TECHNISCHE UNIVERSITÄT DRESDEN

FAKULTÄT MATHEMATIK

Dissertation

for obtaining the academic degree of
Doctor rerum naturalium
(Dr. rer. nat.)

Spectral Discretizations on Surfaces: High-Order Methods for Integration and PDE Solvers

Gentian Zavalani

Supervisor: Prof. Dr. Oliver Sander
Advisor: Prof. Dr. Michael Hecht

Dresden, August 12, 2025

Acknowledgements

There are many people to whom I am deeply grateful. First and foremost, I thank my supervisor, Oliver Sander, and my advisor, Michael Hecht, for their guidance, support, and invaluable insights. Michael — thank you for introducing me to the fascinating world of multivariate approximation and for always being there with your advice, even after I left the institute. Oliver — thank you so much for your generous support during the writing of this thesis, for your constant encouragement, and for the energy you bring to the group.

I would like to thank my friends and fellow PhD students Damar, Janina, Azita, Juan, and Phil, whose presence, insight, and good humor were a constant source of motivation and joy throughout this journey. A special thanks goes to Juan for three years of not only valuable mathematical discussions but also delightful non-mathematical conversations.

This thesis was carried out during my time at CASUS. I am thankful for the opportunity to pursue my research at the institute and for the resources and professional setting that supported this work. After leaving CASUS, I joined the Institute of Numerical Mathematics at TU Dresden. I am grateful to Klaus, Lukas, Simon, Hanne, Anne-Sophie, and Ann-Sophie for their warm welcome and for contributing to a pleasant working environment. I also wish to acknowledge Balázs Kovács for providing the MATLAB implementation of the arbitrary Lagrangian–Eulerian (ALE) map algorithms.

I express my deepest gratitude to Allah s.w.t for giving me the opportunity to be part of this wonderful community and collaborate with remarkable individuals from whom I have learned a lot. I am especially grateful to my parents for their unwavering belief in me and their constant encouragement of my mathematical pursuits. My heartfelt thanks go to my wife, my son Jasin, and my two little nieces, Hojza and Grisela, whose love, joy, and cheerful presence have been a continuous source of motivation and happiness.

This research was partially funded by the Center of Advanced Systems Understanding (CASUS), which is financed by Germany's Federal Ministry of Education and Research (BMBF) and by the Saxon Ministry for Science, Culture, and Tourism (SMWK) with tax funds on the basis of the budget approved by the Saxon State Parliament.

Abstract

Accurate and efficient computation on curved surfaces remains a central challenge in numerical analysis. This dissertation addresses this problem by presenting high-order methods for numerical integration and the discretization of partial differential equations on such surfaces, combining theoretical analysis with the development of efficient algorithms.

We introduce a novel methodology for constructing high-order volume elements (HOVE) as a building block for spectral discretizations on embedded surfaces. The central idea is a reparametrization strategy, referred to as *square-squeezing*, which maps an initial flat triangulation of the surface to a tensor-product domain through a homeomorphic multilinear transformation. This transformation enables the use of tensor-product Chebyshev–Lobatto grids within each element, allowing stable high-order interpolation of both the surface geometry and functions defined on the surface. Combined with spectral differentiation for computing map derivatives and approximating the metric tensor, the resulting framework provides an accurate and computationally efficient approach for high-order surface discretizations. The method is implemented in the SURFGEOPY software package to support further applications and developments.

In addition, we develop a theoretical framework that explains the superior convergence of even-degree polynomials in piecewise polynomial surface parametrizations with Lagrange elements, proves this behavior, and illustrates its application to numerical computations of surface integrals. The whole approach rests on interpolation in equidistant nodes on simplices, making it sensitive to Runge's phenomenon and becoming unstable for high orders. The proposed HOVE method eliminates these stability and accuracy issues while maintaining computational efficiency.

In the final part of the thesis, we present a domain decomposition scheme that employs high-order spectral approximations on each element, coupled with a direct hierarchical solver, forming what is known as the hierarchical Poincaré–Steklov (HPS) scheme. For general triangulated surfaces, we develop two strategies: (i) a hypercube-to-simplex reparametrization, and (ii) a rhombus-based quadrilateralization technique. These strategies make it possible for triangular elements to match the efficiency of quadrilateral elements, thereby extending high-order spectral domain decomposition methods to unstructured, triangulated surface meshes without loss of accuracy or computational performance. While originally designed for stationary, linear elliptic PDEs, the HPS framework is extended to time-dependent systems and applied to a Turing-

type reaction–diffusion model on static surfaces, exploring how geometry, nonlinearities, and coupling influence pattern formation.

Finally, we extend the framework to evolving geometries by adapting the closest point projection method to incorporate time-dependent surface evolution. To mitigate mesh degeneration, particularly the formation of near-degenerate triangles, an arbitrary Lagrangian–Eulerian (ALE) mapping is applied during deformation. Two biologically inspired surface growth models are studied to analyze the effect of curvature and domain evolution on pattern formation, with implications for morphogenesis and material design.

These developments advance the state of high-order spectral methods for surface-based computation, providing both theoretical insight and practical tools for applications in computational geometry, physics, and biology.

Zusammenfassung

Eine genaue und effiziente Berechnung auf gekrümmten Flächen bleibt eine zentrale Herausforderung in der Numerischen Analysis. Diese Dissertation behandelt dieses Problem durch die Vorstellung hochgradiger Verfahren für die numerische Integration und die Diskretisierung partieller Differentialgleichungen auf solchen Flächen, wobei theoretische Analysen mit der Entwicklung effizienter Algorithmen kombiniert werden.

Wir stellen eine neuartige Methodik zur Konstruktion hochordentlicher Volumenelemente (HOVE) als Baustein für spektrale Diskretisierungen auf eingebetteten Oberflächen vor. Die zentrale Idee besteht in einer Reparametrisierungsstrategie, die als *Square-Squeezing* bezeichnet wird und eine anfängliche flache Triangulierung der Oberfläche mittels einer homöomorphen multilinearen Transformation auf eine Tensorprodukt-Domäne abbildet. Diese Transformation ermöglicht die Verwendung von Tensorprodukt-Chebyshev–Lobatto-Gittern innerhalb jeder Zelle und erlaubt dadurch eine stabile hochordentliche Interpolation sowohl der Oberflächengeometrie als auch von auf der Oberfläche definierten Funktionen. In Kombination mit spektraler Differentiation zur Berechnung der Abbildungsableitungen und zur Approximation des metrischen Tensors ergibt sich ein genauer und recheneffizienter Ansatz für hochordentliche Oberflächendiskretisierungen. Die Methode ist in der Softwarebibliothek SURFGEOPY implementiert, um weitere Anwendungen und Entwicklungen zu unterstützen.

Darüber hinaus entwickeln wir einen theoretischen Rahmen, der die überlegene Konvergenz geradzahliger Polynome in stückweise polynomialen Flächenparametrisierungen mit Lagrange-Elementen erklärt, dieses Verhalten beweist und seine Anwendung auf numerische Berechnungen von Flächenintegralen veranschaulicht. Der gesamte Ansatz basiert auf Interpolation in äquidistanten Knoten auf Simplizes, was ihn anfällig für das Runge-Phänomen macht und bei hohen Ordnungen instabil werden lässt. Die vorgeschlagene HOVE-Methode beseitigt diese Stabilitäts- und Genauigkeitsprobleme, während die rechnerische Effizienz beibehalten wird.

Im letzten Teil der Arbeit präsentieren wir ein Domänendekompositionsschema, das auf jedem Element hochgradige spektrale Approximationen verwendet und mit einem direkten hierarchischen Löser gekoppelt ist. Zwei Strategien ermöglichen hochgradige spektrale Operationen auf allgemeinen triangulierten Flächen: (i) eine Hyperwürfel-zu-Simplex-Umparametrisierung und (ii) eine rautenbasierte Quadrangulierungstechnik. Diese Ansätze ermöglichen es, dass Dreieckselemente eine mit Viereckselementen vergleichbare Effizienz erreichen und in hochgradige

spektrale Domänendekompositionsmethoden integriert werden können. Obwohl der HPS-Rahmen ursprünglich für stationäre, lineare elliptische partielle Differentialgleichungen entwickelt wurde, wird er auf zeitabhängige Systeme erweitert und auf ein Reaktions-Diffusions-Modell vom Turing-Typ auf statischen Flächen angewendet, um zu untersuchen, wie Geometrie, Nichtlinearitäten und Kopplung die Musterbildung beeinflussen.

Schließlich erweitern wir den Rahmen auf sich entwickelnde Geometrien, indem wir die Closest-Point-Projection-Methode anpassen, um zeitabhängige Flächenentwicklungen zu berücksichtigen. Um eine Netzdegradation, insbesondere die Bildung von Dreiecken mit nahezu verschwindenden Innenwinkeln, zu verhindern, wird während des Deformationsprozesses eine Arbitrary-Lagrangian–Eulerian-(ALE)-Abbildung eingesetzt. Zwei biologisch inspirierte Wachstumsmodelle für Flächen werden untersucht, um den Einfluss von Krümmung und Flächenentwicklung auf die Dynamik der Musterbildung zu analysieren, mit Anwendungen in der Morphogenese und im Materialdesign.

Diese Entwicklungen treiben den Stand der hochgradigen spektralen Methoden für flächenbasierte Berechnungen voran und liefern sowohl theoretische Erkenntnisse als auch praktische Werkzeuge für Anwendungen in der rechnergestützten Geometrie, Physik und Biologie.

Contents

Chapter 1

Introduction

The motivation for this work comes from surface-bound phenomena that occur across a wide range of applications, including biological pattern formation [76], electromagnetics [90], and fluid dynamics [114]. Modeling such phenomena often leads to partial differential equations (PDEs) posed on surfaces. These include both steady-state problems, such as the Laplace–Beltrami equation and Stokes-type equations, and time-dependent problems, such as diffusion, reaction–diffusion, and Navier–Stokes equations on surfaces. Consequently, the numerical solution of surface PDEs plays an essential role in the simulation and understanding of many surface phenomena.

To solve a surface PDE numerically, the first step is to discretize the geometry. In our setting, the surface is represented by a mesh, so the geometry is given as a triangulation, where each surface patch is locally parameterized over a simplex. Using this local parameterization, we define the metric tensor. Then, with some basic linear algebra, we can derive the main surface differential operators—such as the surface gradient, divergence, and the Laplace–Beltrami operator—and write down the surface integrals that appear in variational formulations. Surface integration itself plays a central role, forming a core component of methods including boundary integral formulations, the finite element method, surface finite element methods, and integral transform methods.

The main challenge in computing surface integrals and surface differential operators lies in the accurate approximation of the metric tensor. This difficulty arises because, for general surfaces, the parameterization maps are not known analytically but are instead provided in an algorithmic form. While the affine map from the reference element is known, the closest-point projection is in general not directly available to the user. It has a closed-form expression only for special geometries such as the sphere and the torus. However, using numerical algorithms, we can

project points from the reference surface to the smooth one.

To overcome this difficulty, one common approach is to replace the exact Jacobians of the surface mapping with those of a polynomial approximation obtained through interpolation. Accurate numerical integration requires high-order approximations of both the geometry and the integrand. A natural first choice is to use high-order piecewise polynomial interpolation, typically based on Lagrange polynomials defined on uniformly distributed points within each triangle. However, polynomial interpolation at equidistant nodes is known to become unstable at higher polynomial degrees due to Runge's phenomenon. As a result, when both the geometry and the integrand are approximated using equidistant nodes, the accuracy of the numerical integration can deteriorate.

Low-order integration schemes are relatively straightforward. For high-order methods, however, integration over curved (triangulated) surfaces involves more than simply applying cubature rules. Accurate results require high-order representations of both the surface geometry and the function being integrated. Therefore, the design of stable and accurate high-order numerical integration schemes on surfaces remains an active research problem.

To meet these requirements, we develop a methodology for constructing stable, high-order surface representations. This surface description is then used to derive high-order volume elements (HOVE) for integrating scalar- and vector-valued functions over regular embedded surfaces. Spectral differentiation techniques are employed to improve the computational efficiency of the volume element construction. In the following, we provide an overview of the results presented in this dissertation.

The dissertation begins with **Chapter 2**, which introduces the mathematical preliminaries and notation used throughout the text. This chapter is primarily expository, providing background material drawn from the literature, with some sections adapted and extended from [146].

Chapter 3 explores Chebyshev polynomials, a classical tool for approximating smooth functions in one, two, and three dimensions. While their properties in low-dimensional settings are well established, the chapter extends these ideas to higher dimensions. Starting from the one-dimensional theory, it develops a general framework for d-dimensional approximation, presents new convergence results, and discusses their relevance for numerical applications.

Chapter 4 lays the foundation for numerical methods on surfaces by examining the mathematical and geometric properties of d-dimensional regular (differentiable) surfaces embedded in Euclidean space. It provides the theoretical background needed for the developments in later chapters.

Chapter 5 brings together results from three studies that advance the development of higher-order numerical integration techniques for surfaces.

- First, we investigate the error behavior of numerical integration based on piecewise polynomial surface parametrizations using Lagrange elements. Our analysis reveals a dis-

tinct difference between even and odd polynomial degrees, with even-degree elements exhibiting superior convergence rates. We establish a theoretical explanation for this phenomenon, supported by a series of numerical experiments.

- Second, we develop a stable high-order volume element (HOVE) construction for integrating scalar- and vector-valued functions on regular embedded surfaces. The key ingredient is the *square-squeezing* transformation, which maps each flat triangular patch of the surface onto a structured cubical mesh in a homeomorphic, multilinear fashion, thereby enabling the application of Chebyshev–Lobatto interpolation for precise and efficient numerical integration.
- Third, we apply spectral differentiation techniques to compute partial derivatives of the cubical reparametrization maps. These derivatives are used to approximate the metric tensor on each element, significantly improving the precision and efficiency of surface integration.

Chapter 6 presents a domain decomposition framework based on spectral collocation discretization on each element. We couple the solutions across element interfaces using Dirichlet-to-Neumann operators in combination with a hierarchical direct solver, forming what is known as the hierarchical Poincaré–Steklov (HPS) scheme. The chapter contributes in three main directions:

- While the HPS scheme can in principle handle general geometries, earlier applications have focused mainly on quadrilateral surface meshes. To enable its use with both triangular and quadrilateral elements, we introduce two strategies:
  1. A *hypercube-to-simplex reparametrization* (square-squeezing) that maps flat triangulations to quadrilateral patches, allowing interpolation, quadrature, and differentiation to be performed on reference squares. Inspired by analytic transplantation techniques [64], we further compose this reparametrization with an analytic transformation, which in practice improves spectral convergence.
  2. A *rhombus-based quadrilateralization* strategy, which converts triangles into quadrilaterals while minimizing the number of elements. Compared to centroid subdivision approaches [51], this method provides a more efficient patchwise representation.
- We conduct a detailed numerical study of a generic Turing system, exploring how domain geometry and coupling mechanisms influence pattern formation on curved surfaces.
- Building on the quadrilateralization strategy in (2), we extend the method to PDEs on evolving surfaces—a common setting in biology and physics where geometry changes over time and influences pattern development. We adapt the closest point projection method to account for time-dependent surface evolution and employ an arbitrary Lagrangian–Eulerian (ALE) mapping to preserve mesh quality and geometric consistency. As a case study, we examine two biologically motivated surface growth models, illustrating how

curvature and geometric evolution shape self-organization dynamics.

Chapter 2

Background

This chapter presents background material for the subsequent chapters. The selection of topics is subjective, allowing readers to skip certain sections, revisit them later, or consult the referenced literature for additional details. In Section 2.1, we introduce fundamental definitions, notation, and relevant results from functional analysis. Multivariate polynomial spaces are introduced in Section 2.2, including the polynomial space corresponding to ℓ^p-ball index sets. Section 2.3 introduces total variation, a key tool for measuring function smoothness in our thesis, which plays a crucial role in obtaining estimations in Chapter 3. An exposition on univariate quadrature rules is presented in Section 2.4. Elements of this chapter are adapted from [146].

2.1 Mathematical foundations

In this section, we introduce the definitions, notation, and foundational results that will be used throughout this thesis. These concepts form the basis for the theoretical and computational framework developed in the subsequent chapters.

2.1.1 Notation and preliminaries

We will use boldface letters to represent vectors and matrices. For instance, a point $\mathbf{x}$ in $\mathbb{R}^d$ or $\mathbb{C}^d$ can be written as

$$\mathbf{x} := (x_1, \ldots, x_d).$$

Throughout the thesis, we denote the closed d-dimensional standard hypercube by $\Omega_d := [-1,1]^d$, $d \geq 1$, and the standard d-simplex in $\mathbb{R}^d$ by $\triangle_d := \{\mathbf{x} \in \mathbb{R}^d : x_1, \ldots, x_d \geq 0, \sum_{i=1}^d |x_i| \leq 1\}$. We also define $\mathbb{R}_+ := \{x \in \mathbb{R} : x \geq 0\}$. For an open set $\Omega \subseteq \mathbb{R}^d$, we denote its interior by $\mathring{\Omega}$, its closure by $\overline{\Omega}$, and its boundary by $\partial\Omega := \overline{\Omega} \setminus \Omega$.

For vectors $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we use $\langle \mathbf{x}, \mathbf{y} \rangle$ to denote the standard Euclidean inner product, and $\|\mathbf{x}\|$ for the corresponding Euclidean norm.

The notation $\mathbb{N}^d$ stands for the set of all d-tuples $\boldsymbol{\alpha} := (\alpha_1, \alpha_2, \ldots, \alpha_d)$, where $\alpha_i \in \mathbb{N} = \{0, 1, 2, \ldots\}$. Such a d-tuple is called a *multi-index*.

Multi-index notation provides a concise and systematic way to express multi-dimensional operations, which is particularly useful for defining polynomial spaces, partial derivatives, and basis functions in higher dimensions. Given multi-indices $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ and $\boldsymbol{\beta} = (\beta_1, \ldots, \beta_d)$, we define the following componentwise operation $\boldsymbol{\alpha} + \boldsymbol{\beta} := (\alpha_1 + \beta_1, \ldots, \alpha_d + \beta_d)$, and use the convention

$$\boldsymbol{\alpha} \leq \boldsymbol{\beta} \iff \alpha_j \leq \beta_j, \quad j = 1, 2, \ldots, d.$$

Let $\mathbf{1} := (1, \ldots, 1) \in \mathbb{N}^d$. For a scalar $t \in \mathbb{R}$, we define $\boldsymbol{\alpha} + t := \boldsymbol{\alpha} + t \cdot \mathbf{1} = (\alpha_1 + t, \ldots, \alpha_d + t)$ and $\boldsymbol{\alpha}' := \prod_{j=1}^d \alpha_j$.

Definition 1 (Product Basis)**.** *Let ϕ and ϕ_α be continuous functions of a single variable, and let $j = 1, \ldots, d$. The* product basis *is defined as follows:*

- *For a point $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$, the function $\phi(\mathbf{x})$ is given by*

$$\phi(\mathbf{x}) := \prod_{j=1}^{d} \phi(x_j).$$

- *For a multi-index $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$, the function $\phi_{\boldsymbol{\alpha}}(\mathbf{x})$ is defined as*

$$\phi_{\boldsymbol{\alpha}}(\mathbf{x}) := \prod_{j=1}^{d} \phi_{\alpha_j}(x_j).$$

In particular, the multivariate monomial associated with the multi-index $\boldsymbol{\alpha}$ is expressed as

$$\mathbf{x}^{\boldsymbol{\alpha}} = \prod_{j=1}^{d} x_j^{\alpha_j}.$$

Definition 2. *A sequence of polynomials $\{\psi_{\boldsymbol{\alpha}}(\boldsymbol{x})\}_{\boldsymbol{\alpha} \in \mathbb{N}^d}$ defined on the domain Ω_d is called a set of **orthogonal polynomials** with respect to a positive function $w(\boldsymbol{x})$ if the following orthogonality condition is satisfied:*

$$\int_{\Omega_d} w(\boldsymbol{x})\, \psi_{\boldsymbol{\alpha}}(\boldsymbol{x}) \psi_{\boldsymbol{\beta}}(\boldsymbol{x})\, d\boldsymbol{x} = \delta_{\boldsymbol{\alpha},\boldsymbol{\beta}}\, c_{\boldsymbol{\alpha}}, \quad \forall \boldsymbol{\alpha}, \boldsymbol{\beta} \in \mathbb{N}^d,$$

where:

- *$w(\boldsymbol{x}) > 0$ is known as the weighting function,*
- *$\delta_{\boldsymbol{\alpha},\boldsymbol{\beta}}$ denotes the Kronecker delta, and $c_{\boldsymbol{\alpha}} > 0$ is a normalization constant.*

*If $c_{\boldsymbol{\alpha}} = 1$ for all $\boldsymbol{\alpha} \in \mathbb{N}^d$, the polynomials are said to be **orthonormal**.*

We define the ℓ^p-norm by:

$$\|\boldsymbol{\alpha}\|_p := \begin{cases} \left(\alpha_1^p + \cdots + \alpha_d^p\right)^{1/p}, & 1 \le p < \infty, \\ \max_{1\le i\le d} \alpha_i, & p = \infty. \end{cases} \tag{2.1}$$

Definition 3 (Mixed Partial Derivative)**.** *Given a multi-index $\boldsymbol{\alpha} = (\alpha_1, \ldots, \alpha_d)$ and a multivariate function $f(\mathbf{x})$, the $\|\boldsymbol{\alpha}\|_1$-th mixed partial derivative of f (provided it exists) is denoted and defined as*

$$\partial^{\boldsymbol{\alpha}} f = \frac{\partial^{\|\boldsymbol{\alpha}\|_1} f}{\partial x_1^{\alpha_1} \cdots \partial x_d^{\alpha_d}} = \partial_{x_1}^{\alpha_1} \cdots \partial_{x_d}^{\alpha_d} f,$$

where $\|\boldsymbol{\alpha}\|_1 = \sum_{j=1}^d \alpha_j$ represents the sum of the components of the multi-index $\boldsymbol{\alpha}$.

Throughout this thesis, we adopt the standard Landau symbols for asymptotic notation, assuming that $g(x) \neq 0$ for all sufficiently large x:

$$f \in O(g) \iff \limsup_{x\to\infty} \frac{|f(x)|}{|g(x)|} < \infty, \qquad f \in o(g) \iff \lim_{x\to\infty} \frac{|f(x)|}{|g(x)|} = 0.$$

Definition 4. *A $d \times d$ matrix $\mathbf{A}$ is called an **orthogonal matrix** if it satisfies the condition:*

$$\mathbf{A}\mathbf{A}^\top = I,$$

where $\mathbf{A}^\top$ denotes the transpose of $\mathbf{A}$, and $\mathbf{I}$ is the identity matrix.

Definition 5. *A function $f : \Omega \subset \mathbb{R}^d \to \mathbb{R}^{d+1}$ is Lipschitz continuous at $\boldsymbol{x} \in \Omega$ if there is a constant C such that*

$$|f(\boldsymbol{y}) - f(\boldsymbol{x})| \le C|\boldsymbol{y} - \boldsymbol{x}|$$

for all $\boldsymbol{y} \in \Omega$ sufficiently near $\boldsymbol{x}$.

Definition 6. *A mapping $f : \Omega \to \mathbb{R}$ is called bi-Lipschitz, or C-bi-Lipschitz to emphasize the constant, if there is a constant $C \ge 1$ such that*

$$C^{-1}|\boldsymbol{y} - \boldsymbol{x}| \le |f(\boldsymbol{y}) - f(\boldsymbol{x})| \le C|\boldsymbol{y} - \boldsymbol{x}|.$$

Definition 7. *The Kronecker tensor product of two matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ of dimensions $p \times q$ and $r \times s$, respectively, is the matrix $\boldsymbol{A} \otimes \boldsymbol{B}$ of dimension $pr \times qs$ with $p \times q$ block form, where the (i, j) block is $a_{ij}\boldsymbol{B}$.*

Example 8. *For instance, if*

$$\boldsymbol{A} = \begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{pmatrix} a & b \\ c & d \end{pmatrix},$$

then

$$\boldsymbol{A}\otimes\boldsymbol{B}=\begin{pmatrix}1\cdot B & 2\cdot B\\ 3\cdot B & 4\cdot B\end{pmatrix}=\begin{pmatrix}1\begin{pmatrix}a & b\\ c & d\end{pmatrix} & 2\begin{pmatrix}a & b\\ c & d\end{pmatrix}\\ 3\begin{pmatrix}a & b\\ c & d\end{pmatrix} & 4\begin{pmatrix}a & b\\ c & d\end{pmatrix}\end{pmatrix}.$$

2.1.2 Banach and Hilbert spaces

We begin by introducing the general concept of a normed vector space, which encompasses both the Banach and Hilbert spaces as specific examples.

Definition 9 (Normed Vector Space)**.** *Let X be a vector space over $\mathbb{R}$ (or $\mathbb{C}$). A norm on X is a map $\|\cdot\|_X : X \to \mathbb{R}_+$ satisfying:*

(i) $\|x+y\|_X \leq \|x\|_X + \|y\|_X$ *for all* $x, y \in X$ **(*triangle inequality*)**.

(ii) $\|tx\|_X = |t| \cdot \|x\|_X$ *for all* $x \in X$ *and* $t \in \mathbb{R}$ *(or* $\mathbb{C}$*)* **(*homogeneity*)**.

(iii) $\|x\|_X = 0$ *if and only if* $x = 0$ **(*positivity definiteness*)**.

The pair $(X, \|\cdot\|_X)$ is called a normed vector space. If the map $\|\cdot\|_X$ satisfies only (i) and (ii), it is called a semi-norm.

The space X is said to be **complete** if every Cauchy sequence $\{x_n\}$ in X, that is, $\|x_n - x_m\|_X \to 0$ as $n, m \to \infty$, converges to some $x \in X$, i.e., $\|x_n - x\|_X \to 0$ as $n \to \infty$.

A complete normed vector space is known as a *Banach space*, formally defined as follows:

Definition 10. *A Banach space is a normed vector space $(X, \|\cdot\|_X)$ that is complete, meaning that every Cauchy sequence $\{x_n\}$ in X converges to some $x \in X$ with respect to the norm $\|\cdot\|_X : X \to \mathbb{R}_+$.*

Definition 11 (Lebesgue Spaces)**.** *The Banach space of all Lebesgue measurable functions with finite L^p-norm is denoted by $L^p(\Omega)$ for $1 \leq p \leq +\infty$, and is defined as:*

$$L^p(\Omega) := \{f : \Omega \to \mathbb{R} \mid \|f\|_{L^p(\Omega)} < +\infty\},$$

where the L^p-norm $\|f\|_{L^p(\Omega)} : L^p(\Omega) \to \mathbb{R}_+$ is defined as:

$$\|f\|_{L^p(\Omega)}^p := \int_\Omega |f|^p \, dx, \quad \text{for } 1 \leq p < \infty.$$

In the case where $p = \infty$, the norm is defined as:

$$\|f\|_{L^\infty(\Omega)} := \inf\{C \geq 0 : |f(x)| \leq C \text{ for almost every } x \in \Omega\}.$$

We use the standard definition of Sobolev spaces; see, for example, [1].

Definition 12 (Sobolev spaces)**.** *Let $k \in \mathbb{N} \cup \{0\}$ and $p \in [1, \infty]$, then the Sobolev space $W^{k,p}(\Omega)$ is defined by*

$$W^{k,p}(\Omega) := \{u \in L^p(\Omega) \,:\, \partial^{\boldsymbol{\alpha}} u \in L^p(\Omega) \; \forall \boldsymbol{\alpha} \text{ with } |\boldsymbol{\alpha}| \leq k\}. \tag{2.2}$$

This space is equipped with the norm

$$\|u\|_{W^{k,p}(\Omega)} := \sum_{|\boldsymbol{\alpha}| \leq k} \|\partial^{\boldsymbol{\alpha}} u\|_{L^p(\Omega)}. \tag{2.3}$$

For $p = \infty$, the norm is defined as:

$$\|u\|_{W^{k,\infty}(\Omega)} := \max_{|\boldsymbol{\alpha}| \leq k} \operatorname{ess\,sup}_{x \in \Omega} |\partial^{\boldsymbol{\alpha}} u(x)|.$$

Definition 13 (Hilbert Space)**.** *A Hilbert space is a normed vector space $(H, \|\cdot\|_H)$ over $\mathbb{R}$ (or $\mathbb{C}$), equipped with an inner product*

$$\langle \cdot, \cdot \rangle_H : H \times H \to \mathbb{R} \quad (\text{or } \mathbb{C}),$$

such that the norm is induced by the inner product:

$$\|f\|_H := \sqrt{\langle f, f \rangle_H}, \quad \text{for all } f \in H.$$

Moreover, H is complete with respect to this norm; that is, every Cauchy sequence in H converges to an element of H.

Theorem 14. *Let $p = 2$. Then the space $L^2(\Omega)$, consisting of all Lebesgue measurable functions $f : \Omega \to \mathbb{R}$ such that $\int_\Omega |f(x)|^2 \, dx < \infty$, is a Hilbert space. It is equipped with the inner product*

$$\langle f, g \rangle_{L^2(\Omega)} := \int_\Omega f(x)\, g(x)\, dx, \quad \text{for all } f, g \in L^2(\Omega).$$

Definition 15 (Sequential Continuity)**.** *Let Ω be an open set, $f : \Omega \to \mathbb{R}$ a function, and $a \in \Omega$. The function f is said to be sequentially continuous at a if, for every sequence (x_n) in Ω such that*

$$\lim_{n \to \infty} x_n = a,$$

it follows that

$$\lim_{n \to \infty} f(x_n) = f(a).$$

We now define the space of r-times continuously differentiable functions.

Definition 16. *The space of continuous functions on $\Omega \subset \mathbb{R}^d$ is defined by*

$$C^0(\Omega) := \{f : \Omega \to \mathbb{R} \mid f \text{ is sequentially continuous}\}. \tag{2.4}$$

More generally, for any integer $r \geq 0$, the space of r-times continuously differentiable functions on Ω is

defined as

$$C^r(\Omega) := \left\{ f \in C^0(\Omega) \;\middle|\; \partial^{\boldsymbol{\alpha}} f \in C^0(\Omega) \text{ for all multi-indices } \boldsymbol{\alpha} \in \mathbb{N}^d \text{ with } \|\boldsymbol{\alpha}\|_1 \leq r \right\}. \tag{2.5}$$

Theorem 17 (Completeness of $C^r(\overline{\Omega})$)**.** *Let $0 \leq r < \infty$, and let $\overline{\Omega}$ be compact. If $C^r(\overline{\Omega})$ denotes the space of functions whose partial derivatives up to order r extend continuously to $\overline{\Omega}$, then $C^r(\overline{\Omega})$, endowed with the norm*

$$\|f\|_{C^r(\overline{\Omega})} = \sum_{\substack{\boldsymbol{\alpha} \in \mathbb{N}^d \\ \|\boldsymbol{\alpha}\|_1 \leq r}} \sup_{\mathbf{x} \in \overline{\Omega}} |\partial^{\boldsymbol{\alpha}} f(\mathbf{x})|, \tag{2.6}$$

is a Banach space; see, for example, [20].

Definition 18. *The space of smooth functions $C^\infty(\Omega)$ is defined as the limit of the $C^r(\Omega)$ spaces,*

$$C^\infty(\Omega) := \bigcap_{r \in \mathbb{N}} C^r(\Omega). \tag{2.7}$$

Furthermore, we define the space of compactly supported smooth functions $C_c^\infty(\Omega)$ as follows:

$$C_c^\infty(\Omega) := \{\phi \in C^\infty(\Omega) : supp(\phi) \subset\subset \Omega\}. \tag{2.8}$$

Here, we would like to distinguish between C^∞ functions and analytic functions. Although both C^∞ and analytic functions possess infinitely many derivatives, analytic functions have the additional characteristic that their Taylor series at each point converges to the function within some neighborhood. More precisely, following the standard definition in [123]:

Definition 19 (Analytic functions)**.** *Let $\Omega \subset \mathbb{C}^d$ be an open set, $\mathbf{z}_0 = (z_{01}, \ldots, z_{0d}) \in \Omega$, and $f : \Omega \to \mathbb{C}$. A function f is analytic at $\mathbf{z}_0$ if f has a power series expansion valid in a neighborhood of $\mathbf{z}_0$. This means that there exists an $\varepsilon > 0$ and a power series*

$$\sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} c_{\boldsymbol{\alpha}} (z_1 - z_{01})^{\alpha_1} \cdots (z_d - z_{0d})^{\alpha_d},$$

that converges in $\mathcal{B}(\mathbf{z}_0, \varepsilon) = \{\mathbf{z} \in \mathbb{C}^d : \|\mathbf{z} - \mathbf{z}_0\| < \varepsilon\}$ and satisfies

$$f(\mathbf{z}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} c_{\boldsymbol{\alpha}} (z_1 - z_{01})^{\alpha_1} \cdots (z_d - z_{0d})^{\alpha_d},$$

for all $\mathbf{z} \in \mathcal{B}(\mathbf{z}_0, \varepsilon)$. A function f is analytic *on a set $\Omega \subset \mathbb{C}^d$ if f is analytic at each $\mathbf{z}_0 \in \Omega$.*

Example 20. *Consider the function $f : \mathbb{R} \to \mathbb{R}$ defined by:*

$$f(x) = \begin{cases} \frac{1}{e^{1/x}}, & \text{if } x > 0, \\ 0, & \text{if } x \leq 0. \end{cases}$$

This function is $C^\infty(\mathbb{R})$ but not analytic. It fails to be analytic at $x = 0$ because the Taylor series expansion around $x = 0$ does not converge to $f(x)$. Specifically, while all derivatives of f at $x = 0$ vanish, the function $f(x)$ is nonzero for $x > 0$.

We now introduce the concept of the Bernstein ellipse, a standard tool in approximation theory [133].

Definition 21. *The Bernstein ellipse $\mathcal{E}_\rho$ is defined by:*

$$\mathcal{E}_\rho := \left\{ z \in \mathbb{C} \mid z = \frac{1}{2}\left(u + u^{-1}\right),\ |u| = \rho > 1 \right\}, \tag{2.9}$$

which has the foci at ± 1 with the major and minor semi-axes given by $\frac{1}{2}\left(\rho + \rho^{-1}\right)$ and $\frac{1}{2}\left(\rho - \rho^{-1}\right)$, respectively. A natural extension of the Bernstein ellipse $\mathcal{E}_\rho$ to $\mathbb{C}^d$ is the polyellipse:

$$\mathcal{E}_\rho := \bigotimes_{j=1}^{d} \mathcal{E}_{\rho_j}. \tag{2.10}$$

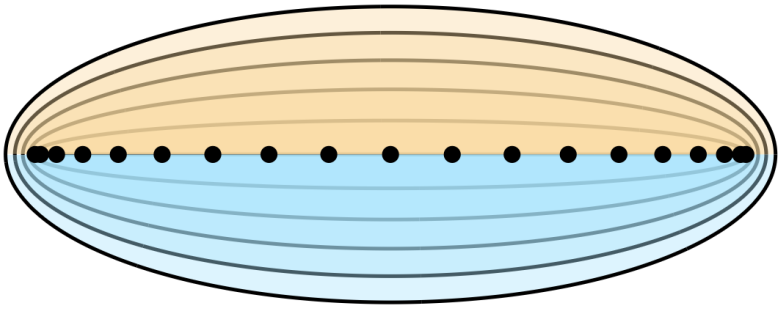


Figure 2.1: Bernstein ellipses for $\rho = 1, 1.2, \ldots, 1.5$, illustrating the variation of ellipse shapes as ρ increases.

2.2 Multivariate polynomial spaces

We consider the real polynomial vector spaces of degree $n \in \mathbb{N}$ defined over the d-dimensional Euclidean space. More precisely:

Definition 22. *Let $\{\mathbf{x}^{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha} \in \mathbb{N}^d}$ denote the set of monomials, where $\mathbf{x}^{\boldsymbol{\alpha}} = \prod_{i=1}^{d} x_i^{\alpha_i}$, with $\mathbf{x} \in \mathbb{R}^d$ and $\boldsymbol{\alpha} \in \mathbb{N}^d$. The monomials are ordered according to the lexicographic order $\preceq$ on multi-indices.*

A polynomial space of total degree at most $n \in \mathbb{N}$ is defined as the span of all monomials with total degree at most n:

$$\Pi_{d,n} := \operatorname{span}\{\mathbf{x}^{\boldsymbol{\alpha}} : \|\boldsymbol{\alpha}\|_1 := \alpha_1 + \alpha_2 + \cdots + \alpha_d \leq n\}.$$

More generally, a set of polynomials $\{b_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in\mathbb{N}^d} \subseteq \Pi_{d,n}$ *is called a polynomial basis if:*

$$\Pi_{d,n} = \text{span}\{b_{\boldsymbol{\alpha}} : \|\boldsymbol{\alpha}\|_1 \leq n\}.$$

The restriction of the polynomial space to a domain $\Omega_d \subseteq \mathbb{R}^d$ *is denoted as:*

$$\Pi_{d,n}(\Omega_d) := \{Q|_{\Omega_d} : Q \in \Pi_{d,n}\}.$$

We define multivariate polynomial spaces associated with ℓ^p-ball index sets in $\mathbb{N}^d$, where the multi-indices are constrained by the ℓ^p-norm, with $1 \leq p \leq \infty$. These spaces generalize the concept of polynomial degree under different norm constraints.

Definition 23. *The polynomial space corresponding to* ℓ^p*-ball index set is defined as* $\Pi_{d,n,p} := span\{x^{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in A_{d,n,p}}$ *where* $A_{d,n,p}$ *is defined as*

$$A_{d,n,p} = \{\boldsymbol{\alpha} \in \mathbb{N}^d : \|\boldsymbol{\alpha}\|_p \leq n\},\ 1 \leq p \leq \infty. \tag{2.11}$$

Note that the index set (2.11) is a downward closed set [1], encompassing several important index sets as specific cases. For instance, the total, Euclidean, and maximal degrees of a multivariate polynomial up to degree n correspond to $p = 1, 2, \infty$ in equation (2.11), respectively. To illustrate the distribution of interpolation points within $A_{d,n,p}$, we present in Figure 2.2 the index set $A_{d,30,p}$ for $d = 2$ and three different values of p.

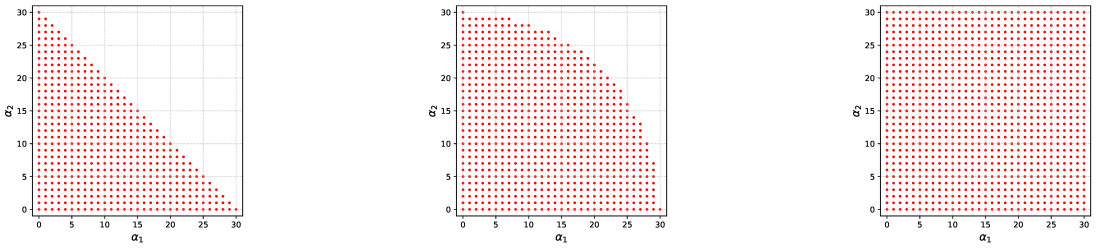


Figure 2.2: The index set $A_{d,30,p}$ for $p = 1$ (left), $p = 2$ (middle), and $p = \infty$ (right) in dimension $d = 2$.

Note that the cardinality of the index set is monotonically increasing in p,

Lemma 24. *Let* $\|\boldsymbol{\alpha}\|_p$ *be defined in equation* (2.1)*. For* $1 \leq a \leq b \leq \infty$ *and* $\boldsymbol{\alpha} \in \mathbb{N}^d$*, we have*

$$\|\boldsymbol{\alpha}\|_a \leq d^{\frac{1}{a}-\frac{1}{b}}\|\boldsymbol{\alpha}\|_b. \tag{2.12}$$

Moreover, this inequality is optimal in the sense that there is no smaller constant such that it still holds for all $\boldsymbol{\alpha} \in \mathbb{N}^d$*.*

Proof. In the case $a = b = \infty$, the bound is trivial. There is nothing to show if $\boldsymbol{\alpha} = 0$ so we may

[1] A set Λ is labelled *downward closed* if $\boldsymbol{\nu} \in \Lambda$ whenever there exists $\boldsymbol{\mu} \in \Lambda$ such that $\boldsymbol{\nu} \leq \boldsymbol{\mu}$.

assume $\boldsymbol{\alpha} \neq 0$. Moreover, if $b = \infty$ and $1 \leq a < \infty$, we have

$$\|\boldsymbol{\alpha}\|_a^a = \sum_{j=1}^{d} |\alpha_j|^a \leq d\|\boldsymbol{\alpha}\|_\infty^a.$$

For $1 \leq a \leq b < \infty$, we can use Hölder's inequality with $r = b/a \geq 1$ and $s \geq 1$ with $\frac{1}{r} + \frac{1}{s} = 1$, i.e., with $s = b/(b-a)$, to derive

$$\|\boldsymbol{\alpha}\|_a^a = \sum_{j=1}^{d} |\alpha_j|^a \leq \left(\sum_{j=1}^{d} 1^s\right)^{1/s} \left(\sum_{j=1}^{d} |\alpha_j|^{ar}\right)^{1/r} = d^{1-a/b}\|\boldsymbol{\alpha}\|_b^a.$$

Taking the ath root yields the desired result. The constant is the best possible because we have equality in the right-hand side of equation (2.12) for the vector $\boldsymbol{\alpha} = (1, \ldots, 1)^T$. □

Remark 25. *We emphasize that the multivariate polynomials associated with the ℓ^p ball index set in $\mathbb{N}^d$ defined in equation (2.11), can be generalized to the range $0 < p \leq \infty$. Additionally, Lemma 24 will take the form*

$$\|\boldsymbol{\alpha}\|_b \leq \|\boldsymbol{\alpha}\|_a \leq d^{\frac{1}{a}-\frac{1}{b}}\|\boldsymbol{\alpha}\|_b,$$

where the case $0 < a < 1$ is covered by reversed Hölder's inequality [70, Theorem 13.6]. An example of the index set $A_{d,30,p}$ for $0 < p < 1$ is illustrated in Figure 2.3.

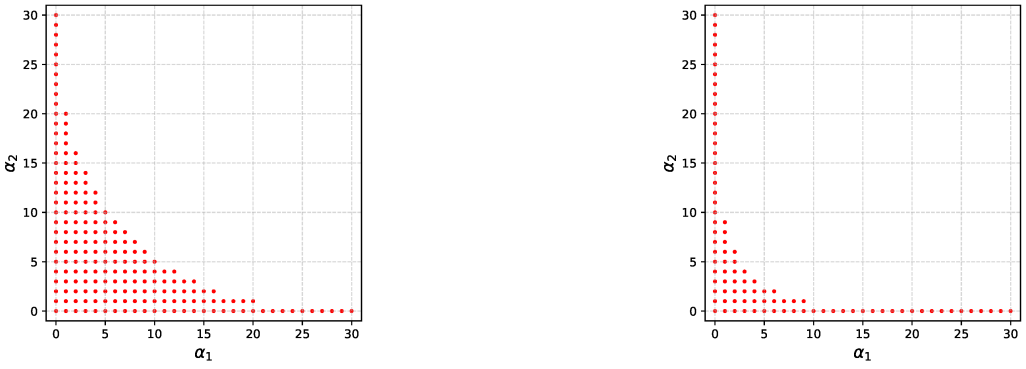


Figure 2.3: The index set $A_{d,30,p}$ for $p = 1/2$ (left), and $p = 1/3$ (right) in dimension $d = 2$.

In contrast to the common *total-degree* polynomial space, which corresponds to the ℓ^1 ball index set (also known as the full polynomial space), we focus on the vector space of all real polynomials associated with the ℓ^∞ ball index set. For the rest of this work, we will restrict our attention exclusively to this space, denoted by

$$\Pi_{d,n} = \Pi_{d,n,\infty} = \operatorname{span}\{\mathbf{x}^{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha} \in A_{d,n}}, \quad A_{d,n} = A_{d,n,\infty}$$

i.e. the space of multivariate polynomials of degree at most n in each variable separately.

2.3 Total variation

A common approach to measure the smoothness of a function is through V_f, the *total variation*, which can be intuitively understood as the cumulative length of the monotone segments of f. This concept was introduced by Jordan in the late nineteenth century and was primarily studied in the context of Fourier series [78, 79]. For a one-dimensional function we have:

Definition 26 (Bounded variation)**.** *For a function $f : [a,b] \to \mathbb{R}$, we define the total variation of f on the interval $[a,b]$ by*

$$V_f := \sup \left\{ \sum_{i=1}^{n} |f(x_i) - f(x_{i-1})| : a = x_0 \leq x_1 \leq \cdots \leq x_n = b \text{ for some } n \in \mathbb{N} \right\}$$

If $V_f < \infty$, we say that f is of bounded variation, and we define

$$\mathrm{BV}([a,\, b]) := \{ f : [a,\, b] \to \mathbb{R} : f \text{ has bounded variation on } [a,\, b] \}.$$

Before moving on to the generalizations of variation in Ω_d, it is useful to review some key concepts regarding the relationship between bounded variation and absolutely continuous functions, as well as the link between absolute continuity and derivatives. Let's first recall the definition of an absolutely continuous function:

Definition 27. *Let I be an interval. A function $f : I \to \mathbb{R}$ is absolutely continuous on I if for each $\varepsilon > 0$, there exists $\delta > 0$ such that*

$$\sum_{i=1}^{n} |f(d_i) - f(c_i)| < \varepsilon$$

whenever $\{[c_i, d_i] : 1 \leq i \leq n\}$ is a finite collection of non-overlapping intervals in I and

$$\sum_{i=1}^{n} (d_i - c_i) < \delta.$$

Theorem 28. *If a function f is absolutely continuous on the interval $[a,b]$, then $f \in \mathrm{BV}([a,\, b])$.*

Proof. See [61]. □

When the derivative of a continuous function f is bounded, we can conclude that f is absolutely continuous. This provides a tool for showing that a function is absolutely continuous, and thus of bounded variation.

Theorem 29 (Bounded derivative implies absolute continuity)**.** *If f is continuous on $[a,b]$ and f' exists and is bounded on (a,b), then f is absolutely continuous on $[a,b]$; see [61].*

In particular, functions of bounded variation are differentiable almost everywhere [4] and sat-

isfy

$$\int_a^b \left|\frac{df}{dx}\right| dx \leq V_f. \tag{2.13}$$

Moreover, if f is absolutely continuous, we have equality

$$V_f = \sup_{\mathcal{Y}} \sum_{i=1}^{n} |f(a_{i+1}) - f(a_i)| = \sup_{\mathcal{Y}} \sum_{i=0}^{n} \left| \int_{x_i}^{x_{i+1}} f'(x)\, dx \right| \leq \sup_{\mathcal{Y}} \sum_{i=0}^{n} \int_{x_i}^{x_{i+1}} \left|f'(x)\right| dx = \int_a^b \left|f'(x)\right| dx,$$

where $\mathcal{Y}$ is collection of ordered $(n+1)$-ples of points $a = x_0 \leq x_1 \leq \cdots \leq x_n = b$, $n \in \mathbb{N}$ in $[a,b]$.

There are examples illustrating the interplay between differentiability and variation.

Example 30. *Consider the function $f(x) = x\sin(x^{-1})$ for $x \in (0,1]$. The function is differentiable but has unbounded variation. On the other hand, discontinuous functions, like the indicator functions of intervals, can possess bounded variation.*

In one dimension, the concept of bounded variation is well-defined and straightforward. However, in higher dimensions, various generalizations exist, each preserving different properties of univariate functions of bounded variation. We first recall the Hardy–Krause variation [2, 101], and then introduce the Chebyshev-weighted directional variation used in the interpolation estimates of Chapter 3.

Following [2], we begin by recalling the definition of the *Vitali variation*.

Definition 31. *The Vitali variation of a smooth function $f : \Omega_d \longrightarrow \mathbb{R}$*

$$V^{\mathcal{V}}_{(f;\Omega_d)} := \int_{\Omega_d} \left|\frac{\partial^d f}{\partial x_1 \dots \partial x_d}\right| dx_1 \dots dx_d\,. \tag{2.14}$$

If $V^{\mathcal{V}}_{(f;\Omega_d)} < \infty$, we say that the function f is of bounded variation in the sense of Vitali.

equation (2.14) holds whenever the indicated partial derivative is continuous on Ω_d.

If we fix $d = 2$, then equation (2.14) takes the form

$$V^{\mathcal{V}}_{(f;[-1,1]^2)} = \int_{[-1,1]^2} \left|\frac{\partial^2 f}{\partial x_1 \partial x_2}\right| dx_1 dx_2 \tag{2.15}$$

Note that a function of bounded variation in the sense of *Vitali*[2] can still be quite wild, because there exist many nonconstant functions f on Ω_d whose *Vitali variation* is zero.

Example 32. *The Vitali variation fails to capture the variation of polynomials with low degrees. For instance, the function $f : \Omega_d \to \mathbb{R}$, $f(\mathbf{x}) = \sum_{i=1}^{d} x_i$ satisfies $V^{V}_{(f;\Omega_d)} = 0$ for $d \geq 2$.*

One very straightforward way to avoid this problem is to just add the lower-dimensional Vitali-variations of the function, as the *Hardy–Krause variation* does. This definition recaptures the

[2]Note first that when $d = 1$, *Vitali variation* is simply total variation (Definition 26), since the rectangles in this case are intervals.

notion of Vitali and Hardy–Krause [2, 101]. That is, the sum of the Vitali variations of f restricted to each face of $[-1,1]^d$ anchored to $x = \mathbf{1}_d := (1, \cdots, 1)$. In higher dimensions, the Hardy-Krause variation may be defined in terms of the integral of partial derivatives;

Definition 33. *The variation of a smooth function f on Ω_d in the sense of Hardy-Krause anchored at $x = \mathbf{1}_d$ is*

$$V^{\mathbf{HK}}_{(f;\Omega_d)} := \sum_{k=1}^{d} \sum_{1\le i_1<\cdots<i_k\le d} V^{(k)}(f; i_1, \ldots, i_k), \tag{2.16}$$

where

$$V^{\mathbf{HK}}_{(f;i_1,\ldots,i_k)} := \int_{\Omega_k} \left| \frac{\partial^k f}{\partial x_{i_1} \cdots \partial x_{i_k}} \right|_{x_j=1,\, j\neq i_1,\ldots,i_k} dx_{i_1} \cdots dx_{i_k}, \tag{2.17}$$

represents the variation of the projection of $k-$variables. If $V^{\mathbf{HK}}_{(f;\Omega_d)} < \infty$, we say that the function f is of bounded variation in the sense of Hardy-Krause.

Combining equation (2.16) and equation (2.17), we obtain

$$V^{\mathbf{HK}}_{(f;\Omega_d)} = \int_{\Omega_d} \left| \frac{\partial^d f}{\partial x_1 \ldots \partial x_d} \right| dx_1 \ldots dx_d + \sum_{i=1}^{d} V^{\mathbf{HK}}_{f_1^{(i)}} \tag{2.18}$$

in which $f_1^{(i)}$ is the restriction of the function f to the boundary $x_i = 1$. Since these restrictions are functions of $d-1$ variables, equation (2.18) is recursive. For $d = 2$, then equation (2.18) takes the form

$$V^{\mathbf{HK}}_{(f(x_1,x_2);[-1,1]^2)} = \int_{[-1,1]^2} \left| \frac{\partial^2 f}{\partial x_1 \partial x_2} \right| dx_1 dx_2 + \int_{-1}^{1} \left| \frac{\partial f(\cdot, y_0)}{\partial x_1} \right| dx_1 + \int_{-1}^{1} \left| \frac{\partial f(x_0, \cdot)}{\partial x_2} \right| dx_2, \tag{2.19}$$

for x_0 and $y_0 \in [-1,1]$. In other words, a smooth function is said to be of bounded variation on $[-1,1]^2$ in the sense of Hardy-Krause, if $f(\cdot, y_0)$, $f(x_0, \cdot)$ are of bounded variation in the sense of Definition 26.

Definition 34 (Chebyshev-weighted r^{th}-order directional variation)**.** *Let $r \in \mathbb{N}_0$ and let $f : \Omega_d \to \mathbb{R}$. For $d = 1$, set*

$$V_{f,r} := V\left(f^{(r)}; [-1,1]\right),$$

where $V(\cdot; [-1,1])$ denotes the one-dimensional total variation from Definition 26. For $d \ge 2$ and $i = 1, \ldots, d$, write

$$\widehat{\mathbf{x}}_i = (x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_d), \qquad \omega^{(i)}_{d-1}(\widehat{\mathbf{x}}_i) := \prod_{j\neq i} \frac{1}{\sqrt{1-x_j^2}}.$$

Assume that, for almost every $\widehat{\mathbf{x}}_i \in [-1,1]^{d-1}$, the one-dimensional function

$$x_i \longmapsto \partial^r_{x_i} f(x_1, \ldots, x_i, \ldots, x_d)$$

is of bounded variation on $[-1,1]$. *We define the directional variations*

$$V_{f,r}^{(i)} := \frac{1}{\pi^{d-1}} \int_{[-1,1]^{d-1}} V\left(\partial_{x_i}^r f(\cdot, \widehat{\mathbf{x}}_i); [-1,1]\right) \omega_{d-1}^{(i)}(\widehat{\mathbf{x}}_i)\, d\widehat{\mathbf{x}}_i, \tag{2.20}$$

and set

$$V_{f,r} := \max_{1 \le i \le d} V_{f,r}^{(i)}. \tag{2.21}$$

We say that f *has bounded Chebyshev-weighted* r^{th}*-order directional variation whenever* $V_{f,r} < \infty$. *If* f *is sufficiently smooth, then*

$$V_{f,r}^{(i)} = \frac{1}{\pi^{d-1}} \int_{\Omega_d} \left|\partial_{x_i}^{r+1} f(\mathbf{x})\right| \prod_{j \ne i} \frac{1}{\sqrt{1-x_j^2}}\, d\mathbf{x}. \tag{2.22}$$

Thus this variation is an axiswise, Chebyshev-weighted analogue of the one-dimensional quantity used in Chebyshev coefficient estimates. It is not the full Hardy–Krause variation; rather, it is the minimal directional quantity needed for the max-index estimates in Chapter 3.

2.4 Univariate quadrature rules

Let $f \in C^0([-1,1]) \cap L^1([-1,1])$. We consider the integral

$$I[f] := \int_{-1}^{1} f(\mathbf{x})\, d\mathbf{x}. \tag{2.23}$$

A quadrature rule approximates this integral by a weighted sum of function evaluations:

$$\mathcal{Q}_n^1[f] := \sum_{p \in P} \omega_p f(p), \tag{2.24}$$

where P is the set of quadrature nodes, and the nonzero constants ω_p are the corresponding weights. We briefly review standard univariate quadrature rules, which will be used in Section 3.8.1 for tensor-product integration in multiple dimensions. These rules are typically defined on the reference interval $[-1,1]$ or $[0,1]$, and can be mapped to other domains as needed. The most commonly used examples include:

Newton–Cotes rules

These rules use uniformly spaced points over the interval $[a,b]$. While easy to implement, they suffer from limited accuracy, especially for high-degree polynomials. For many integrands f, even if analytic, the Newton–Cotes rules can diverge exponentially as the number of points increases. As such, they are rarely used for high-precision integration.

Gauss quadrature

Gauss quadrature, introduced by Gauss [53], achieves high accuracy by choosing the nodes $\{p_j\}$ as the roots of orthogonal polynomials of degree $n+1$, and by assigning optimal positive weights $\{w_j\}$.

Gauss–Legendre rules: In this case, the nodes are chosen as the roots of the Legendre polynomial of degree $n+1$. The resulting quadrature formula is exact for all polynomials of degree up to $2n+1$ and performs exceptionally well in practice.

Clenshaw–Curtis rule

The **Clenshaw–Curtis formula** [35] integrates the degree n polynomial interpolant through $n+1$ Chebyshev points. These points correspond to the extreme values of the Chebyshev polynomials of the first kind. The abscissas are nested, computationally efficient, and are given by:

$$p_j = \frac{1}{2}\left(1 - \cos\left(\frac{\pi j}{n+1}\right)\right), \quad j = 1, \ldots, n.$$

The weights are determined by the interpolatory conditions. For odd n, one obtains:

$$\omega_j = \frac{2}{n+1} \sin\left(\frac{\pi j}{n+1}\right) \sum_{i=1}^{(n+1)/2} \frac{1}{2i-1} \sin\left(\frac{(2i-1)\pi j}{n+1}\right).$$

Generalized Quadrature rules

For integrands containing singularities, the integral can be expressed as:

$$f(x) = \xi(x) g(x),$$

where $\xi(x)$ is a known positive weight function. In such cases:

$$\int_{-1}^{1} f(x)\,dx = \int_{-1}^{1} \xi(x) g(x)\,dx \approx \sum_{j=0}^{n} \omega_j g(p_j).$$

The quadrature points $\{p_j\}$ are the roots of polynomials orthogonal with respect to the weighted scalar product:

$$\langle g, h \rangle_\xi := \int_{-1}^{1} \xi(x) g(x) h(x)\,dx.$$

Examples include:

- **Gauss–Chebyshev rules**: For $\xi(x) = \frac{1}{\sqrt{1-x^2}}$ on $[-1,1]$.
- **Gauss–Jacobi rules**: For $\xi(x) = (1-x)^\alpha (1+x)^\beta$ with parameters $\alpha, \beta > -1$.

Remark 35. *For piecewise polynomial integrands, as in finite element methods, composite rules combine scaled Gauss rules. In particular,* ***Gauss–Lobatto rules*** *include boundary points $p_0 = a$ and $p_n = b$, ensuring continuity across subdomains.*

A fundamental concept in constructing quadrature formulas is the requirement of *exactness* for a specific class of functions. Typically, for integration over an interval, this class consists of

polynomials up to degree n. The quadrature formula is then designed to produce the exact integral of the polynomial that interpolates the given data points $\{f(p_j)\}_{j=0}^{n}$.

However, Trefethen [134] demonstrates that the principle of *exactness* does not necessarily ensure high accuracy in practice. In particular, the Newton–Cotes, Clenshaw–Curtis, and Gauss quadrature formulas on $[-1, 1]$ do not always achieve the expected accuracy despite satisfying the exactness condition.

Chapter 3

Interpolation and integration in the hypercube

"Among all polynomials, the Chebyshev ones know how to dance with error."

— Numerical analysis folklore

Chebyshev polynomial-based approximation methods are widely used in numerical computations involving functions of one, two, or three real variables [41, 67, 130]. These methods are supported by a robust theoretical foundation and a rich collection of practical algorithms [41]. While substantial progress has been made in low-dimensional settings, extending these methods to higher-dimensional spaces remains an active area of research. In this chapter, we explore how Chebyshev approximation principles can be generalized to d-dimensional settings.

To ensure a clear progression of ideas, we first revisit the one-dimensional case. In Sections 3.1, 3.2, and 3.4, we provide a detailed discussion on univariate Chebyshev polynomials and their convergence results, following the approach of Trefethen [133]. This serves as a conceptual foundation for understanding higher-dimensional extensions. Building on this, Section 3.5 extends the methodology to multivariate settings, deriving convergence results and examining their theoretical and numerical implications for differentiable and analytic functions in Subsections 3.6 and 3.7, respectively. The transformed Chebyshev approximation under analytic maps is discussed in Subsection 3.7.1. The results presented in Section 3.5 have been published in [146]. In Section 3.8, we provide an overview of key aspects in the approximation of multivariate integrals. This is followed by a discussion of tensor-product integration methods and their convergence properties in Sections 3.8.1 and 3.8.2.

3.1 Chebyshev polynomials

Chebyshev polynomials are an important family of *orthogonal polynomials* in numerical analysis and scientific computing [52, 94, 133]. We begin by introducing the method in its one-dimensional form that was originally outlined in Trefethen [133].

For $j \geq 0$ the Chebyshev polynomial of degree j is denoted by $T_j(x)$ and is given by:

$$T_j(x) = \cos\left(j \cos^{-1} x\right), \quad x \in [-1,1]. \tag{3.1}$$

Chebyshev polynomials are orthogonal with respect to a weighted inner product, i.e.,

$$\int_{-1}^{1} \frac{T_i(x)T_j(x)}{\sqrt{1-x^2}}\,dx = \begin{cases} \pi, & \text{if } i = j = 0, \\ \pi/2, & \text{if } i = j \geq 1, \\ 0, & \text{if } i \neq j, \end{cases}$$

and as a consequence [105, Section 11.4] satisfy a 3-term recurrence relation [97, 18.9(i)],

$$T_{j+1}(x) = 2xT_j(x) - T_{j-1}(x), \quad j \geq 1,$$

with $T_1(x) = x$ and $T_0(x) = 1$. Chebyshev polynomials provide an efficient basis for representing polynomials on intervals, forming the foundation for a variety of numerical algorithms designed to work with functions of a single real variable.

3.2 Chebyshev polynomial interpolants and projections

The problem of determining optimal interpolation points is closely related to the Chebyshev minimal-amplitude theorem, stated as follows [18]:

Theorem 36 (Chebyshev minimal-amplitude property)**.** *Let $N \geq 1$. Among all monic polynomials p of degree N, the polynomial $2^{1-N}T_N$ has minimal maximum absolute value on $[-1,1]$. Equivalently,*

$$\|p\|_{C^0([-1,1])} \geq \|2^{1-N}T_N\|_{C^0([-1,1])} = 2^{1-N}.$$

In particular, if $x_1, \ldots, x_N \in [-1,1]$ are interpolation nodes, then their monic nodal polynomial satisfies

$$\max_{x\in[-1,1]} \left| \prod_{j=1}^{N} (x - x_j) \right| \geq 2^{1-N},$$

with equality for the zeros of T_N.

This theorem explains why Chebyshev-distributed points are natural for interpolation: the zeros of T_N minimize the leading nodal error factor when the interpolation nodes are unconstrained.

In this thesis, we use the closely related Chebyshev–Lobatto points because they include the endpoints of the interval and are therefore convenient for piecewise interpolation and spectral differentiation.

Let n be a positive integer, and let $\{x_j^{\text{cheb}}\}_{0\le j\le n}$ denote the set of $n+1$ Chebyshev–Lobatto points, defined by

$$\text{Cheb}_n := \left\{ x_j^{\text{cheb}} := \cos\left(\frac{j\pi}{n}\right) \;:\; 0 \le j \le n \right\}. \tag{3.2}$$

These points are the extrema of the Chebyshev polynomial T_n, or equivalently the zeros of $(1-x^2)T_n'(x)$ including the endpoints. They are often referred to as Chebyshev extreme points or Chebyshev points of the second kind. One of their key benefits becomes apparent when extending Chebyshev polynomials to piecewise polynomial interpolation: since the points include both endpoints of the interval, they facilitate seamless concatenation of interpolants across adjacent intervals. For a visual representation, the distribution of Chebyshev–Lobatto points is illustrated in Figure 3.1.

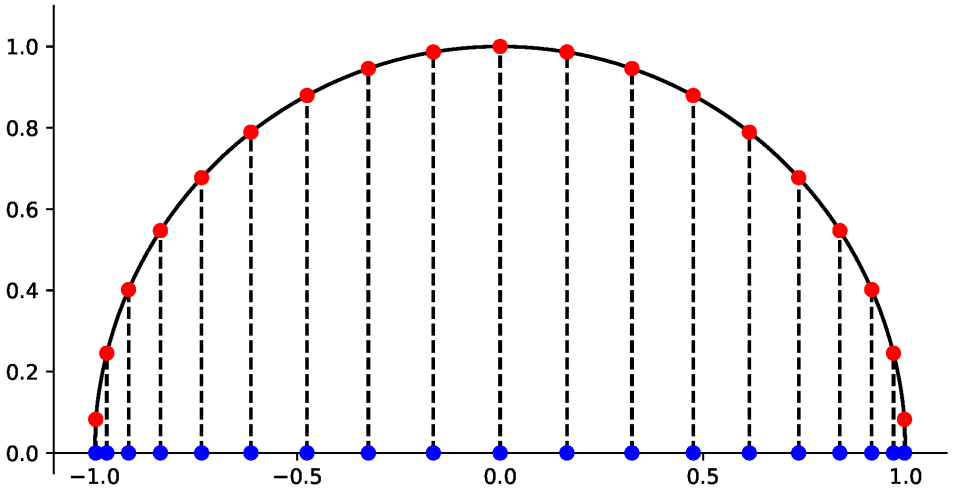


Figure 3.1: Chebyshev points $x_j^{\text{cheb}} \in [-1,1]$ (blue) for degree $n = 20$, along with equidistant auxiliary construction points (red) on the semicircle.

Given a set of data $\{f_j\}_{0\le j\le n}$, there is a unique polynomial $Q_{G_{1,n}}f$ of degree at most n such that $Q_{G_{1,n}}f(x_j^{\text{cheb}}) = f_j$ for $0 \le j \le n$. The polynomial $Q_{G_{1,n}}f_n$ is referred to as the Chebyshev interpolant of f of degree n. If $\{f_j\}_{0\le j\le n}$ are data from a continuous function $f : [-1,1] \to \mathbb{C}$ with $f(x_j^{\text{cheb}}) = f_j$ for $0 \le j \le n$, then [133, Theorems 15.1 and 15.2]

$$\|f - Q_{G_{1,n}}f\|_{C^0([-1,1])} \le \left(2 + \frac{2}{\pi}\log(n+1)\right) \|f - Q^{\text{best}}_{G_{1,n}}f\|_{C^0([-1,1])}, \tag{3.3}$$

where $Q^{\text{best}}_{G_{1,n}} f$ is the best minimax polynomial[1] approximant to f of degree at most n. Chebyshev interpolation is quasi-optimal [24] for polynomial approximation since for any set of $n+1$ points on $[-1,1]$ there is a continuous function f such that

$$\|f - Q_{G_{1,n}} f\|_{C^0([-1,1])} \geq \left(1.52125 + \frac{2}{\pi}\log(n+1)\right) \|f - Q^{\text{best}}_{G_{1,n}} f\|_{C^0([-1,1])}.$$

The polynomial interpolant, $Q_{G_{1,n}} f$, can be represented as a Chebyshev series,

$$Q_{G_{1,n}} f(x) = \sum_{j=0}^{n} c_j T_j(x). \tag{3.4}$$

The coefficients $c_0, \ldots, c_n$ can be numerically computed from the data $f_0, \ldots, f_n$ in $O(n \log n)$ operations by the discrete Chebyshev transform, which is equivalent to the DCT-I (type-I discrete cosine transform) [54]. A further approximation to f is its Chebyshev projection, obtained by truncating the Chebyshev series after $n+1$ terms.

If f is Lipschitz continuous, then it admits an absolutely and uniformly convergent Chebyshev expansion [133, Theorem 3.1] given by

$$f(x) = \sum_{j=0}^{\infty} a_j T_j(x),$$

where the coefficients $\{a_j\}_{j\geq 0}$ are defined by the integrals

$$a_j = \frac{2}{\pi} \int_{-1}^{1} \frac{f(x) T_j(x)}{\sqrt{1-x^2}}\, dx, \quad j \geq 0. \tag{3.5}$$

and with $2/\pi$ replaced by $1/\pi$ in (3.5) if $j = 0$. The Chebyshev expansion for f can be truncated to construct $\mathcal{T}_{f,n}$ as

$$\mathcal{T}_{f,n}(x) = \sum_{j=0}^{n} a_j T_j(x). \tag{3.6}$$

The approximation errors $\|f - \mathcal{T}_{f,n}\|_{C^0([-1,1])}$ and $\|f - Q_{G_{1,n}} f\|_{C^0([-1,1])}$ decay at the same asymptotic rate since the coefficients $\{c_j\}_{j\geq 0}$ in equation (3.4) are related to $\{a_j\}_{j\geq 0}$ in equation (3.6) by an aliasing formula [133, Theorem 4.2]. As a general rule, Chebyshev projections tend to be more convenient for theoretical work, while Chebyshev interpolants are often faster to compute in practice.

[1] In one variable a continuous real-valued function defined on an interval has a best minimax polynomial approximation [105, Theorem 1.2], that is unique [105, Theorem 7.6]. In the multivariate case a continuous function $f : \Omega_d \to \mathbb{R}$ also has a best minimax polynomial approximation, but it is not guaranteed to be unique. Nonuniqueness is a direct consequence of Haar's Unicity Theorem [32, Section 5].

3.3 Convergence behavior in Chebyshev approximation

In the univariate case, it is well known that the approximation error in Chebyshev expansions decays algebraically for differentiable functions and exponentially for analytic functions. This behavior can be summarized by the following statements:

Theorem 37 (Convergence for differentiable functions)**.** *For an integer $r \geq 1$, let f and its derivatives through $f^{(r-1)}$ be absolutely continuous on $[-1,1]$ and suppose the rth derivative $f^{(r)}$ is of bounded total variation $V_{f,r}$. Then, for $n > r$,*

$$\|f - \mathcal{T}_{f,n}\|_{C^0([-1,1])} \leq \frac{2V_{f,r}}{\pi r(n-r)^r},$$

$$\|f - Q_{G_{1,n}} f\|_{C^0([-1,1])} \leq \frac{4V_{f,r}}{\pi r(n-r)^r}.$$

Proof. See [133, Theorem 7.2]. □

Theorem 38 (Convergence for analytic functions)**.** *Let a function f be analytic on $[-1,1]$ and analytically continuable to the open Bernstein ellipse $\mathcal{E}_\rho$, where it satisfies $|f| \leq M < \infty$. Then, for $n \geq 0$,*

$$\|f - \mathcal{T}_{f,n}\|_{C^0([-1,1])} \leq \frac{2M\rho^{-n}}{\rho - 1},$$

$$\|f - Q_{G_{1,n}} f\|_{C^0([-1,1])} \leq \frac{4M\rho^{-n}}{\rho - 1}.$$

Proof. See [133, Theorem 8.2]. □

Theorems 37 and 38 suggest that $Q_{G_{1,n}} f$ and $\mathcal{T}_{f,n}$ have essentially the same asymptotic approximation power.

