NFN - Nationales Forschungsnetzwerk

Geometry + Simulation

http://www.gs.jku.at

Integration by Interpolation and Look-up for Galerkin-based Isogeometric Analysis

A. Mantzaflaris, B. Jüttler

G+S Report No. 14

June 2014

FWF Der Wissenschaftsfonds.

JKU JOHANNES KEPLER UNIVERSITY LINZ

Integration by Interpolation and Look-up for Galerkin-based Isogeometric Analysis

Angelos Mantzaflaris [a,*], Bert Jüttler [b]

[a] *Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Linz, Austria*

[b] *Institute of Applied Geometry, Johannes Kepler University, Linz, Austria*

Abstract

Even though isogeometric analysis has a clear advantage regarding the number of degrees of freedom needed to achieve a certain level of accuracy, the time needed for matrix assembly (by means of numerical integration of stiffness or mass matrix entries) constitutes a severe bottleneck in the process. In order to overcome this difficulty, we propose the new approach of Integration by Interpolation and Lookup (IIL). Firstly, applying spline interpolation to the common factors in the occurring integrals approximately transforms them into integrals of piecewise polynomial functions, whose integrands are expressed in tensor-product B-spline form. The common factors represent the influence of the geometry mapping (i.e., the NURBS domain parameterization) and the contributions of possibly non-constant material coefficients. Secondly, these integrals are evaluated exactly using pre-computed look-up tables for integrals of tri-products of univariate B-splines and their derivatives. For the model case of elliptic problems, we perform a theoretical analysis to demonstrate that the IIL method maintains the overall approximation order of the Galerkin discretization, provided that the spline interpolation is sufficiently accurate. Moreover, we provide a comparison of the computational complexity of our method with that of a standard Gauss quadrature method. Finally, we present experimental results to illustrate the performance of the IIL method and to support our theoretical results.

Key words: isogeometric analysis, stiffness matrix, mass matrix, numerical integration, quadrature

* Corresponding author.

Email addresses: Angelos.Mantzaflaris@oeaw.ac.at (Angelos Mantzaflaris), Bert.Juettler@jku.at (Bert Jüttler).

1 Introduction

Numerical simulation using Isogeometric Analysis (IGA), which was introduced by Hughes et al. [14,21], relies on NURBS-parameterizations of the computational (physical) domain. When solving (e.g.) a boundary value problem (BVP) via Galerkin projection, the numerical solution is obtained by projecting it into a finite-dimensional discretization space. The basis functions spanning the discretization space are defined using NURBS basis functions and the given domain parameterization. The coefficients of the numerical solution are found by solving a linear system. The latter is formed by mass and/or stiffness matrices and a load vector, possibly combined with advection and other terms. The matrix and right-hand side coefficients of the system have to be computed via numerical integration of piecewise rational multivariate functions. As a major difference to the classical finite element analysis (FEA), exact integration is almost never possible, not even for constant material coefficients, due to the influence of the NURBS domain parameterization (also called the geometry mapping).

Consequently, there are two major sources for numerical error which are equally important: the discretization error and the error introduced by using numerical integration (often called quadrature or cubature in the two- and three-dimensional case, respectively) to evaluate the coefficients and the right-hand side of the linear system. For small- to medium-sized problems, the remaining error source of numerically solving the resulting linear system can safely be neglected.

Similarly to FEA, the standard method for numerically evaluating the integrals in IGA is Gaussian quadrature. Recent research has focused on special quadrature rules that reduce the number of quadrature points and weights used in order to reduce the number of evaluations and to save computing time.

1.1 Related work

Quadrature rules for integrals involving B-splines were already considered more than 30 years ago. In [20] the authors derived rules for the moments of (linear, quadratic and cubic) B-splines, in order to solve a parabolic PDE using Galerkin's method. Since the advent of Isogeometric Analysis, there is an increased interest in this topic, due to its importance for the performance and competitiveness of isogeometric methods.

Optimal quadrature rules for the mass and stiffness of uniform B-spline discretizations are presented in [22]. More precisely, the authors derive rules with the minimum number of nodes that are exact for the product of two B-splines, for degree up to three. The rule is referred to as the half-point rule as the number of nodes (points) plus the number of weights in this minimal rule coincides with the dimension of the spline space of integrands. However, the optimal rule is defined over the whole domain of the B-spline space, and the computation of the nodes and weights leads to a global, non-linear system of equations, which is solved using a Newton iteration. Consequently, the applicability of this approach is limited to low degree and to small number of elements. The authors propose a macro-element strategy to extend the range of applicability to larger number of elements.

The spline space of the product of two uniform B-spline basis functions is further investigated in [2], in order to derive a feasible, computable rule. The basis functions are grouped with respect to the size of their support. This leads to a rule which is defined over one or two elements, and which can be obtained as the solution of a local non-linear system that is independent of the number of elements.

An experimental study of the Gauss rule, and the optimal rule on macro-elements of [22], as well as an extension to degree 4 has been presented in [27]. The authors perform experiments on a Poisson problem over a domain given by the identity mapping, with a unit Jacobian determinant. Their focus is on the degree of exactness of different rules as well as their practical computational cost. Since the parameterization is the identity, the shape functions are simply B-splines, therefore exact evaluation of the stiffness matrix is feasible using Gauss quadrature rules with sufficient degree of exactness.

The use of GPU programming for accelerating the assembly process in isogeometric analysis has been proposed recently by Karatarakis et al. [24,23] based on standard Gauss quadrature rules. In order to exploit the capabilities of modern graphics hardware, the authors present a suitable formulation for the stiffness matrix, which supports the parallelization of the assembly process.

A variety of reduced Bézier element quadrature rules for isogeometric discretizations of low degree have been investigated in [30]. By carefully placing quadrature nodes, preferably on element boundaries, a significant speed-up compared to Gauss quadrature is achieved, due to the reduced number of evaluations per element. As another interesting observation, 'reduced' Gauss quadrature (with p^d Gauss points per element, in dimension d and degree p) is found to provide sufficient accuracy to maintain the optimal order of convergence, both for the L^2 norm and the H^1 seminorm. For each of the discussed cases (quadratic and cubic spline discretizations of two- and three-dimensional domains), the authors provide recommendations for choosing appropriate quadrature rules with a small number of evaluations per element.

The aim of the recent preprint [10] is to derive quadrature rules for refinable functions, such as B-splines. The ideas presented there may be relevant for designing numerical integration techniques for IGA. A short review of various quadrature strategies in IGA was presented in [9].

Collocation has been proposed recently as an alternative to Galerkin projection, see [29] for a thorough comparison for the two approaches in IGA. The main advantage is the fact that only evaluation at the collocation points is required to assemble the respective collocation matrix, i.e., one evaluation per degree of freedom is needed. A disadvantage is the lack of theoretical results concerning the convergence properties of the method, as well as a sub-optimal experimental order of convergence (equal to $p-1$) for odd degrees.

This paper builds upon the approach that was outlined in the conference article [26]. Based on interpolation techniques, we presented a method for numerical quadrature in the case of elliptic problems in one-dimensional domains. In the meantime, we developed several theoretical results supporting the preliminary experimental observations. We also succeeded with the generalization of our approach to the multi-dimensional case. Moreover, we developed a C++ implementation of our method, which allows us to extend the experimental investigation to the multi-dimensional setting, to higher order

discretizations and to larger test problems.

1.2 Contributions and Outline

Any process for numerical integration can be regarded as a projection of the integrand to a function space where exact integration is possible. The approximation properties of the projection space induces the approximation power of the integration method. In a typical approach, such as Gauss quadrature, the entire integrand is projected. In our approach, we restrict the projection step to the non-polynomial part of the integrand, i.e., to the contribution of the geometry mapping and the coefficients of the PDE. The remaining components of the integrand are already in a piecewise polynomial space, therefore exact integration is feasible.

Once the projection has been performed, we use an exact method to evaluate the simplified integrals. Taking into account the properties of B-splines, in particular in the uniform case, we are able to construct small look-up tables which contain the values of integrals of B-spline tri-products, and in that way perform the assembly procedure by repeated look-up queries on these tables.

We shall demonstrate that the IIL method presented in this paper requires only one evaluation per degree of freedom, namely evaluation at the Greville points, for applying Galerkin IGA with uniform B-splines. As a highlight, the IIL method circumvents the element-wise assembly that is used in classical FEA. The direct adaptation of that classical approach leads to performance issues, due to the higher degree and larger support of the basis functions used in IGA. In particular, higher-order rules are required for sufficient approximation, resulting in a big number of quadrature points and consequently a big number of evaluations per basis function.

Also, even if a local assembly strategy is adopted, still a large number of access operations need to be performed in the sparse matrix data structure. This is due to the required accumulation of contributions from quadrature points in different elements, due to the increased support of the basis functions (compared to FEA). In contrast, our method is purely local, i.e., every entry of the matrix is accessed only once, and the final value is generated in only one step. This also makes the method fully parallel, since the granularity of the computation can be as small as a single matrix entry.

We represent the contributions of the geometry map *a priori* in terms of uniform B-splines, allowing to reduce the overall number of evaluations to one evaluation per physical element, without sacrificing the convergence rate. Consequently, our method restores the appeal of the Galerkin approach by making its performance similar to the recently established isogeometric collocation method [1,19,29].

The remainder of this paper consists of seven sections. First we introduce the model problem and the isogeometric discretization, and we recall the error sources in the numerical simulation pipeline. Section 3 presents a brief review of evaluation-based approaches for numerical quadrature in IGA. The fourth section introduces the new IIL method. It is followed by Sections 5 and 6, which analyze the convergence properties and the computational complexity of the new method respectively. Section 7 presents

implementation details and numerical examples that demonstrate the performance and support our theoretical results. Finally we conclude the paper.

2 Preliminaries

In this section we settle a model Poisson problem in order to facilitate the presentation and analysis of the IIL method. We derive a standard isogeometric discretization and identify the geometry factor in the stiffness matrix. Finally, we review standard tools for the error analysis of Galerkin methods for elliptic PDEs and we focus on the consistency error term in Strang's first lemma.

2.1 The model problem

In order to describe our method for assembling the matrices needed for isogeometric simulation, we will consider a homogeneous Dirichlet boundary value problem (BVP) for the diffusion equation, possibly with non-constant coefficients, which is defined over a domain of dimension d ($d \leq 3$). More precisely, we consider the problem of finding $u = u(\boldsymbol{x})$ satisfying

$$\begin{aligned} -\nabla \cdot (K \nabla u) &= f \quad \text{in } \Omega\,, \\ u &= 0 \quad \text{on } \partial\Omega \end{aligned} \tag{1}$$

where $K = K(\boldsymbol{x})$ is a symmetric and uniformly positive definite matrix (more precisely: a matrix-valued function) on Ω.

The variational form of the BVP (1) consists in finding

$$u \in V = H_0^1(\Omega) \text{ such that } \quad a(u,v) = \ell(v) \quad \forall v \in V = H_0^1(\Omega)\,, \tag{2}$$

with the bilinear and linear forms

$$a(u,v) = \int_\Omega \nabla u^T(\boldsymbol{x})\, K(\boldsymbol{x})\, \nabla v(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x} \text{ and } \ell(v) = \int_\Omega f(\boldsymbol{x}) v(\boldsymbol{x})\, \mathrm{d}\boldsymbol{x}\ , \tag{3}$$

respectively. The bilinear form $a(\cdot,\cdot)$ is symmetric, coercive, and bounded. Hence, according to the theory of abstract boundary value problems [6,8,11], the existence and uniqueness of the weak solution $u^* \in V$ is ensured.

2.2 Isogeometric discretization

In the setting of isogeometric analysis, the domain Ω is parameterized by a global geometry map $G : \hat{\Omega} \to \Omega$, where the parameter domain $\hat{\Omega}$ is an axis-aligned box in $\mathbb{R}^d$. More generally, one might consider a collection of such boxes for a multi-patch domain parameterization, but this is beyond the scope of the present paper.

Any point $\boldsymbol{x} \in \Omega$ in the physical domain is the image of a point $\hat{\boldsymbol{x}} \in \hat{\Omega}$ in the parameter domain,

$$\boldsymbol{x} = G(\hat{\boldsymbol{x}}) = \sum_{\boldsymbol{i} \in \mathcal{I}} \boldsymbol{g}_{\boldsymbol{i}} R_{\boldsymbol{i}}(\hat{\boldsymbol{x}}), \tag{4}$$

where $R_{\boldsymbol{i}}$ are NURBS basis functions, the coefficients $\boldsymbol{g}_{\boldsymbol{i}}$ are the control points, and $\mathcal{I}$ is a finite index set (typically an axis-aligned box in index space $\mathbb{Z}^d$). The basis functions are defined by user-specified knot vectors and weights.

The knot vectors define mesh hyperplanes (for $d = 2$: mesh lines) that subdivide the parameter domain into *elements*. Typically, one considers knot vectors that are open and uniform.

In most situations, the weights are all equal, except when one needs to exactly represent special shapes such as circles. In order to simplify the presentation, we will restrict the exposition to the *case of equal weights*, where the NURBS basis functions $R_{\boldsymbol{i},p}$ are simply tensor-product B-splines $T_{\boldsymbol{i},p}$, i.e., products of d univariate B-splines of degree p

$$R_{\boldsymbol{i},p}(\hat{\boldsymbol{x}}) = T_{\boldsymbol{i},p}(\hat{\boldsymbol{x}}) = \prod_{k=1}^{d} N_{i_k,p}(\hat{x}_k) \quad , \tag{5}$$

where we use multi-indices $\boldsymbol{i} = (i_1, \ldots, i_d)$ to index basis functions. The extension of our approach to general NURBS parameterizations is outlined in Remark 4. Note that we choose B-splines of the same degree p in all variables [1], even though this is not a requirement of the method, in order to simplify the notation.

We thus obtain piecewise polynomial functions of degree p in each variable, which have smoothness C^{p-1} across the mesh hyperplanes, provided that the knot vectors have only simple interior knots. For a more detailed introduction to spline theory and geometric modeling, the reader is referred to standard textbooks such as [17,16].

The isogeometric discretization takes advantage of the given parameterization of the domain Ω. The discretization space

$$V_h = \operatorname{span}\{\phi_{\boldsymbol{i}} \ : \ \boldsymbol{i} \in \mathcal{I}\} \text{ with } \phi_{\boldsymbol{i}} = T_{\boldsymbol{i},p} \circ G^{-1} \tag{6}$$

is spanned by push-forwards of the tensor-product B-splines. The mesh size h, which characterizes the discretization space, can be identified with the (maximum) element size. The Galerkin projection transforms the variational form (2) of the BVP into the discretized problem, which consists in finding

$$u_h \in V_h \text{ such that } \ a(u_h, v_h) = \ell(v_h), \quad \forall v_h \in V_h \, . \tag{7}$$

For any function which is defined on the physical domain, we use the hat $\hat{}$ to denote its

[1] By a slight abuse of notation, we use the *same* symbol $N_{i,p}$ to denote the univariate B-splines with respect to the d different variables, even though the knot vectors may be different. This could be made precise by referring to the associated knot vector and degree in the notation $N_{i,p,\tau}$. This information has been omitted since it is always clear from the context.

pull-back to the parameter domain, i.e.

$$\hat{u}_h = u_h \circ G, \quad \hat{f} = f \circ G, \quad \hat{\phi}_{\boldsymbol{i}} = \phi_{\boldsymbol{i}} \circ G.$$

We also use $\hat{\nabla}$ to denote the gradient operator in the parameter domain.

