ICES REPORT 13-03

February 2013

Isogeometric Collocation: Cost Comparison with Galerkin Methods and Extension to Adaptive Hierarchical NURBS Discretizations

by

Dominik Schillinger, John A. Evans, Alessandro Reali, Michael A. Scott, Thomas J.R. Hughes

The Institute for Computational Engineering and Sciences
The University of Texas at Austin
Austin, Texas 78712

Reference: Dominik Schillinger, John A. Evans, Alessandro Reali, Michael A. Scott, Thomas J.R. Hughes, Isogeometric Collocation: Cost Comparison with Galerkin Methods and Extension to Adaptive Hierarchical NURBS Discretizations, ICES REPORT 13-03, The Institute for Computational Engineering and Sciences, The University of Texas at Austin, February 2013.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
FEB 2013		**00-00-2013 to 00-00-2013**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Isogeometric Collocation: Cost Comparison with Galerkin Methods and Extension to Adaptive Hierarchical NURBS Discretizations	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Texas at Austin,Institute for Computational Engineering and Sciences,Austin,TX,78712**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT

We compare isogeometric collocation with isogeometric Galerkin and standard C0 finite element methods with respect to the cost of forming the matrix and residual vector, the cost of direct and iterative solvers, the accuracy versus degrees of freedom and the accuracy versus computing time. On this basis, we show that isogeometric collocation has the potential to increase the computational efficiency of isogeometric analysis and to outperform both isogeometric Galerkin and standard C0 finite element methods, when a specified level of accuracy is to be achieved with minimum computational cost. We then explore an adaptive isogeometric collocation method that is based on local hierarchical refinement of NURBS basis functions and collocation points derived from the corresponding multi-level Greville abscissae. We introduce the concept of weighted collocation that can be consistently developed from the weighted residual form and the two-scale relation of B-splines. Using weighted collocation in the transition regions between hierarchical levels, we are able to reliably handle coincident collocation points that naturally occur for multi-level Greville abscissae. The resulting method combines the favorable properties of isogeometric collocation and hierarchical refinement in terms of computational efficiency, local adaptivity, robustness and straightforward implementation, which we illustrate by numerical examples in one, two and three dimensions.

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**105**	

Isogeometric Collocation: Cost Comparison with Galerkin Methods and Extension to Adaptive Hierarchical NURBS Discretizations

Dominik Schillinger[a,*], John A. Evans[a], Alessandro Reali[b],
Michael A. Scott[c], Thomas J.R. Hughes[a]

[a] *Institute for Computational Engineering and Sciences, The University of Texas at Austin, USA*
[b] *Department of Civil Engineering and Architecture, University of Pavia, and IMATI-CNR, Pavia, Italy*
[c] *Department of Civil and Environmental Engineering, Brigham Young University, Provo, USA*

Abstract

We compare isogeometric collocation with isogeometric Galerkin and standard C^0 finite element methods with respect to the cost of forming the matrix and residual vector, the cost of direct and iterative solvers, the accuracy versus degrees of freedom and the accuracy versus computing time. On this basis, we show that isogeometric collocation has the potential to increase the computational efficiency of isogeometric analysis and to outperform both isogeometric Galerkin and standard C^0 finite element methods, when a specified level of accuracy is to be achieved with minimum computational cost. We then explore an adaptive isogeometric collocation method that is based on local hierarchical refinement of NURBS basis functions and collocation points derived from the corresponding multi-level Greville abscissae. We introduce the concept of weighted collocation that can be consistently developed from the weighted residual form and the two-scale relation of B-splines. Using weighted collocation in the transition regions between hierarchical levels, we are able to reliably handle coincident collocation points that naturally occur for multi-level Greville abscissae. The resulting method combines the favorable properties of isogeometric collocation and hierarchical refinement in terms of computational efficiency, local adaptivity, robustness and straightforward implementation, which we illustrate by numerical examples in one, two and three dimensions.

Keywords: Isogeometric analysis, isogeometric collocation methods, hierarchical refinement of NURBS, weighted collocation, reduced quadrature, local adaptivity

*Corresponding author;
Institute for Computational Engineering and Sciences, The University of Texas at Austin, 201 East 24th Street, Austin, TX 78712, USA; Phone: +1 512-232-7767; Fax: +1 512-232-7508; E-mail: dominik@ices.utexas.edu

Contents

1. Introduction

Isogeometric analysis (IGA) was introduced by Hughes and coworkers [1, 2] to bridge the gap between computer aided geometric design (CAGD) and finite element analysis (FEA). The core idea of IGA is to use the same *smooth* and *higher-order* basis functions, e.g. non-uniform rational B-splines (NURBS) or T-splines, for the representation of both geometry in CAGD and the approximation of solutions fields in FEA. The primary goal of IGA is to simplify the cost-intensive mesh generation process required for standard FEA and to support a more tightly connected interaction between CAGD and FEA tools [3–5]. In addition and perhaps even more important, IGA turned out to be a superior computational mechanics technology, which on a per-degree-of-freedom basis exhibits increased accuracy and robustness in comparison to standard finite element methods (FEM) [6, 7]. Of particular importance in this respect is the observation that, unlike standard FEM, the higher modes of IGA basis functions do not diverge with increasing degree, but achieve almost spectral accuracy that improves with degree [8, 9]. IGA has been successfully applied in a variety of areas, such as structural vibrations [2, 8], incompressibility [10, 11], shells [12–15], fluid-structure interaction [16–18], turbulence [19, 20], phase field analysis [21–24], contact mechanics [25–27], shape optimization [28, 29], immersed boundary methods [30–34] and boundary element analysis [35, 36].

