \tag{6.1.2}$$

as closed smooth curves with a large number of inflection points. Lissajous curves can be constructed in a flexible way by changing a few parameters (Figure 6.1.1).

The Lissajous curves describe the complex harmonic motion, where $\hat{A}$ and $\hat{B}$ are the amplitudes of oscillation, a and b are the frequencies and δ is the phase shift. The rational fraction $\dfrac{a}{b}$ yields the closed curve.

6.2 Results

We construct the competitive examples in accordance with the columns of Table 6.1. Figures 6.2.1 and 6.2.2 show the visual comparison of the Lissajous curve approximation with two types of the basis: (i) B-spline (see Figure 0.1.1) and combined expo-rational (see Figure 1.1.1), (ii) their extraction one to another, and (iii) conversion of the control points. The different degrees are also presented: p and d denote the corresponding B-spline basis degree and degree of the local curves of the blending spline, respectively.

Figure 6.2.3 plots the absolute errors in terms of the distance between the target curve and its approximations. We observe that the absolute error decreases with knot vector refinement. Modifications of the B-spline approximation provide more rapid error reduction than the ERBS approximation. Using ERBS extraction, we convert the B-spline control points to corresponding control points of the local geometry and preserve the accuracy of

Standard computation, see equation (6.1.1)	
B-spline approximation	ERBS approximation
$\int\limits_{\Phi} \mathbf{N}^{\mathrm{T}}\mathbf{N}dt\,\mathbf{P} = \int\limits_{\Phi} \mathbf{N}^{\mathrm{T}}F\,dt$	$\int\limits_{\Phi} \mathbf{G}^{\mathrm{T}}\mathbf{G}dt\,\mathbf{Q} = \int\limits_{\Phi} \mathbf{G}^{\mathrm{T}}F\,dt$
Basis extraction, see equations (2.0.3) and (2.0.4)	
B-spline $\to$ ERBS basis	ERBS $\to$ B-spline basis
$(\mathbf{E}^e)^{-1} \int\limits_{\Phi^e} (\mathbf{N}^e)^{\mathrm{T}}\mathbf{N}^e dt\,(\mathbf{E}^e)^{-\mathrm{T}}\,\mathbf{Q}^e =$	$\mathbf{E}^e \int\limits_{\Phi^e} (\mathbf{G}^e)^{\mathrm{T}}\mathbf{G}^e dt\,(\mathbf{E}^e)^{\mathrm{T}}\,\mathbf{P}^e =$
$= (\mathbf{E}^e)^{-1} \int\limits_{\Phi^e} (\mathbf{N}^e)^{\mathrm{T}}F\,dt$	$= \mathbf{E}^e \int\limits_{\Phi^e} (\mathbf{G}^e)^{\mathrm{T}}F\,dt$
Conversion of control points, see equations (2.0.5), (2.0.6)	
B-spline $\to$ ERBS control points	ERBS $\to$ B-spline control points
$A(\xi)^e = ((\mathbf{E}^e)^{\mathrm{T}}\mathbf{P}^e)^{\mathrm{T}}\mathbf{G}^e(\xi)$	$S(\xi)^e = ((\mathbf{E}^e)^{-\mathrm{T}}\mathbf{Q}^e)^{\mathrm{T}}\mathbf{N}^e(\xi)$

Table 6.1: Computational formulas of L^2-projection modifications.

B-spline approximation representing the spline approximation by the interpolation of the local geometry.

Figure 6.2.4 illustrates the relation between the number of elements m on the domain Φ and the error in L^2-norm of the Lissajous curve approximations, where (a) $a/b = 0.5$ and (b) $a/b = 1.5$. All modifications from the Table 6.1 are shown on each chart.

We can divide the results of these modifications into two groups, in accordance with the error values: (1) global L^2-projection obtained with the B-spline basis, extraction of the expo-rational combined basis to the B-spline basis, conversion of the B-spline control points to the blending spline control points; (2) L^2-projection computed with the combined expo-rational basis, extraction of the B-spline basis to the expo-rational basis, conversion from the blending spline control points to the element-level B-spline control points. Figure 6.2.3 shows these groups by dashed and solid lines, respectively. Regardless of the numbers of elements, these groups show similar errors.

One can see that group (1) has the largest error when compared with group (2) for a small number of elements. The error of group (1) decreases significantly with increasing degree of the B-spline. It depends on the shape of the original curve. For example, when the curve has many inflection points, an approximation of such curve requires a sufficient degree of spline (see Figure 6.2.1 (i)-(a)). An approximation with the blending spline does not require a high degree of the local curves in the same case (see Figure 6.2.2 (i)-(a)). Since the conversion of control points handles the projected curve (see Figures 6.2.1 and 6.2.2 (iii)-(a)), the result of such conversions conforms with the original approximation (see Figures 6.2.1 and 6.2.2 (i)-(a)).

The basis extraction leads to similar curves: extraction from the B-spline basis to the combined expo-rational basis, as shown in Figure 6.2.1 (ii) and Figure 6.2.2 (i); and vice versa: extraction from the approximation with blending splines to the B-spline representation, as shown in Figure 6.2.1 (i) and Figure 6.2.2 (ii). The conversion of control points

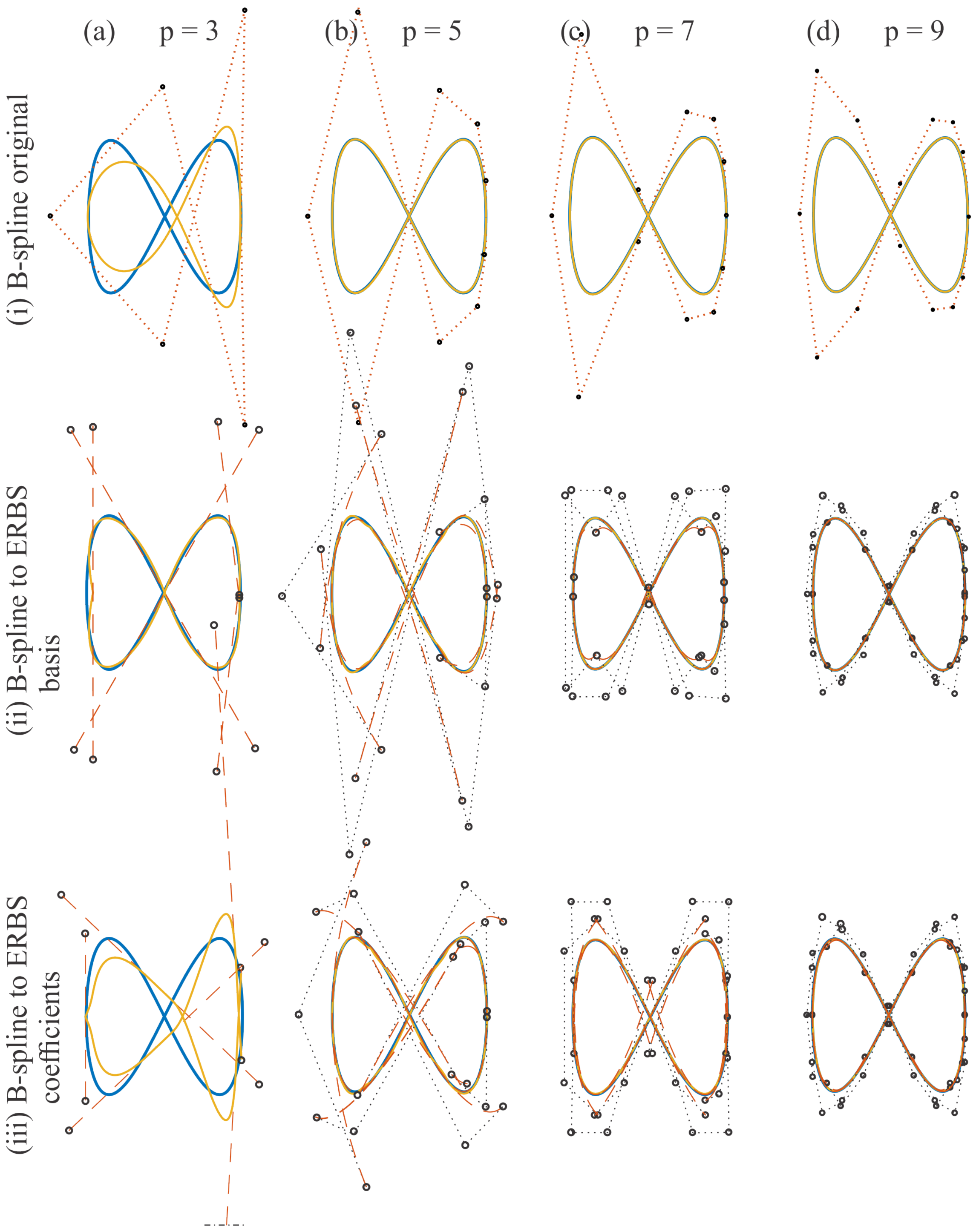


Figure 6.2.1: An approximation of the Lissajous curve (6.1.2) with $\frac{a}{b} = 0.5$ by using (i) the B-spline basis of degree (a) $p = 3$, (b) $p = 5$, (c) $p = 7$, (d) $p = 9$. (ii) An extraction into the expo-rational basis and (iii) transformation of the coefficients from the B-spline to the control points of the local curves of the blending spline.

leads to the similarity with the standard approximation on the element level: Figure 6.2.1 rows (i) and (iii); Figure 6.2.1 rows (i) and (iii) look similar, but the curves have different representation: B-spline and blending spline.

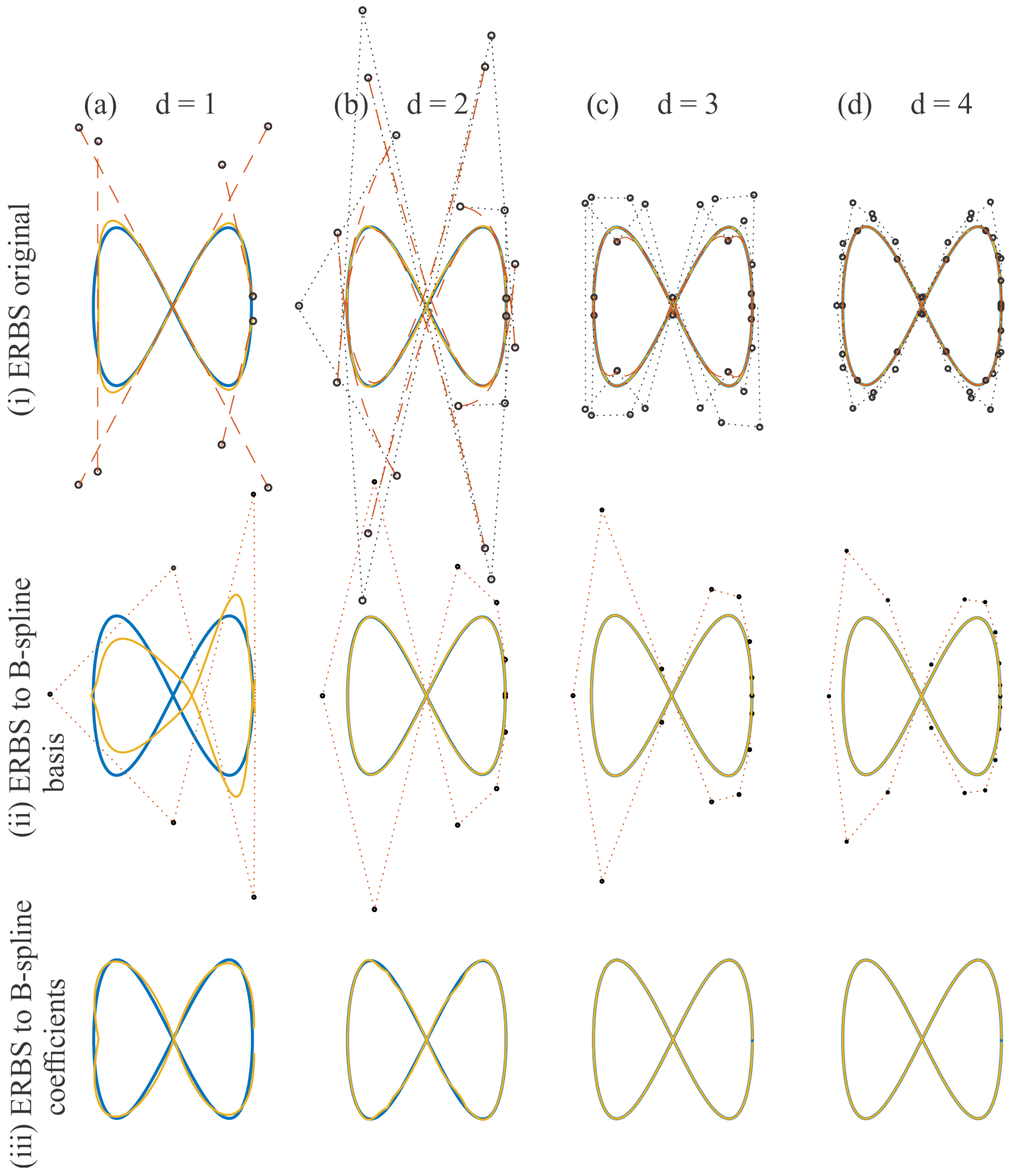


Figure 6.2.2: An approximation of the Lissajous curve (6.1.2) with $\frac{a}{b} = 0.5$ by using (i) the blending spline with local curves of degree (a) $d = 1$, (b) $d = 2$, (c) $d = 3$, (d) $d = 4$. (ii) An extraction into the B-spline basis and (iii) transformation of the coefficients from the blending spline to the control points of the B-spline on the element level (the coefficients are not shown, since most of them are located outside the visible area).