Using the geometry map we rewrite the two forms as

$$a(u_h, v_h) = \int_{\hat{\Omega}} \hat{\nabla} \hat{u}_h^T \, A \, \hat{\nabla} \hat{v}_h \, \mathrm{d}\hat{\boldsymbol{x}} \quad \text{and} \quad \ell(v_h) = \int_{\hat{\Omega}} \hat{f} \, \hat{v}_h \, |\det J| \, \mathrm{d}\hat{\boldsymbol{x}}, \tag{8}$$

where

$$A = |\det J| J^{-1} \, \hat{K} \, J^{-T} = \frac{J_c \, \hat{K} \, J_c^T}{|\det J|} = (a_{rs})_{r,s=1,\ldots,d} \ , \tag{9}$$

with the Jacobian matrix $J = \hat{\nabla} G$, its cofactor matrix $J_c = J^{-1} \det J$, and $\hat{K} = K \circ G$. The matrix $A = A(\hat{\boldsymbol{x}})$ is symmetric and uniformly positive definite. It involves the geometry map, its derivatives and the coefficients of the PDE.

The functions in the discretization space V_h are represented as linear combinations of the basis functions,

$$u_h = \sum_{\boldsymbol{i} \in \mathcal{I}} u_{\boldsymbol{i}} \, \phi_{\boldsymbol{i}} \quad , \tag{10}$$

with certain coefficients $\boldsymbol{u} = (u_{\boldsymbol{i}})_{\boldsymbol{i} \in \mathcal{I}}$. By considering the test functions $v_h = \phi_{\boldsymbol{i}}$, $\boldsymbol{i} \in \mathcal{I}$, in the variational form (7), one arrives at the linear system $S\boldsymbol{u} = \boldsymbol{b}$ characterizing the solution u_h^* with coefficients $\boldsymbol{u}^*$. The stiffness matrix S has the entries

$$S_{\boldsymbol{ij}} = a(\phi_{\boldsymbol{i}}, \phi_{\boldsymbol{j}}) = \int_{\hat{\Omega}} \hat{\nabla} T_{\boldsymbol{i},p}^T \, A \, \hat{\nabla} T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}} = \sum_{r=1}^{d} \sum_{s=1}^{d} \int_{\hat{\Omega}} a_{rs} \, \hat{\partial}_r T_{\boldsymbol{i},p} \, \hat{\partial}_s T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}}, \tag{11}$$

since $\hat{\phi}_{\boldsymbol{i}} = T_{\boldsymbol{i},p}$, where we denote the partial derivatives with $\hat{\partial}_r \hat{u} = (\partial \hat{u})/(\partial \hat{x}_r)$. Similarly we obtain the right-hand side (load vector)

$$b_{\boldsymbol{i}} = \ell(\phi_{\boldsymbol{i}}) = \int_{\hat{\Omega}} \hat{f} \, T_{\boldsymbol{i},p} \, |\det J| \, \mathrm{d}\hat{\boldsymbol{x}} \ . \tag{12}$$

Note that many quantities that appear in the discretization of variational forms of PDEs yield similar expressions. For example, the mass matrix entries take the form

$$M_{\boldsymbol{ij}} = \int_{\Omega} \phi_{\boldsymbol{i}} \, \phi_{\boldsymbol{j}} \, d\hat{\boldsymbol{x}} = \int_{\hat{\Omega}} T_{\boldsymbol{i},p} \, T_{\boldsymbol{j},p} \, |\det J| \, d\hat{\boldsymbol{x}} \ . \tag{13}$$

Analogous expressions arise when considering advection matrix entries or boundary integral terms coming from natural boundary conditions.

Example 1. Throughout the paper we shall use a 1D instance as an illustrative example, in particular

$$-u''(x) = f(x), \ x \in \Omega = (0, 10) \, ,$$

with $u(0) = u(10) = 0$ as boundary conditions. The domain Ω is parameterized with uniform quadratic B-splines (see Figure 1 for the control points). Certainly, this is an artificial construction, since an optimal choice would be a linear parameterization with constant speed. For the right-hand function

$$f(x) = \pi^2 \sin(\pi x),$$

the exact solution has the closed form $u(x) = \sin(\pi x)$ (shown as color-map in Figure 1). In this case, the geometry-related factor that appears in the stiffness matrix entries equals $A = \frac{1}{G'}$, and the determinant of the Jacobian evaluates to $|\det J| = |G'|$. $\diamond$

2.3 Well-posedness and numerical realization

We recall that the original problem (2)-(3), which involves the bilinear form $a : H_0^1(\Omega) \times H_0^1(\Omega) \mapsto \mathbb{R}$, has a unique solution $u^* \in H_0^1(\Omega)$. According to abstract finite element theory and the Lax-Milgram lemma, the discretized problem (7) possesses a unique solution $u_h^* \in V_h$. Its coefficients $u_{\boldsymbol{i}}^*$, $\boldsymbol{i} \in \mathcal{I}$ are found by solving the linear system of equations $S\boldsymbol{u} = \boldsymbol{b}$.

However, the stiffness matrix entries (11) have to be computed approximately. More precisely, the evaluation of the forms in (8) is subject to integration error. This can be interpreted as introducing modified forms $\tilde{a} : V_h \times V_h \mapsto \mathbb{R}$ and $\tilde{\ell} : V_h \mapsto \mathbb{R}$ and replacing the discretized problem (7) by

$$\text{Find } u_h \in V_h \text{ such that } \quad \tilde{a}(u_h, v_h) = \tilde{\ell}(v_h) \quad , \quad \forall\, v_h \in V_h \quad . \tag{14}$$

The existence and uniqueness of a solution of (14) is guaranteed if $\tilde{a}(\cdot,\cdot)$ has the property of uniform V_h-ellipticity, since the Lax-Milgram lemma (cf. [8], for instance) can then be applied.

More precisely, it is sufficient to verify that the bilinear form in (14) satisfies the inequality

$$\tilde{a}(v_h, v_h) \geq \mu |v_h|_{1,\Omega}^2 \quad \forall\, v_h \in V_h \ ,$$

where $|\cdot|_{1,\Omega}$ is the H^1-seminorm on Ω (which is here a norm due to the homogeneous Dirichlet boundary) and the constant μ is independent of the mesh size h. We will later verify this property for our specific construction of the approximate bilinear form $\tilde{a}$, which will be described in the following sections.

The relation between the solutions of the problem (14), which represents the actual numerical realization, and the original variational formulation (2) is described by the following result.

Lemma 2 (Strang's first lemma [32,33])**.** *Under the assumptions of boundedness and uniform V_h-ellipticity of $\tilde{a}(\cdot,\cdot)$, the solution $\tilde{u}_h$ of* (14) *satisfies*

$$\begin{aligned} |u^* - \tilde{u}_h|_{1,\Omega} \leq C \Bigg(& |u^* - u_h|_{1,\Omega} + \sup_{w_h \in V_h} \frac{|a(u_h^*, w_h) - \tilde{a}(u_h^*, w_h)|}{|w_h|_{1,\Omega}} \\ & + \sup_{w_h \in V_h} \frac{|\ell(w_h) - \tilde{\ell}(w_h)|}{|w_h|_{1,\Omega}} \Bigg) \ , \end{aligned} \tag{15}$$

where u^* stands for the exact solution of the variational problem (2), u_h^* is the solution of the discretized problem (7), u_h is the best approximation of u^* in V_h and the constant C is independent of h and f.

Consequently there are two major sources of error:

(1) The projection of the unknown solution into the finite-dimensional space V_h causes the *approximation error* $|u^* - u_h|_{1,\Omega}$.
(2) The numerical evaluation of the forms in (2) introduces the *consistency error* which is bounded by the remaining two terms on the right-hand side in (15).

Compared to the influence of these two error sources, the remaining ones have only a minor effect on the quality of the numerical solution. These include

(3) the *geometry approximation error* for the domain representation, which can be assumed to be zero for isogeometric discretizations,
(4) the error which is due to approximating Dirichlet boundary data (not present for the homogeneous problem (1), but it appears when homogenizing a non-homogeneous problem, cf. [13]), and
(5) the inaccuracy introduced when numerically solving the linear system.

The approximation error[2] has order p with respect to the H^1 seminorm, when B-splines of degree p are used [3,4]. Consequently, this is the optimal convergence rate that we expect to obtain. The aim of this paper is to present an efficient numerical integration procedure, tailored towards IGA, that provably preserves the overall convergence rate by carefully controlling the order of the consistency error.

3 Review of evaluation-based numerical integration

To make the link between the IIL method and quadrature approaches, we may regard numerical integration using a certain quadrature rule as a projection operation to a (typically polynomial) space that is integrated exactly by that rule. Therefore the error in the numerical integration is equal to a best approximation error in the underlying projection space. In this section we review quadrature schemes relevant to IGA, notably providing exact integration in $\mathbb{S}^p_{-1}$ and even $\mathbb{S}^{2p}_{p-1}$.

3.1 Quadrature schemes

A large class of quadrature rules can be derived by constructing interpolating functions which are "easy" to integrate, for instance polynomials.

Using a quadrature formula, the computed bilinear form is a weighted sum of evaluations

$$\tilde{a}(u_h, v_h) = \sum_i \sum_{j=1}^{\varrho^d} w_{i,j}\, \hat{\nabla}\hat{u}_h(\hat{\boldsymbol{x}}_{i,j})^T A(\hat{\boldsymbol{x}}_{i,j}) \hat{\nabla}\hat{v}_h(\hat{\boldsymbol{x}}_{i,j}) \ , \tag{16}$$

[2] Note that by Céa's Lemma, the overall *discretization error* $|u^* - u_h^*|_{1,\Omega}$ is of the same order of magnitude as the approximation error $|u^* - u_h|_{1,\Omega}$.

where the first sum runs over all elements (indexed by i). The symbols $w_{i,j}$ and $\hat{\boldsymbol{x}}_{i,j}$ stand for the quadrature weights and points on the i-th element, respectively. In the case of a tensor-product Gauss rule, the coordinates of the nodes $\hat{\boldsymbol{x}}_{i,j}$ are roots of Legendre polynomials of degree ϱ. Such a rule corresponds to the tensor product of d univariate rules with ϱ nodes in each direction. The rule is defined in the parent element $[-1,1]^d$ and mapped to the parametric (and consequently to the physical) integration element by a coordinate transformation. The weights follow from a simple formula.

We define the coordinate-wise degree of exactness (or algebraic precision) of a rule to be equal to r, if all polynomial functions of degree at most r with respect to each variable are integrated exactly by the rule. If the integrand is sufficiently smooth, then applying a rule of exactness r guarantees an integration error of order $r+1$. A formula of the form (16) can be chosen to be exact on a polynomial space of degree r if it has at least $r+1$ "quadrature degrees of freedom", i.e. $2\varrho \geq r+1$. The optimal choice with respect to the degree of exactness is the Gauss rule; it provides the unique rule with ϱ nodes that is exact on the space $\bigotimes_{i=1}^{d} \mathbb{P}^{2\varrho-1}$ of tensor-product polynomials of degree $2\varrho-1$.

In the classical FEM literature (but also in IGA) it well known that when $\varrho = p+1$ Gauss nodes are used for elements of polynomial degree p, optimal convergence rates are obtained. Proving this property requires to bound both the discretization and the consistency error. Bounding the latter uses assumptions on the exactness of the rule on polynomials of certain degree and the Bramble-Hilbert lemma, which provides bounds on the error of approximating a function by a polynomial of given degree [7,11].

In the context of IGA, (reduced) quadrature has been studied e.g. in [30]. Their experimental data indicate that using Gauss rules with p points in each coordinate direction suffices to obtain the optimal order of convergence in the H^1 and L^2 norms. In fact, this is supported by the following discussion of the order of the first consistency term in the 1D case:

Example 3. We continue the discussion of the one-dimensional Example 1. For a uniform B-spline discretization with mesh size h and using an integration rule which is exact for polynomials of degree q, the order of the first consistency error term in (15) for test functions $w_h = \phi_i = N_{i,p} \circ G^{-1}$ is $\mathcal{O}(h^{q-p+5/2})$, where we use the $\mathcal{O}$-notation to express the asymptotic behavior for $h \to 0$. On the one hand, the integration error in the bilinear form

$$a(u_h^*, \phi_i) = \int_0^1 \frac{\hat{u}_h^{*\prime}}{G'} N_{i,p}' \mathrm{d}\hat{x}$$

has the same asymptotic behavior as its upper bound

$$\underbrace{(p+1)h}_{\text{length of supp}(N_{i,p})} \cdot \underbrace{h^{q+1} \max_j \| \left(\frac{\hat{u}_h^{*\prime}}{G'} N_{i,p}' \right)^{(q+1)} \|_{\infty,j}}_{\text{interpolation error on } j\text{-th element (knot span)}}$$

where $\|.\|_{\infty,j}$ is the maximum norm on the j-th element (knot span). We consider the norm of the derivative of order $q+1$ of the integrand on each element, where $q \geq p-2$,

and obtain

$$\| \left(\frac{\hat{u}_h^{*\prime}}{G'} N'_{i,p} \right)^{(q+1)} \|_{\infty,j} = \| \ldots + \binom{q+1}{p-1} \left(\frac{\hat{u}_h^{*\prime}}{G'} \right)^{(q-p+2)} (N_{i,p})^{(p)} \|_{\infty,j} = \mathcal{O}(h^{-p})$$

since the first term converges to a function with bounded derivatives (as u_h^* can be assumed to approximate both the exact solution and its first p derivatives) and $N_{i,p}^{(p)} = \mathcal{O}(h^{-p})$. Consequently, the integration error satisfies

$$a(u_h^*, \phi_i) - \tilde{a}(u_h^*, \phi_i) = \mathcal{O}(h^{q-p+2})$$

On the other hand, the denominator in the first consistency error term satisfies

$$|\phi_i|_{1,\Omega} = \sqrt{\int_0^1 (N'_{i,p})^2 \frac{1}{G'} \mathrm{d}\hat{x}} = \mathcal{O}(h^{-\frac{1}{2}})$$

where we took into account that $N'_{i,p} = \mathcal{O}(h^{-1})$ and that the support of the integrand is contained in $p+1$ elements of length h. Combining these two observations confirms the order of the first consistency error term. In order to obtain at least the same order $\mathcal{O}(h^p)$ as the discretization error, the degree of exactness needs to satisfy $q \geq 2p - \frac{5}{2}$, which is guaranteed by using Gauss rules with at least p points. This observation [3] is confirmed by the numerical experiments reported in Figure 1. When using a Gauss rule of exactness $q = 2e - 1$, where e is the number of evaluation points per knot span (element), the rate of convergence with respect to the H^1 seminorm is experimentally found to be $\min(p, q - p + 5/2) = \min(p, 2e - p + 3/2)$. In particular, for degree p and using p, $p-1$, $p-2$ Gauss nodes, the respective experimental orders of convergence are p, $p - 1/2$, $p - 5/2$. All tests were performed with quadruple precision arithmetic (long double) for increased accuracy. Nevertheless we reached the limits of machine precision for finer discretizations and larger degrees. In fact, when using p and $p-1$ Gauss nodes, the difference $1/2$ between the convergence rates could be observed only for degrees up to 4. ◇

A number of different quadrature rules in isogeometric analysis have been proposed in the literature. Figure 2 provides a comparison of the evaluation points needed for a Gauss rule with $p+1$ points per direction (a), with the IGA-specific rules described in [22] (b) and [2] (c). The new method described in this paper requires solely the evaluation at the interpolation grid shown in (d). However, it should be noted that computational costs of the new method cannot be characterized by the number of evaluations only, and therefore a comparison with the existing techniques based only on this number would be meaningless. A more detailed comparison, which also addresses the issue of computational complexity, is provided in sections 6 and 7.