Theorem 39 (Converse Theorem 38)**.** *Suppose f is a function on $[-1,1]$ for which there exist polynomial approximations $\{Q_{G_{1,n}} f\}$ satisfying*

$$\|f - Q_{G_{1,n}} f\|_{C^0([-1,1])} \leq M\rho^{-n}, \quad n \geq 0$$

for some constants $\rho > 1$ and $M > 0$. Then f can be analytically continued to an analytic function in the open Bernstein ellipse $\mathcal{E}_\rho$.

Proof. See [133, Theorem 8.3]. □

3.4 Advantages of Chebyshev interpolation

Chebyshev polynomials are closely related to the Fourier and Laurent bases. For $x \in [-1,1]$, define $\theta = \cos^{-1} x$ and $z = e^{i\theta}$. Then, for $j \geq 0$, the jth Chebyshev polynomial satisfies

$$T_j(x) = \Re\left(e^{ij\theta}\right) = \frac{z^j + z^{-j}}{2},$$

where $\Re$ denotes the real part of a complex function. It follows that the following relations hold:

$$\underbrace{\sum_{j=0}^{\infty} \alpha_j T_j(x)}_{\text{Chebyshev}} = \underbrace{\sum_{j=0}^{\infty} \alpha_j \Re\left(e^{ij\theta}\right)}_{\text{even Fourier}} = \alpha_0 + \underbrace{\sum_{\substack{j=-\infty \\ j\neq 0}}^{\infty} \frac{\alpha_{|j|}}{2} z^j}_{\text{Laurent}},$$

where the series are assumed to converge uniformly and absolutely.

As summarized in Table 3.1, a Chebyshev expansion of a function on $[-1,1]$ can be viewed as a Fourier series of an even function on $[-\pi,\pi]$. This can further be interpreted as a palindromic Laurent expansion on the unit circle for a function satisfying $f(z) = f(z^{-1})$, where z^{-1} denotes the complex inverse of z. Just as the Fourier basis is natural for representing periodic functions, Chebyshev polynomials are a natural basis for functions on $[-1,1]$.

Series	Assumptions	Setting	Interpolation points
Chebyshev	none	$x \in [-1,1]$	Chebyshev points
Fourier	$f(\theta) = f(-\theta)$	$\theta \in [-\pi,\pi]$	equispaced points
Laurent	$f(z) = f(z^{-1})$	$z \in$ unit circle	roots of unity

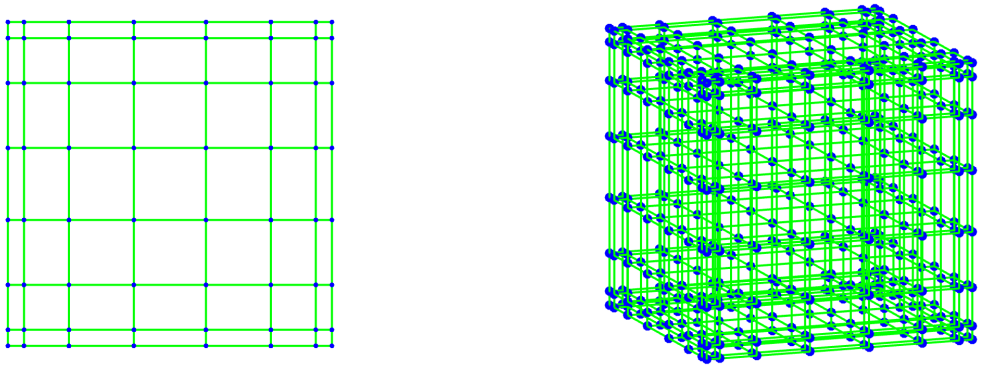
Table 3.1: Fourier, Chebyshev, and Laurent series are closely related. Each representation can be converted into the other by a change of variables. Under this transformation, Chebyshev points, equispaced points, and roots of unity are interconnected.

One reason these connections are important is that they enable the discrete Chebyshev transform, which converts $n+1$ values at Chebyshev points to the Chebyshev coefficients of $Q_{G_{1,n}} f$ in equation (3.4), to be computed via the fast Fourier transform (FFT) [54]. The scope of Chebyshev polynomial interpolation is not limited to univariate applications. A tensor-based extension can handle the multivariate case.

3.5 Multivariate Chebyshev approximation in the hypercube

In this section, we focus on interpolation over tensorial grids, which form the basis for defining multivariate interpolation operators. These grids facilitate the extension of univariate interpolation methods, such as Lagrange and Newton interpolation, to multidimensional settings.

Definition 40 (Interpolation grid)**.** *For numbers $d, n \in \mathbb{N}$ let $P_1, \ldots, P_d \subseteq [-1,1]$ be sets of size $|P_i| = n+1$ each. We call $G_{d,n} = \bigotimes_{i=1}^{d} P_i$ an interpolation grid. For any multi-index $\boldsymbol{\alpha} \in A_{d,n}$ we denote with $p_{\boldsymbol{\alpha}} = (p_{\alpha_1,1}, \ldots, p_{\alpha_d,d}) \in G_{d,n}$, $p_{\alpha_i,i} \in P_i$, the corresponding grid node of $G_{d,n}$.*



(a) $d = 2$ (b) $d = 3$

Figure 3.2: A visualization of Chebyshev–Lobatto points $\mathrm{Cheb}_{d,n}$ (blue dots) for degree $n = 8$ and $d = 2$ (left) and $d = 3$ (right).

We use such a grid to define the corresponding interpolation operator $Q_{G_{d,n}} : C^0(\Omega_d) \to \Pi_{d,n}$, $f \mapsto Q_{G_{d,n}} f$, uniquely determined by $Q_{G_{d,n}} f(p_{\boldsymbol{\alpha}}) = f(p_{\boldsymbol{\alpha}})$ for all $p_{\boldsymbol{\alpha}} \in G_{d,n}$. For an explicit representation, we generalize one-dimensional Lagrange and Newton interpolation to multivariate interpolation on the grids $G_{d,n}$ [36, 68, 69].

Definition 41 (Lagrange and Newton polynomials)**.** *Let $G_{d,n} = \bigotimes_{i=1}^{d} P_i$ be an interpolation grid indexed by a multi-index set $A_{d,n}$. For each $\boldsymbol{\alpha} \in A_{d,n}$ the tensorial multivariate Lagrange polynomial is*

$$L_{\boldsymbol{\alpha}}(x) = \prod_{i=1}^{d} l_{\alpha_i,i}(x), \quad l_{j,i}(x) = \prod_{k=0,k\neq j}^{n} \frac{x_i - p_{k,i}}{p_{j,i} - p_{k,i}}. \tag{3.7}$$

The $\boldsymbol{\alpha}$-th tensorial multivariate Newton polynomial is

$$N_{\boldsymbol{\alpha}}(x) = \prod_{i=1}^{d} \prod_{j=0}^{\alpha_i} (x_i - p_{j,i}), \quad p_{j,i} \in P_i. \tag{3.8}$$

Both the Lagrange and Newton polynomials form bases of the polynomial space $\Pi_{d,n}$ induced by $A_{d,n}$. As the $L_{\boldsymbol{\alpha}}$ satisfy $L_{\boldsymbol{\alpha}}(p_{\boldsymbol{\beta}}) = \delta_{\boldsymbol{\alpha},\boldsymbol{\beta}}$ for all $\boldsymbol{\alpha} \in A_{d,n}$, $p_{\boldsymbol{\beta}} \in G_{d,n}$ we deduce that given a function $f : \Omega_d \to \mathbb{R}$, the interpolant $Q_{G_{d,n}} f \in \Pi_{d,n}$ can be computed as

$$Q_{G_{d,n}} f = \sum_{\boldsymbol{\alpha} \in A_{d,n}} f(p_{\boldsymbol{\alpha}}) L_{\boldsymbol{\alpha}} = \sum_{\boldsymbol{\alpha} \in A_{d,n}} b_{\boldsymbol{\alpha}} N_{\boldsymbol{\alpha}}, \tag{3.9}$$

where the coefficients $b_{\boldsymbol{\alpha}} \in \mathbb{R}$ of the Newton interpolation can be computed in closed form. While Lagrange interpolation is primarily of theoretical interest, the Newton form allows efficient and stable evaluations of $Q_{G_{d,n}} f$ at any point $\mathbf{x} \in \Omega_d$. Alternatively, barycentric (tensorial) Lagrange interpolation provides efficient realisations [13].

The stability of polynomial interpolation is quantified by the *Lebesgue constant*, defined as the

operator norm of the interpolation operator

$$Q_{G_{d,n}} : C^0(\Omega_d) \to \Pi_{d,n}.$$

given by

$$\Lambda(G_{d,n}) := \|Q_{G_{d,n}}\|_{C^0(\Omega_d)} = \sup_{g \in C^0(\Omega_d)\setminus\{0\}} \frac{\|Q_{G_{d,n}} g\|_{C^0(\Omega_d)}}{\|g\|_{C^0(\Omega_d)}} = \Big\| \sum_{\boldsymbol{\alpha} \in A_{d,n}} |L_{\boldsymbol{\alpha}}| \Big\|_{C^0(\Omega_d)}. \tag{3.10}$$

Equivalently, $\Lambda(G_{d,n})$ measures how much interpolation can amplify perturbations in the data and the error of best polynomial approximation. In particular,

$$\|f - Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \leq \big(1 + \Lambda(G_{d,n})\big) \inf_{p \in \Pi_{d,n}} \|f - p\|_{C^0(\Omega_d)}.$$

Thus, small Lebesgue constants are desirable, since they imply that the interpolation error remains close to the best achievable polynomial approximation error. In the case of a one-dimensional interpolation domain $[-1,1]$ and the Chebyshev–Lobatto grid (3.2), the Lebesgue constant $\Lambda(\mathrm{Cheb}_n)$ increases slowly as $n \to \infty$,

$$\Lambda(\mathrm{Cheb}_n) = \frac{2}{\pi}\big(\log(n+1) + \gamma + \log(8/\pi)\big) + \mathcal{O}(1/n^2), \tag{3.11}$$

where $\gamma \approx 0.5772$ is the Euler–Mascheroni constant, see [12], surveyed by [25], see also [133]. We extend this estimate to the d-dimensional case:

Theorem 42. *The Lebesgue constant of the d-dimensional Chebyshev–Lobatto grid*

$$\mathrm{Cheb}_{d,n} = \bigotimes_{i=1}^{d} \mathrm{Cheb}_n$$

satisfies

$$\Lambda(\mathrm{Cheb}_{d,n}) = \Lambda(\mathrm{Cheb}_n)^d \in \mathcal{O}\big(\log(n+1)^d\big).$$

Proof. We consider the tensorial Lagrange polynomials $L_{\boldsymbol{\alpha}}(\mathrm{x}) = \prod_{i=1}^{d} l_{\alpha_i,i}(x_i)$ in the Chebyshev–Lobatto nodes, with $l_{j,i}$ given in (3.7) and obtain

$$\begin{aligned}
\Lambda(\mathrm{Cheb}_{d,n}) &= \left\| \Big\| \sum_{\boldsymbol{\alpha} \in A_{d,n}} |L_{\boldsymbol{\alpha}}| \Big\|_{C^0(\square_d)} \right\| \leq \Big\| \sum_{\boldsymbol{\alpha} \in A_{d,n}} \prod_{i=1}^{d} |l_{\alpha_i,i}| \Big\|_{C^0(\square_d)} \\
&= \Big\| \big(\sum_{j=0}^{n} |l_{j,1}|\big) \cdots \big(\sum_{j=0}^{n} |l_{j,l}|\big) \ \cdots \big(\sum_{j=0}^{n} |l_{j,d}|\big) \Big\|_{C^0(\square_d)}, \quad 1 < l < d \\
&\leq \prod_{i=1}^{d} \Big\| \sum_{j=0}^{n} |l_{j,i}| \Big\|_{C^0(\square_d)} = \prod_{i=1}^{d} \Lambda(\mathrm{Cheb}_n).
\end{aligned} \tag{3.12}$$

With (3.11), this yields $\Lambda(\mathrm{Cheb}_{d,n}) \leq \Lambda(\mathrm{Cheb}_n)^d \in \mathcal{O}(\log(n+1)^d)$. □

Moreover, using Markov brothers' inequality [121], we can establish a bound on the Lebesgue constant in the $W^{k,\infty}$-norm. Beginning with the $W^{1,\infty}$-norm, we apply an iterative approach to extend the result.

$$\begin{aligned}\|Q_{\mathrm{Cheb}_{d,n}}\|_{W^{1,\infty}(\Omega_d)} &= \sup_{g\in W^{1,\infty}(\Omega_d)\setminus\{0\}} \frac{\|Q_{\mathrm{Cheb}_{d,n}}g\|_{W^{1,\infty}(\Omega_d)}}{\|g\|_{W^{1,\infty}(\Omega_d)}} \\ &\leq \sup_{g\in W^{1,\infty}(\Omega_d)\setminus\{0\}} \frac{n^2\|Q_{\mathrm{Cheb}_{d,n}}g\|_{C^0(\Omega_d)}}{\|g\|_{W^{1,\infty}(\Omega_d)}} \\ &\leq \sup_{g\in W^{1,\infty}(\Omega_d)\setminus\{0\}} \frac{n^2\|Q_{\mathrm{Cheb}_{d,n}}\|_{C^0(\Omega_d)}\|g\|_{C^0(\Omega_d)}}{\|g\|_{W^{1,\infty}(\Omega_d)}} \leq n^2\|Q_{\mathrm{Cheb}_{d,n}}\|_{C^0(\Omega_d)}\,.\end{aligned}$$

In the third step, we used that for all $p\in\Pi_{d,n}$

$$\begin{aligned}\max_{i=1}^{d}\left\|\frac{\partial}{\partial x_i}p\right\|_{C^0(\Omega_d)} &= \max_{i=1}^{d}\max_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_d\in\Omega_d}\left\|\left(\frac{\partial}{\partial x_i}p\right)(x_1,\ldots,x_{i-1},\cdot,x_{i+1},\ldots,x_d)\right\|_{C^0([-1,1])} &(3.13)\\ &\leq \max_{i=1}^{d}\max_{x_1,\ldots,x_{i-1},x_{i+1},\ldots,x_d\in\Omega_d} n^2\,\|p(x_1,\ldots,x_{i-1},\cdot,x_{i+1},\ldots,x_d)\|_{C^0([-1,1])} &(3.14)\\ &= n^2\|p\|_{C^0(\Omega_d)}\,. &(3.15)\end{aligned}$$

The above argument can be iterated and obtain the following:

Lemma 43. *The Lebesgue constant of the d-dimensional Chebyshev–Lobatto grid in the Sobolev space $W^{k,\infty}(\Omega_d)$ satisfies the bound*

$$\|Q_{\mathrm{Cheb}_{d,n}}\|_{W^{k,\infty}(\Omega_d)} \leq n^{2k}\|Q_{\mathrm{Cheb}_{d,n}}\|_{C^0(\Omega_d)}\,.$$

Let $f:\Omega_d\to\mathbb{R}$ be a Lipschitz continuous function defined on the hypercube Ω_d. According to [93, Theorem 4.1], any Lipschitz continuous function f admits a uniformly and absolutely convergent multivariate Chebyshev series:

$$f(\mathbf{x}) = \sum_{\boldsymbol{\alpha}\in\mathbb{N}_0^d} c_{\boldsymbol{\alpha}}T_{\boldsymbol{\alpha}}(\mathbf{x}), \tag{3.16}$$

where $T_{\boldsymbol{\alpha}}(\mathbf{x}) = \prod_{i=1}^{d}T_{\alpha_i}(x_i)$ denotes the tensor product of univariate Chebyshev polynomials. The Chebyshev polynomials $T_{\boldsymbol{\alpha}}(\mathbf{x})$ form an orthogonal basis of $L^2(\Omega_d)$ with respect to the weighted L^2 inner product (3.1), where the weight function is

$$\omega_d(\mathbf{x}) = \prod_{i=1}^{d}\frac{1}{\sqrt{1-x_i^2}}.$$

The Chebyshev coefficients $c_{\boldsymbol{\alpha}}$ are given by

$$c_{\boldsymbol{\alpha}} = \frac{2^d}{\pi^d}\int_{\Omega_d} f(\mathbf{x})T_{\boldsymbol{\alpha}}(\mathbf{x})\omega_d(\mathbf{x})\,d\mathbf{x}, \tag{3.17}$$

for all $\boldsymbol{\alpha} \in \mathbb{N}^d$ with $\alpha_1, \dots, \alpha_d \geq 1$, and with each factor $2/\pi$ replaced by $1/\pi$ in (3.17) if $\alpha_i = 0$ for some $1 \leq i \leq d$. Here, $d\mathbf{x} = \prod_{i=1}^d dx_i$.

3.6 Convergence for differentiable functions

In what follows, we provide an estimate for the Chebyshev coefficients $c_{\boldsymbol{\alpha}}$ in terms of the Chebyshev-weighted r^{th}-order directional variation $V_{f,r}$ from Definition 34.

Theorem 44. *Let $d \in \mathbb{N}$, $r \geq 0$, and f be of bounded r^{th} total variation. Then f can be expanded in a Chebyshev series*

$$f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d} c_{\boldsymbol{\alpha}} T_{\boldsymbol{\alpha}}(\mathbf{x}), \tag{3.18}$$

$$\text{with} \quad |c_{\boldsymbol{\alpha}}| \leq V_{f,r} \Big(\frac{2}{\pi q(q-1)\dots(q-r)} \Big)^d, \tag{3.19}$$

whenever $q = \min_{i=1,\dots,d} \alpha_i \geq r+1$.

Proof. By following the argumentation in 1D, Theorems 7.1, 7.2 in [133], the coefficients are bounded by

$$|c_{\boldsymbol{\alpha}}| \leq \Big(\frac{2}{\pi q(q-1)\dots(q-r)} \Big)^d \int_{\Omega_d} |\partial^{\boldsymbol{\beta}} f(\mathbf{x})| d\mathbf{x}, \tag{3.20}$$

where $\boldsymbol{\beta} = (r+1, \dots, r+1)$ and $q = \min_{i=1,\dots,d} \alpha_i \geq r+1$. Consequently, by Definition 34, the estimate

$$|c_{\boldsymbol{\alpha}}| \leq V_{f,r} \Big(\frac{2}{\pi q(q-1)\dots(q-r)} \Big)^d,$$

applies. □

We use this result in order to control the truncation error of the Chebyshev series.

Theorem 45. *Let the assumptions of Theorem 44 be fulfilled. We denote with*

$$\mathcal{T}_{f,n}(\mathrm{x}) = \sum_{\boldsymbol{\alpha} \in A_{d,n}} c_{\boldsymbol{\alpha}} T_{\boldsymbol{\alpha}}(\mathbf{x}), \tag{3.21}$$

the truncated Chebyshev series of $f : \Omega_d \to \mathbb{R}$ with respect to $A_{d,n}$, with $n > r$.

i) The truncation error is bounded by

$$\|f - \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} \leq \frac{2ed^2 V_{f,r}}{\pi(r-d+1)} \left(\frac{n+1}{n+1-r} \right)^{r+1} \cdot \frac{1}{n^{r-d+1}} \in \mathcal{O}\big(n^{-(r-d+1)}\big), \tag{3.22}$$

$r > d-1$.

ii) The truncation error of the first-order partial derivatives is bounded by

$$\|\partial_{x_i} f - \partial_{x_i} \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} \leq \frac{2ed^2 V_{f,r}}{\pi(r-d-1)} \left(\frac{n+1}{n+1-r} \right)^{r+1} \cdot \frac{1}{n^{r-d-1}} \in \mathcal{O}\big(n^{-(r-d-1)}\big), \tag{3.23}$$

$r > d, \forall i = 1, \dots, d.$

Proof. Since $T_k(\cos(x)) = \cos(kx)$ for all $k \in \mathbb{N}$, we observe that $\|T_k\|_{C^0([-1,1])} \leq 1$. Let $\Lambda_n^{(i)} = \{\boldsymbol{\alpha} \in \mathbb{N}^d : \alpha_i > n\}$. Then the truncation error admits the following bound:

$$\begin{aligned}
\|f - \mathcal{T}_{f,n}\|_{C^0(\square_d)} &\leq \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d \setminus A_{d,n}} c_{\boldsymbol{\alpha}} \|T_{\alpha_1}(x_1) \cdots T_{\alpha_d}(x_d)\|_{C^0(\Omega_d)} \leq \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d \setminus A_{d,n}} |c_{\boldsymbol{\alpha}}| \\
&\leq \sum_{i=1}^{d} \sum_{\boldsymbol{\alpha} \in \Lambda_n^{(i)}} \frac{2V_{f,r}}{\pi \alpha_i (\alpha_i - 1) \dots (\alpha_i - r)} \leq \sum_{i=1}^{d} \sum_{\boldsymbol{\alpha} \in \Lambda_n^{(i)}} \frac{2V_{f,r}}{\pi (\alpha_i - r)^{r+1}} \\
&= \frac{2dV_{f,r}}{\pi} \sum_{q=n+1}^{\infty} \frac{|A_{d,q} \setminus A_{d,q-1}|}{(q-r)^{r+1}} \leq \frac{2ed^2 V_{f,r}}{\pi} \sum_{q=n+1}^{\infty} \frac{q^{d-1}}{(q-r)^{r+1}} \\
&\leq \frac{2ed^2 V_{f,r}}{\pi} \left(\frac{n+1}{n+1-r} \right)^{r+1} \sum_{q=n+1}^{\infty} \frac{1}{q^{r-d+2}} \leq \frac{2ed^2 V_{f,r}}{\pi} \left(\frac{n+1}{n+1-r} \right)^{r+1} \int_n^{\infty} \frac{1}{x^{r-d+2}} dx \\
&= \frac{2ed^2 V_{f,r}}{\pi (r-d+1)} \left(\frac{n+1}{n+1-r} \right)^{r+1} \cdot \frac{1}{n^{r-d+1}}
\end{aligned}$$

where for each $\boldsymbol{\alpha} \in \Lambda_n^{(i)}$, we apply the Chebyshev coefficient bound (3.19) in the i-th coordinate only, assuming $q := \alpha_i \geq r+1$. Moreover,we used the estimate $|A_{d,q} \setminus A_{d,q-1}| = (q+1)^d - q^d \leq edq^{d-1}$, $(q > d)$, and bounded the resulting monotonically decreasing sum (for $r - d + 2 > 1$), which ultimately yields the desired estimate (3.22). We show ii) for the partial derivative ∂_{x_i} by writing

$$\left\| \partial_{x_i} f - \partial_{x_i} \mathcal{T}_{f,n} \right\|_{C^0(\Omega_d)} \leq \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d \setminus A_{d,n}} |c_{\boldsymbol{\alpha}}| \left\| T_{\alpha_1} \cdots T_{\alpha_{i-1}} \right\|_{C^0(\Omega_d)} \left\| T'_{\alpha_i} \right\|_{C^0(\Omega_d)} \left\| T_{\alpha_{i+1}} \cdots T_{\alpha_d} \right\|_{C^0(\Omega_d)}.$$

We recall that $T_k(x) = \cos(k \arccos(x))$ for $-1 \leq x \leq 1$ and deduce that for all $k \in \mathbb{N}$

$$T'_k(x) = \frac{k \sin(k \arccos(x))}{\sqrt{1-x^2}} = \frac{k \sin(kt)}{\sin(t)}, \qquad t = \arccos(x), \tag{3.24}$$

yielding $\|T'_{\alpha_i}\|_{C^0(\Omega_d)} = \alpha_i^2$. Following i), we compute

$$\begin{aligned}
\left\| \partial_{x_i} f - \partial_{x_i} \mathcal{T}_{f,n} \right\|_{C^0(\Omega_d)} &\leq \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d \setminus A_{d,n}} |c_{\boldsymbol{\alpha}}| \alpha_i^2 \leq \sum_{i=1}^{d} \sum_{\boldsymbol{\alpha} \in \Lambda_n^{(i)}} \frac{2V_{f,r}}{\pi (\alpha_i - r)^{r+1}} \alpha_i^2 \\
&\leq \frac{2ed^2 V_{f,r}}{\pi} \sum_{q=n+1}^{\infty} \frac{q^{d-1} q^2}{(q-r)^{r+1}} \leq \frac{2ed^2 V_{f,r}}{\pi} \left(\frac{n+1}{n+1-r} \right)^{r+1} \sum_{q=n+1}^{\infty} \frac{1}{q^{r-d}} \\
&\leq \frac{2ed^2 V_{f,r}}{\pi} \left(\frac{n+1}{n+1-r} \right)^{r+1} \int_n^{\infty} \frac{1}{x^{r-d}} = \frac{2ed^2 V_{f,r}}{\pi (r-d-1)} \left(\frac{n+1}{n+1-r} \right)^{r+1} \cdot \frac{1}{n^{r-d-1}},
\end{aligned} \tag{3.25}$$

by bounding the monotonically decreasing sum (for $r - d > 1$) in (3.25). □

Remark 46. *The coefficients $\mathcal{F}_{f,n}$ of the two dimensional truncated Chebyshev expansion can be computed*

in $O(n^2 \log n)$ operations using the fast cosine transform (see, for instance, [73]).

With the previous results, we can bound the approximation error of the Chebyshev–Lobatto interpolant of f.

Corollary 47. *Let the assumption of Theorem 44 be satisfied and $Q_{G_{d,n}} f$ be the interpolant of $f : \Omega_d \to \mathbb{R}$ in the Chebyshev–Lobatto grid $\mathrm{Cheb}_{d,n}$. Then the approximation errors of f and its first derivatives are bounded by*

$$\|f - Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \leq \frac{4ed^2 V_{f,r}}{\pi(r-d+1)} \left(\frac{n+1}{n+1-r}\right)^{r+1} \cdot \frac{1}{n^{r-d+1}} \in \mathcal{O}\left(n^{-(r-d+1)}\right). \tag{3.26}$$

and

$$\|\partial_{x_i} f - \partial_{x_i} Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \leq \frac{4ed^2 V_{f,r}}{\pi(r-d-1)} \left(\frac{n+1}{n+1-r}\right)^{r+1} \cdot \frac{1}{n^{r-d-1}} \in \mathcal{O}\left(n^{-(r-d-1)}\right) \tag{3.27}$$

for all $i = 1, \ldots, d$.

Proof. The statement is a direct consequence of Theorem 44 and the [133, Aliasing Theorems 4.1, 4.2], stating that

$$f(\mathbf{x}) - Q_{G_{d,n}} f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}^d \setminus A_{d,n}} c_{\boldsymbol{\alpha}} \left(T_{\alpha_1}(x_1) \cdots T_{\alpha_d}(x_d) - T_{\beta_1}(x_1) \cdots T_{\beta_d}(x_d)\right), \tag{3.28}$$

where $\beta_i = |(\alpha_i + n - 1) \mathrm{mod} 2n - (n-1)|$. This shows that, when following the estimation in Theorem 45, the approximation error of the interpolant can be bounded by twice the bound that appears for the truncation. □

Note that these bounds for tensorized Chebyshev approximation are slightly better than those obtained by [93, (21) and (22)]. In particular, tensorized Chebyshev approximation yields quasi-optimal approximations of continuous functions by bounded-degree polynomials [93].

3.7 Convergence for analytic functions

We now present the analogue of the previous theorems in the case where the function f is analytic (see Definition 19). To deal with the higher dimensional case $d > 1$, an essential issue here is to extend the Bernstein ellipse to a region in $\mathbb{C}^d$. A natural extension of the Bernstein ellipse $\mathcal{E}_\rho$ to $\mathbb{C}^d$ is the polyellipse. To formalize our analysis, we begin with the following assumption on f.

Assumption 48. *Let $\boldsymbol{\rho} = (\rho_1, \ldots, \rho_d)$, with $\rho_i > 1$ for each i. The function f is analytic in the polyellipse*

$$\mathcal{E}_{\boldsymbol{\rho}} := \bigotimes_{j=1}^{d} \mathcal{E}_{\rho_j}, \tag{3.29}$$

where the one-dimensional Bernstein ellipse $\mathcal{E}_{\rho_j}$ is defined in equation (2.9). Moreover,

$$M := \sup_{\mathbf{z}\in\mathcal{E}_{\boldsymbol{\rho}}} |f(\mathbf{z})| < \infty.$$

The assumption means that f admits a bounded analytic continuation to a tensor product of Bernstein ellipses. In the estimates below we also use the scalar quantity

$$\rho := \min_{1\le i\le d} \rho_i > 1.$$

Theorem 49. *Under Assumption 48, the Chebyshev coefficients (3.18) of f satisfy*

$$|c_{\boldsymbol{\alpha}}| \le 2^d M \prod_{i=1}^{d} \rho_i^{-\alpha_i}, \qquad \boldsymbol{\alpha} \in \mathbb{N}_0^d. \tag{3.30}$$

Consequently, if $q = \|\boldsymbol{\alpha}\|_\infty$, then

$$|c_{\boldsymbol{\alpha}}| \le 2^d M \rho^{-q}.$$

Proof. The tensor-product contour argument for Chebyshev coefficients gives the product estimate (3.30); equivalently, one applies the one-dimensional analytic coefficient estimate successively in each variable, using the bound M on $\mathcal{E}_{\boldsymbol{\rho}}$. Since $\rho_i \ge \rho$ and $\sum_i \alpha_i \ge \|\boldsymbol{\alpha}\|_\infty$, we have

$$\prod_{i=1}^{d} \rho_i^{-\alpha_i} \le \rho^{-\sum_i \alpha_i} \le \rho^{-\|\boldsymbol{\alpha}\|_\infty},$$

which proves the final estimate. □

Before proceeding with the truncation error estimate, we require the following elementary tail bound.

Lemma 50. *Let $\lambda, \nu > 0$ and $N > 0$. Then*

$$\int_N^\infty e^{-\lambda x} x^\nu \, dx \le e^{-\lambda N} \left(\sum_{j=1}^{m_\nu+1} \frac{N^{\nu-j+1}}{\lambda^j} \prod_{i=0}^{j-2} (\nu - i) \right), \tag{3.31}$$

where $m_\nu \in \mathbb{N}$ is given by

$$m_\nu = \begin{cases} \nu, & \text{if } \nu \in \mathbb{N}, \\ \lfloor \nu \rfloor + 1, & \text{if } \nu \notin \mathbb{N}, \end{cases} \tag{3.32}$$

and the empty product is taken to be one when $j < 2$.

Proof. Let m_ν be defined as in (3.32). Integrating by parts m_ν times yields

$$\int_N^\infty e^{-\lambda x} x^\nu \, dx = e^{-\lambda N} \sum_{j=1}^{m_\nu} \frac{N^{\nu-j+1}}{\lambda^j} \prod_{i=0}^{j-2} (\nu - i) + \frac{1}{\lambda^{m_\nu}} \prod_{i=0}^{m_\nu-1} (\nu - i) \int_N^\infty x^{\nu-m_\nu} e^{-\lambda x} \, dx. \tag{3.33}$$

Since $-1 < \nu - m_\nu \le 0$, the final integral satisfies

$$\int_N^\infty x^{\nu-m_\nu} e^{-\lambda x}\,dx \le N^{\nu-m_\nu} \int_N^\infty e^{-\lambda x}\,dx = \frac{N^{\nu-m_\nu}}{\lambda} e^{-\lambda N}. \tag{3.34}$$

Combining (3.33) and (3.34) completes the proof. □

For compactness, define the rescaled exponential tail

$$B_\nu(n,\rho) := \rho^n \sum_{q=n+1}^{\infty} q^\nu \rho^{-q}.$$

For fixed ν and $\rho > 1$, Lemma 50, together with a standard sum–integral comparison, implies $B_\nu(n,\rho) = \mathcal{O}(n^\nu)$.

Theorem 51. *Let the assumptions of Theorem 49 hold. Then:*

i) The truncation error satisfies

$$\|f - \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} \le 2^d e d M \rho^{-n} B_{d-1}(n,\rho). \tag{3.35}$$

ii) The error in the first-order partial derivatives satisfies

$$\|\partial_{x_i} f - \partial_{x_i} \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} \le 2^d e d M \rho^{-n} B_{d+1}(n,\rho), \quad i = 1,\dots,d. \tag{3.36}$$

Proof. Using $\|T_{\boldsymbol{\alpha}}\|_{C^0(\Omega_d)} \le 1$ and Theorem 49, we obtain

$$\begin{aligned}
\|f - \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} &\le \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d \setminus A_{d,n}} |c_{\boldsymbol{\alpha}}| \\
&\le 2^d M \sum_{q=n+1}^{\infty} |A_{d,q} \setminus A_{d,q-1}| \rho^{-q} \\
&\le 2^d e d M \sum_{q=n+1}^{\infty} q^{d-1} \rho^{-q}.
\end{aligned}$$

By the definition of $B_{d-1}(n,\rho)$, this gives (3.35). For the derivative estimate, use $\|T_k'\|_{C^0([-1,1])} = k^2$ and $\alpha_i \le \|\boldsymbol{\alpha}\|_\infty = q$. Then

$$\begin{aligned}
\|\partial_{x_i} f - \partial_{x_i} \mathcal{T}_{f,n}\|_{C^0(\Omega_d)} &\le \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d \setminus A_{d,n}} |c_{\boldsymbol{\alpha}}| \alpha_i^2 \\
&\le 2^d e d M \sum_{q=n+1}^{\infty} q^{d+1} \rho^{-q}.
\end{aligned}$$

By the definition of $B_{d+1}(n,\rho)$, this gives (3.36). □

Applying Corollary 51 and the Chebyshev–Lobatto aliasing formulas [131, Theorems 4.1 and 4.2], we obtain bounds on the approximation error of the Chebyshev–Lobatto interpolation of f.

Corollary 52. *Let the assumptions of Theorem 49 be satisfied, and let $Q_{G_{d,n}} f$ be the interpolant of $f : \Omega_d \to \mathbb{R}$ on the Chebyshev–Lobatto grid* $\mathrm{Cheb}_{d,n}$. *Then*

$$\|f - Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \leq 2^{d+1} e d M \rho^{-n} B_{d-1}(n, \rho), \tag{3.37}$$

and, for all $i = 1, \ldots, d$,

$$\|\partial_{x_i} f - \partial_{x_i} Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \leq 2^{d+1} e d M \rho^{-n} B_{d+1}(n, \rho). \tag{3.38}$$

Proof. The proof is the same as for Corollary 47: the Chebyshev–Lobatto aliasing formulas bound the interpolation error by twice the corresponding truncation error in Theorem 51. □

3.7.1 Multivariate Chebyshev approximation from analytic maps

Chebyshev grids exhibit strong clustering near the endpoints of the interval as illustrated in Figure 3.1. In particular, the spacing between adjacent nodes near the boundaries scales like $1/n^2$, where n denotes the degree of the interpolating polynomial. In contrast, the spacing near the centre of the interval is coarser—approximately a factor of $\pi/2$ larger than that of an equally spaced grid used in trigonometric interpolation [133].

From the perspective of polynomial approximation, this clustering is not only necessary but, in some respects, also optimal. However, it entails a cost: roughly a factor of $\pi/2$ in each spatial dimension is effectively "wasted." Through analytic transplantation [64], one can construct approximations that are up to a factor of $\pi/2$ more efficient. In a three-dimensional calculation, a discretization may need $(\pi/2)^3 \approx 4$ times as many grid points as one would like. Such a factor can significantly affect the cost of linear algebra operations. Various authors have proposed methods to mitigate this effect. Notably, it has been shown in [64, 128] that by using appropriate analytic maps from the ellipse $\mathcal{E}_\rho$ to more regular regions with straighter sides (see Figure 3.4), and applying the same numerical methods in this newly mapped Chebyshev-Lobatto points, the $\pi/2$ factor in convergence rates and node distributions can be recovered. In the remainder of this subsection, we briefly present the idea of analytic transplantation. For a detailed exposition of its construction, we refer the reader to [64, 126, 128, 131] and the references therein.

Let $f : \Omega_d \to \mathbb{R}$ be a Lipschitz continuous function defined on the physical hypercube, and let ζ be an analytic map[2]

$$\zeta : \Omega_d \to \Omega_d, \qquad \mathbf{x} = \zeta(\mathbf{s}), \tag{3.39}$$

with $\partial_{s_i} \zeta(\mathbf{s}) > 0$ for $1 \leq i \leq d$. Define the pullback

$$g(\mathbf{s}) := f(\zeta(\mathbf{s})).$$

We approximate g by a Chebyshev interpolant on the parameter domain and then map the result

[2] Continuous differentiability of ζ is sufficient for the algebraic approximation estimates. Analyticity is used when applying the polyellipse estimates. We use $\mathbf{x}$ for the physical domain and $\mathbf{s}$ for the parameter domain where the Chebyshev points are located.

back to the physical domain; that is, the approximation to f is

$$\mathbf{x} \mapsto \big(Q_{G_{d,n}} g\big)(\zeta^{-1}(\mathbf{x})).$$

According to [93, Theorem 4.1], the pullback g admits a uniformly and absolutely convergent multivariate Chebyshev series,

$$g(\mathbf{s}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d} c_{\boldsymbol{\alpha}} T_{\boldsymbol{\alpha}}(\mathbf{s}), \tag{3.40}$$

or equivalently

$$f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d} c_{\boldsymbol{\alpha}} T_{\boldsymbol{\alpha}}(\zeta^{-1}(\mathbf{x})),$$

with coefficients

$$c_{\boldsymbol{\alpha}} = \frac{2^d}{\pi^d} \int_{\Omega_d} f(\zeta(\mathbf{s})) \prod_{i=1}^{d} \frac{T_{\alpha_i}(s_i)}{\sqrt{1-s_i^2}}\, d\mathbf{s},$$

with $2/\pi$ replaced by $1/\pi$ for any $\alpha_i = 0$.

The theorems above apply to the transformed approximation after replacing f by the pullback $g = f \circ \zeta$. The analytic interpolation estimate becomes the following.

Theorem 53. *Assume that $g = f \circ \zeta$ satisfies Assumption 48 on $\mathcal{E}_{\boldsymbol{\rho}}$, with*

$$M_\zeta := \sup_{\mathbf{z} \in \mathcal{E}_{\boldsymbol{\rho}}} |f(\zeta(\mathbf{z}))| < \infty, \qquad \rho := \min_i \rho_i.$$

Let $Q_{G_{d,n}} g$ be the Chebyshev–Lobatto interpolant of g on $\mathrm{Cheb}_{d,n}$. Then

$$\|f - (Q_{G_{d,n}} g) \circ \zeta^{-1}\|_{C^0(\Omega_d)} \le 2^{d+1} e d\, M_\zeta \rho^{-n} B_{d-1}(n,\rho), \tag{3.41}$$

and, for all $i = 1, \dots, d$,

$$\|\partial_{s_i} g - \partial_{s_i} Q_{G_{d,n}} g\|_{C^0(\Omega_d)} \le 2^{d+1} e d\, M_\zeta \rho^{-n} B_{d+1}(n,\rho). \tag{3.42}$$

For many functions f, the transformed Chebyshev series are more efficient than the standard ones. To approximate such functions efficiently, we employ transformed Chebyshev interpolants using the mapping $\zeta(\boldsymbol{s})$ introduced in [64]:

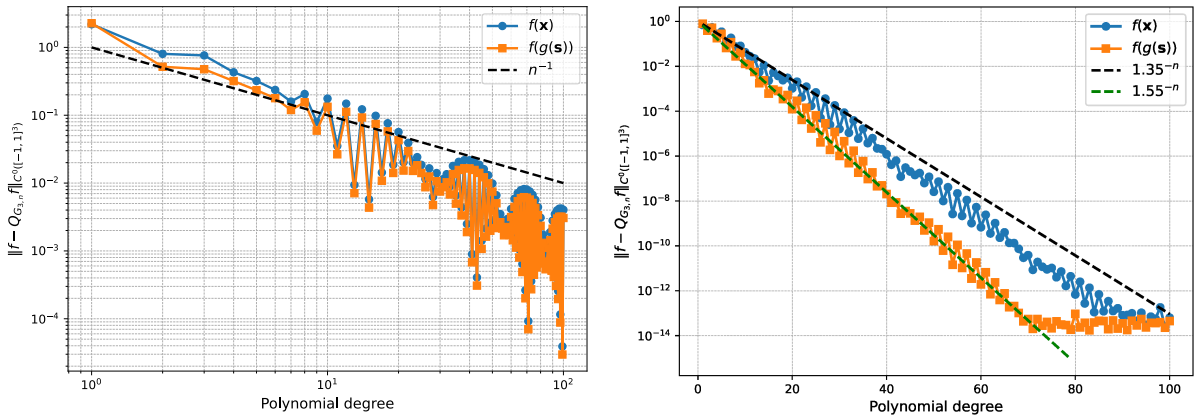
$$\zeta(\boldsymbol{s}) = \frac{1}{53089} \left(40320\boldsymbol{s} + 6720\boldsymbol{s}^3 + 3024\boldsymbol{s}^5 + 1800\boldsymbol{s}^7 + 1225\boldsymbol{s}^9\right). \tag{3.43}$$

This map is constructed by truncating the Taylor series of $\arcsin(s)$[3], then normalised to satisfy $\zeta(\pm 1) = \pm 1$. For example, choosing $n = 1, 3$, or 5 yields the following normalised polynomial maps:

[3] $\arcsin(s) = s + \frac{1}{6}s^3 + \frac{3}{40}s^5 + \frac{5}{112}s^7 + \frac{35}{1152}s^9 + \dots$

$$\zeta(\mathbf{s}) = \mathbf{s},$$
$$\zeta(\mathbf{s}) = \frac{1}{7}(6\mathbf{s} + \mathbf{s}^3),$$
$$\zeta(\mathbf{s}) = \frac{1}{149}(120\mathbf{s} + 20\mathbf{s}^3 + 9\mathbf{s}^5).$$

Maps of this form are polynomials (for a finite degree n) and are therefore analytic across the entire complex plane. Although these polynomial maps only approximate the arcsin function up to a given degree n, they preserve important characteristics—such as stretching the region of analyticity and equalising the distribution of nodes—without introducing singularities. To demonstrate this, we consider the interpolation of the functions defined in the following example.



(a) $f(\mathbf{x}) \in C^2([-1,1]^3)$ (b) $f(\mathbf{x})$ analytic

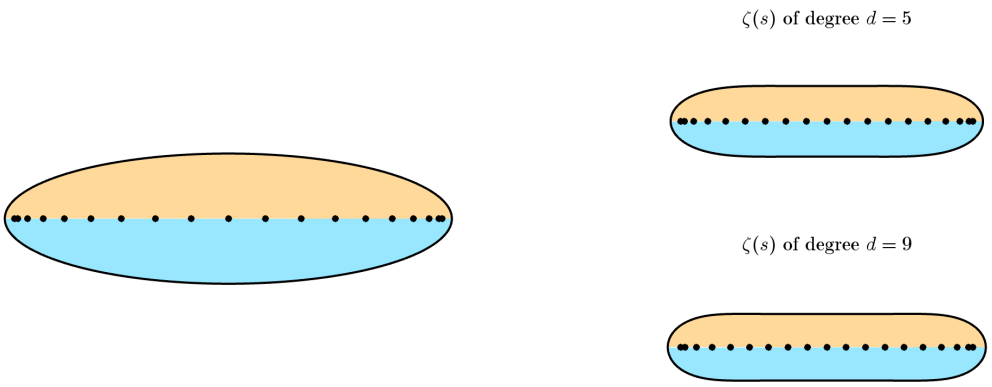
Figure 3.3: Approximation errors using transformed Chebyshev interpolants along with the ideal convergence lines.

Example 54. *We consider the interpolation of the functions* $f(\mathbf{x}) = \sum_{i=1}^{d} |\sin(x_i)|^3$ *and* $f(\mathbf{x}) = \frac{1}{1+10\|\mathbf{x}\|^2}$. *The first function belongs to the class* $C^2(\Omega_d)$, *with discontinuities in the third derivative at* $\mathbf{x} = 0$ *and* $\mathbf{x} = \pm\pi$, *whereas the second function is analytic over* Ω_d. *For the purpose of this analysis, we fix* $d = 3$ *and apply the analytic map given in* (3.43). *For the first function, we take* $r = 3$.

The results confirm the theoretical predictions:

- *The first function exhibits algebraic convergence, as shown in the left plot of Figure 3.4, where the transformed approximants decay at the same rate as the Chebyshev interpolants but lie consistently below them.*
- *The second function demonstrates geometric (exponential) convergence, as illustrated in the right plot of Figure 3.4. The speedup is clear, the poles at* $\pm\frac{i}{\sqrt{10}}$ *make the largest* $\mathcal{E}_\rho$*-ellipse in which* f *is analytic small. The analytic map* (3.43) *enlarges the Bernstein ellipse in which* $f(\zeta(\boldsymbol{s}))$ *is analytic. As a result, the transformed approximants converge at a geometric rate, with an observed decay of approximately* 1.55^{-n}. *It is also worth noting that the interpolation points in Figure 3.4b are more evenly distributed than in Figure 3.4a, with increased density near the center, allowing for finer*

resolution.



(a) $\mathcal{E}_{1.35}$ (b) $\zeta(\mathcal{E}_{1.35})$

Figure 3.4: The Bernstein ellipse together with the Chebyshev–Lobatto points, and their image under the transformation (3.43).

Example 54 illustrates and confirms the findings of [64], namely that the assumption that f is analytic in $\mathcal{E}_\rho$ is unbalanced from a practical point of view. It allows f to be "less analytic" near the ends of the interval, where the ellipse is narrow, than in the middle, where it is wide. This nonuniform analyticity condition underlies Corollary 52 and related results, but it lacks intrinsic justification. Therefore, by analytically mapping $\mathcal{E}_\rho$ to a region with straighter sides, one can achieve improved convergence rates. We close by emphasizing that, for analytic functions, it is not enough for a grid to simply 'look good'; achieving a fast rate of geometric convergence depends on whether it originates from an analytic map covering a large domain of analyticity.

Remark 55. *Before concluding this section, it is essential to address one of the key challenges of tensorial polynomial interpolation: the so-called curse of dimensionality* [74]. *This term, introduced by Richard Bellman* [10], *refers to the exponential increase in complexity as the number of variables (or dimensions) grows. In our context, this phenomenon manifests as follows: If we express the convergence rates in terms of the number of degrees of freedom (Chebyshev coefficients),* $N := |A_{d,n}| = (n+1)^d$, *then equation* (3.22) *becomes*

$$\|f - Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \in \mathcal{O}\big(N^{-\frac{(r-d+1)}{d}}\big). \tag{3.44}$$

Similarly, for equation (3.23) *we have*

$$\|\partial_{x_i} f - \partial_{x_i} Q_{G_{d,n}} f\|_{C^0(\Omega_d)} \in \mathcal{O}\big(N^{-\frac{(r-d-1)}{d}}\big), \quad ,\forall i = 1, \dots, d. \tag{3.45}$$

The convergence rate deteriorates significantly as the dimension d increases. This effect, can be mitigated by considering polynomials corresponding to an ℓ^p ball index with $p < \infty$, whose the number of coefficients scale subexponentially with the dimension. For instance, for $p = 1$, we have $N := |A_{d,n,1}| = \mathcal{O}(d^n)$, and for $p = 2$, we have $N := |A_{d,n,2}| = o(n^d)$. We refer to [68] *for a deeper discussion on this topic.*

3.8 Integration in the hypercube

In this section, we review numerical integration on the reference hypercube $\Omega_d = [-1,1]^d$. Let $f : \Omega_d \to \mathbb{R}$ be continuous and consider

$$I_d[f] := \int_{\Omega_d} f(\mathbf{x})\, d\mathbf{x}. \tag{3.46}$$

We write $I[f]$ when the dimension is clear. Such integrals arise repeatedly in later chapters after surface integrals are pulled back to reference domains: the integrand then contains the physical function, the parametrization, and the geometric Jacobian. Since these integrals are rarely available in closed form, high-order numerical integration is a central part of the discretization.

A cubature formula approximates (3.46) by a finite weighted sum

$$I_d[f] \approx \boldsymbol{Q}_n[f] = \sum_{\boldsymbol{p}\in P} \omega_{\boldsymbol{p}} f(\boldsymbol{p}). \tag{3.47}$$

The non-zero constants $\omega_{\boldsymbol{p}}$ are the weights and the points $\boldsymbol{p} \in P$ are the integration nodes. If $d = 1$, the rule is called a *quadrature formula*; if $d \geq 2$, it is called a *cubature formula*. We also use the neutral term *integration rule*.

Definition 56. *A cubature formula* $\mathbf{Q}$ *for an integral* I *is said to have* algebraic degree of exactness p *if*

$$\mathbf{Q}[\psi] = I[\psi], \quad \forall \psi \in \Pi_{d,p}, \tag{3.48}$$

and there exists $\phi \in \Pi_{d,p+1}$ *such that*

$$\mathbf{Q}[\phi] \neq I[\phi]. \tag{3.49}$$

In contrast, the order *of a quadrature rule will refer to the parameter controlling the number of quadrature points. For example, in one dimension we write* $\mathbf{Q}_n$ *for a rule with* $n+1$ *nodes. With this convention, the* $(n+1)$*-point Clenshaw–Curtis rule has algebraic degree of exactness* n*, whereas the* $(n+1)$*-point Gauss–Legendre rule has algebraic degree of exactness* $2n+1$*.*

Since both $\mathbf{Q}$ and I are linear, and $\Pi_{d,p}$ is a vector space, the condition (3.48) is equivalent to

$$\mathbf{Q}[\psi_j] = I[\psi_j], \quad j = 1, \ldots, \dim \Pi_{d,p}, \tag{3.50}$$

where the ψ_j form any basis of $\Pi_{d,p}$.

In multi-dimensional settings, cubature rules are often constructed by combining one-dimensional quadrature rules; see Section 2.4. This leads naturally to tensor-product methods, which will be the focus of the following section.

3.8.1 Tensor-product techniques for multivariate integration

Tensor-product quadrature methods are particularly effective for integrating high-degree polynomials over Cartesian product domains in $\mathbb{R}^d$. These methods are especially well-suited to

structured domains such as cubes, which commonly arise in finite element and finite volume methods [113]. They are constructed by taking tensor products of the nodes and weights from one-dimensional quadrature rules. The construction is recursive and proceeds as follows:

Let $\boldsymbol{Q}_n^1$ denote a quadrature rule on $[-1,1]$ with nodes and weights

$$\{(p_j, \omega_j)\}_{j=0}^n.$$

Thus $\boldsymbol{Q}_n^1$ uses $n+1$ function evaluations:

$$\int_{-1}^{1} f(x)\,dx \approx \boldsymbol{Q}_n^1[f] = \sum_{j=0}^{n} \omega_j f(p_j).$$

The algebraic degree of exactness depends on the particular rule; for instance, it is n for Clenshaw–Curtis quadrature and $2n+1$ for Gauss–Legendre quadrature. For $d \geq 2$, the rule is extended recursively. Let $\boldsymbol{Q}_n^{d-1}$ represent a quadrature rule on a $(d-1)$-dimensional domain $\Omega_{d-1} \subset \mathbb{R}^{d-1}$, with nodes $\{\tilde{\boldsymbol{p}}_i\}_{i=0}^{N_{d-1}-1}$ and weights $\{\tilde{\omega}_i\}_{i=0}^{N_{d-1}-1}$, where N_{d-1} is the number of nodes in $\boldsymbol{Q}_n^{d-1}$. Define the d-dimensional domain as $\Omega_d := \Omega_{d-1} \times [-1,1]$. The tensor-product rule $\boldsymbol{Q}_n^{\otimes d}$ for Ω_d is constructed from $\boldsymbol{Q}_n^{d-1}$ and $\boldsymbol{Q}_n^1$ by

$$\int_{\Omega_d} f(\mathbf{x})\,d\mathbf{x} \approx \boldsymbol{Q}_n^{\otimes d}[f] := \sum_{i=0}^{N_{d-1}-1} \sum_{j=0}^{n} \tilde{\omega}_i \omega_j f(\tilde{\boldsymbol{p}}_i, p_j), \quad \text{with } (\tilde{\boldsymbol{p}}_i, p_j) \in \Omega_d.$$

Equivalently, for the tensor-product domain $\Omega_d = [-1,1]^d$, this can be written as

$$\boldsymbol{Q}_n^{\otimes d}[f] := \sum_{i_1=0}^{n} \cdots \sum_{i_d=0}^{n} \omega_{i_1} \cdots \omega_{i_d} f(p_{i_1}, \ldots, p_{i_d}). \tag{3.51}$$

This way, a d-dimensional quadrature rule is defined which requires $(n+1)^d$ function evaluations. The relevance of this construction is not that one needs numerical quadrature to integrate individual monomials, which can of course be integrated analytically. Rather, polynomial exactness provides a precise way to design and analyze quadrature rules. In applications, the functions to be integrated are typically not polynomials; for instance, after a surface integral is pulled back to a reference domain, the integrand contains the physical function, the surface parametrization, and the Jacobian determinant. Tensor-product quadrature is useful because it reduces the construction of multidimensional rules on Cartesian product domains to one-dimensional nodes and weights, while retaining high-order accuracy for smooth integrands.

3.8.2 Convergence of Clenshaw–Curtis and Gauss–Legendre Quadrature

In the univariate case, it is well known that the error in approximating the integral (3.46) decays algebraically for differentiable functions and exponentially for analytic functions [133]. This behavior can be summarized by the following statements:

Theorem 57 (Quadrature formulas for differentiable functions)**.** *Let $f \in C([-1,1])$ and suppose f has derivatives up to $f^{(r-1)}$, which are absolutely continuous on $[-1,1]$, where $r \geq 1$. Further assume that the r-th derivative $f^{(r)}$ is of bounded variation V_f. Then the following error bounds hold:*

1. For the $(n+1)$-point Clenshaw–Curtis quadrature with $n > r$,

$$\left| I[f] - \boldsymbol{Q}_n^{CC}[f] \right| \leq \frac{32}{15\,\pi r} \frac{V_f}{(n-r)^r}. \tag{3.52}$$

2. For the $(n+1)$-point Gauss–Legendre quadrature with $n > 2r+1$,

$$\left| I[f] - \boldsymbol{Q}_n^{GL}[f] \right| \leq \frac{32}{15\,\pi r} \frac{V_f}{(n-2r-1)^{2r+1}}. \tag{3.53}$$

Proof. See [133, Theorem 19.4]. □

Theorem 58 (Quadrature formulas for analytic functions)**.** *Let f be an analytic function on $[-1,1]$ that can be analytically extended to the open Bernstein ellipse E_ρ. Suppose $|f(z)| \leq M$ for some constant M and all $z \in E_\rho$. Then the following error bounds hold:*

1. For the $(n+1)$-point Clenshaw–Curtis quadrature with $n \geq 2$,

$$\left| I[f] - \boldsymbol{Q}_n^{CC}[f] \right| \leq \frac{64\,M\,\rho^{1-n}}{15\,(\rho^2-1)}. \tag{3.54}$$

2. For the $(n+1)$-point Gauss–Legendre quadrature with $n \geq 1$,

$$\left| I[f] - \boldsymbol{Q}_n^{GL}[f] \right| \leq \frac{64\,M\,\rho^{-2n}}{15\,(\rho^2-1)}. \tag{3.55}$$

Moreover, the factor ρ^{1-n} in the Clenshaw–Curtis bound can be improved to ρ^{-n} when n is even, and the constant $64/15$ can be replaced by $144/35$ if $n \geq 4$ in the Clenshaw–Curtis case, or if $n \geq 2$ in the Gauss–Legendre quadrature case.

Proof. See [133, Theorem 19.3]. □

The next theorem gives a tensor-product Clenshaw–Curtis estimate on the hypercube. The proof is intentionally conservative: it combines polynomial exactness with positivity of the weights and the uniform Chebyshev truncation estimates derived earlier. Remark 60 then explains how the sharper one-dimensional Chebyshev-mode argument can be lifted dimension by dimension.

Theorem 59 (Stability-based tensor-product Clenshaw–Curtis estimate)**.** *Let f satisfy the assumptions of Theorem 44, and let $d \in \mathbb{N}$. Then the following error bounds hold for the $(n+1)^d$-point Clenshaw–Curtis quadrature:*

(a) *If $n > r$ and $r > d-1$, then*

$$\left|I[f] - \boldsymbol{Q}_n^{\mathrm{CC}\otimes d}[f]\right| \leq \frac{2^{2d+1} e d V_{f,r}}{\pi(r-d+1)} \left(\frac{n+1}{n+1-r}\right)^{r+1} \frac{1}{n^{r-d+1}}. \tag{3.56}$$

(b) *If the assumptions of Theorem 49 are satisfied and $n > 2$, then*

$$\left|I[f] - \boldsymbol{Q}_n^{\mathrm{CC}\otimes d}[f]\right| \leq 2^{2d+1} e d M \rho^{-n} B_{d-1}(n,\rho). \tag{3.57}$$

Proof. Let $\mathcal{T}_{f,n}$ denote the truncated Chebyshev series of $f : \Omega_d \to \mathbb{R}$ with respect to $A_{d,n}$. The tensor-product Clenshaw–Curtis rule is exact for $\mathcal{T}_{f,n}$, hence

$$I[\mathcal{T}_{f,n}] = \mathbf{Q}_n^{\mathrm{CC}\otimes d}[\mathcal{T}_{f,n}].$$

Therefore,

$$\left|I[f] - \mathbf{Q}_n^{\mathrm{CC}\otimes d}[f]\right| \leq \left|I[f - \mathcal{T}_{f,n}]\right| + \left|\mathbf{Q}_n^{\mathrm{CC}\otimes d}[f - \mathcal{T}_{f,n}]\right|. \tag{3.58}$$

The first term is bounded by $2^d \|f - \mathcal{T}_{f,n}\|_{C^0(\Omega_d)}$, since $\mathrm{vol}(\Omega_d) = 2^d$. The second term satisfies the same bound because the tensor-product Clenshaw–Curtis weights are positive and sum to 2^d. Thus,

$$\left|I[f] - \mathbf{Q}_n^{\mathrm{CC}\otimes d}[f]\right| \leq 2^{d+1} \|f - \mathcal{T}_{f,n}\|_{C^0(\Omega_d)}.$$

Applying Theorem 45 gives (3.56), and applying Theorem 51 gives (3.57). □

Remark 60. *The estimate in the preceding theorem is safe but not sharp in its constants, because it bounds the quadrature error by the uniform norm of the Chebyshev tail. The sharper one-dimensional proof does something more specific: it expands the error in Chebyshev modes and estimates only the modes that the rule does not integrate exactly.*

Let $I_1[g] = \int_{-1}^{1} g(x)\,dx$, and let Q_n^1 be a symmetric positive one-dimensional rule with algebraic degree of exactness ν_n. For Clenshaw–Curtis, $\nu_n = n$. Define the one-dimensional modal defect

$$\delta_k^{(n)} := I_1[T_k] - Q_n^1[T_k].$$

Then $\delta_k^{(n)} = 0$ for $k \leq \nu_n$. Moreover, symmetry gives

$$I_1[T_k] = Q_n^1[T_k] = 0 \qquad \text{for odd } k.$$

Thus the one-dimensional error receives contributions only from even Chebyshev modes above the exactness degree. For even modes,

$$I_1[T_{2m}] = \frac{2}{1-4m^2}, \qquad |I_1[T_{2m}]| \leq \frac{2}{15} \quad (m \geq 2),$$

and positivity gives $|Q_n^1[T_k]| \le Q_n^1[1] = 2$. *Hence, for every even non-exact mode with* $k \ge 4$,

$$|\delta_k^{(n)}| \le 2 + \frac{2}{15} = \frac{32}{15}. \tag{3.59}$$

Together with the one-dimensional Chebyshev coefficient estimate and the summation over even modes only, this is the elementary cancellation mechanism behind the Clenshaw–Curtis part of Theorem 57.

The tensor-product extension is obtained by applying the same argument one coordinate at a time. Suppose

$$f(\mathbf{x}) = \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d} c_{\boldsymbol{\alpha}} T_{\boldsymbol{\alpha}}(\mathbf{x}), \qquad T_{\boldsymbol{\alpha}}(\mathbf{x}) = \prod_{j=1}^{d} T_{\alpha_j}(x_j).$$

For one tensor-product mode, both the exact integral and the tensor-product rule factorize, so

$$I_d[f] - Q_n^{1\otimes d}[f] = \sum_{\boldsymbol{\alpha} \in \mathbb{N}_0^d} c_{\boldsymbol{\alpha}} \Delta_{\boldsymbol{\alpha}}^{(n,d)}, \tag{3.60}$$

where

$$\Delta_{\boldsymbol{\alpha}}^{(n,d)} := \prod_{j=1}^{d} I_1[T_{\alpha_j}] - \prod_{j=1}^{d} Q_n^1[T_{\alpha_j}].$$

The product defect can be telescoped as

$$\Delta_{\boldsymbol{\alpha}}^{(n,d)} = \sum_{\ell=1}^{d} \left(\prod_{j<\ell} Q_n^1[T_{\alpha_j}] \right) \delta_{\alpha_\ell}^{(n)} \left(\prod_{j>\ell} I_1[T_{\alpha_j}] \right). \tag{3.61}$$

This is the concrete multidimensional version of the one-dimensional proof: each summand contains exactly one one-dimensional quadrature defect, while the remaining directions contribute only exact or quadrature moments.

The consequences are immediate. First, if one component α_j *is odd, then both products in* $\Delta_{\boldsymbol{\alpha}}^{(n,d)}$ *vanish. Second, if all components satisfy* $\alpha_j \le \nu_n$, *then all one-dimensional defects vanish. Hence only the even non-exact tail*

$$\mathcal{E}_{d,n} := \{\boldsymbol{\alpha} \in (2\mathbb{N}_0)^d : \|\boldsymbol{\alpha}\|_\infty > \nu_n\}$$

can contribute. Let

$$s_n(\boldsymbol{\alpha}) := \#\{j : \alpha_j > \nu_n\}$$

be the number of non-exact coordinate directions. Using $|I_1[T_k]| \le 2$, $|Q_n^1[T_k]| \le 2$, *and (3.59), equation (3.61) gives*

$$|I_d[f] - Q_n^{1\otimes d}[f]| \le 2^{d-1} C_Q \sum_{\boldsymbol{\alpha} \in \mathcal{E}_{d,n}} s_n(\boldsymbol{\alpha}) |c_{\boldsymbol{\alpha}}|, \tag{3.62}$$

where C_Q *is a one-dimensional bound for the even non-exact defects; the elementary bound above gives* $C_Q = 32/15$.

Combining (3.62) with the coefficient estimate from Theorem 44 yields a parity-filtered shell estimate. In

the usual stability proof, the shell $\|\boldsymbol{\alpha}\|_\infty = q$ has cardinality

$$(q+1)^d - q^d \le edq^{d-1}.$$

In the modewise quadrature estimate, only even shells remain. For $q = 2m$,

$$|E^{\text{even}}_{d,2m}| = (m+1)^d - m^d \le ed\left(\frac{q}{2}\right)^{d-1}, \qquad |E^{\text{even}}_{d,2m+1}| = 0.$$

Therefore the contribution of the shell $\|\boldsymbol{\alpha}\|_\infty = q = 2m$ is bounded by

$$2^{d-1} C_Q\, d\, \frac{2^d V_{f,r}}{\pi q(q-1)\cdots(q-r)} \left((m+1)^d - m^d\right),$$

where the factor d comes from the crude bound $s_n(\boldsymbol{\alpha}) \le d$. This is the same summation as in the proof of (3.56), but with the full shell count $(q+1)^d - q^d$ replaced by the parity-filtered count $(m+1)^d - m^d$, and with the one-dimensional modal-defect constant $C_Q = 32/15$ made explicit.

Thus the improved estimate has the same algebraic rate as (3.56), but a smaller tail: all odd shells are removed, and the factor $s_n(\boldsymbol{\alpha})$ records how many directions are actually beyond the exactness degree. In particular, for $d = 1$, equations (3.60)–(3.62) reduce to

$$|I_1[f] - Q^1_n[f]| \le C_Q \sum_{\substack{k > \nu_n \\ k \text{ even}}} |c_k|,$$

which is precisely the one-dimensional Chebyshev-mode tail estimate. For Clenshaw–Curtis, $\nu_n = n$ and $C_Q = 32/15$; with the standard one-dimensional coefficient bound and the sharp even-tail summation, this gives

$$|I_1[f] - Q^{\text{CC}}_n[f]| \le \frac{32}{15\pi r} \frac{V_{f,r}}{(n-r)^r},$$

which is the Clenshaw–Curtis estimate in Theorem 57. The point is not a new rate, but a clearer and sharper tensor-product constant: the multidimensional proof is obtained by tensorizing the same parity and modal-defect argument used in one dimension.

Chapter 4

Representations of surfaces

In this chapter, we review the surface representations that are used in the methods developed later in the thesis. Our attention is restricted to d-dimensional regular surfaces that admit a differentiable structure, allowing them to be treated as d-dimensional manifolds; see Definition 61. The implicit function theorem provides the theoretical justification for this identification by asserting that a sufficiently smooth implicit surface locally defines a manifold [23, Proposition 4.16]. Throughout, we adopt this equivalence and use the terms surface and manifold interchangeably where appropriate.

Definition 61 (Regular Surface)**.** *Let $r \in \mathbb{N}$. A d-dimensional surface $\Gamma \subseteq \mathbb{R}^{d+1}$ of class C^r is defined as follows: for every point $\mathbf{p} \in \Gamma$, there exists an open neighborhood $\mathcal{N}_\delta \subset \mathbb{R}^{d+1}$ containing $\mathbf{p}$, an open set $U \subset \mathbb{R}^d$, and a map $\boldsymbol{\rho_p} : U \to \mathcal{N}_\delta \cap \Gamma$ such that the following conditions hold:*

1. $\boldsymbol{\rho_p} \in C^r(U; \mathbb{R}^{d+1})$.
2. $\boldsymbol{\rho_p}$ *is a homeomorphism.*
3. *For every point $\mathbf{q} \in U$, the Jacobian $d\boldsymbol{\rho}_\mathbf{q} : \mathbb{R}^d \to \mathbb{R}^{d+1}$ is injective (i.e., has full rank d).*

The maps $\boldsymbol{\rho}_\mathbf{p}$ are called local parametrizations or local charts of the surface. We say $f \in C^r(\Gamma)$, if for every local parametrization $\boldsymbol{\rho}_\mathbf{p}$ of Γ we have $f \circ \boldsymbol{\rho}_\mathbf{p} \in C^r(U; \mathbb{R})$. These maps continuously and invertibly identify subsets of $\mathbb{R}^d$ with subsets of Γ (property 2). Properties 1 and 3 ensure that the surface is smoothly embedded in $\mathbb{R}^{d+1}$, meaning it has no cusps. The third property allows combining a set of parametrizations into an atlas covering the entire surface. Such a (finite) covering always exists since surfaces are compact subsets of $\mathbb{R}^{d+1}$. A surface Γ can be represented in various ways, such as through a parametrization over a reference domain, an implicit representation as the level set of a function, or by closest-point projection of coordinates

onto another surface in a neighborhood close to Γ.

At the most abstract level, these representations are categorized into two primary types: *implicit* and *parametric* representations. Each representation method has its own advantages and limitations [44], which we discuss in Sections 4.1 and 4.2, respectively.

4.1 Implicit representations

In implicit representations, surfaces are described as the *kernel* or zero-level set of a function:

Definition 62. *An implicit (or volumetric) surface in $\mathbb{R}^{d+1}$ is defined as the zero-set of a scalar-valued function $F_\Gamma : \mathbb{R}^{d+1} \to \mathbb{R}$, i.e.,*

$$\Gamma = \{\mathbf{x} \in \mathbb{R}^{d+1} \mid F_\Gamma(\mathbf{x}) = 0\}.$$

From the implicit function theorem it may be shown that for $F_\Gamma(\mathbf{x}) = 0$, where 0 is a regular value[1] of F_Γ, the implicit surface is a d-dimensional manifold [23, Proposition 4.16]. The *Jordan-Brouwer Separation Theorem* states that such a manifold separates space into the surface itself and two connected open sets: an infinite *outside* and a finite *inside* [63]. More precisely, a given domain $\Omega \subset \mathbb{R}^{d+1}$ can be decomposed into an inner part Ω_1, an outer part Ω_1^c, and their common interface Γ by requiring that $\forall \mathbf{x} \in \Omega$

$$\begin{cases} F_\Gamma(\mathbf{x}) < 0 \iff \mathbf{x} \in \Omega_1, \\ F_\Gamma(\mathbf{x}) = 0 \iff \mathbf{x} \in \Gamma, \\ F_\Gamma(\mathbf{x}) > 0 \iff \mathbf{x} \in \Omega_1^c, \end{cases}$$

where we define Ω_1^c as the (open) complement of Ω_1 in Ω. F_Γ does not explicitly describe the surface, but implies its existence.

Example 63. *Consider two examples for which no manifold exists:*

- *Trivial Case: $F_\Gamma(\mathbf{x}) := 0$. Here, $\nabla F_\Gamma(\mathbf{x})$ is everywhere zero, resulting in no well-defined normal direction or tangent space at any point. Consequently, there is no well-defined inside or outside, and no boundary separates the two regions.*
- *Degenerate Sphere: $F_\Gamma(\mathbf{x}) = \|\mathbf{x}\|^2$. Here, $\nabla F_\Gamma(\mathbf{x}) = (2x_1, 2x_2, \ldots, 2x_{d+1})$, which vanishes at the origin, the only point satisfying $F_\Gamma(\mathbf{x}) = 0$. This singularity at the origin prevents the surface from being a smooth manifold.*

The function $F_\Gamma(\mathbf{x})$ is typically defined using one of three primary approaches:

[1] If the gradient of F_Γ is non-null at a point $\mathbf{x}$, then $\mathbf{x}$ is said to be regular and $\nabla F_\Gamma(\mathbf{x})$ is normal (i.e., perpendicular) to the surface at $\mathbf{x}$. If, however, the gradient (or, equivalently, the tangent vector) is indeterminate, the point is singular (also called critical or non-regular). For example, the apex of a cone is a singular point. If the surface is regular and the second partial derivatives are continuous, then the surface will have continuous curvature (i.e., the surface is C^2 continuous). Also, if the surface is regular, it is a *manifold*.