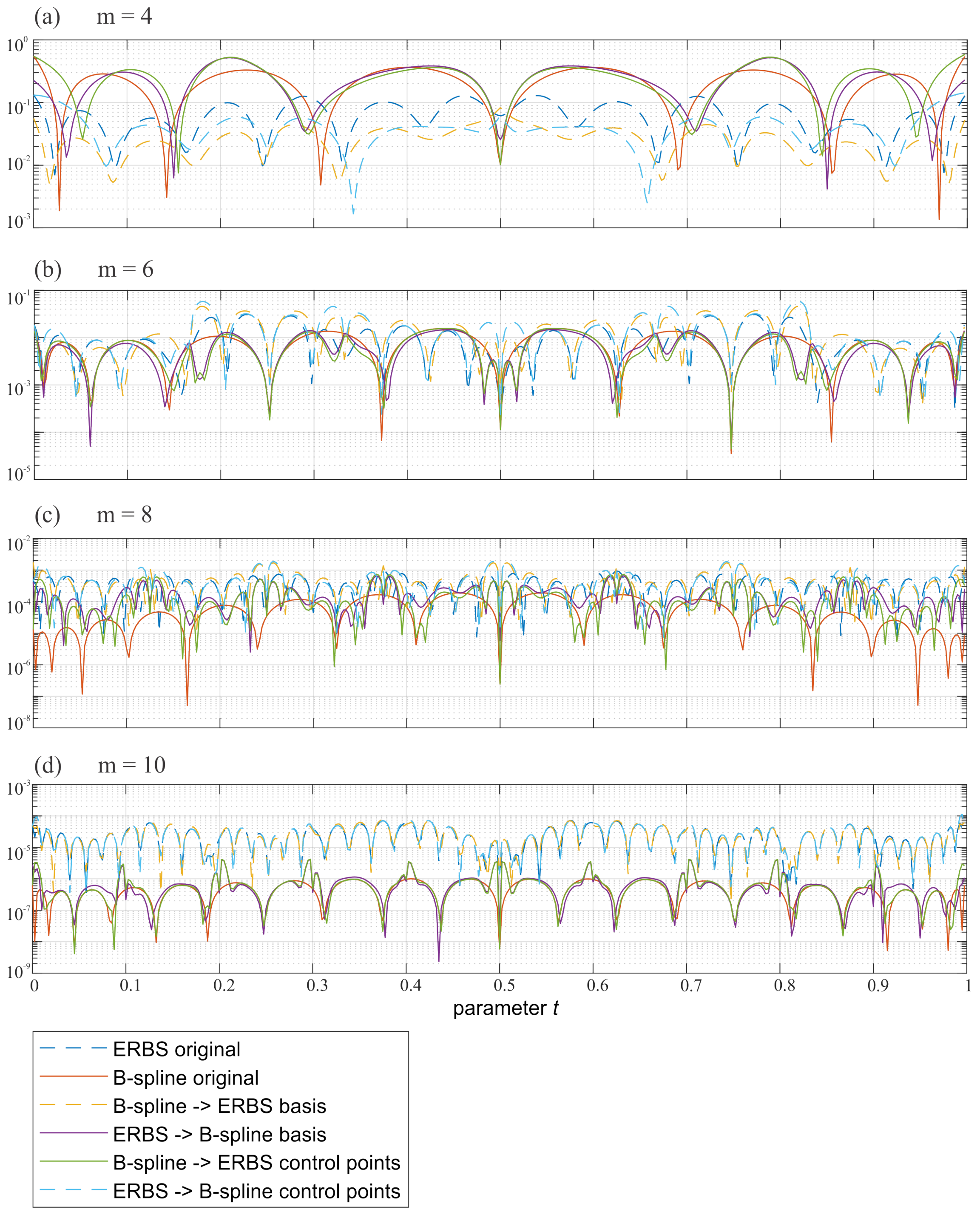
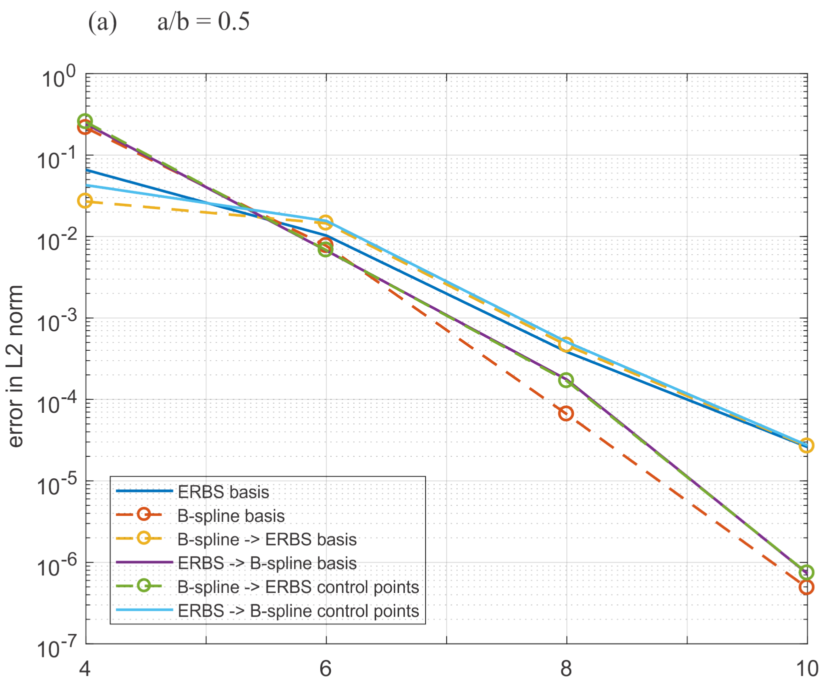


Figure 6.2.3: Absolute errors of approximations, shown in Figures 6.2.1 and 6.2.2, which are the distance between an original and an approximated curve. The errors are estimated for the different number of elements over the entire domain Φ: (a) $m = 4$, (b) $m = 6$, (c) $m = 8$, (d) $m = 10$.



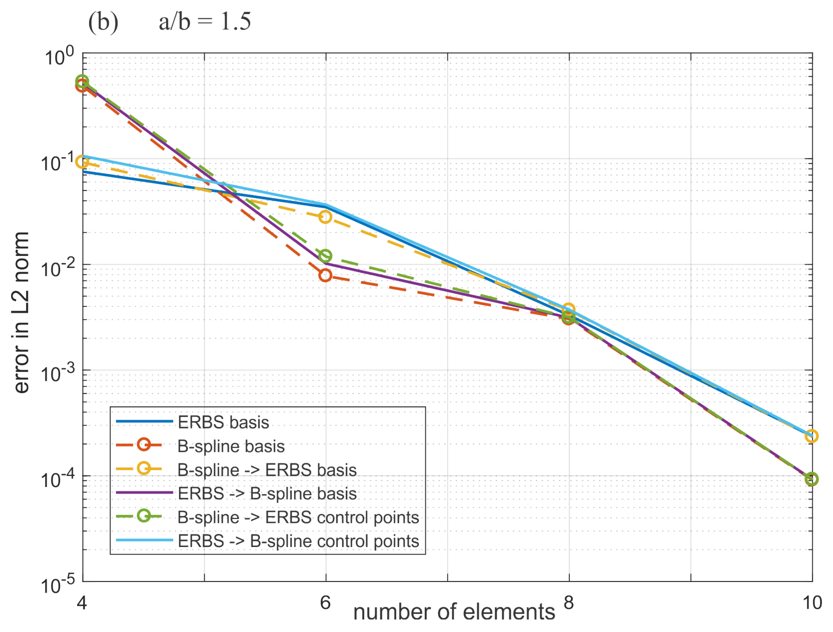


Figure 6.2.4: Errors in L^2 norm of global B-spline projection and local expo-rational projection with different modifications: extraction between basis functions and conversion between control points. (a) Approximation of the Lissajous curve with $a/b = 0.5$ (Figure 6.1.1(c)). (b) Approximation of the Lissajous curve with $a/b = 1.5$ (Figure 6.1.1(a)).

Chapter 7

Heat equation

7.1 Model problem

In this example we compare two types of bases and conversion between corresponding coefficients of the PDE solution obtained by employing the finite element method. In order to compare their performance, we opt for a time-dependent model problem whose initial condition is not smooth, while its steady-state solution is smooth. For simplicity, we consider a rectangular domain and Dirichlet boundary conditions.

A time-dependent heat equation with Robin boundary conditions is given by

$$\frac{\partial T}{\partial \tau} - \nabla \cdot (a \nabla T) = f, \qquad \text{in } \Omega = [0,1] \times [0,1],\ \tau > 0, \tag{7.1.1}$$

$$\text{BC:} \qquad -a\frac{\partial T}{\partial \overline{n}} = \kappa(T - T_D) - T_N, \qquad \text{on } \partial\Omega,\ \tau \geq 0, \tag{7.1.2}$$

$$\text{IC:} \qquad T(\Omega, 0) = T_0, \qquad \text{in } \Omega, \tag{7.1.3}$$

where Ω is a bounded convex domain with boundary $\partial\Omega$, time $\tau > 0$, and $\dfrac{\partial}{\partial \overline{n}}$ is the differentiation in the outward normal direction to $\partial\Omega$. T is the temperature to determine; $f(x, y, \tau)$ is a given source function; κ is a constant to determine a type of boundary conditions; $a(x, y) > 0$, T_D, T_N are given functions; $T_0(x, y)$ is a given initial condition. For more information regarding a PDE problem setting we refer to Section 0.1.3.

We specify given functions in the model problem (7.1.1)-(7.1.3) as follows:

$$a = 1, \quad f = -2xy(x^2 + y^2 - 2),$$

$$T_0 = (0.5 - |0.5 - x|)(0.5 - |0.5 - y|), \tag{7.1.4}$$

and to obtain the homogeneous Dirichlet boundary condition we set $\kappa = 10^6$, $T_N = 0$ and $T_D = 0$.

An exact steady-state solution for this problem is

$$T(\Omega, \infty) = \frac{1}{3}(x^3 - x)(y^3 - y). \tag{7.1.5}$$

In order to discretize the variational formulation in space, we represent a real domain Ω as a parametric domain $\Theta = [0,1] \times [0,1]$ with two independent parameters (u, v) and introduce a mesh with m elements. The temperature field is represented as

$$T_h(\Theta,\tau) = \zeta(\tau)^{\mathrm{T}}\,\mathbf{H}, \tag{7.1.6}$$

where $\zeta(\tau)$ is a vector of time-dependent coefficients and $\mathbf{H}$ is a set of bivariate basis functions defined on the parametric domain Θ.

Substituting (7.1.6) into the variational formulation and replacing test functions with basis functions $\mathbf{H}$, as demonstrated in Section 0.1.3, we obtain the variational formulation dizcretized in space.

To discretize the problem in time, let $0 = \tau_0 < \tau_1 < ... \leq \tau_L$ be a time grid with the time steps $\Delta\tau_l = \tau_l - \tau_{l-1}$, $l = 1, 2, ..., L$. Then the solution T_h can be expressed for every fixed time step. This approximation of the solution is fully discrete in the sense that it is only defined for the discrete times τ_l.

After the discretization in both space and time we obtain the following matrix equation

$$\left(\int\limits_{\Theta} \mathbf{H}^{\mathrm{T}}\mathbf{H}\,dudv + \Delta\tau_l\left(a\int\limits_{\Theta}\nabla\mathbf{H}^{\mathrm{T}}\nabla\mathbf{H}\,dudv + \int\limits_{\partial\Theta}\kappa\mathbf{H}^{\mathrm{T}}\mathbf{H}\,d\sigma\right)\right)\zeta_l =$$
$$= \int\limits_{\Theta}\mathbf{H}^{\mathrm{T}}\mathbf{H}\,dudv\,\zeta_{l-1} + \Delta\tau_l\int\limits_{\Theta} f_l\,\mathbf{H}^{\mathrm{T}}\,dudv. \tag{7.1.7}$$

An initial condition T_0 (7.1.3) can be approximated by using the L^2-projection, which principle is described in Section 6.

7.2 Results

We compare finite element approaches based on B-spline basis, combined expo-rational basis, and conversion between corresponding coefficients of the approximated surface (see Figure 2.0.5) with exact solutions for $\tau = 0$ (7.1.4) and $\tau = \infty$ (7.1.5), which are shown in Figure 7.2.1.

Several mesh types are used for the illustration. Figure 7.2.3 shows some examples of the mesh configurations, corresponding control nets for the B-spline surface and local surfaces of the blending type tensor product surface. In accordance with the restriction (2.0.1), the B-spline and ERBS representation of the surface are consistent, i.e. convertible one to another, for each row 1)-3) in Figure 7.2.3. Note that the local surfaces in Figure 7.2.3(c) are scaled to be easier recognizable. In reality, they partially overlap neighbor local surfaces. The support of each local surface covers four neighbor elements.