[3] Strictly speaking, the choice of the specific test function ϕ_i generates a lower bound for the upper bound of the error that is provided by Strang's lemma. However, the upper bound of Strang's lemma has the correct asymptotic behavior, and one may expect that the same is true when considering the specific test function. This is confirmed by our numerical experiments.

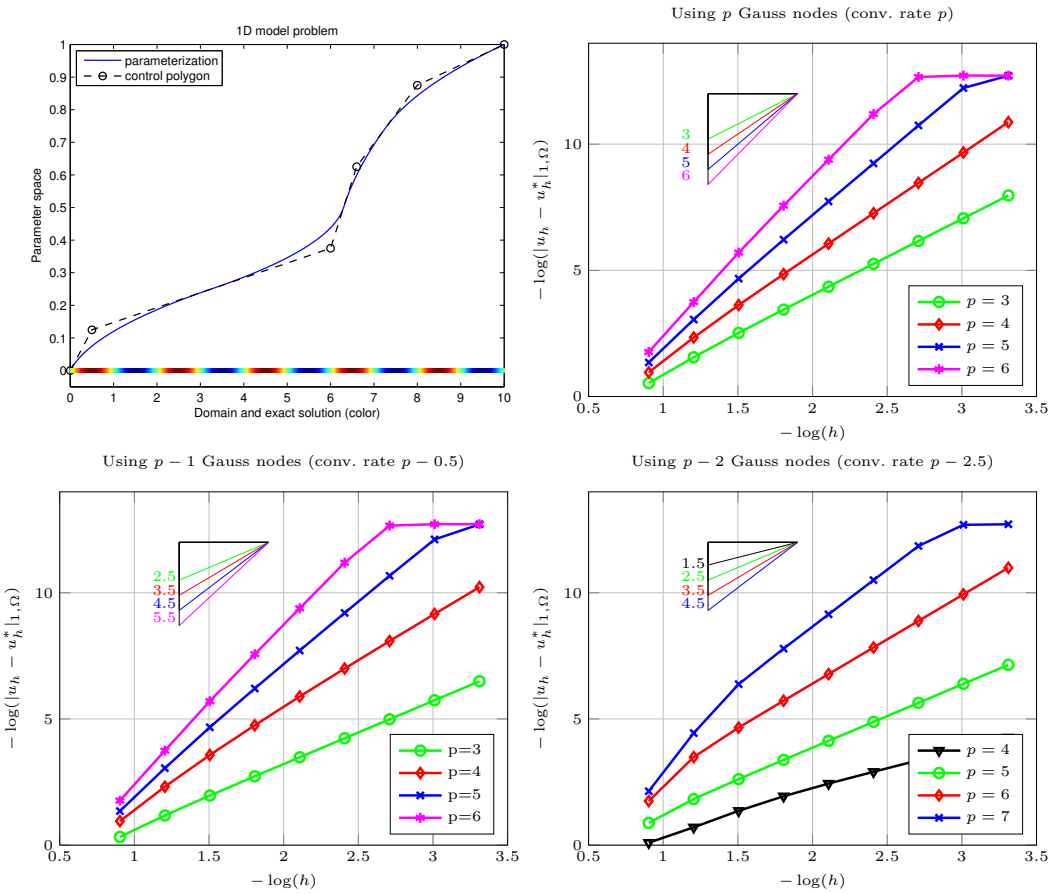


Fig. 1. Error plots and convergence rates obtained by using different Gauss quadrature rules in Examples 1 and 3.

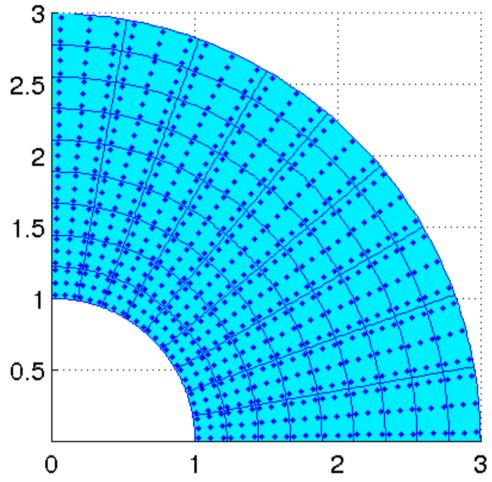
4 Integration by interpolation and lookup (IIL)

We present a quadrature-free method for the matrix assembly in isogeometric analysis. The quadrature will be replaced by an approximation step, by means of interpolation or quasi-interpolation, of those parts of the integrands (11), (12) or (13) that contain the geometry mapping G, its partial derivatives and possibly the coefficients of the PDE. Subsequently we use exact integration via look-up tables for evaluating the entries of stiffness matrix and load vector. This strategy allows to reduce the number of evaluations compared to quadrature-based approaches.

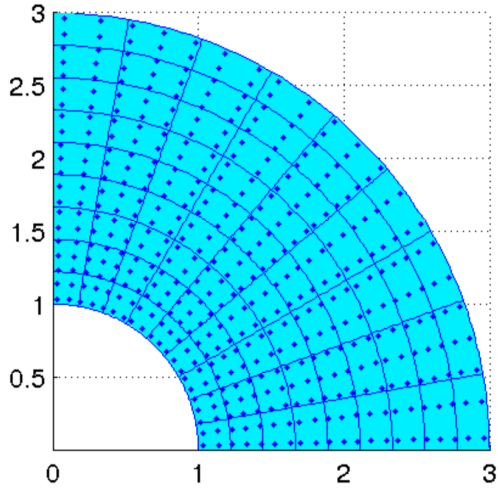
4.1 Outline

The integrals in question (e.g. each of the summands contributing to the double sum in (11)) are sums of expressions possessing the general form

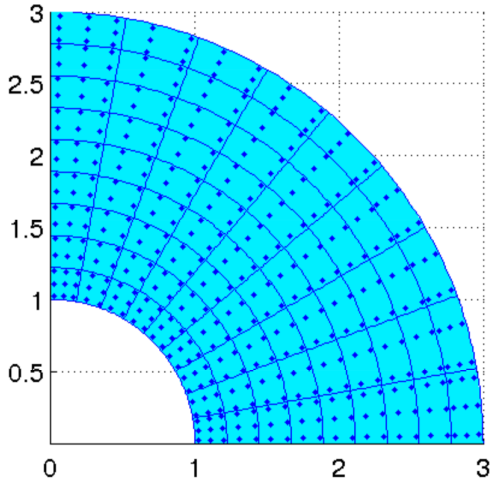
$$\int_{\hat{\Omega}} \Phi(\hat{\boldsymbol{x}}) \Psi_h(\hat{\boldsymbol{x}}) \mathrm{d}\hat{\boldsymbol{x}} \ . \tag{17}$$



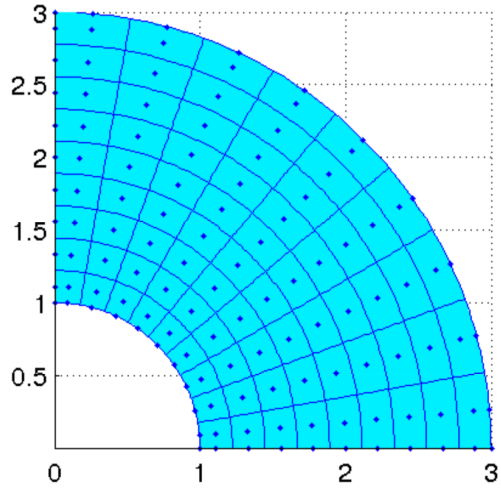
(a) Gauss Quadrature with 3 nodes per direction (729 nodes).



(b) Half-point macro-element rule (cf. [22]), 441 nodes.



(c) Periodic quadrature rule (cf. [2]), 400 nodes.



(d) Interpolation grid (Greville nodes), 121 nodes.

Fig. 2. Evaluation points for different numerical integration rules, for a 9×9 quadratic B-spline discretization of a quarter annulus. The new approach uses the grid in (d), but has a higher cost per evaluation point than the other approaches.

The first factor Φ depends on the coefficients of the partial differential equation and on the geometry mapping, but not on the mesh (more precisely, not on the knot vectors) used to obtain the discretization. Moreover, the same factor Φ is shared by many integrals. For our model problem and its discretization with non-rational splines, this factor occurs d^2 times per entry when assembling the stiffness matrix. In contrast, the second factor Ψ_h, which is a product of univariate B-splines and derivatives thereof, is independent of geometry mapping and coefficients.

The difficulties for integration are caused only by the factor Φ. In particular, the latter is generally non-polynomial and, even though it is rational in many cases, its degree can be quite large. Since it contains complicated expressions (e.g. the Jacobian

determinant) it cannot be expressed in terms of refined basis functions. In contrast, the factor Ψ_h has a very regular structure. Indeed, when performing uniform $h-$refinement, this factor contains recurring expressions of scaled B-spline basis functions. However, this structure cannot be exploited, due to the presence of the first factor, Φ.

We propose to replace Φ that appears in the integrand by a suitable approximation, followed by an exact and fast computation of the resulting integrals. We shall demonstrate that these integrals are linear combinations of a certain number of recurring *elementary integrals* that can be retrieved by use of a relatively small (degree-dependent) number of look-up operations. We note that the values of the lookup tables are exact, since the integrands are piecewise polynomials.

Consequently, the core idea of our method is to replace (8) by the approximate bilinear form

$$\tilde{a}(u_h, v_h) = \int_{\hat{\Omega}} \hat{\nabla}\hat{u}_h \, \tilde{A} \, \hat{\nabla}\hat{v}_h \, \mathrm{d}\hat{\boldsymbol{x}} \quad , \tag{18}$$

where $\tilde{A}$ is an approximation of the matrix A defined in (9), that allows for a simple closed-form evaluation of the induced stiffness matrix,

$$\tilde{S}_{\boldsymbol{ij}} = \tilde{a}(\phi_{\boldsymbol{i}}, \phi_{\boldsymbol{j}}) = \int_{\hat{\Omega}} \hat{\nabla} T_{\boldsymbol{i},p}^T \, \tilde{A} \, \hat{\nabla} T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}} = \sum_{r=1}^{d} \sum_{s=1}^{d} \int_{\hat{\Omega}} \tilde{a}_{rs} \, \hat{\partial}_r T_{\boldsymbol{i},p} \, \hat{\partial}_s T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}}. \tag{19}$$

In this case, each stiffness matrix entry is the sum of d^2 expressions of the form (17).

Suitable approximations $\tilde{a}_{rs}$ of the entries of A (which correspond to the factor Φ in (17)) will be found by means of interpolation or quasi-interpolation. Subsequently we will exploit the periodic nature of uniform B-spline basis functions in order to obtain compact lookup tables for B-spline tri-products.

Remark 4. General NURBS basis functions take the form

$$R_{\boldsymbol{i}}(\hat{\boldsymbol{x}}) = \frac{T_{\boldsymbol{i},p}(\hat{\boldsymbol{x}})}{\omega(\hat{\boldsymbol{x}})}$$

with a common denominator (or weight function) ω, which is typically provided by the input geometry. If these functions are used to define the discretization space V_h, i.e., $\phi_{\boldsymbol{i}} = R_{\boldsymbol{i}} \circ G^{-1}$, this denominator stays invariant under $h-p-k$ refinement. In particular, the stiffness matrix entries (11) are

$$\begin{aligned} S_{\boldsymbol{ij}} = a(\phi_{\boldsymbol{i}}, \phi_{\boldsymbol{j}}) &= \int_{\hat{\Omega}} \hat{\nabla} R_{\boldsymbol{i}}^T \, A \, \hat{\nabla} R_{\boldsymbol{j}} \, \mathrm{d}\hat{\boldsymbol{x}} \\ &= \int_{\hat{\Omega}} \hat{\nabla} T_{\boldsymbol{i},p}^T \, \frac{1}{\omega^2} \, A \, \hat{\nabla} T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}} - \int_{\hat{\Omega}} \hat{\nabla} T_{\boldsymbol{i},p}^T \, T_{\boldsymbol{j},p} \, \frac{1}{\omega^3} \, A \, \hat{\nabla}\omega \, \mathrm{d}\hat{\boldsymbol{x}} \\ &- \int_{\hat{\Omega}} \hat{\nabla}\omega^T \, \frac{1}{\omega^3} \, A \, T_{\boldsymbol{i},p} \, \hat{\nabla} T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}} + \int_{\hat{\Omega}} \hat{\nabla}\omega^T \, \frac{1}{\omega^4} \, A \, \hat{\nabla}\omega T_{\boldsymbol{i},p} \, T_{\boldsymbol{j},p} \, \mathrm{d}\hat{\boldsymbol{x}}. \end{aligned}$$

Each of these four terms is again a sum of expressions of the general form (17). In fact, each stiffness matrix entry is now the sum of $(d+1)^2$ expressions of this type. In order to utilize the IIL method to NURBS, we can apply it to each of these expressions. $\diamond$

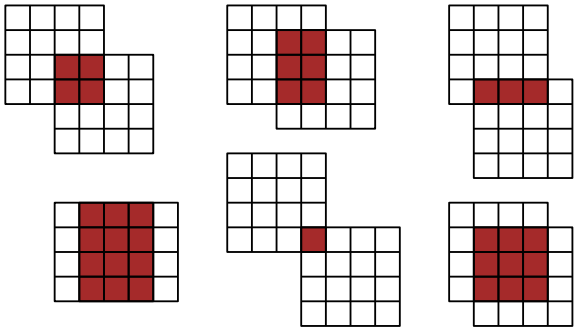


Fig. 3. Different configurations for the support intersection of *two* cubic B-splines. The integrals in (21), corresponding to one look-up operation, are nonzero only for tensor B-splines $T_{\boldsymbol{k},q}$ which possess a non-empty intersection with the filled region.

4.2 The algorithm

We describe our construction in the case of the stiffness term (11). The remaining terms can be dealt with similarly.

As before we consider a parameter domain of dimension d, and we assume a uniform degree p for all variables.