- Through *discrete samples*, which are generally uniformly distributed within a finite volume. These samples often arise from physical measurements such as opacity, density, or temperature. The associated isosurface is defined as the level set $F_\Gamma^{-1}(c) = \{\mathbf{x} \in \mathbb{R}^{d+1} : F_\Gamma(\mathbf{x}) = c\}$, where c is the *isocontour value*.

- By *mathematical functions*, where one or more equations specify the coordinates of the points $\mathbf{x}$. If the defining function F_Γ is polynomial, the surface is called *algebraic*, and lies within the classical domain of algebraic geometry. In this setting, the zero set of F_Γ forms the surface, and its properties can be analyzed symbolically. If F_Γ is instead defined using non-polynomial expressions—such as exponential, trigonometric, or logarithmic terms—the resulting surface is termed *transcendental*, and often arises in applied mathematical models.

- Via *procedural methods*, where F_Γ is specified through an algorithmic or black-box process. In such cases, no closed-form expression for the surface exists, and geometric features must be inferred by sampling F_Γ numerically over the domain.

These approaches are versatile and particularly effective for representing surfaces with highly complex topologies. One of the key advantages of these methods is that the global consistency of the surface is guaranteed by construction.

Among the most common implicit representations of surfaces are:

- *Level set methods* [99, 100, 118–120], which are widely used for evolving interfaces;

- *Local kernel* (*radial basis function*) *parametrizations* [28, 29, 40, 72, 102], which allow for flexible surface fitting and reconstruction;

- *Closest point methods* [86–88, 112], where the surface is represented in terms of points nearest to it in the embedding space.

Despite their different mathematical structures, all of these methods share a common feature: *they approximate the surface using discrete points, typically distributed uniformly within a finite volume*. This discretization simplifies complex surface modeling and representation.

4.1.1 Curvatures and differential operators on surfaces

The practical value of implicit representations becomes especially clear when one computes differential quantities on d-dimensional regular surfaces. In numerical settings, quantities such as curvature are often computed on discrete surface approximations. This introduces a second layer of approximation, involving the discretization of surface differential operators on the discrete surface.

While these methods are highly effective, this dual approximation (both of the surface and its differential properties) can limit the accuracy achievable, particularly for higher-order geometric quantities, which often require precise surface differentials. We provide explicit formulas

for computing mean curvature and Gauss curvature. We consider the implicit surface descriptions of the two-dimensional surfaces $\Gamma \subseteq \mathbb{R}^{d+1}$, where our surfaces are defined as the zero-level set $F_\Gamma^{-1}(0) = \Gamma$. To derive explicit formulas for the basic geometric quantities of Γ, we follow [59] within $\mathbb{R}^{d+1}$, using the standard inner product $\langle e_i, e_j \rangle = \delta_{i,j}$ and the standard basis $\{e_i\}_{i=1,\dots,d+1}$.

If Γ is orientable, then the *gradient* $\nabla F_\Gamma = (\partial_{x_1} F_\Gamma, \partial_{x_2} F_\Gamma, \dots \partial_{x_{d+1}} F_\Gamma) \in \mathbb{R}^{d+1}$ never vanishes on Γ and provides, together with the *Hessian* $H_\Gamma = \nabla(\nabla F_\Gamma) \in \mathbb{R}^{(d+1)\times(d+1)}$ of F_Γ

$$H_\Gamma = \begin{pmatrix} \frac{\partial^2 F_\Gamma}{\partial x_1^2} & \frac{\partial^2 F_\Gamma}{\partial x_1 \partial x_2} & \cdots & \frac{\partial^2 F_\Gamma}{\partial x_1 \partial x_{d+1}} \\ \frac{\partial^2 F_\Gamma}{\partial x_2 \partial x_1} & \frac{\partial^2 F_\Gamma}{\partial x_2^2} & \cdots & \frac{\partial^2 F_\Gamma}{\partial x_2 \partial x_{d+1}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 F_\Gamma}{\partial x_{d+1} \partial x_1} & \frac{\partial^2 F_\Gamma}{\partial x_{d+1} \partial x_2} & \cdots & \frac{\partial^2 F_\Gamma}{\partial x_{d+1}^2} \end{pmatrix},$$

the main ingredients for computing Gauss and mean curvature. Both curvatures can be derived from these quantities [59] as:

$$K_{\text{Gauss}} = \frac{\det \begin{pmatrix} H_\Gamma & \nabla F_\Gamma^T \\ \nabla F_\Gamma & 0 \end{pmatrix}}{\|\nabla F_\Gamma\|^4} \tag{4.1}$$

$$K_{\text{mean}} = \frac{\nabla F_\Gamma H_\Gamma \nabla F_\Gamma^T - \|\nabla F_\Gamma\|^2 \text{trace}(H_\Gamma)}{2\|\nabla F_\Gamma\|^3}. \tag{4.2}$$

While several alternative formulas exist [59], the two mentioned above enable stable and numerically accurate evaluation, as demonstrated in [138, Section 10.1].

4.1.2 Surface regularity and closest point projection

Assume that our d-dimensional regular surface Γ is a smooth, connected, orientable, and closed C^r manifold (with $r \geq 2$), embedded in a $(d+1)$-dimensional ambient space. Given $\Gamma \subset \mathbb{R}^{d+1}$ for each point $\mathbf{x} \in \Gamma$ the tangent space $T_\mathbf{x}\Gamma \subseteq \mathbb{R}^{d+1}$ is a linear subspace. The disjoint union of tangent spaces to Γ is the tangent bundle of Γ:

$$T\Gamma := \bigcup_{\mathbf{x}\in\Gamma} T_\mathbf{x}\Gamma = \{(\mathbf{x}, \mathbf{y}) \,|\, \mathbf{x} \in \Gamma \text{ and } \mathbf{y} \in T_\mathbf{x}\Gamma\}.$$

The tangent bundle is a smooth manifold of dimension $2d$. The normal space of Γ in $\mathbb{R}^{d+1}$ at $\mathbf{x}$ is the orthogonal complement of the tangent space $T_\mathbf{x}\Gamma$, namely $N_\mathbf{x}\Gamma = \left(T_\mathbf{x}\mathbb{R}^{d+1}\right)^\perp$. The normal bundle is a smooth embedded submanifold of $\mathbb{R}^{d+1} \times \mathbb{R}^{d+1}$ of dimension d,

$$N\Gamma := \bigcup_{\mathbf{x}\in\Gamma} N_\mathbf{x}\Gamma = \{(\mathbf{x}, \mathbf{z}) \,|\, \mathbf{x} \in \Gamma \text{ and } \mathbf{z} \in N_\mathbf{x}\Gamma\}.$$

Let $\delta : \Gamma \longrightarrow \mathbb{R}_+$ be a positive, continuous function. Consider the following open tubular neighborhood of the normal bundle:

$$N_\delta = \{(\mathbf{x}, \mathbf{y}_\mathbf{x}) \mid \|\mathbf{y}_\mathbf{x}\| < \delta(\mathbf{x})\},$$

where $\mathbf{y}_\mathbf{x}$ is normal vector attached at $\mathbf{x}$. The map $\mathcal{F} : N\Gamma \longrightarrow \mathbb{R}^{d+1}$ $(\mathbf{x}, \mathbf{y}) \mapsto \mathbf{x} + \mathbf{y}$ is smooth and there exists a δ such that the restriction $\mathcal{F}|_{N_\delta}$ becomes a diffeomorphism onto its image [84]. Consequently, $\mathcal{N}_\delta = \mathcal{F}(N_\delta)$ is a $d-$dimensional, open, smooth, embedded submanifold of $\mathbb{R}^{d+1}$ that forms a tubular neighborhood of Γ.

Definition 64. *The signed distance function $d : \mathcal{N}_\delta \to \Gamma$ is defined for every $\mathbf{x} \in \mathcal{N}_\delta$ as follows: if $\mathbf{x}$ belongs to the exterior of Γ, it is the distance from $\mathbf{x}$ to Γ, denoted* $\operatorname{dist}(\mathbf{x}, \Gamma)$. *Conversely, if $\mathbf{x}$ belongs to the interior of Γ, it is defined as* $-\operatorname{dist}(\mathbf{x}, \Gamma)$. *Thus,*

$$|d(\mathbf{x})| = \operatorname{dist}(\mathbf{x}, \Gamma) \quad \forall \mathbf{x} \in \mathcal{N}_\delta.$$

Recall the tubular neighborhood $\mathcal{N}_\delta \subset \mathbb{R}^{d+1}$ defined via the normal bundle. If $\Gamma \in C^2$, the signed distance function d is well-defined and smooth in $\mathcal{N}_\delta$. To ensure the existence and uniqueness of the closest point projection, we restrict δ further based on curvature bounds.

Let $\kappa_i(\mathbf{x})$, for $i = 1, \ldots, d$, denote the principal curvatures at $\mathbf{x} \in \Gamma$, and define

$$K(\mathbf{x}) := \max_{1 \le i \le d} |\kappa_i(\mathbf{x})|, \quad \text{and} \quad K_\infty := \|K\|_{L^\infty(\Gamma)}.$$

Then, for $\delta := \frac{1}{2K_\infty}$, we consider the curvature-controlled neighborhood

$$\mathcal{N}_\delta := \left\{ \mathbf{x} \in \mathbb{R}^{d+1} : \operatorname{dist}(\mathbf{x}, \Gamma) < \frac{1}{2K_\infty} \right\} \subset \mathcal{F}(N_\delta), \tag{4.3}$$

which guarantees the existence and uniqueness of the projection onto Γ.

For C^r surfaces, where $r \ge 2$, the gradient $\nabla d(\mathbf{x})$ (with ∇ denoting the standard gradient in $\mathbb{R}^{d+1}$, see [38] for more details) is well-defined for all $\mathbf{x} \in \mathcal{N}_\delta$. When computed on Γ, this gradient gives the unit normal $\mathbf{n}(\mathbf{x})$ pointing outwards:

$$\mathbf{n}(\mathbf{x}) = \nabla d(\mathbf{x}) \quad \forall \mathbf{x} \in \Gamma.$$

Since $\nabla d(\mathbf{x})$ is defined in $\mathcal{N}_\delta$, it naturally extends $\mathbf{n}$ to $\mathcal{N}_\delta$.

The neighborhood $\mathcal{N}_\delta$ is sufficiently small such that for every $\mathbf{x} \in \mathcal{N}_\delta$, there exists a unique closest point projection $\pi(\mathbf{x}) \in \Gamma$, defined by

$$\pi(\mathbf{x}) = \mathbf{x} - d(\mathbf{x}) \nabla d(\mathbf{x}) \quad \forall \mathbf{x} \in \mathcal{N}_\delta.$$

An important property is that $\nabla d(\mathbf{x})$ coincides at $\mathbf{x} \in \mathcal{N}_\delta$ and $\pi(\mathbf{x}) \in \Gamma$:

$$\nabla d(\mathbf{x}) = \nabla d(\pi(\mathbf{x})) = \nabla d(\mathbf{x} - d(\mathbf{x})\nabla d(\mathbf{x})) \quad \forall \mathbf{x} \in \mathcal{N}_\delta.$$

It is natural to inquire which properties of the distance function and closest point projection extend for C^r surfaces, where $r < 2$. We use the classical regularity result stated in [57].

Lemma 65 (Properties of distance functions for C^r surfaces)**.** *Let Γ be a C^r surface, $r \geq 2$. Then there exists a positive constant δ depending on Γ such that $d \in \mathrm{C}^r(\mathcal{N}_\delta)$. In addition, the closest point projection $\pi(\mathbf{x}) = \mathbf{x} - d(\mathbf{x})\nabla d(\mathbf{x})$ is defined and of class C^{r-1} on $\mathcal{N}_\delta$ with $\delta < \frac{1}{K_\infty}$.*

However, it has been demonstrated [16, Theorem 11] that for less regular C^r, where $r < 2$, such as $C^{1,1}$ surfaces the distance function and closest point projection are of limited utility. Specifically, in this scenario, the closest point projection is not uniquely defined on any tubular neighborhood of Γ and the regularity of the distance function/the closest point projection does not vary continuously with that of Γ.

In summary, the properties of the distance function and the associated closest-point projection for C^r, $r \geq 2$, surfaces are intrinsically linked to surfaces of bounded curvature. *This C^r regularity assumption on Γ will be maintained throughout this thesis.*

4.1.3 Tangential differential calculus on surfaces

Let Γ be a smooth surface embedded in $\mathbb{R}^{d+1}$ and covered by a finite family of local parametrizations

$$\rho_i : \Omega_d \longrightarrow \mathbb{R}^{d+1}, \qquad i = 1, 2, \ldots, K.$$

Here K denotes the number of coordinate patches in the chosen local cover. Let $\partial_j \rho_i(\mathbf{y})$ be the column vector of j-th partial derivatives of ρ_i for $1 \leq j \leq d$ at $\mathbf{y} \in \Omega_d$. For immersed surfaces $\Gamma \subset \mathbb{R}^{d+1}$ we will write $D\rho_i(\mathbf{y}) : \mathbb{R}^d \to \mathbb{R}^{d+1}$ for the Jacobian of the parametrization ρ_i at $\mathbf{y}$. By Definition 61, the rank of $D\rho_i(\mathbf{y}) = (\partial_j \rho_i(\mathbf{y}))_{j=1}^d \in \mathbb{R}^{(d+1)\times d}$ is d (full rank). This implies that $\{\partial_j \rho_i(\mathbf{y})\}_{j=1}^d$ are linearly independent and span the tangent hyperplane to Γ at $\mathbf{x} = \rho_i(\mathbf{y})$.

The *first fundamental form* is the symmetric and positive definite matrix $\mathbf{g} \in \mathbb{R}^{d\times d}$ defined by

$$\mathbf{g}(\mathbf{y}) := D\rho_i(\mathbf{y})^T D\rho_i(\mathbf{y}). \tag{4.4}$$

If $\mathbf{g} = (g_{\alpha\beta})_{\alpha,\beta=1}^d$, then the components $g_{\alpha\beta}$ read

$$g_{\alpha\beta} = \partial_\alpha \rho_i^T \partial_\beta \rho_i = \partial_\alpha \rho_i \cdot \partial_\beta \rho_i,$$

which depends on the choice of parametrization.

Given $v \in C^1(\Omega_d, \mathbb{R})$, let $\tilde{v} : \rho_i(\Omega_d) \to \mathbb{R}$ be the corresponding function on the surface patch, de-

fined by

$$\tilde{v}(\mathbf{x}) = v(\rho_i^{-1}(\mathbf{x})), \qquad \mathbf{x} = \rho_i(\mathbf{y}).$$

The *tangential* (*or surface*) *gradient* of $\tilde{v}$ is defined as the vector tangent to Γ that satisfies the chain rule

$$\nabla v(\mathbf{y}) = D\rho_i(\mathbf{y})^T \nabla_\Gamma \tilde{v}(\mathbf{x}) \quad \forall \mathbf{y} \in \Omega_d.$$

Since $\nabla_\Gamma \tilde{v}$ is spanned by $\{\partial_j \rho_i\}_{j=1}^d$, we get $\nabla_\Gamma \tilde{v} = D\rho_i w$ for some $w \in \mathbb{R}^d$ whence $w = \mathbf{g}^{-1}\nabla v$ and

$$\nabla_\Gamma \tilde{v}(\mathbf{x}) = D\rho_i(\mathbf{y})\mathbf{g}(\mathbf{y})^{-1}\nabla v(\mathbf{y}) \quad \forall \mathbf{y} \in \Omega_d. \tag{4.5}$$

If $\tilde{\mathbf{v}} = (\tilde{v}_i)_{i=1}^{d+1} : \Gamma \to \mathbb{R}^{d+1}$ is a vector field of class C^1, we define its tangential differential $D_\Gamma \tilde{\mathbf{v}} \in \mathbb{R}^{(d+1)\times(d+1)}$ as a matrix whose i-th row is $(\nabla_\Gamma \tilde{v}_i)^T$.

$$D_\Gamma \tilde{\mathbf{v}} = \begin{pmatrix} (\nabla_\Gamma \tilde{v}_1)^T \\ (\nabla_\Gamma \tilde{v}_2)^T \\ \vdots \\ (\nabla_\Gamma \tilde{v}_{d+1})^T \end{pmatrix} = \begin{pmatrix} (\partial_1^\Gamma \tilde{v}_1, \ldots, \partial_{d+1}^\Gamma \tilde{v}_1) \\ (\partial_1^\Gamma \tilde{v}_2, \ldots, \partial_{d+1}^\Gamma \tilde{v}_2) \\ \vdots \\ (\partial_1^\Gamma \tilde{v}_{d+1}, \ldots, \partial_{d+1}^\Gamma \tilde{v}_{d+1}) \end{pmatrix}, \tag{4.6}$$

for the $d+1$ components of the tangential gradient. Here, we identify ∂_i^Γ with the tangential derivative ∂_i along the surface Γ.

In addition, the *tangential divergence* of $\tilde{\mathbf{v}}$ is the trace of $D_\Gamma \tilde{\mathbf{v}}$

$$\operatorname{div}_\Gamma \tilde{\mathbf{v}}(\mathbf{x}) = \operatorname{trace}(D_\Gamma \tilde{\mathbf{v}}(\mathbf{x})) = \sum_{\alpha.\beta=1}^{d} g^{\alpha\beta}(\mathbf{y}) \partial_\alpha \rho_i(\mathbf{y}) \cdot \partial_\beta \mathbf{v}(\mathbf{y}) \quad \forall \mathbf{y} \in \Omega_d, \tag{4.7}$$

where $\mathbf{v} = \tilde{\mathbf{v}} \circ \rho_i$ and $g^{-1} = (g^{\alpha\beta})_{\alpha.\beta=1}^d$. If both Γ and $\tilde{v} : \Gamma \to \mathbb{R}$ are of class C^2, then the *Laplace–Beltrami* (*or surface Laplace*) *operator* is defined by

$$\Delta_\Gamma \tilde{v} = \frac{1}{\sqrt{\det \mathbf{g}(\mathbf{y})}} \operatorname{div}\left(\sqrt{\det \mathbf{g}(\mathbf{y})}\mathbf{g}(\mathbf{y})^{-1}\nabla v(\mathbf{y})\right) \quad \forall \mathbf{y} \in \Omega_d. \tag{4.8}$$

4.1.4 Global Polynomial Level Set

The methods mentioned in Section 4.1—including level set formulations, radial basis function parametrizations, and closest point methods—all represent surfaces through discrete data such as points, meshes, or grids. Geometric quantities are then computed numerically on these representations. Since both the surface and its differential operators are discretized, the resulting errors accumulate, limiting the attainable accuracy—particularly for higher-order quantities.

The *Global Polynomial Level Set* (GPLS) method [138] circumvents this limitation by constructing a global representation of the surface as an algebraic variety. Given a point cloud $P \subseteq \Gamma \subset \mathbb{R}^3$, the

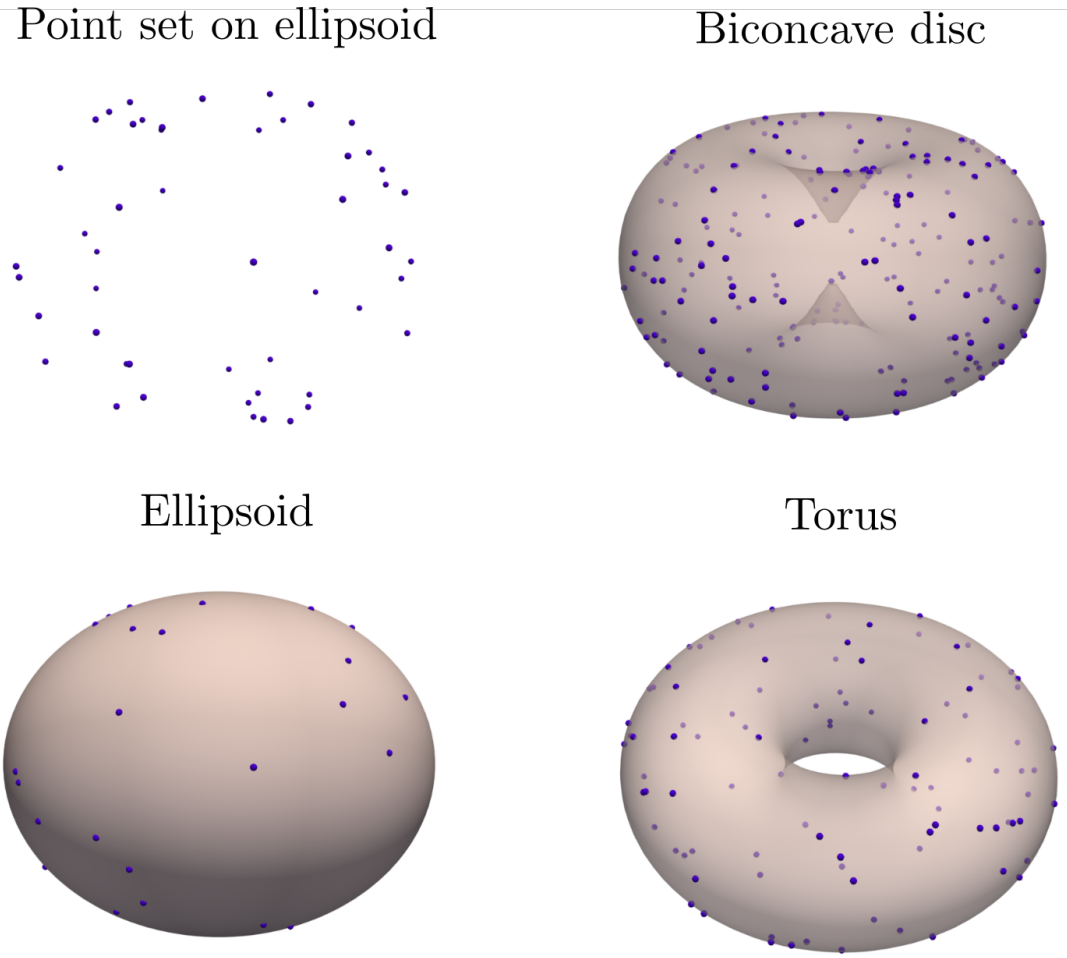


Figure 4.1: The GPLS surface $Q_\Gamma^{-1}(0)$ constructed from randomly sampled points (blue dots) on three algebraic test surfaces.

method computes a multivariate polynomial $Q_\Gamma(\mathbf{x})$ such that $\Gamma \approx Q_\Gamma^{-1}(0)$. This allows geometric and differential computations to be carried out directly on a globally defined surface, without requiring a secondary discretization of differential operators.

Using GPLS, the surface $\Gamma \approx Q_\Gamma^{-1}(0)$ is approximated (up to the interpolation or fitting error) by a uniquely determined algebraic variety, as shown in [138, Theorem 7.1 (iv–vi)]. Table 4.1 reports the reconstruction errors obtained by the GPLS method using randomly sampled surface points from several algebraic surfaces. In this table, N denotes the number of random surface samples used to construct the GPLS approximation, not the number of points at which the error is merely evaluated. The results confirm the high accuracy of GPLS across different geometries, highlighting its suitability for precise surface representation. Later, in Chapter 5, we employ the GPLS method together with the integration scheme developed in this thesis, demonstrating that GPLS is a strong candidate for performing integration on non-parametrized surfaces.

While implicit representations—such as the Global Polynomial Level Set method—provide smooth

Algebraic 2D surface	L_∞ errors of: surface fitting	coefficients	validation	N
Ellipsoid ($a = 0.8, b = 0.9, c = 1.0$)	$(9 \pm 2) \cdot 10^{-16}$	$(1 \pm 1) \cdot 10^{-15}$	$(9 \pm 2) \cdot 10^{-16}$	50
Biconcave disc ($d = 0.5, c = 0.375$)	$(7 \pm 4) \cdot 10^{-14}$	$(3 \pm 4) \cdot 10^{-7}$	$(1 \pm 3) \cdot 10^{-10}$	200
Torus ($R = 0.5, r = 0.3$)	$(3 \pm 1) \cdot 10^{-14}$	$(1 \pm 1) \cdot 10^{-12}$	$(8 \pm 6) \cdot 10^{-14}$	100

Table 4.1: Reconstruction errors (mean and maximum deviation) of [138, Experiments 1.2], based on N random surface point samples used to construct the GPLS approximation of the 2D surfaces shown in Figure 4.1.

and often highly accurate models of surfaces, many practical applications demand a more explicit representation. This is especially true in areas such as visualization, finite element analysis, and mesh-based numerical simulation.

In such settings, the surface is typically approximated by a mesh of triangles or polygons; that is, a discrete collection of piecewise-linear, semi-disjoint elements. Provided that the surface F_Γ is at least C^1, such a triangulation is always possible: every differentiable manifold admits a triangulation [141, 142]. This brings us naturally to the second major category of surface representations.

4.2 Parametric representations

Parametric representations such as splines [47], subdivision surfaces [148], or triangle meshes are based on a mapping from a (piecewise) planar domain into $\mathbb{R}^{d+1}$. In other words, parametric (or explicit) surfaces are defined by a vector-valued parametrization function $f : \Omega \to \Gamma$, that maps a d-dimensional parameter domain $\Omega \subset \mathbb{R}^d$ to the surface $\Gamma \subseteq \mathbb{R}^{d+1}$. Triangle meshes[2] are arguably the most predominant surface representation, both in geometry processing and computer graphics, as well as in other fields such as computational geometry and topology. The popularity of triangle meshes comes from their simplicity, flexibility, and the existence of many data structures for efficient mesh navigation and manipulation [15, 129]. Many methods have been developed to compute or modify triangulations of given surfaces or point clouds, while promoting properties such as alignment to shape features (e.g., ridges or creases), adapting sampling density to geometric detail, or improving triangle aspect ratios; see [11] for an overview. Here we do not cover the detailed process of meshing a surface; for that topic, we refer to [14] and the references therein. We instead use this section to introduce the *triangulation of regular surfaces in* $\mathbb{R}^3$:

Definition 66 (Triangulation of surfaces in $\mathbb{R}^3$)**.** *Let Γ be a regular surface. A triangle in Γ is a closed subset T which is the image of a homeomorphism $\psi : T^* \to T$ where T^* is a triangle in the plane $\mathbb{R}^2$. (i.e., a compact subset of $\mathbb{R}^2$ bounded by three distinct straight lines). The vertices, edges, and face of T are the images of the vertices, edges, and face under ψ.*

[2] Triangle meshes, being explicitly defined, are categorized as an explicit surface representation, albeit not a parametric one.

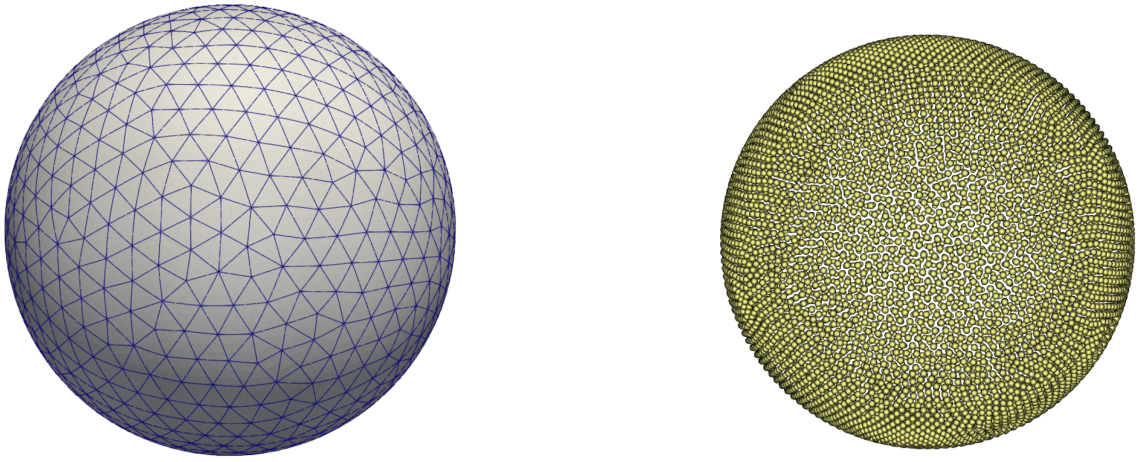


Figure 4.2: Illustrations of a surface discretized with a triangular tessellation (left) and with a point cloud (right).

Definition 67. *A triangulation*[3] *of a compact surface* Γ *consists of a finite family of closed subsets* $\{T_1, T_2, \ldots, T_n\}$ *such that any two distinct triangles,* T_i *and* T_j *are either disjoint, have a single vertex in common, or have one entire edge in common.*

Implicit and parametric surface representations each offer distinct advantages, making it valuable to be able to convert between the two forms. For instance, when calculating the intersection of two parametric surfaces, the parametric equation of one surface can be substituted into the implicit form of the other [71].

The process of converting a parametric surface to an implicit form is known as *implicitization* and can be applied to any rational parametric surface or curve [117]. Implicitization involves eliminating the parameters from the parametric equations. For example, eliminating s and t from a set of rational equations produces the implicit form in terms of $\mathbf{x}$ [71, 80]. Conversely, the conversion from implicit to parametric form is referred to as *parametrization*. This process, called inversion, involves associating a point $\mathbf{x}$ with its corresponding parametric coordinates (s,t) [115]. However, parametrizing a surface in terms of rational functions is not always possible; for instance, many implicit surfaces defined by polynomials of degree four or higher cannot be expressed through rational parametrizations [116]. Despite this, conversion is always possible for non-degenerate quadrics and some cubics that have a singular point.

[3] Radó (1925) showed *that every surface has a triangulation*. It might come as a surprise that for every $d \geq 4$, there is a compact $d-$ dimensional surface (without boundary) that does not have a triangulation. The latter results are very deep. For dimension 4 it follows from results of Freedman (1982) and Casson (1990) – this is actually part of the work Freedman won the fields medal for. For dimensions at least 5 the result is due to Manolescu (2013). In contrast, it was proved much earlier by Cairns (1935) and Whitehead (1940) that every *differentiable surface* has a triangulation.

4.2.1 Simplicial complexes

Definition 68. *A simplicial complex $\mathcal{T}_h$ in $\mathbb{R}^d$ is a finite set of simplices in $\mathbb{R}^d$ that satisfies the following conditions*

1. *Any subsimplex of a simplex from $\mathcal{T}_h$ is also in $\mathcal{T}_h$.*
2. *The intersection of any two simplices $T_1, T_2 \in \mathcal{T}_h$ is a face of both T_1 and T_2.*

In other words, the first condition asks $\mathcal{T}_h$ to be closed under taking subsimplices, and the second condition asks that the intersection of any two simplices is either a common subsimplex or empty because the empty set is a face of every simplex.

Any subset $\mathcal{T}_h' \subseteq \mathcal{T}_h$ that is itself a simplicial complex is called a *subcomplex* of $\mathcal{T}_h$. We say that a *simplicial k-complex* $\mathcal{T}_h$ is a simplicial complex where the largest dimension of any simplex in $\mathcal{T}_h$ is k. Thus, a simplicial 2-complex does not contain tetrahedra or higher-dimensional simplices. The 0-complex of $\mathcal{T}_h$ is called the *vertex set* of $\mathcal{T}_h$. We can also think of a simplicial complex as a space with a triangulation, which is the division of a surface or a plane polygon into a set of 2-simplices.

Throughout this thesis we work with d-dimensional closed surfaces, $\Gamma \subseteq \mathbb{R}^{d+1}$, with $\partial\Gamma = \emptyset$. We assume that we have a polyhedral reference surface Γ_h in Euclidean three-space, which is defined to be a compact subset $\Gamma_h \subseteq \mathbb{R}^{d+1}$ topologically equivalent to the smooth surface Γ. This discrete surface is composed of finitely many triangles whose vertices are located in Γ, ensuring that each edge is contained in a certain (affine) line and each face is contained in a certain (affine) plane: Γ_h is the union of finitely many non-degenerate (closed) $d-$ simplices in $\mathbb{R}^{d+1}$.

We let $\mathcal{T}_h$ denote the set of these simplices:

$$\Gamma_h = \bigcup_{T \in \mathcal{T}_h} T, \qquad |\mathcal{T}_h| = K\,. \tag{4.9}$$

A triangulation $\mathcal{T}_h$ is *conforming* if it satisfies the following property: for $T_i, T_j \in \mathcal{T}_h$, with $i \neq j$, the intersection $T_i \cap T_j$ is either empty or a proper subsimplex common to both T_i and T_j. In other words, a conforming mesh means there are no vertices in the interior of edges. If this condition is not met, the triangulation is referred to as *non-conforming*. The triangulation defined here can be treated equivalently as simplicial complex as it is a finite set of simplices satisfying Definition 68. Let $T \in \mathcal{T}_h$ be a k-dimensional simplex, where $k \leq d$, with vertices $[\mathbf{x}_0, \ldots, \mathbf{x}_k] \in \mathbb{R}^d$. We define the *diameter* of T by

$$\operatorname{diam}(T) = \max_{0 \leq i < j \leq k} \|\mathbf{x}_i - \mathbf{x}_j\|,$$

that is, the length of the longest edge. In particular, if T consists of a single vertex, then $\operatorname{diam}(T) = 0$.

Additionally,

$$h = \max_{T \in \mathcal{T}_h} \operatorname{diam}(T), \quad r = \min_{T \in \mathcal{T}_h} \rho(T).$$

and the quantity

$$\sigma = \max_{T \in \mathcal{T}_h} \sigma(T), \quad \sigma(T) = \frac{h(T)}{r(T)}$$

is bounded independently of the mesh size h.

4.2.2 Projection to zero-level set

Determining the closest point projection onto a surface, denoted by $\pi(\mathbf{x})$, is a fundamental operation in many applications. While closed-form solutions exist for simple surfaces such as spheres and tori, for more complex surfaces, especially those implicitly defined as the zero-level set of a function, the closest-point projection must be determined iteratively. This can be achieved using methods such as Newton's method or fixed-point iteration. The essential property used in these algorithms is that the normal vector to the surface is given by the normalized gradient of the implicit function.

Let us assume that our surface is given as $\Gamma = \{\mathbf{x} \in \mathbb{R}^{d+1} : \varphi(\mathbf{x}) = 0\}$. Given an initial guess for the projected point $\mathbf{x}_0$, the authors of [103] describe a scheme to iteratively compute better approximations of the projection of $\mathbf{x}_0$ to Γ. The scheme approximates the closest-point property $\pi(\mathbf{x}) = \mathbf{x} - d(\mathbf{x})\mathbf{n}(\pi(\mathbf{x}))$ using the approximate distance $d(\mathbf{x}) \approx \frac{\varphi(\mathbf{x})}{\|\nabla\varphi(\mathbf{x})\|}$ and the normal vector $\mathbf{n}(\mathbf{x}) = \frac{\nabla\varphi(\mathbf{x})}{\|\nabla\varphi(\mathbf{x})\|}$, for $\mathbf{x} \in \Gamma$. This leads to an iterative scheme inspired by gradient descent:

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \nabla\varphi(\mathbf{x}_i)\frac{\varphi(\mathbf{x}_i)}{\|\nabla\varphi(\mathbf{x}_i)\|^2}. \tag{4.10}$$

This scheme iteratively updates the estimate of the projected point by moving along the negative gradient direction, with the step size proportional to the approximate distance to the surface.

While the gradient descent approach is computationally efficient, it may not always converge to the true closest point. To address this, an improved scheme, as proposed in [39], refines the initial estimate obtained from the gradient descent step.

This refined scheme proceeds as follows:

$$\tilde{\mathbf{x}}_{i+1} = \mathbf{x}_i - \nabla\varphi(\mathbf{x}_i)\frac{\varphi(\mathbf{x}_i)}{\|\nabla\varphi(\mathbf{x}_i)\|^2}, \quad \text{dist} = \text{sign}(\varphi(\mathbf{x}_0))\|\tilde{\mathbf{x}}_{i+1} - \mathbf{x}_0\|.$$

$$\mathbf{x}_{i+1} = \mathbf{x}_0 - \nabla\varphi(\tilde{\mathbf{x}}_{i+1})\frac{\text{dist}}{\|\nabla\varphi(\tilde{\mathbf{x}}_{i+1})\|}$$

First, the gradient descent step is performed to obtain an initial estimate $\tilde{\mathbf{x}}_{i+1}$. Then, this point is used as an initial approximation of the closest-point projection to obtain a better estimate of the actual distance from $\mathbf{x}_0$ to Γ. The improved scheme generally exhibits better convergence properties, albeit at a higher computational cost. The choice of initial guess $\mathbf{x}_0$ can significantly influence the convergence behavior and speed. The accuracy of the projection depends on the

smoothness of the implicit function $\varphi(\mathbf{x})$ and the chosen stopping criteria, such as an error estimate err_i or the distance to the surface.

Chapter 5

Integration on regular surfaces

In computational mathematics, accurate numerical integration techniques are crucial for a broad range of numerical methods, including the boundary integral method, the finite element method, surface finite elements, integral transforms, and the finite volume method. This chapter addresses the problem of integrating functions over triangulated curved surfaces using high-order methods. It provides a detailed account of the work presented in three of my published papers [145–147], each of which examines the challenges associated with attaining optimal convergence rates under both h- and p-refinement.

Piecewise polynomial surface parametrization using Lagrange interpolation with equidistant points shows an error behavior that is different for even and odd polynomial orders. This phenomenon was first noted numerically in the work of [109], where it was observed—though not theoretically explained—that even-degree polynomials often yield better convergence rates than their odd-degree counterparts. Further numerical evidence is provided in [66], where Hardering and Praetorius explicitly discuss this behavior. In Section 5.1 of [66], they state:

> *A fourth observation in the data shown in Table 1 is that the numerical convergence rates for even k_g and small values of α are often better than the expected convergence orders. This behavior is not explained in the analysis section of this paper. We have observed experimentally that for even k_g the non-standard geometry estimates (11) behave even better and seem to be of order $k_g + 2$. This observation needs further investigation that goes beyond the scope of this paper.*

Despite these experimental findings, the underlying theoretical justification for this behavior has remained an open question. Motivated by these observations, this chapter first addresses this gap by studying the cancellation mechanism that appears for equidistant interpolation

nodes on refined triangular meshes. The analysis is carried out for both scalar and vector integrands.

Theorems 78 and 82 establish rigorous error estimates for integration over discrete triangulated surfaces, providing a theoretical foundation for the superior performance of even-degree polynomials and thereby proving Conjecture 72. These contributions not only resolve the open question but also clarify the role played by mesh symmetry and interpolation degree in high-order surface quadrature.

The first part of the chapter also exposes a limitation of simplex-based interpolation: even when the convergence mechanism is understood, equidistant triangular nodes remain vulnerable to Runge's phenomenon at high polynomial degree. This motivates the second part of the chapter. In Section 5.5, we present a methodology for constructing stable, high-order surface representations by pulling the interpolation problem back from triangles to tensor-product domains. These representations serve as the basis for deriving high-order volume elements (HOVE) used in the integration of scalar- and vector-valued functions over regular embedded surfaces. At the core of this approach lies the *square-squeezing* technique, which reparametrizes flat triangulations into cubical meshes. By interpolating on Chebyshev–Lobatto grids, the method mitigates Runge's phenomenon and provides closed-form expressions for high-order quadrature. Newly derived error bounds demonstrate exponential convergence for smooth integrals, with p-refinements significantly outperforming h-refinements.

The third part of the chapter then accelerates the same HOVE framework through spectral differentiation. Leveraging Chebyshev–Lobatto nodes, this method constructs Chebyshev differentiation matrices to compute partial derivatives of k-order cubical reparametrization maps. These derivatives are used to approximate the metric tensor g_i for each element, as detailed in Section 5.6. Thus, the chapter moves from an explanation of the even–odd convergence effect, to a stable tensor-product representation, and finally to a faster spectral implementation. Section 5.8 closes with potential directions for future work.

5.1 Related work

Prior to presenting the proposed approaches, it is essential to understand the current landscape of numerical integration techniques for curved surfaces. To contextualize the discussion, we first review several key developments addressing the computation of integrals on surfaces. These methods, which range from mesh-free approaches to high-order polynomial interpolation techniques, provide the foundation for our investigation and highlight the need for more robust and efficient solutions.

R1) Belytschko et al. [104] explore mesh-free methods like *moving least squares* (MLS), which provide a robust framework for approximating integrals, especially those with discontinuous functions. Mesh-free methods avoid the complications of mesh generation and element connectivity, making them effective for dynamic simulations. However, partition of unity is

essential for stabilizing these methods, and computational costs can rise significantly when handling discontinuities or irregular geometries. A notable limitation of these approaches is the instability of regression methods when faced with complex integral partitions, as discussed by Belytschko and his team.

R2) Ray et al. [109] realize *High-Order Integration over Discrete Triangulated Surfaces* (IDS) based on stabilized least squares, deriving k^{th}-order surface approximations. While the stabilized least-squares regression avoids Runge's phenomenon the computational costs rapidly increase with the order of the approximation. Recent extensions [85] address the task of computing integrals over non-parametrized surfaces.

R3) Piecewise polynomial approximations of regular hyper-surfaces $\Gamma = F_\Gamma^{-1}(0)$ in $\mathbb{R}^3$ are studied by Dziuk and Elliot [44]. Realizations are given by Demlow [38], Chien and Atkinson [6], and Praetorius and Stenger [106]. However, all approaches rest on interpolation in equidistant nodes on simplices. Consequently, they are sensitive to Runge's phenomenon and become unstable for high orders. An extended investigation of the error analysis provided by [38, 44, 106] is given in [147].

R4) Reeger et al. [110] propose to use local radial basis function-generated finite differences (RBD-FD) for efficiently generating quadrature weights for arbitrary node sets. This enables the approximation of surface integrals for any given function data.

While **R2)** and **R4)** address the harder problem of integrating f based on samples of f given at particular point sets, even in the case of regular surface integrals all approaches are limited to achieve prior specified algebraic approximation rates.

5.2 Integrals over smooth surfaces

Simplex meshes offer several advantages over cube meshes, particularly when dealing with general domains. Their primary benefit lies in their flexibility in conforming to intricate shapes, making them a natural starting point for our study [103].

Let Γ be a smooth, d-dimensional surface embedded in $\mathbb{R}^{d+1}$. We aim to compute integrals over Γ. To facilitate numerical integration, we decompose Γ into a finite union of non-overlapping regions:

$$\Gamma = \bigcup_{i=1}^{K} \overline{V_i}, \quad V_i \cap V_j = \emptyset \quad \text{for } i \neq j,$$

where each V_i is an open subset of Γ, and K denotes the number of surface patches in the decomposition.

To proceed, we introduce a parametrization for each region V_i. Let

$$\rho_i : \triangle_d \longrightarrow V_i \subseteq \Gamma, \tag{5.1}$$

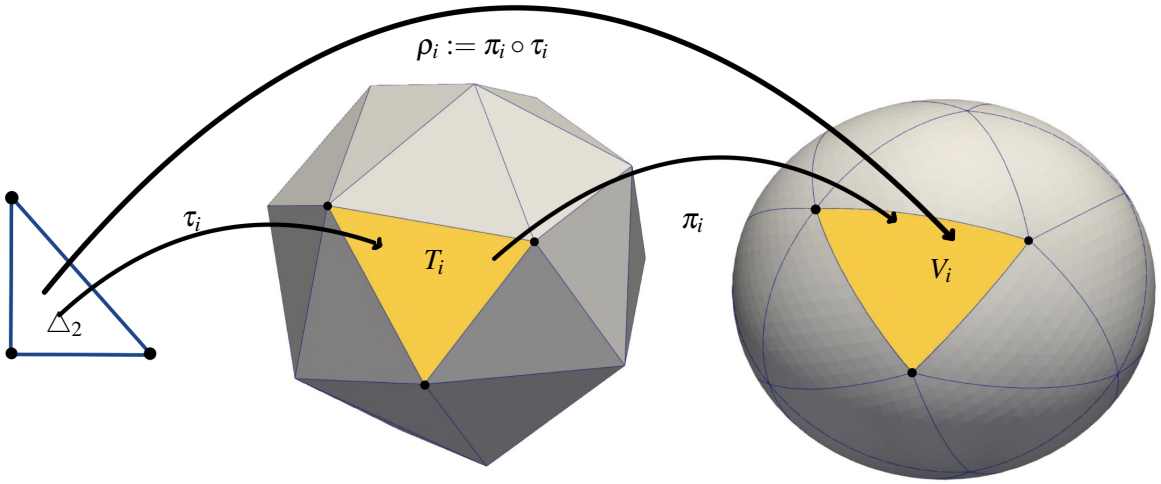


Figure 5.1: Representation of a smooth surface parametrization, where every region V_i forms a curved triangle.

be a smooth, bijective map, where $\triangle_d$ is the standard d-dimensional simplex. We write $\mathbf{y} \in \triangle_d$ for reference coordinates and $\mathbf{x} = \rho_i(\mathbf{y}) \in V_i$ for the corresponding point on the surface. The Jacobian matrix of this parametrization, denoted by $\mathbf{D}\rho_i(\mathbf{y})$, plays a crucial role in transforming integrals from the surface to the simplex domain.

Recalling that the *first fundamental form*, denoted by $\mathbf{g} \in \mathbb{R}^{d\times d}$, is defined in equation (4.4), we now turn our attention to the surface normal. Let $\mathbf{N}(\mathbf{x})$ denote a normal vector to Γ at the point $\mathbf{x} = \rho_i(\mathbf{y}) \in \Gamma$, and define the corresponding unit normal by $\mathbf{n}(\mathbf{x}) := \frac{\mathbf{N}(\mathbf{x})}{\|\mathbf{N}(\mathbf{x})\|}$. The *volume element* $g_i(\mathbf{y})$ is the local surface measure induced by the parametrization. It is the volume of the parallelepiped in the tangent plane to Γ spanned by the tangent vectors $\{\partial_\alpha \rho_i(\mathbf{y})\}_{\alpha=1}^d$:

$$g_i(\mathbf{y}) := \det\left([\mathbf{n}(\rho_i(\mathbf{y})), \mathbf{D}\rho_i(\mathbf{y})]\right) \quad \forall \mathbf{y} \in \triangle_d.$$

To obtain a more familiar form of g we argue as follows:

$$g_i(\mathbf{y}) = \frac{1}{|\mathbf{N}|}\det([\mathbf{N}, \mathbf{D}\rho_i]) = \frac{1}{|\mathbf{N}|}\det\left([\mathbf{N}, \mathbf{D}\rho_i]^t[\mathbf{N}, \mathbf{D}\rho_i]\right)^{\frac{1}{2}} = \sqrt{\det \mathbf{g}_i(\mathbf{y})}, \tag{5.2}$$

because

$$\det\left([\mathbf{N}, \mathbf{D}\rho_i]^t[\mathbf{N}, \mathbf{D}\rho_i]\right) = |\mathbf{N}|^2 \det(\mathbf{D}\rho_i^t \mathbf{D}\rho_i) = |\mathbf{N}|^2 \det \mathbf{g}_i. \tag{5.3}$$

An integrable function $\tilde{v} : V_i \to \mathbb{R}$ induces its pullback $v : \triangle_d \to \mathbb{R}$ by composition,

$$v(\mathbf{y}) = \tilde{v}(\rho_i(\mathbf{y})).$$

Conversely, a function v on $\triangle_d$ defines a corresponding function on V_i by $\tilde{v}(\mathbf{x}) = v(\rho_i^{-1}(\mathbf{x}))$. The area element allows for integration over V_i via the formula

$$\int_{V_i} \tilde{v}\, dS = \int_{\triangle_d} v(\mathbf{y}) g_i(\mathbf{y})\, d\mathbf{y} \quad \forall v \in L_1(\triangle_d). \tag{5.4}$$

For immersed surfaces $\Gamma \subset \mathbb{R}^{d+1}$, the parametrizations ρ_i allow us to compute integrals simplex by simplex. As a result, the problem of surface integration becomes:

Definition 69. *Given a conforming triangulation of* Γ*, the integral of an integrable function* $f : \Gamma \to \mathbb{R}$ *is*

$$\int_\Gamma f dS = \sum_{i=1}^{K} \int_{\triangle_d} f\big(\rho_i(\mathbf{y})\big) g_i(\mathbf{y}) d\mathbf{y}, \tag{5.5}$$

with $g_i(\mathbf{y}) = \sqrt{\det \mathbf{g}_i(\mathbf{y})}$*,* $i = 1, \cdots, K$*, and* $\mathbf{g}_i$ *the first fundamental form given elementwise.*

Remark 70. *This decomposition relies on the existence of a smooth partition of unity subordinate to the open cover* $\{V_i\}_{i=1}^K$*. A partition of unity is a collection of smooth functions* $\{\chi_i\}_{i=1}^K$ *such that:*

- ***Non-negativity:*** $\chi_i(\mathbf{x}) \geq 0$ *for all* $\mathbf{x} \in \Gamma$ *and all* $i = 1, \ldots, K$*.*
- ***Local Support:*** *Each function* χ_i *has compact support contained within* V_i*.*
- ***Partition of Unity:*** $\sum_{i=1}^K \chi_i(\mathbf{x}) = 1$ *for all* $\mathbf{x} \in \Gamma$*.*

The existence of a smooth partition of unity is guaranteed for any smooth surface. Using the partition of unity, we can express the integral over Γ as follows:

$$\int_\Gamma f \, d\mathbf{x} = \int_\Gamma f \left(\sum_{i=1}^{K} \chi_i(\mathbf{x}) \right) d\mathbf{x} = \sum_{i=1}^{K} \int_\Gamma f \chi_i(\mathbf{x}) d\mathbf{x}.$$

Since χ_i has compact support in V_i, the integral $\int_\Gamma f(\mathbf{x}) \chi_i(\mathbf{x}) d\mathbf{x}$ can be computed using the parametrization ρ_i, leading to the integral decomposition presented in equation (5.5).

Remark 71. *In practice, triangulations of an embedded surface* Γ *is given as the union of finitely many non-degenerate (closed)* $d-$ *simplices (4.9) in the embedding space* $\mathbb{R}^{d+1}$*. In view of (4.9), the discrete first fundamental form* $\mathbf{g}_T$ *and area element* g_T *of* Γ_h *are given element-wise by*

$$\mathbf{g}_T := \mathbf{D}\rho_T(\mathbf{x})^T \mathbf{D}\rho_T(\mathbf{x}), \quad g_T := \sqrt{\det \mathbf{g}_T}, \quad \forall T \in \mathcal{T}_h, \tag{5.6}$$

and satisfy

$$eigen(\mathbf{g}_T) \approx h_T^2, \quad g_T \approx h_T^d, \quad \forall T \in \mathcal{T}_h. \tag{5.7}$$

They give rise to the piecewise constant functions $\mathbf{g}_{\Gamma_h} := \{\mathbf{g}_T\}_{T \in \mathcal{T}_h}$ *and* $g_{\Gamma_h} := \{g_T\}_{T \in \mathcal{T}_h}$*.*

The primary challenge in providing a numerical approximation of (5.5) is the determination of the unknown derivatives $\mathbf{D}\rho$ that appear in the volume element $g(\mathbf{x})$. Typically, ρ is approximated by the closest point projection $\rho(\mathbf{x}) \approx \mathbf{x} - d(\mathbf{x})\mathbf{n}(\mathbf{x})$, with $\mathbf{n}(\mathbf{x})$ representing the normal vector.

In such approximations, the Jacobian $\mathbf{D}\rho(\mathbf{x})$ is highly sensitive to the quality of the normal field $\mathbf{n}(\mathbf{x})$. While standard geometric shapes such as spheres and tori allow for an analytical determination of $\mathbf{n}(\mathbf{x})$, high-order approximations of $\mathbf{D}\rho(\mathbf{x})$ for more general surfaces cannot be achieved

using this method. For piecewise polynomial surface approximations, we adopt the general notation from [38, 106] and substitute the Jacobians $\mathbf{D}\rho$ with those of a polynomial approximation, typically obtained via interpolation on a set of interpolation points in $\triangle_d$.

Let $\mathbb{I}$ denote the (componentwise) Lagrange interpolation operator, and define $\mathbb{I}^k\rho := Q_{G_{d,k}}\rho \in \Pi_{d,k}(\triangle_d)$ as the kth-order Lagrange polynomial interpolation of the mapping ρ on the reference element $\triangle_d$, with Lagrange nodes positioned on the smooth surface Γ. A kth-order interpolation of the area element is then given by

$$g^k(\mathbf{x}) = \sqrt{\det\left(\mathbf{D}Q_{G_{d,k}}\rho_i(\mathbf{x})^T \mathbf{D}Q_{G_{d,k}}\rho_i(\mathbf{x})\right)}.$$

Consequently, the integral is approximated by numerically computing

$$\sum_{i=1}^{K} \int_{\triangle_d} Q_{G_{d,k}} f(\rho_i(\mathbf{x})) g_i^k(\mathbf{x})\, d\mathbf{x}, \tag{5.8}$$

where $Q_{G_{d,k}} f(\rho_i(\mathbf{x}))$ is a $k-$th order polynomial approximating the integrand $f : \Gamma \to \mathbb{R}$.

Having outlined the motivation for high-order integration on curved surfaces, we now introduce the local parameterization of surfaces, which forms the basis for transforming surface integrals into computationally manageable expressions. This parameterization is crucial for constructing high-order polynomial approximations, allowing us to investigate and prove the following conjecture:

Conjecture 72. *Let Γ be a smooth closed embedded hypersurface, and let $f \in C^{k+2}(\Gamma, \mathbb{R})$. Consider a piecewise linear triangulation Γ_h with mesh size h, whose vertices lie on Γ, and let $\Gamma_{h,k}$ be its k-th order polynomial approximation.*

If $Q_{G_{2,k}} f : \Gamma_{h,k} \to \mathbb{R}$ is a k-th order polynomial approximating f, then the integration error satisfies

$$\left| \int_\Gamma f\, dS - \int_{\Gamma_{h,k}} Q_{G_{2,k}} f\, dS \right| = \mathcal{O}\left(h^{k+\delta_k}\right), \quad \text{where} \quad \delta_k = \begin{cases} 2, & k \text{ is even}, \\ 1, & k \text{ is odd}. \end{cases} \tag{5.9}$$

Thus, even-degree polynomial approximations yield a higher order of convergence than odd-degree ones.

5.3 Local parameterization of surfaces

We adopt the viewpoint that the surface Γ is described as the deformation of an 2-dimensional polyhedral surface Γ_h by a globally bi-Lipschitz *homeomorphism*[1] $\rho : \Gamma_h \to \Gamma \subset \mathbb{R}^3$. Thus there exists $L > 0$ such that for all $\mathbf{x}_1, \mathbf{x}_2 \in \Gamma_h$

[1] A globally bi-Lipschitz homeomorphism "stretches" or "shrinks" distances between points uniformly. It doesn't allow for extreme distortions where distances between some points are dramatically increased or decreased compared to others

$$L^{-1}|\mathbf{x}_1-\mathbf{x}_2| \leq |\tilde{\mathbf{x}}_1-\tilde{\mathbf{x}}_2| \leq L|\mathbf{x}_1-\mathbf{x}_2|, \quad \tilde{\mathbf{x}}_i=\rho(\mathbf{x}_i), \quad i=1,2. \tag{64}$$

The (closed) facets of Γ_h are denoted by T, and form the collection $\mathcal{T}_h=\{T\}$,

$$\Gamma_h=\bigcup_{T\in\mathcal{T}_h} T, \quad |\mathcal{T}_h|=K.$$

The partition $\mathcal{T}_h$ of Γ_h induces the partition $\tilde{\mathcal{T}}_h=\{\tilde{T}\}_{T\in\mathcal{T}_h}$ of Γ upon setting

$$\tilde{T}:=\rho(T) \quad \forall T\in\mathcal{T}_h.$$

Assume that Γ_h is contained in the tubular neighborhood $\mathcal{N}_\delta$. Under these conditions, we define a unique nonlinear closest point projection map:

$$\pi:\mathcal{N}_\delta\supset\Gamma_h\longrightarrow\Gamma\subset\mathbb{R}^3 \tag{5.10}$$

by

$$\pi(\mathbf{x})=\mathbf{x}-d(\mathbf{x})\,\mathbf{n}(\mathbf{x}),$$

such that it assigns to every $\mathbf{x}\in\Gamma_h$ the closest point on Γ, so that $(\mathbf{x}-\pi(\mathbf{x}))\perp T_{\pi(\mathbf{x})}\Gamma$, $\forall\mathbf{x}\in\Gamma_h$. The computation of the closest point projection (5.10) is a local minimizer problem [19] in the sense:

$$\pi:\Gamma_h\ni\mathbf{x}\mapsto \underset{\tilde{\mathbf{x}}\in\Gamma}{argmin}\,\|\tilde{\mathbf{x}}-\mathbf{x}\|, \tag{5.11}$$

where the nonlinear projection maps a point $\mathbf{x}\in\Gamma_h$ to the point $\tilde{\mathbf{x}}$ on Γ that minimizes the distance to $\mathbf{x}$.

In the following proposition, we demonstrate that the meshes generated via the closest point projection method are also shape-regular.

Proposition 73. *Assuming that $\mathcal{T}_h=\{T\}$, is a shape-regular triangulation of an 2-dimensional polyhedral surface Γ_h, and $\tilde{\mathcal{T}}_h=\{\tilde{T}\}_{T\in\mathcal{T}_h}$ is a triangulation of the surface Γ such that $\tilde{T}=\pi(T)$, then $\tilde{T}$ is a shape-regular triangulation of the surface Γ.*

Proof. There are two points $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_2$ on the surface triangle $\tilde{T}$ such that $h_{\tilde{T}}=d(\tilde{\mathbf{x}}_1,\tilde{\mathbf{x}}_2)$, here $d(\tilde{\mathbf{x}}_1,\tilde{\mathbf{x}}_2)$ is the length of the shortest curve joining the points $\tilde{\mathbf{x}}_1$ and $\tilde{\mathbf{x}}_1$ on Γ. From [83, Lemma 6.2] , we get $d(\tilde{\mathbf{x}}_1,\tilde{\mathbf{x}}_2)<C\|\tilde{\mathbf{x}}_1-\tilde{\mathbf{x}}_2\|$. Let $\mathbf{x}_1$ and $\mathbf{x}_2$ be in the corresponding planar triangle T with $\tilde{\mathbf{x}}_1=\pi(\mathbf{x}_1)$ and $\tilde{\mathbf{x}}_2=\pi(\mathbf{x}_2)$. Since the projection operator is Lipschitz continuous, there exists a constant L_1 such that $\|\tilde{\mathbf{x}}_1-\tilde{\mathbf{x}}_2\|=\|\pi(\mathbf{x}_1)-\pi(\mathbf{x}_2)\|\leq L_1\|\mathbf{x}_1-\mathbf{x}_2\|$. Hence, $d(\tilde{\mathbf{x}}_1,\tilde{\mathbf{x}}_2)<C\|\tilde{\mathbf{x}}_1-\tilde{\mathbf{x}}_2\|\leq CL_1\|\mathbf{x}_1-\mathbf{x}_2\|$. Thus, $h(\tilde{T})\leq CL_1h(T)$.

Let $\mathbf{x}_0$ be the center of the inscribed circle in the planar triangle T, and set $\tilde{\mathbf{x}}_0=\pi(\mathbf{x}_0)$. Let L_2 be the Lipschitz constant of the inverse projection operator π^{-1}, then $r(T)=\inf_{\mathbf{x}\in\partial T}\|\mathbf{x}-\mathbf{x}_0\|$, and

$\inf_{\mathbf{y}\in\partial\tilde{T}} \|\pi^{-1}(\tilde{\mathbf{x}}) - \pi^{-1}(\tilde{\mathbf{x}}_0)\|$. Since π^{-1} is Lipschitz continuous, it follows that $r(T) \leq L_2 \inf_{\tilde{\mathbf{x}}\in\partial\tilde{T}} \|\tilde{\mathbf{x}} - \tilde{\mathbf{x}}_0\|$. Therefore, $\frac{h(\tilde{T})}{r(\tilde{T})} \leq CL_1L_2$. □

Next, we present a local parametric description of the surface. Let $\triangle_2 \subset \mathbb{R}^2$ denote the unit reference simplex, defined as

$$\triangle_2 := \{(s,t) \in \mathbb{R}^2 \mid 0 \leq s \leq 1, 0 \leq t \leq 1-s\},$$

and let $\tau : \mathbb{R}^2 \to \mathbb{R}^3$ be the affine map such that $T = \tau(\triangle_2)$.

We now construct a surface $\Gamma_{h,k}$, which is locally parametrized over $\triangle_2$ by polynomials of degree k, interpolating the smooth surface Γ. As previously mentioned, we start with a piecewise flat triangulation Γ_h of the smooth surface, where the vertices lie on Γ. To simplify the notation, the subscripts from the transformation map (5.1) and the triangulation of Γ_h are dropped. For any triangle $T \in \mathcal{T}_h$ with vertices $q_1, q_2, q_3 \in \mathbb{R}^3$ we define the following map

$$\tau : \triangle_2 \longrightarrow T, \quad q_i = \tau(\hat{p}_i), \quad 1 \leq i \leq 3 \quad \tau(s,t) = q_1 + (q_3 - q_1)s + (q_2 - q_1)t$$

to be the affine linear parametrization which maps each vertex $\hat{p}_i$ of $\triangle_2$ to the vertex q_i of T. We denote the images of the non-vertex nodes of $\triangle_2$ on the simplex T by $\bar{q}_i = \tau(\hat{p}_i), 3 < i \leq N(2,k)$, where $N(2,k)$ is the dimension of the vector space $\Pi_{2,k}(\triangle_2)$ of bivariate polynomials of degree k. Let $\mathcal{L}_1^k(\hat{p}), \mathcal{L}_2^k(\hat{p}), \ldots, \mathcal{L}_{N(2,k)}^k(\hat{p})$ be the local Lagrange basis functions of degree k on $\triangle_2$ corresponding to the nodal points $\hat{p}_1, \ldots \hat{p}_{N(2,k)}$.

We are now ready to introduce the local *nonoverlapping* parametrization ρ of Γ. Let $\rho := \pi \circ \tau : \triangle_2 \to \tilde{T}$ be the corresponding local parametrization of $\tilde{T}$ and $\rho := \{\rho_i\}_{i=1,\cdots,K}$;

$$h|\mathbf{w}| \lesssim |\mathbf{D}\rho(s,t)\mathbf{w}| \lesssim |T|^{\frac{1}{2}}|\mathbf{w}|, \quad \forall \mathbf{w} \in \mathbb{R}^2, (s,t) \in \triangle_2. \tag{5.12}$$

Set

$$p_i := \pi(\bar{q}_i) = (\pi \circ \tau)(\hat{p}_i) = \rho(\hat{p}_i), \quad \text{where,} \quad 3 < i \leq N(2,K)$$

and define $Q_{G_{2,k}}\rho$ to be a k^{th}-order vector-valued polynomial interpolation of the mapping ρ:

Definition 74. *The interpolation operator*

$$\mathbb{I}_k : C^0(\triangle_2, \mathbb{R}) \longrightarrow \Pi_{2,k}(\triangle_2)$$

$$\rho \mapsto Q_{G_{2,k}}\rho, \quad Q_{G_{2,k}}\rho(\hat{p}) := \sum_{i=1}^{N(2,K)} p_i \mathcal{L}_i^k(\hat{p}) \tag{5.13}$$

is called the Lagrange interpolation operator of degree k.

The mapping

$$Q_{G_{2,k}}\rho : \triangle_2 \to \tilde{T}^k := Q_{G_{2,k}}\rho(\triangle_2)$$

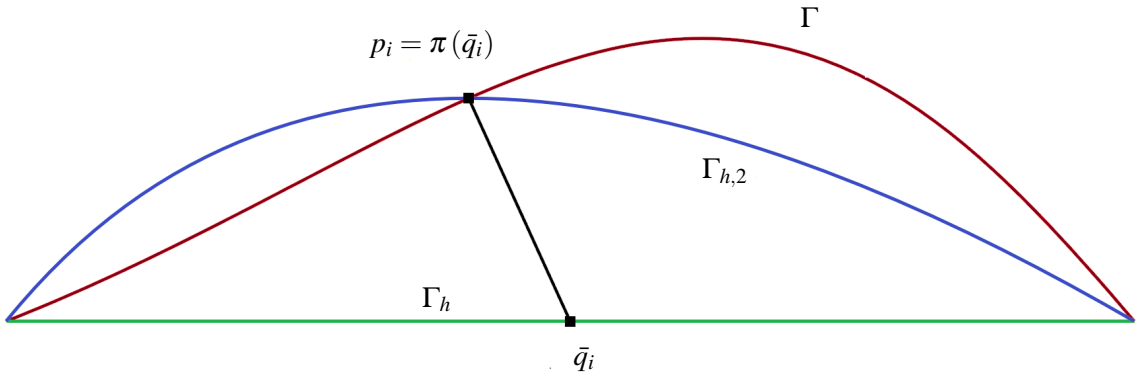


Figure 5.2: Construction of the second order approximation of the smooth surface $\Gamma_{h,2}$ (blue line). A simplex of a "base" triangulation Γ_h (green line) is shown. The interpolation nodes, here the center $\bar{q}_i$ of an edge, are projected (grey line) onto the smooth surface Γ (red line) via the projection π. The projected nodal points $\pi(\bar{q}_i)$ and the vertices of Γ_h are then interpolated, giving the second order approximation of the smooth surface $\Gamma_{h,2}$.

defines a polynomial mapping $Q_{G_{2,k}}\rho$ by interpolating the points $p_i \in \Gamma$ with the Lagrange-polynomials $\mathcal{L}_1^k(\hat{p}), \mathcal{L}_2^k(\hat{p}), \ldots, \mathcal{L}_{N(2,k)}^k(\hat{p})$. Thus, for every simplex $T \in \mathcal{T}_h$ we compute the projection $\pi(\bar{q}_i)$ and define an isoparametric simplex $\tilde{T}^k$ by applying Lagrange interpolation of order k to the coordinates of the projected equidistant nodes (see Figure 5.2). Furthermore, if the base-triangulation $\mathcal{T}_h$ is fine enough, then the map $Q_{G_{2,k}}\rho$ is a diffeomorphism. By differentiating the interpolation polynomial of the map ρ, we obtain:

$$\partial_s Q_{G_{2,k}}\rho(\hat{p}) := \sum_{i=1}^{N(2,K)} p_i \partial_s \mathcal{L}_i^k(\hat{p}), \quad \partial_t Q_{G_{2,k}}\rho(\hat{p}) := \sum_{i=1}^{N(2,K)} p_i \partial_t \mathcal{L}_i^k(\hat{p}). \tag{5.14}$$

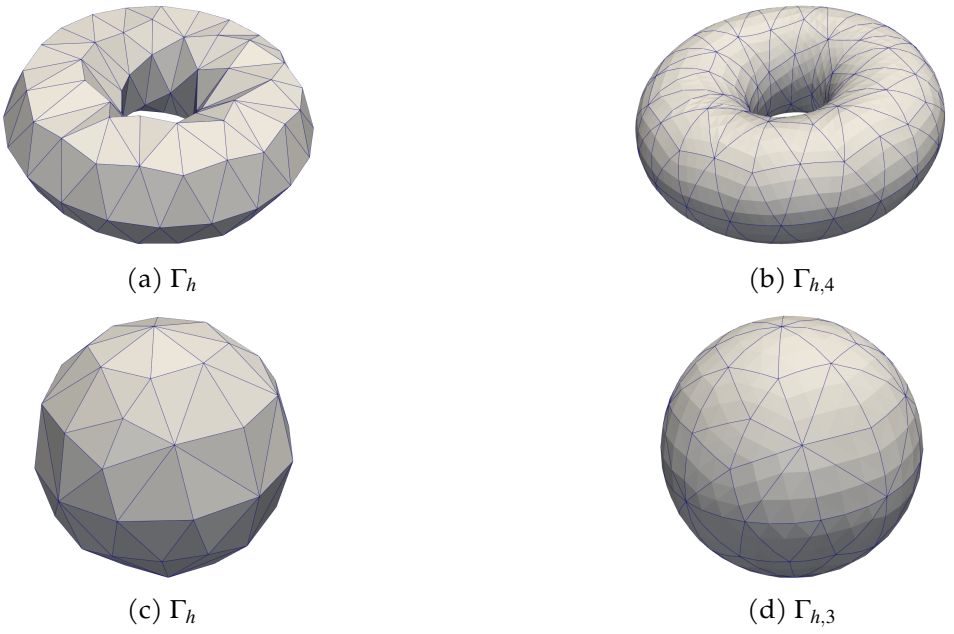
The Jacobian of the transformation is computed using equation (5.14). If $Q_{G_{2,k}}\rho$ is a diffeomorphism for every $T \in \mathcal{T}_h$, then the union of the non-overlapping mapped elements gives

$$\Gamma_{h,k} := \bigcup_{T\in\mathcal{T}_h} Q_{G_{2,k}}\rho(T) = \bigcup_{\substack{T\in\mathcal{T}_h \\ \hat{p}\in\triangle_2}} \sum_{i=1}^{N(2,K)} \pi\big(\tau(\hat{p}_i)\big)\mathcal{L}_i^k(\hat{p}). \tag{5.15}$$

Therefore, we have obtained a k^{th} order discrete approximation $\Gamma_{h,k}$ of the continuous surface Γ. The curved mesh is built on top of a piecewise flat reference grid provided by [56, 103]. Figure 5.3 depicts an illustration of this procedure applied to a torus and a sphere.

5.3.1 Odd behavior of parametrized geometries

In this section, we prove the main theorem on the accuracy of integration when replacing Γ with its k^{th} order polygonal approximation $\Gamma_{h,k}$. We show that the presence of symmetric triangles in the mesh, combined with even-degree polynomials, enhances the global error order from $k+1$ to $k+2$. We begin with the following definition.



(a) Γ_h (b) $\Gamma_{h,4}$

(c) Γ_h (d) $\Gamma_{h,3}$

Figure 5.3: Lagrange parametrization for a torus and a sphere using equidistant nodes.