Figure 7.2.2 shows solutions of the heat equation described in the beginning of this section on the squared domain Ω. The problem is solved by using both the B-spline basis and the combined expo-rational basis. The first and the third columns demonstrate the results obtained by solving the matrix equation (7.1.7), where the basis $\mathbf{H}$ is replaced with the B-spline basis $\mathbf{N}$ and the combined expo-rational basis $\mathbf{G}$, respectively. The second column illustrates the result of conversion from control points of the B-spline surface to control points of the local geometry of the expo-rational tensor product surface, obtained by formula (2.0.9). The last column shows the conversion from ERBS surface to B-spline surface, obtained by formula (2.0.10). The mesh configuration shown in Figure 7.2.3(1) is used. A mesh of size 3×3, third degree B-spline and first degree local surfaces are selected to better illustrate the different approaches. From the locality of the combined expo-rational basis it follows that we obtain improved approximation of the non-smooth surfaces

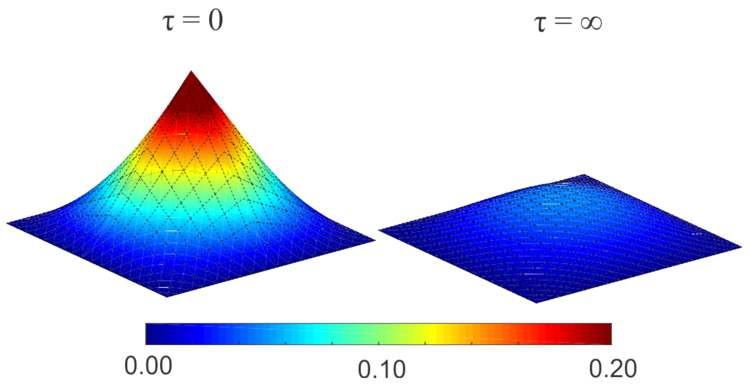


Figure 7.2.1: An exact solution of the heat conduction problem. An initial condition ($\tau = 0$) and a steady-state solution ($\tau = \infty$) are shown.

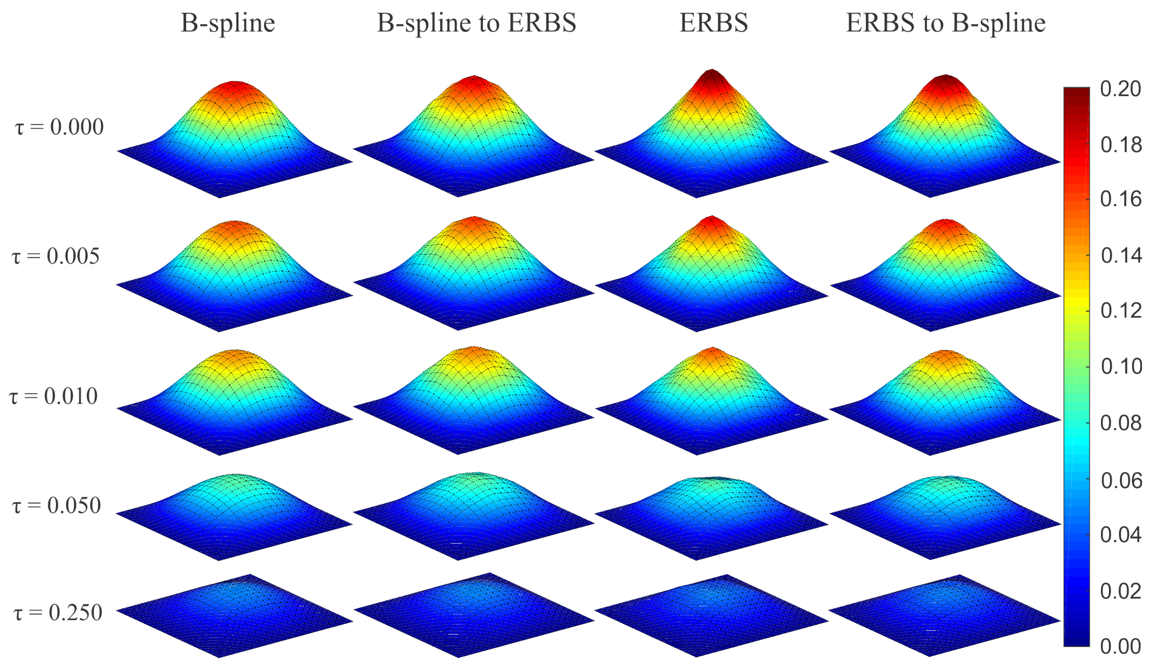


Figure 7.2.2: Time-dependent simulation of the heat conduction problem.

when compared to the B-spline of the low degree on the uniform domain. However, the conversion from ERBS to B-spline (the last column) demonstrates closer approximation to the exact initial condition than the original B-spline. A smooth steady-state solution is approximated better by the B-spline tensor product surface, which is also demonstrated in Figure 7.2.4.

Figure 7.2.4 compares absolute errors for each considered type of approximation for $\tau = 0$ and $\tau = \infty$, on the mesh configuration shown in Figure 7.2.3(3). The error scaling is selected differently for the initial condition approximation and for the steady-sate solution approximation, because an approximation of the non-smooth surface yields a larger error. Converted surfaces are evaluated at the element level, which is reflected in error plots. An ERBS surface obtained from B-spline approximation of the steady-state solution (Figure 7.2.4(2b)) has a smaller error than the original ERBS approximation (Figure 7.2.4(3b)). A B-spline conversion from the ERBS approximation of the initial

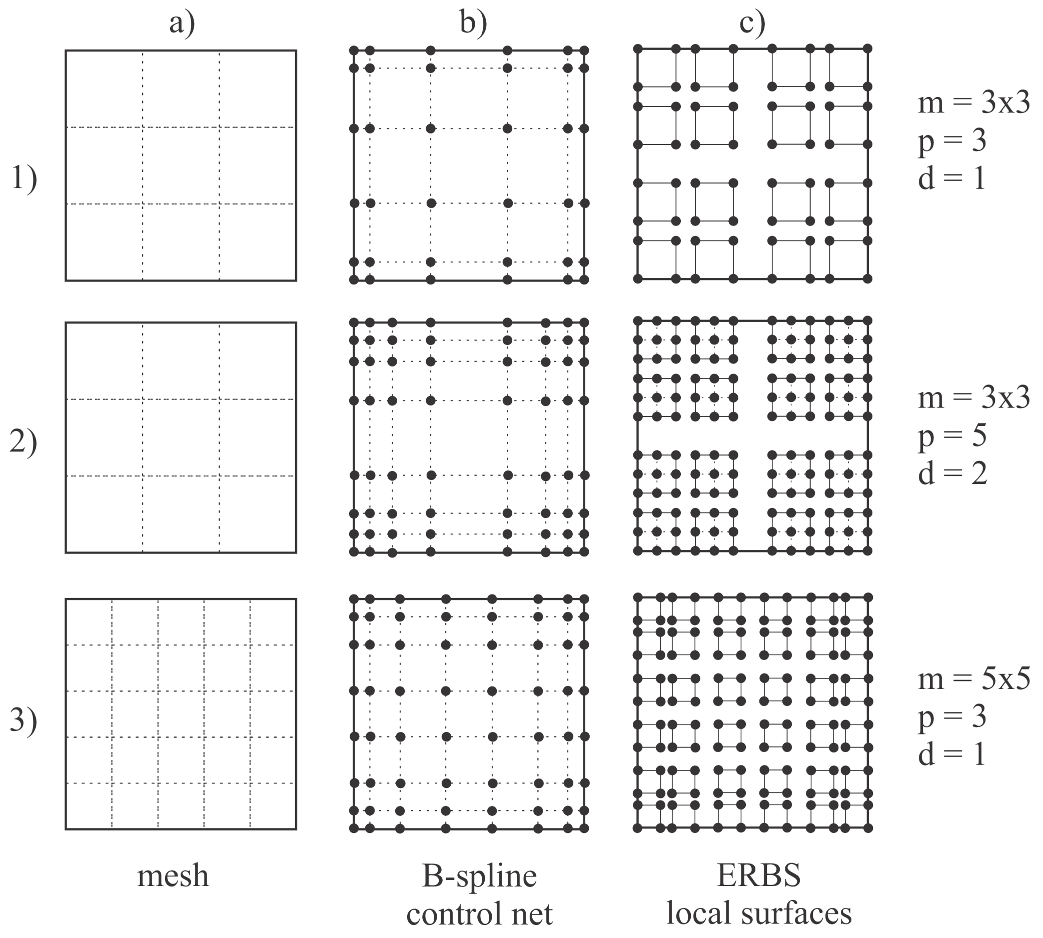


Figure 7.2.3: Examples of some mesh configurations. The columns b) and c) demonstrate correspondence between the B-spline (degree p) control net and the set of local surfaces (degree d) of blending tensor product surface.

condition (Figure 7.2.4(4a)) demonstrates improved performance over the original B-spline approximation (Figure 7.2.4(2a)).

Different types of error for the proposed examples and mesh configurations are listed in Table 7.1. One can see that increasing of both spline degree and number of elements eliminates the difference between an original approximation (B-spline or ERBS) and its conversion to another one. Moreover, the blending spline representation of B-spline (B-spline to ERBS) is more accurate than ERBS to B-spline. The reason for this is the locality of the expo-rational basis, which provides better flexibility of the approximated surface shape.

The considered method can be directly utilized on the isoparametric rectangular elements conserving all the properties and features of the expo-rational basis.

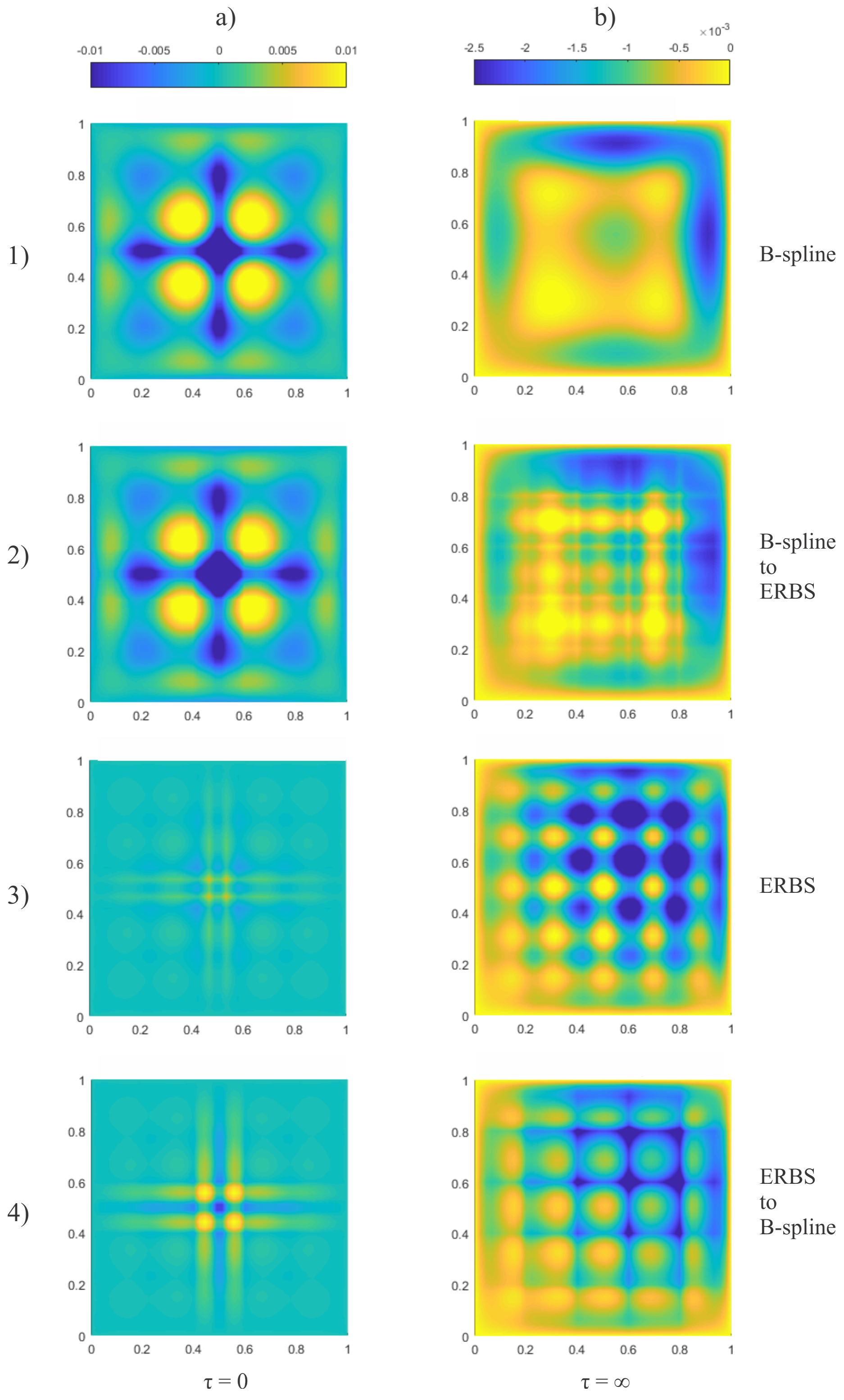


Figure 7.2.4: Comparison of absolute errors for B-spline and combined expo-rational bases, conversion from B-spline surface to blending surface and vice versa. The mesh of size 5 × 5, the third degree B-spline basis and the first degree of local surfaces are used.