1. First we approximate the matrix A by projecting it into the tensor-product spline space $\bigotimes_{t=1}^{d} \mathbb{S}_s^q$ containing functions of degree q and smoothness s. Applying an interpolation or quasi-interpolation operator Π_h to A, we obtain

$$\tilde{A}(\hat{\boldsymbol{x}}) = \Pi_h A(\hat{\boldsymbol{x}}) = \sum_{\boldsymbol{k}\in\mathcal{I}} \tilde{A}_{\boldsymbol{k}} T_{\boldsymbol{k},q}(\hat{\boldsymbol{x}}) \tag{20}$$

   with the coefficient matrices $\tilde{A}_{\boldsymbol{k}} = (\, a_{rs}^{\boldsymbol{k}} \,)_{r,s=1,\dots,d}$. Consequently, the stiffness matrix entries defined in (19) are given by integrals of piecewise polynomial functions. This is the only approximation step that we need in the entire process.
2. Using (20) and (19) we can break down the expression of the stiffness matrix entries to a sum of elementary integrals of (derivatives of) tensor-product B-splines,

$$\tilde{S}_{\boldsymbol{ij}} = \sum_{\boldsymbol{k}\in\mathcal{I}} \int_{\hat{\Omega}} \hat{\nabla} T_{\boldsymbol{i},p}^T \, \tilde{A}_{\boldsymbol{k}} \, \hat{\nabla} T_{\boldsymbol{j},p} \, T_{\boldsymbol{k},q} \, d\hat{\boldsymbol{x}} = \sum_{\boldsymbol{k}\in\mathcal{I}} \sum_{r,s=1}^{d} a_{rs}^{\boldsymbol{k}} \int_{\hat{\Omega}} \hat{\partial}_r T_{\boldsymbol{i},p} \, \hat{\partial}_s T_{\boldsymbol{j},p} \, T_{\boldsymbol{k},q} \, d\hat{\boldsymbol{x}} \; . \tag{21}$$

   Due to the local support of B-splines, most terms of this sum vanish, cf. Fig. 3. Non-zero terms occur only if supports of $T_{\boldsymbol{k},q}$, $T_{\boldsymbol{i},p}$ and $T_{\boldsymbol{j},p}$ possess a non-empty intersection.
3. In order to perform exact integration of the piecewise polynomial integrands in (19), we construct (or load) a look-up table for the elementary integrals of the form

$$L_{ijkt}^{\alpha\beta} = \int_{\mathbb{R}} N_{i,p}^{(\alpha)}(\hat{x}_t) N_{j,p}^{(\beta)}(\hat{x}) N_{k,q}(\hat{x}_t) \mathrm{d}\hat{x}_t \; , \tag{22}$$

   where $N_{i,p}$, $N_{j,p}$ and $N_{k,q}$ are uniform B-splines of degree q and p and the upper indices $\alpha, \beta \in \{0, 1\}$ denote the order of differentiation. We will assume that the

spline spaces used for the projection in (20) and for the isogeometric discretization possess the same inner knots (possibly with different multiplicities). In this situation, the number of entries in the look-up tables is relatively small and independent of h. The structure of this look-up table will be analyzed later.

The definition (5) of the tensor-product B-splines implies the identity

$$\int_{\hat{\Omega}} \hat{\partial}_r T_{\boldsymbol{i},p}\, \hat{\partial}_s T_{\boldsymbol{j},p}\, T_{\boldsymbol{k},q}\, d\hat{\boldsymbol{x}} = \prod_{t=1}^{d} L_{i_t,j_t,k_t,t}^{\delta(r,t),\delta(s,t)} = L_{rs}^{\boldsymbol{ijk}} \tag{23}$$

where $\delta(\cdot,\cdot)$ is the Kronecker delta. Consequently, we are able to compute $\tilde{S}_{\boldsymbol{ij}}$ from (21) as a weighted sum of contributions taken from the look-up table; the weights are the entries $a_{rs}^{\boldsymbol{k}}$ of the coefficient matrices $\tilde{A}(\hat{\boldsymbol{x}})$ in (20). More precisely

$$\tilde{S}_{\boldsymbol{ij}} = \sum_{\boldsymbol{k}\in\mathcal{I}} \tilde{A}_{\boldsymbol{k}} : L^{\boldsymbol{ijk}}\ , \tag{24}$$

where ':' stands for the Frobenius inner product.

Similar constructions are available for the mass matrix (by projecting $|\det J|$ to the spline space) and the load vector (applying Π_h to $f \circ G$ and using look-up tables for products of two functions).

Summing up, we obtain the approximate bilinear form (18) given by

$$\tilde{a}(u_h, v_h) = \int_{\hat{\Omega}} \hat{\nabla}\, \hat{u}_h \Pi_h A\, \hat{\nabla} \hat{v}_h \,\mathrm{d}\hat{\boldsymbol{x}} \tag{25}$$

and the approximate linear form

$$\tilde{\ell}(v_h) = \int_{\hat{\Omega}} \Pi_h(\hat{f}\,|\det J|)\ \hat{v}_h\ \mathrm{d}\hat{\boldsymbol{x}}.$$

When applied to functions from the isogeometric discretization space V_h, these forms can be evaluated exactly.

4.3 Approximating the matrix A

Similarly to quadrature-based approaches, our construction takes advantage of the tensor product structure. The interpolation or quasi-interpolation operator is defined for univariate spline spaces and is then applied coordinate-wise for higher dimensions.

More precisely, we apply an interpolation or quasi-interpolation operator Π_h to each entry of $A(\hat{\boldsymbol{x}})$, which is constructed as the tensor-product of d univariate operators. Consequently, the computation of the (quasi-) interpolant can be performed coordinate-wise.

There are two sequences of knots involved in an isogeometric discretization: On the one hand, there are the initial knots needed to define the geometry map. We will denote them as *geometry knots*. On the other hand, there are additional knots inserted via h-refinement in order to obtain a sufficiently fine discretization space. Typically, the

geometry knots form the coarsest mesh for analysis, therefore they are much fewer than the new knots coming from h-refinement.

We note that the geometry map and its derivatives are C^∞ smooth except for the mesh hyperplanes defined by the geometry knots. When performing h-refinement, new knots are inserted. These knots, however, do not introduce additional discontinuities of the geometry map or if its derivatives.

In order to simplify the presentation we shall assume that the matrix K is also C^∞ smooth. Under this assumption, the matrix A defined in (9) is $C^{p-1-\mu}$-smooth at all mesh hyperplanes defined by geometry knots with multiplicity μ, and it is C^∞ smooth elsewhere.

We will need the following assumption to guarantee that the IIL method preserves the overall approximation order of the scheme. This assumption states that the interpolation or quasi-interpolation operator Π_h realizes the optimal order of approximation, which is possible in the spline space $\bigotimes_{t=1}^d \mathbb{S}_s^q$.

Assumption 5. *The interpolation or quasi-interpolation operator Π_h in (20) provides an entry-wise error bound ε_h,*

$$\max_{\hat{\boldsymbol{x}}\in\hat{\Omega}} \|A(\hat{\boldsymbol{x}}) - \tilde{A}(\hat{\boldsymbol{x}})\|_{\max} = \max_{\hat{\boldsymbol{x}}\in\hat{\Omega}} \max_{r,s=1,\dots,d} |a_{rs}(\hat{\boldsymbol{x}}) - \tilde{a}_{rs}(\hat{\boldsymbol{x}})| \leq \varepsilon_h = h^{q+1} C_A \tag{26}$$

where the constant C_A depends on the geometry map and on the matrix-valued function K, but is independent of h.

We describe two possibilities to construct an operator that satisfies this assumption. The first approach is based on quasi-interpolation operators and leads to theoretical guarantees, since it can be supported by classical results from spline approximation theory. The second approach relies on a simple interpolation technique and has been used in our implementation.

The *first approach* starts by splitting the geometry map (and hence the matrix A) into its C^∞-smooth segments. More precisely, the multiplicity of all the (interior) geometry knots is raised to p by knot insertion. This procedure allows us to subdivide the geometry map into its Bézier patches. The number of these patches is independent of h, since the new knots introduced by h-refinement do not introduce additional discontinuities in A.

Then we construct quasi-interpolation operators for the spline spaces obtained after inserting the additional knots (the non-geometry ones) into the Bézier patches of the geometry map. This is done by considering the tensor-products of d univariate quasi-interpolation operators. According to classical results from spline approximation theory, each of these operators satisfies Assumption 5 under certain conditions concerning the spacing of the knots that are introduced by h-refinement (which in particular are always satisfied for uniform refinement). See [16,31] for more information.

The *second approach*, which is simpler to implement, proceeds by interpolation. The spline space $\bigotimes_{t=1}^d \mathbb{S}_s^q$, which is used in the first step of the procedure described in Section 4.2, is defined using the knots inserted by h-refinement, each with multiplicity 1, and by the knots which are needed by the geometry map. The multiplicity of these knots is chosen to match the smoothness of the matrix A. More precisely, if a knot has multiplicity

μ in the geometry map, then, the corresponding knot for the interpolation spline space is chosen to have multiplicity $\max(1, 1+\mu+q-p)$. By this construction, the regularity of the interpolating spline space matches the regularity of the entries of A.

It is generally observed that the Greville points provide a practically optimal choice as interpolation sites, see [16]. The approximation of the matrix A is therefore constructed by tensor-product spline interpolation at the Greville abscissas. We compute one sparse LU factorization for the collocation matrix of every coordinate basis $N_{i,q}(\hat{x}_t)$ with respect to the Greville abscissas. The coefficients of the tensor-product interpolant are obtained by iteratively evaluating intermediate coefficients via back-substitution, until all dimensions are exhausted (cf. [5,15]).

It is possible to combine the subdivision into Bézier patches of the geometry map with the interpolation procedure in the second approach, and this would even provide theoretical guarantees for certain degrees and under certain assumptions regarding the knot spacing, see [16]. A more detailed investigation of the approximation power of interpolating splines for functions with reduced regularity is is beyond the scope of the present paper.

4.4 Building the lookup tables

We consider B-splines $\tilde{N}_{i,p}$, $\tilde{N}_{k,q}$ over the bi-infinite sequence $\mathbb{Z}$ of integer knots, where the index identifies the leftmost knot, and we define the normalized elementary integrals

$$I_{ijk}^{\alpha\beta} = \int_{\mathbb{R}} \tilde{N}_{i,p}^{(\alpha)}(\tilde{x}) \tilde{N}_{j,p}^{(\beta)}(\tilde{x}) \tilde{N}_{k,q}(\tilde{x}) \mathrm{d}\tilde{x}, \tag{27}$$

where the upper indices $\alpha, \beta \in \{0, 1\}$ indicate the order of differentiation. These integrals satisfy the identities

$$I_{ijk}^{\alpha\beta} = I_{jik}^{\beta\alpha} = I_{i+s,j+s,k+s}^{\alpha\beta},$$

where $s \in \mathbb{Z}$ is an index shift. Consequently, swapping the first two indices if $j < i$ and choosing $s = -\min(i, j, k)$ transforms them to $I_{i'j'k'}^{\alpha'\beta'}$ where either

(a) $i' = 0$, $0 \le j', k' \le p$ and $\alpha', \beta' \in \{0, 1\}$, or
(b) $k' = 0$ and $1 \le i' \le j' \le q$ and $\alpha', \beta' \in \{0, 1\}$.

These values of the normalized elementary integrals are generated off-line and stored in a look-up table.

In order to deal with the multiple knots at the boundaries, we extend the definition (27) by introducing another index m that specifies the multiplicity of the leftmost (for positive values of m) or rightmost knot (for negative values of m) of the combined knot sequence defining the three B-splines in the integral (27), e.g. if $i = 0$ the knots are

$$(\underbrace{0, \ldots, 0}_{m \text{ times}}, 1,\ 2,\ 3,\ \ldots)\ .$$

For each of the cases (a) and (b), we need to generate the look-up table for all multiplicities satisfying $1 \le |m| \le \max(p, q) + 1$. Further we exploit the symmetry of the two

boundaries to halve the number of entries. We arrive at a look-up table with

$$4((p+1)^2 + \binom{q+1}{2})(\max(p,q)+1)$$

entries. In order to keep the presentation simple, we do not discuss the case of multiple inner knots in this paper.

Finally, we use this look-up table to evaluate the elementary integrals by the formula

$$L_{ijkt}^{\alpha\beta} = \frac{1}{h_t^{\alpha+\beta-1}} I^{\sigma(\alpha,\beta)}_{\sigma(i-s,j-s),k-s,m(i,j)} \quad , \tag{28}$$

where the index shift has been defined before and the permutation σ swaps its arguments if $i < j$ and keeps them unchanged otherwise. Note that we omit the multiplicity $m = m(i,j)$ on the left-hand side, since it can be deduced from the underlying combined knot sequence of N_i and N_j.

Remark 6. As we will see later in Section 7, using $q = p$ appears to be the most reasonable choice. In this case, each entry of the look-up table is generated by three uniform B-splines of the same degree. We may take advantage of this fact by rewriting the elementary integrals (27) as

$$I_{ijk}^{\alpha\beta\gamma} = \int_{\mathbb{R}} \tilde{N}_{i,p}^{(\alpha)}(\tilde{x}) \tilde{N}_{j,p}^{(\beta)}(\tilde{x}) \tilde{N}_{k,p}^{(\gamma)}(\tilde{x}) \mathrm{d}\tilde{x},$$

with $\alpha\beta\gamma = 0$, $\alpha, \beta, \gamma \in \{0,1\}$ and $i \le j \le k$.

Using a similar reasoning as before (sorting and shifting indices), it is easy to see that these integrals suffice to assemble the stiffness matrix. To estimate the size of this table, note that i, j, k (modulo shifts) lead to $(p+1)(p+2)/2$ possibilities, while derivation orders α, β, γ lead to 7 cases. Finally, by taking into account knot multiplicities at boundary integrals as before, we obtain $p+1$ additional degrees of freedom. Consequently the total size of the lookup table is

$$\frac{7}{2}(p+1)^2(p+2) \ ,$$

for B-splines with a fixed degree $p = q$. Figure 4 shows a slice of this table for degrees $p = q = 2$. ◇

Example 7. For an illustration, we construct explicitly two entries of the 6×6 mass matrix of our running 1D example. The interpolation step (using degree two splines) provides the coefficient vector $(a^k) = (4.0, 16.5, 13.3, 1.4, 10.6, 16.0)$ for the Jacobian determinant. Since $h = 0.25$, it suffices to scale everything by this number, so that according to (28) and (23) $L^{ijk} = \frac{1}{4} I^{00}_{i,j,k,m}$. The general formula is, following (20),

$$M_{ij} = h \sum_{k=\max(j-2,1)}^{\min(i+2,6)} a^k L^{ijk} \quad , \quad i,j = 1,\ldots,6 \ .$$

j	k	$I^{0,0}_{0,j,k,1}$	$I^{0,0}_{0,j,k,2}$	$I^{0,0}_{0,j,k,3}$
0	0	12/35	13/70	1/7
0	1	43/420	11/120	11/210
0	2	1/840	1/840	1/210
1	1	43/420	17/168	23/420
1	2	1/168	1/168	1/105
2	2	1/840	1/840	1/420

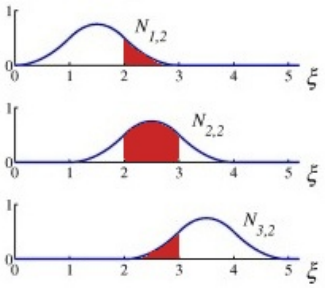


Fig. 4. Table for elementary integrals for the case $p = q = 2$, $\alpha = \beta = 0$ (no derivatives) and $m = 1, 2, 3$. For the configuration shown on the right, the result is $I^{0,0}_{0,1,2,1} = 1/168$.