Definition 75 (symmetric triangles)**.** *Given a collection of triangles $\mathcal{T}_h$ [111], a pair of triangles that belong to the same congruence class, $T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}$ and $T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5} \in \mathcal{T}_h$, as shown in Figure 5.4, are called symmetric with respect to their common vertex $\mathbf{x}_1$ if they satisfy the following property:*

$$(\mathbf{x}_3 - \mathbf{x}_1) = -(\mathbf{x}_5 - \mathbf{x}_1), \quad (\mathbf{x}_2 - \mathbf{x}_1) = -(\mathbf{x}_4 - \mathbf{x}_1). \tag{5.16}$$

Traditionally, the specific technique employed for triangulation refinement of a triangulation mesh $\mathcal{T}_h$ is often regarded as unimportant, provided that the grid size $h \longrightarrow 0$ as $n \longrightarrow \infty$. However, a more intricate picture emerges when the influence of the integration process is taken into account. The choice of the *appropriate* type of triangulation can potentially result in a serendipitous cancellation of errors. Now, when examining a conforming triangulation $\mathcal{T}_h$, we refine each triangle $T \in \mathcal{T}_h$ into four smaller triangles using uniform refinement, which involves connecting the midpoints of its edges with straight lines. The new elements are all congruent, and they are similar to T. In fact, let A, B, C be the vertices of the triangle T and let X, Y, Z be the midpoints of the edge AB, BC and AC. An application of uniform refinement produces the triangles $T_1 = [A, X, Z], T_2 = [X, B, Y], T_3 = [Z, Y, C]$ and $T_4 = [Y, Z, X]$.

We have line XY parallel to line AC, i.e., $XY \parallel AC$, so $\angle BXY = \angle XAZ$. Similarly, since $XZ \parallel BC$, we have $\angle XBY = \angle AXZ$. Since X is the midpoint of line AB, then $|AX| = |BX|$. In short, we have

$$\angle BXY = \angle XAZ, \quad |BX| = |AX|, \quad \angle XBY = \angle AXZ.$$

Therefore, $\triangle BXY \cong \triangle XAZ$. Similarly, we can prove the four triangles are congruent to each other, i.e., $\triangle BXY \cong \triangle XAZ \cong \triangle YZC \cong \triangle ZYX$, so they are in the same congruent class, which

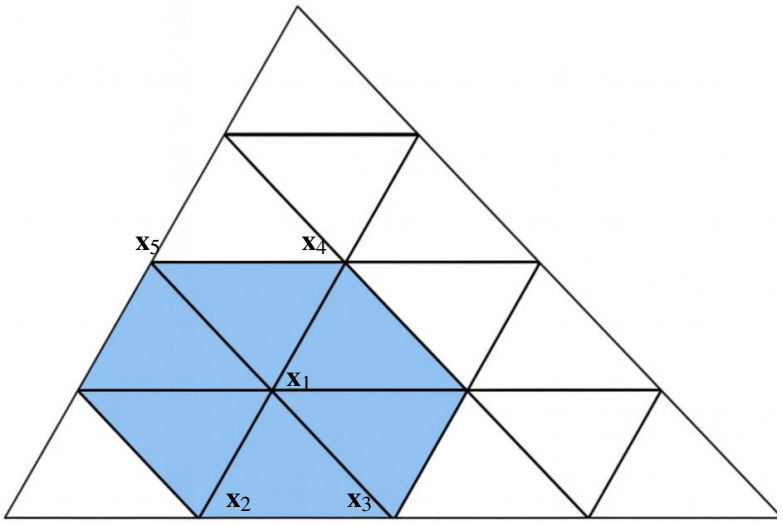


Figure 5.4: Illustration of uniform refinement; the blue elements indicate a set of symmetric triangles.

is the same as the one of the original triangle $\triangle BAC$. Thus, we have proved the following result

Lemma 76. *The triangle T_1, T_2, T_3 and T_4 produced by the uniform refinement have the same congruence class as T.*

Repeating the uniform refinement $m-$ times, the total number of triangles at that level will be 2^{2m}, which can be proved by induction . An advantage of this form of refinement is that each set of mesh points contains those mesh points at the preceding level. During this refinement procedure, triangular elements of different sizes are obtained. For example, Figure 5.4 shows three distinct triangular element sizes.

Consider $\mathcal{C}_m, m \in \mathbb{N}$ as the set of triangles with equal size within $T \in \mathcal{T}_h$ at the m-th refinement level, and let K_m represent the number of triangles in this set, then we can write

$$\mathcal{C}_m = \bigcup_{i=1}^{K_m} \{T^i_{h(\frac{1}{2})^m}\}, \text{ where } h = \text{diam}(T).$$

For example, from the illustration in Figure 5.4, we have

$$\mathcal{C}_0 = T,\ \mathcal{C}_1 = \bigcup_{i=1}^{K_1}\{T^i_{(\frac{h}{2})}\},\ \mathcal{C}_2 = \bigcup_{i=1}^{K_2}\{T^i_{h(\frac{1}{2})^2}\},\ \mathcal{C}_3 = \bigcup_{i=1}^{K_3}\{T^i_{h(\frac{1}{2})^3}\},\ \mathcal{C}_4 = \bigcup_{i=1}^{K_4}\{T^i_{h(\frac{1}{2})^4}\}.$$

$\mathcal{C}_1$ contains triangles in congruence class 1, while $\mathcal{C}_2$, $\mathcal{C}_3$, and $\mathcal{C}_4$ contain triangles in congruence classes 2, 3, and 4, respectively.

Observe that during each uniform refinement step, at least two triangles within a congruence class fail to meet the criteria specified in Definition 75. As a result, the density of symmetric pairs of triangles within $T \in \mathcal{T}_h$ after the m-th refinement level is $\mathcal{O}\left(\tilde{K}_m\right)$ while the remaining triangles exhibit a density of $\mathcal{O}\left(\sqrt{\tilde{K}_m}\right)$. Here, $\tilde{K}_m$ denotes the total number of triangles within $T \in \mathcal{T}_h$ after the m-th refinement.

Remark 77 (symmetric triangulation)**.** *Any (smoothed closed) surface triangulation mesh that has undergone the refinement process previously described will be referred to as a symmetric triangulation.*

In light of these facts, we state the main theorem of this article, showing that even degree approximation is superior to odd degree approximation for *symmetric triangulations*.

Theorem 78. *Let Γ be a smooth closed embedded hypersurface and $f \in C^{k+2}(\Gamma, \mathbb{R})$. Consider a piecewise linear triangulation Γ_h with mesh size h of the smooth surface having vertices lie on Γ and let $\Gamma_{h,k}$ be the $k-$th order approximation of the smooth surface constructed using local fittings of $\pi \in C^{k+3}\left(\Gamma_h, \mathbb{R}^3\right)$. Consider a symmetric triangulation, consisting of $\mathcal{O}(K) = \mathcal{O}\left(h^{-2}\right)$ symmetric triangles, whereas the number of non-symmetric triangles is bounded by $\mathcal{O}\left(\sqrt{K}\right) = \mathcal{O}\left(h^{-1}\right)$. Then*

$$\left|\int_\Gamma f\, dS - \int_{\Gamma_{h,k}} Q_{G_{2,k}} f\, dS\right| \leq \begin{cases} Ch^{k+2}, & k \equiv 0\ (mod\ 2) \\ Ch^{k+1}, & k \equiv 1\ (mod\ 2), \end{cases} \tag{5.17}$$

where $Q_{G_{2,k}} f : \Gamma_{h,k} \to \mathbb{R}$ is a $k-$th order polynomial approximating the integrand f.

Lemma 79. *Let T be a triangle in $\mathcal{T}_h$ and assume that $\rho \in C^{k+1}(T)$ and $k \geq 0$, then we have*

$$\left\|\rho - Q_{G_{2,k}}\rho\right\|_{C^0(T)} \leq ch^{k+1} \max_{l+p+q=k+1}^{l,p,q\geq 0} \max_{(x,y,z)\in T} \left|\frac{\partial^{k+1}\rho(x,y,z)}{\partial x^l \partial y^p \partial z^q}\right| \tag{5.18}$$

with $h = diam(T)$. The constant c depends on k, but it is independent of both ρ and T.

Proof. In order to prove this lemma, we will use Taylor's formula in several variables. Let's denote with $\boldsymbol{\alpha} = (\alpha_1, \alpha_2)$, where $\|\boldsymbol{\alpha}\|_1 = |\alpha_1| + |\alpha_2|$, $h := (h_1, h_2) = (s-0, t-0)$ and $\partial_{\alpha_1} = \partial_s$, $\partial_{\alpha_2} = \partial_t$.

As a first step, let's consider the $k-$ order interpolation of the map ρ

$$Q_{G_{2,k}}\rho(\hat{p}) := \sum_{i=1}^{N(2,K)} p_i \mathcal{L}_i^k(\hat{p}).$$

Using Taylor's formula for ρ around the point $(0,0)$ we obtain:

$$\rho(\hat{p}) = \sum_{\|\boldsymbol{\alpha}\|_1=k} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(0)\, h^{\boldsymbol{\alpha}}\right) + \frac{1}{k!}\int_0^1 (1-\mu)^k \frac{d^{k+1}\rho(s\mu,t\mu)}{d^{k+1}\mu} d\mu. \tag{5.19}$$

Let us denote $H(\hat{p}) := \frac{1}{k!}\int_0^1 (1-\mu)^k \frac{d^{k+1}\rho(s\mu,t\mu)}{d^{k+1}\mu} d\mu$, by interpolating each term in equation (5.19) using a polynomial of order k and using the fact that the interpolation of the first term is exact because it is a polynomial of degree $\leq k$, namely

$$\mathbb{I}^k\Big(\sum_{\|\boldsymbol{\alpha}\|_1=k} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(0)\, h^{\boldsymbol{\alpha}}\right)\Big) = \sum_{\|\boldsymbol{\alpha}\|_1=k} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(0)\, h^{\boldsymbol{\alpha}}\right), \tag{5.20}$$

we obtain

$$Q_{G_{2,k}}\rho(\hat{p}) = \sum_{\|\boldsymbol{\alpha}\|_1=k} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(0)\, h^{\boldsymbol{\alpha}}\right) + \sum_{i=1}^{N(2,K)} H(\hat{p}_i)\, \mathcal{L}_i^k(\hat{p}). \tag{5.21}$$

Subtracting equation (5.21) from equation (5.19), we have

$$\rho(\hat{p}) - Q_{G_{2,k}}\rho(\hat{p}) = H(\hat{p}) - \sum_{i=1}^{N(2,K)} H(\hat{p}_i)\, \mathcal{L}_i^k(\hat{p}). \tag{5.22}$$

The right-hand side of equation (5.22) can be written

$$H(\hat{p}) - \sum_{i=1}^{N(2,K)} H(\hat{p}_i)\, \mathcal{L}_i^k(\hat{p}) = \frac{1}{k!}\int_0^1 (1-\mu)^k E(\mu;s,t)\, d\mu, \tag{5.23}$$

where $E(\mu;s,t)$ has the following form

$$E(\mu;s,t) := \frac{d^{k+1}\rho(s\mu,t\mu)}{d^{k+1}\mu} - \sum_{i=1}^{N(2,K)} \mathcal{L}_i^k(\hat{p})\, \frac{d^{k+1}\rho(V_i\mu,t_i\mu)}{d^{k+1}\mu}. \tag{5.24}$$

It is important to note that the left-hand side of equation (5.22) is affected by the behavior of the term $E(\mu;s,t)$, which itself is influenced by $\frac{d^{k+1}\rho(s\mu,t\mu)}{d^{k+1}\mu}$. For $k=0$, we have

$$\max_{\substack{0\leq\mu\leq 1\\ \hat{p}\in\triangle_2}} \left|\frac{\mathbf{D}\rho(s\mu,t\mu)}{d\mu}\right| \leq ch \max\left\{\left\|\frac{\partial\rho}{\partial x}\right\|_{L^\infty}, \left\|\frac{\partial\rho}{\partial y}\right\|_{L^\infty}, \left\|\frac{\partial\rho}{\partial z}\right\|_{L^\infty}\right\}, \tag{5.25}$$

where

$$\left\|\frac{\partial\rho}{\partial x}\right\|_{L^\infty} := \max_{(x,y,z)\in T}\left|\frac{\partial\rho}{\partial x}\right|$$

and analogously for $\frac{\partial\rho}{\partial y}$, $\frac{\partial\rho}{\partial z}$. Following the same line of reasoning with the higher-order derivatives of ρ, we obtain

$$\max_{\substack{0\leq\mu\leq 1\\ \hat{p}\in\triangle_2}} \left|\frac{d^{k+1}\rho(s\mu,t\mu)}{d^{k+1}\mu}\right| \leq ch^{k+1} \max\left\{\left\|\frac{\partial\rho}{\partial x}\right\|_{L^\infty}, \left\|\frac{\partial\rho}{\partial y}\right\|_{L^\infty}, \left\|\frac{\partial\rho}{\partial z}\right\|_{L^\infty}\right\}. \tag{5.26}$$

Using equation (5.23), we obtain

$$H(\hat{p}) - \sum_{i=1}^{N(2,K)} H(\hat{p}_i)\,\mathcal{L}_i^k(\hat{p}) = \mathcal{O}\left(h^{k+1}\right). \tag{5.27}$$

Combining equation (5.22) with equation (5.27) yields equation (5.18). □

Next, we recall the following lemma from [33].

Lemma 80 (Cancellation property of even-degree interpolation)**.** *Let k be an even integer. Let $\rho(\hat{p}) \in \Pi_{2,k+1}$ be a polynomial of degree $k+1$ on $\triangle_2$, and let $Q_{G_{2,k}}\rho(\hat{p})$ be its interpolant of degree k. Then, for each $\hat{p} \in \triangle_2$*

$$\int_{\triangle_2} \partial_s\left(\rho(\hat{p}) - Q_{G_{2,k}}\rho(\hat{p})\right) dsdt = 0, \quad \int_{\triangle_2} \partial_t\left(\rho(\hat{p}) - Q_{G_{2,k}}\rho(\hat{p})\right) dsdt = 0. \tag{5.28}$$

Lemma 81. *Let Γ be a smooth closed hypersurface and consider a piecewise linear triangulation Γ_h with mesh size h of the smooth surface Γ and let $\Gamma_{h,k}$ be the $k-$th order approximation of the smooth surface constructed using local fittings of $\pi \in C^{k+3}\left(\Gamma_h, \mathbb{R}^3\right)$. Consider a symmetric triangulation, consisting of $\mathcal{O}(K) = \mathcal{O}\left(h^{-2}\right)$ symmetric triangles, whereas the number of non-symmetric triangles is bounded by $\mathcal{O}\left(\sqrt{K}\right) = \mathcal{O}\left(h^{-1}\right)$. Then*

$$\left|\int_{\Gamma} dS - \int_{\Gamma_{h,k}} dS\right| \leq \begin{cases} Ch^{k+1}, & k \equiv 1 \ (mod\ 2) \\ Ch^{k+2}, & k \equiv 0 \ (mod\ 2). \end{cases} \tag{5.29}$$

Proof. First, let us write every term in equation (5.29) over a reference simplex

$$\int_{\Gamma} dS = \sum_{i=1}^{K} \int_{\triangle_2} g_i dsdt, \quad \int_{\Gamma_{h,k}} dS = \sum_{i=1}^{K} \int_{\triangle_2} g_i^k dsdt. \tag{5.30}$$

In the same manner, as in Lemma 79, we obtain

$$\rho(\hat{p}) - Q_{G_{2,k}}\rho(\hat{p}) = \mathcal{R}_{k+1} + \mathcal{R}_{k+2} + \mathcal{O}\left(h^{k+3}\right), \tag{5.31}$$

where

$$\mathcal{R}_{k+1} := \sum_{\|\boldsymbol{\alpha}\|_1 = k+1} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(\hat{p})\,h^{\boldsymbol{\alpha}} - \sum_{i=1}^{N(2,K)} h^{\boldsymbol{\alpha}} \frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(\hat{p})\,\mathcal{L}_i^k(\hat{p})\right)$$

$$\mathcal{R}_{k+2} := \sum_{\|\boldsymbol{\alpha}\|_1 = k+2} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(\hat{p})\,h^{\boldsymbol{\alpha}} - \sum_{i=1}^{N(2,K)} h^{\boldsymbol{\alpha}} \frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!}\rho(\hat{p})\,\mathcal{L}_i^k(\hat{p})\right).$$

Analogously, an expansion can be given for the errors in the partial derivatives of $\partial_{\alpha_1}\rho(\hat{p})$, $\partial_{\alpha_2}\rho(\hat{p})$, amounting to apply the derivative on (5.31). Therefore, we have

$$\partial_{\alpha_1}\rho(\hat{p}) - \partial_{\alpha_1} Q_{G_{2,k}}\rho(\hat{p}) = \partial_{\alpha_1}\mathcal{R}_{k+1} + \partial_{\alpha_1}\mathcal{R}_{k+2} + \mathcal{O}\left(h^{k+2}\right) \tag{5.32}$$

$$\partial_{\alpha_2}\rho\left(\hat{p}\right)-\partial_{\alpha_2}Q_{G_{2,k}}\rho\left(\hat{p}\right)=\partial_{\alpha_2}\mathcal{R}_{k+1}+\partial_{\alpha_2}\mathcal{R}_{k+2}+\mathcal{O}\left(h^{k+2}\right). \tag{5.33}$$

Using Taylor's theorem and expanding about $(s=0,t=0)$, we obtain

$$\begin{aligned} g_i-g_i^k &:= \sqrt{\det \mathbf{g}_i}-\sqrt{\det \mathbf{g}_i^k} \\ &= \mathcal{E}\left(h^{k+2};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)+\mathcal{E}\left(h^{k+3};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)+\mathcal{O}\left(h^{k+4}\right), \end{aligned} \tag{5.34}$$

where $\mathcal{E}\left(h^{k+2};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)$ and $\mathcal{E}\left(h^{k+3};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)$, describes the collection of terms with order $k+2,k+3$ in h, whose coefficients are combinations of the vertices of triangles with appropriate indices j, ℓ, and k. For example for a triangle $T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}$ the coefficients are combination of $(\mathbf{x}_3-\mathbf{x}_1)$ and $(\mathbf{x}_2-\mathbf{x}_1)$ both for $\mathcal{E}\left(h^{k+2};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)$ and $\mathcal{E}\left(h^{k+3};\left(\mathbf{x}_k-\mathbf{x}_j\right),\left(\mathbf{x}_\ell-\mathbf{x}_j\right)\right)$. We write $\mathcal{E}(k+r)$ for terms of order h^{k+r} whose coefficients depend on combinations of the triangle vertices. The computation above shows that $|g_i-g_i^k|$ is at least of order $\mathcal{O}\left(h^{k+2}\right)$, confirming also the result obtained in [109, Theorem 7]. However, this result can be further improved assuming that the triangulation mesh comprises symmetric triangles. Thus, integrating equation (5.34) over a reference simplex, we have

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt=\int_{\triangle_2}\mathcal{E}\left(k+2\right)dsdt+\int_{\triangle_2}\mathcal{E}\left(k+3\right)dsdt+\mathcal{O}\left(h^{k+4}\right). \tag{5.35}$$

Now if k is odd, yields

$$\int_{\triangle_2}\mathcal{E}\left(k+2\right)dsdt\neq 0$$

then the left-hand side of equation (5.35) is at least of order $\mathcal{O}\left(h^{k+2}\right)$ for every $T\in\mathcal{T}_h$. Writing equation (5.35) with respect to a pair of symmetric triangles $T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}$ and $T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}$, we obtain

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}}=\int_{\triangle_2}\mathcal{E}\left(k+2\right)dsdt+\int_{\triangle_2}\mathcal{E}\left(k+3\right)dsdt+\mathcal{O}\left(h^{k+4}\right) \tag{5.36}$$

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}}=\int_{\triangle_2}\mathcal{E}\left(k+2\right)dsdt+\int_{\triangle_2}\mathcal{E}\left(k+3\right)dsdt+\mathcal{O}\left(h^{k+4}\right). \tag{5.37}$$

Summing the contributions of the two symmetric triangles yields

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}}+\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}}=\int_{\triangle_2}\mathcal{E}(k+3)\,dsdt+\mathcal{O}(h^{k+4}). \tag{5.38}$$

This follows because the first right-hand-side integrands in equations (5.36) and (5.37) collect the terms of order $k+2$ in h. Their coefficients are combinations of triangle vertices, and each such integrand is an odd function. In general for a triangle $T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}$ we may write:

$$\mathcal{E}\left(h^{k+2};-\left(\mathbf{x}_3-\mathbf{x}_1\right),-\left(\mathbf{x}_2-\mathbf{x}_1\right)\right)=-\mathcal{E}\left(h^{k+2};\left(\mathbf{x}_3-\mathbf{x}_1\right),\left(\mathbf{x}_2-\mathbf{x}_1\right)\right). \tag{5.39}$$

Similarly, for $T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}$ we have

$$\mathcal{E}\left(h^{k+2};-(\mathbf{x}_4-\mathbf{x}_1),-(\mathbf{x}_5-\mathbf{x}_1)\right)=-\mathcal{E}\left(h^{k+2};(\mathbf{x}_4-\mathbf{x}_1),(\mathbf{x}_5-\mathbf{x}_1)\right). \tag{5.40}$$

Using the property (5.16), equation (5.39) and equation (5.40), we obtain

$$\mathcal{E}\left(h^{k+2};(\mathbf{x}_3-\mathbf{x}_1),(\mathbf{x}_2-\mathbf{x}_1)\right)+\mathcal{E}\left(h^{k+2};(\mathbf{x}_4-\mathbf{x}_1),(\mathbf{x}_5-\mathbf{x}_1)\right)=0.$$

The first part of inequality (5.29) is obtained by leveraging the fact that the number of symmetric triangles in the mesh Γ_h is of order $\mathcal{O}(K)=\mathcal{O}\left(h^{-2}\right)$, while the number of non-symmetric triangles is bounded by $\mathcal{O}\left(\sqrt{K}\right)=\mathcal{O}\left(h^{-1}\right)$, in combination with equation (5.35) and equation (5.38).

At this point, we notice that if k is even, then

$$\int_{\triangle_2}\mathcal{E}(k+2)\,dsdt=0.$$

This follows from Lemma 80 because $\mathcal{E}(k+2)$ represents errors when integrating a polynomial of order $k+1$. Thus, we have

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt=\int_{\triangle_2}\mathcal{E}(k+3)\,dsdt+\mathcal{O}\left(h^{k+4}\right). \tag{5.41}$$

Writing equation (5.41) for the symmetric pair $T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}$ and $T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}$, we obtain

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}}=\int_{\triangle_2}\mathcal{E}(k+3)\,dsdt+\mathcal{O}\left(h^{k+4}\right) \tag{5.42}$$

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}}=\int_{\triangle_2}\mathcal{E}(k+3)\,dsdt+\mathcal{O}\left(h^{k+4}\right). \tag{5.43}$$

If k is even and the triangles are pairwise symmetric, then the error contribution is

$$\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_2,\mathbf{x}_3}}+\int_{\triangle_2}\left(g_i-g_i^k\right)dsdt\Big|_{T_{\mathbf{x}_1,\mathbf{x}_4,\mathbf{x}_5}}=\mathcal{O}\left(h^{k+4}\right). \tag{5.44}$$

For even k, this follows from the identity

$$\mathcal{E}\left(h^{k+3};(\mathbf{x}_3-\mathbf{x}_1),(\mathbf{x}_2-\mathbf{x}_1)\right)+\mathcal{E}\left(h^{k+3};(\mathbf{x}_4-\mathbf{x}_1),(\mathbf{x}_5-\mathbf{x}_1)\right)=0.$$

As in the odd case, combining equations (5.41) and (5.44) yields the second part of inequality (5.29). □

We can now prove Theorem 78.

Proof of Theorem 78. As in Lemma 81, expanding the function f using Taylor's formula around the point $\rho(\hat{p})$ with $\hat{p}\in\triangle_2$ as local Lagrange nodes, yields

$$f(\rho(\hat{p})) - Q_{G_{2,k}} f(\rho(\hat{p})) = \mathcal{R}_{k+1} + \mathcal{O}\left(h^{k+2}\right), \tag{5.45}$$

where $\mathcal{R}_{k+1}$ has the form

$$\mathcal{R}_{k+1} := \sum_{\|\boldsymbol{\alpha}\|_1 = k+1} \left(\frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!} f(\rho(\hat{p})) h^{\boldsymbol{\alpha}} - \sum_{i=1}^{N(2,K)} h^{\boldsymbol{\alpha}} \frac{\partial^{\boldsymbol{\alpha}}}{\boldsymbol{\alpha}!} f(\rho(\hat{p})) \mathcal{L}_i^k(\hat{p}) \right).$$

Integrating both sides of equation (5.45) gives

$$\int_{\triangle_2} \left(f(\rho(\hat{p})) - Q_{G_{2,k}} f(\rho(\hat{p})) \right) g_i ds dt = \int_{\triangle_2} \mathcal{R}_{k+1} g_i ds dt + \mathcal{O}\left(h^{k+4}\right). \tag{5.46}$$

Using $|g_i| = \mathcal{O}\left(h^2\right)$ (see equation (5.7) for $d = 2$), the integral term on the right-hand side is at least of order $\mathcal{O}\left(h^{k+3}\right)$. If k is even, the symmetry of the triangles makes the first term on the right-hand side vanish. Hence the left-hand integral is accurate to order $\mathcal{O}\left(h^{k+4}\right)$. For odd k, we instead have

$$\int_{\triangle_2} \left(f(\rho(\hat{p})) - Q_{G_{2,k}} f(\rho(\hat{p})) \right) g_i ds dt = \mathcal{O}\left(h^{k+3}\right). \tag{5.47}$$

Assume that $\Gamma_{h,k}$ is composed of triangles $\tilde{T}^k$, i.e. $\Gamma_{h,k} = \bigcup_{i=1}^K \tilde{T}_i^k$ and the smooth closed surface, parameterized using (5.1) $\Gamma = \bigcup_{i=1}^K V_i$. Then

$$\int_\Gamma f dS = \sum_{i=1}^K \int_{V_i} f dS, \quad \int_{\Gamma_{h,k}} Q_{G_{2,k}} f dS = \sum_{i=1}^K \int_{\tilde{T}_i^k} Q_{G_{2,k}} f dS. \tag{5.48}$$

We rewrite equation (5.48) over the reference simplex on which the quadrature rules are defined:

$$\int_\Gamma f dS = \sum_{i=1}^K \int_{\triangle_2} f(\rho(\hat{p})) g_i ds dt, \quad \int_{\Gamma_{h,k}} Q_{G_{2,k}} f dS = \sum_{i=1}^K \int_{\triangle_2} Q_{G_{2,k}} f(\rho(\hat{p})) g_i^k ds dt. \tag{5.49}$$

By making use of Lemma 81, we have

$$\begin{aligned}
\left| \int_\Gamma f\, dS - \int_{\Gamma_{h,k}} Q_{G_{2,k}} f\, dS \right| &\leq \sum_{i=1}^K \left| \int_{\triangle_2} \left(f(\rho(\hat{p})) g_i - Q_{G_{2,k}} f(\rho(\hat{p})) g_i^k \right) ds\, dt \right| \\
&\leq \sum_{i=1}^K \int_{\triangle_2} \left| \left(f(\rho(\hat{p})) - Q_{G_{2,k}} f(\rho(\hat{p})) \right) g_i \right| ds\, dt \\
&\quad + \sum_{i=1}^K \int_{\triangle_2} \left| Q_{G_{2,k}} f(\rho(\hat{p})) \left(g_i - g_i^k \right) \right| ds\, dt \\
&\leq \begin{cases} Ch^{k+2}, & k \equiv 0 \pmod 2, \\ Ch^{k+1}, & k \equiv 1 \pmod 2. \end{cases}
\end{aligned}$$

□

For the final inequality, we have applied equation (5.46) and equation (5.47). Furthermore,

using the results established in Theorem 78, we now extend the analysis to vector-valued functions. Specifically, Corollary 82 provides an analogous error estimate for surface integrals of vector fields, using the same polynomial approximation framework outlined in the theorem. This extension broadens the applicability of the results, particularly in scenarios where vector fields must be integrated over curved surfaces, thereby confirming the robustness of the high-order polynomial approximation framework.

Theorem 78 can also be extended to estimate errors in surface integrals of vector-valued functions.

Theorem 82. *Let Γ be a smooth closed embedded hypersurface and $\mathbf{f}:\Gamma\to\mathbb{R}^3$ be a C^{k+2} vector valued function over the smooth hypersurface. Let Γ_h be a triangulation of Γ with mesh size h having vertices lie on Γ and let $\Gamma_{h,k}$ be the $k-$th order approximation of the smooth surface constructed using local fittings of $\pi\in C^{k+3}\left(\Gamma_h,\mathbb{R}^3\right)$. Consider a symmetric triangulation, consisting of $\mathcal{O}(K)=\mathcal{O}\left(h^{-2}\right)$ symmetric triangles, whereas the number of non-symmetric triangles is bounded by $\mathcal{O}\left(\sqrt{K}\right)=\mathcal{O}\left(h^{-1}\right)$. Then*

$$\left|\int_{\Gamma}\mathbf{f}\cdot d\mathbf{S}-\int_{\Gamma_{h,k}}Q_{G_{2,k}}\mathbf{f}\cdot d\mathrm{S}\right|\leq\begin{cases}Ch^{k+2}, & k\equiv 0\ (mod\ 2)\\ Ch^{k+1}, & k\equiv 1\ (mod\ 2),\end{cases}\tag{5.50}$$

where $Q_{G_{2,k}}\mathbf{f}:\Gamma_{h,k}\to\mathbb{R}^3 e$ is a $k-$th order polynomial approximating the integrand $\mathbf{f}$.

As we approach the proof of Theorem 82,

Proof. The proof follows naturally, because the surface integrals of a vector valued function f

$$\int_{\Gamma}\mathbf{f}\cdot d\mathbf{S}=\int_{\Gamma}\mathbf{f}\cdot\mathbf{n}\,dS$$

can be reduced to the scalar integral of $\mathbf{f}\cdot\mathbf{n}$, which can be approximated from Theorem 78. □

Building on Theorems 78 and 82, which analyze the behavior of piecewise polynomial surface parametrizations using Lagrange elements of even and odd orders, the following corollary examines the interplay between differing polynomial orders used for the geometry and for the integrand.

Corollary 83. *Let $\mathbf{f}\in C^{m+2}\left(\Gamma,\mathbb{R}^3\right)$. and the conditions of the Corollary 82 be fulfilled. Then,*

$$\left|\int_{\Gamma}\mathbf{f}\cdot d\mathbf{S}-\int_{\Gamma_{h,k}}Q_{G_{2,m}}\mathbf{f}\cdot d\mathrm{S}\right|\leq Ch^{\min\{\widehat{k},\widehat{m}\}}\tag{5.51}$$

where

$$\widehat{k}=\begin{cases}k+1, & k\equiv 1\ (mod\ 2)\\ k+2, & k\equiv 0\ (mod\ 2)\end{cases},\quad \widehat{m}=\begin{cases}m+1, & m\equiv 1\ (mod\ 2)\\ m+2, & m\equiv 0\ (mod\ 2)\end{cases},$$

and $Q_{G_{2,m}}\mathbf{f}$ is a $m-$th order polynomial approximating the integrand $\mathbf{f}$.

Proof. The proof is analogous to that of Theorem 78 and is therefore omitted. □

5.3.2 Computational results

We design numerical experiments based on the Gauss–Bonnet theorem [107], [122],

$$\int_{\Gamma} K_{\text{Gauss}}\, dS = 2\pi\chi(\Gamma), \tag{5.52}$$

to validate our results. These tests are carried out on a set of classical smooth closed surfaces defined by the following equations:

1) Ellipsoid $\quad \frac{x^2}{a^2}+\frac{y^2}{b^2}+\frac{z^2}{c^2}=1, \quad a,b,c\in\mathbb{R}\setminus\{0\}.$

2) Torus $\quad (x^2+y^2+z^2+R^2-r^2)^2-4R^2(x^2+y^2)=0, \quad 0<r<R\in\mathbb{R}.$

3) Sphere $\quad x^2+y^2+z^2=R^2, \quad R\in\mathbb{R}\setminus\{0\}.$

The analytic expressions for the Gaussian Curvature are

1) Ellipsoid $\quad K_{\text{Gauss}}=\frac{1}{(abc)^2\left(\frac{x^2}{a^4}+\frac{y^2}{b^4}+\frac{z^2}{c^4}\right)^2}, \quad a,b,c\in\mathbb{R}\setminus\{0\}.$

2) Torus $\quad K_{\text{Gauss}}=\frac{\cos v}{r(R+r\cos v)}$, where we used toric coordinates:

$$(x,y,z)=\big((R+r\cos\theta)\cos\varphi,(R+r\cos\theta)\sin\varphi,r\sin\theta\big),\ \varphi,\theta\in[0,2\pi).$$

3) Sphere $\quad K_{\text{Gauss}}=\frac{1}{R^2}, \quad R\in\mathbb{R}\setminus\{0\}.$

In all experiments we utilize the algorithm of Persson and Strang [103] to generate Delaunay triangulations[2] serving as the initial Γ_h surface approximation. In the numerical evaluation of the surface integrals, we use the analytical expression of the Gaussian curvature as the integrand. The integrals are then approximated using a Gaussian quadrature rule of degree 12 on each triangle, employing 32 quadrature points [43].

In Figure 5.5 and Figure 5.6, we show the relative errors under mesh refinement. To calculate the relative error, we integrate the Gaussian curvature on the surface and compare it with the prediction of the Gauss–Bonnet theorem. The convergence rates shown in the plots coincide, confirming our theoretical results.

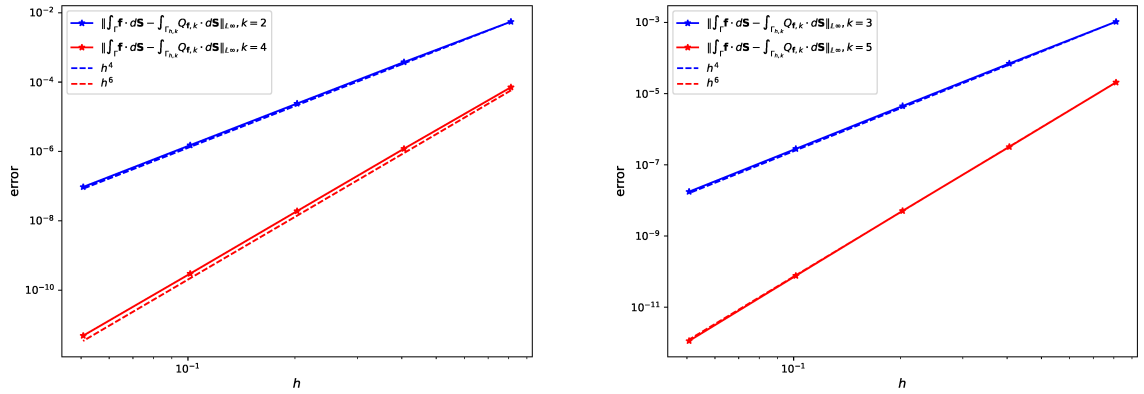
The next experiment considers surface integrals of vector-valued functions. We choose the vector field so that the surface integral yields the volume of the solid enclosed by the surface.

We first recall the divergence theorem, which states:

$$\int_V (\nabla\cdot\mathbf{f})\, dV = \int_{\Gamma} (\mathbf{f}\cdot\mathbf{n})\, dS,$$

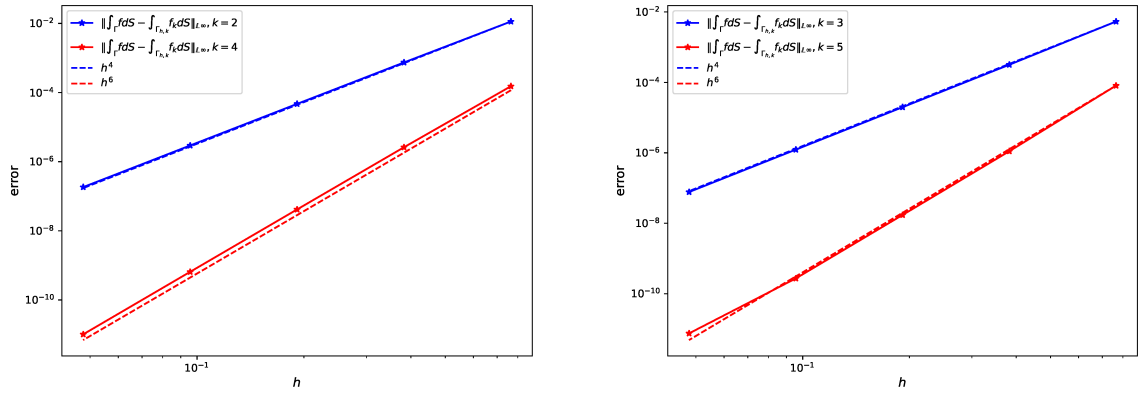
where Γ is the surface boundary of V and $\mathbf{n}$ its outward normal.

[2] [56] presents another viable option for mesh generation. Please note that the Euler characteristic $\chi(\Gamma)$ does not depend on the choice of triangulation, and for any two diffeomorphic surfaces Γ_1 and Γ_2, $\chi(\Gamma_1)=\chi(\Gamma_2)$.



(a) Even order (b) Odd order

Figure 5.5: Relative errors obtained by integrating the Gaussian curvature over the torus with radii $R = 2$, $r = 1$, together with the ideal convergence lines h^n.



(a) Even order (b) Odd order

Figure 5.6: Relative errors by integrating the Gaussian curvature over the unit sphere with the ideal convergence lines h^n.

For simple geometries such as the unit sphere and the torus, the exact volumes are $\frac{4}{3}\pi r^3$ and $2\pi^2 R r^2$, respectively. We use these analytic volumes as reference solutions. In the numerical experiment, we choose

$$\mathbf{f}(x_1, x_2, x_3) = (0, 0, x_3)^T,$$

so that $\nabla \cdot \mathbf{f} = 1$. Hence the enclosed volume is computed from the surface integral

$$|V| = \int_\Gamma \mathbf{f} \cdot \mathbf{n}\, dS = \int_\Gamma x_3 n_3\, dS,$$

where n_3 is the third component of the outward unit normal. Thus, the quantity approximated in the following plots is the scalar surface integral $\int_\Gamma x_3 n_3\, dS$, evaluated using the high-order surface representation.

Given the zero level set of the torus, the gradient $\nabla\phi$ and its unit normal vector are given by:

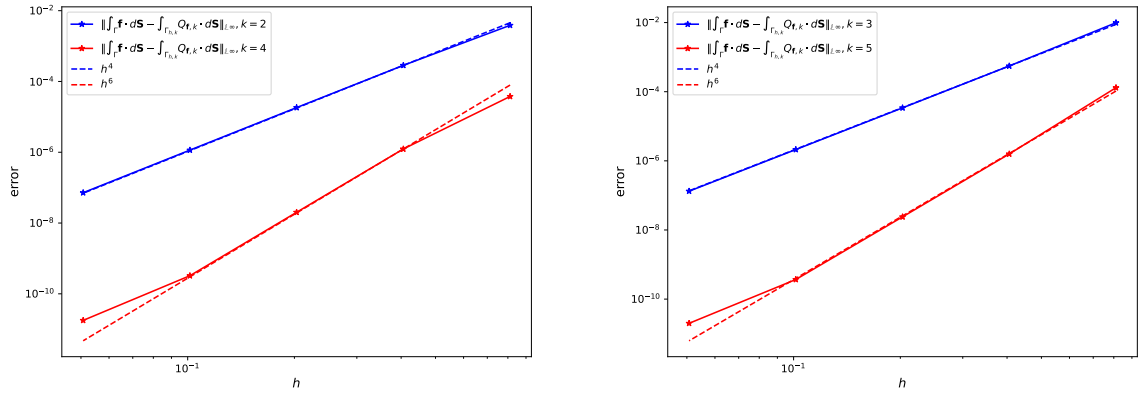
$$\nabla\phi = \left[-2R\frac{x_1}{\sqrt{x_1^2+x_2^2}}+2x_1, -2R\frac{x_2}{\sqrt{x_1^2+x_2^2}}+2x_2, 2x_3\right]$$

$$\|\nabla\phi\| = \sqrt{\left(-2R\frac{x_1}{\sqrt{x_1^2+x_2^2}}+2x_1\right)^2+\left(-2R\frac{x_2}{\sqrt{x_1^2+x_2^2}}+2x_2\right)^2+(2x_3)^2}$$

$$\mathbf{n} = \frac{\nabla\phi}{\|\nabla\phi\|}, \text{ and } n_z = \frac{2x_3}{\|\nabla\phi\|}$$

For the torus, this gives the explicit integrand

$$V = \int_\Gamma \left(x_3 \cdot \frac{2x_3}{\sqrt{\left(-2R\frac{x_1}{\sqrt{x_1^2+x_2^2}}+2x_1\right)^2+\left(-2R\frac{x_2}{\sqrt{x_1^2+x_2^2}}+2x_2\right)^2+(2x_3)^2}} \right) dS$$



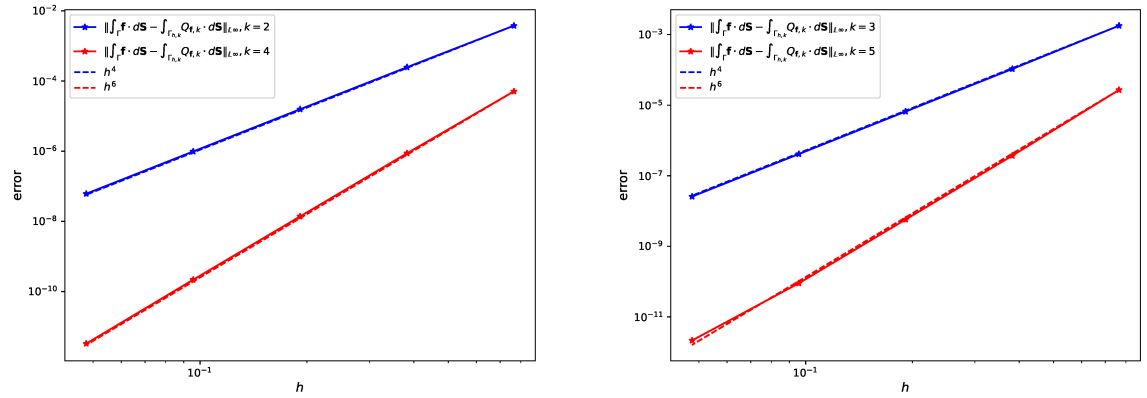
(a) Even order

(b) Odd order

Figure 5.7: Relative errors for the volume of the torus with radii $R = 2$, $r = 1$ with the ideal convergence lines h^n.

Figures 5.7 and 5.8 show the relative errors of the computed volume of the sphere and torus with respect to the degree of the polynomial interpolation of the geometry, where the explicit expression of the integrand is used. The convergence rates shown in the plots coincide with the results predicted by Corollary 82.

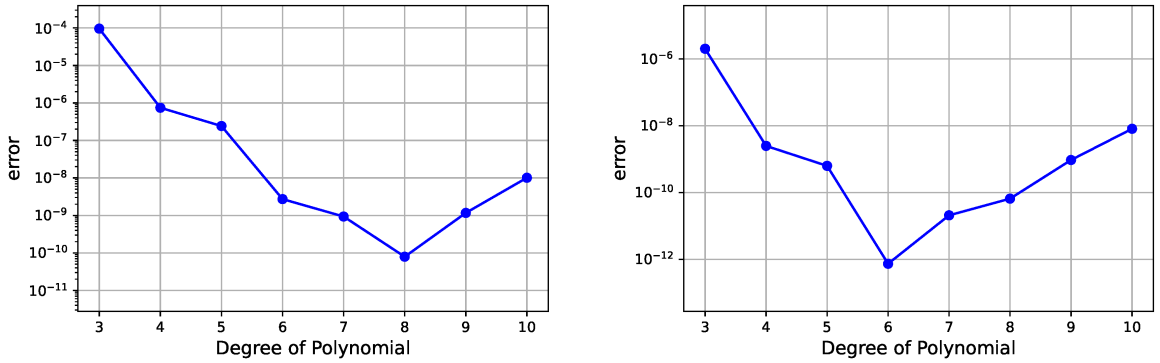
However, as shown in Figure 5.9, numerical integration becomes unstable at higher polynomial



(a) Even order (b) Odd order

Figure 5.8: Relative errors for the volume of the unit sphere with the ideal convergence lines h^n.

degrees when the geometry is interpolated using equidistant nodes, due to the onset of Runge's phenomenon. This behavior can be explained by the fact that the Lebesgue constant Λ associated with equidistant nodes grows exponentially [96].



(a) torus with radii $R = 2, r = 1$ (b) ellipsoid with $a = b = 1, c = 0.6$.

Figure 5.9: Relative errors by integrating the Gaussian curvature over the torus and the ellipsoid using $N_\Delta = 2528$, $N_\Delta = 6152$ respectively.

5.4 Towards optimal interpolation node distributions on simplices

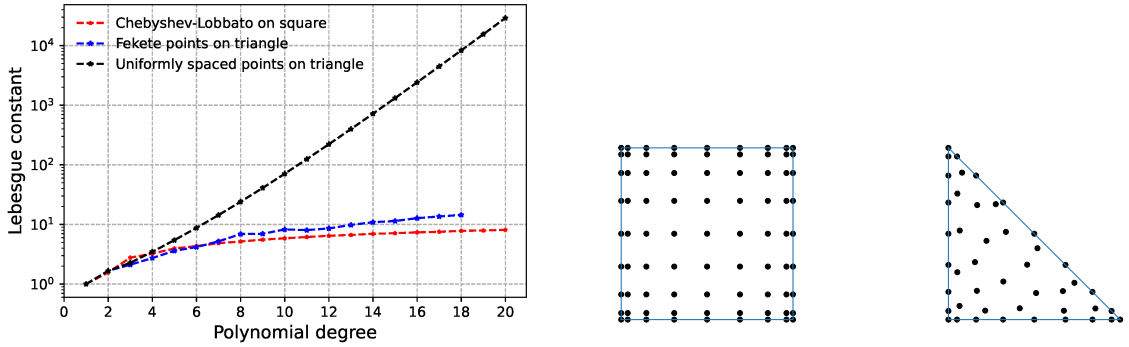
Interpolation at uniformly spaced nodes often exhibits undesirable oscillations, even for analytic functions, as the polynomial degree increases. This effect, known as Runge's phenomenon[3], is illustrated in Figure 5.10. While the problem of node distribution in tensor-product domains is well addressed using Chebyshev or Gauss–Lobatto points, the extension of such strategies to non-tensor-product domains, such as simplices, remains a challenging task.

[3]It arises because the Lebesgue constant Λ for equidistant nodes grows exponentially [96]

The construction of well-distributed interpolation nodes on simplices (e.g., triangles or tetrahedra) is a long-standing problem in stable high-order polynomial interpolation [31, 75, 127, 140]. Classical approaches include Fekete-type points and optimization of Vandermonde determinants, while more recent work has explored explicit and recursive constructions based on one-dimensional node families. A central objective is to identify nodal sets with small Lebesgue constants, since the Lebesgue constant quantifies the stability of the interpolation operator and bounds the interpolation error relative to the best polynomial approximation. Despite this progress, there is no universally accepted analogue of tensor-product Chebyshev–Lobatto grids for general simplices that is simultaneously explicit, stable at high order, inexpensive to generate, and convenient for quadrature and differentiation.

An early attempt to construct optimal points on simplices was made by Bos [17], who proposed determining the nodes by maximizing the determinant of the Vandermonde matrix formed from L_2-orthogonal polynomials. This approach, applied to polynomials of degree up to 7, produced what are known as *Fekete nodes*. In the one-dimensional case, Fekete points coincide with Gauss–Lobatto–Legendre (GLL) points [48]. Extensions of this methodology achieved higher degrees, such as degree 13 in [30] and degree 18 in [21]. Although direct optimization can produce high-quality node sets, it quickly becomes prohibitively expensive for higher degrees and in multiple dimensions. This has led to the exploration of alternative approaches that avoid large-scale optimization. Among these, mapping-based techniques are especially attractive, as an appropriate mapping allows interpolation, differentiation, and quadrature on simplex elements to be performed using their corresponding reference hypercubes. The approach proposed by Karniadakis and Sherwin [81] achieves this by transforming quadrature nodes from a quadrilateral to a triangle via a change of coordinates. While computationally efficient, this method introduces asymmetry in the node distribution, causing points to cluster disproportionately near one vertex of the triangle. In this chapter, we propose a new hypercube–simplex mapping (see Subsection 5.5.1) that yields favorable distributions of interpolation and quadrature points on the triangle, without sacrificing either accuracy or implementation efficiency.

We compare these node-generation strategies using their Lebesgue constants, a quantitative measure of interpolation stability. The constants were computed by evaluating (3.12) on a very fine grid for two cases: maximal degree Chebyshev–Lobatto interpolation on the square $[-1,1]^2$ and total degree interpolation on a uniform grid over the simplex $\triangle_2$. For total degree interpolation in Fekete points on the triangle, we use the values reported in [21]. Figure 5.10 shows the results. The Lebesgue constant for uniform triangle-grid interpolation increases rapidly with polynomial degree, whereas that for Chebyshev–Lobatto interpolation grows much more slowly. Fekete points perform only marginally worse, but are currently known only up to degree 18 [21] for total degree interpolation and are unavailable for the tensorial maximal degree setting.



(a) Lebesgue constant (b) $\text{Cheb}_{2,n}$ (c) Fekete points

Figure 5.10: Lebesgue constants for uniformly spaced points on the triangle, Fekete points, and Chebyshev–Lobatto nodes (5.10a), together with the Chebyshev–Lobatto grid (5.10b) and Fekete points (5.10c) for $n = 8$.

5.5 Stable high-order approximation of triangulated surfaces

To accommodate greater flexibility in mesh generation and avoid restrictive compatibility requirements, we consider a non-conforming triangulation strategy for arbitrary dimensions $d > 2$. In this setting, vertices may be placed inside edges, and adjacent simplices do not need to match exactly. We represent the surface through the decomposition

$$\rho_i : \triangle_d \to V_i \subseteq \Gamma, \quad \bigcup_{i=1}^{K} \overline{V_i} = \Gamma, \tag{5.53}$$

where each ρ_i is a diffeomorphism on the interior $\mathring{\triangle}_d$. Following the notation in Section 5.3, we assume the smooth surface Γ is topologically equivalent to a d-dimensional non-conforming polyhedral surface Γ_h made up of finitely many regular (flat) simplices of diameter h.

$$\Gamma_h = \bigcup_{T \in \mathcal{T}_h} T, \text{ with } K = |\mathcal{T}_h|,$$

such that the maps $\rho_i = \pi_i \circ \tau_i : \triangle_d \to \mathbb{R}^{d+1}$ induce a partition in Γ in the sense of (5.53).

The primary challenge in numerically approximating (5.5) lies in determining the unknown derivatives $\mathbf{D}\rho_i$ that appear in the volume element. As noted in earlier sections, one approach is to replace $\mathbf{D}\rho_i$ with those of a polynomial approximation obtained by interpolation at a prescribed set of nodes in $\triangle_d$. Since the optimal placement of such nodes in simplices for stable, high-order interpolation remains unresolved [31, 127], we instead reparametrize the geometric functions $\rho_i : \triangle_d \to \mathbb{R}^{d+1}$ using a specific hypercube-to-simplex transformation, $\sigma_* : \Omega_d \to \triangle_d$, known as *square-squeezing* (Definition 84). This mapping enables stable, high-order polynomial interpolation without relying on optimal node distributions in simplices. The reparametrized functions ρ_i are then interpolated on each hypercube using k-th order tensorial Chebyshev–Lobatto nodes, a choice well known for mitigating Runge's phenomenon in regular interpola-

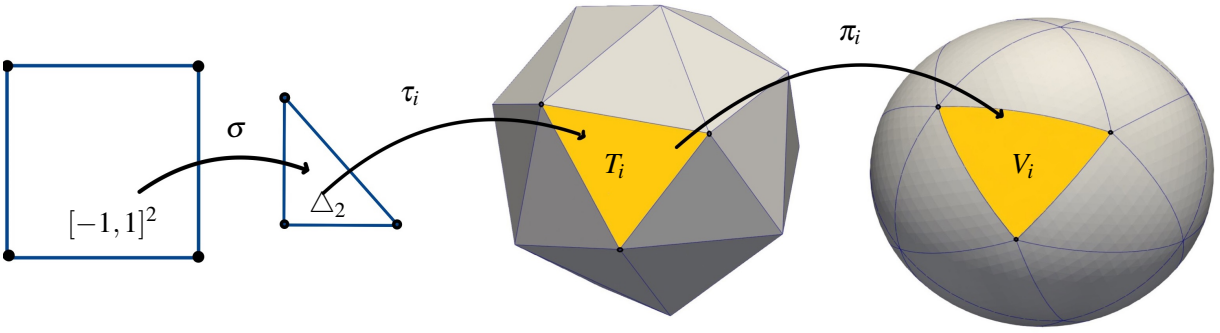


Figure 5.11: Construction of a surface parametrization over $\triangle_2$ by closest-point projection from a piecewise affine approximate mesh, and reparametrization over the square Ω_2.

tion tasks.

Definition 84 (Re-parametrization over cubes)**.** *Let $\sigma : \Omega_d \to \triangle_d$ be a homeomorphism whose restriction $\sigma_{|\mathring{\Omega}_d} : \mathring{\Omega}_d \to \mathring{\triangle}_2$ to the interior is a C^r-diffeomorphism, $r \geq 0$. We call*

$$\varphi_i : \Omega_d \to V_i\,, \quad \varphi_i = \rho_i \circ \sigma = \pi_i \circ \tau_i \circ \sigma\,, \quad i = 1, \ldots, K\,, \tag{5.54}$$

a r-regular cubical reparametrization whenever the coordinate functions of φ_i are of bounded Chebyshev-weighted r^{th}-order directional variation in the sense of Definition 34, for all $i = 1, \ldots, K$.

By reparametrizing in this way, we can set up a hypercube mesh in parallel with the simplex mesh, enabling the construction of arbitrarily stable high-order volume elements (HOVE) for each cube. In this framework, the integral (5.5) is expressed in the form

$$\sum_{i=1}^{K} \int_{\Omega_d} Q_{G_{d,n}}\big(f \circ \varphi_i\big)(\mathbf{x}) \sqrt{\det\Big(\big(\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x})\big)^T \mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x})\Big)}\, d\mathbf{x}, \tag{5.55}$$

where $Q_{G_{d,k}}\varphi_i(\mathbf{x})$ is a kth-order polynomial that approximates the map φ_i, and $Q_{G_{d,n}}\big(f \circ \varphi_i\big)(\mathbf{x})$ is an nth-order polynomial that approximates the integrand $f : \Gamma \to \mathbb{R}$.

5.5.1 The square-squeezing reparametrization map

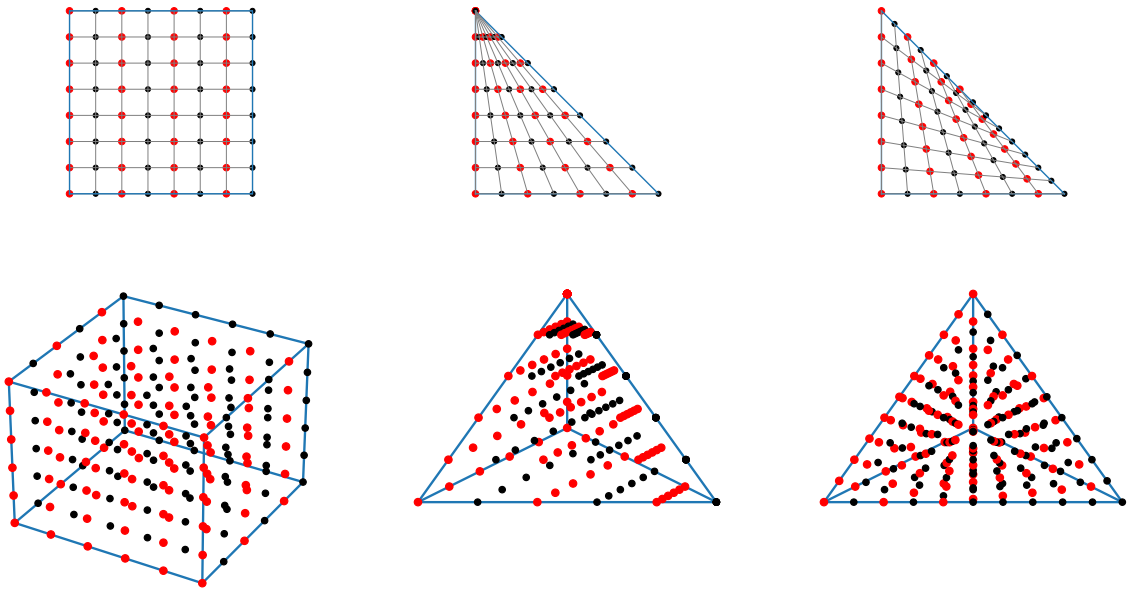
For the hypercube–simplex reparametrization, we propose to use the following multilinear map.

Definition 85 (Square-squeezing)**.** *Let $[0,1]^d$ denote the d-dimensional unit cube with the vertex set $A_{d,2}$. We define the map* square-squeezing *as $\sigma_* : [0,1]^d \to \triangle_d$, which maps the corners $\gamma \in A_{d,2} = \{0,1\}^d$ to*

$$\sigma_*(\gamma) = \begin{cases} (0, \ldots, 0) & \text{if } \gamma = (0, \ldots, 0) \\ \frac{\gamma}{\|\gamma\|_1} & \text{otherwise,} \end{cases}$$

and uses multilinear interpolation for the rest of the domain.

Note that all vertices of the simplex are mapped to themselves. In other words: $\sigma_* : [0,1]^d \to \triangle_d$,



(a) Standard hypercube (b) Duffy's transformation (c) Square-squeezing

Figure 5.12: Multilinear cube–simplex transformations for $d = 2$ and $d = 3$: deformations of equidistant grids (5.12a), under Duffy's transformation (5.12b), and square-squeezing (5.12c).

$d \in \mathbb{N}$ is given by

$$\sigma_*(\mathbf{x}) = \sum_{\gamma \in A_{d,2}} \frac{\gamma}{|\gamma|} \Phi_\gamma, \quad \Phi_\gamma = \prod_{i=1}^{d} x_i^{\gamma_i} (1 - x_i)^{1-\gamma_i}. \tag{5.56}$$

Since, in this section, we operate on the standard cube $\Omega_d \neq [0,1]^d$, we re-scale $\mathbf{x} \mapsto \tilde{\mathbf{x}} = (x_1 + 1, \ldots, x_m + 1)/2$ for defining $\sigma_*(\mathbf{x}) := \sigma_*(\tilde{\mathbf{x}})$ on Ω_d. For illustration, we consider the important two-dimensional case in more detail:

Example 86 (Square-squeezing in two dimensions)**.** *We re-scale* $[-1,1]^2$ *to* $[0,1]^2$ *by setting* $\tilde{x}_1 = (x_1 + 1)/2$, $\tilde{x}_2 = (x_2 + 1)/2$. *The square-squeezing transformation on* $[0,1]^2$ *becomes*

$$\sigma_* : [0,1]^2 \to \triangle_2, \quad \sigma_*(\tilde{x}_1, \tilde{x}_2) = \left(\tilde{x}_1 - \frac{\tilde{x}_1 \tilde{x}_2}{2}, \tilde{x}_2 - \frac{\tilde{x}_1 \tilde{x}_2}{2} \right)^T. \tag{5.57}$$

The inverse map $\sigma_*^{-1} : \triangle_2 \to [-1,1]^2$ *is given by*

$$\sigma_*^{-1}(u,v) = \begin{pmatrix} 1 + (u - v) - \sqrt{(u-v)^2 + 4(1-u-v)} \\ 1 - (u - v) - \sqrt{(u-v)^2 + 4(1-u-v)} \end{pmatrix}. \tag{5.58}$$

Both σ_* *and* σ_*^{-1} *are continuous on* $\Omega_d, \triangle_d$, *respectively, showing square-squeezing to be a homeomorphism. The square-root term in* (5.58) *is smooth for all* (u,v) *in* $\triangle_2 \setminus \{(\frac{1}{2}, \frac{1}{2})\}$. *Hence, the restriction to the interior* $\sigma_{*|[-1,1]^2} : [-\mathring{1}, 1]^2 \to \mathring{\triangle}_2$ *is a diffeomorphism. Further, it is easy to show that* $\|\mathbf{D}\sigma_*\|_{C^0([-1,1]^2)} \leq 1$.

Example 87 (Square-squeezing in three dimensions). *In dimension $d = 3$, we term $\sigma_* : \Omega_3 \to \triangle_3$, $(x,y,z) \mapsto (u,v,w)$ cube-squeezing, visualized in Figure 5.12, explicitly given when re-scaling to $[0,1]^3$ by*

$$\sigma_* : [0,1]^3 \to \triangle_3, \qquad \sigma_*(\tilde{x}_1,\tilde{x}_2,\tilde{x}_3) = \begin{pmatrix} (\tilde{x}_1 - \frac{\tilde{x}_1\tilde{x}_2}{2})(1 - \frac{\tilde{x}_3}{2} + \frac{\tilde{x}_2\tilde{x}_3}{6}) \\ (\tilde{x}_2 - \frac{\tilde{x}_1\tilde{x}_2}{2})(1 - \frac{\tilde{x}_3}{2} + \frac{\tilde{x}_1\tilde{x}_3}{6}) \\ (\tilde{x}_3 - \frac{\tilde{x}_3\tilde{x}_1}{2})(1 - \frac{\tilde{x}_2}{2} + \frac{\tilde{x}_1\tilde{x}_2}{6}) \end{pmatrix}.$$

Remark 88. *Note that the commonly used* Duffy transformation [42]

$$\sigma_{Duffy} : [-1,1]^2 \to \triangle_2, \quad \sigma_{Duffy}(x,y) = \left(\frac{1}{4}(1+x)(1-y), \frac{1+y}{2}\right), \tag{5.59}$$

collapses one entire edge of the square to the single vertex $(0,1)$. Thus, σ_{Duffy} defines a homeomorphism between the interiors, $[-1,1]^2$ and $\mathring{\triangle}_2$, but not between the closed sets, $[-1,1]^2$ and $\triangle_2$. Consequently, σ_{Duffy} can only transform interpolation or quadrature points from $[-1,1]^2$ to $\triangle_2$ and back if none of these points is on the collapsed edge of $[-1,1]^2$ or the point $(0,1)$ of $\triangle_2$, excluding the case of Chebyshev–Lobatto nodes (3.2) *that are commonly considered as the optimal choice for interpolation tasks on hypercubes.*

We now construct stable polynomial approximations of the geometry functions $\rho_i : \triangle_d \to \mathbb{R}^{d+1}$. The resulting approximation can be pulled back to the triangle domain via $\sigma^{-1} : \triangle_d \to \Omega_d$.

Definition 89 (local r-regular cubical reparametrization). *Building on the previous setup, we define the local r-regular cubical reparametrization of the surface Γ. Let $\varphi_i = \rho_i \circ \sigma = \pi_i \circ \tau_i \circ \sigma : \Omega_d \to \tilde{T}$ be the corresponding local parametrization of $\tilde{T}$, for $i = 1,\dots,K$. Recall that the partition $\mathcal{T}_h$ of Γ_h induces the partition $\tilde{\mathcal{T}}_h = \{\tilde{T}\}_{T\in\mathcal{T}_h}$ of Γ.*

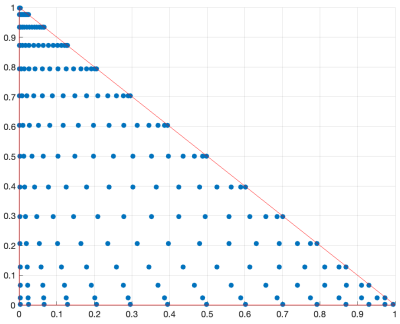
We define $Q_{G_{d,k}}\varphi_i$ as the k^{th}-order vector-valued polynomial interpolation of the mapping φ_i:

$$Q_{G_{d,k}} : C^0(\Omega_d) \longrightarrow \Pi_{d,k}, \quad \varphi \mapsto Q_{G_{d,k}}\varphi, \quad Q_{G_{d,k}}\varphi_i(\mathbf{x}) = \sum_{\substack{\boldsymbol{\alpha}\in G_{d,k} \\ p_{\boldsymbol{\alpha}}\in \mathrm{Cheb}_{d,k}}} \varphi_i(p_{\boldsymbol{\alpha}}) L_{\boldsymbol{\alpha}}. \tag{5.60}$$

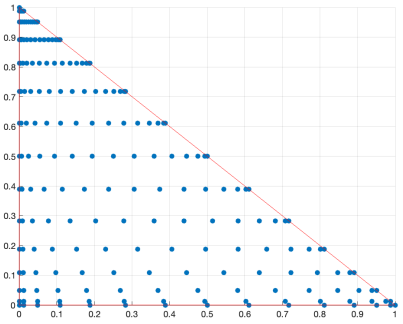
Definition 90 (k^{th}-order cubical reparametrization). *Given an r-regular cubical reparametrization mesh (5.54). We say that the mesh is of order k if each element has been provided as a set of nodes $\{\varphi_i(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,k}}$ sampled at $\{p_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in G_{d,k}}$.*

This implies that on each element, we can approximate the r-regular cubical reparametrization maps through interpolation using the nodes $\{\varphi_i(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,k}}$. For every triangle $T \in \mathcal{T}_h$, we compute $\{\varphi_i(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,k}}$ and define a k-th order d-dimensional triangle $\tilde{T}^k$ by applying Lagrange/Newton interpolation of order k to the coordinates of the projected nodes $\{\varphi_i(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,k}}$.

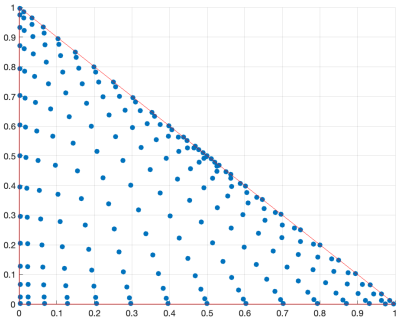
Given $Q_{G_{d,k}}\varphi$ for all $T \in \mathcal{T}_h$, we have successfully obtained a k-th order discrete approximation



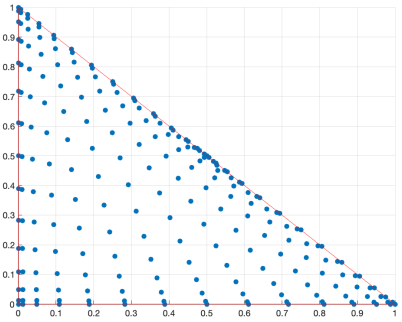
(a) 1st kind Chebyshev nodes



(b) 2nd kind Chebyshev nodes



(c) 1st kind Chebyshev nodes



(d) 2nd kind Chebyshev nodes

Figure 5.13: Comparison of Chebyshev points of the first and second kinds (degree $\deg = 20$) mapped onto a simplex using two different mapping techniques: the Duffy map (Figure 5.13a and 5.13b) and the square-squeezing map (Figure 5.13c and 5.13d).

$\Gamma_{h,k}$ of the continuous surface Γ:

$$\Gamma_{h,k} := \bigcup_{T\in\mathcal{T}_h} Q_{G_{d,k}}\varphi(T) = \bigcup_{\substack{T\in\mathcal{T}_h \\ p_{\boldsymbol{\alpha}}\in \mathrm{Cheb}_{d,k}}} \sum_{\boldsymbol{\alpha}\in G_{d,k}} \varphi_i(p_{\boldsymbol{\alpha}})L_{\boldsymbol{\alpha}}. \tag{5.61}$$

Remark 91. *We highlight that Chebyshev nodes of the second kind provide higher approximation power compared to those of the first kind, as reflected in the corresponding Lebesgue constants, as discussed in Section 3.5. Moreover, any version of Chebyshev nodes (or other interpolation nodes) that exclude the endpoints still leads to clustering near the upper corner when the Duffy transformation is applied. For instance, in Figure 5.13, we compare Chebyshev points of the first and second kinds and observe the clustering effect in the upper corner under the Duffy transformation.*

In addition, polynomials on Chebyshev nodes of the second kind offer a notable advantage: they enable the use of the Fast Fourier Transform (FFT) for differentiation, resulting in an $\mathcal{O}(N\log N)$ implementation (see Section 5.6). Additionally, these grids provide a slight edge in terms of their ability to approximate smooth functions.

5.5.2 HOVE integration errors

We derive the integration error for replacing the surface geometry φ_i and the integrand f by Chebyshev–Lobatto interpolants $Q_{G_{d,k}}\varphi_i$, $Q_{G_{d,n}}(f\circ\varphi_i)$, respectively. As we show in Corollary 95, the resulting closed form expression of the integral can be computed precisely by high-order quadrature rules:

$$\begin{aligned}\int_\Gamma f\,dS &\approx \sum_{i=1}^{K}\int_{\Omega_d} Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x})\big)}\,d\mathbf{x}\\ &\approx \sum_{i=1}^{K}\sum_{\mathrm{p}\in P}\omega_{\mathrm{p}}\,Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{p})\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p})\big)}\,. \end{aligned}\tag{5.62}$$

We start by bounding the approximation error of the geometry.

Lemma 92. *Let Γ be a d-dimensional C^{r+1}-surface, $r\geq 1$, and $\varphi_i=\rho_i\circ\sigma:\Omega_d\to\mathbb{R}^{d+1}$, $i=1,\dots,K$ be a r-regular cubical reparametrization, Definition 84. Let $Q_{G_{d,k}}\varphi_i$ be the vector-valued tensor-polynomial interpolant of φ_i in the Chebyshev–Lobatto grid* $\mathrm{Cheb}_{d,k}$.

i) The Jacobians of φ_i and its interpolant $Q_{G_{d,k}}\varphi_i$ differ by

$$\|\mathbf{D}\varphi_i-\mathbf{D}Q_{G_{d,k}}\varphi_i\|_{C^0(\Omega_d)}\leq\frac{4ed^2V_{\varphi,r}}{\pi(r-d-1)}\left(\frac{k+1}{k+1-r}\right)^{r+1}\cdot\frac{1}{k^{r-d-1}}\,,\tag{5.63}$$

where $V_{\varphi,r}$ is the maximum Chebyshev-weighted r^{th}-order directional variation of the coordinate functions of φ_i.

ii) The difference of the volume elements is bounded by

$$\big\|\sqrt{\det(\Phi_i)}-\sqrt{\det(\Psi_i)}\big\|_{C^0(\Omega_d)}\;\leq\;d\,C^{d-1}\,\|\mathbf{D}\varphi_i-\mathbf{D}Q_{G_{d,k}}\varphi_i\|_{C^0(\Omega_d)}\,,$$

where $\Phi_i=(\mathbf{D}\varphi_i)^T\mathbf{D}\varphi_i$, $\Psi_i=(\mathbf{D}Q_{G_{d,k}}\varphi_i)^T\mathbf{D}Q_{G_{d,k}}\varphi_i$.