Example	Mesh size	Degree	Basis	Max error	Average error	L^2 error
$\tau = 0$	3×3	$p = 3$	B-spline	0.0418	5.85×10^{-3}	0.00702
			B-spline to ERBS	0.0351	5.41×10^{-3}	0.00661
		$d = 1$	ERBS	0.0083	7.90×10^{-4}	0.00101
			ERBS to B-spline	0.0157	1.81×10^{-3}	0.00254
		$p = 5$	B-spline	0.0259	3.50×10^{-3}	0.00387
			B-spline to ERBS	0.0259	3.41×10^{-3}	0.00386
		$d = 2$	ERBS	0.0076	5.53×10^{-4}	0.00085
			ERBS to B-spline	0.0081	7.10×10^{-4}	0.00108
	5×5	$p = 3$	B-spline	0.0330	3.23×10^{-3}	0.00404
			B-spline to ERBS	0.0295	3.14×10^{-3}	0.00386
		$d = 1$	ERBS	0.0054	3.47×10^{-4}	0.00045
			ERBS to B-spline	0.0102	7.39×10^{-4}	0.00143
		$p = 5$	B-spline	0.0245	2.35×10^{-3}	0.00282
			B-spline to ERBS	0.0245	2.31×10^{-3}	0.00281
		$d = 2$	ERBS	0.0048	2.24×10^{-4}	0.00036
			ERBS to B-spline	0.0043	2.51×10^{-4}	0.00032
$\tau = \infty$	3×3	$p = 3$	B-spline	0.0040	1.46×10^{-3}	0.00188
			B-spline to ERBS	0.0046	1.67×10^{-3}	0.00207
		$d = 1$	ERBS	0.0067	2.10×10^{-3}	0.00263
			ERBS to B-spline	0.0067	1.83×10^{-3}	0.00224
		$p = 5$	B-spline	0.0033	9.13×10^{-4}	0.00119
			B-spline to ERBS	0.0033	8.99×10^{-4}	0.00117
		$d = 2$	ERBS	0.0032	1.26×10^{-3}	0.00151
			ERBS to B-spline	0.0040	1.27×10^{-3}	0.00154
	5×5	$p = 3$	B-spline	0.0024	7.20×10^{-4}	0.00094
			B-spline to ERBS	0.0024	7.93×10^{-4}	0.00102
		$d = 1$	ERBS	0.0036	1.17×10^{-3}	0.00147
			ERBS to B-spline	0.0036	1.05×10^{-3}	0.00130
		$p = 5$	B-spline	0.0019	4.33×10^{-4}	0.00060
			B-spline to ERBS	0.0019	4.28×10^{-4}	0.00059
		$d = 2$	ERBS	0.0019	6.24×10^{-4}	0.00079
			ERBS to B-spline	0.0020	6.18×10^{-4}	0.00063

Table 7.1: Comparison of different types of error for two examples: approximation of the initial condition ($\tau = 0$) and the steady-state solution ($\tau = \infty$), obtained by solving the time-dependent heat equation (7.1.1)-(7.1.3). Two types of basis functions: B-spline and combined expo-rational are presented. For each combination of mesh size, spline type and its degree the corresponding conversions are provided: B-spline control points to local surfaces, and local surfaces to B-spline control points. Here, p is the B-spline degree and d is the degree of the local Bézier surfaces.

Chapter 8

Poisson's equation

8.1 Model problem

To illustrate the ERBS-based finite element method, we consider a test problem with a known solution. This model problem is compiled from examples in two sources [62] and [86]. As shown in Figure 8.1.1(a), we choose a domain Ω which is bounded by the quarter annulus, located within the positive quadrant of the Cartesian coordinate system. A boundary $\partial\Omega$ consists of four edges $\partial\Omega = \{\gamma_1, \gamma_2, \gamma_3, \gamma_4\}$.

A model problem is a Poisson's equation given by

$$-\Delta\vartheta = f, \quad \text{in } \Omega. \tag{8.1.1}$$

The boundary conditions are

$$\begin{cases} \vartheta = g_D, & \text{on } \gamma_1, \\ \vartheta = 0, & \text{on } \gamma_2,\ \gamma_3,\ \gamma_4, \end{cases} \tag{8.1.2}$$

where $g_D = (x^4 - 16x^2 + 17)\sin(x)$.

We define the load f in such a way that the exact solution, see Figure 8.1.1(b), reads

$$\vartheta = (\rho^2 - 1)(\rho^2 - 16)\sin(x).$$

The finite element method for the problem (8.1.1)-(8.1.2) is formulated as follows

$$\left(\int\limits_{\Omega} \nabla\mathbf{G}^{\mathrm{T}}\nabla\mathbf{G}\, d\Omega + \int\limits_{\partial\Omega} \kappa\, \mathbf{G}^{\mathrm{T}}\mathbf{G}\, d(\partial\Omega) \right) \zeta = \int\limits_{\Omega} f\, \mathbf{G}^{\mathrm{T}}\, d\Omega + \int\limits_{\partial\Omega} \kappa\, g_D\, \mathbf{G}^{\mathrm{T}}\, d(\partial\Omega), \tag{8.1.3}$$

where κ is a constant which affects the type of the boundary conditions. Sufficiently high κ gives the Dirichlet boundary conditions. For more details we refer to Section 0.1.3.

In the compact matrix form (8.1.3) can be rewritten as

$$(\mathcal{A} + \mathcal{R})\zeta = b + r, \tag{8.1.4}$$

where $\mathcal{A}$ is a stiffness matrix, b is a load vector, $\mathcal{R}$ and r are the boundary matrix and vector, respectively.

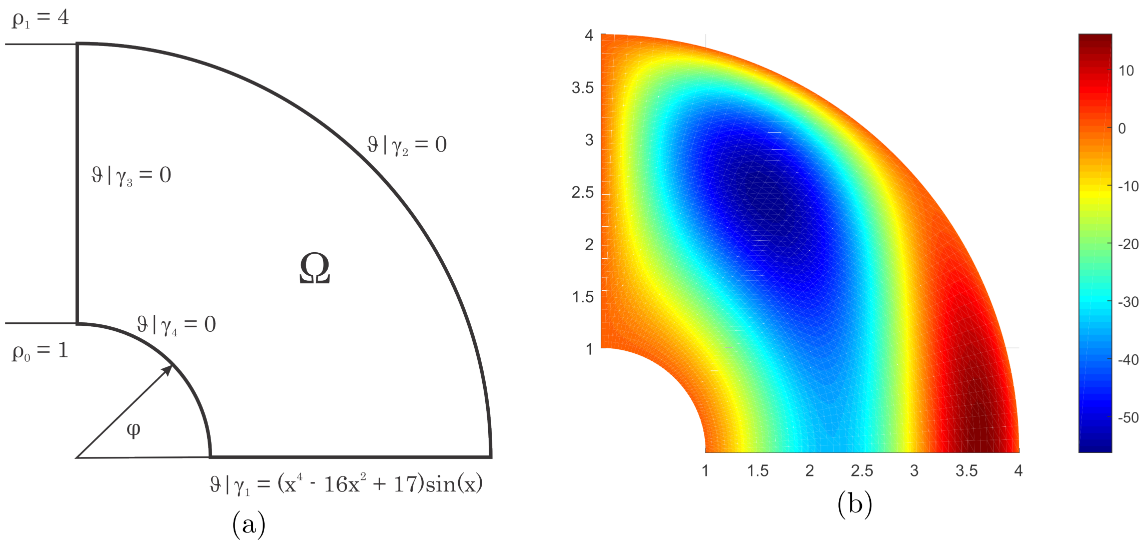


Figure 8.1.1: (a) Geometry and boundary conditions. (b) Exact solution.

8.2 Numerical solution

We first discretize the domain by introducing a mapping $U_h : \Theta = [0,1] \times [0,1] \to \Omega \subset \mathbb{R}^2$. We use L^2-projection (see Section 3.2) to find coefficients $\mathbf{Q} \in \mathbb{R}^2$. A linear combination of combined expo-rational basis functions and corresponding control points is used as a domain for further computations

$$U_h = \mathbf{Q}^{\mathrm{T}}\,\mathbf{G}. \tag{8.2.1}$$

A mapping (8.2.1) represents a mesh, where elements are given by tensor product of two knot vectors. A number of coefficients $\mathbf{Q}$ corresponds to the number of basis functions $\mathbf{G}$. An example of 5×5 mesh is shown in Figure 8.2.1. A set of control points, which are shown as light blue points in Figure 8.2.1, form a 6×6 set of local Bézier surfaces of degree 1. These local surfaces overlap each other and interpolate the resulting blending surface.

We compute integrals from (8.1.3) on each element Ω^e and evaluate the integral at any point, defined in local coordinates (ξ, η), by changing the coefficients $\mathbf{Q} = \{\mathbf{x}, \mathbf{y}\}$. An approach to coordinate transformations is detailed in Section 3.3. We perform the computations in accordance with Algorithm 3, i.e., for each element Ω^e the element stiffness matrix is computed as

$$\mathcal{A}^e = \int\limits_{\Omega^e} \nabla\mathbf{G}^{\mathrm{T}}\nabla\mathbf{G}\,dxdy = \int\limits_0^1\int\limits_0^1 \frac{1}{|J|}\begin{bmatrix} D_\xi\mathbf{G}(\xi,\eta) \\ D_\eta\mathbf{G}(\xi,\eta)\end{bmatrix}^{\mathrm{T}} \mathcal{B} \begin{bmatrix} D_\xi\mathbf{G}(\xi,\eta) \\ D_\eta\mathbf{G}(\xi,\eta)\end{bmatrix} d\xi d\eta, \tag{8.2.2}$$

where

$$|J| = \mathbf{x}^{\mathrm{T}}\left[D_\xi\mathbf{G}^{\mathrm{T}}D_\eta\mathbf{G} - D_\eta\mathbf{G}^{\mathrm{T}}D_\xi\mathbf{G}\right]\mathbf{y}$$

and

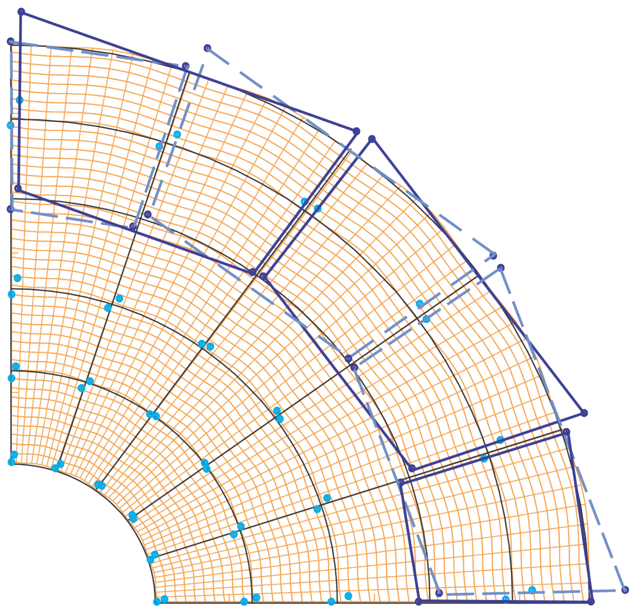


Figure 8.2.1: Domain discretization. An initial mesh has 5×5 elements. Control points and some of local surfaces of degree 1 are shown.

$$\mathcal{B} = \begin{bmatrix} \mathbf{x}^{\mathrm{T}} D_\eta \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\eta \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y} & -\left(\mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y}\right) \\ -\left(\mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\eta \mathbf{G}\, \mathbf{y}\right) & \mathbf{x}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\xi \mathbf{G}\, \mathbf{x} + \mathbf{y}^{\mathrm{T}} D_\xi \mathbf{G}^{\mathrm{T}} D_\xi \mathbf{G}\, \mathbf{y} \end{bmatrix}.$$

The load vector is evaluated as follows

$$b^e = \int\limits_{\Omega^e} f\, \mathbf{G}^{\mathrm{T}}\, dx dy = \int\limits_0^1 \int\limits_0^1 f(U_h(\xi,\eta))\mathbf{G}^{\mathrm{T}}\, |J|\, d\xi d\eta. \tag{8.2.3}$$

Boundaries and their derivatives are computed in accordance with Algorithm 4. In our case, only the first edge γ_1 has inhomogeneous Dirichlet boundary conditions. Then, the boundary matrix $\mathcal{R}$ can be obtained by the formula (3.3.14) and the formula (3.3.15) for the boundary vector r reduces to

$$r = \int\limits_0^1 \kappa\, g_D|_{\gamma_1}\, \mathbf{G}|_{\gamma_1}^{\mathrm{T}}\, ||D\gamma_1||\, d\sigma. \tag{8.2.4}$$