For instance, we compute

$$\begin{aligned} M_{44} &= a^1 L^{3,3,1} + a^2 L^{3,3,2} + a^3 L^{3,3,3} + a^4 L^{3,3,4} + a^5 L^{3,3,5} \\ &= h(a^1 I^{0,0}_{0,2,2,2} + a^2 I^{0,0}_{0,1,1,1} + a^3 I^{0,0}_{0,0,0,1} + a^4 I^{0,0}_{0,1,1,2} + a^5 I^{0,0}_{0,2,2,3}) = 0.739\ldots \end{aligned}$$

To pass to elementary integrals, we used the shift invariance and symmetry properties. The second line contains values referring to the table in Figure 4. Similarly,

$$M_{12} = a^1 L^{0,1,0} + a^2 L^{0,1,1} + a^3 L^{0,1,2} = h(a^1 I^{0,0}_{0,0,1,3} + a^2 I^{0,0}_{0,1,1,3} + a^3 I^{0,0}_{0,1,2,3}) = 0.310\ldots$$

Note that each entry is computed as the sum of at most five terms. ◇

Remark 8. The IIL method can be extended to the case of non-uniform knot vectors, simply by generating look-up tables for all non-zero integrals of tri-products of univariate B-splines in each of the d coordinate directions in parameter space. If we assume to have approximately the same number of knots in each coordinate direction, then we need d look-up tables with $\mathcal{O}(n^{1/d}p^2)$ non-zero entries, where n is the total number of degrees of freedom. This is still an acceptable size for $d > 1$ as the stiffness matrix has $\mathcal{O}(np^d)$ non-zero entries. ◇

5 Error analysis

In this section we shall assume that the Jacobian determinant of the geometry map G is bounded away from zero. Our goal is to use Strang's lemma to demonstrate that the consistency error of the IIL assembly method possesses the same order of magnitude as the interpolation error in $\tilde{A}$ as $h \to 0$. In order to use Strang's lemma, we need to establish the uniform V_h-ellipticity of the bilinear form in (14).

The road map of our analysis is as follows. First we establish a bound on the absolute error in the computed bilinear form (14). Second, we use this result to establish uniform V_h-ellipticity as well as a bound on the consistency error term in Strang's Lemma, which depends on the error that is introduced in the approximation of the matrix A. Finally, we use these two ingredients to formulate our main result in Theorem 13.

First we establish the equivalence of H^1 seminorms on the parameter domain and the physical space and relate the constants to the matrix A, see also [3].

Lemma 9. *The seminorms $|\hat{v}|_{1,\hat{\Omega}}$ and $|v|_{1,\Omega}$ of a function $v \in V$ and its push-back $\hat{v} = v \circ G$ are equivalent. More precisely, there exist constants C_0, $C_1 \in \mathbb{R}_{>0}$ such that*

$$C_0|\hat{v}|_{1,\hat{\Omega}} \leq |v|_{1,\Omega} \leq C_1|\hat{v}|_{1,\hat{\Omega}} \ , \quad \forall v \in V.$$

Proof. Consider the matrices $A(\hat{\boldsymbol{x}}) = |\det J|J^{-1}\,J^{-T}$, $\hat{\boldsymbol{x}} \in \hat{\Omega}$, that corresponds to the Laplace operator ∇^2, cf. (9). Let $\lambda_{\max}(A)(\hat{\boldsymbol{x}})$ be the maximum eigenvalue of $A(\hat{\boldsymbol{x}})$. It is real and positive as A is symmetric and positive definite, therefore

$$|v|^2_{1,\Omega} = \int_\Omega \nabla v^T \nabla v \mathrm{d}\boldsymbol{x} = \int_{\hat{\Omega}} \hat{\nabla}\hat{v}^T\, A\, \hat{\nabla}\hat{v}\,\mathrm{d}\hat{\boldsymbol{x}} \leq \int_{\hat{\Omega}} \lambda_{\max}(A)\hat{\nabla}\hat{v}^T\,\hat{\nabla}\hat{v}\,\mathrm{d}\hat{\boldsymbol{x}}.$$

Since $\hat{\Omega}$ is closed and bounded, and the maximum eigenvalue of A depends continuously on $\hat{\boldsymbol{x}}$, there exist $C = \max_{\hat{\boldsymbol{x}}\in\hat{\Omega}} \lambda_{\max}(A)$ and

$$|v|^2_{1,\Omega} \leq C \int_{\hat{\Omega}} \hat{\nabla}\hat{v}^T\,\hat{\nabla}\hat{v}\,\mathrm{d}\hat{\boldsymbol{x}} = C|\hat{v}|^2_{1,\hat{\Omega}}.$$

One can choose the constants C_1 as the square root of the maximum of $\lambda_{\max}(A)$. A lower bound on $|v|_{1,\Omega}$ is obtained similarly, using the minimum eigenvalue $\lambda_{\min}(A)$ of A. □

Now we analyze the numerator of the consistency error term in Strang's lemma. We will see that the approximation error in (26) governs the error in $\tilde{a}(\cdot,\cdot)$.

Lemma 10. *The bilinear form* (18) *satisfies*

$$|a(u_h, v_h) - \tilde{a}(u_h, v_h)| \leq \varepsilon_h\, d\, |\hat{u}_h|_{1,\hat{\Omega}}\, |\hat{v}_h|_{1,\hat{\Omega}} \ . \tag{29}$$

Proof. Using the definition of the forms and the Cauchy-Schwarz inequality for the Euclidean norm $\|.\|_{\mathbb{R}^d}$, we obtain

$$\begin{aligned} |a(u_h, v_h) - \tilde{a}(u_h, v_h)| &\leq |\int_{\hat{\Omega}} \hat{\nabla}\hat{u}_h^T(A - \tilde{A})\hat{\nabla}\hat{v}_h\,\mathrm{d}\hat{\boldsymbol{x}}| \\ &\leq \int_{\hat{\Omega}} \|\hat{\nabla}\hat{u}_h\|_{\mathbb{R}^d}\, \|(A - \tilde{A})\hat{\nabla}\hat{v}_h\|_{\mathbb{R}^d}\,\mathrm{d}\hat{\boldsymbol{x}} \ . \end{aligned}$$

Let $\|M\|_2$ denote the spectral norm of a matrix $M \in \mathbb{R}^{d\times d}$. Since

$$\|(A - \tilde{A})\hat{\nabla}\hat{v}_h\|_{\mathbb{R}^d} \leq \|A - \tilde{A}\|_2 \|\hat{\nabla}\hat{v}_h\|_{\mathbb{R}^d},$$

and by taking maximum over all $\hat{\boldsymbol{x}} \in \hat{\Omega}$,

$$|a(u_h, v_h) - \tilde{a}(u_h, v_h)| \leq \max_{\hat{\boldsymbol{x}}\in\hat{\Omega}} \|A(\hat{\boldsymbol{x}}) - \tilde{A}(\hat{\boldsymbol{x}})\|_2 \int_{\hat{\Omega}} \|\hat{\nabla}\hat{u}_h\|_{\mathbb{R}^d}\, \|\hat{\nabla}\hat{v}_h\|_{\mathbb{R}^d}\,\mathrm{d}\hat{\boldsymbol{x}} \ .$$

By the Cauchy-Schwarz inequality $\langle f, g\rangle_{L^2(\hat{\Omega})} \leq \|f\|_{L^2(\hat{\Omega})}\, \|g\|_{L^2(\hat{\Omega})}$ in $L^2(\hat{\Omega})$ with $f = \|\hat{\nabla}\hat{u}_h\|_{\mathbb{R}^d}$ and $g = \|\hat{\nabla}\hat{v}_h\|_{\mathbb{R}^d}$ and taking into account that $\|\,\|\hat{\nabla}\hat{w}_h\|_{\mathbb{R}^d}\|_{L^2(\hat{\Omega})} = |\hat{w}_h|_{1,\hat{\Omega}}$ it follows that

$$|a(u_h, v_h) - \tilde{a}(u_h, v_h)| \leq \max_{\hat{\boldsymbol{x}}\in\hat{\Omega}} \|A(\hat{\boldsymbol{x}}) - \tilde{A}(\hat{\boldsymbol{x}})\|_2 |\hat{u}_h|_{1,\hat{\Omega}}\, |\hat{v}_h|_{1,\hat{\Omega}}.$$

For the spectral norm $\|M\|_2$ we use the inequality $\|M\|_2 \leq d\|M\|_{\max}$, where $\|M\|_{\max}$ is the element-wise max-norm (as in (26)). We arrive at

$$|a(u_h, v_h) - \tilde{a}(u_h, v_h)| \leq d \max_{\hat{\boldsymbol{x}} \in \hat{\Omega}} \|A(\hat{\boldsymbol{x}}) - \tilde{A}(\hat{\boldsymbol{x}})\|_{\max} |\hat{u}_h|_{1,\hat{\Omega}} \, |\hat{v}_h|_{1,\hat{\Omega}}.$$

Finally, in view of (26), we confirm (29). □

The following result makes sure that the assumptions of Strang's lemma are satisfied.

Lemma 11. *There exists a constant $h^\star$ such that the bilinear form $\tilde{a}(\cdot,\cdot)$ is uniformly V_h-elliptic for all $h \leq h^\star$.*

Proof. For our model problem, the bilinear form $a(\cdot,\cdot)$ in (3) is known to be $V-$elliptic. Consequently, since $V_h \subset V$, the form $a(\cdot,\cdot)$ is also uniformly V_h-elliptic, that is, there exists a constant $\mu > 0$ such that $\mu |v_h|^2_{1,(\Omega)} \leq a(v_h, v_h)$. We have

$$\begin{aligned}
\mu |v_h|^2_{1,\Omega} &\leq a(v_h, v_h) - \tilde{a}(v_h, v_h) + \tilde{a}(v_h, v_h) \\
&\leq \tilde{a}(v_h, v_h) + |a(v_h, v_h) - \tilde{a}(v_h, v_h)| \\
&\leq \tilde{a}(v_h, v_h) + \varepsilon_h \, d \, |\hat{v}_h|^2_{1,\hat{\Omega}} \\
&\leq \tilde{a}(v_h, v_h) + \varepsilon_h \, \frac{d}{C_0^2} \, |v_h|^2_{1,\Omega} \, ,
\end{aligned}$$

where we used Lemma 10 and the equivalence of H^1 seminorms (Lemma 9). Consequently,

$$\left(\mu - \varepsilon_h \, \frac{d}{C_0^2} \right) |v_h|^2_{1,\Omega} \leq \tilde{a}(v_h, v_h).$$

Since $\varepsilon_h \to 0$ as $h \to 0$ according to (26), there exists a mesh size $h^\star \in \mathbb{R}$ such that $\mu - \varepsilon_h \, \frac{d}{C_0^2} \geq \mu/2$ for all $h \leq h^\star$. □

The following Corollary establishes the required bound for the first consistency error term in Strang's lemma.

Corollary 12. *There exists a constant C_2, which is independent of h, such that the consistency error bound for the bilinear form* (18)*, which is computed using the approximation operator Π_h, satisfies*

$$\sup_{w_h \in V_h} \frac{|a(u_h^*, w_h) - \tilde{a}(u_h^*, w_h)|}{|w_h|_{1,\Omega}} \leq \varepsilon_h C_2 |u^*|_{1,\Omega} \, . \tag{30}$$

Proof. Consider $w_h \in V_h$. Using Lemma 10 and the equivalence of H^1 seminorms gives

$$|a(u_h^*, w_h) - \tilde{a}(u_h^*, w_h)| \leq \varepsilon_h \, d \, |\hat{u}_h^*|_{1,\hat{\Omega}} \, |\hat{w}_h|_{1,\hat{\Omega}} \leq \varepsilon_h \frac{d}{C_0^2} |u_h^*|_{1,\Omega} \, |w_h|_{1,\Omega}.$$

Using the fact that $|u_h^*|_{1,\Omega} \leq |u^*|_{1,\Omega}$ (since u_h^* is the Galerkin projection of u^*) and dividing by $|w_h|_{1,\Omega}$ implies (30). □

Finally we arrive at our main result, which is a consequence of the error bound (26), of the previous lemmas and of Strang's lemma:

Theorem 13. *When using an interpolation or quasi-interpolation operator Π_h which is based on splines of degree $q \geq p-1$ and which satisfies Assumption 5, the solution $\tilde{u}_h$ of (14) obtained using the method of integration by interpolation and lookup (IIL) possesses the optimal approximation order p with respect to the H^1 seminorm as $h \to 0$.*

Proof. Strang's lemma provides an upper bound on the error, which is the sum of discretization and consistency error bounds.

The discretization error in Strang's lemma is bounded by

$$|u^* - u_h^*|_{1,\Omega} \leq C_3 h^p \ , \tag{31}$$

where the constant C_3 is independent of h, see [3]. More precisely, the H^1 seminorm of the error has order p when using splines of degree p.

According to Corollary 12 and (26), the first consistency error term (30) in Strang's lemma has order $q+1$. Similar considerations apply to the second consistency error term in Strang's lemma. Consequently, the overall error satisfies

$$|u^* - \tilde{u}_h|_{1,\Omega} \leq \ C_3 \, h^p + C_A \, h^{q+1}.$$

Thus choosing q greater or equal to $p-1$ ensures the optimal order of approximation. □

This result is verified computationally in Section 7, see Table 1. However, we observe in our experiments that choosing $q = p-1$ does not lead to the optimal L^2 convergence rate for even degrees, while choosing $q = p$ gives optimal convergence rates both for the H^1 seminorm and the L^2 norm.

6 Complexity analysis

We quantify the asymptotic advantage of the IIL method in comparison to a typical Gauss quadrature assembly. As before, in order to simplify the analysis, we consider a domain of dimension d and we assume a common degree p for all directions. Further we denote with n the number of basis functions. It is roughly the same as the number of elements, since we consider single interior knots only. We use the $\mathcal{O}$ notation to express the asymptotic behavior of the computation time as p and n tend to infinity. The dimension d of the computational domain is considered to be a constant.

In the standard finite element practice, small (e.g. up to cubic) polynomial degrees are used in simulations. Nevertheless, the higher order of convergence achieved when utilizing a moderate but higher degree shall provide the desired level of accuracy with significantly fewer degrees of freedom and can therefore be quite appealing, provided that the computations can be carried out within affordable time. Moreover, if geometries are described using higher degree splines, which is sometimes the case in real-world applications, then it is natural to use splines of the same or higher degree for the isogeometric simulation as well, due to the isoparametric principle. For these reasons it makes sense to perform an analysis of the complexity with respect to the degree.

The IIL method, which is summarized in Algorithm 1, consists of three steps: Building the look-up tables, interpolation of the first factor in the required integrals, and matrix assembly.

Algorithm 6.1: IIL stiffness matrix assembly

Input: A geometry map G and two uniform B-spline bases of resp. degrees p and q.
Output: The stiffness matrix S.