Proof. i) follows directly from Corollary 47, (3.27), whereas ii) can be estimated in terms of singular values. For $\Phi_i=\mathbf{D}\varphi_i^T\mathbf{D}\varphi_i$, $\Psi_i=(\mathbf{D}Q_{G_{d,k}}\varphi_i)^T\mathbf{D}Q_{G_{d,k}}\varphi_i$. let $\sigma_1(\cdot)\geq\cdots\geq\sigma_d(\cdot)$ denote the singular values. Then

$$\sqrt{\det(\Phi_i)}=\prod_{j=1}^{d}\sigma_j(\mathbf{D}\varphi_i),\qquad\sqrt{\det(\Psi_i)}=\prod_{j=1}^{d}\sigma_j(\mathbf{D}Q_{G_{d,k}}\varphi_i).\tag{5.64}$$

By the mean-value inequality for products,

$$\big|\sqrt{\det(\Phi_i)}-\sqrt{\det(\Psi_i)}\big|\;\leq\;\sum_{\ell=1}^{d}|\sigma_\ell(\mathbf{D}\varphi_i)-\sigma_\ell(\mathbf{D}Q_{G_{d,k}}\varphi_i)|\prod_{j\neq\ell}\max\{\sigma_j(\mathbf{D}\varphi_i),\sigma_j(\mathbf{D}Q_{G_{d,k}}\varphi_i)\}.\tag{5.65}$$

Now $\sigma_j(\mathbf{D}\varphi_i) \leq \|\mathbf{D}\varphi_i\|$ and, for k large enough,

$$\begin{aligned}\sigma_j(\mathbf{D}Q_{G_{d,k}}\varphi_i) &\leq \|\mathbf{D}Q_{G_{d,k}}\varphi_i\|_{C^0(\Omega_d)} \\ &\leq \|\mathbf{D}\varphi_i\|_{C^0(\Omega_d)} + \|\mathbf{D}Q_{G_{d,k}}\varphi_i - \mathbf{D}\varphi_i\|_{C^0(\Omega_d)} \\ &\leq \|\mathbf{D}\varphi_i\|_{C^0(\Omega_d)} + 1.\end{aligned}$$

We set $C := \max_{i=1,\dots,K} \|\mathbf{D}\varphi_i\|_{C^0(\Omega_d)} + 1$ and obtain

$$\begin{aligned}\left|\sqrt{\det(\Phi_i)} - \sqrt{\det(\Psi_i)}\right| &\leq \sum_{k=1}^{d} |\sigma_k(\mathbf{D}\varphi_i) - \sigma_k(\mathbf{D}Q_{G_{d,k}}\varphi_i)|\, C^{d-1} \\ &\leq d\, C^{d-1} \|\mathbf{D}\varphi_i - \mathbf{D}Q_{G_{d,k}}\varphi_i\|_{C^0(\Omega_d)},\end{aligned}$$

where we used Weyl's inequality for the second line. □

With the help of the previous result, we bound the integration error.

Theorem 93 (Integration error)**.** *Let the assumptions of Lemma 92 be satisfied, and let $k, n \in \mathbb{N}$ denote the interpolation degrees used for the geometry and the integrand, respectively. Thus $Q_{G_{d,k}}\varphi_i$ denotes the tensor-polynomial interpolant of the geometry map φ_i on the grid $\mathrm{Cheb}_{d,k}$, while $Q_{G_{d,n}}(f\circ\varphi_i)$ denotes the tensor-polynomial interpolant of the local scalar pullback $f\circ\varphi_i$ on the grid $\mathrm{Cheb}_{d,n}$. Let $f : \Gamma \to \mathbb{R}$ be such that its local pullbacks $f\circ\varphi_i$ have bounded Chebyshev-weighted r^{th}-order directional variation, and set $V_{f,r} := \max_i V_{f\circ\varphi_i,r}$. Then the integration error induced by approximating the geometry φ_i and the integrand $f\circ\varphi_i$ is*

$$\begin{aligned}&\left|\int_\Gamma f\,dS - \sum_{i=1}^{K}\int_{\Omega_d} Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{x})\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{x}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{x})\big)}\,d\mathrm{x}\right| \\ &\qquad\leq \varepsilon_f \mathrm{vol}(\Gamma) + \varepsilon_f\varepsilon_\varphi \mathrm{vol}(\Omega_d) + \|f\|_{C^0(\Gamma)}\varepsilon_\varphi\,\mathrm{vol}(\Omega_d) = \mathcal{O}\Big(\frac{1}{n^{r-d+1}}\Big) + \mathcal{O}\Big(\frac{1}{k^{r-d-1}}\Big),\end{aligned}$$

where $\mathrm{vol}(\Gamma)$ *and* $\mathrm{vol}(\Omega_d)$ *denote the volumes of* Γ *and* Ω_d*, respectively, and*

$$\varepsilon_f = \frac{4ed^2V_{f,r}}{\pi(r-d+1)}\left(\frac{n+1}{n+1-r}\right)^{r+1}\cdot\frac{1}{n^{r-d+1}}, \quad \mu_\varphi = \frac{4ed^2V_{\varphi,r}}{\pi(r-d-1)}\left(\frac{k+1}{k+1-r}\right)^{r+1}\cdot\frac{1}{k^{r-d-1}},$$

with

$$\varepsilon_\varphi \;=\; d\,C^{d-1}\,\mu_\varphi, \qquad C := \max_{i=1,\dots,K}\|\mathbf{D}\varphi_i\|_{C^0(\Omega_d)} + 1 \qquad V_{\varphi,r} = \max_{i=1,\dots,K} V_{\varphi_i,r}\,.$$

Proof. We set $\Phi_i = (\mathbf{D}\varphi_i)^T\mathbf{D}\varphi_i$ and $\Psi_i = (\mathbf{D}Q_{G_{d,k}}\varphi_i)^T\mathbf{D}Q_{G_{d,k}}\varphi_i$. Adding and subtracting the inter-

polated integrand and the interpolated volume element gives

$$\begin{aligned}
&\Big|\int_\Gamma f\,dS - \sum_{i=1}^{K}\int_{\Omega_d} Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\sqrt{\det(\Phi_i(\mathbf{x}))}\,d\mathbf{x}\Big| \\
&\qquad\le \sum_{i=1}^{K}\int_{\Omega_d}\big|f(\varphi_i(\mathbf{x})) - Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\big|\sqrt{\det(\Psi_i(\mathbf{x}))}\,d\mathbf{x} \\
&\qquad+ \sum_{i=1}^{K}\int_{\Omega_d}\big|f(\varphi_i(\mathbf{x})) - Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\big|\cdot\Big\|\sqrt{\det(\Phi_i(\mathbf{x}))} - \sqrt{\det(\Psi_i(\mathbf{x}))}\Big\|_{C^0(\Omega_d)}\,d\mathbf{x} \\
&\qquad+ \sum_{i=1}^{K}\int_{\Omega_d}\big|f(\varphi_i(\mathbf{x}))\big|\cdot\Big\|\sqrt{\det(\Phi_i(\mathbf{x}))} - \sqrt{\det(\Psi_i(\mathbf{x}))}\Big\|_{C^0(\Omega_d)}\,d\mathbf{x} \\
&\qquad\le \varepsilon_f\mathrm{vol}(\Gamma) + \varepsilon_f\varepsilon_\varphi\mathrm{vol}(\Omega_d) + \|f\|_{C^0(\Gamma)}\varepsilon_\varphi\mathrm{vol}(\Omega_d)\,.
\end{aligned}$$

The first term is controlled by the interpolation error of $f\circ\varphi_i$, while the second and third terms are controlled by the volume-element estimate of Lemma 92. The estimates for ε_f, ε_φ, μ_φ follow from Corollary 47 and Lemma 92, concluding the proof. □

To the best of our knowledge, this estimate is the first to guarantee convergence to the correct integral solely by increasing the polynomial order. In contrast, as demonstrated in the previous section, numerical methods for surface integration that rely on equidistant points only show:

$$E(f,\Gamma) \le Ch^{\min\{\widehat{k},\widehat{n}\}}, \quad C = C(n,k,f,\Gamma) > 0\,, \tag{5.66}$$

where

$$\widehat{k} = \begin{cases} k+1, & k\equiv 1 \pmod 2 \\ k+2, & k\equiv 0 \pmod 2 \end{cases}, \quad \widehat{n} = \begin{cases} n+1, & n\equiv 1 \pmod 2 \\ n+2, & n\equiv 0 \pmod 2 \end{cases},$$

and $h>0$ is the mesh size. Here, the constant C explicitly depends on the degrees n and k. Notably, the influence of f on the interpolation error cannot be controlled, as f is problem-dependent. Consequently, $C(n,k,f,\Gamma)\to\infty$ with $n,k\to\infty$, in contrast to Theorem 93, no guarantees of higher accuracy or even convergence is given for p-refinements, increasing n, k. Additionally, the approximation rate is only algebraic in the mesh size h.

If, furthermore, both the integrand and the surface are analytic, then by applying equation (3.41) and equation (3.42), it follows that the integration error decreases exponentially:

$$E(f,\Gamma) \le MR^{-\min\{n,k\}}, \qquad \text{for some} \quad R>1, \quad M = M(f,\Gamma) > 0. \tag{5.67}$$

Corollary 94 (Integration error for vector-valued fluxes)**.** *Let $\mathbf{f}:\Gamma\to\mathbb{R}^3$ be a vector-valued function and set*

$$F(\mathbf{x}) := \mathbf{f}(\mathbf{x})\cdot\mathbf{n}(\mathbf{x})\,.$$

Using the same interpolation degrees n and k as in Theorem 93, if F has bounded Chebyshev-weighted

r^{th}-order directional variation $V_{F,r}$, then the scalar estimate of Theorem 93 applied to F yields

$$\left|\int_\Gamma \mathbf{f}\cdot d\mathbf{S} - A_{n,k}(F)\right| \le \varepsilon_F \operatorname{vol}(\Gamma) + \varepsilon_F\varepsilon_\varphi \operatorname{vol}(\Omega_d) + \|F\|_{C^0(\Gamma)}\varepsilon_\varphi \operatorname{vol}(\Omega_d)$$
$$= \mathcal{O}\Big(\frac{1}{n^{r-d+1}}\Big) + \mathcal{O}\Big(\frac{1}{k^{r-d-1}}\Big).$$

where $A_{n,k}(F)$ denotes the HOVE approximation in Theorem 93 with the scalar integrand F, and

$$\varepsilon_F = \frac{4ed^2V_{F,r}}{\pi(r-d+1)}\left(\frac{n+1}{n+1-r}\right)^{r+1}\cdot\frac{1}{n^{r-d+1}}.$$

Proof. The flux integral satisfies

$$\int_\Gamma \mathbf{f}\cdot d\mathbf{S} = \int_\Gamma \mathbf{f}\cdot\mathbf{n}\,dS = \int_\Gamma F\,dS.$$

The claim follows by applying Theorem 93 to the scalar function F. □

The approximated integral (5.62) can now be evaluated using an appropriate quadrature rule. Two principal approaches are available: one may apply a quadrature rule directly on the cubical domain Ω_d, or alternatively employ a simplex-based rule and map it back to Ω_d via the inverse of the square-squeezing transformation σ_* (which amounts to integrating over the original triangulation $\{\rho_i\}$ of Γ). In the latter case, we obtain the following result:

Corollary 95 (Quadrature rule error)**.** *Under the assumptions of Theorem 93, let $\Phi_i = (\mathbf{D}\varphi_i)^T\mathbf{D}\varphi_i$ and $\Psi_i = (\mathbf{D}Q_{G_{d,k}}\varphi_i)^T\mathbf{D}Q_{G_{d,k}}\varphi_i$. The degrees n and k have the same meaning as in Theorem 93: $Q_{G_{d,n}}(f\circ\varphi_i)\in\Pi_{d,n}$ approximates the scalar pullback of the integrand, while $Q_{G_{d,k}}\varphi_i$ approximates the geometry. For the truncation index $l\in\mathbb{N}$ used in the polynomial approximation of the square-root factor, there is $0<\nu<1$, independent of l, such that for $1\le l$ large enough:*

i) Let $\mathrm{p}\in P,\omega_\mathrm{p}$ be the points and weights of the tensorial Gauss–Legendre quadrature on Ω_d [125] of order $N\in\mathbb{N}$, integrating any polynomial $Q\in\Pi_{d,M}$ of l_∞-degree $M=2kdl+n$ exactly. Then

$$\int_{\Omega_d} Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x})\big)}\,d\mathbf{x}$$
$$= \sum_{\mathrm{p}\in P}\omega_\mathrm{p}Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{p})\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p})\big)} + \mathcal{O}(\nu^{l+1}). \quad (5.68)$$

ii) Let $\sigma:\Omega_d\to\triangle_d$ be a cube–simplex transformation diffeomorphic in the interior $\mathring{\Omega}_d$, $P^*=\{\mathrm{p}^*=\sigma^{-1}(\mathrm{q}):\mathrm{q}\in P\subseteq\mathring{\triangle}_d\}$, $\omega_{\mathrm{p}^*}=\omega_\mathrm{q}\sqrt{\det((\mathbf{D}\sigma^{-1}(\mathrm{q}))^T\mathbf{D}\sigma^{-1}(\mathrm{q}))}$ be the σ-pull-back rule of a simplex rule of order $N^*\in\mathbb{N}$, integrating any polynomial $Q\in\Pi_{d,M}$ of l_∞-degree $M=2kdl+n$ exactly on

$\triangle_d$. *Then*

$$\int_{\triangle_d} Q_{G_{d,n}}(f\circ\varphi_i)(\sigma^{-1}(\mathrm{y}))\sqrt{\det\big((\mathbf{D}\rho_i(\mathrm{y}))^T\mathbf{D}\rho_i(\mathrm{y})\big)}d\mathrm{y}$$
$$=\sum_{\mathrm{p}^*\in P^*}\omega_{\mathrm{p}^*}Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{p}^*)\sqrt{\det\big((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p}^*))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathrm{p}^*)\big)}+\mathcal{O}(\varepsilon_i I_{i,\Omega_d})+\mathcal{O}(v^{l+1}), \tag{5.69}$$

with $\varepsilon_i=\|\sqrt{\det(\Phi_i)}-\sqrt{\det(\Psi_i)}\|_{C^0(\Omega_d)}$ *as in Lemma 92 and* $I_{i,\Omega_d}=\int_{\Omega_d}Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})d\mathbf{x}$.

Proof. To prove i), we choose $\kappa>\|\sqrt{\det(\Psi_i)}\|_{C^0(\Omega_d)}$ and rewrite:

$$\kappa\sqrt{\det\big(\frac{1}{\kappa^2}\Psi_i(\mathbf{x})\big)}=\kappa\sqrt{1+x},\ x=\frac{1}{\kappa^2}\det(\Psi_i(\mathbf{x}))-1\,.$$

We recall that $\sqrt{1+x}=\sum_{s=0}^{\infty}\frac{(-1)^s 2s!}{(1-2s)(s!)^2(4^s)}x^s$, for $|x|<1$, and deduce that

$$\int_{\Omega_d}Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\kappa\sqrt{\det(\frac{1}{\kappa^2}\Psi_i(\mathbf{x}))}\,d\mathbf{x}=\int_{\Omega_d}Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})Q(\mathbf{x})d\mathbf{x}+\mathcal{O}(v^{l+1}),$$

where Q has l_∞-degree $M-n$. Hence, $\int_{\Omega_d}Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})Q(\mathbf{x})d\mathbf{x}=\sum_{\mathrm{p}\in P}\omega_{\mathrm{p}}Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{p})Q_l(\mathrm{p})$ can be computed exactly due to the Gauss–Legendre quadrature of order N. Consequently, i) is proven. Now ii) follows from i) by

$$\begin{aligned}
&\int_{\triangle_d}Q_{G_{d,n}}(f\circ\varphi_i)(\sigma^{-1}(\mathrm{y}))\sqrt{\det\big((\mathbf{D}\rho_i(\mathrm{y}))^T\mathbf{D}\rho_i(\mathrm{y})\big)}\,d\mathrm{y}\\
&\quad=\int_{\triangle_d}Q_{G_{d,n}}(f\circ\varphi_i)(\sigma^{-1}(\mathrm{y}))\sqrt{\det\big((\mathbf{D}\sigma^{-1}(\mathrm{y}))^T(\mathbf{D}\varphi_i(\sigma^{-1}(\mathrm{y})))^T\mathbf{D}\varphi_i(\sigma^{-1}(\mathrm{y}))\mathbf{D}\sigma^{-1}(\mathrm{y})\big)}\,d\mathrm{y}\\
&\quad=\int_{\triangle_d}Q_{G_{d,n}}(f\circ\varphi_i)(\sigma^{-1}(\mathrm{y}))\sqrt{\det(\Psi_i(\sigma^{-1}(\mathrm{y})))\det(\mathbf{D}\sigma^{-1}(\mathrm{y})^T\mathbf{D}\sigma^{-1}(\mathrm{y}))}\,d\mathrm{y}\\
&\quad+\int_{\Omega_d}Q_{G_{d,n}}(f\circ\varphi_i)(\mathbf{x})\Big(\sqrt{\det(\Phi_i(\mathbf{x}))}-\sqrt{\det(\Psi_i(\mathbf{x}))}\Big)d\mathbf{x}\\
&\quad=\sum_{\mathrm{p}^*\in P^*}\omega_{\mathrm{p}^*}Q_{G_{d,n}}(f\circ\varphi_i)(\mathrm{p}^*)\sqrt{\det(\Psi_i(\mathrm{p}^*))}+\mathcal{O}(\varepsilon_i I_{i,\Omega_d})+\mathcal{O}(v^{l+1}),
\end{aligned}$$

proving the statement. □

Remark 96. *In fact, Corollary 95 ii) applies for the square-squeezing transformation* σ_* *and Duffy transformation* σ_{Duffy} *in combination with the symmetric Gauss quadrature* $\mathrm{q}\in P,\omega_{\mathrm{q}}$ *of the triangle* $\triangle_2$ [43] *(both are diffeomorphisms in the interior* $\mathring{\Omega}_2$ *and* $P\subseteq\mathring{\triangle}_2$*).*

While Corollary 95 suggests the necessity of a high order quadrature, $M\gg k,n$, as part of the next section, we empirically find that choosing $M=k=n$ equally to the interpolation degrees suffices for achieving computations reaching machine precision.

The derived error bounds indicate that the integration error can decay algebraically or even

exponentially with increasing interpolation degree. To validate these theoretical predictions, we present several numerical experiments that demonstrate the performance of the proposed method across different integration tasks. We focus on the important case of two-dimensional surfaces exclusively.

5.5.3 Computational results

We triangulate these surfaces by first approximating them by piecewise affine triangulations in $\mathbb{R}^3$, constructed by the algorithm of Persson and Strang [103]. The flat triangles are then equipped with the Euclidean closest-point projections, approximating the maps $\pi_i : T_i \to \Gamma$ as described in Section 4.2.2.

We compare *HOVE* with the Dune-CurvedGrid (DCG) method, a Dune[4] module for surface parametrization [106]. As discussed in [147], DCG interpolates the closest-point projection directly on each triangle, using l_1-degree polynomials on a uniform point set.

Unless stated otherwise, *HOVE* uses square-squeezing pull-backs of symmetric Gauss triangle rules [43] as quadratures on $[-1,1]^2$ (Corollary 95 ii)). Similarly, for DCG, we also make use of symmetric Gauss triangle rules of the same degree as used in *HOVE*. Our implementation of *HOVE* is part of a Python package called surfgeopy.[5] The examples and results of this subsection using Dune-CurvedGrid are summarized and made available in a separate repository.[6]

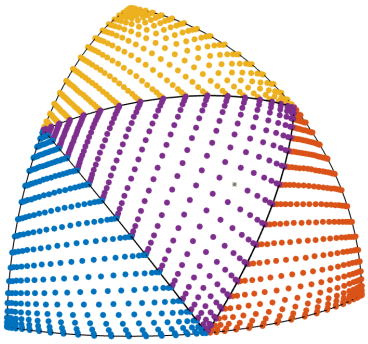
Unless stated otherwise, the relative errors in this subsection are measured against analytical reference values for the prescribed surface integrals. The polynomial degree denotes the degree used for the geometry representation, and the quadrature degree is chosen to match the approximation degree except in experiments where a fixed rule is explicitly specified. Runtime comparisons use the implementations and repositories cited above.

Duffy-transform integration vs square-squeezing integration

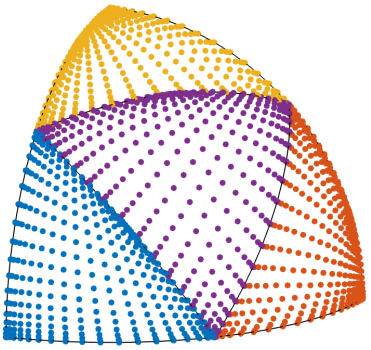
This first experiment investigates the impact of interpolating the volume element of the sphere with Chebyshev nodes of the 1^{st} or 2^{nd} kind in conjunction with the Duffy and square-squeezing transformations, respectively, and subsequently computing the area of one octant of the unit sphere. Here, Fejér's rule is applied for 1^{st} kind Chebyshev nodes, while Clenshaw–Curtis quadrature is employed for 2^{nd} kind Chebyshev nodes, each of order equal to the interpolation degree.

Figure 5.14c shows the resulting relative errors. In both cases, we observe exponential error decay. However, square-squeezing integration achieves two orders of magnitude higher accuracy. Specifically, for $\deg = 20$ Duffy-transform-integration results in an error of 2.4736×10^{-13}, while

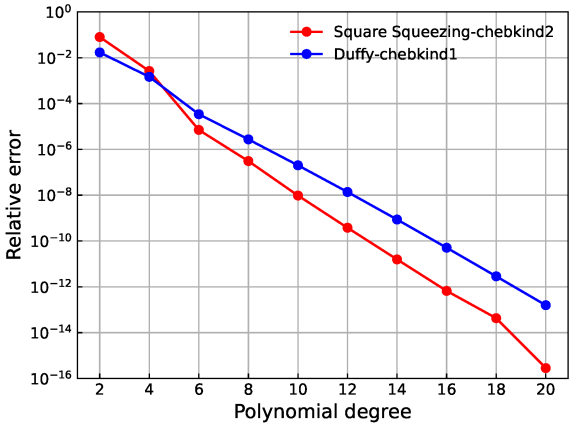
[4] www.dune-project.org
[5] https://github.com/casus/surfgeopy
[6] https://github.com/casus/dune-surface_int



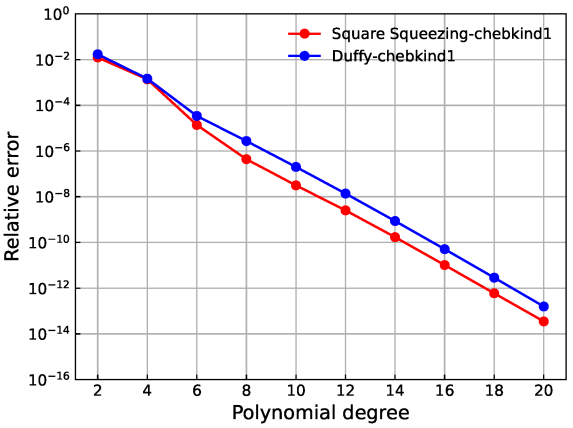
(a) Chebyshev nodes of the 1st kind mapped by the Duffy transformation.



(b) Chebyshev nodes of the 2nd kind mapped by square-squeezing.



(c) Square-squeezing integration with 2nd kind Chebyshev nodes vs Duffy integration with 1st kind Chebyshev nodes.



(d) Square-squeezing integration vs Duffy-transform integration using 1st kind Chebyshev nodes in both cases.

Figure 5.14: Chebyshev nodes mapped onto a triangulation of one octant of the unit sphere (5.14a), (5.14b), together with the relative errors of square-squeezing integration (5.14c) and Duffy-transform integration (5.14d).

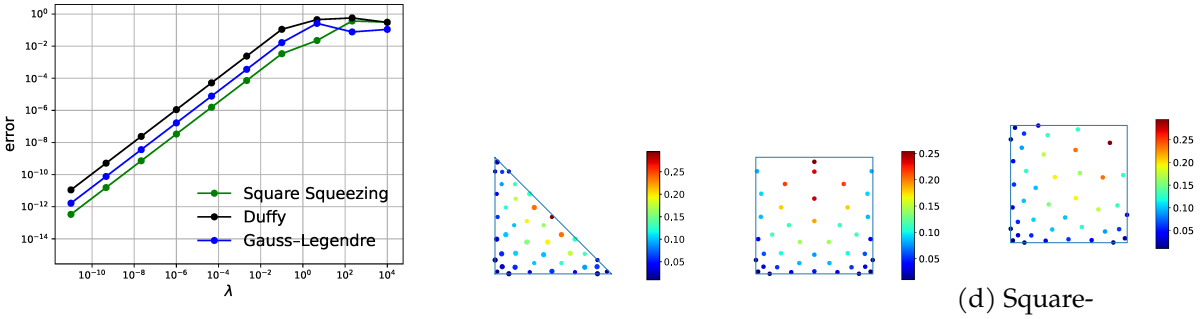
square-squeezing-integration achieves 4.4409×10^{-16}. Additionally, comparing both transformations while using 1st kind Chebyshev nodes, Figure 5.14d still shows an advantage in exploiting square-squeezing instead of the Duffy transformation.

This significant enhancement in accuracy for a simple integration task suggests the impact of the HOVE approach, which is investigated further below.

Integration on a single triangle

In the first experiment, we investigate the impact that the square–triangle transformation has on the integration performance. For this we compute the integral of a function $f : [-1, 1]^2 \to \mathbb{R}$ on the standard square using three different quadrature rules (all of degree 14):

1. The tensorial Gauss–Legendre rule (196 points),



(a) Integration errors (b) Triangle rule (c) Duffy (d) Square-squeezing

Figure 5.15: (5.15a): The integration errors of $f(x_1, x_2) = \sin(\lambda x_1)$ as a function of the parameter λ. (5.15b): Quadrature points and weights (indicated by the color bar) of the symmetric Gauss quadrature rule [43] of degree $n = 14$ on $\triangle_2$. Same for the resulting pull-back rules of the inverse Duffy transformation (5.15c), and inverse square-squeezing (5.15d). The color bar indicates the quadrature weights.

2. The symmetric Gauss rule for the triangle [43], pulled back to $[-1,1]^2$ via the square-squeezing map σ_* (42 points), as in Corollary 95 ii).
3. The same rule, but pulled back by Duffy's transformation σ_{Duffy} (also 42 points), again as in Corollary 95 ii).

As the integrand, we use the function

$$f(x_1, x_2) = \sin(\lambda x_1), \quad \lambda \in \mathbb{R},$$

that, by symmetry, vanishes when being integrated over $[-1,1]^2$ for all parameter choices $\lambda \in \mathbb{R}$.

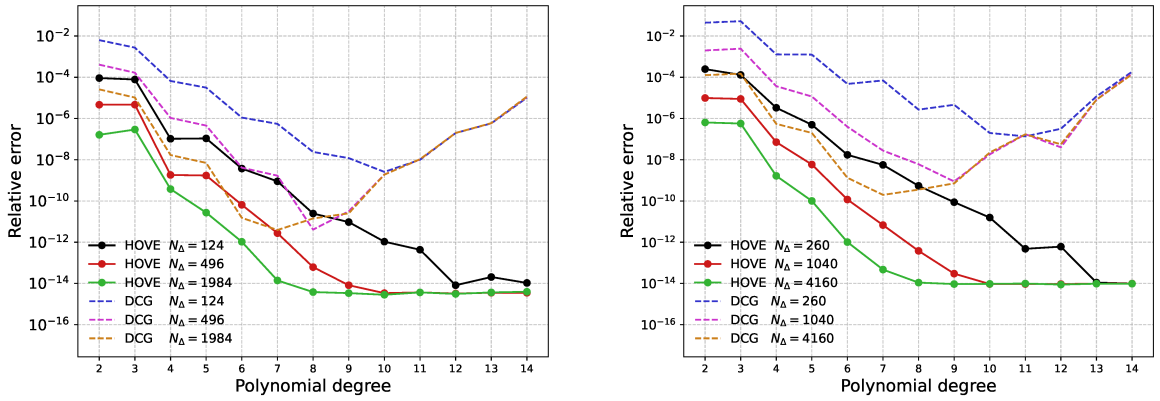
Figure 5.15 shows the absolute integration errors as a function of λ in the range $[10^{-11}, 10^4]$, appearing as linearly depending on λ for values below 1. For larger values the quadrature rule cannot resolve the oscillatory integrand, and the error is essentially random.

Integration with respect to square-squeezing shows almost two orders of magnitude better accuracy than integration with respect to Duffy's transformation for small values of λ. Additionally, it shows a one-order-of-magnitude enhancement in accuracy compared to integration using tensorial Gauss–Legendre quadrature.

Figure 5.15 also shows the points and weights of the two pulled-back simplex rules. We observe that the weights are in the same range for both square-squeezing and the Duffy transformation. The results, however, suggest that the σ_*–pullback rule is the superior choice.

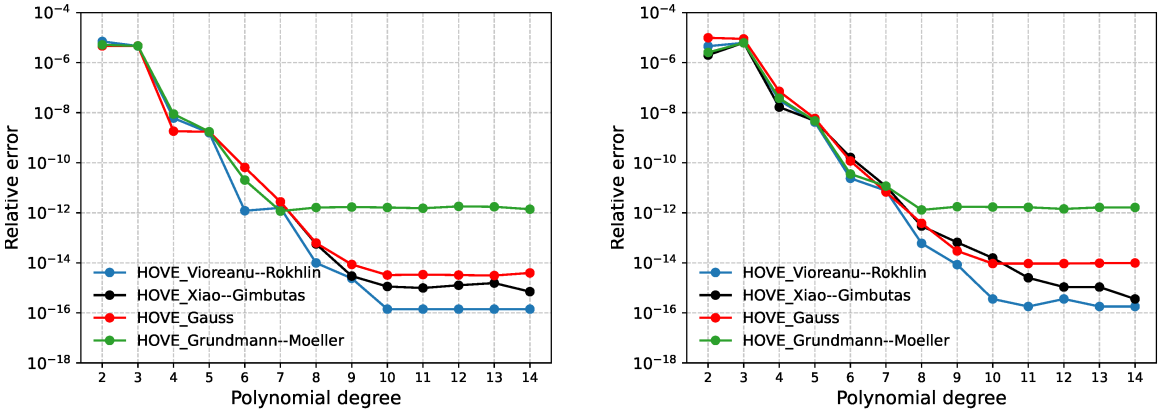
Surface area

The next experiment is the first to involve an actual integration over a surface S. We integrate the constant function $f = 1$ over the unit sphere S^2 and the torus $T^2_{r,R}$ with inner radius $r = 1$ and



(a) Unit sphere (b) Torus with radii $r = 1$ and $R = 2$

Figure 5.16: Relative errors of DCG and *HOVE* for the surface area of the unit sphere and the torus, using three different meshes



(a) Unit sphere (b) Torus with radii $r = 1$ and $R = 2$

Figure 5.17: Relative errors of HOVE, using Vioreanu–Rokhlin, Xiao–Gimbutas, symmetric Gauss, and Grundmann–Moeller simplex rules, integrating the surface areas of the unit sphere and the torus.

outer radius $R = 2$. The expected result is the surface area, which is 4π for the unit sphere and $4\pi^2 rR$ for the torus. We choose initial triangulations of size $N_\Delta = 124$ for the sphere and of size $N_\Delta = 260$ for the torus and apply the symmetric Gauss quadrature rule for the triangle $\triangle_2$ of $\deg = 14$ with 42 quadrature points [43].

Note that as the integrand f is constant, its approximation $Q_{G_{d,n}}$ is f itself, and there is no approximation error.

Figure 5.16 shows the relative errors with respect to the degree of the polynomial interpolation of the geometry. *HOVE* stably converges to machine precision with a high algebraic rate, as predicted by Theorem 93. In contrast, DCG becomes unstable for orders larger than $\deg = 8$. We interpret the instability as the appearance of Runge's phenomenon caused by the choice of

Table 5.1: Mesh data

mesh	# vertices	# vertices for IDS [109]
0	272	544
1	1088	1896
2	4352	7528
3	17 408	31 392

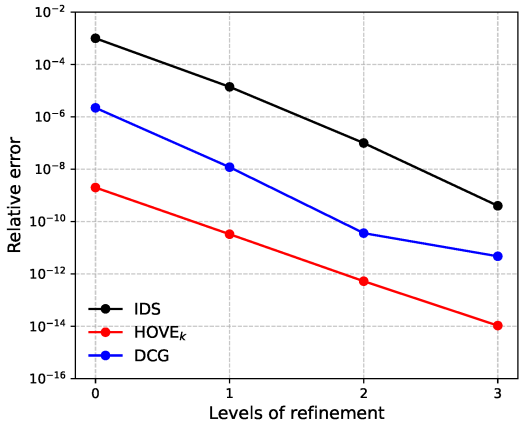
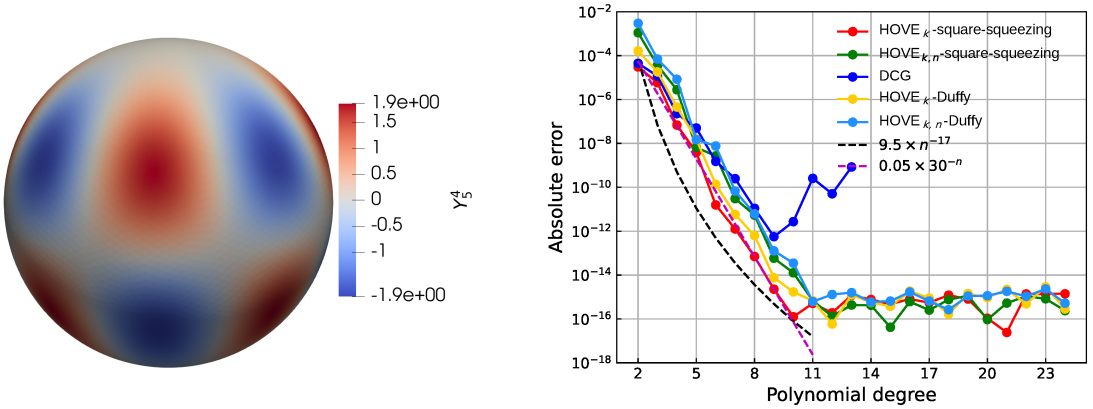


Figure 5.18: Relative errors of IDS [109], DCG, and HOVE$_k$ for the surface area of the torus, using a polynomial of degree 6, are presented on four different meshes, as detailed in the table on the left.

equidistant interpolation nodes for DCG. Compare this with the plot of the Lebesgue constants in Figure 5.10. There, the difference between the constants for Chebyshev–Lobatto interpolation on the square and uniform interpolation on the triangle becomes noticeable around the order $k = 6$.

Furthermore, for each initial mesh, we use HOVE with square-squeezing pull-backs applied to state-of-the-art simplex quadrature rules. These rules include the symmetric Gauss rule [43], Grundmann–Möller quadrature [62], Xiao–Gimbutas quadrature [144], and Vioreanu–Rokhlin simplex quadrature [139], which are employed as quadratures on $[-1,1]^2$ (Corollary 95, item ii)). Figure 5.17 shows the relative errors, demonstrating the superior accuracy of HOVE with Vioreanu–Rokhlin quadrature, but only in the range of machine precision $(10^{-14} \sim 10^{-15})$ for higher polynomial degrees. In contrast, the Grundmann–Möller quadrature performs poorly, which might be attributed to the presence of both negative and positive weights.

Remark 97. *In the case of the torus Ray et al. [109], conducted a very similar experiment for tori of radii $r = 0.7$, $R = 1.3$, using the* High-Order Integration over Discrete Surfaces *(IDS) algorithm [109, Figure 5], resting on total l_1-interpolation degree k, with maximum choice $k = 6$. We perform the same experiment here for DCG and* HOVE *with interpolation degree $k = 6$, employing an initial mesh composed of 544 triangles or equivalently 272 vertices. We subsequently refine the mesh three times, resulting in similar but coarser meshes than the ones reported by [109] Figure 5.18 reports the mesh sizes and the relative errors of all methods. Even though IDS uses meshes of higher resolution, both DCG and* HOVE *outperform IDS. For the rest of this section, we will therefore disregard the IDS algorithm and only compare DCG and HOVE, whereas, for the sake of simplicity, the latter is executed for symmetric or tensorial Gauss rules.*



(a) Spherical harmonic Y_5^4 (b) Integration errors

Figure 5.19: Visualization of the spherical harmonic Y_5^4 (left). Integration errors of DCG and *HOVE* with respect to the interpolation degree. Abbreviations: $HOVE_k$ – interpolating only the geometry, $HOVE_{k,n}$ – interpolating the geometry and the integrand.

Spherical harmonics

The next experiment uses a non-constant integrand. We integrate the 4$^{\text{th}}$-order spherical harmonic, visualized in Figure 5.19, over the unit sphere

$$\int_\Gamma Y_5^4 \, dS = 0\,, \quad Y_5^4(x_1,x_2,x_3) = \frac{3\sqrt{385}(x_1^4 - 6x_2^2x_1^2 + x_2^4)x_3}{16\sqrt{\pi}}\,,$$

vanishing by the L_2-orthogonality of the spherical harmonics. We approximate the unit sphere by a piecewise flat mesh with 496 triangles and compare DCG, *HOVE*, and *HOVE* with the Duffy transformation. The actual integration is performed using a symmetric Gauss triangle rule [43] of order deg $= 25$. Figure 5.19 (right) shows the absolute integration errors as a function of the polynomial degree for two interpolation scenarios:

1. $HOVE_k$ – only interpolating the geometry and sampling the integrand directly in the quadrature nodes of a degree-k-rule.

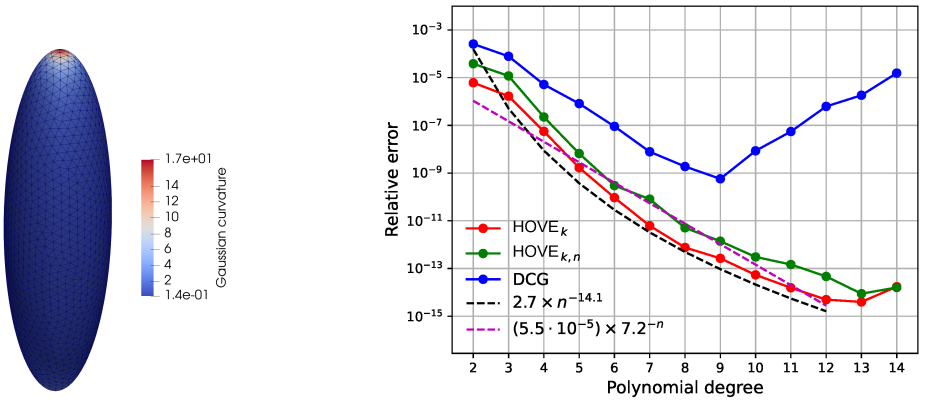
2. $HOVE_{k,n}$ – interpolating the integrand and the geometry with degree $n = k$ and subsequently computing the approximated integral by a degree-k rule.

Both $HOVE_k$ and $HOVE_{k,n}$ converge with exponential rates, $0.05 \cdot 30^{-n}$ fitted for $HOVE_k$, as predicted by Theorem 93. The best fit of an algebraic rate, $9.5 \cdot n^{-17}$, does not assert rapid convergence.

The exponential and algebraic approximation rates of HOVE are fitted using the models $y = c\rho^{-n}$, $y = an^{-b}$, respectively, both achieving an R^2 value of at least 0.99. We observe that all three methods behave similarly for interpolation degrees below 9. For higher degrees, DCG becomes unstable, whereas *HOVE* reaches machine precision for degrees above 10. *HOVE* reaches

one-order-of-magnitude higher accuracy when using square-squeezing instead of Duffy's transformation.

Integrating the Gauss curvature



(a) Ellipsoid with 4024 triangles (b) Integration errors

Figure 5.20: Gauss–Bonnet validation for an ellipsoid with $a = 0.6$, $b = 0.8$, $c = 2$.

In this experiment, we use the Gauss curvature as a non-trivial integrand. By the Gauss–Bonnet theorem [107, 122], integrating the Gauss curvature over a closed surface yields

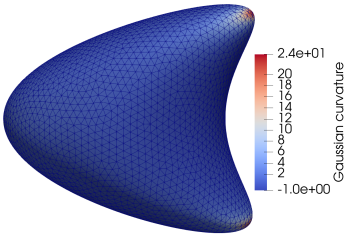
$$\int_\Gamma K_{\text{Gauss}}\, dS = 2\pi\chi(\Gamma), \tag{5.70}$$

where $\chi(\Gamma)$ denotes the Euler characteristic of the surface. We use five surfaces as integration domains. They are given as the zero sets of the following five polynomials:

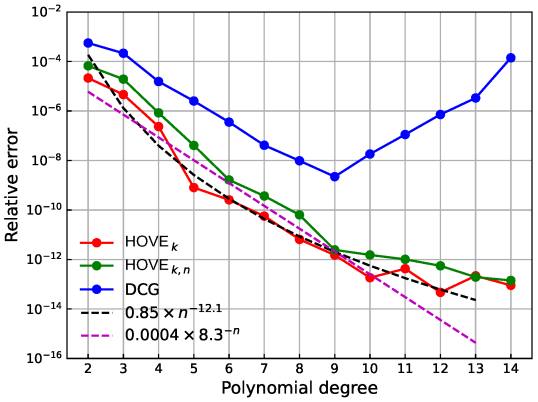
1)	Ellipsoid	$\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 1, \quad a,b,c \in \mathbb{R} \setminus \{0\}$
2)	Torus	$(x^2+y^2+z^2+R^2-r^2)^2 - 4R^2(x^2+y^2) = 0, \quad 0 < r < R \in \mathbb{R}$
3)	Genus 2 surface	$2y(y^2-3x^2)(1-z^2) + (x^2+y^2)^2 - (9z^2-1)(1-z^2) = 0$
4)	Dziuk's surface	$(x-z^2)^2 + y^2 + z^2 - 1 = 0$
5)	Double torus	$\left((x^2+y^2)^2 - x^2 + y^2\right)^2 + z^2 - a^2 = 0, \quad a \in \mathbb{R} \setminus \{0\}$

The surfaces, their parameter choices, and the mesh sizes are shown in Figure 5.20–5.24. The Gauss curvature is computed symbolically from the implicit surface descriptions using MATHEMATICA 11.3. *HOVE* and DCG use (square-squeezing pull-backs of) the symmetric Gauss simplex rules [43] of order 14.

We keep the experimental design from Section 5.5.3 and plot the errors as functions of the polynomial degree in Figure 5.20–5.24. Both $HOVE_k$ and $HOVE_{k,n}$ rapidly converge with exponential rates to the correct value $2\pi\chi(\Gamma)$, except for the thin ellipsoid in Figure 5.20a, where both reach super-algebraic rates. In contrast, DCG fails to reach machine-precision accuracy in all of the cases and becomes unstable when using interpolation degrees k larger than 8.

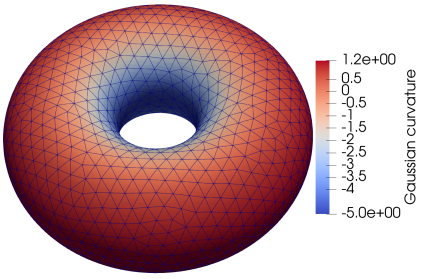


(a) Dziuk's surface with 8088 triangles

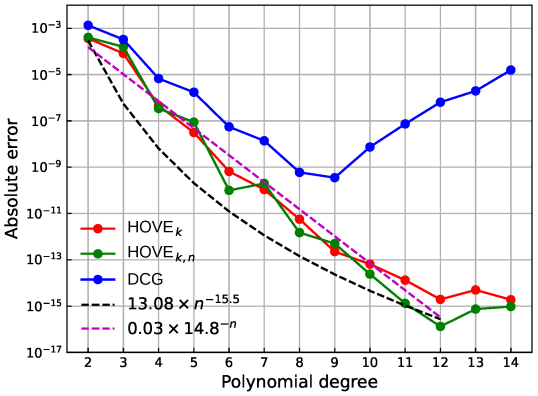


(b) Integration errors for Dziuk's surface

Figure 5.21: Gauss–Bonnet validation for Dziuk's surface.

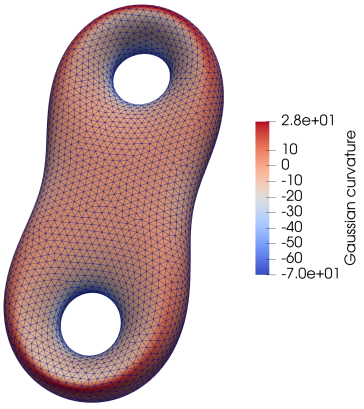


(a) Torus with 1232 triangles

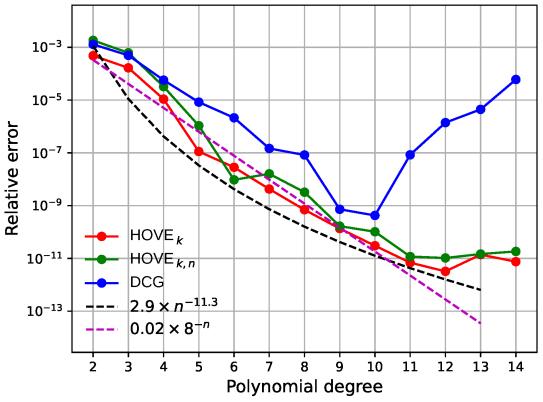


(b) Torus with radii $R = 2$, $r = 1$

Figure 5.22: Gauss–Bonnet validation for a torus with radii $R = 2$, $r = 1$.

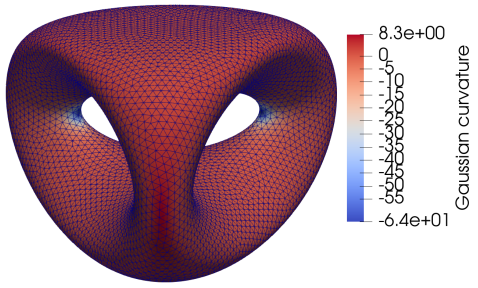


(a) Double torus with 8360 triangles

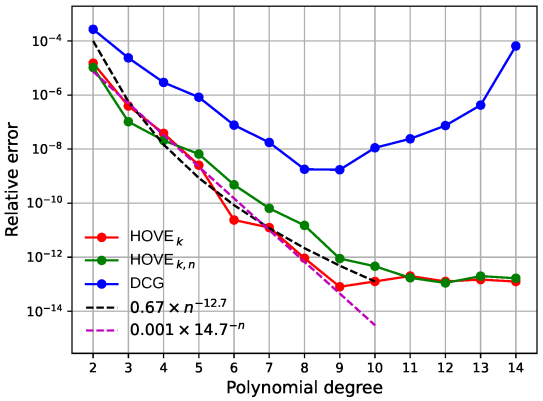


(b) Double torus

Figure 5.23: Gauss–Bonnet validation for a double torus with $a = 0.2$.

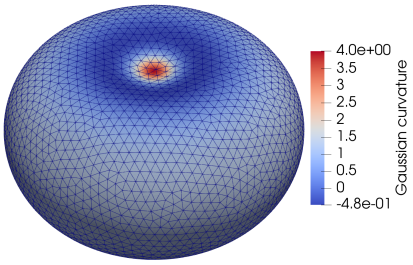


(a) Genus 2 surface, with 15 632 triangles

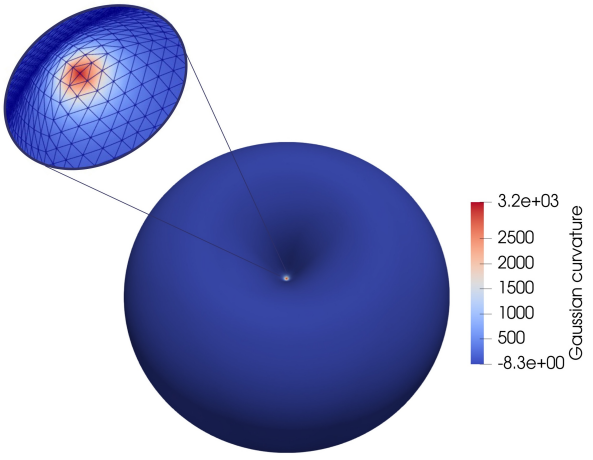


(b) Genus-2 surface

Figure 5.24: Gauss–Bonnet validation for a genus 2 surface.



(a) Biconcave disc $c = -0.934$, $d = 0.80$, with 5980 triangles.



(b) Biconcave disc $c = 0.375$, $d = 0.5$ with 3144 triangles.

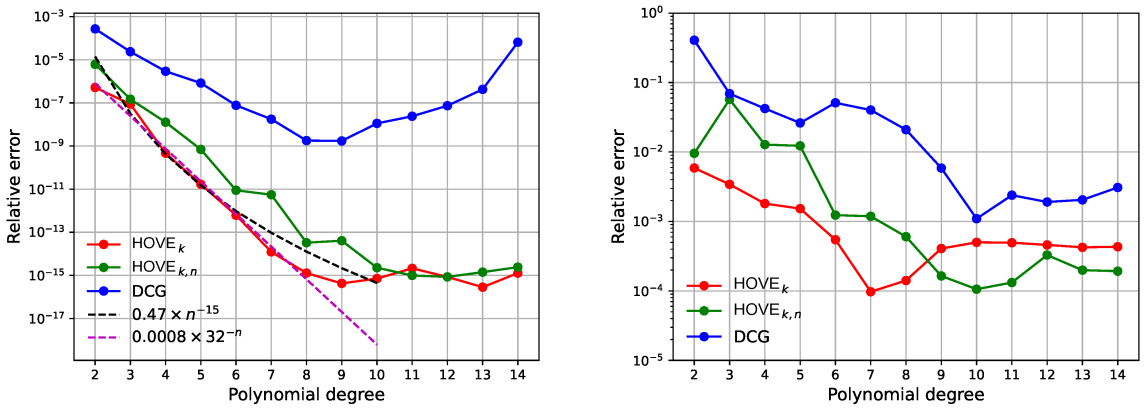
Figure 5.25: Gauss–Bonnet validation for biconcave discs approaching a vertex singularity. The Gauss curvature ranges between $[-4.8 \cdot 10^{-1}]$ and $[4.0]$, Figure 5.25a, and $[-8.3]$ and $[3.2 \cdot 10^{3}]$, Figure 5.25b

A geometry with a near-singularity

The geometries of the previous section have all been well-behaved. In contrast, in this section we now test *HOVE* on a surface that is close to being singular. For this, we consider the biconcave discs shown in Figure 5.25, which are the zero sets of the polynomial

$$P_{\text{bicon}}(x,y,z) = (d^2+x^2+y^2+z^2)^3 - 8d^2(y^2+z^2) - c^4, \qquad c < d \in \mathbb{R}\setminus\{0\}.$$

As long as the parameters c, d are chosen such that $0 \notin P_{\text{bicon}}$ the surfaces are smooth. We consider the two cases $c = -0.934$, $d = 0.8$ and $c = 0.375$, $d = 0.5$, for which the Gauss curvature ranges between $[-4.8 \cdot 10^{-1}]$ and $[4.0]$, and $[-8.3]$ and $[3.2 \cdot 10^{3}]$, respectively, see Figure 5.25b. In the latter case, the Gauss curvature increases rapidly by four orders of magnitude when approaching the center, mimicking cone-like singularities [60] as a challenge for high-accuracy integration.



(a) Biconcave disc, $c = -0.934$, $d = 0.8$

(b) Biconcave disc, $c = 0.375$, $d = 0.5$

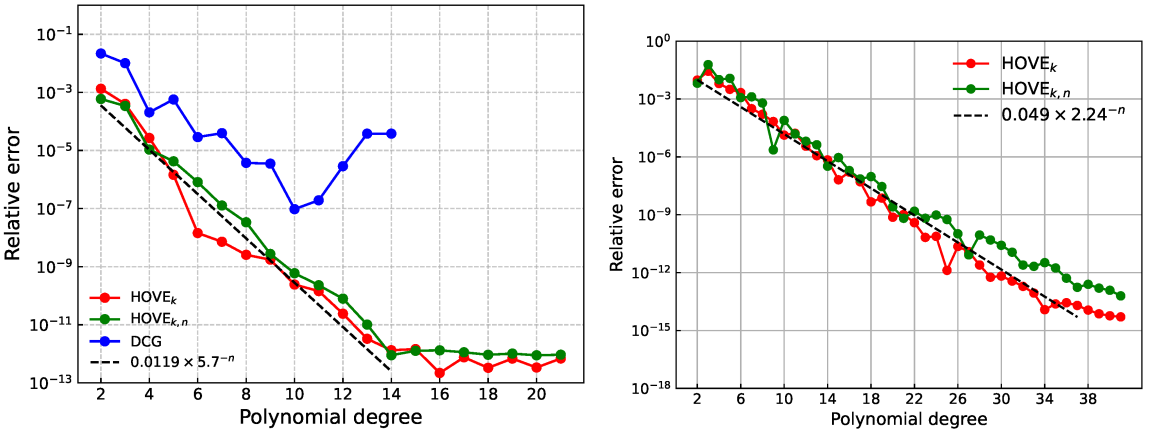
Figure 5.26: Gauss–Bonnet validation for *HOVE* and DCG in case of biconcave discs.

Figure 5.26a shows the Gauss–Bonnet results for the low-curvature case of Figure 5.25a, with $HOVE_k$, $HOVE_{k,n}$ as in Section 5.5.3. Both *HOVE* and DCG converge exponentially up to $\deg = 9$, but DCG has a slower rate, resulting in 5 orders of magnitude higher precision for *HOVE*. For higher orders, the *HOVE* error tends to plateau close to a machine precision level. As in the earlier experiments, DCG becomes unstable in this high-order range.

For the vertex-singularity case in Figure 5.25b, high-order convergence is lost on a uniform surface mesh, independently of the polynomial order employed. In such situations, local mesh refinement in the vicinity of the vertex singularity combined with high-order approximation can be used to recover improved accuracy. To mitigate this effect, we apply a local refinement strategy concentrated near the vertex singularity. Specifically, the mesh is refined only in a neighborhood of that point, resulting in a mesh with 4414 triangles.

The resulting errors are shown in Figure 5.27a. The local refinement restores the expected exponential convergence behavior for both *HOVE* and $HOVE_{k,n}$, with errors closely following the

model curve 0.0119×5.7^{-n} until reaching machine precision. In contrast, the DCG method initially improves but loses stability for higher polynomial degrees and stagnates at significantly larger error levels. Overall, *HOVE* and $HOVE_{k,n}$ outperform DCG by several orders of magnitude while maintaining stable behavior even for high interpolation degrees.



(a) $h-$refinement (b) $p-$refinement

Figure 5.27: Gauss–Bonnet validation for the biconcave disc of Figure 5.25b following a h-refinement strategy with 4414 triangles (5.27b) and a p-refinement strategy for 3144 triangles (5.27b).

As *HOVE* imposes no restrictions on the polynomial degree, we revisit the initial mesh, consisting of 3144 triangles, and increase the geometry approximation degree up to $k = 1, \ldots, 40$. To simplify the integration process, we employ a tensorial Gauss–Legendre quadrature rule of order k equally to the interpolation degrees $k = n$.

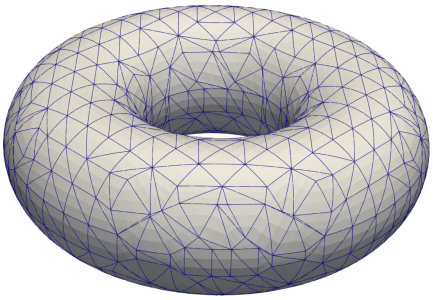
Figure 5.27b shows that *HOVE* achieves exponential convergence under p-refinement until machine precision is reached. These results confirm that the proposed *HOVE* strategy provides a robust framework for integrating over surfaces containing geometric singularities. By combining high-order approximation with localized hp-adaptivity, *HOVE* achieves stable exponential convergence even in the vicinity of the vertex singularity, significantly outperforming DCG.

Mesh quality

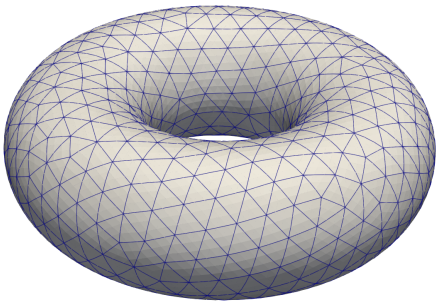
Since the integration error measured here involves in particular the error of approximating the geometry by polynomials it is reasonable to ask whether the integration error depends on the quality of the triangulation of S. To investigate this, we repeat the Gauss–Bonnet validation one final time for the torus of Figure 5.16b.

We generate two meshes for the torus geometry, both with 1232 triangles, shown in Figure 5.28 together with a plot giving the distributions of the interior angles. One of the grids is of high quality, with all angles near 60°. The second mesh was deliberately constructed to be of low quality, featuring a wide range of angles, and even triangles with inverted orientation.

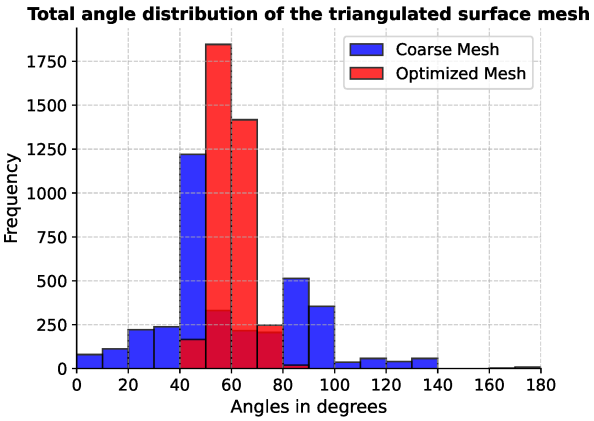
We evaluate the performance of DCG and *HOVE* based on square-squeezing and on Duffy's



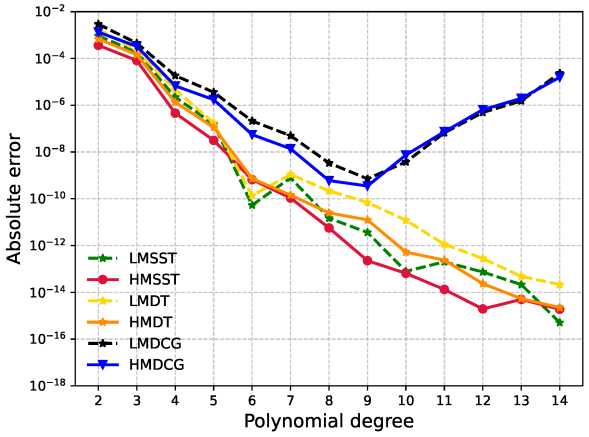
(a) Low-quality mesh with folded triangles



(b) High-quality mesh



(c) Angles of the two meshes



(d) Integration errors

Figure 5.28: Effect of mesh quality on Gauss curvature integration over a torus with radii $r = 1$, $R = 2$. Both low- and high-quality meshes contain 1232 triangles. Abbreviations: LMSST/HMSST denote square-squeezing on low-/high-quality meshes, LMDT/HMDT denote Duffy pullbacks, and LMDCG/HMDCG denote DCG.

transformation, using pull-back quadrature rules of order 14.

The results are given in Figure 5.28d. Neither *HOVE* nor DCG shows a strong dependence on the mesh quality. As usual, *HOVE* converges faster than DCG, and it converges all the way to the machine precision limit. DCG shows the same behavior as in all prior experiments, becoming unstable for geometry approximation orders beyond 8.

Integrating the Gauss curvature using GPLS

In this paragraph, we revisit our earlier experiment on integrating the Gauss curvature over closed surfaces. Our aim is to shift away from the symbolic computation of Gauss curvature using MATHEMATICA 11.3 and instead employ the *global polynomial level set* (GPLS) method [138]. This method is the first step towards enhancing *HOVE*'s capabilities to perform integration on non-parametrized surfaces. For our experiment, we use the vertices of the mesh as sample points and compute their Gauss curvatures based on the GPLS approximation, following equations (4.1). Furthermore, GPLS uniquely enables the evaluation of curvature formulae at any point $\mathbf{x}_0 \in \Gamma$. The meshes utilized in this experiment are identical to those employed in the previous experiment.

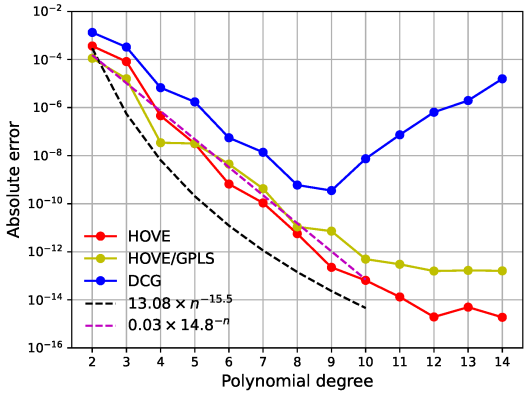
The results are illustrated in Figure 5.29, where we present the errors as functions of the polynomial degree. Both the standard *HOVE* and the *HOVE* enhanced with GPLS exhibit rapid convergence, achieving exponential rates towards the correct value $2\pi\chi(\Gamma)$. However, for the thin ellipsoid shown in Figure 5.20a, both methods attain super-algebraic convergence rates.

This comparison focuses on accuracy. A full runtime comparison is less direct because GPLS includes an additional preprocessing step: the construction of the global polynomial level-set approximation from point-cloud data. Once this representation is available, HOVE can be applied to the resulting implicit surface data. Thus, the computational cost separates into the GPLS fitting cost and the subsequent quadrature cost, whereas the parametrized HOVE experiments measure only the quadrature stage.

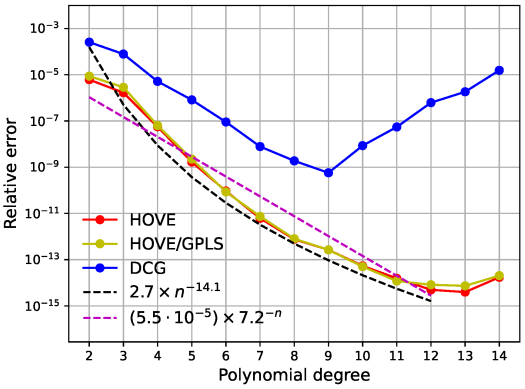
Volume of the surfaces

We revisit the experiment from the previous section, once again considering a unit sphere and a torus. The exact volumes of these shapes can be computed analytically: $\frac{4}{3}\pi r^3$ for the sphere and $2\pi^2 R r^2$ for the torus. These analytical volumes serve as reference solutions in our test. We set $\mathbf{f}(x_1, x_2, x_3) = [0, 0, x_3]$, as derived in earlier sections, to obtain the volume of the solid enclosed by Γ.

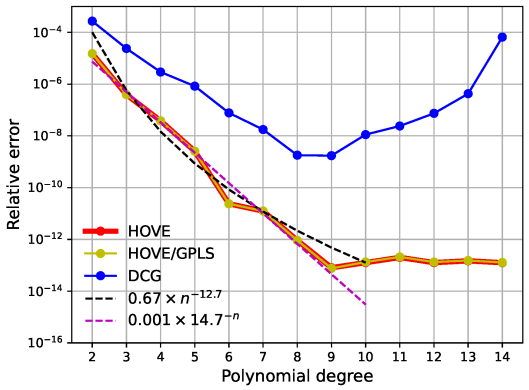
$$V = \int_\Gamma \left(x_3 \cdot \frac{2x_3}{\sqrt{\left(-2R\frac{x_1}{\sqrt{x_1^2+x_2^2}} + 2x_1\right)^2 + \left(-2R\frac{x_2}{\sqrt{x_1^2+x_2^2}} + 2x_2\right)^2 + (2x_3)^2}} \right) dS$$



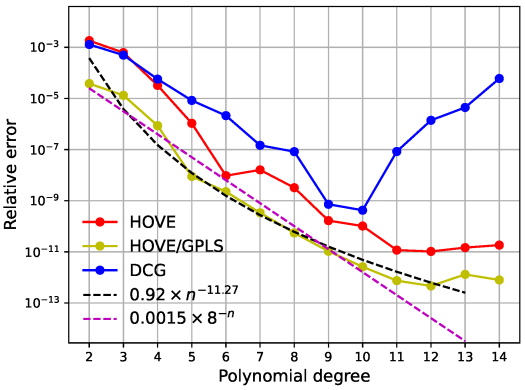
(a) Torus with radii $R = 2$, $r = 1$



(b) Ellipsoid with $a = 0.6$, $b = 0.8$, $c = 2$



(c) Genus 2− surface



(d) Double torus with $a = 0.2$

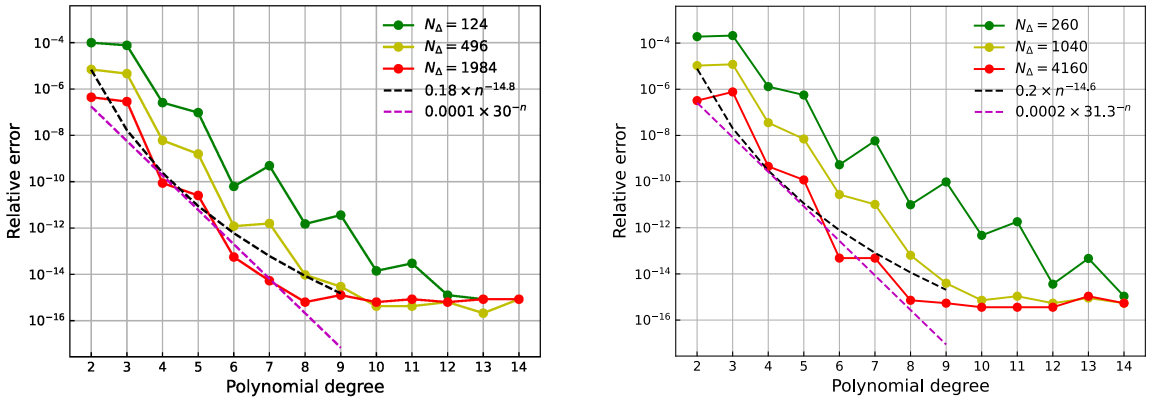
Figure 5.29: Gauss–Bonnet validation using GPLS curvature recovery on four implicit surfaces: torus, ellipsoid, genus 2 surface, and double torus.

Figure 5.30 shows the relative errors of the computed volume of the sphere and torus with respect to the degree of the polynomial interpolation of the geometry. *HOVE* stably converges to machine precision with a high algebraic rate, as predicted by Corollary 94.

5.6 Accelerating surface integration through spectral differentiation

The previous construction stabilizes high-order interpolation by pulling the geometry and integrand back to a tensor-product domain. The remaining computational bottleneck is the evaluation of derivatives of the cubical reparametrization maps. In this section, we address this cost by replacing derivative evaluations based on Newton interpolation polynomials with spectral differentiation, leading to what we call Spectral HOVE. By leveraging Chebyshev–Lobatto nodes (3.2), we construct Chebyshev differentiation matrices to compute the partial derivatives of the k^{th}-order cubical reparametrization maps. These derivatives are then used to approximate the metric tensor, g_i, for each element.

Following [132], the scheme is as follows. Given a grid function v defined on the Chebyshev–Lobatto points x_j^{cheb}, one can obtain the discrete derivative w in two steps:



(a) Unit sphere

(b) Torus with radii $r = 1$ and $R = 2$

Figure 5.30: Relative errors of $HOVE_k$ for computing the enclosed volume of the unit sphere and torus using three successively refined meshes.

1. Let $Q_{G_{1,k}} f$ be a unique polynomial of degree $\leq k$ with $Q_{G_{1,k}} f(x_j^{\text{cheb}}) = f_j$, $0 \leq j \leq k$. This operation is linear, so it can be represented by multiplication by an $(k+1) \times (k+1)$ matrix.
2. Set $w_j = \partial_x Q_{G_{1,k}} f(x_j^{\text{cheb}})$.

We shall denote this matrix by D, so that $w = Df$. The first-order differentiation matrix is described by the following theorem:

Theorem 98. [132] *For each $k \geq 1$, let the rows and columns of the $(k+1) \times (k+1)$ Chebyshev differentiation matrix D be indexed from 0 to k. The entries of this differentiation matrix are given by:*

$$D_{ij} = \begin{cases} \frac{2k^2+1}{6}, & \text{if } i = j = 0, \\ -\frac{2k^2+1}{6}, & \text{if } i = j = k, \\ \frac{c_i}{c_j} \frac{(-1)^{i+j}}{x_i^{cheb} - x_j^{cheb}}, & \text{if } i \neq j, \\ 0, & \text{otherwise}, \end{cases} \tag{5.71}$$

where $i, j = 0, 1, \ldots, k$, x_j^{cheb} are the Chebyshev–Lobatto points, and c_i are given by:

$$c_i = \begin{cases} 2, & \text{if } i = 0 \text{ or } i = k, \\ 1, & \text{otherwise}. \end{cases} \tag{5.72}$$

For $i, j = 1, \ldots, k-1$, the entries D_{ij} make the pattern clearer:

$$D_{ij} = \frac{c_i}{c_j} \frac{(-1)^{i+j}}{x_i^{cheb} - x_j^{cheb}}, \quad i \neq j. \tag{5.73}$$

To obtain the partial derivatives and complete the Jacobian matrix $\mathbf{D}Q_{G_{d,k}}\varphi_i$ on the reference square $[-1,1]^2$, specifically in the tensor grid, the simplest method is to use tensor products in linear algebra, also known as *Kronecker products*. Now, let $D \in \mathbb{C}^{(k+1)\times(k+1)}$ be the one-dimensional spectral differentiation matrix associated with Chebyshev–Lobatto nodes on the interval $[-1,1]$, and let $I \in \mathbb{C}^{(k+1)\times(k+1)}$ be the identity matrix. The differentiation matrices in the x and y directions on the reference square $[-1,1]^2$ are defined as follows:

$$D_{\mathbf{x}}Q_{G_{2,k}}\varphi_i = D \otimes I, \quad D_{\mathrm{y}}Q_{G_{2,k}}\varphi_i = I \otimes D, \tag{5.74}$$

where $D_{\mathrm{x}}Q_{G_{2,k}}\varphi_i, D_{\mathrm{y}}Q_{G_{2,k}}\varphi_i \in \mathbb{C}^{(k+1)^2\times(k+1)^2}$.