Substituting (8.2.2), (8.2.3), (3.3.14) and (8.2.4) into (8.1.4) and solving this matrix equation, we finally obtain a discrete approximation of the solution as

$$\vartheta_h = \zeta^{\mathrm{T}}\, \mathbf{G}.$$

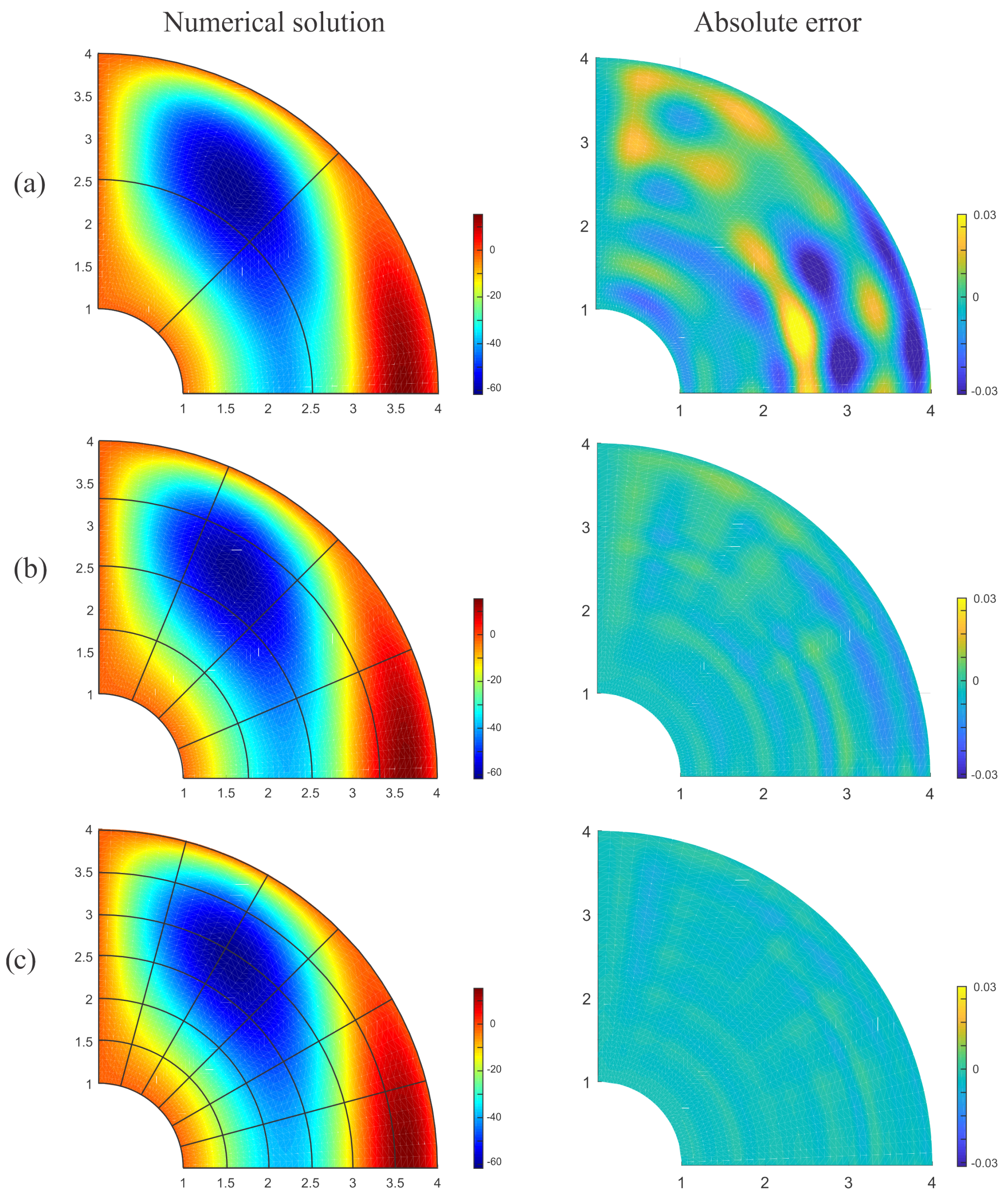


Figure 8.3.1: Numerical solutions and absolute errors for the Poisson's equation solved via ERBS finite elements. The mesh of size (a) 2×2, (b) 4×4, (c) 6×6, and the second degree local surfaces are used.

8.3 Results

In this section we show the result of applying ERBS finite elements in the context of solving PDEs by following the algorithm described in Section 8.2. We briefly demonstrate the accuracy of the usage of ERBS-based finite elements for the imposition of boundary conditions and the approximation of a solution on a mesh represented by the blending tensor product surface.

Figure 8.3.1 shows several solutions of the PDE problem (8.1.1)-(8.1.2) and the corresponding absolute errors. The blending tensor product surface with local Bézier surfaces of polynomial degree two are used. Absolute errors show convergence with respect to the uniform mesh refinement.

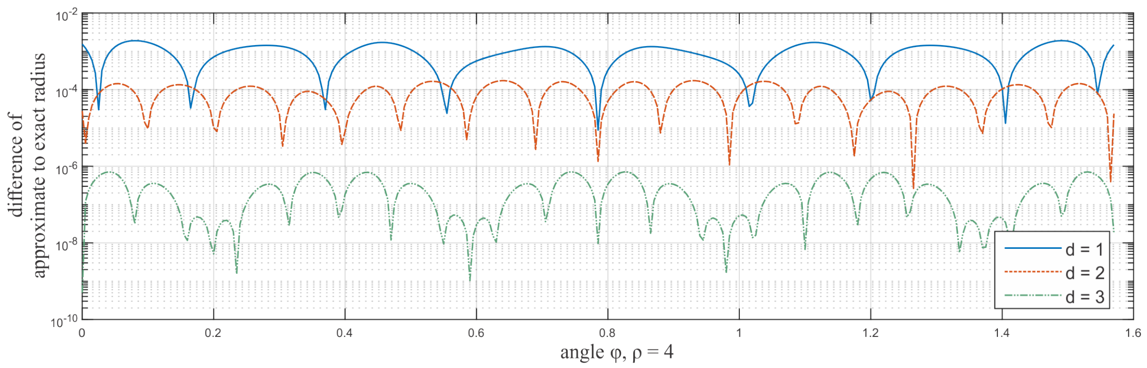


Figure 8.3.2: Absolute difference between the outer edge of the domain approximation (parameter u is equal to 1) and the exact outer radius $\rho = 4$. This error is plotted for the mesh of size 4×4 and three different degrees of local surfaces.

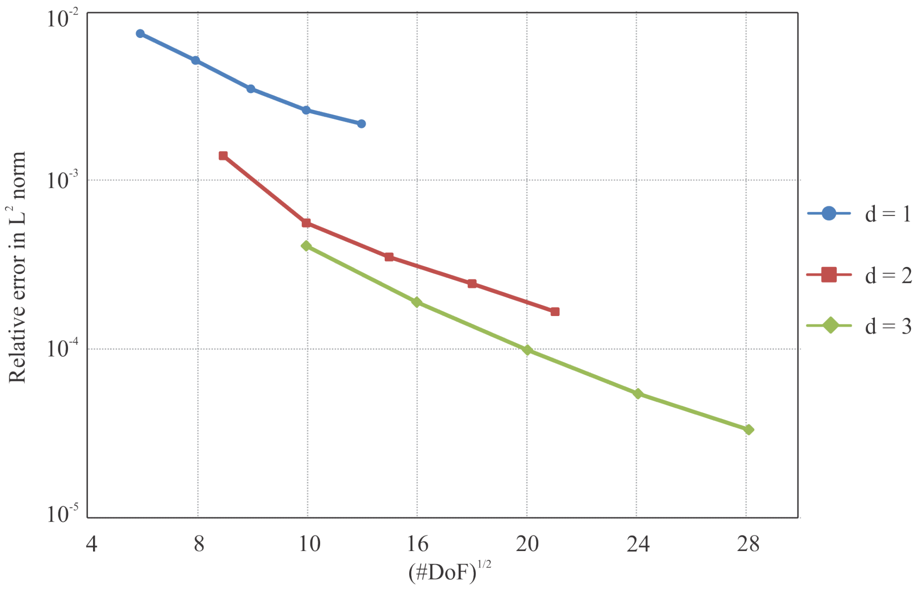


Figure 8.3.3: Error in L^2 norm of the Poisson's problem given in Figure 8.1.1 approximated by using the ERBS finite element approach. Three different degrees of the local surfaces are demonstrated: $d = 1$, $d = 2$, $d = 3$.

Figure 8.3.2 plots the geometry error in terms of the difference between the exact radius of the circular annulus and its approximation evaluated along the outer edge. We observe that the error rapidly decreases with increasing the degree of local surfaces.

To be more general, let us compare errors in L^2 norm for the different meshes and local degrees. Figure 8.3.3 demonstrates the convergence of the algorithm under the uniform mesh refinement.

We conclude that the blending tensor surface representation of finite element method applied to Poisson's equation with inhomogeneous boundary conditions yields reasonable convergence rate for the considered problem specifications. To be more precise in accuracy estimation, a deeper analysis should be performed.

Chapter 9

Eigenvalue problem

This chapter is a modified version of [65].

9.1 Model problem

We now demonstrate a process of solving an eigenvalue problem on a circular membrane with fixed outer boundary by using ERBS triangles.

A detailed description of obtaining an analytical solution of the problem of membrane vibrations can be found in [52]. Consider a circular membrane Ω having a radius $\rho = a$ and a fixed outer boundary $\partial\Omega$. Introduce a given material constant $\hat{c}$ and a circular frequency $\hat{\omega}$.

Define the eigenvalue problem in the following form

$$\Delta\vartheta + \frac{\hat{\omega}^2}{\hat{c}^2}\vartheta = 0, \quad \text{in } \Omega, \tag{9.1.1}$$

$$\vartheta = 0, \quad \text{on } \partial\Omega. \tag{9.1.2}$$

An analytical solution of the problem (9.1.1) with boundary condition (9.1.2) is represented by two independent orthogonal eigenfunctions, referred to as the cosine and the sine modes, respectively

$$\vartheta_C^{(\mathtt{m},\mathtt{n})} = J_{\mathtt{m}}(\hat{\omega}_{(\mathtt{m},\mathtt{n})}\rho/\hat{c})\cos(\mathtt{m}\varphi) \quad \text{and} \quad \vartheta_S^{(\mathtt{m},\mathtt{n})} = J_{\mathtt{m}}(\hat{\omega}_{(\mathtt{m},\mathtt{n})}\rho/\hat{c})\sin(\mathtt{m}\varphi), \tag{9.1.3}$$

where $J_{\mathtt{m}}$ is a Bessel function.

The circular eigenfrequencies $\hat{\omega}_{(\mathtt{m},\mathtt{n})}$ of the $(\mathtt{m},\mathtt{n})$ mode can be found from the formula

$$\hat{\gamma} = \hat{\omega}/\hat{c}.$$

The eigenvalues $a\hat{\gamma}_{(\mathtt{m},\mathtt{n})}$ denote from the boundary condition (9.1.2), which yields the characteristic equation

$$J_{\mathtt{m}}(\hat{\gamma}a) = 0.$$

The first few mode-shapes are shown in Figure 9.1.1.

By replacing $\hat{\omega}^2/\hat{c}^2 = \lambda$, one can formulate the finite element method for the problem (9.1.1)-(9.1.2) as follows

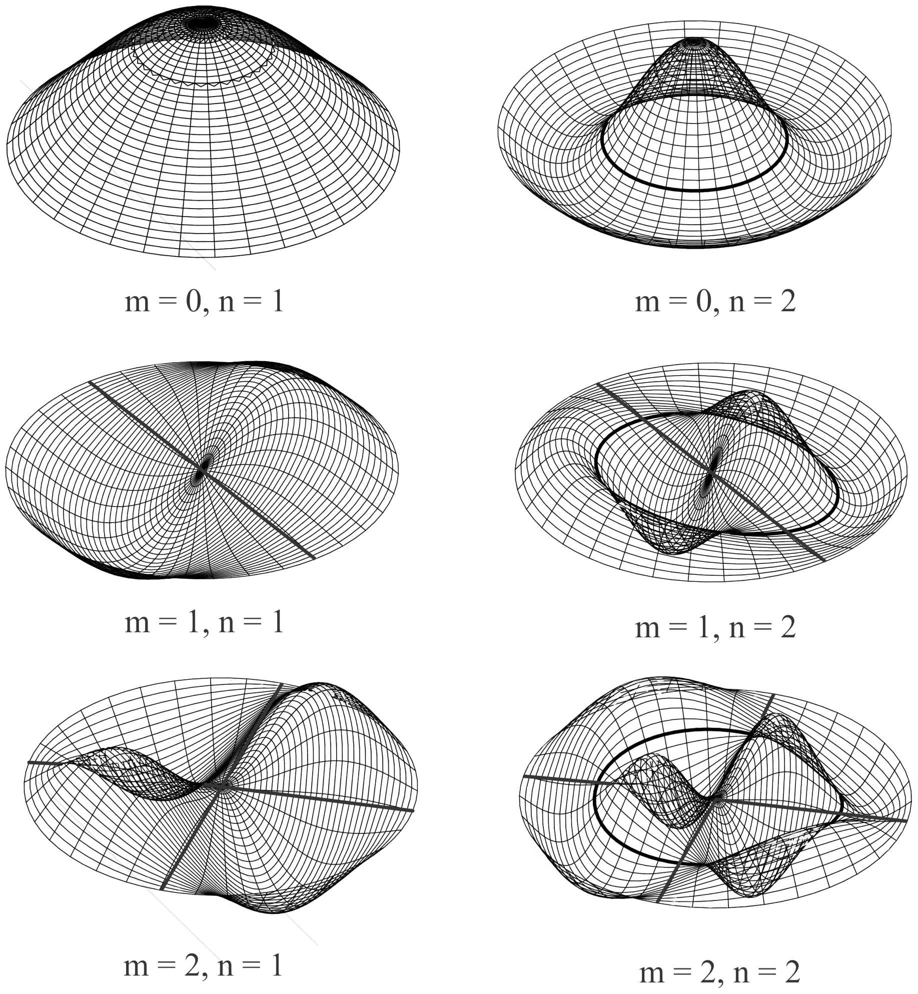


Figure 9.1.1: First few mode-shapes of a circular membrane with fixed outer boundary.

$$\left(\int\limits_{\Omega} \nabla\mathbf{G}^{\mathrm{T}}\nabla\mathbf{G}\,d\Omega + \int\limits_{\partial\Omega} \kappa\mathbf{G}^{\mathrm{T}}\mathbf{G}\,d(\partial\Omega)\right)\zeta = \lambda\int\limits_{\Omega}\mathbf{G}^{\mathrm{T}}\mathbf{G}\,d\Omega\,\zeta. \tag{9.1.4}$$

In a compact matrix representation (9.1.4) is written as

$$(\mathcal{A}+\mathcal{R})\zeta = \Lambda\mathcal{M}\zeta. \tag{9.1.5}$$

The eigenvectors ζ and eigenvalues Λ come in pairs (ζ,Λ), and there are as many pairs $(\zeta_i,\Lambda_i)_{i=1}^{n_G}$ as there are basis functions $\mathbf{G}$. Then the solution ζ_i substitutes the z-coordinate of the control points $\mathbf{Q}$ and thus the approximation of the eigenfunction ϑ_h can be obtained by the linear combination of control points and basis functions.