Set $\boldsymbol{b}_0$ equal to $A = |\det J| J^{-1} \hat{K} J^{-T}$ evaluated on the n Greville abscissae;
for $i = 1, \dots, d$ **do**
    Construct the collocation matrix C of the $i-$th coordinate B-spline basis;
    Solve $C\boldsymbol{b}_i = \boldsymbol{b}_{i-1}$ /*multiple (re-ordered) right-hand sides */;
/* Coefficients $\tilde{A}_{\boldsymbol{k}}$ are given as a re-ordering of $\boldsymbol{b}_d$*/
for $\boldsymbol{i} \in \mathcal{I}$ **do**
    for $\boldsymbol{j}$ *with* $\operatorname{supp} T_{\boldsymbol{i},p} \cap \operatorname{supp} T_{\boldsymbol{j},p} \neq \emptyset$ **do**
        sum $\leftarrow 0$;
        for $\boldsymbol{k}$ *with* $\operatorname{supp} T_{\boldsymbol{k},q} \cap \operatorname{supp} T_{\boldsymbol{i},p} \cap \operatorname{supp} T_{\boldsymbol{j},p} \neq \emptyset$ **do**
            Evaluate the $d \times d$ matrix $L^{\boldsymbol{ijk}}$ with entries defined in Eq. (23)
            sum $\leftarrow$ sum $+ \tilde{A}_{\boldsymbol{k}} : L^{\boldsymbol{ijk}}$ /* ':' is the Frobenius inner product */ ;
    $S_{\boldsymbol{ij}} \leftarrow$ sum ;
return S ;

Building the look-up table is not costly, since the number of entries is $\mathcal{O}(p^3)$ for uniform knots. We may assume that this is precomputed and just fetched from a database [4]. We use this look-up table to generate the matrices $L^{\boldsymbol{ijk}}$. The algorithm could be accelerated further by precomputing these matrices also; this would require the precomputation of $\mathcal{O}(p^{2d})$ matrices.

For the (quasi-) interpolation step, we need to evaluate A on n Greville abscissae. When using de Boor's algorithm, each evaluation takes $\mathcal{O}(p^{d+1})$ time. However, we can exploit the uniformity of the knot vectors and precompute the values of the $(p+1)^d$ tensor-product B-splines that act on each Greville point. This reduces the complexity of each evaluation to $\mathcal{O}(p^d)$.

In order to compute the coefficient matrices $\tilde{A}_{\boldsymbol{k}}$ by interpolation, we proceed coordinate-wise and consider d univariate interpolation problems of size $n^{1/d}$ with $n^{(d-1)/d}$ right-hand sides each. Creating their matrices for uniform knots takes $\mathcal{O}(n^{1/d}p)$ time. We then compute the sparse LU factorizations of these d banded square matrices (with

[4] In the case of non-uniform knots (see Remark 8), a lookup table for each of the d coordinate directions in parameter space can be constructed by Gaussian quadrature. The size of the table is $\mathcal{O}(n^{1/d}p^2)$, since we need to store $\mathcal{O}(p^2)$ tri-product integral values for each of the $\mathcal{O}(n^{1/d})$ B-splines per direction (here we assume to have roughly the same number of knots in all directions). The cost of construction per entry (using $\mathcal{O}(p)$ Gauss nodes) is $\mathcal{O}(p^2)$, since the support of each B-spline contains $\mathcal{O}(p)$ knot spans. If $d > 1$, then these computational costs are negligible in relation to the remaining computations.

bandwidth $p+1$) of size $n^{1/d}$. Once these factorizations are available, we can find the coefficient matrices by iterated back substitution with a computational effort of $\mathcal{O}(dn^{(d-1)/d}n^{1/d}p) = \mathcal{O}(np)$. Alternatively we can compute the coefficient matrices directly by quasi-interpolation, with total computational costs of $\mathcal{O}(np^d)$. We will see that the time needed for the interpolation or quasi-interpolation is negligible compared to the overall computational costs, since the time needed for the entire algorithm is dominated by the matrix assembly step.

In the matrix assembly step, based on the coefficient matrices $\tilde{A}_{\boldsymbol{k}}$, we use formulas (21) and (23) to construct the entries of the stiffness matrix. The number of operations equals $\mathcal{O}(np^{2d})$. That is, for each of the n matrix rows we compute the $\mathcal{O}(p^d)$ non-zero matrix entries, and each entry is computed as a sum of $\mathcal{O}(p^d)$ terms (in particular $(2p+1)^d$ in the worst case, see Figure 3). Note that we need to evaluate the $d \times d$ matrix $L^{\boldsymbol{ijk}}$ for each triplet of indices, which requires d multiplications for each element. However, since we consider the dimension d as a constant, this does not have an impact on the overall complexity.

Summing up, we arrive at a total time complexity of order

$$\mathcal{O}(np^{2d}).$$

6.2 Gauss quadrature

We consider a typical Gauss assembly with ϱ^d quadrature points per element, see Algorithm 2. The core operation is the computation of the local stiffness matrix of every element. This involves evaluation of the basis functions at ϱ^d Gauss points, with cost $\mathcal{O}(p^{d+1})$ per point for de Boor's recursion and for the evaluation of the geometry factor. Similar to the previous discussion for the IIL method, one may exploit the uniformity of the knots to reduce the total complexity to $\mathcal{O}(n\varrho^d p^d)$ by precomputing the values of the $(p+1)^d$ basis functions on each element on the ϱ^d quadrature points.

Constructing the local element matrix involves $\mathcal{O}(p^{2d})$ iteration steps for each quadrature point, since $(p+1)^d$ basis functions take non-zero values at this point. As the most expensive step of the computation, we need to compute the inverse of the Jacobian matrix, which requires $\mathcal{O}(d^3)$ operations. Again, since we consider the dimension d as constant, this does not contribute to the overall computational complexity.

Summing up, the algorithm needs a total time of order

$$\mathcal{O}(n\varrho^d p^{2d}).$$

Note that the dominating term is not caused by the evaluations, but by the accumulation of the quadrature point contributions to the stiffness matrix. This observation implies that precomputing the values of basis functions at the quadrature points (which is clearly possible, as we consider the case of uniform knots), does not have a considerable effect on the asymptotic time. This observation is also reported in [24].

Both for the "full" Gauss quadrature with $p+1$ nodes and for the reduced version

Algorithm 6.2: Gauss stiffness matrix assembly

Input: A geometry map G and a tensor uniform B-spline basis of degree p.
Output: The stiffness matrix S.

Initialize matrix S with zeros ;
for *all elements* **do**
  Compute element quadrature points $\boldsymbol{x}_k$ and weights w_k, $k = 1, \ldots, (p+1)^d$;
  Initialize $E = 0$, the local element matrix of size $(p+1)^d \times (p+1)^d$;
  for *all quadrature points* $k = 1, \ldots (p+1)^d$ **do**
    Evaluate G, and $A = |\det J| J^{-1} \hat{K} J^{-T}$ on the quadrature point $\boldsymbol{x}_k$;
    Evaluate gradients $\hat{\nabla} T_{\boldsymbol{i}}(\boldsymbol{x}_k)$ of all active basis functions $T_{\boldsymbol{i}}$;
    for *For all* $(p+1)^d$ *active functions* $T_{\boldsymbol{r}}$ *on* $\boldsymbol{x}_k$ **do**
      for *For all* $(p+1)^d$ *active functions* $T_{\boldsymbol{s}}$ *on* $\boldsymbol{x}_k$ **do**
        $E \leftarrow E + \hat{\nabla} T^T_{\boldsymbol{r},p}(\boldsymbol{x}_k)\, A(\boldsymbol{x}_k)\, \hat{\nabla} T_{\boldsymbol{s},p}(\boldsymbol{x}_k)$
  Add contributions from E to S ;
return S ;

with p nodes, the total complexity reaches

$$\mathcal{O}(np^{3d}).$$

For small values of p, we do not expect significant savings in time from the IIL method over reduced Gauss quadrature with p nodes per knot-span. In fact, for the linear case and in any dimension, a reduced quadrature with a single quadrature point suffices to obtain an optimal convergence rate. Similarly, for degrees two and three, we expect a modest gain in the computation time, comparable to the savings obtained by using reduced quadrature rules. However, the gains should become substantial for moderate polynomial degrees, for splines of degree four or higher.

Note that this expectation is not a contradiction to the presented asymptotic time complexity, since constant factors in front of the dominating term, which are hidden in the big-$\mathcal{O}$ notation, come into play when using a small degree. For instance, in both approaches there is certainly an omitted factor of d^3 which is due to either matrix computations (for Gauss) or look-up operations (for IIL). Clearly, the actual constants are also highly dependent on the implementation. The next section presents several benchmarks, which will confirm both our theoretical observations and the practical efficiency of the IIL method.

7 Implementation and numerical results

We have implemented the proposed method in C++ using the G+Smo library[5]. In this section we compare experimental computation times for the stiffness matrix for 2D and 3D problems, using uniform B-spline discretizations of various degrees. Also, we verify the convergence rate of our method.

[5] See `http://www.gs.jku.at`

More precisely, we compare the performance of the IIL method (Algorithm 1) against an optimized Gauss quadrature assembly (Algorithm 2). A crucial task is the efficient set-up of the look-up tables, enabling fast access. Look-up queries, together with inner product operations, are the core operation in the IIL assembly procedure, therefore we have taken special care to optimize them. For our experiments we used no parallelization other than the vectorization enhancements present in the CPU. For all the experiments, double precision arithmetic is utilized.

7.1 Test examples

We use two model problems, which are based on the geometries (physical domains) displayed in Figure 5. Both geometries are represented in terms of non-rational B-spline patches. The 2D model problem has the source function

$$f(x, y) = 2\pi^2 \sin(\pi\, x) \sin(\pi\, y)$$

and the known exact solution

$$u(x, y) = \sin(\pi\, x) \sin(\pi\, y)\ .$$

Similarly, the 3D problem has the source function

$$f(x, y, z) = 3\pi^2 \sin(\pi\, x) \sin(\pi\, y) \sin(\pi\, z)$$

and known exact solution

$$u(x, y, z) = \sin(\pi\, x) \sin(\pi\, y) \sin(\pi\, z)\ .$$

The (inhomogeneous) Dirichlet boundary conditions are obtained by restricting the exact solutions to the boundary curves and surfaces of the physical domains. We use B-spline interpolation at the boundary Greville points to determine the coefficients of the approximate solution along the boundary.

7.2 Convergence rate

In order to verify the convergence rate of the IIL method, we solve the two model problems for spline discretizations of degrees $p = 2, \dots, 6$. For each degree, we test different values of the degree $q = \max(1, p-5), \dots, p$ that is used to interpolate the matrix A. The element size in the parameter domain $[0, 1]^d$ varies between $h_{\min} = 1.95 \cdot 10^{-3}$ and $h_{\max} = 2.5 \cdot 10^{-1}$. Consequently, the number n of degrees of freedom takes values in the range of 10^2 to $2.5 \cdot 10^5$.

The numerical results for dimension $d = 2$ and $d = 3$ are reported in Figures 6, 7 and 8, 9, respectively. We analyze both the order of convergence with respect to the H^1 seminorm (left) and with respect to the L^2 norm (right). The triangles indicate the experimentally observed rates of convergence. For fine discretizations and large degree we reached the limits of machine precision.

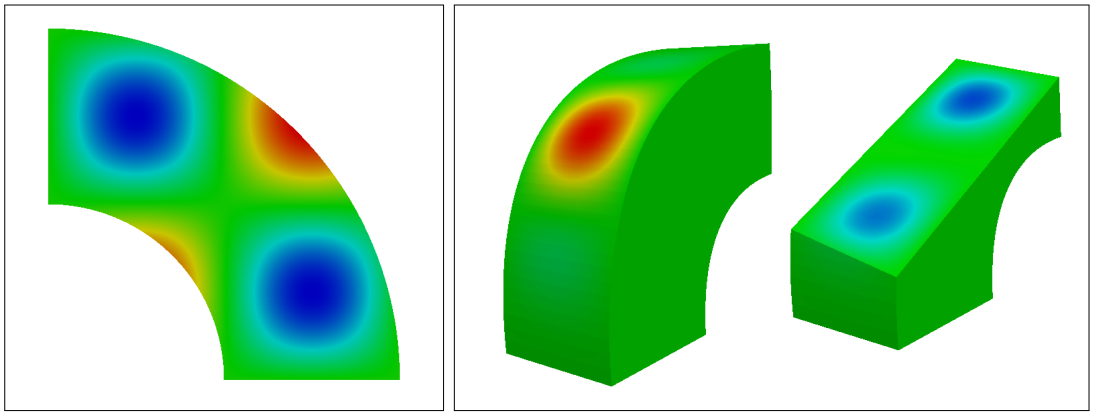


Fig. 5. Tensor B-spline domains for our test problems: A B-spline quarter annulus (left), as well as a B-spline 3D pipe and a slice of it. In both bases, the color distribution represents the exact solution of our model problem.

Table 1
Summary of the experimentally observed convergence rates: For each degree p (rows) and q (columns) we specify the rate of convergence for the H^1 seminorm and the L^2 norm. Optimal rates are bold.

$p\backslash q$	1	2	3	4	5	6
2	**2**,2	**2**,**3**	-	-	-	-
3	2,2	**3**,**4**	**3**,**4**	-	-	-
4	2,2	**4**,4	**4**,4	**4**,**5**	-	-
5	2,2	4,4	4,4	**5**,**6**	**5**,**6**	-
6	2,2	4,4	4,4	**6**,6	**6**,6	**6**,**7**

We see that $q = p - 1$ suffices to obtain the optimal rate in the H^1 seminorm. This confirms our theoretical results, which were summarized in Theorem 13 .

However, for even degree p, the observed order of convergence with respect to the L^2 norm is not optimal when choosing $q = p-1$. If we choose $q = p$, then the optimal order is obtained for any degree. Therefore we recommend the use of the same degree $p = q$ for both the discretization and for the interpolation operator. This does not only improve the convergence properties, but it also simplifies the implementation and reduces the size of the look-up table.

We summarize the experimentally obtained convergence rates in Table 1. The combinations of the degrees p, q giving optimal rates of convergence with respect to the L^2 norm are printed in bold font.

We also performed the same numerical experiments using Gauss quadrature with $p + 1$ nodes per element and obtained results that were virtually identical to those obtained by the IIL method with $q = p$. In order to keep the plots simple, we did not include these additional graphs.