Representing the Jacobian as $\mathbf{D}Q_{G_{d,k}}\varphi_i = [D_{\mathrm{x}}Q_{G_{d,k}}\varphi_i \,|\, D_{\mathrm{y}}Q_{G_{d,k}}\varphi_i]$, the ingredients above realise the HOVE, computing the surface integral as:

$$\int_\Gamma f\,dS \approx \sum_{i=1}^{K} \int_{[-1,1]^2} f(\varphi_i(\mathbf{x}))\sqrt{\det((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))}\,d\mathbf{x} \tag{5.75}$$

$$\approx \sum_{i=1}^{K}\sum_{\mathrm{p}\in P} \omega_{\mathrm{p}} f(\varphi_i(\mathrm{p}))\sqrt{\det((\mathbf{D}Q_{G_{d,k}}\varphi_i(p))^T\mathbf{D}Q_{G_{d,k}}\varphi_i(p))}\,, \tag{5.76}$$

where $\mathrm{p} \in P$, $\omega_{\mathrm{p}} \in \mathbb{R}^+$ can be the points and weights of any quadrature of $[-1,1]^2$ e.g. the Gauss–Legendre or Clenshaw–Curtis quadrature [133, 143].

It is important to note that computing nodes and weights for a n-point Gauss–Legendre rule requires $\mathcal{O}(n^2)$ operations, while the Clenshaw–Curtis method uses $\mathcal{O}(n\log n)$ operations with the *Discrete Cosine Transform* for evaluation.

The integration error of HOVE has two primary sources. The first is the approximation of the 2D integral using the tensor-product Clenshaw–Curtis quadrature provided in Theorem 99. The second is the approximation error in representing the geometry φ_i, which scales with $\mathcal{O}\left(k^{-(r-d-1)}\right)$, where k denotes the chosen polynomial degree; see Theorem 93.

We directly estimate the integration error from approximating a 2D integral using tensor-product Clenshaw–Curtis quadrature:

Theorem 99. *Consider a C^{r+1} surface Γ, where $r \geq 1$, with r-regular quadrilateral reparametrization $\varphi_i : [-1,1]^2 \to \Gamma$,*

$$\varphi_i = \rho_i \circ \sigma = \pi_i \circ \tau_i \circ \sigma, \quad i = 1,\ldots,K.$$

Let $\mathrm{p} \in P$, and ω_{p} be the points and weights of the tensorial $(n+1)$-order Clenshaw–Curtis quadrature rule, $f : \Gamma \to \mathbb{R}$ be a function with absolutely continuous derivatives up to order $(r-1)$ and the r^{th} derivative $f^{(r)}$ is of bounded variation $\|f^{(r)}\|_T = V < +\infty$, such that f induces a negligible "remainder of the remainder" [46]. Then the integration error can be estimated as

$$\mathcal{E}_f = \sum_{i=1}^{K}\int_{[-1,1]^2} f(\varphi_i(\mathbf{x}))g_i(\mathbf{x})\,d\mathbf{x} - \sum_{i=1}^{K}\sum_{\mathrm{p}\in P}\omega_{\mathrm{p}} f(\varphi_i(\mathrm{p}))g_i(\mathrm{p}) \leq \frac{128\tilde{V}}{15\pi r(n-r)^r}, \tag{5.77}$$

where $g_i(\mathbf{x}) = \sqrt{\det((\mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))^T \mathbf{D}Q_{G_{d,k}}\varphi_i(\mathbf{x}))}$ *and* $\tilde{V}$ *is defined as*

$$\tilde{V} = \max_{i=1,\dots,K} \max_{\alpha,\beta \le r} \left\| \frac{\partial^{\alpha+\beta}}{\partial x^\alpha \partial y^\beta} \Big(f(\varphi_i(\mathbf{x})) g_i(\mathbf{x}) \Big) \right\|_T \quad \text{with } \|\xi\|_T = \left\| \frac{\xi'}{\sqrt{1-x^2}} \right\|_1 . \tag{5.78}$$

Proof. We denote $\psi(x,y) := f(\varphi_i(\mathbf{x}))g_i(\mathbf{x})$ and

$$R_n^{(2)}[\psi] = I^{(2)}[\psi] - Q_n^{(2)}[\psi] \tag{5.79}$$

as the remainder of the exact integral and the $n+1$-order quadrature rule

$$I^{(2)}[\psi] = \int_{[-1,1]^2} \psi(x,y)\,dx\,dy, \quad Q_n^{(2)}[\psi] = \sum_{j=0}^{n}\sum_{i=0}^{n} \psi(x_i,y_j)\,w_i\,w_j\,. \tag{5.80}$$

For a bivariate function $\psi(x,y)$, $I_x[\psi] = \int_{-1}^{1} \psi(x,\cdot)\,dx$ denotes the integration with respect to the x variable only, yielding a function of y. The subscript notation extends to $R_{n,x}[\psi]$ and $Q_{n,y}[\psi]$, when replacing the roles of x and y. Fubini's theorem [20] implies:

$$I^{(2)}[\psi] = \int_{[-1,1]^2} \psi(x,y)\,dx\,dy = \int_{-1}^{1} I_x[\psi]\,dy = I_y[I_x[\psi]]\,. \tag{5.81}$$

Upon substitution into equation (5.81), we obtain

$$\begin{aligned} I^{(2)}[\psi] &= I_y[R_{n,x}[\psi] + Q_{n,x}[\psi]] \\ &= R_{n,y}[R_{n,x}[\psi] + Q_{n,x}[\psi]] + Q_{n,y}[R_{n,x}[\psi] + Q_{n,x}[\psi]] \\ &= R_{n,y}[R_{n,x}[\psi]] + Q_{n,x}[R_{n,y}[\psi]] + Q_{n,y}[R_{n,x}[\psi]] + Q_{n,y}[Q_{n,x}[\psi]] \end{aligned} \tag{5.82}$$

Following [46] we assume the "remainder of the remainder" – first term in equation (5.82) to contribute negligibly to the error, enabling us to establish a sufficiently tight upper bound:

For large n, the quadrature rule approaches the value of the integral, i.e., $Q_{n,\beta} \approx I_\beta$ for $\beta = x,y$, we are left with:

$$I^{(2)}[\psi] \approx I_x[R_{n,y}[\psi]] + I_y[R_{n,x}[\psi]] + Q_n^{(2)}[\psi]\,. \tag{5.83}$$

Hence,

$$R_n^{(2)}[\psi] \le I_x[R_{n,y}[\psi]] + I_y[R_{n,x}[\psi]], \tag{5.84}$$

As noted in [133], considering a function $\theta(x)$ defined on the interval $[-1,1]$, when computing $Q_n[\theta]$ using Clenshaw–Curtis quadrature for $\theta \in C^r$ and $\|\theta^{(r)}\|_T < V$ for a real finite value V, then for sufficiently large n, the subsequent inequality holds:

$$R_n[\theta] \le \frac{32V}{15\pi r(n-r)^r}\,. \tag{5.85}$$

Consequently, applying equation (5.85) to equation (5.84) yields

$$R_n^{(2)}[\psi] \leq \frac{32}{15\pi r(n-r)^r}[I_x[V_y(x)] + I_y[V_x(y)]], \tag{5.86}$$

where $V_y(x) = \max_y \|\psi^{(r)}(x,y)\|_T$ and $V_x(y) = \max_x \|\psi^{(r)}(x,y)\|_T$ for fixed x and y, respectively.

Since both integrations are over $[-1,1]$, and since $V_x(y) \leq \tilde{V}$ and $V_y(x) \leq \tilde{V}$ by the definition of $\tilde{V}$, we have

$$I_x[V_y(x)] + I_y[V_x(y)] \leq 2\tilde{V} + 2\tilde{V} = 4\tilde{V}.$$

Substituting this bound into the previous inequality gives (5.77).

Applying Clenshaw–Curtis quadrature therefore leads to an error approximation for a 2D surface integral with an order of $\mathcal{O}(n^{-r})$, where n denotes the quadrature's order. □

5.7 Computational results

We demonstrate the effectiveness of Spectral HOVE with Clenshaw–Curtis quadrature (referred to in our plots as HOVE$_k$-CC), for surface integration through two sets of numerical examples. The implementation of Spectral HOVE is provided in the Python-based codebase surfpy,[7] while the examples and results presented in this manuscript, using Dune-CurvedGrid, are summarized and made accessible in the following repository:[8]

Unless otherwise stated, the spectral experiments report relative errors against analytical reference values and use the same geometry degree for interpolation, spectral differentiation, and quadrature. Runtime plots include the construction and evaluation costs of the corresponding method, so the comparison reflects the practical effect of replacing Newton interpolation derivatives by Chebyshev differentiation matrices.

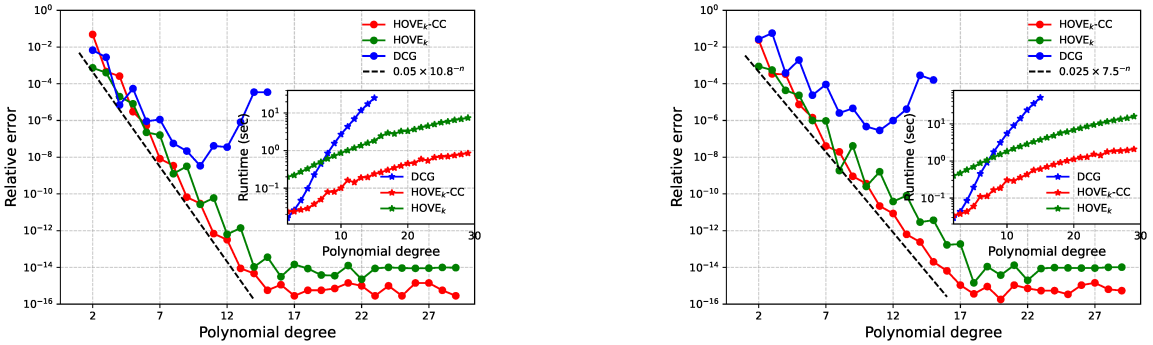
Spectral HOVE is compared with both the earlier approach, referred to in our plots simply as HOVE$_k$, where derivatives are computed using Newton interpolation polynomials, and the DCG method [106], which employs total-degree polynomial interpolation and is known to be sensitive to Runge's phenomenon due to overfitting.

Area computation

We integrate $f = 1$ over the unit sphere and the torus $T^2_{r,R}$ with inner radius $r = 1$ and outer radius $R = 2$. The surface areas are given by 4π and $4\pi^2 Rr$, respectively, which allows us to measure the relative errors.

We choose initial triangulations of size $N_\Delta = 124$ for the sphere and of size $N_\Delta = 256$ for the torus and apply Clenshaw–Curtis quadrature [133], with a degree matching the geometry approximation. We use symmetric Gauss quadrature on a simplex [43] with a matching degree for the geometry approximation in the case of HOVE$_k$ and DCG.

[7] https://github.com/casus/surfpy.
[8] https://github.com/casus/dune-surface_int.



(a) Unit sphere

(b) Torus with radii $R = 2$ and $r = 1$

Figure 5.31: Relative errors and computational runtimes for DCG, HOVE$_k$, and Spectral HOVE in the computation of the surface area of the unit sphere (5.31a) and the torus (5.31b).

Both Spectral HOVE and HOVE$_k$ exhibit stable convergence to machine precision with exponential rates, 0.05×10.8^{-n} and 0.025×7.5^{-n} fitted for the sphere and torus, respectively, in accordance with the predictions from Theorem 99. In contrast, DCG becomes unstable for orders larger than $\deg = 10$. We interpret the instability as the appearance of Runge's phenomenon caused by the choice of midpoint-triangle refinements yielding equidistant interpolation nodes for DCG.

In terms of execution time, the PYTHON implementation of Spectral HOVE, leveraging spectral differentiation, outperforms both the earlier HOVE$_k$ implementation based on Newton interpolation polynomials and the C++ implementation of DCG, as illustrated by the runtime plots. While HOVE$_k$ already improves over DCG, the spectral differentiation strategy used in Spectral HOVE leads to further reductions in computational cost. Moreover, the numerical experiments indicate that DCG becomes unreliable for polynomial degrees $\deg \gtrsim 15$, due to the emergence of a singular matrix during the construction of the interpolation basis.

Gauss–Bonnet validation

We consider the Gaussian curvature as a non-trivial integrand. Due to the Gauss–Bonnet theorem [122], integrating the Gaussian curvature over a closed surface yields

$$\int_\Gamma K_{\text{Gauss}}\, dS = 2\pi\chi(\Gamma)\,, \tag{5.87}$$

where $\chi(\Gamma)$ denotes the Euler characteristic of the surface. Here, we consider:

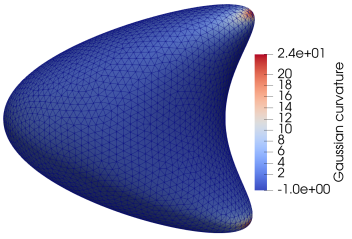
1) Dziuk's surface: $(x - z^2)^2 + y^2 + z^2 - 1 = 0$.

2) Double torus: $\left((x^2+y^2)^2 - x^2 + y^2\right)^2 + z^2 - a^2 = 0$, with $a \in \mathbb{R} \setminus \{0\}$.

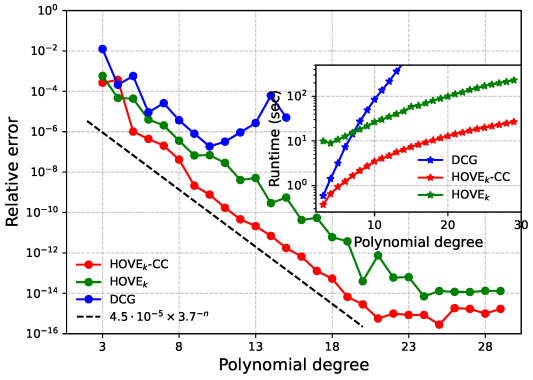
3) Swiss cheese block:

$$\left(x^2+y^2-4\right)^2+\left(z^2-1\right)^2+\left(y^2+z^2-4\right)^2\left(x^2-1\right)^2 \\ +\left(z^2+x^2-4\right)^2+\left(y^2-1\right)^2-15=0.$$

The Gaussian curvature is computed symbolically from the implicit surface descriptions using MATHEMATICA 11.3, which allows us to measure the errors produced by HOVE$_k$, DCG, and Spectral HOVE when integrating the Gaussian curvature. We maintain the experimental design outlined in Experiment 5.7 and display error plots based on the polynomial degree in Figures 5.32–5.34.



(a) Dziuk's surface with 8088 triangles



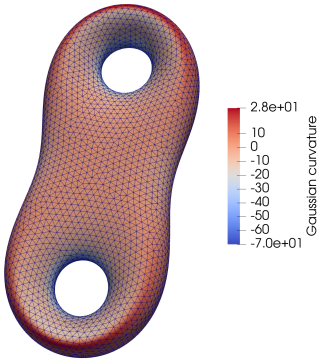
(b) Errors and runtimes for Dziuk's surface

Figure 5.32: Gauss–Bonnet validation for Dziuk's surface using DCG, HOVE$_k$, and Spectral HOVE.

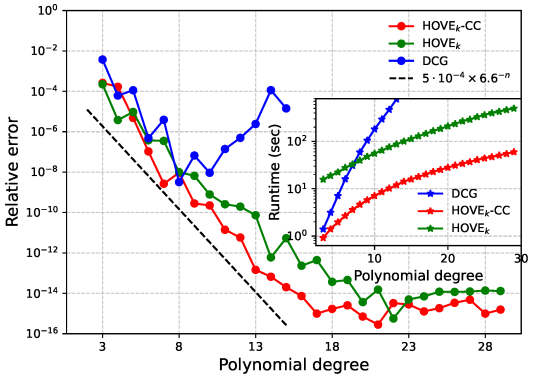
Again, both Spectral HOVE and HOVE$_k$ consistently demonstrate exponential convergence toward the exact value $2\pi\chi(\Gamma)$. The observed exponential rates are approximately $4.5\cdot 10^{-5}\times 3.7^{-n}$ for Dziuk's surface, $5\cdot 10^{-4}\times 6.6^{-n}$ for the double torus, and 0.054×4.72^{-n} for the Swiss cheese block. In contrast, the best algebraic fits—$3.95\times n^{-12.16}$ for Dziuk's surface, $1.7\times n^{-12.3}$ for the double torus, and $5.2\times n^{-9.88}$ for the Swiss cheese block—indicate significantly slower convergence. As the polynomial degree increases, the errors for both Spectral HOVE and HOVE$_k$ eventually stabilize near machine precision. By contrast, DCG fails to reach machine-precision accuracy in all cases and becomes unstable for polynomial degrees $k \geq 10$. In terms of execution time, Spectral HOVE, leveraging spectral differentiation, clearly outperforms both HOVE$_k$ and DCG.

In this final experiment, we compare the proposed approach with state-of-the-art interpolation node sets on the simplex. In particular, we consider recursive interpolation nodes [75] constructed from one-dimensional interpolation points. The resulting node sets are fully symmetric within the triangle and reproduce the corresponding one-dimensional interpolation nodes along its edges.

The left plot illustrates the growth of the Lebesgue constant for the different node distributions,

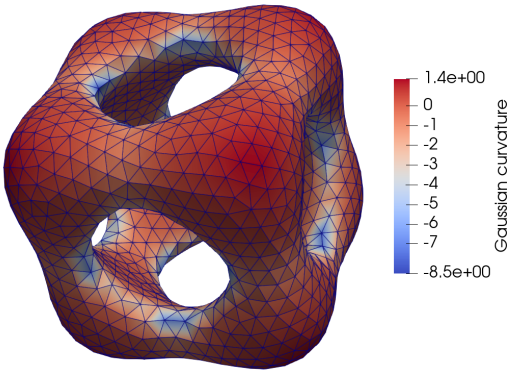


(a) Double torus with 8360 triangles and $a = 0.2$

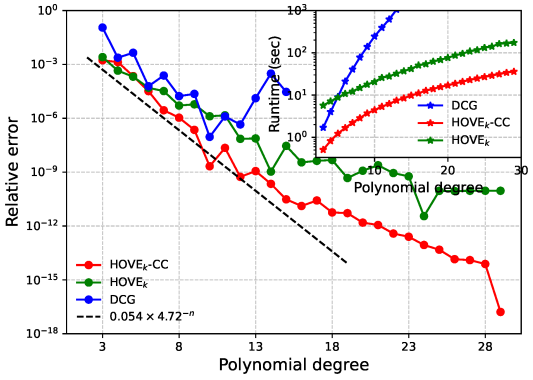


(b) Errors and runtimes for the double torus

Figure 5.33: Gauss–Bonnet validation for a double torus using DCG, HOVE$_k$, and Spectral HOVE.



(a) Swiss cheese block with 2944 triangles



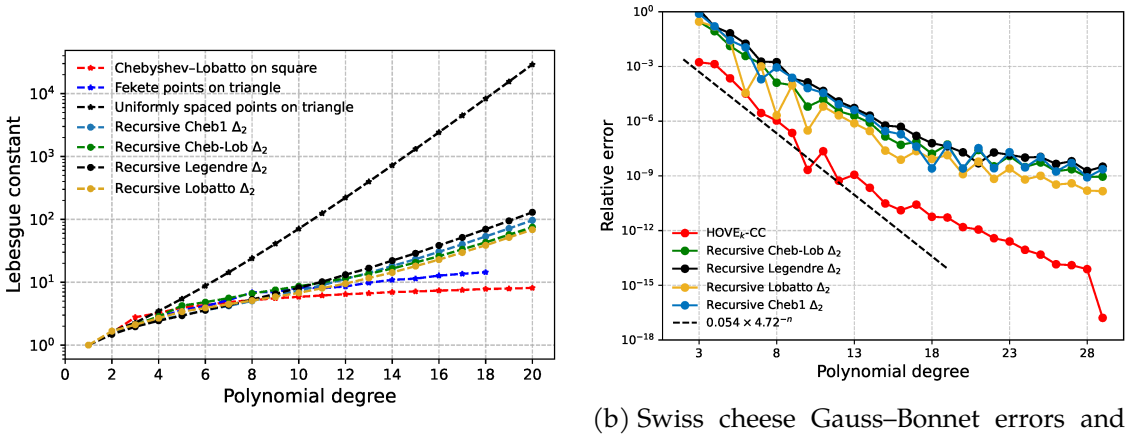
(b) Errors and runtimes for the Swiss cheese block

Figure 5.34: Gauss–Bonnet validation for the Swiss cheese block using DCG, HOVE$_k$, and Spectral HOVE.

providing a measure of interpolation stability. The right plot shows the corresponding integration errors as a function of the polynomial degree. As expected, node sets with controlled Lebesgue growth lead to significantly improved numerical accuracy.

The results indicate that Spectral HOVE achieves the fastest convergence and reaches machine precision, outperforming the competing constructions, including interpolation nodes close to Fekete points on the simplex.

These experiments demonstrate that Spectral HOVE achieves spectral accuracy while maintaining numerical stability and improved computational efficiency, making it a reliable high-order method for the surface discretizations developed in this work as well as for practical surface integration tasks. A key advantage of the proposed approach is that it bypasses the need to construct optimal interpolation or quadrature nodes directly on triangles. Instead, the interpolation problem is pulled back to a tensor-product domain, where well-understood nodes such as Chebyshev points can be employed and their favorable approximation properties can be fully



(a) Lebesgue constants of simplex node sets

(b) Swiss cheese Gauss–Bonnet errors and runtimes

Figure 5.35: Lebesgue constants (5.35a) of uniformly spaced points, Fekete points, Chebyshev–Lobatto points, and recursive interpolation points on simplices, together with Gauss–Bonnet validation for the Swiss cheese block (5.35b).

exploited.

Overall, the chapter connects three complementary contributions. First, the even–odd convergence effect for equidistant interpolation on refined triangular meshes is explained through a cancellation mechanism. Second, the square-squeezing construction replaces unstable high-degree simplex interpolation by stable tensor-product interpolation, leading to HOVE and its high-order error estimates. Third, spectral differentiation reduces the cost of evaluating the same tensor-product representation while preserving the observed accuracy. These points provide the basis for the future directions discussed next.

5.8 Future work

While this chapter has made significant progress in high-order numerical integration on surfaces, one particularly challenging yet promising direction—suggested by our preliminary results (see Figure 5.29)—is the extension of our methods to non-parametrized surfaces using the Global Polynomial Level Set (GPLS) framework. By integrating regression-based techniques [137], this approach could enable the computation of surface integrals even when the integrand is known only at a priori given data points.

Chapter 6

Fast spectral methods on triangular and deforming surfaces

Partial differential equations (PDEs) posed on both static and evolving surfaces have been discretized using a range of numerical techniques. Among the most widely used are finite difference, finite element, and spectral methods. Finite difference methods are relatively easy to implement but exhibit only algebraic convergence, necessitating a large number of grid points and, consequently, significant memory usage. In contrast, finite element methods—particularly the surface finite element methods (SFEMs) introduced by Dziuk and Elliott [44]—approximate the solution using low-order basis functions defined over a collection of elements. Surface discretizations based on SFEMs typically employ at most cubic or quartic polynomial approximations [77]. The resulting linear systems may be ill-conditioned. Nevertheless, the use of efficient solvers—such as multigrid methods [22] and sparse direct solvers like UMFPACK [37]—has contributed to the popularity of this approach. These methods offer geometric flexibility, support for variable coefficients, and are well integrated into existing software frameworks like DUNE [9]. However, achieving high-order accuracy with such methods remains challenging, particularly when the underlying surface mesh possesses only low-order smoothness.

Spectral methods, on the other hand, attain high accuracy using relatively few elements by employing high-degree polynomial approximations. This approach is especially efficient for smooth problems defined on regular geometries. Spectral element methods were introduced in [49] with the underlying principle of combining the generality of finite element methods for complex geometries with the accuracy of spectral methods. However, they often lead to dense,

poorly conditioned systems resulting in $\mathcal{O}\big((n_x n_y)^3\big)$ complexity [1] in space. Nevertheless, when combined with domain decomposition techniques—such as the Schwarz algorithm [27] or the hierarchical Poincaré–Steklov scheme [58]—spectral element methods can achieve a solver complexity of $\mathcal{O}\big(N^{3/2}\big)$ or even $\mathcal{O}(N)$, where N denotes the total number of degrees of freedom. Such methods always compute solutions that are piecewise smooth, which can lead to a suboptimal number of degrees of freedom required. These methods benefit from keeping the polynomial order per element relatively low, which improves conditioning, while still achieving high global accuracy due to the spectral nature of the approximation.

The method described in this chapter is a domain decomposition scheme with spectral collocation discretization on each element. While spectral collocation is used throughout, it is important to note that, in principle, other element-wise discretization techniques—such as finite element methods—could also be employed. By combining the spectral domain decomposition with a hierarchical direct solver closely related to the classical nested dissection technique [55], the resulting approach is known in the literature as the hierarchical Poincaré–Steklov (HPS) solver [50, 58, 91, 92]. The HPS scheme operates by constructing a binary tree of surface elements (or "leaves"). At the leaf level, local degrees of freedom are eliminated—typically by constructing Dirichlet-to-Neumann (or equivalent Schur complement) operators. These interface operators are then hierarchically merged in a recursive fashion, coupling neighboring patches and progressively assembling the global solution. This structure allows for efficient and scalable direct solvers without forming or factorizing a global stiffness matrix explicitly. These solvers are particularly robust in the presence of large condition numbers and are well-suited for oscillatory or stiff PDEs, where constructing effective preconditioners is often challenging. This contrasts with iterative solvers, which can suffer from slower convergence in non-trivial geometries or when dealing with higher-order discretizations.

For fixed geometries, the computational work of the HPS method is naturally divided into a build stage and a solve stage. During the build stage, the local solution operators and the hierarchical merge operators are constructed; this is the expensive part of the method. Once this stage has been completed, the resulting factorization can be reused for multiple right-hand sides, or across many time steps when the surface and the linear operator remain unchanged. Thus, the build cost is amortized in applications requiring repeated solves on the same geometry.

Although the spectral domain decomposition coupled with the hierarchical direct solver introduced in Section 6.3 is, in principle, applicable to a broad range of geometries, its successful application to date has been primarily limited to high-order spectral discretizations on quadrilateral surface meshes [50, 91]. This is largely due to the availability of well-developed high-order points and basis functions on the reference square $[-1,1]^2$, which—when combined with smooth parameterizations—enable accurate discretization of PDEs on quadrilateral patches [50]. A similar approach can be extended to more general surface representations, such as triangulated meshes commonly used in practical applications. In such cases, the overall methodology

[1] for $2-$D problems where n_x and n_y are the numbers of discretization points along the x- and y-axes, respectively.

remains largely unchanged, as long as high-order discretizations can be defined on a reference triangle, allowing each element to be treated with the same level of accuracy as in the quadrilateral setting.

In this chapter, we take an initial step toward incorporating triangulated surfaces into the HPS framework by proposing two strategies. The first sections recall the HPS machinery needed later in the chapter. The main new geometric contributions begin in Section 6.4, where we address how triangulated surface data can be made compatible with a solver whose local operators are naturally built on tensor-product quadrilateral patches. The standard HPS construction and merging procedure are reviewed as the solver backbone; the new emphasis is on the geometric treatment of triangulated surface data and on the use of the resulting high-order framework for static and prescribed evolving geometries. The chapter is organized into two parts:

To avoid repeating the foundations developed earlier in the thesis, we use the Chebyshev tensor-product interpolation notation from Chapter 3, the surface differential calculus and Laplace–Beltrami notation from Section 4.1.3, and the closest-point projection framework from Section 4.1.2. The square-squeezing map used below is the hypercube-to-simplex map of Definition 85, and the spectral differentiation viewpoint follows Section 5.6. Thus, the additional mathematical issue treated in this chapter is not the construction of these objects themselves, but their coupling through the HPS interface operators.

The first part describes the domain decomposition strategy underlying the HPS scheme, originally developed for quadrilateral meshes [50, 91], and introduces two parametrization approaches for triangular surface meshes. We begin in Section 6.1 by discussing high-order representations of surface geometry and proceed in Section 6.2 with the construction of local spectral collocation operators on individual surface elements. Section 6.3 then details the domain decomposition approach on surfaces, where spectral collocation is applied locally on each element and coupling between neighboring elements is handled via interface conditions. Section 6.4 presents the two strategies for incorporating triangulated surfaces into the HPS framework: (i) a hypercube-to-simplex reparametrization, and (ii) a quadrilateralization technique designed to reduce the number of elements added by the tripling scheme of Fortunato et al. [51]. The quadrilateralization strategy is the practical route used for the HPS computations in this chapter, whereas the hypercube-to-simplex construction is presented as a promising approach whose full integration into HPS requires a more flexible interface treatment. In Section 6.5, we discuss how the resulting high-order surface discretization can be applied to time-dependent problems. Finally, we conclude the first part with Section 6.6, which presents numerical examples investigating the Turing model and extends the investigation to patterns exhibited by interacting reaction–diffusion systems.

The second part extends the framework developed in the first part to prescribed evolving surfaces. Section 6.7 introduces a time-dependent surface representation and describes the extension of the projection method to include surface evolution. Section 6.8 discusses the treatment of

evolving discrete surfaces and introduces an *Arbitrary Lagrangian–Eulerian* (ALE) map to manage mesh distortion. In Sections 6.9 and 6.10, we focus on two growth models for the evolution of the surface and investigate how these models affect the emergence, spatial distribution, and symmetry of patterns over time. Finally, Section 6.11 explores potential directions for future work.

6.1 High-order parametric surface approximation

We consider a general elliptic surface PDE on Γ,

$$\mathcal{L}_\Gamma u(\mathbf{x}) = f(\mathbf{x}), \quad \mathbf{x} \in \Gamma, \tag{6.1}$$

where $f(\mathbf{x})$ is a smooth function on Γ and $\mathcal{L}_\Gamma$ is a variable-coefficient linear second-order elliptic surface operator.

If Γ is not a closed surface, equation (6.1) may also be subject to boundary conditions, e.g., $u(\mathbf{x}) = h(\mathbf{x})$ for $\mathbf{x} \in \partial\Gamma$ and some function h . For pure Laplace–Beltrami problems on a closed surface, the usual compatibility and normalization conditions are required: $\int_\Gamma f\, dS = 0$ and the solution is fixed, for example, by imposing $\int_\Gamma u\, dS = 0$. This nullspace issue is absent for shifted operators such as $-\Delta_\Gamma + I$ and for the patchwise Dirichlet tests below.

To numerically solve equation (6.1), we need to discretize $\mathcal{L}_\Gamma, f$, and h, and represent these objects in some finite-dimensional basis. The numerical error involved in a discretization of PDEs on curved surfaces depends on two properties of the discretization, the representation of the objective function and the representation of the geometry. Thus, a higher-order scheme is only possible with also a higher-order description of the surface approximation. Let $\Gamma_h^{\text{quad}} \subset \mathcal{N}_\delta$ be a shape-regular, composed of finitely many regular and quasi-uniform d-dimensional hypercubes with diameter h, topologically equivalent to the smooth surface Γ. The collection of these hypercubes is denoted by $\widehat{\mathcal{K}}_h$, which provides a representation of Γ_h^{quad}:

$$\Gamma_h^{\text{quad}} = \bigcup_{\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h} \widehat{\mathcal{E}}, \text{ with } K = |\widehat{\mathcal{K}}_h|. \tag{6.2}$$

We assume that the elements do not overlap, i.e., for $\widehat{\mathcal{E}}_i, \widehat{\mathcal{E}}_j \in \widehat{\mathcal{K}}_h$, we have that $\text{int}(\widehat{\mathcal{E}}_i) \cap \text{int}(\widehat{\mathcal{E}}_j) = \emptyset$ and if $\widehat{\mathcal{E}}_i \cap \widehat{\mathcal{E}}_j = I \neq \emptyset$ and $\dim(I) = d-1$, it is called an intersection of $\widehat{\mathcal{E}}_i$ and $\widehat{\mathcal{E}}_j$ and is assumed to be a subset of an $(d-1)$-dimensional facet of $\widehat{\mathcal{E}}_i$ and $\widehat{\mathcal{E}}_j$, respectively.

Each element $\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h$ is parametrized over a reference element $\Omega_d \subset \mathbb{R}^d$ by an invertible and differentiable mapping:

$$\mu_{\widehat{\mathcal{E}}} : \Omega_d \to \widehat{\mathcal{E}}, \tag{6.3}$$

referred to as the affine mapping of $\widehat{\mathcal{E}}$.

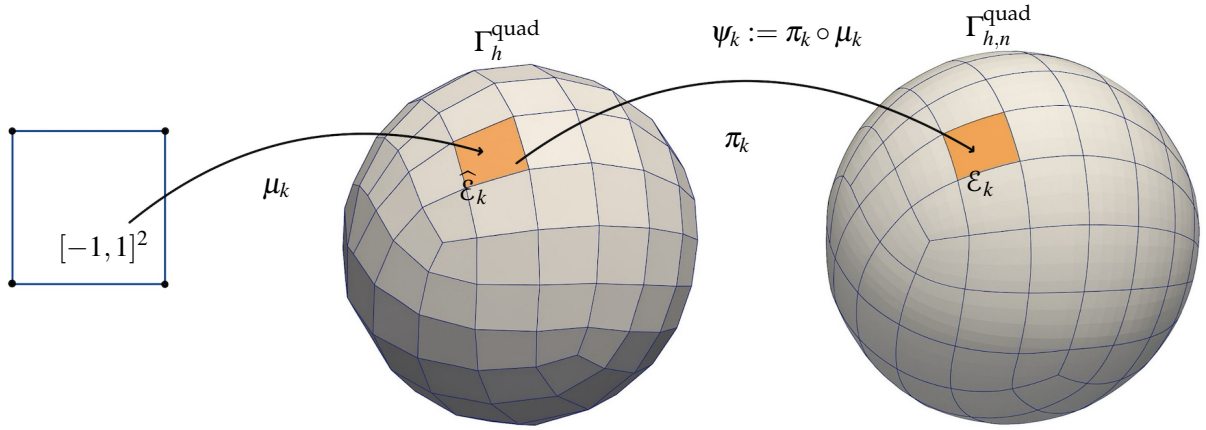


Figure 6.1: Construction of a surface parametrization over the reference square $[-1,1]^2$ via closest-point projection from the piecewise affine approximation Γ_h^{quad}.

Additionally, we assume the existence of a bijective mapping $\mathbf{X} : \Gamma_h^{\text{quad}} \to \Gamma$, such that the smooth surface Γ can be represented as a union of non-overlapping mapped elements:

$$\Gamma = \bigcup_{\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h} \mathbf{X}(\widehat{\mathcal{E}}) = \bigcup_{\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h} \mathbf{X}(\mu_{\widehat{\mathcal{E}}}(\Omega_d)) =: \bigcup_{\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h} \mathbf{X}_{\widehat{\mathcal{E}}}(\Omega_d). \tag{6.4}$$

With this property, Γ_h^{quad} is referred to as the reference surface (or reference domain) of Γ, and the collection $\{\mathbf{X}_{\widehat{\mathcal{E}}}\}_{\widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h}$ as its reference parametrization.

The Jacobian of the parametrization $\mathbf{X}_{\widehat{\mathcal{E}}}$ at $\mathbf{x}$ is

$$D\mathbf{X}_{\widehat{\mathcal{E}}}(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^{d+1}.$$

The first fundamental form is the symmetric and positive definite matrix $\mathbf{g} \in \mathbb{R}^{d \times d}$ defined as

$$\mathbf{g}(\mathbf{x}) := (D\mathbf{X}_{\widehat{\mathcal{E}}}(\mathbf{x}))^T D\mathbf{X}_{\widehat{\mathcal{E}}}(\mathbf{x}) \quad \forall \mathbf{x} \in \Omega_d.$$

The corresponding volume element $g(\mathbf{x})$ is given by

$$g(\mathbf{x}) = \sqrt{\det \mathbf{g}(\mathbf{x})}.$$

Let $Q_{G_{d,n}}\mathbf{X} \in \Pi_{d,n}$ be the nth-order Lagrange polynomial interpolation of the mapping $\mathbf{X}$ on the reference element Ω_d. In the following, we consider the restriction of the closest-point projection π to Γ_h^{quad} as a mapping $\pi : \Gamma_h^{\text{quad}} \to \mathbb{R}^{d+1}$, and define n^{th} -order cubic parameterization of the surface.

Definition 100 (n^{th}-order parametrization)**.** *Given an r-regular cubical parametrization* $\psi_k := \pi_k \circ \mu_k : \Omega_d \to \widehat{\mathcal{E}}_k, k = 1, \ldots, K, r \geq 2$, *where* π_k *is the closest point projection onto the target surface* Γ *and* μ_k

is a multilinear map (see Figure 6.1). We say that the mesh Γ_h^{quad} is of order n if each element is obtained by polynomial interpolation of the values $\{\psi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in A_{d,n}}$ sampled at the points $\{p_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in A_{d,n}}$.

This implies that on each element, we can approximate the r-regular parametrization maps through interpolation using the points $\{\psi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in A_{d,n}}$. This involves computing a n^{th}-order vector-valued polynomial approximation:

$$Q_{G_{d,n}}\psi_k(\mathrm{x}) = \sum_{\boldsymbol{\alpha}\in A_{d,n}} \psi_k(p_{\boldsymbol{\alpha}}) L_{\boldsymbol{\alpha}}, \quad k = 1,\dots,K. \tag{6.5}$$

Partial derivatives of the elemental coordinate maps, $\partial_{\xi_j} Q_{G_{d,n}}\psi_k$, for $k = 1,\dots,K$ and $j = 1,\dots,d$, are used to construct an n^{th}-order approximation of the volume element $g(\mathbf{x})$

$$g^n(\mathbf{x}) = \sqrt{\det\left(\mathbf{D}Q_{G_{d,n}}\psi_k(\mathbf{x})\right)^T \mathbf{D}Q_{G_{d,n}}\psi_k(\mathbf{x})\big)}. \tag{6.6}$$

Mapping the piecewise flat surface results in the higher-order approximation

$$\Gamma_{h,n}^{\text{quad}} := Q_{G_{d,n}}\psi_k(\Gamma_h^{\text{quad}})$$

of the surface Γ, which is contained in $\mathcal{N}_\delta$ if the mesh size is small enough. In the following, we assume that h is chosen correspondingly. Associated with the discrete surface $\Gamma_{h,n}^{\text{quad}}$ is a set of surface elements

$$\mathcal{K}_{h,n} := \{Q_{G_{d,n}}\psi_k(\widehat{\mathcal{E}}) \mid \widehat{\mathcal{E}} \in \widehat{\mathcal{K}}_h\},$$

such that $\Gamma_{h,n}^{\text{quad}} = \bigcup_{\mathcal{E}\in\mathcal{K}_{h,n}} \mathcal{E}$.

We now turn to estimating the geometric approximation error of $\Gamma_{h,n}^{\text{quad}}$.

Theorem 101. *Let Γ be a d-dimensional C^{r+1}-surface, $r > d+1$, and $\psi_k := \pi_k \circ \mu_k : \Omega_d \to \mathbb{R}^{d+1}$, $k = 1,\dots,K$ be a r-regular cubical parametrization, Definition 100. Let $Q_{G_{d,n}}\psi_k$ be the vector-valued tensor-polynomial interpolant of ψ_k in the Chebyshev–Lobatto grid* $\text{Cheb}_{d,n}$.

i) The r-regular cubical parametrization and its interpolant $Q_{G_{d,n}}\psi_k$ differ by

$$\|\psi_k(\mathbf{x}) - Q_{G_{d,n}}\psi_k(\mathbf{x})\|_{C^0(\Omega_d)} \le \frac{4ed^2 V_{\psi,r}}{\pi(r-d+1)} \left(\frac{n+1}{n+1-r}\right)^{r+1} \cdot \frac{1}{n^{r-d+1}}. \tag{6.7}$$

ii) The Jacobians of ψ_k and its interpolant $Q_{G_{d,n}}\psi_k$ differ by

$$\|\mathbf{D}\psi_k(\mathbf{x}) - \mathbf{D}Q_{G_{d,n}}\psi_k(\mathbf{x})\|_{C^0(\Omega_d)} \le \frac{4ed^2 V_{\psi,r}}{\pi(r-d-1)} \left(\frac{n+1}{n+1-r}\right)^{r+1} \cdot \frac{1}{n^{r-d-1}}, \tag{6.8}$$

where $V_{\psi,r}$ is the maximum Chebyshev-weighted r^{th}-order directional variation of the coordinate functions of ψ_k.

iii) The difference of the volume elements is bounded by

$$\|g(\mathbf{x}) - g^n(\mathbf{x})\|_{C^0(\Omega_d)} \leq dC^{d-1} \, \|\mathbf{D}\psi_k(\mathbf{x}) - \mathbf{D}\mathcal{Q}_{G_{d,n}}\psi_k(\mathbf{x})\|_{C^0(\Omega_d)},$$

where

$$g(\mathbf{x}) = \sqrt{\det\big(\mathbf{D}\psi_k(\mathbf{x})^T \, \mathbf{D}\psi_k(\mathbf{x})\big)},$$

and $g^n(\mathbf{x})$ is defined as in equation (6.6).

Proof. The argument is completely analogous to that of Lemma 92 (Chapter 5); we omit the details. □

As shown in equation (6.7), our parametrization map achieves a convergence rate of $\mathcal{O}(n^{-(r-d+1)})$, in contrast to the classical estimate [65]

$$\|\psi_k(\mathbf{x}) - \mathcal{Q}_{G_{d,n}}\psi_k(\mathbf{x})\|_{C^0(\Omega_d)} \leq Ch^{n+1},$$

which is based on interpolation at equidistant points.

6.2 High-order spectral collocation on a single surface element

We now describe a spectral collocation method for discretizing equation (6.1) on a single surface element $\mathcal{E}_k$. The method is formulated directly from the strong form in equation (6.1). Let $p_{\boldsymbol{\alpha}} \in \text{Cheb}_{d,n}$, $\boldsymbol{\alpha} \in A_{d,n}$ denote points in a tensor product Chebyshev-Lobatto grid. Let $\mathbf{u} \in \mathbb{R}^{(n+1)^d}$ denote a vector holding approximations to u at $p_{\boldsymbol{\alpha}} \in \text{Cheb}_{d,n}$. Analogously to equation (6.5), we have

$$\mathcal{Q}_{G_{d,n}}\mathbf{u}\,(\mathrm{x}) = \sum_{\boldsymbol{\alpha}\in A_{d,n}} \mathbf{u}(p_{\boldsymbol{\alpha}})L_{\boldsymbol{\alpha}}, \quad p_{\boldsymbol{\alpha}} \in \text{Cheb}_{d,n}. \tag{6.9}$$

To discretize $\mathcal{L}_\Gamma$ on the element $\mathcal{E}_k$, we compute discrete operators on the reference cube Ω_d and map them to $\mathcal{E}_k$ using the numerical coordinate mapping ψ_k for $k = 1, 2, \ldots, K$. Let $\mathcal{D} \in \mathbb{R}^{(n+1)\times(n+1)}$ be the one-dimensional spectral differentiation matrix [132] associated with Chebyshev-Lobatto points on the interval $[-1, 1]$ and let $I \in \mathbb{R}^{(n+1)\times(n+1)}$ be the identity matrix. The d-dimensional differentiation matrices on the reference hypercube Ω_d in the $\xi_1, \xi_2, \ldots, \xi_d$-directions can be constructed through the Kronecker products

$$\begin{aligned}
\mathcal{D}_{\xi_1} &= I \otimes \cdots \otimes I \otimes \mathcal{D} \\
\mathcal{D}_{\xi_2} &= I \otimes \cdots \otimes \mathcal{D} \otimes I \\
\mathcal{D}_{\xi_i} &= I \otimes \cdots \otimes \mathcal{D} \otimes \cdots \otimes I \\
&\qquad (\text{with } \mathcal{D} \text{ in the } (d-i+1) \text{ position}) \\
&\vdots \\
\mathcal{D}_{\xi_d} &= \mathcal{D} \otimes I \otimes \cdots \otimes I
\end{aligned}$$

where $\mathcal{D}_{\xi_i} \in \mathbb{R}^{(n+1)^d \times (n+1)^d}$, $i = 1, 2, \ldots, d$. Let $\mathbf{M}[\mathbf{u}] \in \mathbb{R}^{(n+1)^d \times (n+1)^d}$ denote the diagonal multiplication matrix formed by placing the entries of $\mathbf{u}$ along the diagonal. Using equation (4.6), differentiation matrices corresponding to the components of the surface gradient on $\mathcal{E}_k$ are given by[2]

$$\mathcal{D}^{\Gamma}_{x_j} = \sum_{i=1}^{d} \mathbf{M}[\partial_{x_j} \xi_i] \mathcal{D}_{\xi_i}, \; j = 1, 2, \ldots, d+1 \,. \tag{6.10}$$

The operator $\mathcal{L}_\Gamma$ given in equation (6.1) is then discretized on the element $\mathcal{E}_k$ as the $(n+1)^d \times (n+1)^d$ matrix

$$\mathbf{L}_{\mathcal{E}_k} = \sum_{i=1}^{d+1} \sum_{j=i}^{d+1} \mathbf{M}[a_{ij}] \mathcal{D}^{\Gamma}_i \mathcal{D}^{\Gamma}_j + \sum_{i=1}^{d+1} \mathbf{M}[b_i] \mathcal{D}^{\Gamma}_i + \mathbf{M}[c], \tag{6.11}$$

with all variable coefficients sampled on the grid points $p_{\boldsymbol{\alpha}} \in \mathrm{Cheb}_{d,n}$. In other words, we can think of the matrix $\mathbf{L}_{\mathcal{E}_k}$ as a discrete approximation to the differential operator $\mathcal{L}_{\mathcal{E}_k}$. Using this discretization scheme, equation (6.1) can be written as $(n+1)^d \times (n+1)^d$ linear system.

$$\mathbf{L}_{\mathcal{E}_k} \mathbf{u} = \mathbf{f} \,. \tag{6.12}$$

To prepare for the imposition of boundary conditions and local elimination, we partition the index set $\{1, \ldots, (n+1)^d\}$ for a given element $\mathcal{E}_k$ into interior (I_i) and boundary (I_b) subsets, so that

$$\{1, 2, \ldots, (n+1)^d\} = I_i \cup I_b,$$

with $(n-1)^d$ points in the interior and $2nd^{d-1}$ on the boundary. This partition is used to divide both vectors and matrices into blocks. For instance, given a matrix $\mathbf{A}$, the submatrix $\mathbf{A}^{i,b}$ consists of the rows indexed by I_i and columns indexed by I_b. The solution vector is written as

$$\mathbf{u}_i = \mathbf{u}(I_i) \quad \text{and} \quad \mathbf{u}_b = \mathbf{u}(I_b).$$

Partitioning the $\mathbf{L}_{\mathcal{E}_k}$ and reordering the degrees of freedom in (6.12) in the order $\{I_i, I_b\}$ (i.e. interior then boundary) gives a block linear system,

$$\begin{bmatrix} \mathbf{L}^{i,i}_{\mathcal{E}_k} & \mathbf{L}^{i,b}_{\mathcal{E}_k} \\ \mathbf{L}^{b,i}_{\mathcal{E}_k} & \mathbf{L}^{b,b}_{\mathcal{E}_k} \end{bmatrix} \begin{bmatrix} \mathbf{u}^i \\ \mathbf{u}^b \end{bmatrix} = \begin{bmatrix} \mathbf{f}^i \\ \mathbf{f}^b \end{bmatrix}. \tag{6.13}$$

To impose Dirichlet boundary conditions, we set $\mathbf{u} = \mathbf{h}^b$ on $\partial\mathcal{E}_k$, where $\mathbf{h}^b \in \mathbb{R}^{2dn^{d-1} \times 1}$ denotes the vector of boundary values of $h(x)$. Substituting these into the system and eliminating the boundary unknowns $\mathbf{u}_b$ via a Schur complement [95] yields the reduced system:

$$\mathbf{L}^{i,i}_{\mathcal{E}_k} \mathbf{u}^i = \mathbf{f}^i - \mathbf{L}^{i,b}_{\mathcal{E}_k} \mathbf{h}^b.$$

[2] For instance, the discrete Laplace–Beltrami operator may be discretized as $\Delta_\Gamma \approx \sum_{i=1}^{d+1} (\mathcal{D}^{\Gamma}_{x_i})^2$.

The interior solution is then expressed as

$$\mathbf{u}^i = \left(\mathbf{L}_{\mathcal{E}_k}^{i,i}\right)^{-1} \mathbf{f}^i - \mathbf{S}_{\mathcal{E}_k} \mathbf{h}^b,$$

where $\mathbf{S}_{\mathcal{E}_k} := -\left(\mathbf{L}_{\mathcal{E}_k}^{i,i}\right)^{-1} \mathbf{L}_{\mathcal{E}_k}^{i,b}$ is known as the solution operator and is of size $= (n+1)^d \times 2dn^{d-1}$.

6.3 Domain decomposition methods

The spectral collocation method described in the previous section converges very quickly as the number of points n increases—provided that the solution u of equation (6.1) is smooth. The matrix $\mathbf{L}_{\mathcal{E}_k}$ that arises from the discretization has some structure and contains many zeros, but it is still considerably denser than the matrices produced by finite difference or finite element methods. One way to reduce this density is to use domain decomposition methods, which also lend themselves well to implementation on parallel computing architectures.

In a domain decomposition approach, the computational domain Γ is partitioned into smaller elements $\mathcal{E}_k$, for $k = 1, \ldots, K$, which may touch or overlap. The original problem in equation (6.1) is then reformulated on each element, resulting in a family of smaller subproblems, where each subproblem can be solved independently using spectral collocation. However, to ensure that the local solutions $u_k(\mathbf{x})$, each defined solely on an element $\mathcal{E}_k$ for $k = 1, \ldots, K$, fit together and form a smooth solution of the PDE in equation (6.1) on the entire computational domain Γ, they have to satisfy matching conditions.

- For touching (non-overlapping) elements $\mathcal{E}_1$ and $\mathcal{E}_2$, the solution is required to be C^1-smooth across the shared interface; that is, both the function and its binormal derivative must be smooth on the surface where the elements touch:

$$u_1(\mathbf{x}) = u_2(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{E}_1 \cap \partial\mathcal{E}_2 \tag{6.14}$$

$$\partial_{\mathbf{n}_b} u_1(\mathbf{x}) = -\partial_{\mathbf{n}_b} u_2(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{E}_1 \cap \partial\mathcal{E}_2 \tag{6.15}$$

  where $\mathbf{n}_b$ denotes the binormal and the minus sign is because we use the outward-pointing binormal in each domain.

- For overlapping elements $\mathcal{E}_1$ and $\mathcal{E}_2$, the functions u_1 and u_2 must be equal on the intersection $\mathcal{E}_1 \cap \mathcal{E}_2$. By uniqueness of the PDE, it suffices to require that the functions are identical in any subregion of the overlapping domain:

$$u_1(\mathbf{x}) = u_2(\mathbf{x}), \quad \mathbf{x} \in \partial\left(\mathcal{E}_1 \cap \mathcal{E}_2\right)$$

The use of Chebyshev basis functions leads to shared boundary degrees of freedom between adjacent quadrilateral elements, so imposing (6.14) becomes straightforward.

Among the various techniques for enforcing continuity of binormal derivatives across patch interfaces, an effective strategy is the use of the Poincaré–Steklov operator [108], also known as the Dirichlet-to-Neumann (DtN) map—originally introduced by V.A. Steklov. This operator maps Dirichlet data (solution values) to Neumann data (binormal derivatives of the solution) on the boundary.

For a given element $\mathcal{E}_k$, the Dirichlet-to-Neumann operator, denoted by $\mathrm{DtN}_{\mathcal{E}_k}$, computes the outward fluxes corresponding to prescribed Dirichlet boundary data. Having constructed the solution operators $\mathbf{S}_{\mathcal{E}_k}$ that solve the PDE locally on each element $\mathcal{E}_k$, for $k = 1, \ldots, K$, the Dirichlet-to-Neumann operator is given by the product of the binormal derivative operator $\mathcal{D}_{\mathcal{E}_k}$ and the solution operator $\mathbf{S}_{\mathcal{E}_k}$:

$$\mathrm{DtN}_{\mathcal{E}_k} = \mathcal{D}_{\mathcal{E}_k} \mathbf{S}_{\mathcal{E}_k}.$$

From this point onward, we restrict our discussion to two-dimensional surfaces embedded in $\mathbb{R}^3$. Consequently, the discretization scheme is defined within each quadrilateral, spanning the space of polynomials $\Pi_{2,n}$.

6.3.1 Merging Dirichlet-to-Neumann maps

Assume now that Γ is divided into *touching elements*. We consider the simplest case in which we have only two elements $\mathcal{E}_1$ and $\mathcal{E}_2$ like in Figure 6.2, and denote by $\mathcal{I}_{12} = \mathcal{E}_1 \cap \mathcal{E}_2$ their common boundary.

$$\mathcal{L}_\Gamma u_1(\mathbf{x}) = f_1(\mathbf{x}), \quad \mathbf{x} \in \mathcal{E}_1, \quad \text{with} \quad u_1(\mathbf{x}) = h_1(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{E}_1 \setminus \mathcal{I}_{12}, \tag{6.16}$$

$$\mathcal{L}_\Gamma u_2(\mathbf{x}) = f_2(\mathbf{x}), \quad \mathbf{x} \in \mathcal{E}_2, \quad \text{with} \quad u_2(\mathbf{x}) = h_2(\mathbf{x}), \quad \mathbf{x} \in \partial\mathcal{E}_2 \setminus \mathcal{I}_{12}, \tag{6.17}$$

with continuity conditions:

$$\begin{aligned} u_1(\mathbf{x}) &= u_2(\mathbf{x}), \quad \mathbf{x} \in \mathcal{I}_{12}, \\ \partial_{\mathbf{n}_b} u_1(\mathbf{x}) &= -\partial_{\mathbf{n}_b} u_2(\mathbf{x}), \quad \mathbf{x} \in \mathcal{I}_{12}. \end{aligned}$$

The patching problem in equation (6.16) and (6.17) can be regarded as two decoupled, four-sided Dirichlet problems when given a suitable piece of Dirichlet data along $\mathcal{I}_{12}$. That is, there exists an interface function u_{glue} such that (6.16) and (6.17) is equivalent to

$$\begin{aligned} &\mathcal{L}_\Gamma u_1(\mathbf{x}) = f_1(\mathbf{x}), && \mathbf{x} \in \mathcal{E}_1, \\ &u_1(\mathbf{x}) = h_1(\mathbf{x}), && \mathbf{x} \in \partial\mathcal{E}_1 \setminus \mathcal{I}_{12}, \\ &u_1(\mathbf{x}) = u_{\text{glue}}(\mathbf{x}), && \mathbf{x} \in \mathcal{I}_{12}. \end{aligned} \tag{6.18}$$

$$\begin{aligned} &\mathcal{L}_\Gamma u_2(\mathbf{x}) = f_2(\mathbf{x}), && \mathbf{x} \in \mathcal{E}_2, \\ &u_2(\mathbf{x}) = h_2(\mathbf{x}), && \mathbf{x} \in \partial\mathcal{E}_2 \setminus \mathcal{I}_{12}, \\ &u_2(\mathbf{x}) = u_{\text{glue}}(\mathbf{x}), && \mathbf{x} \in \mathcal{I}_{12}. \end{aligned} \tag{6.19}$$

To determine the unknown interface function u_{glue}, we construct a solver referred to as the *interface solution operator*, denoted by $\mathbf{S}_{\text{glue}}$, such that $u_{\text{glue}}(\mathbf{x}) = \mathbf{S}_{\text{glue}} \begin{bmatrix} h_1(\mathbf{x}) \\ h_2(\mathbf{x}) \end{bmatrix}$. This solver is constructed using local operators from each element. Specifically, we first build local solvers for the elements $\mathcal{E}_1$ and $\mathcal{E}_2$, and then combine parts of these operators to create the interface solution operator $\mathbf{S}_{\text{glue}}$.

Once the interface function u_{glue} is determined, the two subproblems from equations (6.18) and (6.19) become independent and can be solved separately by applying spectral collocation on $\mathcal{E}_1$ and $\mathcal{E}_2$.

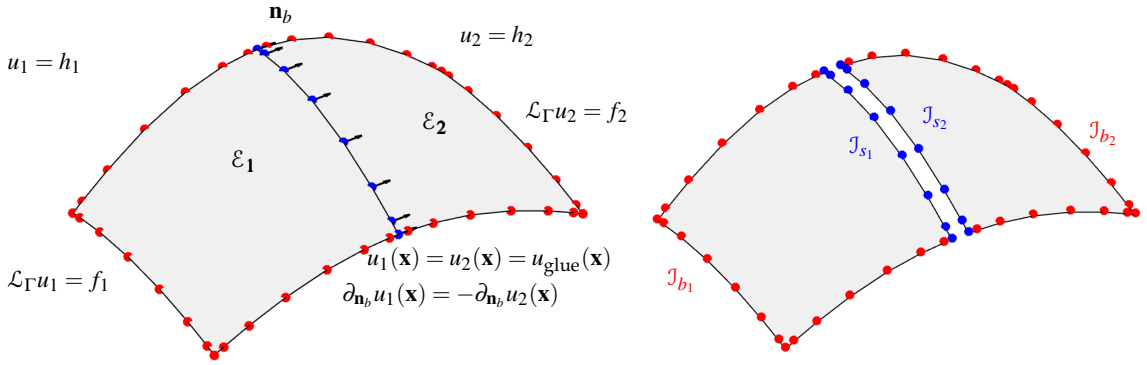


Figure 6.2: Interface coupling of two surface elements via continuity of the solution and its binormal derivative. Red points indicate boundary collocation points $\mathcal{I}_{b_1}$ and $\mathcal{I}_{b_2}$, while blue points represent interface collocation points $\mathcal{I}_{s_1}$ and $\mathcal{I}_{s_2}$, aligned for coupling. The right panel highlights the shared interface and point alignment used in the spectral collocation framework.

Let $\mathcal{I}_{s_1}, \mathcal{I}_{s_2} \subset \{1, \ldots, 2dn^{d-1}\}$ be index sets corresponding to the shared interface points (i.e., the points which are interior to the merged domain $\mathcal{E}_{\text{glue}} := \mathcal{E}_1 \cup \mathcal{E}_2$) with respect to elements $\mathcal{E}_1$ and $\mathcal{E}_2$, and $\mathcal{I}_1$ and $\mathcal{I}_2$ the remaining indices corresponding to the boundary points. See Figure 6.2 for a diagram. As the points of $\mathcal{E}_1$ and $\mathcal{E}_2$ are identical along the shared interface $\mathcal{I}_{12}$, continuity of the solution across $\mathcal{I}_{12}$ simply means that[3] $\mathbf{u}_1^{s_1} = \mathbf{u}_2^{s_2}$, so let us denote these solution values by $\mathbf{u}_{\text{glue}}$.

Continuity conditions across element interfaces are imposed locally using discrete Dirichlet-to-Neumann (DtN) operators.

$$\text{DtN}_{\mathcal{E}_k} = \mathcal{D}_{\mathcal{E}_k} \mathbf{S}_{\mathcal{E}_k}, \quad k = 1, 2.$$

Since $\text{DtN}_{\mathcal{E}_k}$ maps from the quadrilateral boundary to itself, it is a square matrix of size $2dn^{d-1} \times 2dn^{d-1}$. We define the boundary data vectors $\mathbf{h}_1^b, \mathbf{h}_2^b \in \mathbb{R}^{2dn^{d-1}}$ for the two neighboring elements as

$$\mathbf{h}_1^b(\mathbf{x}) := \begin{bmatrix} \mathbf{h}_1(\mathbf{x}) \\ \mathbf{u}_{\text{glue}}(\mathbf{x}) \end{bmatrix}, \quad \mathbf{h}_2^b(\mathbf{x}) := \begin{bmatrix} \mathbf{h}_2(\mathbf{x}) \\ \mathbf{u}_{\text{glue}}(\mathbf{x}) \end{bmatrix},$$

where $\mathbf{h}_1(\mathbf{x})$ and $\mathbf{h}_2(\mathbf{x})$ correspond to the non-shared boundary degrees of freedom, and $\mathbf{u}_{\text{glue}}(\mathbf{x})$

[3] The superscripts s_1, s_2 denote restriction to interface indices

represents the shared interface values between the two elements.

The corresponding homogeneous solutions on each element are then given by

$$\mathbf{w}_k = \mathbf{S}_{\mathcal{E}_k}\mathbf{h}_k^b, \quad k = 1, 2.$$

The continuity condition for the binormal derivative across the shared interface requires that

$$\left(\mathrm{DtN}_{\mathcal{E}_1}\mathbf{h}_1^b + \mathcal{D}_{\mathcal{E}_1}\boldsymbol{\nu}_1\right)^{s_1} + \left(\mathrm{DtN}_{\mathcal{E}_2}\mathbf{h}_2^b + \mathcal{D}_{\mathcal{E}_2}\boldsymbol{\nu}_2\right)^{s_2} = 0, \tag{6.20}$$

where $\boldsymbol{\nu}_k' := \mathcal{D}_{\mathcal{E}_k}\boldsymbol{\nu}_k$ denotes particular fluxes associated with particular solutions on each element.

Rewriting the system (6.20) in terms of the interface unknown $\mathbf{u}_{\text{glue}}$, we obtain separate linear systems for the homogeneous and particular interface unknowns:

$$\left(\mathrm{DtN}_{\mathcal{E}_1}^{s_1 s_1} + \mathrm{DtN}_{\mathcal{E}_2}^{s_2 s_2}\right)\mathbf{w}_{\text{glue}} = -\left(\mathrm{DtN}_{\mathcal{E}_1}^{s_1 b_1}\mathbf{h}_1 + \mathrm{DtN}_{\mathcal{E}_2}^{s_2 b_2}\mathbf{h}_2\right), \tag{6.21}$$

$$\left(\mathrm{DtN}_{\mathcal{E}_1}^{s_1 s_1} + \mathrm{DtN}_{\mathcal{E}_2}^{s_2 s_2}\right)\boldsymbol{\nu}_{\text{glue}} = -\left(\boldsymbol{\nu}_1'^{s_1} + \boldsymbol{\nu}_2'^{s_2}\right). \tag{6.22}$$

Here, the local operators are constructed on each element. By solving equation (6.21), we obtain the interface solution operator $\mathbf{S}_{\text{glue}}$, which is defined as follows:

$$\mathbf{S}_{\text{glue}} := -\left(\mathrm{DtN}_{\mathcal{E}_1}^{s_1 s_1} + \mathrm{DtN}_{\mathcal{E}_2}^{s_2 s_2}\right)^{-1}\begin{bmatrix}\mathrm{DtN}_{\mathcal{E}_1}^{s_1 b_1} & \mathrm{DtN}_{\mathcal{E}_2}^{s_2 b_2}\end{bmatrix}. \tag{6.23}$$

The Schur complement also allows us to write down the Dirichlet-to-Neumann operator for the merged domain. Using the new interface solution operator $\mathbf{S}_{\text{glue}}$, we construct the new Dirichlet-to-Neumann operator

$$\mathrm{DtN}_{\text{glue}} := \begin{bmatrix}\mathrm{DtN}_{\mathcal{E}_1}^{b_1 b_1} & 0 \\ 0 & \mathrm{DtN}_{\mathcal{E}_2}^{b_2 b_2}\end{bmatrix} + \begin{bmatrix}\mathrm{DtN}_{\mathcal{E}_1}^{b_1 s_1} \\ \mathrm{DtN}_{\mathcal{E}_2}^{b_2 s_2}\end{bmatrix}\mathbf{S}_{\text{glue}}. \tag{6.24}$$

In a similar manner, the specific flux for the merged domain is given by

$$\boldsymbol{\nu}_{\text{glue}}' = \begin{bmatrix}\boldsymbol{\nu}_1'^{b_1} \\ \boldsymbol{\nu}_2'^{b_2}\end{bmatrix} + \begin{bmatrix}\mathrm{DtN}_{\mathcal{E}_1}^{b_1 s_1} \\ \mathrm{DtN}_{\mathcal{E}_2}^{b_2 s_2}\end{bmatrix}\boldsymbol{\nu}_{\text{glue}}. \tag{6.25}$$

6.3.2 The hierarchical scheme

The term *hierarchical* refers to the organization of the computational mesh into a hierarchical binary tree structure [92]. Each element of the mesh, before any merging or gluing occurs, corresponds to a leaf box in this tree. At each stage of the merging process, two neighboring leaves are combined to form a larger domain. The result of merging two such elements is a pair of operators defined on the merged domain: $\mathcal{E}_{\text{glue}}$: (1) a solution operator, $\mathbf{S}_{\text{glue}}$, that solves for the

unknown interface values $\mathbf{u}_{\text{glue}}$ inside $\mathcal{E}_{\text{glue}}$; and (2) a Dirichlet-to-Neumann (DtN) operator, DtN_{glue}, that maps boundary data to outward fluxes on $\mathcal{E}_{\text{glue}}$. These operators encapsulate all necessary information to solve the PDE on $\mathcal{E}_{\text{glue}}$. After merging, the new parent element $\mathcal{E}_{\text{glue}}$ is functionally equivalent to the original elements $\mathcal{E}_1$ or $\mathcal{E}_2$, and thus can be treated as a single element, ready to be merged again with additional domains. This hierarchical merging procedure,

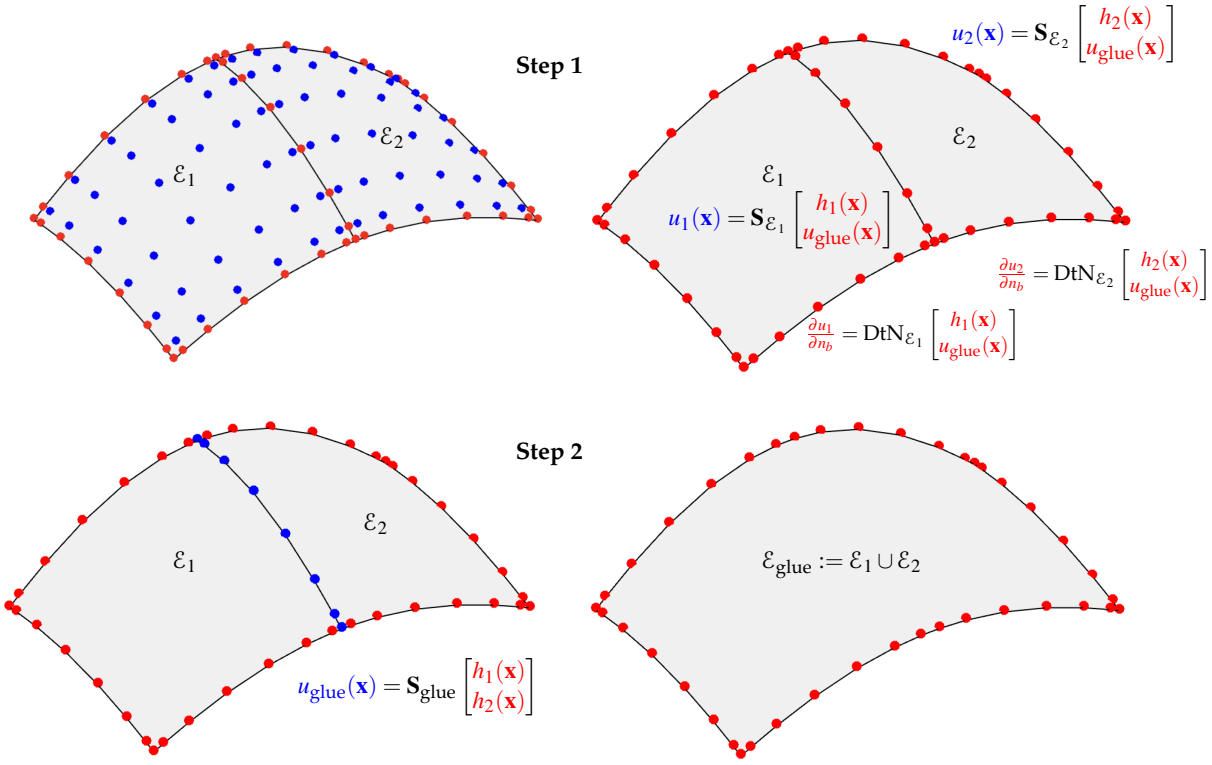


Figure 6.3: An overview of the hierarchical merge process. **Step 1:** For each patch $\mathcal{E}_i$, remove the internal points (blue) and retain only the boundary points (red). Use the remaining data to construct the solution operator $\mathbf{S}_{\mathcal{E}_i}$ and the Dirichlet-to-Neumann operator $\text{DtN}_{\mathcal{E}_i}$, $i = 1, 2$. **Step 2:** Merge patches in pairs as described in Subsection 6.3.1. Form the equilibrium equation using $\text{DtN}_{\mathcal{E}_1}$ and $\text{DtN}_{\mathcal{E}_2}$ and eliminate the interior points of the new larger patches. Construct the new solution operator $\mathbf{S}_{\text{glue}}$ and corresponding DtN operator.

enabled by local operators, constitutes the hierarchical Poincaré–Steklov scheme. An illustrated overview of the merge process is described in Figure 6.3.