9.2 Domain construction

The main interest of the considered implementation is to explore how the very coarse mesh and low-degree local triangles handle a complex shape of the solution.

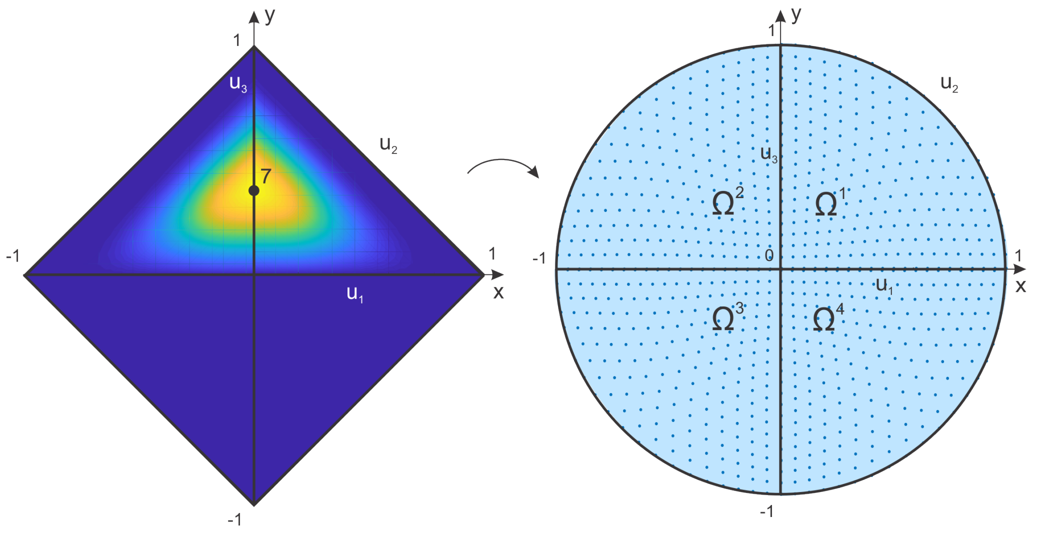


Figure 9.2.1: A mapping between triangular mesh and circular domain. As an example, a contour plot of seventh combined expo-rational basis function is shown in the figure on the left. Parameter lines of the circular domain are shown in the figure on the right .

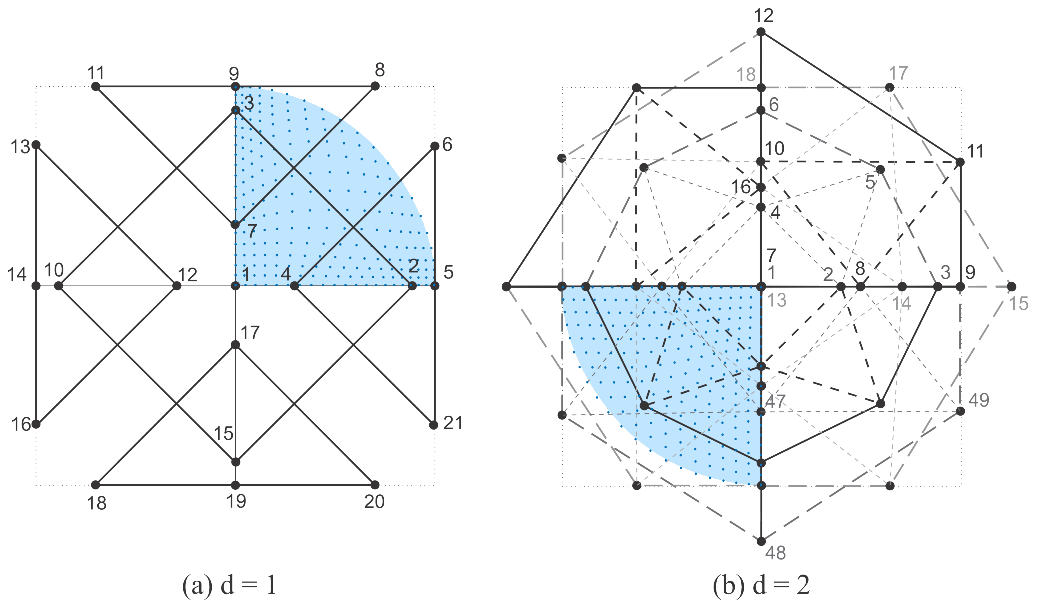


Figure 9.2.2: Local triangles and their control points of a circular domain constructed by four ERBS triangles. (a) Local triangles of first degree. (b) Local triangles of second degree.

A set of combined expo-rational basis functions for one element was defined in Section 4.1. Each basis function G_i, $i = 1, ..., n_G$, is continuous, piecewise, and has its support on a set of corresponding elements.

Let us divide the entire domain symmetrically into four elements. One element is represented as an ERBS triangle. A set of four elements is a triangular mesh. A mapping, obtained by a linear combination of expo-rational basis functions and local triangles, gives us the real domain. This mapping is shown in Figure 9.2.1. The set of control points of local triangles is called *a control net*.

A control net based on ERBS triangles has a layered structure in our example. An upper layer consists of four local triangles, which are connected by a central point. A bottom layer consists of triangles, connected by two, which construct the outer boundary. To avoid discontinuity in the domain, we merge matching points. The number of coefficients n varies depending on the point configuration. A number of basis functions on the entire domain corresponds to a number of the control points.

Figure 9.2.2 shows two examples of a control net for a mesh, which approximate a circular domain Ω. Figure 9.2.2(a) demonstrates local triangles of degree 1, Figure 9.2.2(b) shows local triangles of degree 2. The outer boundary for both cases is approximated by L^2-projection and blending splines. One can see that in Figure 9.2.2(b) parameters are distributed more uniformly, compared to Figure 9.2.2(a). By flexibility of ERBS triangle construction, we can construct specific parameter distribution, taking into account derivatives of the target surface.

The point matrix and connectivity matrix for the construction, shown in Figure 9.2.2(a), are given by

$$\mathbf{Q} = \left[\begin{matrix} 0 & a & 0 & 1-\pi/4 & a & a & 0 & \pi/4 & 0 & -a & -\pi/4 \\ 0 & 0 & a & 0 & 0 & \pi/4 & 1-\pi/4 & a & a & 0 & a \end{matrix}\right. \cdots$$

$$\cdots \left.\begin{matrix} -1+\pi/4 & -a & -a & 0 & -a & 0 & -\pi/4 & 0 & \pi/4 & a \\ 0 & \pi/4 & 0 & -a & -\pi/4 & -1+\pi/4 & -a & -a & -a & -\pi/4 \end{matrix}\right],$$

where a is the membrane radius.

$$\mathtt{T} = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 2 & 3 & 10 & 15 \\ 3 & 10 & 15 & 2 \\ 4 & 7 & 2 & 17 \\ 5 & 9 & 14 & 19 \\ 6 & 11 & 16 & 20 \\ 7 & 12 & 17 & 4 \\ 8 & 13 & 18 & 21 \\ 9 & 14 & 19 & 5 \end{bmatrix}.$$

When the connectivity matrix $\mathtt{T}$ is established for the mesh, one can fill in the global matrices $\mathcal{A}$, $\mathcal{M}$ and $\mathcal{R}$ using the local-to-global mapping.

For example, the global matrix $\mathcal{A}$ has $n_G \times n_G$ elements, where n_G is the number of control points $\mathbf{Q}$, while the element matrix $\mathcal{A}^e$ of the element Ω^e has $n_G^K \times n_G^K$ elements,

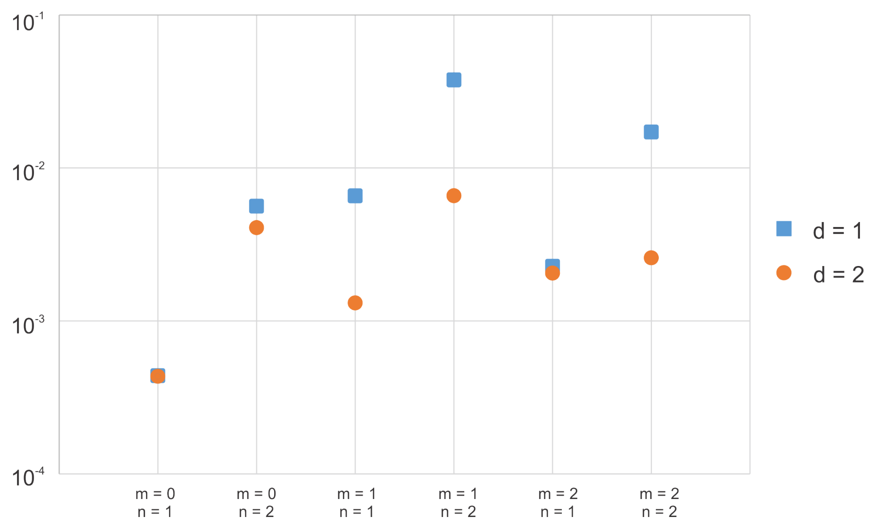


Figure 9.2.3: Comparison of L^2-error for different mode shapes and their FEM approximation by using two types of local triangles: of the first ($d = 1$) and second ($d = 2$) degree.

where $n_G^K = 3\begin{pmatrix} d+2 \\ 2 \end{pmatrix}$, d is the degree of local triangles. The local-to-global mapping appears from the connectivity matrix $\mathtt{T}$, whose e^{th} column provides a set of indexes where the element matrix will be placed relative to the global matrix. This way, the global matrix breaks up into sums of elemental contributions $\mathcal{A}^e$, $e = 1, ..., m$, where m is the number of triangular elements constituting the mesh.

The terms $\mathcal{A}^e$ are added to the appropriate locations in $\mathcal{A}$ in accordance with connectivity matrix $\mathtt{T}^e$. The same approach is valid for the matrices $\mathcal{M}$ and $\mathcal{R}$. The stiffness, mass and boundary matrices constitute the equation (9.1.5). Since our finite elements are curvilinear, coordinate transformation, considered in Section 4.3, is involved.

9.3 Results

The model problem (9.1.1) with Dirichlet boundary conditions (9.1.2) is solved employing the ERBS triangles as finite elements. A few mode shapes are found on the circular domain Ω, which is constructed as shown in Figure 9.2.2. Approximations are compared with the exact solution, found by the formulas (9.1.3). Comparison of L^2 errors for cases $d = 1$ and $d = 2$ is presented in Figure 9.2.3.

Figure 9.3.1 demonstrates solutions on meshes constructed by using ERBS triangles which are blended both first ($d = 1$) and second ($d = 2$) degree local Bézier triangles (dotted lines). Comparing the results visually with the exact solution in Figure 9.1.1 one can see that simple shapes of the solution can be handled by the first degree of local triangles at the same accuracy level as for the second degree local triangles, for example for modes $(0, 1)$, $(0, 2)$, $(2, 1)$. On the other hand, the appearance of the nodal circles together with nodal diameters immediately implies incrementation of the degree of local triangles. For example, the mode $(1, 2)$ has some irregularities on the local degree $d = 1$, comparing with degree 2. This can be explained as the shape of this mode is too complex for such low degree of local triangles.