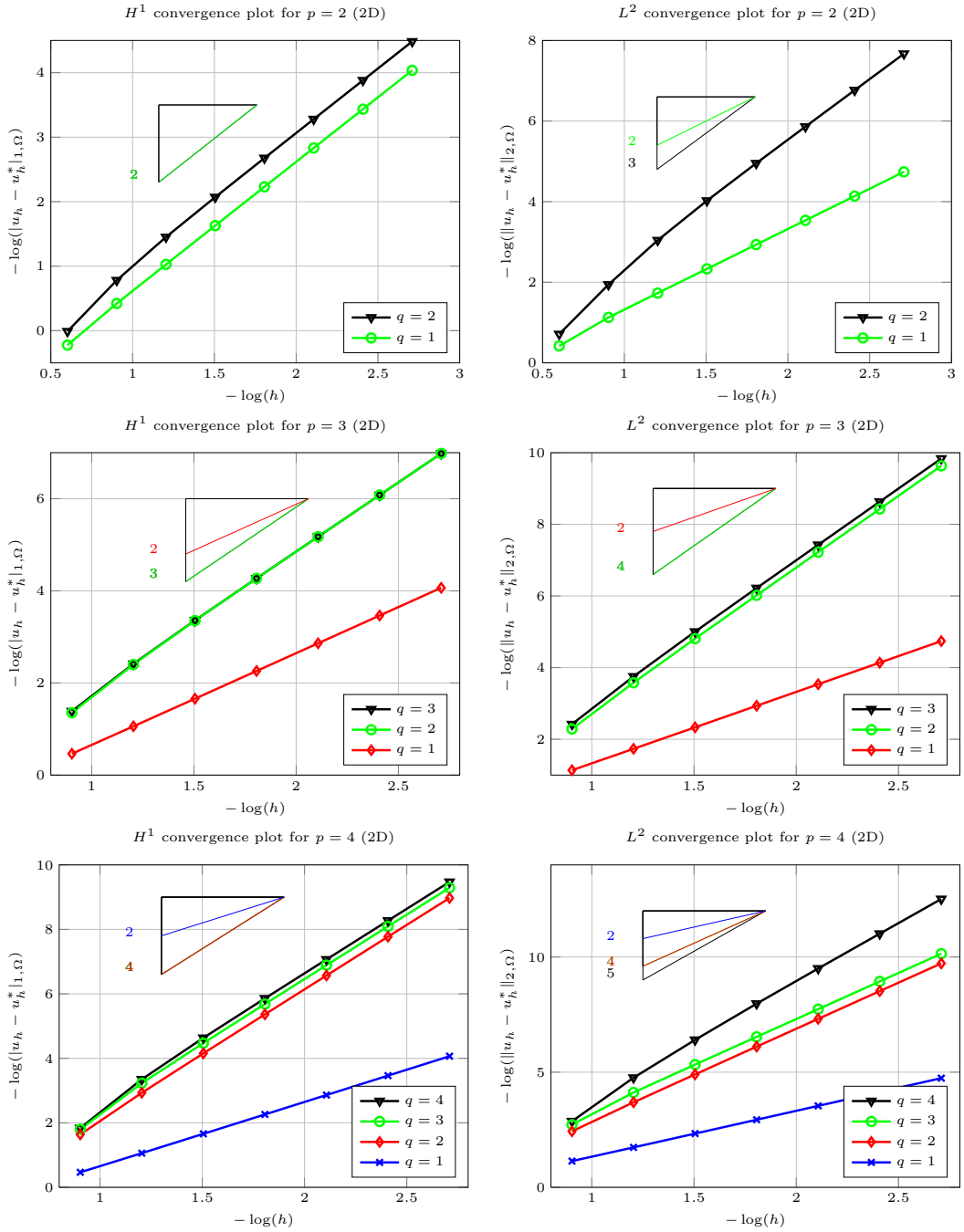


Fig. 6. 2D model problem: Error plots and convergence rates of the IIL method using degrees $p = 2, \ldots, 4$ and $q = 1, \ldots, p$.

7.3 Computation time

We conducted a series of experiments with various polynomial degrees and levels of refinement, both for the 2D and the 3D model problems, and analyzed the behavior of

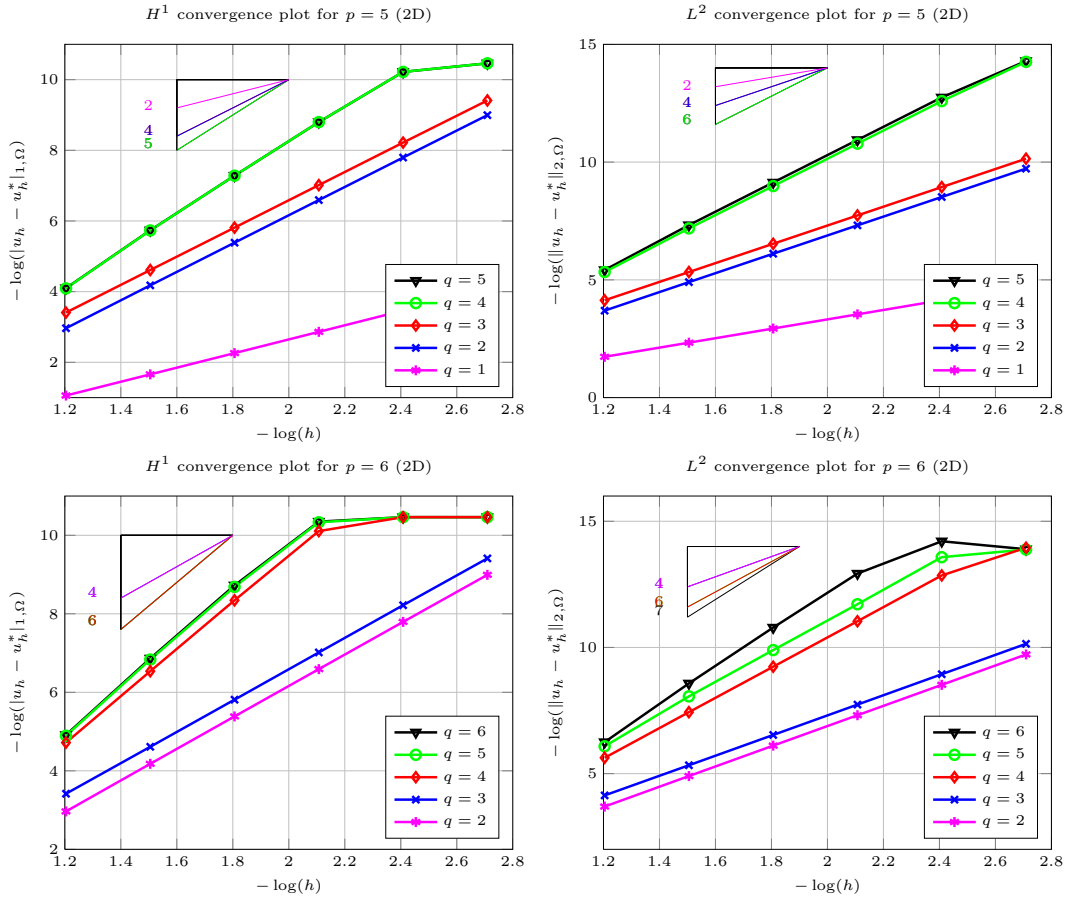


Fig. 7. 2D model problem: Error plots and convergence rates of the IIL method using degrees $p = 5, 6$ and $q = 1, \ldots, p$.

the computation time of the IIL method with $p = q$. As reference methods, we used the "full" Gaussian quadrature (with $\varrho = p + 1$) and the "reduced" Gaussian quadrature (with $\varrho = p$).

It should be noted that the reported computation times have been obtained using our implementations of these methods in G+Smo. Both implementations (for IIL and Gaussian quadrature) still provide various possibilities for improving their efficiency. Consequently, the absolute values of the computation times have to be considered with care. Nevertheless, we feel that these numbers – in particular their dependencies on degrees, dimensions and numbers of variables – provide useful insights into the properties of the different approaches to stiffness matrix assembly.

First we verified that the time needed to assemble the stiffness matrix grows linearly with the number of degrees of freedom for all methods. Table 2 presents the observed timings for different degrees and $h-$refined meshes.

Since we observed a nearly perfect linear scaling with n, we fixed the number of degrees of freedom to approximately 10^4 and $1.5 \cdot 10^4$ for the 2D and 3D problems in the remaining experiments, respectively.

Figures 10 and 11 present the results of a series of experiments, which help to explore

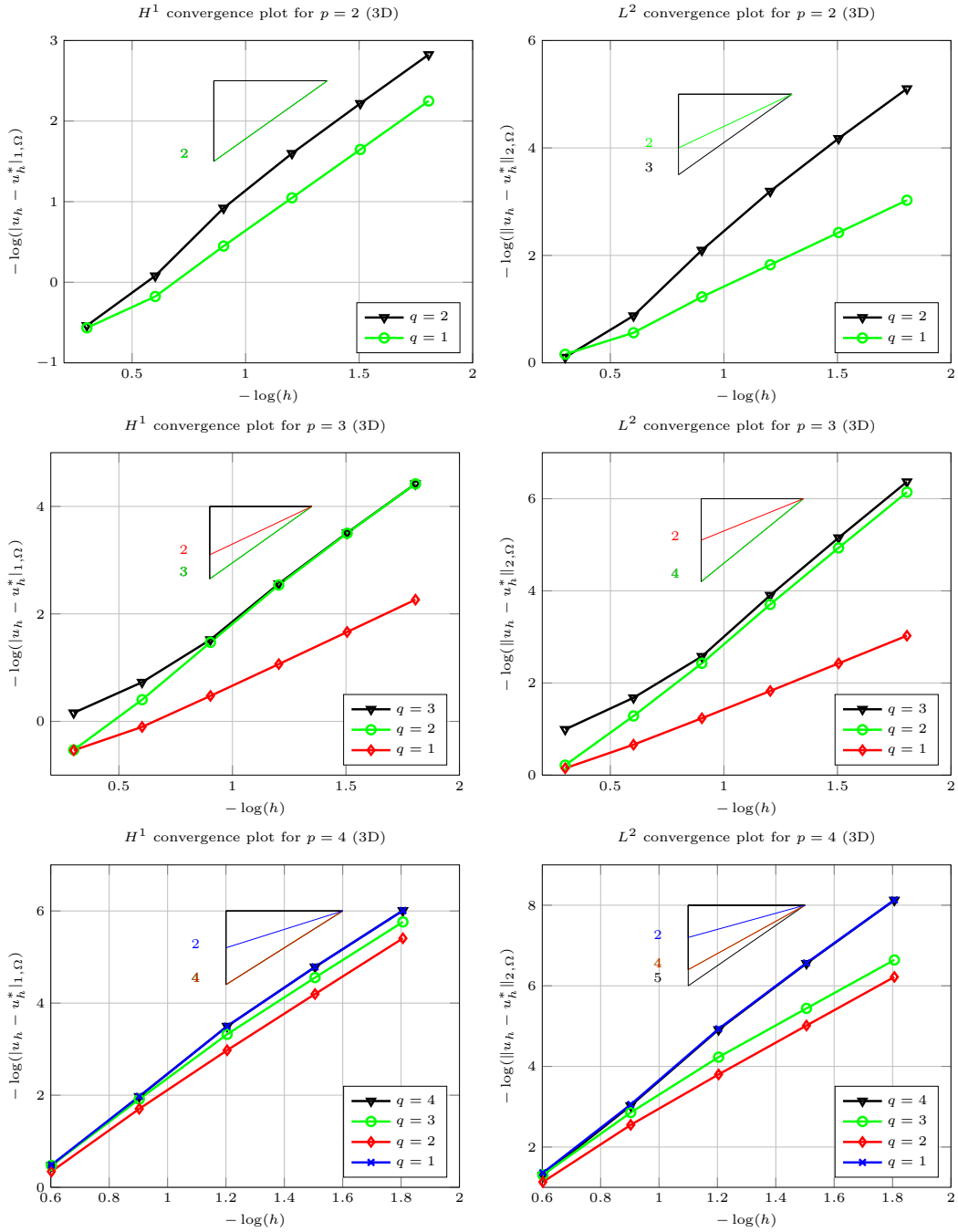


Fig. 8. 3D model problem: Error plots and convergence rates of the IIL method using different degree $p = 2, \ldots, 4$ and $q = 1, \ldots, p$.

the relation between the computation time and the polynomial degree p. In order to analyze the asymptotic behavior, we consider relatively large values of p (up to 15) and we measured the experimentally observed rate of growth (shown by the triangles in the doubly logarithmic plots on the right-hand side). Even though we did not yet

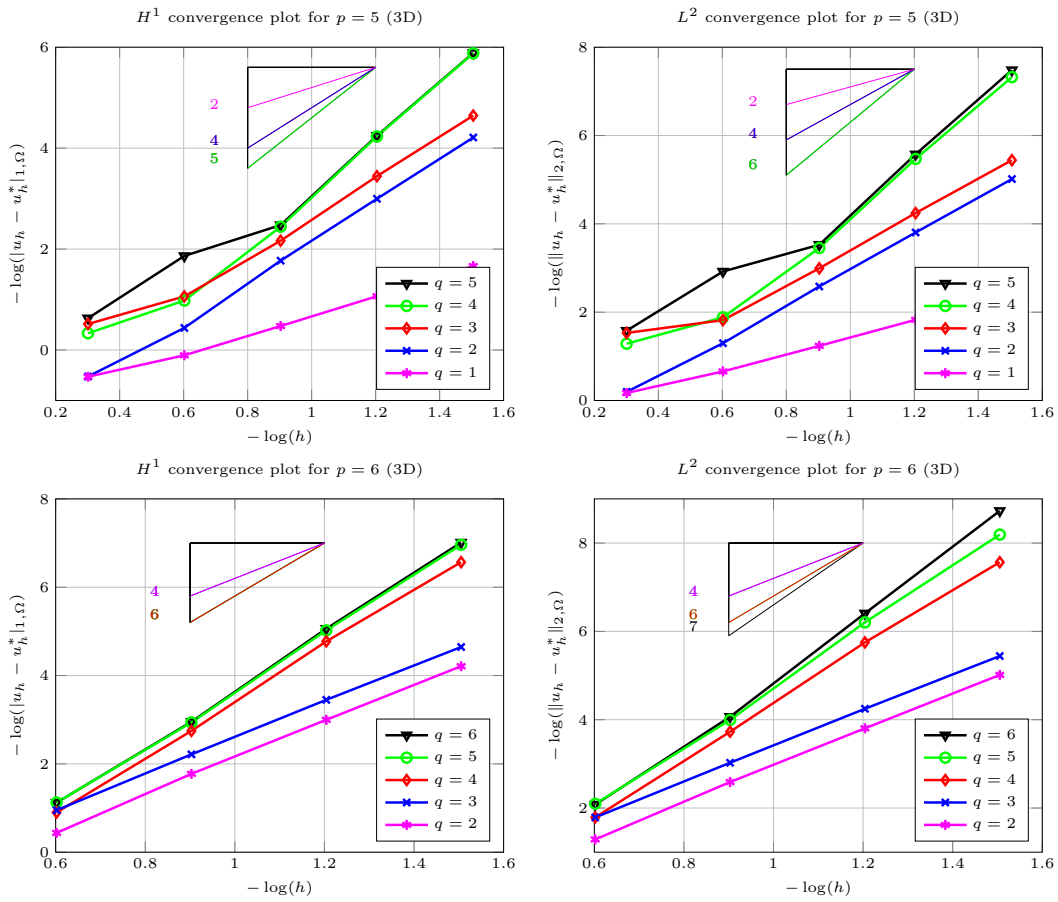


Fig. 9. 3D model problem: Error plots and convergence rates of the IIL method using degree $p = 5, 6$ and $q = 1, \ldots, p$.

Table 2
Assembly times for different polynomial degrees and problem sizes. The time consumed by the IIL method as well as by full Gauss quadrature (GQ) is shown (in seconds).