6.4 Accommodating triangulated surfaces

The quadrilateral-based domain decomposition scheme described thus far is built on the assumption that the smooth surface $\Gamma \subset \mathbb{R}^{d+1}$ is discretized into a high-order quadrilateral mesh Γ_h^{quad}, constructed from a set of r-regular quadrilateral parametrizations.

$$\left\{ \psi_k : \Omega_d \to \widehat{\mathcal{E}}_k \right\}_{k=1}^{K},$$

where each patch $\widehat{\mathcal{E}}_k$ is sampled using mapped points $\{\psi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in A_{d,n}}$, where $\{p_{\boldsymbol{\alpha}}\} \subset \mathrm{Cheb}_{d,n}$ is the standard tensor-product Chebyshev-Lobatto grid on Ω_d.

In many practical applications, the surface Γ is given as a triangulated mesh $\bigcup_{T\in\mathcal{T}_h} T$. Simplicial domains—such as triangles and tetrahedra, which are not tensor-product domains—offer far greater flexibility than Cartesian products of intervals (e.g., squares or cubes) when it comes to handling complex geometries through partitioning methods [34]. Spectral methods, on the other hand, are remarkably efficient on tensor-product domains because of the inherent structure of the expansions they employ. Therefore, it is tempting to try to marry the efficiency of tensor products with the flexibility of triangular geometries. The procedure described above remains unchanged, provided that suitable high-order points and basis functions are constructed on the reference triangle, allowing the PDE to be discretized with high-order accuracy on each element, similarly to the quadrilateral case. We propose two distinct strategies for handling triangles within the described HPS framework. The first approach is the *hypercube-to-simplex reparametrization* (see Definition 85). The second method is rhombus-based remeshing, a quadrilateralization technique that converts triangles into quadrilaterals while minimizing the number of elements introduced by the tripling scheme of Fortunato et al. [51]. Both approaches are presented in detail in the following subsections.

6.4.1 Hypercube-to-simplex reparametrization

In Section 5.5, we demonstrated that by employing mappings such as square-squeezing, key numerical operations—interpolation, quadrature, and differentiation—on triangular elements can be carried out on a reference square. This approach enables triangular elements to use the same tensor-product Chebyshev machinery as quadrilateral elements, thereby making them suitable for high-order spectral collocation. The additional issue for HPS is that the local square representation must also induce a boundary trace that can be merged with neighboring elements.

Building on this foundation, let us now consider a triangulation of the surface Γ, denoted by

$$\Gamma_h^{\mathrm{tri}} = \bigcup_{\widehat{T}\in\widehat{\mathcal{T}}_h} \widehat{T}, \quad |\widehat{\mathcal{T}}_h| = K,$$

which is not necessarily a high-order approximation of Γ. In the following, we adopt the notations used in Section 5.5, and define n^{th}-order cubical re-parameterization of the surface for each $\widehat{T} \in \widehat{\mathcal{T}}_h$

$$\varphi_k : \Omega_d \to \widehat{T}, \quad \varphi_k = \pi_k \circ \tau_k \circ \sigma, \quad k = 1,\dots,K. \tag{6.26}$$

Definition 102 (n^{th}-order cubical reparametrization)**.** *Given an r-regular cubical reparametrization $\varphi_k = \pi_k \circ \tau_k \circ \sigma : \Omega_d \to \widehat{T}_k, k = 1,\dots,K, r \geq 2$, where π_k is the closest point projection onto the target surface Γ, τ_k is an affine map and σ is the square squeezing map. We say that the mesh Γ_h^{tri} is of order n if each element is obtained by polynomial interpolation of the values $\{\varphi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in A_{d,n}}$ sampled at $\{p_{\boldsymbol{\alpha}}\}_{\boldsymbol{\alpha}\in A_{d,n}}$.*

For every triangle $\widehat{T} \in \widehat{\mathcal{T}}_h$, we compute $\{\varphi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,n}}$ and define an n-th order d-dimensional triangle T by applying polynomial interpolation of order n to the coordinates of the projected points $\{\varphi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha}\in G_{d,n}}$.

Mapping the piecewise flat surface results in a higher-order approximation $\Gamma_{h,n}^{\text{tri}} := Q_{G_{d,n}}\varphi_k(\Gamma_h^{\text{tri}})$ of the surface Γ that is contained in $\mathcal{N}_\delta$ if h is small enough. Associated with the discrete surface $\Gamma_{h,n}^{\text{tri}}$ is a set of surface elements

$$\mathcal{T}_{h,n} := \{Q_{G_{d,n}}\varphi_k(\widehat{T}) \mid \widehat{T} \in \mathcal{T}_h\},$$

such that

$$\Gamma_{h,n}^{\text{tri}} = \bigcup_{T\in\mathcal{T}_{h,n}} T\,.$$

Given $\Gamma_{h,n}^{\text{tri}}$, we can compute an n^{th}-order approximation of the volume element $g(\mathbf{x})$,

$$g^n(\mathbf{x}) = \sqrt{\det\left(\mathbf{D}Q_{G_{d,n}}\varphi_k(\mathbf{x})^T\mathbf{D}Q_{G_{d,n}}\varphi_k(\mathbf{x})\right)}.$$

Remark 103. *The geometric approximation error estimates for $\Gamma_{h,n}^{\text{tri}}$ in Lemma 92 (Chapter 5) remain valid in this context.*

Having defined the mapped Chebyshev interpolation in Subsection 3.7.1, we now derive n^{th} mapped cubical re-parameterization of the surface Γ. Let ζ be an analytic function, such that:

$$\zeta : \Omega_d \to \Omega_d,\ \mathbf{x} = \zeta(\mathbf{s}) \tag{6.27}$$

with $\partial_{s_i}\zeta(\mathbf{s}) > 0,\ 1 \le i \le d$. We define

$$\tilde{\varphi}_k = \varphi_k \circ \zeta = \pi_k \circ \tau_k \circ \sigma \circ \zeta : \Omega_d \to \widehat{T}_k, \quad k = 1,\dots,K, \quad r \ge 2. \tag{6.28}$$

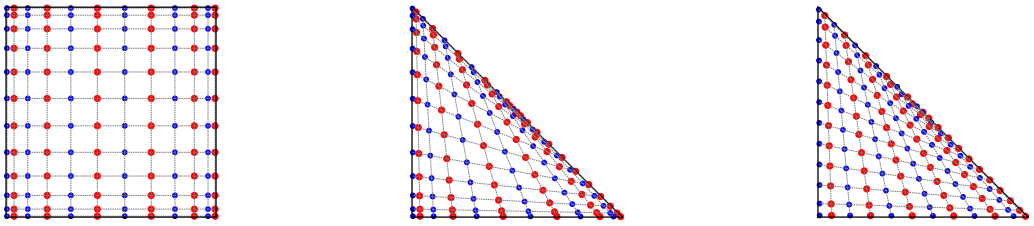
the r-regular *mapped* cubical re-parameterization of the surface Γ. One benefit of the mapped square-squeezing is that it leads to a more uniform distribution of Chebyshev-Lobatto points, with fewer points concentrated near the boundary of the square, resulting in a lesser number of points on the hypotenuse of simplex $\triangle_2$ (see Figure 6.4).

Spectral collocation on a triangular patch via square-squeezing reparametrization

To assess the impact of the triangle-to-quadrilateral reparametrization (6.26), both in its standard and mapped forms using the analytic function $\zeta(\mathbf{s})$ from (6.28), we consider the Laplace–Beltrami equation:

$$-\Delta_\Gamma u = f\,. \tag{6.29}$$

We test two geometric configurations: one-eighth of the sphere (triangular patch), and one-sixth of the sphere (quadrilateral patch). The exact solution is the spherical harmonic $u(\mathbf{x}) =$



(a) Standard square (b) Square-squeezing (c) Mapped Square-squeezing

Figure 6.4: Distribution of Chebyshev–Lobatto points for $n = 11$ on the reference square (left), after the square-squeezing map to the triangle (center), and after the mapped square-squeezing transformation (right).

$Y_\ell^m(\mathbf{x})$ with $(\ell, m) = (3, 2)$, yielding the right-hand side $f(\mathbf{x}) = \ell(\ell+1)Y_\ell^m(\mathbf{x})$. Dirichlet boundary conditions are given by

$$h(x, y, z) = 0.25\sqrt{105/\pi}\,(x^2 - y^2)z.$$

For the analytic map, we select the polynomial mapping defined in equation (3.43) with degree $n = 5$:

$$\zeta(\mathbf{s}) = \frac{1}{149}(120\mathbf{s} + 20\mathbf{s}^3 + 9\mathbf{s}^5). \tag{6.30}$$

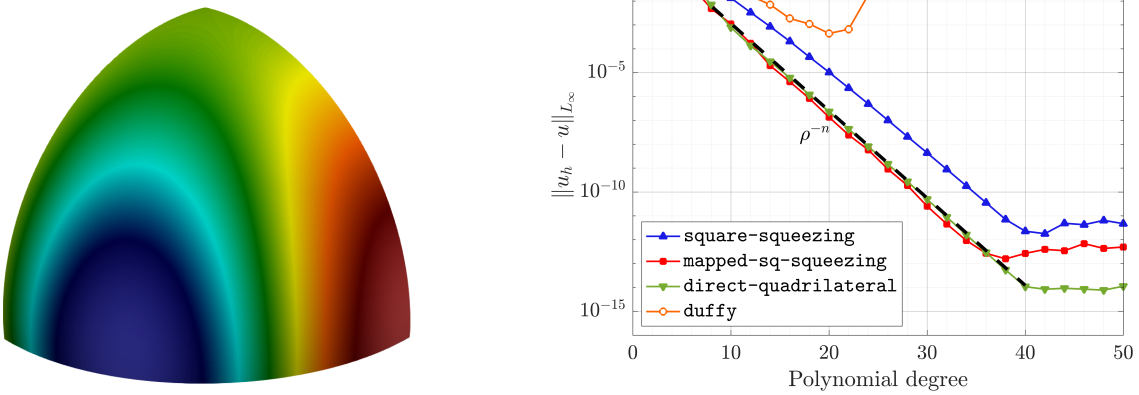


Figure 6.5: Relative L^∞-error for the Laplace–Beltrami problem in equation (6.29) on an eighth-sphere triangular patch, using the exact spherical harmonic Y_3^2. The triangle is reparametrized through square-squeezing, mapped square-squeezing, and the Duffy transform; a direct quadrilateral patch is included as a reference. The observed exponential decay is compared with the fit rate ρ^{-n}, where $\rho \approx 2.3$.

In the numerical experiments presented, the error values shown in the convergence plots represent the *discretization error* with respect to the exact solution of the Laplace–Beltrami equation. Figure 6.5 illustrates the spatial convergence to the exact solution under p-refinement, considering different reparametrization strategies:

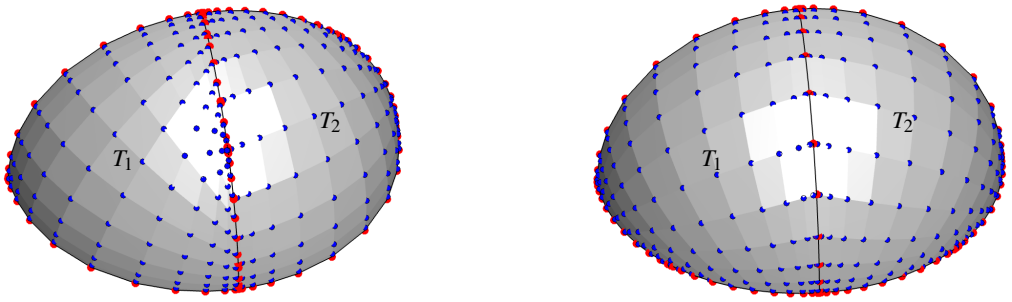


Figure 6.6: Trace-point distribution induced by square squeezing for two neighboring triangular elements at degree $n = 13$. The hypotenuse carries twice as many points as the other edges; the configuration shown aligns this doubled edge with the outer boundary to avoid an inconsistent interior merge.

1. `direct-quad`: refers to the discretization on a single quadrilateral patch (see Definition 100),
2. `square-squeezing`: uses a hypercube-to-simplex reparametrization based on the square-squeezing transform (see Definition 102),
3. `mapped-sq-squeezing`: applies the square-squeezing transformation enriched by an analytic map $\zeta(\mathbf{s})$ (see equation (6.30)),
4. `duffy`: uses a reparametrization from the unit square to the triangle based on the Duffy map (see Remark 88).

In this experiment, the square-squeezing mapping achieves accuracy comparable to the quadrilateral parametrization, whereas the Duffy mapping performs poorly due to its degeneracy along the collapsed edge. This degeneracy adversely affects the computation of binormals, which—as previously emphasized—are crucial to ensuring the success of the merging process.

By constructing the transformed Chebyshev interpolant of the n^{th}-order cubical reparametrization of the geometry via the analytic map (6.30), the mapped square-squeezing approach (represented by the red line in the plot) reaches machine precision more rapidly in this test. Nonetheless, quadrilateral patches tend to achieve higher accuracy at elevated polynomial degrees ($\geq$ 38), a regime that is not commonly used in practice. These observations suggest that square squeezing remains a promising strategy for extending high-order solvers to triangulated surfaces.

Next, we extend the previous example to the upper hemisphere of the sphere. For this purpose, we developed the Python package `pysurfacefun`[4], which is currently under active development and is designed to support the surface reparametrization tools required in our framework. One

[4] https://pysurfacefun.readthedocs.io/en/latest/

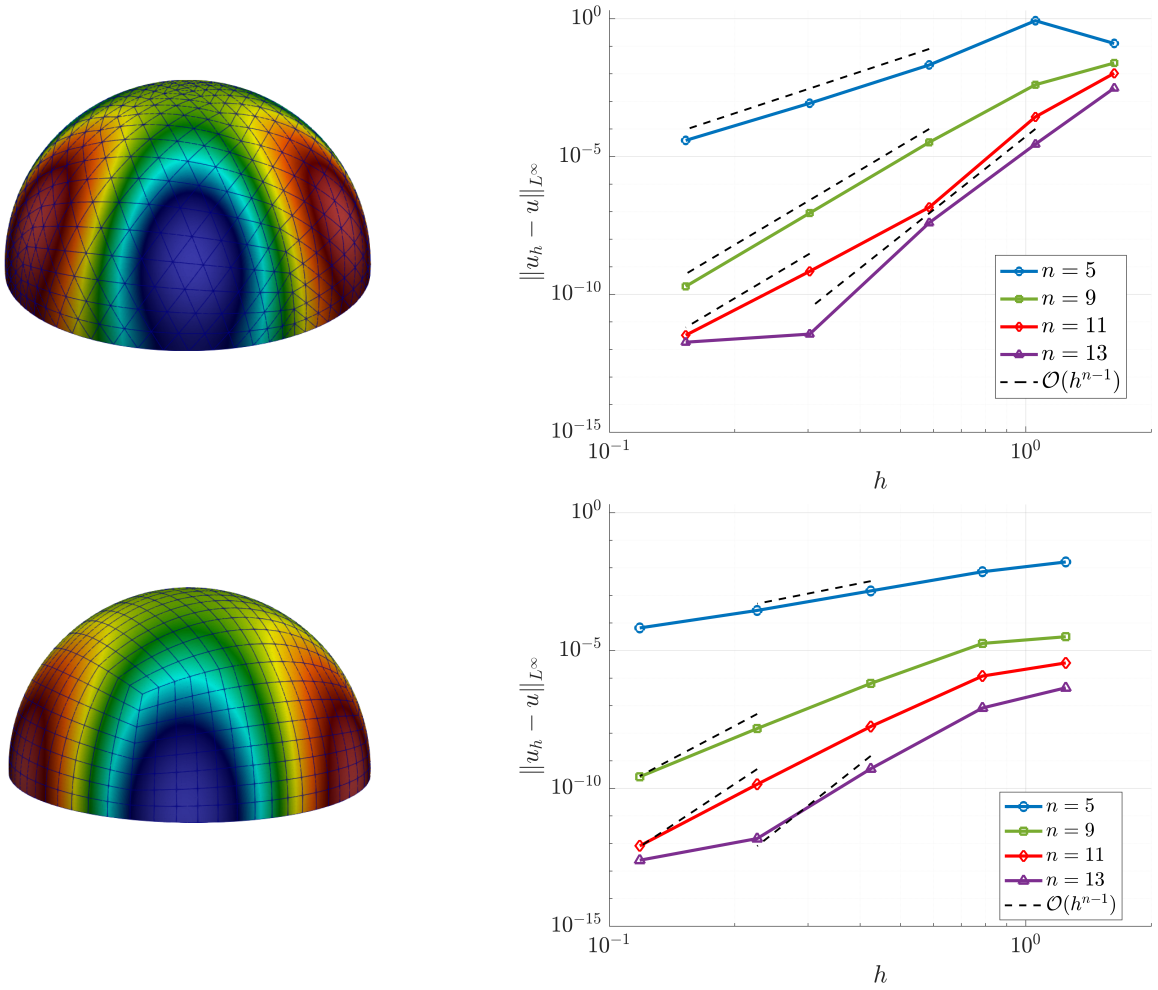


Figure 6.7: Relative L^∞-error for the Laplace–Beltrami equation on the upper hemisphere under mesh refinement. The top row uses high-order triangular patches and the bottom row uses quadrilateral patch domains. Curves are shown for polynomial degrees $n = 5, 9, 11, 13$, with reference slopes consistent with $\mathcal{O}(h^{n-1})$.

challenge introduced by this reparametrization is the non-uniform distribution of points, as illustrated in Figure 6.4b. In particular, the number of points along the hypotenuse of $\triangle_2$ is twice that of the other edges. As a result, tabulation points along shared boundaries of physical elements T may not coincide, creating challenges for patch merging. This issue can be addressed by interpolating the values defined on the side with $2n$ points to match those on the side with only n points. The interpolation is stable and introduces only minimal discretization errors, though it does complicate the implementation. For further details, see [7]. In our example, we avoid inconsistencies by properly configuring the elements and aligning the side with $2n$ points along the equatorial boundary, as illustrated in Figure 6.6.

In Figure 6.7, we report the convergence of the L^∞ relative error for the numerical solution of the Laplace–Beltrami equation on the upper hemisphere, using high-order patch-based discretizations with *triangular elements* (top) and *square elements* (bottom). Convergence is shown for polynomial degrees $n = 5, 9, 11, 13$, alongside reference slopes of $\mathcal{O}(h^{n-1})$. The surface plots (left)

depict the computed solutions over each mesh.

In both cases, plotting the relative L^∞-error against the mesh resolution h reveals an algebraic convergence rate. The error decreases rapidly as the mesh is refined ($h \to 0$) and the polynomial degree n increases, following the expected rate of $\mathcal{O}(h^{n-1})$. Notably, for $n = 13$, both mesh types achieve low errors, reflecting the high accuracy and stability of the method. These results confirm the efficacy of the high-order patch-based discretization approach for solving PDEs on curved surfaces, with both mesh types providing highly accurate solutions under h− and p− refinement.

However, the current algorithm still relies on configuring the mesh elements beforehand, which becomes challenging as the mesh density increases. This limitation is not caused by the local square-squeezed discretization itself, but by the interface representation required by the HPS merge. Locally, the reparametrization allows interpolation and differentiation to be performed on the reference square, where tensor-product Chebyshev tools are available. Globally, however, neighboring triangular elements must exchange boundary data along physical edges, and the square-squeezed trace does not automatically provide a uniform three-edge triangular skeleton.

For this reason, the hypercube-to-simplex construction should be viewed here as a promising local discretization strategy rather than as a complete practical HPS method for arbitrary unstructured triangulations. A natural next step is to decouple the square-squeezed volume discretization from the interface discretization: one may compute the local solution and fluxes on the tensor-product square grid, but expose to the HPS merge a triangle-compatible trace space with three physical edges. The transfer between these two representations could be carried out by interpolation, projection, or a mortar-type coupling operator. Such a non-matching trace formulation would remove the need for manual orientation of elements and would make the square-squeezing approach applicable to general triangulated meshes.

6.4.2 Quadrilateralization

To this end, we define a *quadrilateralization operator*

$$\mathcal{Q} : \Gamma_h^{\text{tri}} \longrightarrow \Gamma_h^{\text{quad}},$$

which replaces the original triangulated mesh with a conforming set of quadrilateral patches $\{\widehat{\mathcal{E}}_k\}_{k=1}^K$, each admitting a smooth mapping $\psi_k : \Omega_d \to \widehat{\mathcal{E}}_k$, and a point set $\{\psi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha} \in A_{d,n}}$ for use in high-order interpolation.

We review two established strategies for assembling the interface operator $\mathcal{Q}$ before presenting our own rhombus–based remeshing:

(1) Blossom-Based Matching.

Let $\{\widehat{T}_k\}_{k=1}^K$ denote the set of triangles in Γ_h^{tri}. A dual graph is constructed in which each vertex represents a triangle and edges connect pairs of adjacent triangles. Using a maximum matching algorithm, such as the Blossom algorithm [45], the method identifies as many disjoint pairs of adjacent triangles as possible. Each matched pair $(\widehat{T}_i, \widehat{T}_j)$ is then merged to form a quadrilateral patch $\widehat{\mathcal{E}}_k = \widehat{T}_i \cup \widehat{T}_j$. The resulting set of quadrilaterals defines a partial quadrilateralization of the triangulated mesh. Unmatched triangles are then converted into quadrilateral elements through a centroid-based subdivision process. However, to maintain a conforming mesh, this approach necessitates splitting the already grouped quadrilaterals as well. As a consequence, the refinement yields a denser mesh with a higher number of degrees of freedom, as illustrated in Figure 6.8b.

(2) Centroid-Based Subdivision.

The approach consists of splitting each triangle into three quadrilaterals, following the method described in [51]. Specifically, for each triangle $\widehat{T} \in \widehat{\mathcal{T}}_h$ with vertices $\{v_1, v_2, v_3\}$, we introduce the centroid

$$c_T = \tfrac{1}{3}(v_1 + v_2 + v_3)$$

and the edge midpoints

$$m_{ij} = \tfrac{1}{2}(v_i + v_j), \quad 1 \le i < j \le 3.$$

We then form three quadrilaterals

$$\widehat{\mathcal{E}}_i = \text{Quad}\big(v_i,\, m_{ij},\, c_T,\, m_{ik}\big),$$

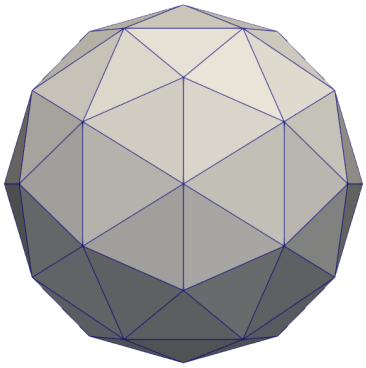
for $\{i, j, k\} = \{1, 2, 3\}$. The result is a quadrilateral mesh Γ_h^{quad} with $3K$ elements—three times the original triangulation. Although this subdivision is regular and systematic, it inflates the total degrees of freedom and produces smaller quads clustered around each centroid (see Figure 6.8c).

(3) Rhombus-Based Remeshing (Proposed).

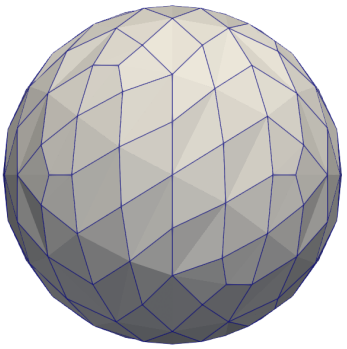
Let $\{\widehat{T}_k\}_{k=1}^N$ denote the set of triangles in Γ_h^{tri}. For any pair of adjacent triangles $\widehat{T}_i, \widehat{T}_j \in \widehat{\mathcal{T}}_h$ sharing an edge e_{ij}, let c_i and c_j denote the centroids of the triangles, and let $\{v_a, v_b\} \in e_{ij}$ be the vertices of the shared edge. A quadrilateral patch is then formed as

$$\widehat{\mathcal{E}}_k = \text{Quad}(v_a, c_j, v_b, c_i).$$

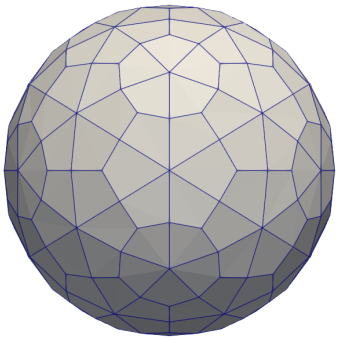
The elements $\widehat{\mathcal{E}}_k$ (see Figure 6.8d) can be naturally parameterized using cubical maps $\psi_k : \Omega_d \to \widehat{\mathcal{E}}_k$, and a corresponding set of mapped points $\{\psi_k(p_{\boldsymbol{\alpha}})\}_{\boldsymbol{\alpha} \in A_{d,n}} \subset \widehat{\mathcal{E}}_k \cap \Gamma$. Then, the resulting mesh Γ_h^{quad} can be interpolated to obtain an n^{th}-order cubical parametrization in the sense of Definition 100, thereby enabling direct application of the HPS scheme introduced in the preceding sections. Among the quadrilateralization strategies considered, our proposed approach—forming



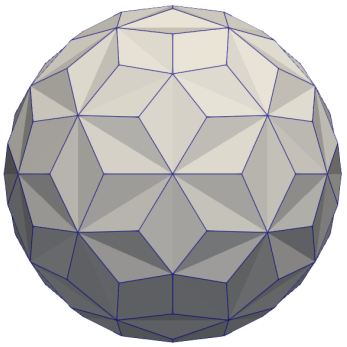
(a) Reference mesh Γ_h^{tri} (80 triangles)



(b) First method: 172 quads



(c) Second method: 240 quads



(d) Third method: 120 quads

Figure 6.8: Comparison of three quadrilateral meshing strategies applied to a coarse triangular surface mesh (top-left): Blossom-based matching (top-right), centroid-based subdivision (bottom-left), and the proposed rhombus-based remeshing (bottom-right).

rhombuses from the triangular mesh—yields the smallest total number of quadrilateral elements.

6.4.3 Computational asymptotic complexity

For an order-n discretization, the total number of degrees of freedom associated with the mesh $\{\mathcal{E}_k\}_{k=1}^K$ scales as $N \sim K(n+1)^2$. The cost has three distinct components. Constructing the local solution and Dirichlet-to-Neumann operators on the leaf elements costs $\mathcal{O}(K(n+1)^6)$, or equivalently $\mathcal{O}(N(n+1)^4)$. For fixed polynomial order, the hierarchical merge stage has the nested-dissection scaling $\mathcal{O}(N^{3/2})$ described in [50, 91]; if n is varied, the polynomial-degree factors must be retained separately. Once the hierarchy has been built, the downward solve for a new right-hand side costs $\mathcal{O}(N \log K + N(n+1))$. The storage requirement is dominated by the dense interface operators retained across the tree and scales like the fixed-order solve storage, up to the larger constants typical of direct solvers.

Comparison of HPS and surface FEM on the sphere

In this experiment, we compare the performance of the hierarchical Poincaré–Steklov (HPS) method with a higher-order surface finite element method (SFEM) [44]. We consider the Laplace–

Beltrami equation on the unit sphere:

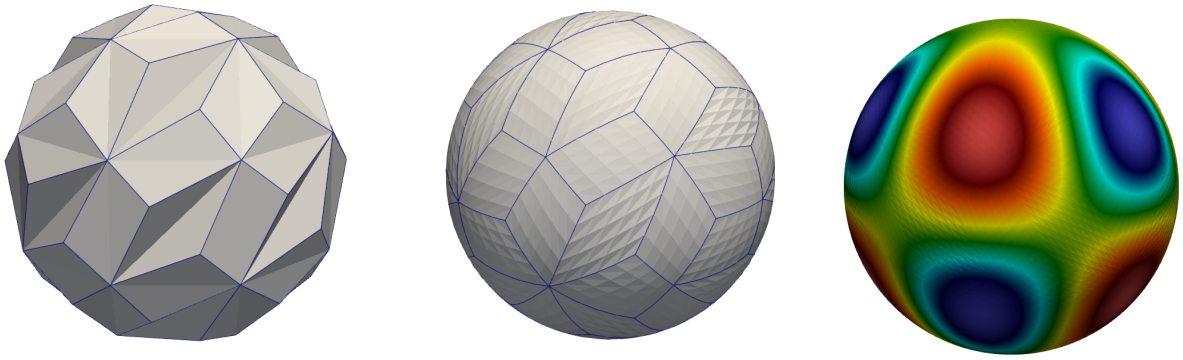
$$-\Delta_\Gamma u + u = f, \quad \text{on } \Gamma = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x}\|_2 = 1\}, \tag{6.31}$$

where the exact solution is given by the spherical harmonic $u(\mathbf{x}) = Y_\ell^m(\mathbf{x})$ with $(\ell, m) = (4, 3)$. The corresponding right-hand side is defined as

$$f(\mathbf{x}) = (\ell(\ell+1)+1)\, Y_\ell^m(\mathbf{x}),$$

so that u satisfies equation (6.31) exactly.

For the discretization, we parametrize the sphere Γ using surface finite elements constructed from a coarse reference mesh. The initial triangular mesh is transformed into quadrilateral elements with rhombus-like shapes, and the vertices are projected onto the sphere using the closest-point map $\pi(\mathbf{x}) := \frac{\mathbf{x}}{\|\mathbf{x}\|}$. The resulting high-order geometry is then obtained by interpolating π piecewise over the flat quadrilaterals. The numerical implementation is carried out using the DUNE framework [9, 113], with DUNE-ALUGRID [3] for managing the reference mesh and DUNE-CURVEDGRID [106] to define higher-order geometry mappings. The resulting linear systems are solved using the direct solver UMFPACK [37]. In the performance and accuracy plots below, DOF denotes the total number of global degrees of freedom, timings are reported for the assembly and solve stages, and errors are measured against the exact spherical harmonic solution.

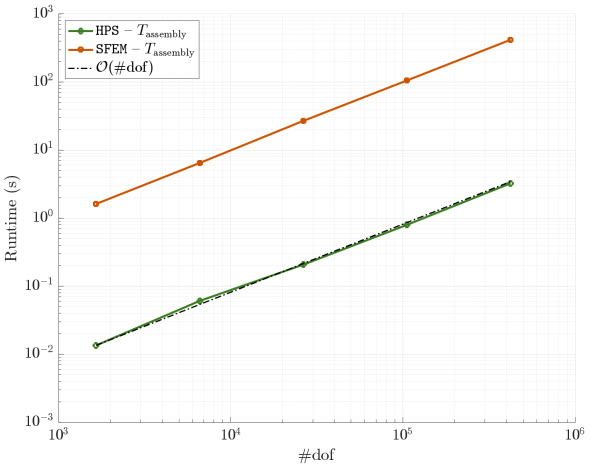


(a) Reference mesh Γ_h^{quad} (b) Curved geometry $\Gamma_{h,n}^{\text{quad}}$ (c) Computed solution

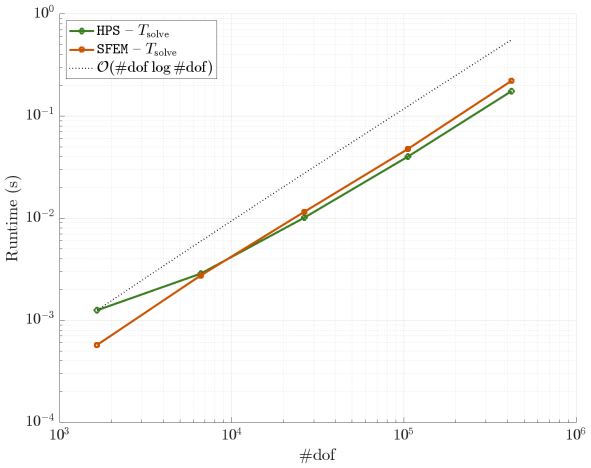
Figure 6.9: Geometry and solution for the sphere model problem in equation (6.31). Left: quadrilateral reference mesh Γ_h^{quad}. Center: curved high-order geometry $\Gamma_{h,n}^{\text{quad}}$ represented with a Lagrange parametrization of order $n = 5$ in SFEM. Right: computed spherical harmonic solution Y_4^3.

To distinguish between the two approaches evaluated, we label our method as `HPS`, while the reference method, which uses classical surface finite elements on curved geometries, is denoted as `SFEM`.

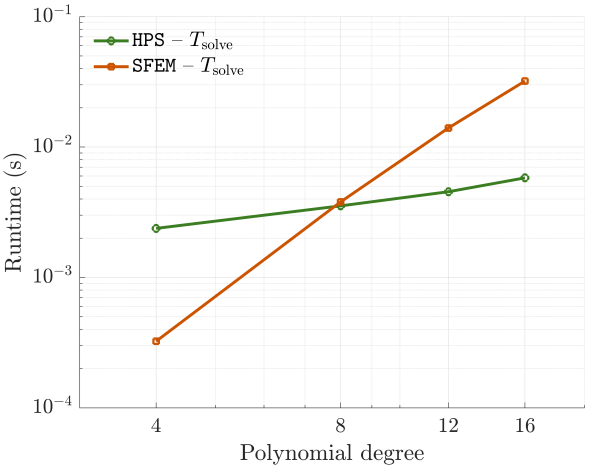
Both methods are evaluated on the same sequence of h-refined meshes using a fixed polynomial



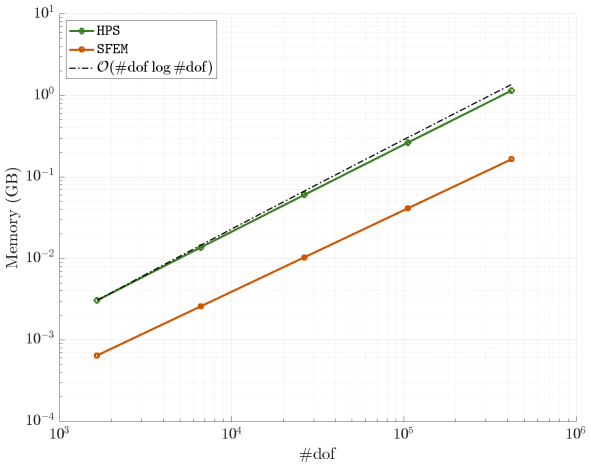
(a) Assembly time vs. DOF.



(b) Solve time vs. DOF.



(c) Solve time vs. polynomial degree.



(d) Memory usage vs. DOF.

Figure 6.10: Performance comparison between the HPS and SFEM discretizations for the sphere model problem in equation (6.31). Assembly time, solve time, and memory usage are plotted against the total number of degrees of freedom. The bottom-right panel fixes the mesh and varies the polynomial degree to compare high-order solve costs.

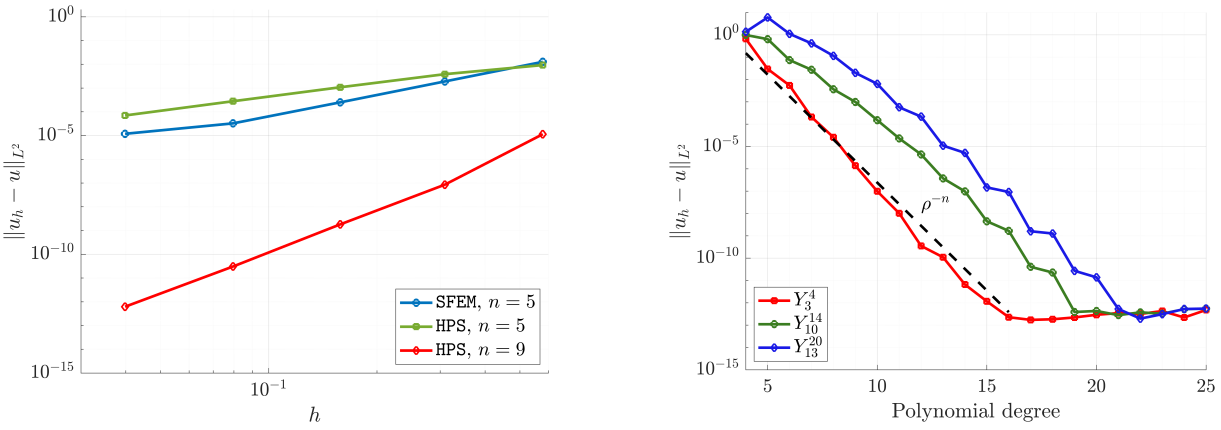


Figure 6.11: (Left) L^2-error versus mesh size h for SFEM and HPS applied to the exact solution Y_4^3, using degrees $n = 5$ and $n = 9$.
(Right) L^2-error versus polynomial degree for three spherical-harmonic test functions, compared with an exponential fit rate ρ^{-n} where $\rho \approx 9.3$.

degree $n = 5$, which represents the highest practically usable order for SFEM in this setting. As shown in Figure 6.10, the HPS method outperforms SFEM in both the assembly and solve phases. For fixed polynomial order, the observed solve time is consistent with the expected near-linear downward-pass cost after the HPS hierarchy has been constructed.

However, this speed advantage comes with increased memory usage. As with most direct solvers, the scheme requires more memory compared to iterative methods. As illustrated in Figure 6.10, HPS consumes significantly more memory than SFEM, a result of storing all required operators in the HPS approach. Both methods show memory growth consistent with $\mathcal{O}(N \log N)$, but with larger prefactors in HPS. We also evaluate performance with increasing polynomial order on a fixed mesh; the mesh is fixed and the polynomial degree is varied. As shown, SFEM remains competitive at low degrees but degrades rapidly at higher orders, despite both methods exhibiting similar algebraic complexity (e.g., when using UMFPACK).

In terms of accuracy, Figure 6.11(left) reports the L^2-error against a reference solution Y_ℓ^m with $(\ell, m) = (4, 3)$. While both methods perform similarly for low-order discretizations ($n = 5$), we observe that HPS achieves significantly higher accuracy by increasing the polynomial degree from $n = 5$ to $n = 9$ under mesh refinement, effectively performing an hp-refinement. The right plot displays the decay of the L^2-error as a function of polynomial degree, using three different spherical harmonics Y_ℓ^m as exact solutions. The results confirm exponential convergence consistent with spectral accuracy. The best-exponential fit rate ρ^{-n}, with $\rho \approx 9.3$, is included to indicate the expected decay rate. These results highlight HPS as a highly efficient solver for high-accuracy simulations, provided sufficient memory is available and the solution is sufficiently smooth.

6.5 Time-dependent equations

While the HPS scheme is designed to solve "stationary" problems modeled by *linear* elliptic PDEs, it can also be useful for accelerating time-dependent problems modeled by parabolic PDEs. We start by considering the reaction–diffusion systems, which are widely regarded as key mechanisms for pattern formation in a variety of contexts, including biological, chemical, physical, and even economic processes. A general reaction–diffusion system describing N interacting species defined on a closed, smooth surface $\Gamma \subset \Omega \subset \mathbb{R}^{d+1}$ can be written in the form:

$$\frac{\partial \boldsymbol{u}}{\partial t} = \nabla_\Gamma \cdot (\mathbf{D} \nabla_\Gamma \boldsymbol{u}) + \boldsymbol{F}(\boldsymbol{u}), \tag{6.32}$$

where $\boldsymbol{u} := (u_1, u_2, \ldots, u_N)$, $\boldsymbol{F}$ represents the reaction kinetics, also referred to as the source term, and $\mathbf{D}$ denotes the diffusion tensor. A fast direct solver for elliptic equations is useful here because each time step in (6.32) requires solving an elliptic problem. On a fixed surface, the geometry, diffusion coefficients, and time step are fixed, so the implicit matrix introduced below is the same at every time step. The HPS factorization can therefore be built once and reused for the sequence of right-hand sides generated by the explicit reaction term.

In the discretization process, we handle the spatial discretization, as described in the previous sections, using a domain decomposition approach with spectral collocation on each element. This yields a system of ordinary differential equations (ODEs) in time, given by:

$$\frac{d\boldsymbol{u}}{dt} = L_\Gamma \boldsymbol{u} + \boldsymbol{F}(\boldsymbol{u}), \tag{6.33}$$

The term $L_\Gamma \boldsymbol{u}$ arises from the diffusion components, while $\boldsymbol{F}(\boldsymbol{u})$ originates from the reaction components. Because the diffusive term is typically stiff [124], the use of explicit schemes usually necessitates excessively small time steps. This can result in computations which are prohibitively expensive in three spatial dimensions. Fully implicit treatment, however, requires the implicit treatment of the nonlinear reaction term, $\boldsymbol{F}(\boldsymbol{u})$, at every time step. This can be particularly expensive and undesirable because the Jacobian of $\boldsymbol{F}(\boldsymbol{u})$ could be dense and is typically non-definite, or non-symmetric which makes fast iterative solution techniques [136] less efficient and more difficult to implement. Moreover, explicitly handling the nonlinear reaction term $\boldsymbol{F}(\boldsymbol{u})$ is easy to implement and adds relatively little computational effort per time step. Additionally, many well-known time-stepping methods applied to (6.33) are either first-order (such as backward Euler) or result in only a weak reduction of high-frequency error components (such as Crank-Nicolson). In this work, we employ the Implicit-Explicit Backward Differentiation Formula (IMEX-BDF) family of schemes [5], combining backward differentiation for implicit terms with Adams-Bashforth for explicit terms.

Let $\Delta t > 0$ denote the time step, and $\boldsymbol{u}^n(\mathbf{x}) \approx \boldsymbol{u}(\mathbf{x}, n\Delta t)$ represent the approximate solution at step n. Time discretization with the M^{th} order IMEX-BDF scheme yields a steady-state problem at each step:

$$(I - \omega \Delta t L_\Gamma)\, \boldsymbol{u}^{n+1} = \sum_{i=0}^{M-1} a_i \boldsymbol{u}^{n-i} + \Delta t \sum_{i=0}^{M-1} b_i \boldsymbol{F}(\boldsymbol{u}^{n-i}), \tag{6.34}$$

which can be written more compactly as

$$A\boldsymbol{u}^{n+1} = \boldsymbol{f}^n,$$

where

$$A := I - \omega \Delta t L_\Gamma, \quad \boldsymbol{f}^n := \sum_{i=0}^{M-1} a_i \boldsymbol{u}^{n-i} + \Delta t \sum_{i=0}^{M-1} b_i \boldsymbol{F}(\boldsymbol{u}^{n-i}).$$

For the static-surface problems considered in this section, the matrix A is independent of n. The expensive stage is therefore the construction of the HPS solver for A; after this build stage, each time step requires only applying the stored solver to the new right-hand side $\boldsymbol{f}^n$. This is the sense in which the setup cost is amortized over many time steps. If the geometry, diffusion tensor, or time step changes, A changes as well and the factorization can no longer be reused without modification. For instance, using the IMEX-BDF1 scheme, where

$$\omega = 1, \quad a_0 = 1, \quad b_0 = 1,$$

yields the equation

$$(I - \Delta t L_\Gamma)\, \boldsymbol{u}^{n+1} = \boldsymbol{u}^n + \Delta t\, \boldsymbol{F}(\boldsymbol{u}^n), \tag{6.35}$$

which must be solved once per time step to compute $\boldsymbol{u}^{n+1}$ from $\boldsymbol{u}^n$, yielding a per-step solve cost of $\mathcal{O}(N \log K + N(n+1))$ after the hierarchy has been constructed.

The values of ω, a_i, and b_i for IMEX-BDF schemes of order 2 to 4 are given as follows:

$$\begin{aligned}
&\text{IMEX-BDF2:} && \omega = \tfrac{2}{3}, && a = \left(\tfrac{4}{3}, -\tfrac{1}{3}\right), && b = \left(\tfrac{4}{3}, -\tfrac{2}{3}\right), \\
&\text{IMEX-BDF3:} && \omega = \tfrac{6}{11}, && a = \left(\tfrac{18}{11}, -\tfrac{9}{11}, \tfrac{2}{11}\right), && b = \left(\tfrac{18}{11}, -\tfrac{18}{11}, \tfrac{6}{11}\right), \\
&\text{IMEX-BDF4:} && \omega = \tfrac{12}{25}, && a = \left(\tfrac{48}{25}, -\tfrac{36}{25}, \tfrac{16}{25}, -\tfrac{3}{25}\right), && b = \left(\tfrac{48}{25}, -\tfrac{72}{25}, \tfrac{48}{25}, -\tfrac{12}{25}\right).
\end{aligned}$$

The following theorem summarizes the accuracy of IMEX–BDF schemes; see [5, Theorem 2.1].

Theorem 104 (Order of IMEX–BDF schemes)**.** *An s-step IMEX-BDF scheme, as defined in* (6.34)*, has order at most s.*

In the numerical experiments below, the time step is chosen sufficiently small so that the temporal discretization error does not dominate the spatial discretization error. The reported convergence behavior therefore primarily reflects the high-order spatial discretization and the HPS solver, rather than the order of the time integrator. A systematic study of adaptive time-step selection is left outside the scope of this work.

We conclude this section by evaluating the accuracy and convergence of the HPS scheme for a time-dependent problem. As a benchmark with a known analytic solution, we consider isotropic

diffusion on the surface of the unit sphere,

$$\frac{\partial u}{\partial t} = \Delta_\Gamma u \quad \text{on } \Gamma = \{\mathbf{x} \in \mathbb{R}^3 \mid \|\mathbf{x}\|_2 = 1\}. \tag{6.36}$$

The exact solution in spherical coordinates (ϑ, φ), where $\vartheta \in [0, \pi]$ is the polar angle and $\varphi \in [0, 2\pi)$ is the azimuthal angle, is obtained via expansion in spherical harmonics Y_ℓ^m:

$$u(t, \vartheta, \varphi) = \sum_{\ell=0}^{\infty} \sum_{m=-\ell}^{\ell} c_{\ell m}(0) Y_\ell^m(\vartheta, \varphi) e^{-\ell(\ell+1)t}, \tag{6.37}$$

with modal coefficients determined by the projection of the initial data:

$$c_{\ell m}(0) = \int_\Gamma (-1)^m Y_\ell^{-m}(\vartheta, \varphi)\, u(0, \vartheta, \varphi)\, dS. \tag{6.38}$$

To avoid effects due to spectral truncation and to isolate the numerical error, we consider the initial condition

$$u(0, \vartheta, \varphi) = Y_1^0(\vartheta, \varphi) = \sqrt{\frac{3}{4\pi}} \cos \vartheta, \tag{6.39}$$

which leads to a closed-form analytic solution due to the orthogonality of the spherical harmonics:

$$u(t, \vartheta, \varphi) = Y_1^0(\vartheta, \varphi)\, e^{-2t}. \tag{6.40}$$

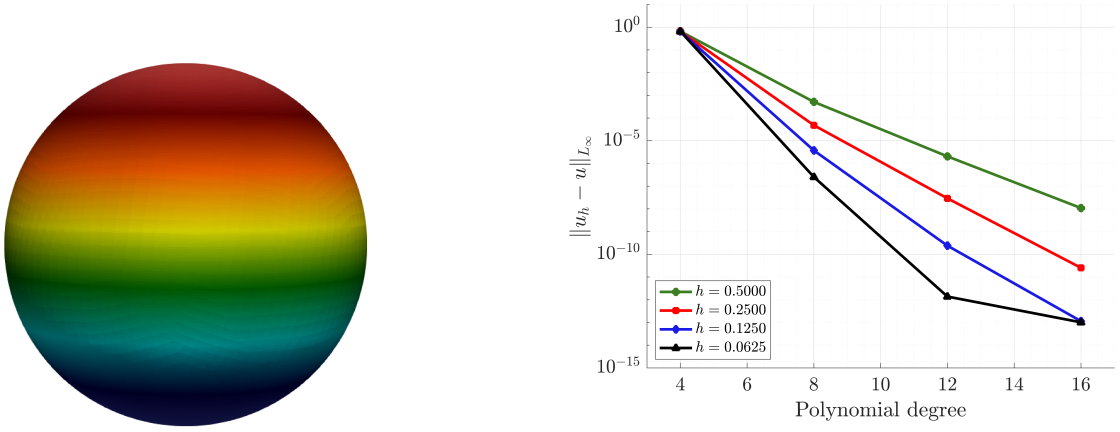


Figure 6.12: (Left) Computed solution of the pure diffusion problem on the unit sphere at $t = 1$, using initial data Y_1^0 and time step $\Delta t = 10^{-3}$. (Right) L_∞-error versus polynomial degree for different mesh sizes h, measured against the exact solution $Y_1^0 e^{-2t}$. The simulations use the implicit–explicit backward differentiation formula (IMEX–BDF4) scheme.

The left panel of Figure 6.12 shows the computed solution at the final time $t = 1$, while the right panel illustrates the L_∞-error as a function of polynomial degree for different mesh sizes. As expected, increasing the polynomial degree results in exponential convergence of the error, especially for finer meshes.

6.6 Numerical study of spatial pattern formation in Turing systems

In Section 6.5, we examined the pure surface diffusion equation as a benchmark problem to assess the accuracy and convergence of the proposed HPS scheme. Having established the method's robustness for this simpler case, we now turn to a more complex and biologically motivated setting: the Turing model of spatial pattern formation, first proposed by Turing [135]. He showed that a system of two reacting and diffusing chemicals could give rise to spatial patterns in chemical concentrations from initial near-homogeneity. A comprehensive overview of commonly used reaction kinetics and their underlying motivation can be found in [89]. In this section, we consider the Turing model on a surface Γ involving an activator–inhibitor system [8]. Our aim is to investigate how the domain geometry, nonlinearities, and coupling of such systems influence the emergence of spatial patterns. The specific form of the system considered is:

$$\frac{\partial u_1}{\partial t} = \delta_{u_1}\Delta_\Gamma u_1 + \alpha u_1\left(1 - r_1 u_2^2\right) + u_2\left(1 - r_2 u_1\right) \qquad \text{with } u_1(\mathbf{x},0) = u_1^0(\mathbf{x}), \tag{6.41a}$$

$$\frac{\partial u_2}{\partial t} = \delta_{u_2}\Delta_\Gamma u_2 + \beta u_2\left(1 + \frac{\alpha r_1}{\beta} u_1 u_2\right) + u_1\left(\gamma + r_2 u_2\right) \quad \text{with } u_2(\mathbf{x},0) = u_2^0(\mathbf{x}), \tag{6.41b}$$

where α, β, γ, r_1, r_2, δ_{u_1}, and δ_{u_2} are the parameters of the reaction–diffusion system. In the context of equation (6.32), we have the following:

$$\mathbf{D} = \begin{pmatrix} \delta_{u_1} & 0 \\ 0 & \delta_{u_2} \end{pmatrix}, \quad \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix}, \quad \mathbf{F} = \begin{pmatrix} \alpha u_1\left(1 - r_1 u_2^2\right) + u_2\left(1 - r_2 u_1\right) \\ \beta u_2\left(1 + \frac{\alpha r_1}{\beta} u_1 u_2\right) + u_1\left(\gamma + r_2 u_2\right) \end{pmatrix}.$$

In this system, u_1 and u_2 are morphogens with u_1 as the "activator" and u_2 as the "inhibitor". If $\alpha = -\gamma$, then $(u_1, u_2) = (0,0)$ is a unique equilibrium point of this system. The reaction term contributes to the formation of concentration peaks of u_1 and u_2, whereas the diffusion term tends to smooth these peaks. The interplay between these opposing processes—reaction-driven peak formation and diffusion-induced peak smoothing—leads to the emergence of characteristic Turing patterns.

To understand when such patterns arise, we consider the conditions under which the reaction–diffusion system (6.41) supports them. Specifically, model (6.41) exhibits Turing pattern formation when the following two conditions, known as the Turing criteria, are satisfied:

(i) In the absence of diffusion, the system tends toward a spatially uniform, linearly stable steady state.

(ii) The steady state becomes unstable in the presence of diffusion due to the introduction of random perturbations.

It is well known that patterns produced by the Turing model are influenced by domain geometry [26], and the nonlinear reaction term $\mathbf{F}(\mathbf{u})$, containing quadratic and cubic interactions, plays a key role in driving pattern development. To examine how geometry and nonlinearities

affect pattern emergence, we simulate the reaction–diffusion system (6.41) on curved surfaces using the high-order hierarchical Poincaré–Steklov (HPS) method.

The resulting stable patterns exhibit either spot-like or stripe-like structures, depending on the values of the coupling parameters r_1 and r_2. The cubic coupling parameter r_1 promotes the emergence of stripes, whereas the quadratic coupling parameter r_2 tends to favor spot formation [8]. Our numerical simulations indicate that, in general, spot patterns are more robust than stripes and reach a steady state significantly faster. Stripe patterns only emerge for very small values of r_2, and their orientation varies depending on the initial conditions. In Figure 6.13, we illustrate examples of these basic cases. For the simulations, we adopt the parameter values [5] from [26]: $\alpha = 0.899$, $\beta = -0.91, \gamma = -\alpha$, and $\delta_{u_1} = 0.516\,\delta_{u_2}$, with $\delta_{u_2} = 5 \cdot 10^{-3}$. The initial conditions are taken to be random: $u_1(\mathbf{x}, 0) = \text{rand}(\mathbf{x})$, $u_2(\mathbf{x}, 0) = \text{rand}(\mathbf{x})$.

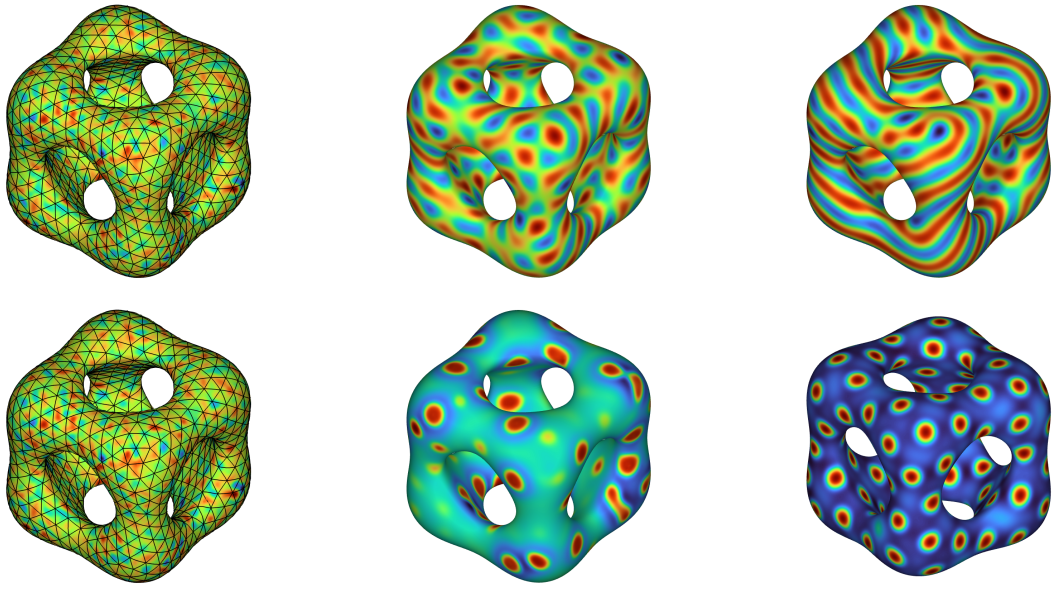


Figure 6.13: Activator concentration u_1 for the Turing model on a Swiss cheese surface. The top row uses $r_1 = 3.5$, $r_2 = 0$ and shows $t = 0, 200, 600$; the bottom row uses $r_1 = 0.02$, $r_2 = 0.2$ and shows $t = 0, 20, 200$. Both simulations use IMEX–BDF1 time stepping.

In Figure 6.14, we illustrate the competition between stripe and spot patterns when both nonlinear parameters are non-zero with $r_1 = 4.5$, $r_2 = 0.2$. The results indicate that, in scenarios where spots compete with stripes, the spotted patterns are remarkably robust and tend to emerge consistently whenever the coefficient of the quadratic term is non-zero.

Next, we simulate the Turing model on a variety of surfaces to investigate the influence of geometry on pattern formation. One example is an asymmetric torus, defined implicitly by

$$f(\mathbf{x}) = \left(x^2 + y^2 + z^2 - d^2 + b^2\right)^2 - 4\left(ax + c^2 d\right)^2 - 4b^2 y^2,$$

[5] Parameters follow the non-dimensionalized Turing model in [26], where all coefficients are dimensionless. The diffusion ratio ensures scale separation needed for pattern formation.

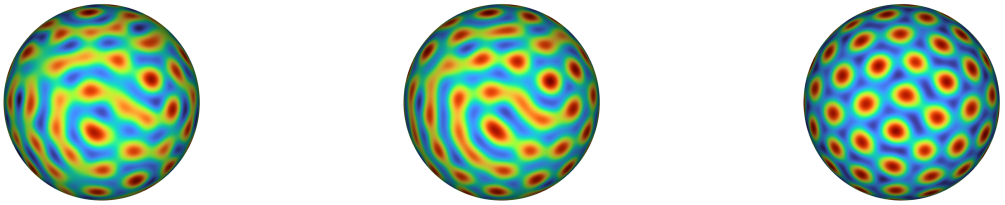


Figure 6.14: Activator concentration u_1 for the reaction–diffusion model on a sphere of radius 2, with parameters $r_1 = 4.5$ and $r_2 = 0.2$. The snapshots correspond to times $t = 100$, $t = 350$, and $t = 3000$ and are computed using the IMEX–BDF1 scheme.

with parameters $a = 2$, $b = 1.9$, $d = 1$ and $c^2 = a^2 - b^2$. Using the same system parameters, we also apply the model to a triangulated Stanford Bunny. All simulations are carried out using the IMEX-BDF1 time-stepping scheme with reaction parameters $r_1 = 0.02$, $r_2 = 0.2$. We evolve the system for 2000 time steps with a fixed time step size of $\Delta t = 0.1$, corresponding to a final time of $t = 200$.

As shown in Figure 6.15, each row represents a different surface geometry (asymmetric torus, Stanford Bunny), while columns display the solution u at times $t = 0$, $t = 20$, and $t = 200$. For both meshes, we employ 8^{th}-order elements to capture the solution. The emergence and arrangement of spots differ noticeably across geometries, emphasizing that surface shape, in addition to the system parameters, plays a critical role in the development of Turing patterns. Unlike in one- or two-dimensional scenarios, where geometry is often simplified or ignored, solving reaction–diffusion equations on various surfaces with fixed parameters reveals distinct pattern transitions.

In Figure 6.16, we show simulations on spherical meshes with radii $r = 1, 2, 4$, using 10^{th}-order elements and random initial data. The parameters are chosen to promote stripe formation, with $r_1 = 1.5$, $r_2 = 0$, $\alpha = 1.899$, $\gamma = -\alpha$, and $\beta = -0.95$, and diffusion coefficients set as $\delta_{u_1} = 0.516\,\delta_{u_2}$, where $\delta_{u_2} = 5 \cdot 10^{-3}$. Simulations are run up to $t = 600$ with a time step of $\Delta t = 0.05$. The results indicate that, as the domain expands, the number of high-activator stripes increases consistently, confirming theoretical and numerical predictions for two-dimensional growing domains [76].

We close this section by investigating interacting Turing systems. Specifically, we consider a second Turing system in chemicals (u_1, u_2) that modulates the kinetic terms in the (v_1, v_2) system

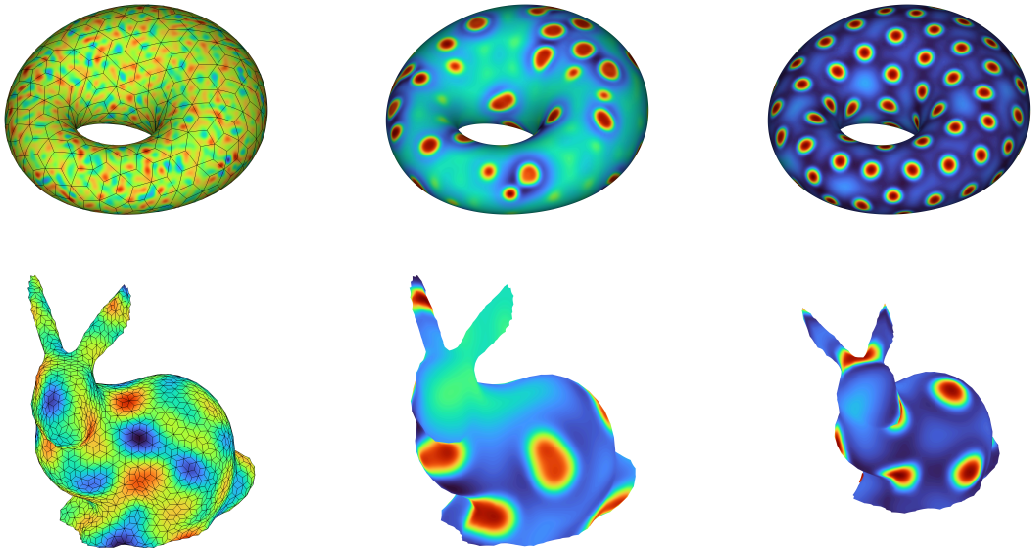


Figure 6.15: Activator concentration u_1 for the Turing model on two geometries: asymmetric torus (top row) and Stanford Bunny (bottom row). Columns show $t = 0$, 20, and 200. The simulations use 8th-order elements, IMEX–BDF1 time stepping, and parameters $r_1 = 0.02$, $r_2 = 0.2$.

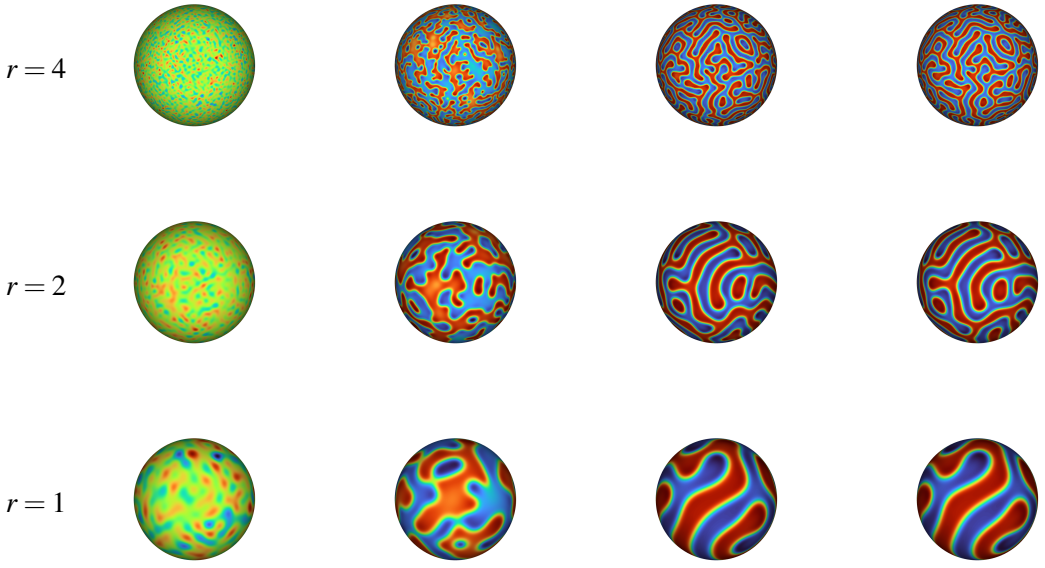


Figure 6.16: Activator concentration for the reaction–diffusion model on spheres of different radii. Rows correspond to radii $r = 4$, $r = 2$, and $r = 1$, and columns correspond to times $t = 0$, 10, 300, and 600. The stripe-forming parameter set is evolved using the IMEX–BDF1 scheme.

to give the model:

$$\begin{aligned}
\frac{\partial v_1}{\partial t} &= \delta_{v_1}\Delta_\Gamma v_1 + \alpha' v_1(1-r_1 v_2^2) + v_2(1-r_2 v_1) + q_1 u_1 + q_2 u_1 v_2 + q_3 u_1 v_2^2 \\
\frac{\partial v_2}{\partial t} &= \delta_{v_2}\Delta_\Gamma v_2 + \beta' v_2\left(1+\frac{\alpha' r_1}{\beta'} v_1 v_2\right) + v_1(\gamma' + r_2 v_2) - q_2 u_2 v_1 - q_3 u_2^2 v_1 \\
\frac{\partial u_1}{\partial t} &= \delta_{u_1}\Delta_\Gamma u_1 + \alpha u_1(1-r_1 u_2^2) + u_2(1-r_2 u_1) \\
\frac{\partial u_2}{\partial t} &= \delta_{u_2}\Delta_\Gamma u_2 + \beta u_2\left(1+\frac{\alpha r_1}{\beta} u_1 u_2\right) + u_1(\gamma + r_2 u_2).
\end{aligned} \tag{6.42}$$

This setup allows us to investigate how pattern properties change due to the interaction between the two coupled systems. For the simulations, we use the following parameter values:

- For the (v_1, v_2) system:

$$\alpha' = 0.398, \quad \beta' = -0.41, \quad \gamma' = -\alpha', \quad \delta_{v_2} = 5\times 10^{-3}, \quad \delta_{v_1} = 0.122\,\delta_{v_2}.$$

- For the (u_1, u_2) system:

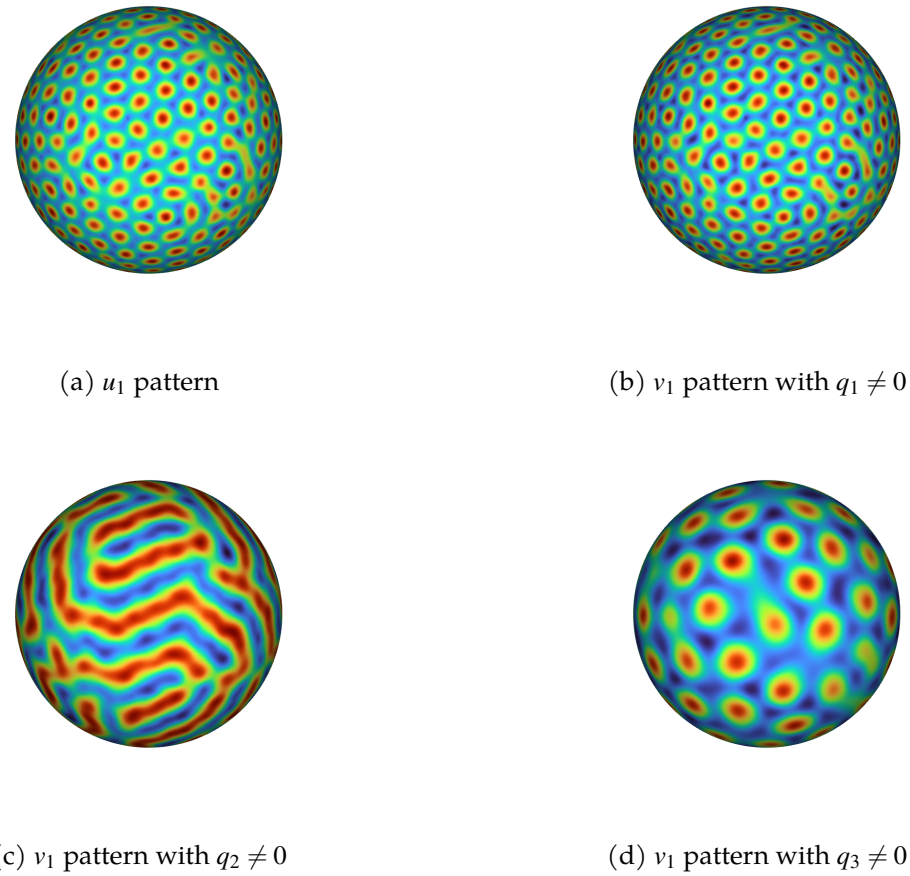
$$\alpha = 0.899, \quad \beta = -0.91, \quad \gamma = -\alpha, \quad \delta_{u_2} = \delta_{v_2}, \quad \delta_{u_1} = 0.516\,\delta_{u_2}.$$

In Figure 6.17, we show the resulting patterns for the cases of linear, quadratic, and cubic coupling. When only linear coupling is present, the pattern of v_1 becomes identical to that of u_1, indicating that the coupling completely overrides the dynamics of v_1. With cubic coupling, the solution still consists of spots, and the overall structure remains similar to the uncoupled case, suggesting that this type of interaction does not significantly alter the pattern. The most noticeable change occurs with quadratic coupling: the spot pattern is distorted and appears overlaid on a background of stripes, showing that the quadratic term introduces a strong modulation and leads to a mixed pattern with both spots and labyrinthine features.

When the signs of selected coupling coefficients are changed (see Figure 6.18), the behavior changes significantly. The linear term produces a negative image of u_1 in v_1, effectively inverting the pattern. In the other cases, the patterns in v_1 appear as a superposition of stripes and spots. Specifically, quadratic coupling favors spot formation, while cubic coupling leads to more spatially ordered spots.