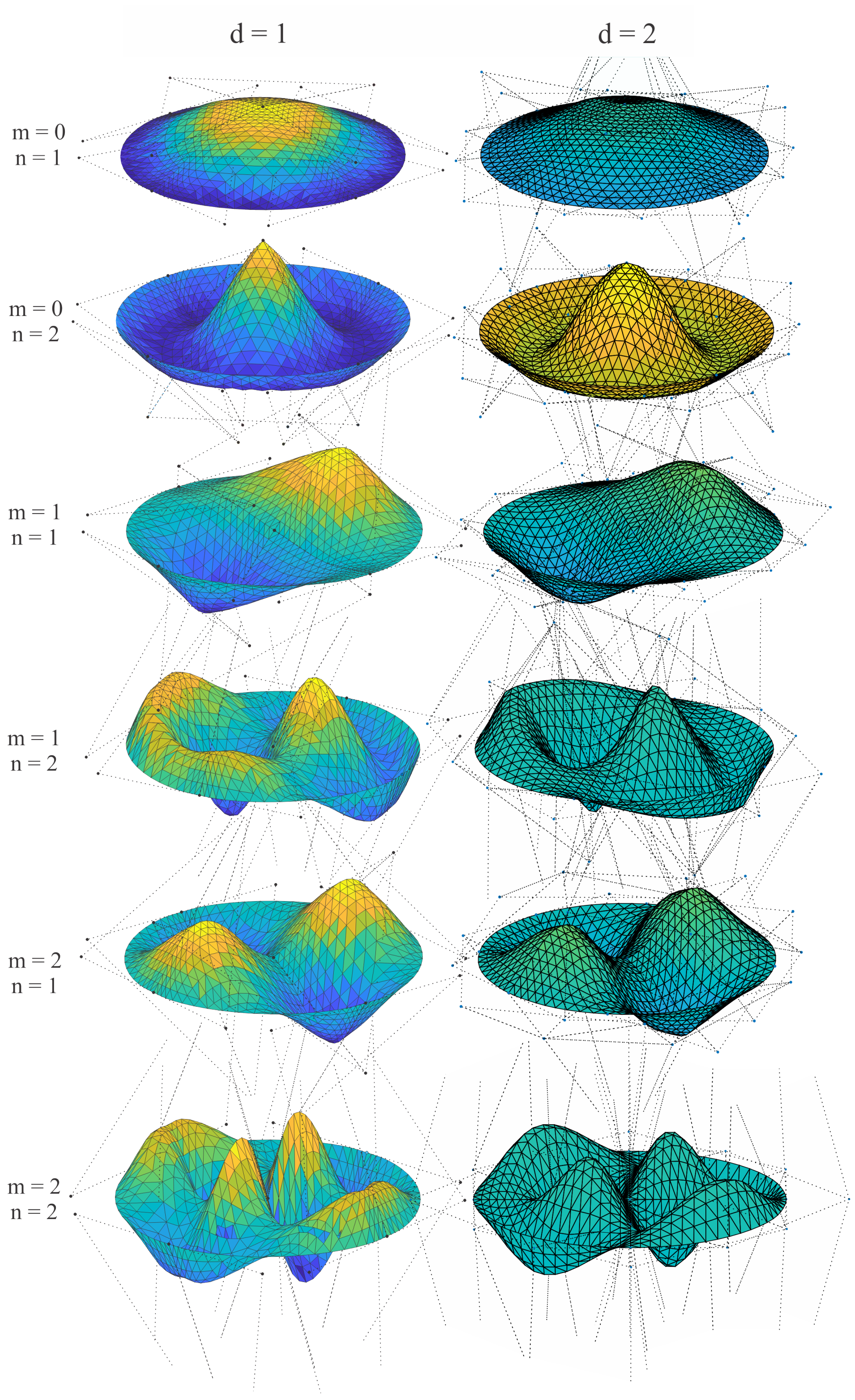


Figure 9.3.1: First few mode shapes, obtained by FEM utilizing ERBS triangles as elements. Two types of local triangles are presented: Bézier triangles of the first degree (left hand side), and of the second degree (right hand side). Local triangles are shown by points and dotted lines.

Increasing the degree of local triangles, one can provide many different approximations of the initial surface, which satisfy the required intrinsic properties of its geometry. Blending splines allow for accurate approximation of the boundary while keeping a coarse discretization of the domain. Flexible smooth domains can be constructed on a base of a few triangular elements. The overlapping of local triangles allows us to provide a flexible handling of the surface while preserving the smoothness of the initial domain, also over the nodes and edges.

Chapter 10

Conclusion

The main result of the current study is an introduction of the blending spline geometry into the finite element context. We developed a tool for solving partial differential equations in isogeometric manner, i.e. both domain and solution are represented in terms of the same basis functions. This tool is a set of independent computational routines, the combination of which provides a solution to the physical problem. In this work we aim to develop a universal framework which accommodates solving typical boundary value problems having a weak formulation as a variational problem. The domains are allowed to be triangulated or represented as tensor product surfaces. Both manual initialization of the domain and projective methods are supported. The boundary conditions are generalized and can be established in a mixed way on disjointed parts of the boundary. Hence, for the purposes above, we focused our attention on the algorithms, generalization of the methods and representation of the structures.

The blending spline type construction implies an interpolation of the local geometry, in particular, Bézier curves, surfaces or triangles. To be able to utilize this type of spline in the isogeometric context, we introduced so-called combined expo-rational basis, which combines both local and global bases. Thus, the blending surface construction is represented as a linear combination of control points and basis functions while being an interpolation of the local geometry. The blending type spline construction conserves the smoothness between elements and the minimal support of basis functions. These main properties allow us to construct both rectangular and triangular elements based on the expo-rational basis satisfying the standard concept of handling these types of elements.

A method for convertibility of the blending spline constructions to other types of splines was provided. By this method, we obtain a link between surface representations. A decomposition of the combined expo-rational basis onto the B-spline basis yields an extraction operator, which allows us to convert the B-spline control points to Bézier local geometry and vice versa. The extraction operator maps both bases and control points one to another one over the element. The conversion between B-spline and blending surfaces is intended to be useful in modeling in both IGA or CAGD contexts.

We presented a special form of basis representation in an array, which composes the local element representation and the representation on the entire domain. The use of basis symmetry leads to the simplification of the basis evaluation and it provides the possibility to parallelize the evaluation process. Moreover, one can simplify the process of assembling the finite element matrices. In the case of expo-rational blending splines, it leads to a special block-tridiagonal matrix form, regardless of the degree of the local geometry.

Numerical experiments, considered in this research, cover several important features that can be used in industry:

1. The possibility to utilize standard FEM algorithms and routines for ERBS finite elements.

2. The conversion between B-spline (and similar, e.g. NURBS) and ERBS representation can be directly utilized as a tool for domain reconstruction in the case when an initial surface is constructed in the external software.

3. Approximation of non-smooth surfaces can be used for construction of sharped domains, which can be easily modified via modifying the local geometry.

4. Adaptive mesh refinement can be implemented by adjusting knot vectors with local affect to the complete surface.

5. Complex domains can be constructed involving knot multiplicity for tensor product surfaces and flexible arrangement of local triangles for triangulated domains.

A number of possible applications may provide specific advantages when using ERBS finite elements. Their use in engineering problems such as linear elasticity, turbulent flows, high frequency waves, and similar, is expected to provide a solution representation, which is supposed to utilize intrinsic properties of the local geometry due to the interpolatory property of the blending surfaces.

Remarks and future work

A base for creating a framework for solving PDEs with ERBS finite elements as a main tool is presented in the thesis. The future work involves research topics to complete this development.

1. The numerical experiments were obtained by using MATLAB®. This tool has several advantages, for example, symbolic computations and piecewise functions were used. MATLAB® facilitates matrix computations. However, the implementation can be optimized by using more advanced level coding language. In the thesis we provided a general approach to the structure of the algorithms, which can be used for specific and universal custom applications. Computational efficiency of the considered algorithms is a broad area for future research.

2. We basically focused on the algorithmic component of the finite element method and its applications. The developed tool consists of several routines, combination of which provides opportunities to solve many standard PDE problems employing not only ERBS finite elements, but any smooth basis.

3. Some error estimates were provided and illustrated by computing the approximation convergence rate in the practical example. More thorough analysis, such as basis stability, needs to be included in future work.

4. A PDE solver based on ERBS finite elements is planned to be applied to problems with extreme conditions including holes, corners and singularities. Both domain construction and solution approximation need to be investigated. It is of interest

to examine the construction of the complex domains for both tensor product surfaces and triangulated meshes. Tensor product surfaces require knot multiplicity to provide holes in the domain, while construction of triangulated domains implies development of algorithms for generation of local triangles. Due to these restrictions, approximation of the PDE solution might lead to unexpected errors.

5. An important area for future research is investigating the use of expo-rational finite elements and the ERBS extraction operator in mesh refinement. There are various approaches to improving an accuracy of the solution. We plan to focus on the adjusting of the grid density. This can be achieved by two basic procedures: knot insertion and local geometry displacement.

6. It would be of interest to examine non-symmetric expo-rational basis functions in FEM application. The use of this type of function in blending is similar to some kind of weighted splines, as NURBS.

7. In this research we noticed some mesh generation problems. Blending surfaces support flexible domain handling, smooth boundary is obtained while the mesh itself is very coarse. However, the automated mesh generation is not provided. The following issues need to be solved:

   (a) Advanced mapping for tensor product domain construction. Projective methods allow for approximation of the parameterized functions. Alternative mappings, such as boundary fitting, should be implemented.

   (b) Automatic generation for triangulated domains. Two steps of triangulated domain building are involved: initial coarse triangulation and approximation of the smooth boundary. Even complex shape domains can be approximated by a few ERBS triangular elements, but an optimal position of these elements and corresponding local triangles should be algorithmically supported.

8. The approach considered in this work can be extended to volumes. Volume representation with expo-rational basis covers many additional topics and its application in a finite element context is considered as future work.

Bibliography

[1] B. Bang, L.T. Dechevsky, A. Lakså, and P. Zanaty. Blending Functions for Hermite Interpolation by Beta-Function B-Splines on Triangulations. *In Large-Scale Scientific Computing 2011*, 7116 of Lecture Notes in Computer Science:393–401, 2012.

[2] P. Bézier. Définition numérique des courbes et surfaces I. *Automatisme*, XI:625–632, 1966.

[3] P. Bézier. Définition numérique des courbes et surfaces II. *Automatisme*, XII:17–21, 1967.

[4] K. Bittner and H.G. Brachtendorf. Fast algorithms for adaptive free-knot spline approximation using non-uniform biorthogonal spline wavelets. *SIAM J. Scient. Computing*, 37(2):283–304, 2015.

[5] W. Boehm. Inserting new knots into B-spline curves. *Computer Aided Design*, 12(4):199–201, 1980.

[6] M.J. Borden, M.A. Scott, J.A. Evans, and T.R.J. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87:15–47, 2010.

[7] M.J. Borden, M.A. Scott, C.V. Verhoosel, T.W. Sederberg, and T.R.J. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 00:1–40, 2010.

[8] B. Bornemann and F. Cirak. A subdivision-based implementation of the hierarchical B-spline finite element method. *Computer Methods in Applied Mechanics and Engineering*, 253:584–598, 2013.

[9] J.H. Bramble and S.R. Hilbert. Estimation of linear functionals on Sobolev spaces with application to Fourier transforms and spline interpolation. *SIAM Journal on Numerical Analysis*, 7(1):112–124, 1970.

[10] S.C. Brenner and L.R. Scott. *The Mathematical Theory of Finite Element Methods*, volume 15. Springer, New York, NY, 3rd edition, 2008.

[11] I.G. Bubnov. *Review of S.P. Timoshenko's work, "On the Stability of Elastic Systems"*. Collected Works, Leningrad, 1956, in Russian.

[12] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Math.Comp.*, 22:745–762, 1968.

[13] P.G. Ciarlet. The finite element method for elliptic problems. *Classics in Applied Mathematics*, 40:1–511, 2002.

[14] P.G. Ciarlet and P.-A. Raviart. General Lagrange and Hermite Interpolation in R^n with applications to Finite Element Methods. *Archive for Rational Mechanics and Analysis*, 46(3):177–199, 1974.

[15] Cirak, M. Ortiz and P. Schröder. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. *International Journal for Numerical Methods in Engineering*, 47(12):2039–2072, 2000.

[16] E. Cohen, T. Lyche, and R. Riesenfeld. Discrete B-splines and subdivision techniques in Computer-Aided Geometric Design and computer graphics. *Computer Graphics and Image Processing*, 14(2):87–111, 1980.

[17] J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA.* Wiley, Chicester, U.K., 2009.

[18] W. Dahmen, C.A. Micchelli, and H.-P. Seidel. Blossoming begets B-splines built better by B-patches. *Mathematics of Computation*, 59(199):97–115, 1992.

[19] R. Dalmo. Local refinement of ERBS curves. *In V. Pasheva and G. Venkov, editors, 39th International conference applications of mathematics in engineering and economics*, 1570 of AIP Conference Proceedings:204–211, 2013.

[20] R. Dalmo, J. Bratlie, B. Bang, and A. Lakså. Smooth spline blending surface approximation over a triangulated irregular network. *International Journal of Applied Mathematics*, 27(1):109–119, 2014.

[21] C. de Boor. On calculating with B-splines. *J. Approx. Theory*, 6:50–62, 1972.

[22] C de Boor. *Splines as linear combinations of B-splines. A survey.* Technical summary report #1667, University of Wisconsin-Madison, Mathematics Research Center, 1976.