$p = 2$ (3D)			$p = 3$ (3D)			$p = 4$ (3D)		
DoFs	IIL	GQ	DoFs	IIL	GQ	DoFs	IIL	GQ
1000	0.05	0.07	1331	0.37	0.78	1728	1.72	5.03
5832	0.39	0.56	6859	2.56	6.28	8000	12.7	43.4
39304	3.06	4.55	42875	21.4	51.8	46656	92.6	353
287496	25.6	36.2	300763	163	439	314432	628	2312
$p = 5$ (2D)			$p = 6$ (2D)			$p = 7$ (2D)		
DoFs	IIL	GQ	DoFs	IIL	GQ	DoFs	IIL	GQ
4761	0.26	0.99	7396	0.77	3.71	3025	0.46	2.75
17689	1	3.91	27556	3.09	15.8	10609	1.93	11.6
68121	3.97	15.9	106276	13.6	63.5	39601	7.96	50.3
267289	17.1	67.6	417316	50.2	272	152881	30.4	205

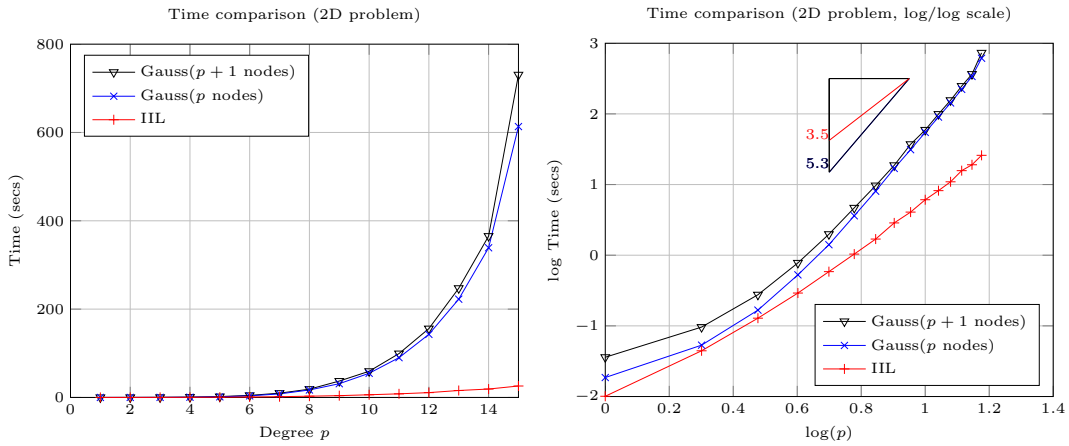


Fig. 10. 2D model problem: Computation time for assembling the stiffness matrices for degree $p = 1, \ldots, 15$. Standard (left) and doubly logarithmic plot (right).

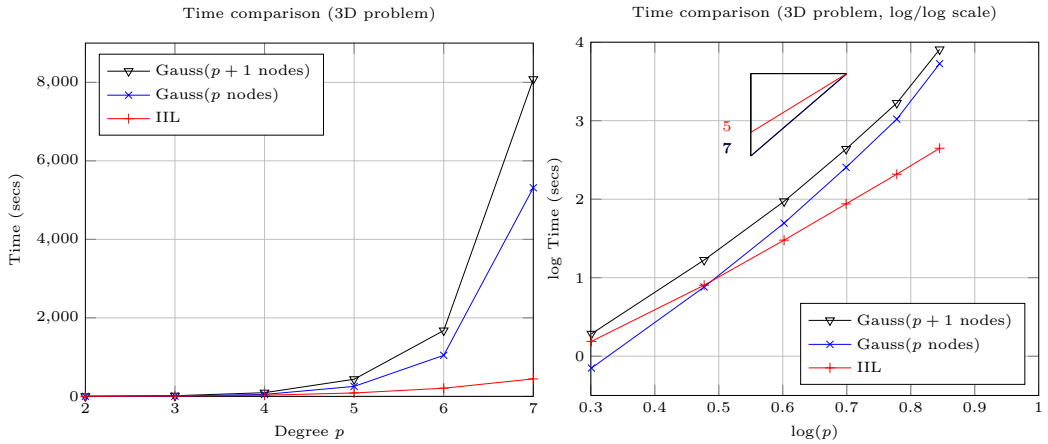


Fig. 11. 3D model problem: Computation time for assembling the stiffness matrices for degree $p = 1, \ldots, 7$. Standard (left) and doubly logarithmic plot (right).

reach the asymptotic range, where the rates should tend to 4 (IIL) and 6 (Gauss) for the 2D case, and to 6 (IIL) and 9 (Gauss) for the 3D case, the experimental results support our theoretical predictions regarding the computational complexity. Also, the theoretical predictions did not take the effects of modern hardware (built-in parallelism) into account.

Finally we report the speed-up (i.e., the ratio of the computation times) of the IIL method (shown in red) and of the reduced Gaussian quadrature (shown in blue) with respect to the full Gaussian quadrature in Figure 12, both in the 2D case (left) and in the 3D case (right). For the IIL method, the speed-up varies between 2.2 for $p = 2$ and 28 for $p = 15$ in the 2D case, and between 1.2 for $p = 2$ and 18 for $p = 7$ in the 3D case. In contrast, the speed-up obtained by using the "reduced" Gauss quadrature is significant only for low degrees. Note that this method uses only one evaluation per element for $p = 2$, hence it performs faster than the IIL method in this situation.

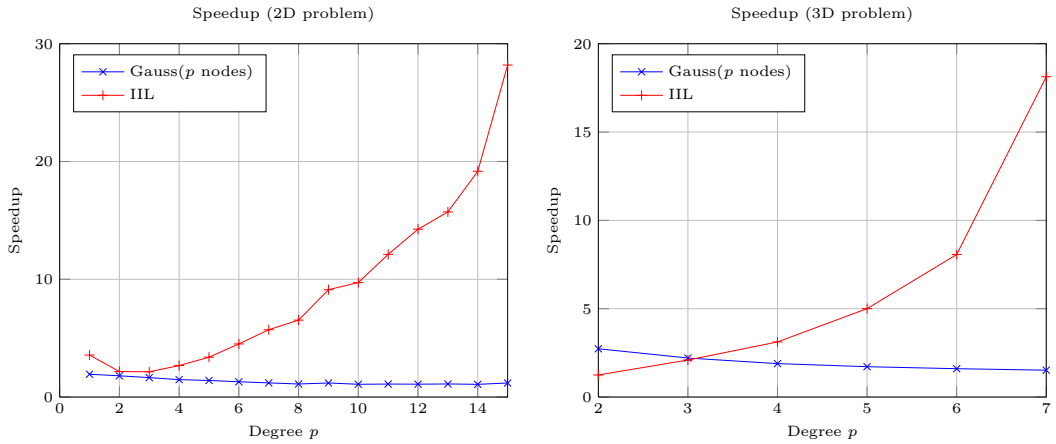


Fig. 12. The experimentally observed speedup for various polynomial degrees: 2D model problem (left) and 3D model problem (right).

8 Concluding remarks

We presented the method of Integration by Interpolation and Look-up for the numerical evaluation of the stiffness matrix integrals occurring in isogeometric analysis. It is based on the observation that these integrals share a small number of common factors, which represent the influence of the geometry mapping (i.e., the NURBS domain parameterization) and the contributions of possibly non-constant material coefficients. By applying spline interpolation to the common factors in the occurring integrals, we were able to transform them into integrals of piecewise polynomial functions. Their integrands are expressed in tensor-product B-spline form. Consequently, in the case of uniform spline discretizations, these integrals can be evaluated exactly and efficiently using pre-computed look-up tables for integrals of tri-products of univariate B-splines and their derivatives.

We performed a theoretical analysis in the case of elliptic problems. We showed that the IIL method maintains the overall approximation order of the Galerkin discretization, provided that the spline interpolation is sufficiently accurate. In addition we analyzed the computational complexity of our method and compared it with the standard Gauss quadrature method. In order to support these theoretical results, we conducted a series of numerical experiments which verify the findings in the paper.

Future work will concentrate on several issues related to the IIL method.

First, we will explore the extension of the IIL approach to the case of non-uniform knot vectors and to hierarchical splines [18,25,28,34]. On the one hand, symbolic techniques for evaluating integrals could allow for extension to the non-uniform case. There are intriguing connections between spline integrals and combinatorial quantities such as the Euler numbers (see [26] for a related discussion). On the other hand, imposing certain restrictions on the size of the knot spans (e.g., by requiring that the length of adjacent knot spans varies by factors 0.5, 1 and 2 only) and using restricted domain hierarchies for hierarchical spline spaces should lead to computationally efficient integration methods also in these cases, possibly requiring extended look-up tables.

Second, we observed that our current implementation does not provide substantial advantages for low polynomial degrees 2 and 3, which are quite popular in applications. We are currently exploring several options for accelerating our code, including increasing the number of pre-computed values in the look-up tables (such as pre-computing the entire matrices L^{ijk} defined in (23) instead of re-assembling it on the fly).

Third, we plan to extend the IIL approach to other types of splines, such as box-splines. These splines are closely related to certain classes of subdivision surfaces (in particular Loop and Catmull-Clark subdivision surfaces), which are increasingly important in isogeometric analysis, due to their built-in topological flexibility and their widespread use in Computer Graphics [12].

Fourth, throughout this paper we have assumed a smooth, well behaved geometry mapping. In the presence of severe distortions (or even singularities) in the mapping, a more careful analysis should be carried out and the use of adaptive quadrature methods will be in order. Adaptive generalizations of the IIL method are therefore of interest.

Acknowledgements

This research was supported by the National Research Network "Geometry + Simulation" (NFN S117, 2012–2016), funded by the Austrian Science Fund (FWF). The authors are grateful to Ulrich Langer for his encouragement and his help, and in particular for commenting on several preliminary versions of this text. We also thank the anonymous reviewers for their helpful comments.

References

[1] F. Auricchio, L. Beirão da Veiga, T. J. R. Hughes, A. Reali, and G. Sangalli. Isogeometric collocation methods. *Mathematical Models and Methods in Applied Sciences*, 20(11):2075–2107, 2010.

[2] F. Auricchio, F. Calabrò, T. Hughes, A. Reali, and G. Sangalli. A simple algorithm for obtaining nearly optimal quadrature rules for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 2012.

[3] Y. Bazilevs, L. Beirão da Veiga, J. A. Cottrell, T. J. R. Hughes, and G. Sangalli. Isogeometric analysis: Approximation, stability and error estimates for h-refined meshes. *Math. Models Methods Appl. Sci.*, 16(07):1031–1090, 2006.

[4] L. Beirão da Veiga, A. Buffa, J. Rivas, and G. Sangalli. Some estimates for h-p-k-refinement in isogeometric analysis. *Numerische Mathematik*, 118:271–305, 2011.

[5] G. Beylkin and M. Mohlenkamp. Algorithms for numerical analysis in high dimensions. *SIAM Journal on Scientific Computing*, 26(6):2133–2159, 2005.

[6] D. Braess. Finite elements: Theory, fast solvers and applications in solid mechanics, third edition. 2007.

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

[8] S. Brenner and L. Scott. *The Mathematical Theory of Finite Element Methods*. Springer-Verlag, 2002.

[9] F. Calabrò and C. Manni. The choice of quadrature in NURBS-based isogeometric analysis. In M. Papadrakakis, M. Kojic, and I. Tuncer, editors, *Proc. of the 3rd South-East European Conference on Computational Mechanics (SEECCM)*, 2013. http://www.eccomasproceedings.org/cs2013.

[10] F. Calabrò, C. Manni, and F. Pitolli. Computation of quadrature rules for integration with respect to refinable functions on fixed nodes. Submitted, 2014.

[11] P. G. Ciarlet. *Finite Element Method for Elliptic Problems*. Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

[12] F. Cirak, M. Ortiz, and P. Schröder. Subdivision surfaces: A new paradigm for thin-shell finite-element analysis. *Int. J. Numer. Meth. Engrg.*, 47:2039–2072, 2000.

[13] P. Costantini, C. Manni, F. Pelosi, and M. L. Sampoli. Quasi-interpolation in isogeometric analysis based on generalized B-splines. *Computer Aided Geometric Design*, 27(8):656–668, 2010.

[14] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis: Toward Integration of CAD and FEA*. John Wiley & Sons, Chichester, England, 2009.

[15] C. de Boor. Efficient computer manipulation of tensor products. *ACM Trans. Math. Softw.*, 5(2):173–182, June 1979.

[16] C. De Boor. *A practical guide to splines; rev. ed.* Applied mathematical sciences. Springer, Berlin, 2001.

[17] G. Farin. *Curves and surfaces for CAGD: a practical guide*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2002.

[18] C. Giannelli, B. Jüttler, and H. Speleers. THB–splines: The truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29:485–498, 2012.

[19] H. Gomez, A. Reali, and G. Sangalli. Accurate, efficient, and (iso)geometrically flexible collocation methods for phase-field models. *Journal of Computational Physics*, 262(0):153 – 171, 2014.

[20] T. Hopkins and R. Wait. Some quadrature rules for Galerkin methods using B-spline basis functions. *Computer Methods in Applied Mechanics and Engineering*, 19(3):401–416, 1979.

[21] T. Hughes, J. Cottrell, and Y. Bazilevs. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. *Computer Methods in Applied Mechanics and Engineering*, 194(39–41):4135–4195, 2005.

[22] T. Hughes, A. Reali, and G. Sangalli. Efficient quadrature for NURBS-based isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199(58):301–313, 2010.

[23] A. Karatarakis, P. Karakitsios, and M. Papadrakakis. Computation of the isogeometric analysis stiffness matrix on GPU. In M. Papadrakakis, M. Kojic, and I. Tuncer, editors, *Proc. of the 3rd South-East European Conference on Computational Mechanics (SEECCM)*, 2013. http://www.eccomasproceedings.org/cs2013.

[24] A. Karatarakis, P. Karakitsios, and M. Papadrakakis. GPU accelerated computation of the isogeometric analysis stiffness matrix. *Computer Methods in Applied Mechanics and Engineering*, 269(0):334 – 355, 2014.

[25] G. Kuru, C. Verhoosel, K. van der Zee, and E. van Brummelen. Goal-adaptive isogeometric analysis with hierarchical splines. *Computer Methods in Applied Mechanics and Engineering*, 270(0):270 – 292, 2014.

[26] A. Mantzaflaris and B. Jüttler. Exploring matrix generation strategies in isogeometric analysis. In M. Floater et al., editors, *Mathematical Methods for Curves and Surfaces*, volume 8177 of *Lecture Notes in Computer Science*, pages 364–382. Springer Berlin Heidelberg, 2014.

[27] B. Patzák and D. Rypl. Study of computational efficiency of numerical quadrature schemes in the isogeometric analysis. In *Proc. of the 18th Int'l Conf. Engineering Mechanics*, EM'12, pages 1135–1143, 2012.

[28] D. Schillinger, L. Dedè, M. Scott, J. Evans, M. Borden, E. Rank, and T. Hughes. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and t-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249252(0):116 – 150, 2012.

[29] D. Schillinger, J. Evans, A. Reali, M. Scott, and T. Hughes. Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations. *Computer Methods in Applied Mechanics and Engineering*, 267:170–232, 2013.

[30] D. Schillinger, S. Hossain, and T. Hughes. Reduced Bézier element quadrature rules for quadratic and cubic splines in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 277(0):1 – 45, 2014.

[31] L. Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, 3 edition, 2007.

[32] G. Strang. Approximation in the finite element method. *Numerische Mathematik*, 19:81–98, 1972.

[33] G. Strang and G. J. Fix. *An Analysis of the Finite Element Method*. Prentice-Hall, Englewood Cliffs, N.J., 1973.

[34] A.-V. Vuong, C. Giannelli, B. Jüttler, and B. Simeon. A hierarchical approach to adaptive local refinement in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 200(4952):3554 – 3567, 2011.