6.7 Evolving surfaces

The previous simulations on static surfaces show clearly how surface geometry affects the formation of reaction–diffusion patterns. Yet in many biological and physical systems, the surface itself changes over time, influencing the development and structure of these patterns. Accounting for surface evolution adds considerable complexity: the resulting models are non-autonomous, nonlinear parabolic PDEs, for which standard tools—like linear stability analysis



(a) u_1 pattern

(b) v_1 pattern with $q_1 \neq 0$

(c) v_1 pattern with $q_2 \neq 0$

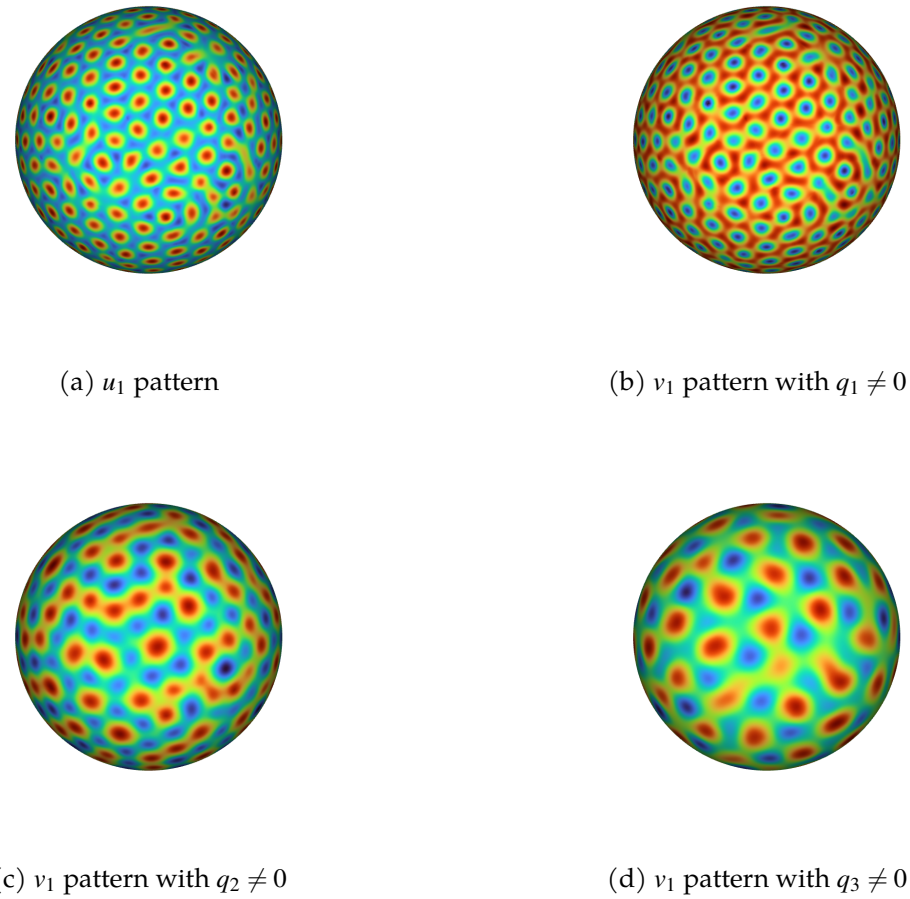
(d) v_1 pattern with $q_3 \neq 0$

Figure 6.17: Patterns obtained with the coupled system equation (6.42). Each column shows the effect of a single coupling term: linear (q_1), quadratic (q_2), or cubic (q_3), all with coupling strength 0.55.

near bifurcations—are no longer effective. This makes it essential to develop numerical methods that can handle such systems accurately and efficiently.

In response, we extend the high-order spectral framework presented in the first part of this chapter to evolving surfaces, in which both the geometry and the solution vary over time. Our approach uses a decoupled strategy: at each time step, we first update the surface geometry using a prescribed evolution law, then solve the PDE on the updated surface. In the numerical examples below, the surface motion is prescribed independently of the solution; material-transport effects such as advective and dilution terms are therefore not included in the PDE model.

There is an important computational distinction between this setting and the static time-dependent problems of Section 6.5. On a fixed surface, the implicit matrix A in (6.34) is unchanged and a single HPS factorization can be reused over all time steps. On an evolving surface, both the geometry and the local differential operators change with time, so the HPS operators generally have to be rebuilt, or at least updated, when the surface changes. Consequently, the evolving-surface examples in this chapter should be viewed as a demonstration that the high-order geo-



(a) u_1 pattern

(b) v_1 pattern with $q_1 \neq 0$

(c) v_1 pattern with $q_2 \neq 0$

(d) v_1 pattern with $q_3 \neq 0$

Figure 6.18: Patterns obtained with the coupled system equation (6.42), using the same parameters as in Figure 6.17, but with selected coupling signs changed.

metric discretization and HPS framework can be applied to each time slice, not as evidence that a fast direct solver is automatically optimal for every deforming-surface simulation. Whether rebuilding is worthwhile depends on the number of right-hand sides solved per geometry, the size of the deformation between time steps, and the possibility of reusing previous factorizations as preconditioners.

A time-dependent surface $\Gamma(t)$ is typically described using either an implicit or a Lagrangian representation. A widely used implicit approach defines $\Gamma(t)$ as the zero level set of a time-dependent scalar function $\varphi : \mathbb{R}^{d+1} \times [0,T] \to \mathbb{R}$, i.e.,

$$\Gamma(t) = \{\mathbf{x} \in \mathbb{R}^{d+1} : \varphi(\mathbf{x},t) = 0\}. \tag{6.43}$$

In this formulation, the evolution of the surface is governed by the dynamics of the level set function φ. Alternatively, a Lagrangian description tracks the motion of surface points explicitly through a velocity field $\boldsymbol{v} : \mathbb{R}^{d+1} \times [0,T] \to \mathbb{R}^{d+1}$, which prescribes the flow driving the deformation of the surface over time. In this case, the surface $\Gamma(t)$ is parameterized over its initial

configuration $\Gamma(0)$ by following the trajectories of points on the surface. For an initial point $\mathbf{x}_0 \in \Gamma(0)$, the trajectory $\mathbf{x}(t)$ satisfies the ordinary differential equation

$$\frac{d}{dt}\mathbf{x}(t) = \boldsymbol{\nu}(\mathbf{x}(t),t), \quad \mathbf{x}(0) = \mathbf{x}_0. \tag{6.44}$$

If equation (6.44) can be solved in closed form, yielding an explicit expression for $\mathbf{x}(t)$, then we define the parameterization of $\Gamma(t)$ over the initial surface $\Gamma(0)$ as:

$$\Gamma(t) := \{\mathbf{x}(t) \mid \mathbf{x}_0 \in \Gamma(0)\}.$$

If an analytic solution is not known, we use the explicit Euler method to approximate the ODE system in equation (6.44):

$$\mathbf{x}_{m+1} = \mathbf{x}_m + \Delta t\, \boldsymbol{\nu}_m.$$

To study numerical approximations of reaction–diffusion systems on evolving surfaces, we incorporate the surface evolution law into the closest-point projection operator in equation (5.10). This yields a time-dependent closest-point projection operator.

6.7.1 Time-dependent oriented distance function

We assume that there exists an open bounded set $U(t) \subset \mathbb{R}^{d+1}$ such that $\partial U(t) = \Gamma(t)$. The *time-dependent oriented distance function* $d_{\Gamma(t)}(\mathbf{x},t)$ is defined as

$$\mathbb{R}^{d+1} \times [0,T] \to \mathbb{R}, \quad d_{\Gamma(t)}(\mathbf{x},t) := \begin{cases} \operatorname{dist}(\mathbf{x},\Gamma(t)) & \mathbf{x} \in \mathbb{R}^{d+1} \setminus U(t), \\ -\operatorname{dist}(\mathbf{x},\Gamma(t)) & \mathbf{x} \in U(t). \end{cases}$$

For $\delta > 0$ we define $\mathcal{N}_\delta^t := \left\{\mathbf{x} \in \mathbb{R}^{d+1} \,\middle|\, \operatorname{dist}(\mathbf{x},\Gamma(t)) < \delta\right\}$. Clearly $\mathcal{N}_\delta^t$ is an open neighborhood of $\Gamma(t)$. For each $\mathbf{x} \in \mathcal{N}_\delta^t$, there exists a unique time-dependent closest point projection operator $\pi_t(\mathbf{x}) \in \Gamma(t)$, defined by

$$\pi_t(\mathbf{x}) = \mathbf{x} + d_{\Gamma(t)}(\mathbf{x},t)\nabla d_{\Gamma(t)}(\mathbf{x},t).$$

By analogy with the stationary case (see Lemma 65), the existence of $d_{\Gamma(t)}$ is readily established, and its regularity follows directly from the smoothness of the parametrization of the surface.

To account for surface evolution over time, we compose the stationary closest point projection π_0 with a *Lipschitz continuous mapping* $\mathcal{L}_t : \Gamma(0) \to \Gamma(t)$, which tracks the deformation of surface points. The inverse $\mathcal{L}_t^{-1}$ is also assumed to exist and to be Lipschitz continuous.

Definition 105. *Let $\mathcal{L}_t : \Gamma(0) \to \Gamma(t)$, $t \in [0,T]$, be Lipschitz continuous with Lipschitz continuous inverse $\mathcal{L}_t^{-1} : \Gamma(t) \to \Gamma(0)$. In other words, there exist constants $c, C > 0$ such that*

$$\|\mathcal{L}_t(\mathbf{x}_1) - \mathcal{L}_t(\mathbf{x}_2)\| \le c\|\mathbf{x}_1 - \mathbf{x}_2\| \quad \textit{and} \quad \|\mathcal{L}_t^{-1}(\mathbf{y}_1) - \mathcal{L}_t^{-1}(\mathbf{y}_2)\| \le C\|\mathbf{y}_1 - \mathbf{y}_2\|.$$

Assume further that $\mathcal{L}_t$ extends to a tubular neighborhood of $\Gamma(0)$. The time-dependent closest point projection operator is then defined by transporting a point to the initial configuration, projecting it onto $\Gamma(0)$, and mapping the result back to $\Gamma(t)$:

$$\pi_t(\mathbf{x}) := \mathcal{L}_t\left(\pi_0\left(\mathcal{L}_t^{-1}(\mathbf{x})\right)\right), \qquad \mathbf{x} \in \mathcal{N}_\delta^t. \tag{6.45}$$

This composition ensures that the projection operator π_t accounts for both the geometry of the initial surface and its deformation in time. The closest point projection on the right-hand side is the stationary map π_0, while $\mathcal{L}_t$ carries the time dependence.

We apply the time-dependent projection operator in the numerical solution of PDEs on evolving surfaces and in the construction of triangulated surface meshes at each time step. In this context, the mapping $\mathcal{L}_t$ is often referred to as an *arbitrary Lagrangian–Eulerian* (*ALE*) map [82].

6.8 Surface approximation and ALE-based regularization

Let a smooth initial surface $\Gamma(0)$ be approximated by a triangulated surface $\Gamma_h^{\text{tri}}(0)$, i.e., by a quasi-uniform family of triangulations $\widehat{\mathcal{T}}_h(0)$ of maximal element diameter h. Let $\mathbf{x}_k(0)$, $(k = 1,2,\ldots,N)$ denote the points of $\Gamma_h^{\text{tri}}(0)$ lying on the initial smooth surface $\Gamma(0)$. The points will be evolved in time with the given normal velocity $\boldsymbol{v}$, by solving the ODE

$$\frac{d}{dt}\mathbf{x}_k(t) = \boldsymbol{v}(\mathbf{x}_k(t),t) \quad (k = 1,2,\ldots,N), \tag{6.46}$$

where $\boldsymbol{v}$ represents the velocity field. For the prescribed evolution considered here, the points remain on the surface $\Gamma(t)$ for all times, i.e., $d_{\Gamma(t)}(\mathbf{x}_k(t),t) = 0$ for $k = 1,2,\ldots,N$ and for all $t \in [0,T]$.

Let $\triangle_d \subset \mathbb{R}^d$ be the standard simplex of dimension d and let $[0,T]$ be a time interval. A linear evolving mesh or evolving discrete surface on $[0,T]$ consists of continuously time-dependent points $\mathbf{x}_k(t)_{k=1}^N \subset \mathbb{R}^{d+1}$ and d-dimensional simplex relations $\widehat{T}(t) \in \widehat{\mathcal{T}}_h(t)$, where we identify $\widehat{T}(t) \subset \mathbb{R}^{d+1}$ with the simplex itself, which we require to satisfy:

- $\widehat{T}(t) \in \widehat{\mathcal{T}}_h(t)$ is non-degenerate, i.e., the map $\rho : \triangle_d \times [0,T] \to \widehat{T}(t)$, is a bijection.
- The intersection of two simplices is a common edge, a common point, or empty.
- There are no boundary simplices, i.e., every edge is the intersection of two different simplices.

We set the mesh width h as

$$h(t) := \max_{\widehat{T}(t)\in\widehat{\mathcal{T}}_h(t)} \operatorname{diam}(\widehat{T}(t)), \quad h := \sup_{t\in[0,T]} h(t),$$

where diam denotes the 2-dimensional diameter, the *in-ball radius* at time t is defined as

$$r(t) := \min_{\widehat{T}(t)\in\widehat{\mathcal{T}}_h(t)} \rho(\widehat{T}(t)),$$

where ρ denotes the radius of the maximum inner circle. We set

$$\mathbf{x}_h(t) := (\mathbf{x}_k(t))_{k=1}^N \in \mathbb{R}^N \otimes \mathbb{R}^{d+1}$$

and

$$\Gamma(\mathbf{x}_h) := \Gamma(\mathbf{x}_h(t)) := \Gamma_h^{\mathrm{tri}}(t) := \bigcup_{\widehat{T}(t)\in\widehat{\mathcal{T}}_h(t)} \widehat{T}(t).$$

A family $(\Gamma_h^{\mathrm{tri}}(t))_{h>0}$ is called *quasi-uniform*, if and only if

$$\sup_{t\in[0,T]} \frac{h(t)}{r(t)} \leq c.$$

However, the assumption of quasi-uniformity over time is generally not preserved during surface evolution. To illustrate this, we evolved a surface according to the ODE (6.46) over the interval $[0, 0.9]$, as shown in Figure 6.19. Although the initial mesh (left) is quasi-uniform and the deformation appears mild, the mesh quality deteriorates noticeably over time. In particular, the normal velocity causes the formation of poorly shaped elements, with triangles collapsing into near-zero angles. As vertices move according to the velocity field, the resulting triangulations may become unsuitable for numerical simulation, with degeneracies caused by extreme angle distortion.

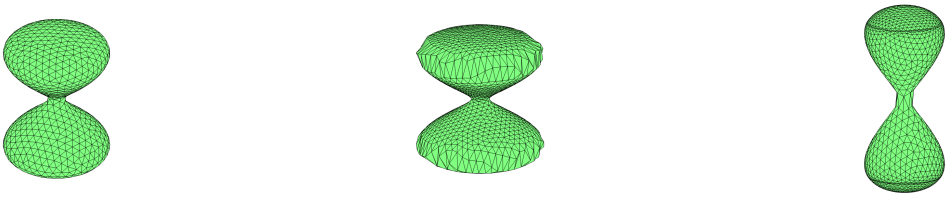


Figure 6.19: Closed-surface mesh evolved by the prescribed normal velocity at times $t = 0$, 0.2, and 0.9. The snapshots illustrate the deterioration of element quality under normal motion alone.

Standard remeshing techniques [56, 103] can be employed at each time step to maintain high mesh quality. However, these methods often introduce or remove mesh points, requiring the construction of mappings between successive meshes. While this is theoretically manageable, from an implementation perspective, it is highly undesirable due to the computational complexity and cost involved. To address this, we adopt the *Arbitrary Lagrangian–Eulerian* (ALE)

approach proposed in [82], which enables mesh-preserving surface evolution without frequent remeshing. The surface mesh then evolves according to

$$\begin{aligned} \frac{d}{dt}\mathbf{x}_k(t) &= \boldsymbol{v}(\mathbf{x}_k(t),t) + \boldsymbol{w}(\mathbf{x}_k(t),t), \quad k = 1,\dots,N, \\ d_{\Gamma(t)}(\mathbf{x}_k(t),t) &= 0, \end{aligned} \tag{6.47}$$

where $\boldsymbol{v}$ is the physical velocity and $\boldsymbol{w}$ is an ALE velocity that ensures mesh quality. Unlike purely tangential velocities, $\boldsymbol{w}$ may include normal components and is combined with a constraint to keep points on the surface, resulting in a differential–algebraic formulation.

The resulting differential algebraic equation system is:

$$\begin{aligned} \frac{d}{dt}\mathbf{x}(t) &= \boldsymbol{v}(\mathbf{x}(t),t) + \boldsymbol{w}(\mathbf{x}(t),t) - D(\mathbf{x}(t))^T \lambda(t), \\ d_{\Gamma(t)}(\mathbf{x}(t),t) &= 0, \ \text{ with } \ \mathbf{x}(0) = \mathbf{x}_0 \end{aligned} \tag{6.48}$$

with $D(\mathbf{x}(t)) := \frac{\partial d_{\Gamma(t)}(\cdot,t)}{\partial \mathbf{x}}$ and Lagrange multiplier $\lambda(t)$.

The ALE velocity $\boldsymbol{w}$ is computed using a virtual spring model defined along the mesh edges:

$$\boldsymbol{w}(\mathbf{x},t) = \kappa \mathbf{F}(\mathbf{x}) \tag{6.49}$$

where the force function $\mathbf{F}$ is computed based on the connectivity, and by the forces over the edges based on the desired length of springs. We refer the reader to [82, 103] for more details.

The numerical solution of the differential–algebraic system (6.48) yields the updated point positions while preserving mesh quality. Figure 6.20 illustrates an example of dumbbell-shaped surface evolution using the ALE framework, which, in contrast to normal evolution (see Figure 6.19), effectively prevents mesh distortion and maintains element regularity over time.

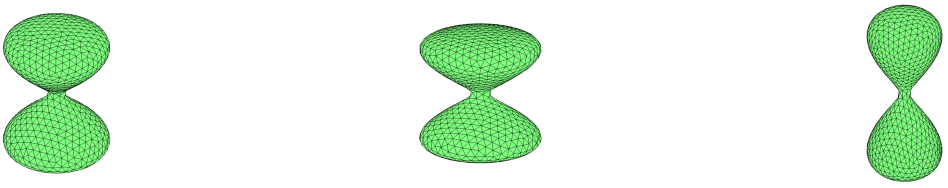


Figure 6.20: Closed-surface mesh evolved with the ALE regularization at times $t = 0$, 0.2, and 0.9. Compared with the purely normal evolution in Figure 6.19, the ALE velocity preserves mesh quality during deformation.

To optimize computational efficiency, we precompute the meshes used in our algorithm and store them for reuse, thereby avoiding the costly overhead of dynamic mesh generation at run-

time. Given such a triangulation, we incorporate it into the HPS framework by applying the parametrization strategies discussed earlier.

6.9 Surface growth

We focus on two growth models for the evolution of the surface.

6.9.1 Isotropic growth

Here we are interested in prescribed velocity laws associated with surface growth. As an illustrative example, consider the time-dependent surface $\Gamma(t)$ given by the zero level set of the distance function

$$d_{\Gamma(t)}(\mathbf{x},t) = a(t)^2 b\left(\frac{x_1^2}{c(t)^2}\right) + x_2^2 + x_3^2 - a(t)^2 .$$

Different surfaces may be obtained by prescribing the non-negative shape functions $a(\cdot)$, $b(\cdot)$, and $c(\cdot)$ with $b(0) = 0$ and $b(1) = 1$.

- **Tangential motion**
  For the described surface, material points can move both tangentially and normally. We represent the surface $\Gamma(t)$ using material points $(\tilde{x}_1(t), \tilde{x}_2(t), \tilde{x}_3(t))$, and parameterize it as follows: starting from $\tilde{x}_1(0) \in [-c(0),\, c(0)]$, we define $(\tilde{x}_2(0), \tilde{x}_3(0))$ to lie on the circle satisfying

  $$\tilde{x}_2(0)^2 + \tilde{x}_3(0)^2 = a(0)^2 \left(1 - b\left(\frac{\tilde{x}_1(0)^2}{c(0)^2}\right)\right),$$

  with the time evolution of the coordinates given by

  $$\tilde{x}_1(t) = \tilde{x}_1(0)\frac{c(t)}{c(0)}, \quad \tilde{x}_2(t) = \tilde{x}_2(0)\frac{a(t)}{a(0)}, \quad \tilde{x}_3(t) = \tilde{x}_3(0)\frac{a(t)}{a(0)}.$$

  Thus, the condition $d_{\Gamma(t)}(\tilde{\mathbf{x}}(t), t) = 0$ holds for the surface at any time t.

- **Normal motion**
  Alternatively, we may choose material points on the same surface such that the velocity of the material points, $\boldsymbol{v}(\tilde{\mathbf{x}}(t), t)$, is in the normal direction. The material velocity is described by

  $$\boldsymbol{v}(\mathbf{x},t) = V(\mathbf{x},t)\mathbf{n}(\mathbf{x},t), \quad \mathbf{n}(\mathbf{x},t) = \frac{\nabla d_{\Gamma(t)}(\mathbf{x},t)}{|\nabla d_{\Gamma(t)}(\mathbf{x},t)|}, \quad \text{and} \quad V(\mathbf{x},t) = -\frac{\partial_t d_{\Gamma(t)}(\mathbf{x},t)}{|\nabla d_{\Gamma(t)}(\mathbf{x},t)|}, \tag{6.50}$$

  where $\mathbf{n}(\mathbf{x},t)$ represents the unit normal to the surface at each point.

6.10 Pattern formation on evolving surfaces

In this section, we investigate the impact of surface evolution on Turing pattern formation by solving reaction–diffusion systems on time-dependent geometries. Through a series of examples, we illustrate how both isotropic and anisotropic surface deformations affect the emergence,

spatial distribution, and symmetry of patterns over time. The reaction–diffusion system is advanced in time using a fourth-order implicit–explicit backward differentiation formula (IMEX–BDF4) scheme.

Throughout these examples, the reaction–diffusion equations are treated in the decoupled sense described above: first the geometry is updated, and then the concentrations are advanced on the current surface. Thus the equations below do not include the additional advective or dilution terms that arise in a fully conservative material formulation on an evolving surface.

6.10.1 Isotropic growth and decay

Logistic growth

In this example, we consider an evolving sphere with initial radius $r = 1$, whose evolution is driven by a *logistic growth function* $\eta(t)$, defined as follows:

$$\eta(t) = \frac{e^{g_{\text{rate}}t}}{1 + \frac{1}{K}\left(e^{g_{\text{rate}}t} - 1\right)}, \quad \text{and} \quad \mathcal{L}_t(\mathbf{x}_0) := \eta(t)\mathbf{x}_0, \tag{6.51}$$

where $g_{\text{rate}} = 0.1$ and $K = 1.5$. Here g_{rate} is the growth rate of the surface and K is the limiting final fixed size of the radius. To project points onto the initial sphere $\Gamma(0)$, we use the closest point projection operator $\pi(\mathbf{x}) := \frac{\mathbf{x}}{\|\mathbf{x}\|}$. The time-dependent closest point projection $\pi_t(\mathbf{x}) : \mathcal{N}_\delta^t \to \Gamma(t)$ is then given by the composition

$$\pi_t(\mathbf{x}) := (\mathcal{L}_t \circ \pi)(\mathbf{x}) = \eta(t)\frac{\mathbf{x}}{\|\mathbf{x}\|}, \tag{6.52}$$

which ensures that $\pi_t(\mathbf{x})$ lies on the evolving surface $\Gamma(t)$ with radius $\eta(t)$. This composition accounts for both the geometry of the initial surface and its deformation over time.

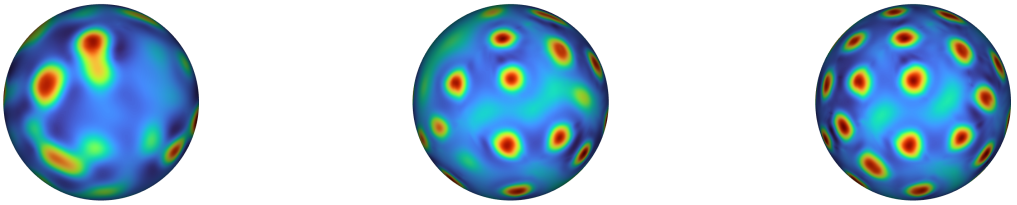
We consider reaction–diffusion systems defined on evolving surfaces $\Gamma(t)$,

$$\frac{\partial u_1}{\partial t} = \delta_{u_1}\Delta_{\Gamma(t)}u_1 + \alpha u_1\left(1 - r_1 u_2^2\right) + u_2\left(1 - r_2 u_1\right), \tag{6.53a}$$

$$\frac{\partial u_2}{\partial t} = \delta_{u_2}\Delta_{\Gamma(t)}u_2 + \beta u_2\left(1 + \frac{\alpha r_1}{\beta}u_1 u_2\right) + u_1\left(\gamma + r_2 u_2\right). \tag{6.53b}$$

For the simulation, we take $\alpha = 0.899$, $\beta = -0.91$, $\gamma = -\alpha$, $r_1 = 0.02$, $r_2 = 0.15$, $\delta_{u_1} = 0.0026$, and $\delta_{u_2} = 0.005$. Whenever a parameter differs from these default values in a specific experiment, it is explicitly stated.

Figure 6.21 shows the evolution of the u_1 concentration in the Turing model on a sphere that expands isotropically according to the logistic growth law (6.51). The plots correspond to times $t = 10, 30$, and 50. Initially, the pattern begins to emerge from small perturbations around the homogeneous state. As the surface grows, more spots form and gradually stabilize. The number and spacing of the spots appear to be influenced by the increasing surface area, which allows the activation of higher spatial modes over time.



(a) $t = 10$ (b) $t = 30$ (c) $t = 50$

Figure 6.21: Activator concentration u_1 for the Turing model on an isotropically expanding sphere. The radius evolves according to the logistic growth law (6.51); snapshots are shown at $t = 10$, 30, and 50, using time step $\Delta t = 10^{-2}$.

Linear contracting sphere

In the second example, we repeat the experiment using the same reaction–diffusion system (6.53) and parameter values as in the expanding case. However, instead of surface expansion, we consider surface contraction with $\eta(t) = 1 - g_{\text{rate}}t$ and $g_{\text{rate}} = 0.02$. The initial surface $\Gamma(0)$ is a sphere of radius $r = 3$, which contracts over time to a sphere of radius 0.1. As shown in Figure 6.22, the final pattern on the sphere with radius 0.1 is similar to the pattern that forms on a stationary sphere with radius $r = 3$ when $\delta_{u_1} = 0.2758$ (see Figure 6.23). Hence, this experiment demonstrates that contracting the surface can be interpreted as increasing the effective value of δ_{u_1} without surface evolution. Fewer patterns form as the surface continues to contract, in contrast to the formation of more patterns when the surface expands as shown in Figure 6.21.

6.10.2 Anisotropic growth

Anisotropic evolution of a sphere into an ellipsoid.

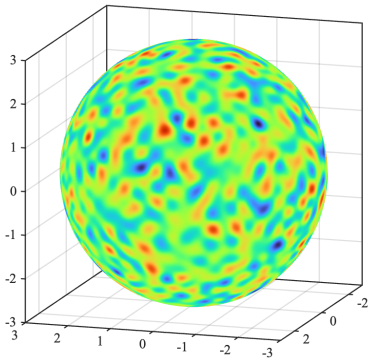
We begin with the initial surface $\Gamma(0)$ and assume that it moves with a velocity field given by $\boldsymbol{v}(\mathbf{x}(t),t) := (0,0,g_{\text{rate}}x_3(0))$, where g_{rate} is a fixed growth factor. This implies that the anisotropic deformation of the surface is governed by the following time-dependent closest point projection

$$\pi(\mathbf{x}(t),t) := (\mathcal{L}_t \circ \pi)(\mathbf{x}(0),t) = \left(\frac{x_1(0)}{\|\mathbf{x}(0)\|}, \frac{x_2(0)}{\|\mathbf{x}(0)\|}, \frac{x_3(0)}{\|\mathbf{x}(0)\|}(1 + g_{\text{rate}}t) \right)^{\top}. \tag{6.54}$$

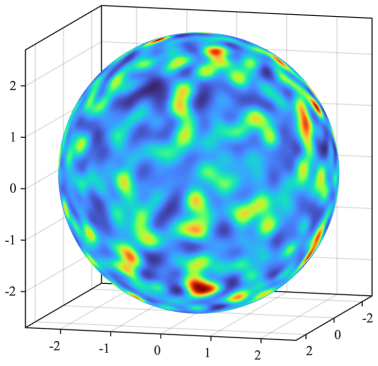
Accordingly, the surface at time t can be described as the zero level set of a time-dependent distance function:

$$\Gamma(t) := \left\{ \mathbf{x} \in \mathbb{R}^3 \,:\, d_{\Gamma(t)}(\mathbf{x},t) = 0 \right\}, \quad \text{where} \quad d_{\Gamma(t)}(\mathbf{x}(t),t) := x_1^2 + x_2^2 + \frac{x_3^2}{(1 + g_{\text{rate}}t)^2} - 1.$$

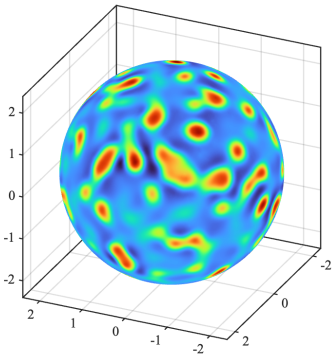
The final time is set to $T = 60$ and the growth rate is $g_{\text{rate}} = 0.04$, so that the final surface $\Gamma(T)$ is an ellipsoid whose equation is $x_1^2 + x_2^2 + \frac{x_3^2}{(\frac{17}{5})^2} = 1$. In Figure 6.24, we observe the continuous formation of spots as the surface evolves. This demonstrates how anisotropic deformation into a prolate ellipsoid enhances directional effects, leading to elongated and spatially organized



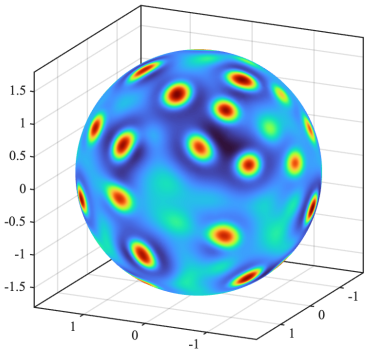
(a) $t = 0$



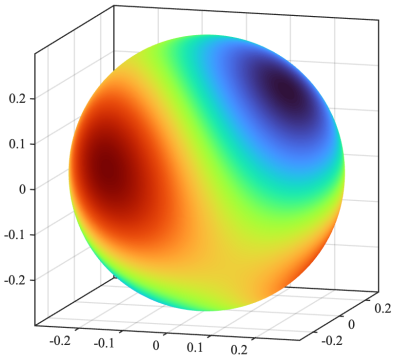
(b) $t = 5$



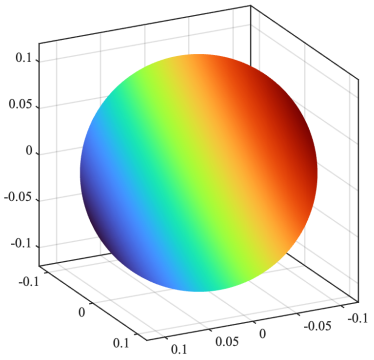
(c) $t = 10$



(d) $t = 20$



(e) $t = 45$



(f) $t = 48$

Figure 6.22: Activator concentration u_1 for the reaction–diffusion system (6.41) on a linearly contracting sphere. The initial radius is $r = 3$, the sphere contracts toward radius 0.1, and snapshots are shown at $t = 0$, 5, 10, 20, 45, and 48.

patterns aligned with the principal axis of growth.

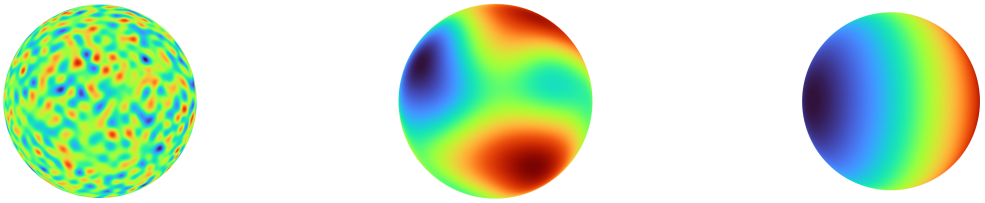
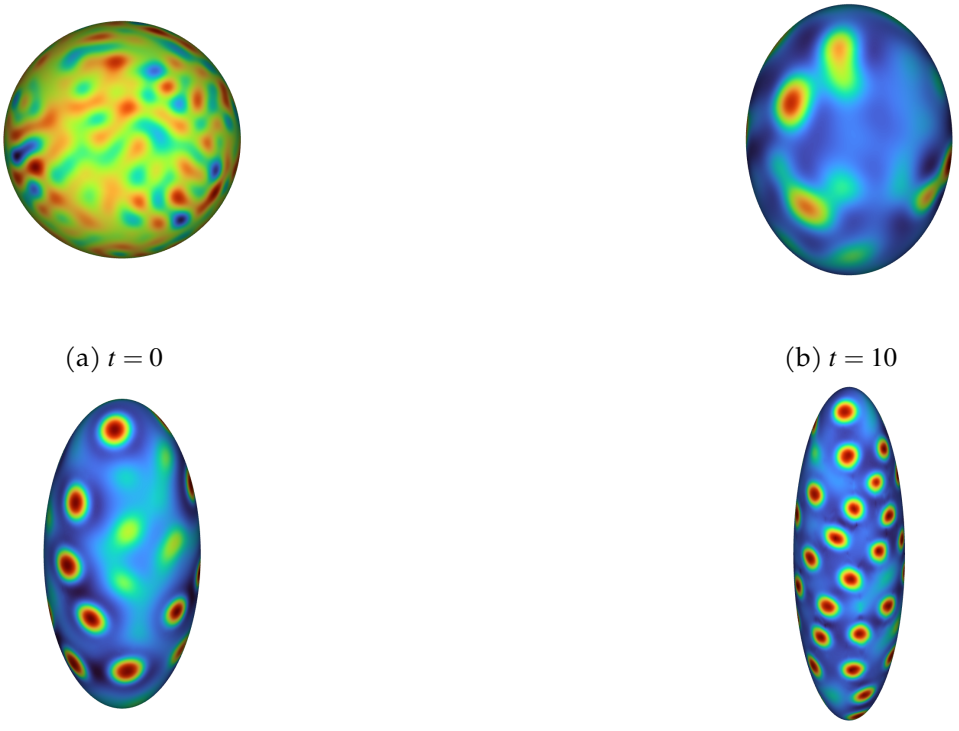


Figure 6.23: Activator concentration u_1 for the reaction–diffusion system (6.41) on a stationary sphere with $r = 3$. From left to right, the diffusion coefficient is increased through $\delta_{u_1} = 0.0026$, $\delta_{u_1} = 0.0176$, and $\delta_{u_1} = 0.2758$, providing a comparison for the contracting-sphere experiment.



(a) $t = 0$ (b) $t = 10$

(c) $t = 30$ (d) $t = 60$

Figure 6.24: Activator concentration u_1 for the reaction–diffusion system (6.41) on an anisotropically growing unit sphere that evolves into a prolate ellipsoid. Snapshots are shown at $t = 0$, 10, 30, and 60; the simulation uses IMEX–BDF4 with $\Delta t = 0.1$.

An anisotropic evolution of the dumbbell

We define the evolving closed surface $\Gamma(t)$ as the zero level set of a time-dependent distance function:

$$d_{\Gamma(t)}(\mathbf{x}(t),t) = x_1^2 + x_2^2 + a(t)^2 b\left(\frac{x_3^2}{c(t)^2}\right) - a(t)^2, \quad \text{i.e., } \Gamma(t) := \{\mathbf{x} \in \mathbb{R}^3 \,:\, d_{\Gamma(t)}(\mathbf{x}(t),t) = 0\}. \quad (6.55)$$

Here the functions a, b, and c are given by

$$a(t) = 0.1 + 0.05\sin(2\pi t), \quad b(s) = 200s\left(s - \frac{199}{200}\right), \quad c(t) = 1 + 0.2\sin(4\pi t).$$

The surface velocity $\boldsymbol{v}$ at $\mathbf{x}_k(t)$ is given by equation (6.50), and the coordinates evolve in time according to

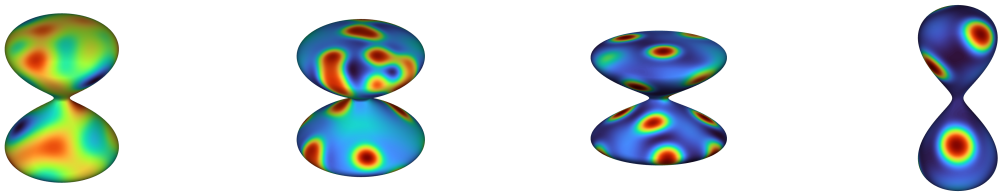
$$(\mathbf{x}_k(t))_1 = (\mathbf{x}_k(0))_1 \frac{a(t)}{a(0)}, \quad (\mathbf{x}_k(t))_2 = (\mathbf{x}_k(0))_2 \frac{a(t)}{a(0)}, \quad (\mathbf{x}_k(t))_3 = (\mathbf{x}_k(0))_3 \frac{c(t)}{c(0)}.$$

Figure 6.25 shows the time evolution of the u_1 solution in the Turing model, obtained with the spot-forming parameter set listed in Table 6.1. The simulation runs until final time $T = 60$, with snapshots shown at $t = 0, 10, 40$, and 60. As the surface deforms, localized spot patterns emerge and stabilize, influenced by the curvature variations near the neck and lobes. These geometric features appear to guide the spatial distribution and persistence of the spots, indicating a strong coupling between surface shape and pattern dynamics.

In contrast, Figure 6.26 presents results obtained with a different parameter set that favors stripe formation. While the underlying surface evolution remains the same, the altered kinetics lead to the emergence of stripe-like structures aligned along the axial direction. This highlights the role of both geometry and reaction parameters in determining the symmetry and mode of pattern formation.

Table 6.1: Reaction–diffusion parameters used for the anisotropically evolving dumbbell experiments in Figures 6.25 and 6.26. In both cases, $\delta_{u_1} = 0.5166\delta_{u_2}$.

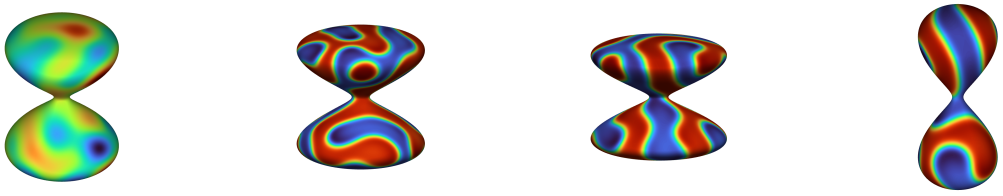
Pattern	δ_{u_2}	α	β	γ	r_1	r_2	Final time
Spots	4.5×10^{-3}	0.899	-0.91	-0.899	0.02	0.15	60
Stripes	5×10^{-3}	1.899	-0.95	-1.899	1.5	0	60



$t = 0$ $t = 10$ $t = 40$ $t = 60$

Figure 6.25: Spot-forming activator concentration u_1 for the reaction–diffusion system (6.41) on an anisotropically growing dumbbell surface. The parameter set is given in Table 6.1; snapshots are shown at $t = 0$, 10, 40, and 60, using IMEX–BDF4 with $\Delta t = 0.1$.

The results in this chapter show that high-order patchwise spectral discretizations, combined with HPS merging, provide an effective framework for elliptic and time-dependent PDEs on smooth surfaces. The quadrilateralization strategy makes triangulated data compatible with



$t = 0$ $t = 10$ $t = 20$ $t = 40$

Figure 6.26: Stripe-forming activator concentration for the reaction–diffusion system (6.41) on the same anisotropically growing dumbbell surface. The parameter set is given in Table 6.1; snapshots are shown at $t = 0$, 10, 20, and 40. Red and blue denote high and low activator concentration, respectively.

the tensor-product structure of the solver, while the hypercube-to-simplex reparametrization identifies a promising route toward treating triangular elements more directly. The evolving-surface experiments demonstrate that the same geometric framework can be applied to prescribed time-dependent surfaces, although the need to rebuild geometry-dependent operators limits the direct reuse of HPS factorizations.

6.11 Future work

As previously mentioned, in this chapter we have considered the decoupled case, where the governing equation is solved on a given surface $\Gamma(t)$, assuming that its evolution is prescribed independently. Future work will investigate the fully coupled system, where the surface evolution

$$\frac{\partial \Gamma}{\partial t} = \mathcal{F}(\Gamma, u) \tag{6.56}$$

is solved simultaneously with the governing equation, introducing additional geometric constraints and numerical challenges. This approach will allow for a more accurate and efficient numerical treatment of coupled reaction–diffusion or transport processes on evolving manifolds.

Further challenges and opportunities for extension are:

- *Hypercube-to-simplex reparametrization*: A promising direction for making the hypercube-to-simplex reparametrization useful in practical HPS computations is to decouple the volume discretization from the interface discretization. The square-squeezing map allows the differential operators on a triangular element to be computed on the reference square, where tensor-product Chebyshev differentiation matrices are stable and efficient. However, the induced trace discretization is not naturally triangular: one physical edge, corresponding to the squeezed hypotenuse, inherits boundary points from two square edges and therefore carries a different number of trace points than the other two edges. This prevents a direct use of the standard HPS merging procedure on general triangulated meshes.

  A possible remedy is to keep the square-squeezed tensor-product grid for the interior element solve, while replacing the boundary representation by a triangle-compatible trace

space. In such a formulation, each triangular element would expose three physical edges with comparable one-dimensional interpolation nodes, while the interior solution and the normal fluxes would still be computed through the square-squeezed parametrization. The transfer between the square boundary data and the triangular trace data could be performed by interpolation, projection, or a mortar-type coupling operator. This would lead to a non-matching interface HPS method in which neighboring triangles are merged through weak continuity of the solution and flux balance, rather than by requiring identical Chebyshev point distributions on both sides of an interface.

This approach would turn the square-squeezing idea into a practical triangular HPS method: the local spectral differentiation would retain the numerical advantages of tensor-product square discretizations, while the global solver would see a conforming triangular mesh skeleton. Such a method would be especially interesting to compare against direct triangular polynomial discretizations, such as Koornwinder–Dubiner or PKD-type bases, on model Laplace–Beltrami problems on the sphere.

- *Extending Methodology to Vector-Valued PDEs:* The study of viscous flows on curved membranes is fundamental for understanding biological transport processes. A promising avenue is the expansion of our approach to include vector-valued partial differential equations (PDEs), with particular emphasis on solving surface Stokes equations.
- *Handling non-parametric Surfaces:* The meshfree geometric-computing framework described in [138] uses polynomial regression over point-cloud level sets to represent complex-shaped and dynamic surfaces and geometries. This approach is particularly useful for problems involving non-parametric or dynamic geometries. It can be combined with standard mesh generation algorithms at each time step to produce high-quality unstructured triangular meshes.
- *Orthonormal triangular bases:* Finally, an interesting direction for future work is the incorporation of orthonormal triangular bases—e.g., Koornwinder–Dubiner polynomials—into the HPS scheme, which would allow high-order spectral approximation to be carried out directly on unstructured triangular meshes.

Evolving and deformable surfaces remain a challenging aspect for the solver, as a single factorization can no longer be leveraged to accelerate all time steps. This limits the direct benefit of HPS for a single right-hand side on a rapidly deforming mesh and raises the need for effective preconditioners for PDEs on moving surfaces based on high-order domain decomposition methods. While fast direct solvers for static surfaces enable high-order accurate solutions for multiple right-hand sides—for example within implicit time-stepping schemes—they must typically be rebuilt when the surface evolves, leading to significant computational cost.

A natural direction is therefore to investigate how a direct solver constructed for a previous surface configuration can be reused as a preconditioner for a nearby surface PDE, thereby amortizing the setup cost over multiple time steps.

In addition, to reduce the cost of constructing local solution operators from $\mathcal{O}(n^6)$ to $\mathcal{O}(n^4)$, we propose the use of ultraspherical spectral methods [98] for element-wise discretization. These methods exploit orthogonal polynomial representations and sparse recurrence relations between operators, offering a promising pathway toward more efficient high-order solvers for surface PDEs.

Bibliography

[1] Robert A Adams and John JF Fournier. *Sobolev spaces*. Elsevier, 2003.

[2] Christoph Aistleitner and Josef Dick. Functions of bounded variation, signed measures, and a general Koksma-Hlawka inequality. *arXiv preprint arXiv:1406.0230*, 2014.

[3] Martin Alkämper, Andreas Dedner, Robert Klöfkorn, and Martin Nolte. The Dune-alugrid module. *arXiv preprint arXiv:1407.6954*, 2014.

[4] Jürgen Appell, Józef Banas, and Nelson José Merentes Díaz. *Bounded variation and around*, volume 17. Walter de Gruyter, 2013.

[5] Uri M Ascher, Steven J Ruuth, and Brian TR Wetton. Implicit-explicit methods for time-dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 32(3):797–823, 1995.

[6] Kendall E Atkinson and David Chien. Piecewise polynomial collocation for boundary integral equations. *SIAM Journal on Scientific Computing*, 16(3):651–681, 1995.

[7] Tracy Babb, Adrianna Gillman, Sijia Hao, and Per-Gunnar Martinsson. An accelerated Poisson solver based on multidomain spectral discretization. *BIT Numerical Mathematics*, 58:851–879, 2018.

[8] RA Barrio, C Varea, JL Aragón, and PK Maini. A two-dimensional numerical study of spatial pattern formation in interacting Turing systems. *Bulletin of mathematical biology*, 61(3):483–505, 1999.

[9] Peter Bastian, Markus Blatt, Andreas Dedner, Nils-Arne Dreier, Christian Engwer, René Fritze, Carsten Gräser, Christoph Grüninger, Dominic Kempf, Robert Klöfkorn, et al. The Dune framework: Basic concepts and recent developments. *Computers & Mathematics with Applications*, 81:75–112, 2021.

[10] Richard Bellman, Robert E Kalaba, and Arthur E Bryson. *Dynamic programming and modern control theory*, volume 81. Citeseer, 1965.

[11] Matthew Berger, Andrea Tagliasacchi, Lee M Seversky, Pierre Alliez, Gael Guennebaud, Joshua A Levine, Andrei Sharf, and Claudio T Silva. A survey of surface reconstruction

from point clouds. In *Computer graphics forum*, volume 36, pages 301–329. Wiley Online Library, 2017.

[12] Serge Bernstein. Sur la limitation des valeurs d'un polynôme $p_n(x)$ de degré n sur tout un segment par ses valeurs en $(n+1)$ points du segment. *Izv. Akad. Nauk SSSR*, 7:1025–1050, 1931.

[13] Jean-Paul Berrut and Lloyd N. Trefethen. Barycentric Lagrange interpolation. *SIAM review*, 46(3):501–517, 2004.

[14] Jean-Daniel Boissonnat, David Cohen-Steiner, Bernard Mourrain, Günter Rote, and Gert Vegter. Meshing of surfaces. In *Effective Computational Geometry for Curves and Surfaces*, pages 181–229. Springer, 2006.

[15] Jean-Daniel Boissonnat, Olivier Devillers, Sylvain Pion, Monique Teillaud, and Mariette Yvinec. Triangulations in CGAL. *Computational Geometry*, 22:5–19, 2002.

[16] Andrea Bonito and Ricardo H Nochetto. *Geometric Partial Differential Equations — Part I*. Elsevier, 2020.

[17] Len Bos. On certain configurations of points in $\mathbb{R}^n$ which are unisolvent for polynomial interpolation. *Journal of approximation theory*, 64(3):271–280, 1991.

[18] John P Boyd. *Chebyshev and Fourier spectral methods*. Courier Corporation, 2001.

[19] Paul Breiding and Nick Vannieuwenhoven. The condition number of Riemannian approximation problems. *SIAM Journal on Optimization*, 31(1):1049–1077, jan 2021.

[20] Haim Brezis. *Functional analysis, Sobolev spaces and partial differential equations*, volume 2. Springer, 2011.

[21] Matteo Briani, Alvise Sommariva, and Marco Vianello. Computing Fekete and Lebesgue points: Simplex, Square, Disk. *Journal of Computational and Applied Mathematics*, 236(9):2477–2486, 2012.

[22] William L Briggs, Van Emden Henson, and Steve F McCormick. *A multigrid tutorial*. SIAM, 2000.

[23] James William Bruce and Peter John Giblin. *Curves and Singularities: a geometrical introduction to singularity theory*. Cambridge University Press, 1992.

[24] Lev Brutman. On the Lebesgue function for polynomial interpolation. *SIAM Journal on Numerical Analysis*, 15(4):694–704, 1978.

[25] Lev Brutman. Lebesgue functions for polynomial interpolation – a survey. *Annals of Numerical Mathematics*, 4:111–128, 1996.

[26] Barry Bunow, Jean-Pierre Kernevez, Gislaine Joly, and Daniel Thomas. Pattern formation

by reaction-diffusion instabilities: Application to morphogenesis in Drosophila. *Journal of theoretical biology*, 84(4):629–649, 1980.

[27] Claudio Canuto and Daniele Funaro. The Schwarz algorithm for spectral methods. *SIAM journal on numerical analysis*, 25(1):24–40, 1988.

[28] Jonathan C. Carr, Richard K. Beatson, Jon B. Cherrie, Tim J. Mitchell, W. Richard Fright, Bruce C. McCallum, and Tim R. Evans. Reconstruction and representation of 3D objects with radial basis functions. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 2001.

[29] Giulio Casciola, Damiana Lazzaro, Laura Bacchelli Montefusco, and Serena Morigi. Shape preserving surface reconstruction using locally anisotropic radial basis function interpolants. *Comput. Math. Appl.*, 51:1185–1198, 2006.

[30] Qi Chen and Ivo Babuška. Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle. *Computer Methods in Applied Mechanics and Engineering*, 128(3-4):405–417, 1995.

[31] Qi Chen and Ivo Babuška. Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle. *Computer Methods in Applied Mechanics and Engineering*, 128(3):405–417, 1995.

[32] EW Cheney et al. *Introduction to approximation theory*. Providence: AMS Chelsea Publishing, 1966.

[33] David Da-Kwun Chien. Piecewise polynomial collocation for integral equations with a smooth kernel on surfaces in three dimensions. *The Journal of Integral Equations and Applications*, 5:315–44, 1993.

[34] PG Ciarlet. The finite element method for elliptic problems (siam, philadelphia). 2002.

[35] Charles W Clenshaw and Alan R Curtis. A method for numerical integration on an automatic computer. *Numerische Mathematik*, 2:197–205, 1960.

[36] Albert Cohen and Giovanni Migliorati. Multivariate approximation in downward closed polynomial spaces. In *Contemporary Computational Mathematics – A celebration of the 80th birthday of Ian Sloan*, pages 233–282. Springer, 2018.

[37] Timothy A. Davis. Algorithm 832: UMFPACK v4.3—an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software (TOMS)*, 30(2):196–199, 2004.

[38] Alan Demlow. Higher-order finite element methods and pointwise error estimates for elliptic problems on surfaces. *SIAM Journal on Numerical Analysis*, 47(2):805–827, 2009.

[39] Alan Demlow and Gerhard Dziuk. An adaptive finite element method for the Laplace–Beltrami operator on implicitly defined surfaces. *SIAM Journal on Numerical Analysis*, 45(1):421–442, 2007.

[40] Kathryn P Drake, Edward J Fuselier, and Grady B Wright. Implicit surface reconstruction with a curl-free radial basis function partition of unity method. *SIAM Journal on Scientific Computing*, 44(5):A3018–A3040, 2022.

[41] Tobin A Driscoll, Nicholas Hale, and Lloyd N Trefethen. Chebfun guide, 2014.

[42] M. G. Duffy. Quadrature over a pyramid or cube of integrands with a singularity at a vertex. *SIAM Journal on Numerical Analysis*, 19:1260–1262, 1982.

[43] David A. Dunavant. High degree efficient symmetrical Gaussian quadrature rules for the triangle. *International Journal for Numerical Methods in Engineering*, 21(6):1129–1148, 1985.

[44] Gerhard Dziuk and Charles M Elliott. Finite element methods for surface PDEs. *Acta Numerica*, 22:289–396, 2013.

[45] Jack Edmonds. Paths, Trees, and Flowers. *Canadian Journal of Mathematics*, 17:449–467, 1965.

[46] David Elliott, Peter R Johnston, and Barbara M Johnston. Estimates of the error in Gauss–Legendre quadrature for double integrals. *Journal of Computational and Applied Mathematics*, 236(6):1552–1561, 2011.

[47] Gerald E Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann, 2002.

[48] Leopold Fejér. Bestimmung derjenigen abszissen eines intervalles, für welche die quadratsumme der grundfunktionen der Lagrangeschen interpolation im intervalle ein möglichst kleines maximum besitzt. *Annali della Scuola Normale Superiore di Pisa, Scienze Fisiche e Matematiche*, 1(3):263–276, 1932.

[49] Bengt Fornberg. *A Practical Guide to Pseudospectral Methods*. Cambridge University Press, 1998.

[50] Daniel Fortunato. A high-order fast direct solver for surface pdes. *SIAM Journal on Scientific Computing*, 46(4):A2582–A2606, 2024.

[51] Daniel Fortunato, Nicholas Hale, and Alex Townsend. The ultraspherical spectral element method. *Journal of Computational Physics*, 436:110087, 2021.

[52] Leslie Fox and Ian Bax Parker. *Chebyshev Polynomials in Numerical Analysis*. Oxford University Press, 1968.

[53] Carl Friedrich Gauss. *Methodus nova integralium valores per approximationem inveniendi*. 1814.

[54] W Morven Gentleman. Implementing Clenshaw-Curtis quadrature, ii computing the cosine transformation. *Communications of the ACM*, 15(5):343–346, 1972.

[55] Alan George. Nested dissection of a regular finite element mesh. *SIAM journal on numerical analysis*, 10(2):345–363, 1973.

[56] Christophe Geuzaine and Jean-François Remacle. Gmsh: A 3-d finite element mesh generator with built-in pre-and post-processing facilities. *International journal for numerical methods in engineering*, 79(11):1309–1331, 2009.

[57] David Gilbarg, Neil S Trudinger, David Gilbarg, and NS Trudinger. *Elliptic partial differential equations of second order*, volume 224. Springer, 1977.

[58] Adrianna Gillman and Per-Gunnar Martinsson. A direct solver with $\mathcal{O}(n)$ complexity for variable coefficient elliptic pdes discretized via a high-order composite spectral collocation method. *SIAM Journal on Scientific Computing*, 36(4):A2023–A2046, 2014.

[59] Ron Goldman. Curvature formulas for implicit curves and surfaces. *Computer Aided Geometric Design*, 22(7):632–658, 2005.

[60] Tristan Goodwill and Michael O'Neil. On the numerical solution of the Laplace-Beltrami problem on piecewise-smooth surfaces. *arXiv e-prints*, pages arXiv–2108, 2021.

[61] Noella Grady. Functions of bounded variation. *Dostopno prek: https://www. whitman. edu/Documents/Academics/Mathematics/gra dy. pdf (Dostopano: 7.2. 2017)*, 2009.

[62] Axel Grundmann and Hans-Michael Möller. Invariant integration formulas for the n-simplex by combinatorial methods. *SIAM Journal on Numerical Analysis*, 15(2):282–290, 1978.

[63] Victor Guillemin and Alan Pollack. *Differential topology*, volume 370. American Mathematical Soc., 2010.

[64] Nicholas Hale and Lloyd N Trefethen. New quadrature formulas from conformal maps. *SIAM Journal on Numerical Analysis*, 46(2):930–948, 2008.

[65] Peter Hansbo, Mats G Larson, and Karl Larsson. Analysis of finite element methods for vector Laplacians on surfaces. *IMA Journal of Numerical Analysis*, 40(3):1652–1701, 2020.

[66] Hanne Hardering and Simon Praetorius. Tangential errors of tensor surface finite elements. *IMA Journal of Numerical Analysis*, 43(3):1543–1585, 2023.

[67] Behnam Hashemi and Lloyd N Trefethen. Chebfun in three dimensions. *SIAM Journal on Scientific Computing*, 39(5):C341–C363, 2017.

[68] Michael Hecht, Krzysztof Gonciarz, Jannik Michelfeit, Vladimir Sivkin, and Ivo F Sbalzarini. Multivariate interpolation in unisolvent nodes–lifting the curse of dimensionality. *arXiv preprint arXiv:2010.10824*, 2020.

[69] Michael Hecht, Karl B. Hoffmann, Bevan L Cheeseman, and Ivo F Sbalzarini. Multivariate Newton interpolation. *arXiv preprint arXiv:1812.04256*, 2018.

[70] Edwin Hewitt and Karl Stromberg. *Real and abstract analysis: a modern treatment of the theory of functions of a real variable*. Springer Science & Business Media, 2012.

[71] Christoph M Hoffmann. Implicit curves and surfaces in CAGD. *IEEE Computer Graphics and Applications*, 13(1):79–88, 1993.

[72] Hui Huang, Dan Li, Hao Zhang, Uri M. Ascher, and Daniel Cohen-Or. Consolidation of unorganized point clouds for surface reconstruction. *ACM SIGGRAPH Asia 2009 papers*, 2009.

[73] M Yousuff Hussaini and Thomas A Zang. Spectral methods in fluid dynamics. Technical report, 1986.

[74] Martin Hutzenthaler, Arnulf Jentzen, Thomas Kruse, Tuan Anh Nguyen, and Philippe von Wurstemberger. Overcoming the curse of dimensionality in the numerical approximation of semilinear parabolic partial differential equations. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 476(2244), December 2020.

[75] Tobin Isaac. Recursive, parameter-free, explicitly defined interpolation nodes for simplices. *SIAM Journal on Scientific Computing*, 42(6):A4046–A4062, 2020.

[76] Darae Jeong, Yibao Li, Yongho Choi, Minhyun Yoo, Dooyoung Kang, Junyoung Park, Jaewon Choi, and Junseok Kim. Numerical simulation of the zebra pattern formation on a three-dimensional model. *Physica A: Statistical Mechanics and its Applications*, 475:106–116, 2017.

[77] Tobias Jonsson, Mats G Larson, and Karl Larsson. Cut finite element methods for elliptic problems on multipatch parametric surfaces. *Computer Methods in Applied Mechanics and Engineering*, 324:366–394, 2017.

[78] Camille Jordan. Sur la series de Fourier. *CR Acad. Sci., Paris*, 92:228–230, 1881.

[79] Camille Jordan. *Cours d'analyse de l'École polytechnique*, volume 1. Gauthier-Villars et fils, 1893.

[80] Deepak Kapur and Yagati N Lakshman. *Elimination methods: An introduction*. State University of New York at Albany, Department of Computer Science, 1991.

[81] George Em Karniadakis, George Karniadakis, and Spencer Sherwin. *Spectral/hp element methods for computational fluid dynamics*. Oxford University Press, 2005.

[82] Balázs Kovács. Computing arbitrary Lagrangian Eulerian maps for evolving surfaces. *Numerical Methods for Partial Differential Equations*, 35(3):1093–1112, 2019.

[83] John M Lee. Riemannian manifolds, volume 176 of. *Graduate Texts in Mathematics*, 1997.

[84] John M Lee. Smooth manifolds. In *Introduction to smooth manifolds*, pages 1–31. Springer, 2013.

[85] Yipeng Li, Xinglin Zhao, Navamita Ray, and Xiangmin Jiao. Compact feature-aware

Hermite-style high-order surface reconstruction. *Engineering with Computers*, 37(1):187–210, 2021.

[86] Colin B Macdonald, Jeremy Brandman, and Steven J Ruuth. Solving eigenvalue problems on curved surfaces using the closest point method. *Journal of Computational Physics*, 230(22):7944–7956, 2011.

[87] Colin B Macdonald, Barry Merriman, and Steven J Ruuth. Simple computation of reaction–diffusion processes on point clouds. *Proceedings of the National Academy of Sciences*, 110(23):9209–9214, 2013.

[88] Colin B Macdonald and Steven J Ruuth. Level set equations on surfaces via the closest point method. *Journal of Scientific Computing*, 35(2):219–240, 2008.

[89] Philip K Maini, Kevin J Painter, and Helene Nguyen Phong Chau. Spatial pattern formation in chemical and biological systems. *Journal of the Chemical Society, Faraday Transactions*, 93(20):3601–3610, 1997.

[90] Dhairya Malhotra, Antoine J Cerfon, Michael O'Neil, and Evan Toler. Efficient high-order singular quadrature schemes in magnetic fusion. *Plasma Physics and Controlled Fusion*, 62(2):024004, 2020.

[91] Per-Gunnar Martinsson. A direct solver for variable coefficient elliptic pdes discretized via a composite spectral collocation method. *Journal of Computational Physics*, 242:460–479, 2013.

[92] Per-Gunnar Martinsson. *Fast direct solvers for elliptic PDEs*. SIAM, 2019.

[93] J.C. Mason. Near-best multivariate approximation by Fourier series, Chebyshev series and Chebyshev interpolation. *Journal of Approximation Theory*, 28(4):349–358, 1980.

[94] John C Mason and David C Handscomb. *Chebyshev polynomials*. Chapman and Hall/CRC, 2002.

[95] Tarek Poonithara Abraham Mathew. *Domain decomposition methods for the numerical solution of partial differential equations*. Springer, 2008.

[96] T. M. Mills and Simon Jeffrey Smith. The Lebesgue constant for Lagrange interpolation on equidistant nodes. *Numerische Mathematik*, 61:111–115, 1992.

[97] Frank WJ Olver. *NIST handbook of mathematical functions hardback and CD-ROM*. Cambridge university press, 2010.

[98] Sheehan Olver and Alex Townsend. A fast and well-conditioned spectral method. *siam REVIEW*, 55(3):462–489, 2013.

[99] Stanley Osher and Ronald P Fedkiw. *Level set methods and dynamic implicit surfaces*, volume 1. Springer New York, 2005.

[100] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. *Journal of Computational Physics*, 79(1):12–49, 1988.

[101] Art B. Owen. Multidimensional variation for quasi-Monte Carlo. In *Contemporary Multivariate Analysis And Design Of Experiments: In Celebration of Professor Kai-Tai Fang's 65th Birthday*, pages 49–74. World Scientific, 2005.

[102] A. Cengiz Öztireli, Gaël Guennebaud, and Markus H. Gross. Feature preserving point set surfaces based on non-linear kernel regression. *Computer Graphics Forum*, 28, 2009.

[103] Per-Olof Persson and Gilbert Strang. A simple mesh generator in MATLAB. *SIAM Review*, 46(2):329–345, 2004.

[104] Rodrigo B Platte, Lloyd N Trefethen, and Arno BJ Kuijlaars. Impossibility of fast stable approximation of analytic functions from equispaced samples. *SIAM review*, 53(2):308–318, 2011.

[105] Michael James David Powell. *Approximation theory and methods*. Cambridge university press, 1981.

[106] Simon Praetorius and Florian Stenger. Dune-CurvedGrid – a Dune module for surface parametrization. *Archive of Numerical Software*, page Vol. 1 No. 1 (2022), 2022.

[107] L.M.A. Pressley, A. Pressley, M. Chaplain, and J.F. Toland. *Elementary Differential Geometry*. Springer undergraduate mathematics series. Springer, 2001.

[108] Alfio Quarteroni and Alberto Valli. *Numerical approximation of partial differential equations*, volume 23. Springer Science & Business Media, 2008.

[109] Navamita Ray, Duo Wang, Xiangmin Jiao, and James Glimm. High-order numerical integration over discrete surfaces. *SIAM Journal on Numerical Analysis*, 50(6):3061–3083, 2012.

[110] JA Reeger, B Fornberg, and ML Watts. Numerical quadrature over smooth, closed surfaces. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 472(2194):20160401, 2016.

[111] J. Ruppert. A Delaunay refinement algorithm for quality 2-dimensional mesh generation. *Journal of Algorithms*, 18(3):548–585, 1995.

[112] Steven J Ruuth and Barry Merriman. A simple embedding method for solving partial differential equations on surfaces. *Journal of Computational Physics*, 227(3):1943–1961, 2008.

[113] Oliver Sander. *Dune—The distributed and unified numerics environment*, volume 140. Springer Nature, 2020.

[114] Robert I Saye and James A Sethian. Multiscale modeling of membrane rearrangement, drainage, and rupture in evolving foams. *Science*, 340(6133):720–724, 2013.

[115] Thomas W Sederberg and John P Snively. Parametrization of cubic algebraic surfaces. In *Proceedings on Mathematics of surfaces II*, pages 299–319, 1987.

[116] Thomas W Sederberg and Jianmin Zheng. Algebraic methods for computer aided geometric design. *Handbook of computer aided geometric design*, pages 363–387, 2002.

[117] Thomas Warren Sederberg. *Implicit and parametric curves and surfaces for computer aided geometric design*. Purdue University, 1983.

[118] James A Sethian. Theory, algorithms, and applications of level set methods for propagating interfaces. *Acta numerica*, 5:309–395, 1996.

[119] James A Sethian. Tracking interfaces with level sets: An" act of violence" helps solve evolving interface problems in geometry, fluid mechanics, robotic navigation and materials sciences. *American Scientist*, 85(3):254–263, 1997.

[120] James Albert Sethian. *Level set methods and fast marching methods: evolving interfaces in computational geometry, fluid mechanics, computer vision, and materials science*, volume 3. Cambridge university press, 1999.

[121] Aleksei Shadrin. Twelve proofs of the markov inequality. *Approximation theory: a volume dedicated to Borislav Bojanov*, pages 233–298, 2004.

[122] M. Spivak. *A Comprehensive Introduction to Differential Geometry*, volume 1. Publish or Perish Incorporated, 1999.

[123] Elias M Stein and Rami Shakarchi. *Complex analysis*, volume 2. Princeton University Press, 2010.

[124] John C Strikwerda. *Finite difference schemes and partial differential equations*. SIAM, 2004.

[125] AH Stroud. *Approximate calculation of multiple integrals: Prentice-Hall series in automatic computation*. Prentice-Hall (Englewood Cliffs, NJ), 1971.

[126] Gabor Szegö. Conformal mapping of the interior of an ellipse onto a circle. *The American mathematical monthly*, 57(7):474–478, 1950.

[127] Mark A Taylor and BA Wingate. A generalized diagonal mass matrix spectral element method for non-quadrilateral elements. *Applied Numerical Mathematics*, 33(1-4):259–265, 2000.

[128] T Wynn Tee and Lloyd N Trefethen. A rational spectral collocation method with adaptively transformed Chebyshev grid points. *SIAM Journal on Scientific Computing*, 28(5):1798–1811, 2006.

[129] Csaba D Toth, Joseph O'Rourke, and Jacob E Goodman. *Handbook of discrete and computational geometry*. CRC press, 2017.

[130] Alex Townsend and Lloyd N Trefethen. An extension of chebfun to two dimensions. *SIAM Journal on Scientific Computing*, 35(6):C495–C518, 2013.

[131] Lloyd N Trefethen. Numerical computation of the Schwarz–Christoffel transformation. *SIAM Journal on Scientific and Statistical Computing*, 1(1):82–102, 1980.

[132] Lloyd N. Trefethen. *Spectral Methods in MATLAB*. Society for Industrial and Applied Mathematics, 2000.

[133] Lloyd N. Trefethen. *Approximation theory and approximation practice*, volume 164. SIAM, 2019.

[134] Lloyd N Trefethen. Exactness of quadrature formulas. *Siam Review*, 64(1):132–150, 2022.

[135] Alan Mathison Turing. The chemical basis of morphogenesis. *Bulletin of mathematical biology*, 52:153–197, 1990.

[136] Richard S Varga. *Matrix iterative methods*. Prentice Hall Incorporated, 1962.

[137] Sachin K Thekke Veettil, Yuxi Zheng, Uwe Hernandez Acosta, Damar Wicaksono, and Michael Hecht. Multivariate polynomial regression of euclidean degree extends the stability for fast approximations of trefethen functions. *arXiv preprint arXiv:2212.11706*, 2022.

[138] Sachin Krishnan Thekke Veettil, Gentian Zavalani, Uwe Hernandez Acosta, Ivo F. Sbalzarini, and Michael Hecht. Global polynomial level sets for numerical differential geometry of smooth closed surfaces. *SIAM Journal on Scientific Computing*, 45(4):A1995–A2018, 2023.

[139] Bogdan Vioreanu and Vladimir Rokhlin. Spectra of multiplication operators as a numerical tool. *SIAM Journal on Scientific Computing*, 36(1):A267–A288, 2014.

[140] Tim Warburton. An explicit construction of interpolation nodes on the simplex. *Journal of engineering mathematics*, 56(3):247–262, 2006.

[141] Hassler Whitney. Elementary structure of real algebraic varieties. *Hassler Whitney Collected Papers*, pages 456–467, 1992.

[142] Hassler Whitney. *Geometric integration theory*. Courier Corporation, 2012.

[143] Shuhuang Xiang and Folkmar A. Bornemann. On the convergence rates of Gauss and Clenshaw-Curtis quadrature for functions of limited regularity. *SIAM J. Numer. Anal.*, 50:2581–2587, 2012.

[144] Hong Xiao and Zydrunas Gimbutas. A numerical algorithm for the construction of efficient quadrature rules in two and higher dimensions. *Computers & mathematics with applications*, 59(2):663–676, 2010.

[145] Gentian Zavalani and Michael Hecht. High-order numerical integration on regular embedded surfaces. In *European Conference on Numerical Mathematics and Advanced Applications*, pages 494–504. Springer, 2023.

[146] Gentian Zavalani, Oliver Sander, and Michael Hecht. High-order integration on regular triangulated manifolds reaches superalgebraic approximation rates through cubical reparametrizations. *SIAM Journal on Numerical Analysis*, 63(6):2454–2482, 2025.

[147] Gentian Zavalani, Elima Shehu, and Michael Hecht. A note on the rate of convergence of integration schemes for closed surfaces. *Computational and Applied Mathematics*, 43(2):1–17, 2024.

[148] Denis Zorin. Subdivision for modeling and animation. *ACM SIGGRAPH'00 Course Notes*, 2000.

Erklärung nach §5.5 der Promotionsordnung

Hiermit versichere ich, dass ich die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe; die aus fremden Quellen direkt oder indirekt übernommenen Gedanken sind als solche kenntlich gemacht. Die Arbeit wurde bisher weder im Inland noch im Ausland in gleicher oder ähnlicher Form einer anderen Prüfungsbehörde vorgelegt. Die Dissertation wurde im Zeitraum von 19.04.2021 bis 12.08.2025 am Center for Advanced Systems Understanding verfasst und durch Prof. Dr. Oliver Sander, Professur für Numerik partieller Differentialglei-chungen der TU Dresden, betreut. Ich erkläre hiermit, dass keine früheren erfolglosen Promotionsverfahren stattgefunden haben. Ich erkenne die Promotionsordnung der Fakultät für Mathematik und Naturwissenschaften der Technische Universität Dresden vom 23.02.2011 an.

Dresden, August 12, 2025 _____

Gentian Zavalani