[23] C. de Boor. Package for calculating with B-splines. *SIAM J. Numer. Anal.*, 14:441–472, 1977.

[24] C. de Boor. *A Practical Guide to Splines.* Springer-Verlag, New York, 1978.

[25] P. de Casteljau. *Courbes et surfaces a pôles.* André Citroën Automobiles SA, Paris, 1959.

[26] P. de Casteljau. *Shape mathematics and CAD.* Kogan Page, 1986.

[27] L.T. Dechevsky. *Expo-Rational B-Splines.* Communication at the Sixth International Conference on Mathematical Methods for Curves and Surfaces, Tromsø, Norway, 2004.

[28] L.T. Dechevsky, B. Bang, and A. Lakså. Generalized Expo-Rational B-Splines. *International Journal of Pure and Applied Mathematics*, 57(6):833–872, 2009.

[29] L.T. Dechevsky, A. Lakså, and B. Bang. Expo-Rational B-Splines. *International Journal of Pure and Applied Mathematics*, 27(3):319–367, 2006.

[30] L.T. Dechevsky and P. Zanaty. First instances of univariate and tensor-product multivariate generalized expo-rational finite elements. *In: Applications of Mathematics and Engineering and Economics 2011*, 1410:128–138, 2011.

[31] L.T. Dechevsky and P. Zanaty. Beta-function B-spline Smoothing on Triangulations. *In: Three-Dimensional Image Processing (3DIP) and Applications 2013, volume 8650 of Proceedings of SPIE, page 865004. International Society for Optics and Photonics, SPIE*, 2013.

[32] L.T. Dechevsky and P. Zanaty. Smooth GERBS, orthogonal systems and energy minimization. *In: Applications of Mathematics in Engineering and Economics 2013*, 1570:135–162, 2013.

[33] L.T. Dechevsky, P. Zanaty, A. Lakså, and B. Bang. First instances of generalized expo-rational finite elements on triangulations. *In: Applications of Mathematics and Engineering and Economics 2011*, 1410:49–61, 2011.

[34] E.H. Doha, A.H. Bhrawy, and M.A. Saker. On generalized Jacobi-Bernstein basis transformation: Application of multi-degree reduction of Bézier curves and surfaces. *Journal of Computing and Information Science in Engineering*, 14, 2014.

[35] T. Dokken, T. Lyche, and K.F. Pettersen. Polynomial splines over locally refined box-partitions. *Computer Aided Geometric Design*, 30(21):331–356, 2013.

[36] T. Dokken, V. Skytt, and O. Barrowclough. Trivariate spline representations for computer aided design and additive manufacturing. *Computer and Mathematics with Applications*, 2018.

[37] T.A. Driscoll and L.N. Trefethen. *Schwarz-Christoffel Mapping.* Number 8 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, New York, 2002.

[38] D.W. Hahn and M.N. Özisik. *Heat conduction.* John Wiley & Sons, Inc., Hoboken, New Jersey, 3rd edition, 2012.

[39] H. Edelsbrunner. *Geometry and Topology for Mesh Generation.* Number 6 in Cambridge Monographs on Applied and Computational Mathematics. Cambridge University Press, New York, 2001.

[40] J.A. Evans, M.A. Scott, X. Li, and D.C. Thomas. Hierarchical T-splines: analysis-suitability, Bézier extraction, and application as an adaptive basis for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 284:1–20, 2015.

[41] L.C. Evans. *Partial Differential Equations.* American Mathematical Society, 2nd edition, 2010.

[42] G. Farin. Triangular Bernstein-Bézier patches. 3(2):83–127, 1986.

[43] G. Farin. *Curves and surfaces for CAGD: a practical guide.* Computer Science and Scientific Computing. Academic press, San Diego, CA, USA, 4th edition, 1997.

[44] J. Ferguson. Multivariable curve interpolation. *Journal of the Association for computing Machinery*, 11(2):221–228, 1964.

[45] M.S. Floater. Meshless parameterization and B-Spline surface approximation. *In: Cipolla R., Martin R. (eds) The Mathematics of Surfaces IX*, 2000.

[46] M.S. Floater. Generalized barycentric coordinates and applications. *Acta Numerica*, 24:161–214, 2015.

[47] D.R. Forsey and Bartels R.H. Hierarchical B-spline refinement. *Computer Graphics*, 22(4):205–212, 1988.

[48] J.H. Friedman. Multivariate adaptive regression splines. *The Annals of Statistics*, 19:1–67, 1990.

[49] B.G. Galerkin. Rods and plates. application of series in some problems of equilibriums of rods and plates. *Vestnik Inzhinerov*, 1(19):897–908, 1915, in Russian.

[50] E.I. Grigoluk. On the Bubnov-Galerkin method. *Investigation of Theory of Plates and Shells*, XI:3–41, 1975, in Russian.

[51] I. Guttman. *Introductory Engineering Statistics.* John Wiley & Sons, Inc., 1965.

[52] P. Hagedorn and A. DasGupta. *Vibrations and waves in continuous mechanical systems.* Wiley, 1st edition, 2007.

[53] I. Harari and T.J.R. Hughes. Finite element method for the Helmholtz equation in an exterior domain: Model problems. *Comp. Meth. Appl. Mech. Eng.*, 87:59—96, 1991.

[54] K. Höllig. Stability of the B-spline basis via knot insertion. *Computer Aided Geometric Design*, 17:447–450, 2000.

[55] K. Höllig, J. Hörner, and A. Hoffacker. Finite Element Analysis with B-Splines: Weighted and Isogeometric Methods. *In: Boissonnat J.-D. et al. (eds.) Curves and Surfaces 2011. Lecture Notes in Computer Science*, 6920:330–350, 2012.

[56] T.J.R. Hughes. The finite element method: linear static and dynamic finite element analysis. *Computer-aided civil and infrastructure engineering*, 4(3):245–246, 1989.

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

[58] I. Babuška, U. Banerjee. Stable Generalized Finite Element Method (SGFEM). *Computer Methods in Applied Mechanics and Engineering*, 201-204(0):91–111, 2012.

[59] G. Jekabsons. *ARESLab: Adaptive Regression Splines toolbox for Matlab/Octave.* Institute of Applied Computer Systems, Riga Technical University, Riga, Latvia, 2009.

[60] Y. Jia, Y. Zhang, G. Xu, X. Zhuang, and T. Rabczuk. Reproducing kernel triangular B-spline-based FEM for solving PDEs. *Computer Methods in Applied Mechanics and Engineering*, 267:342–358, 2013.

[61] J.M. Melenk, I. Babuška. The partition of unity finite element method: Basic theory and applications. *Comput. Methods Appl. Mech. Engrg.*, 139:289–314, 1996.

[62] K. Höllig. *Finite Element Methods with B-Splines.* Society for Industrial and Applied Mathematics, Philadelphia, 2003.

[63] E. Kılıç and P. Stanica. The inverse of banded matrices. *Journal of Computational and Applied Mathematics*, 237(1):126–135, 2013.

[64] J. Koko. Vectorized Matlab codes for linear two-dimensional elasticity. *Scientific programming*, 15:157–172, 2007.

[65] T. Kravetc. Finite element method application of ERBS triangles. *NIK: Norsk Informatikkonferanse 2019. ISSN 1892-0721*, 2019.

[66] T. Kravetc, B. Bang, and R. Dalmo. Regression analysis using a blending type spline construction. *Mathematical Methods for Curves and Surfaces: 9th International Conference, MMCS 2016 Tønsberg, Norway*, pages 145–161, 2017.

[67] A.R. Kristoffersen, L.T. Dechevsky, A. Lakså, and B. Bang. Comparison between polynomial, Euler beta-function and expo-rational B-spline bases. *AIP Conference Proceedings*, 1410(1):98–110, 2011.

[68] M.-J. Lai and L.L. Schumaker. *Spline Functions on Triangulations*, volume volume 110 of Encyclopedia of Mathematics and Its Applications. Cambrige University Press, Cambridge, UK, 2007.

[69] A. Lakså. *Basic properties of Expo-Rational B-splines and practical use in Computer Aided Design.* University of Oslo, PhD thesis, 2007.

[70] A. Lakså, B. Bang, and L.T. Dechevsky. Exploring expo-rational B-splines for curves and surfaces. *In: M. Dæhlen, K. Mørken, L.L. Schumaker, editors, Mathematical methods for Curves and Surfaces*, pages 253–262, 2005.

[71] A. Lakså, B. Bang, and L.T. Dechevsky. Exploring expo-rational B-Splines for curves and surfaces. *Mathematical Methods for Curves and Surfaces*, pages 253–262, 2005.

[72] H.P. Langtangen and A. Logg. *Solving PDEs in Python. The FEniCS Tutorial I*, volume 3. Simula SpringerBriefs on Computing. Springer International Publishing, 2016.

[73] M.G. Larson and F. Bengzon. *The Finite Element Method: Theory, Implementation, and Practice.* Springer, 2010.

[74] T. Lyche and K. Mørken. *Spline methods.* Lecture notes at University of Oslo (Draft), 2018.

[75] G.I. Marchuk and V.I. Agoshkov. *Introduction in Projective Grid Methods.* Nauka, Moscow, 1981, in Russian.

[76] L. Meirovitch. *Principles and Techniques of Vibrations.* Prentice-Hall, Inc., Upper Saddle River, New Jersey, 1997.

[77] G. Meurant. A review on the inverse of symmetric tridiagonal and block tridiagonal matrices. *SIAM Journal on Matrix Analysis and Applications*, 13(3):707–728, 1992.

[78] L. Mu, J. Wang, X. Ye, and S. Zhao. A Numerical Study on the Weak Galerkin Method for the Helmholtz Equation with Large Wave Numbers. *arXiv e-prints*, page arXiv:1111.0671, 2011.

[79] G.P. Nikishkov. *Introduction to the Finite Element Method.* Lecture Notes, University of Aizu, Aizu-Wakamatsu 965-8580, Japan, 2004.

[80] P.M. Prenter. *Splines and variational methods.* Wiley New York, 1975.

[81] J.W.S. Rayleigh. *The Theory of Sound.* London, 2nd edition, 1894 and 1896.

[82] W. Ritz. Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik. *J. Reine Angew. Math.*, 135:1–61, 1908.

[83] M.-C. Rivara. New mathematical tools and techniques for the refinement and/or improvement of unstructured triangulations. *in Proceedings: 5th International Meshing Roundtable*, pages 77–86, 1996.

[84] J. Ruppert. A Delaunay refinement algorithm for quality two-dimensional mesh generation. *Journal of Algorithms*, 18:548–585, 1995.

[85] D. Schillinger, L. Dede, M.A. Scott, J.A. Evans, M.J. Borden, E. Rank, and T.R.J. Hughes. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249-250:116–150, 2012.

[86] D. Schillinger, P.K. Ruthala, and L.H. Nguyen. Lagrange extraction and projection for NURBS basis functions: A direct link between isogeometric and standard nodal finite element formulations. *International Journal for Numerical Methods in Engineering*, 108:515–534, 2016.

[87] I.J. Schoenberg. *Contribution to the problem of approximation of equidistant data by analytic functions.* Ballistic Research Laboratories, University of Pennsylvania, Aberdeen Proving Ground, USA, 1946.

[88] L.L. Schumaker. *Spline functions.* Cambrige University Press, Cambridge, UK, 3rd edition, 2007.

[89] Weather service yr.no. Norwegian meteorological institute and norwegian broadcasting corporation, 2007 - 2017, dates 01.12.2016-03.01.2017.

[90] V.V. Shaidurov. *Multigrid Methods of Finite Elements.* Nauka, Moscow, 1989, in Russian.

[91] V.V. Showalter. *Hilbert Space Methods for Partial Differential Equations.* Dover Publications, 1st edition, 2010.

[92] G. Strang and G.J. Fix. *An analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, NJ, 1973.

[93] A.H. Stroud and D. Secrest. *Gaussian Quadrature Formulas.* Prentice-Hall, Englewood Cliffs, New Jersey, 1992.

[94] B. Szabó and I. Babuška. *Finite Element Analysis.* Wiley, New York, 1991.

[95] W. Härdle. *Applied Nonparametric Regression.* Cambridge University Press, revised edition, 1992.

[96] G. Xu, T.-H. Kwok, and C.C.L. Wang. Isogeometric computation reuse method for complex objects with topology-consistent volumetric parameterization. *Computer-Aided Design*, 91:1–13, 2017.

[97] G. Xu, B. Li, L. Shu, L. Chen, J. Xu, and T Khajah. Efficient r-adaptive isogeometric analysis with Winslow's mapping and monitor function approach. *Journal of Computational and Applied Mathematics*, 351:186–197, 2019.

[98] P. Zanaty. Finite element methods based on a generalized expo-rational B-splines with harmonic polynomial coefficients. *International Journal of Applied Mathematics*, 26(3):379–390, 2013.

[99] C.O. Zienkiewicz. Achievements and some unsolved problems of finite element method. *International Journal for Numerical Methods in Engineering - Int J Numer Method Eng*, 47:9–28, 01 2000.

[100] O.C. Zienkiewicz, R.L. Taylor, and J.Z. Zhu. *The Finite Element Method: Its Basis and Fundamentals.* Elsevier Butterworth-Heinemann Linacre House, Jordan Hill, Oxford, 6th edition, 2005.

List of Figures