Beyond their favorable approximation properties, the technical aspects of smooth higher-order basis functions raise the question of their efficient implementation. The integration of IGA capabilities into existing standard FEM codes was recently simplified by the concept of Bézier extraction for NURBS [37] and T-splines [38, 39]. The development of suitable direct and iterative solvers has been recently initiated on shared [40, 41] and distributed architectures [42]. A further important issue is computationally efficient quadrature rules that achieve exact integration of system matrices with the smallest possible number of quadrature points. In Galerkin-type formulations, element-wise Gauss quadrature is optimal for standard FEM, but sub-optimal for IGA, since it ignores the inter-element continuity of its smooth basis functions. Taking into account the smoothness across element boundaries, a number of more efficient quadrature rules with reduced sets of quadrature points were recently developed by Hughes et al. [43, 44]. Motivated by their work on quadrature, they also initiated research on isogeometric collocation methods [45, 46], which can be interpreted as a one-point quadrature rule in the IGA context.

1.1. Is isogeometric collocation a game changer?

In contrast to Galerkin-type formulations, collocation is based on the discretization of the strong form of the underlying partial differential equations (PDE), which requires basis functions of sufficiently high order and smoothness. Consequently, the use of IGA for collocation suggests itself, since spline functions such as NURBS or T-splines can be readily adjusted to any order of polynomial degree and continuity required by the differential operators at hand. Furthermore, they can be generated for domains of arbitrary geometric and topological complexity, directly linked to and fully supported by CAGD technology. The major advantage of isogeometric collocation over Galerkin-type methods is the minimization of the computational effort with respect to quadrature, since for each degree of freedom only one point evaluation at a so-called collocation point is required. This exceptional property constitutes a significant advantage of isogeometric collocation for applications where the efficiency and success of an analysis technology is directly related to the cost of quadrature. Furthermore, the bandwidth and non-zero population within the band are significantly reduced compared with a Galerkin method. This improves the performance of both direct and iterative equation solvers as well.

The most salient example of an application whose speed is almost entirely dependent on the cost of quadrature is explicit structural dynamics, where the computational cost is dominated by stress divergence evaluations at quadrature points for the calculation of the residual force vector. Codes used extensively for crash dynamics and metal forming, such as LS-DYNA, rely almost exclusively on low-order quadrilateral and hexahedral elements with one-point quadrature. This minimizes memory requirements and the number of constitutive evaluations and thus allows for the efficient computation of very large industrial problems with standard hardware in a reasonable time. However, one-point quadrature leads to rank deficient system matrices, which in turn induces mesh instabilities, e.g. "hourglass modes" [47]. Therefore, an additional stabilization by artificial viscous and/or elastic mechanisms becomes necessary, whose parameters usually require fine-tuning by computationally expensive and time-consuming sensitivity studies. Isogeometric collocation can be viewed as a one-point quadrature scheme that is rank sufficient. It provides the same advantages as standard techniques in terms of memory and computational efficiency, but, in addition, is free of mesh instabilities. Hence, IGA collocation methods eliminate the need for ad hoc hourglass stabilization techniques and their tuning parameters. Furthermore, they show great promise for the development of higher-order accurate time integration schemes due to the convergence of the high modes in the eigenspectrum [46] as well as for the development

of locking free beam, plate and shell elements [48].

A further promising example is computational fluid dynamics (CFD). Over the last two decades, B-spline basis functions have been successfully used in the analysis of Navier-Stokes problems, in particular wall-bounded turbulent flows [19, 20, 49–52]. Due to their maximum smoothness, B-splines exhibit a high resolution power, which allows the representation of a broad range of scales of a turbulent flow. This eliminates the need to construct separate boundary schemes and leads to enhanced numerical results in comparison to other CFD approaches. For incompressible flows, divergence-conforming tensor-product B-splines are capable of exactly satisfying the incompressibility constraint a priori [53–55]. However, simulations based on Galerkin discretizations with B-splines are dominated by the high cost of quadrature. The evaluation of the nonlinear convection terms alone usually consumes more than 50% of the total analysis time [53]. B-spline collocation has been proposed and examined in several studies as an economical alternative that fully inherits the advantages of B-spline discretizations, but requires only a fraction of the computational time and memory due to the minimization of point evaluations [56–60]. Unfortunately, these studies seem to have sparked little interest in the CFD community so far, most probably since they have been limited to very simple geometries and because an efficient technology to incorporate more complex geometries has been missing. We believe that this gap can be ideally filled by isogeometric collocation methods, which are capable of naturally incorporating complex geometries based on CAGD technology. In our opinion, and based on the promising results of [57–59], isogeometric collocation opens the door for an efficient higher-oder accurate and robust CFD analysis technology that works with a minimum number of quadrature points and fully embraces the geometric capabilities of IGA.

1.2. How does isogeometric collocation perform with respect to Galerkin methods?

To shed light on this essential question, the present paper will first compare isogeometric collocation (IGA-C) with isogeometric Galerkin (IGA-G) and standard C^0 finite element methods (FEA-G) in terms of their computational cost and efficiency. Aiming at a broad and complete picture, we highlight three different aspects: First, we assess the computational cost for forming and assembling discrete systems that emerge from IGA-C, IGA-G and FEA-G. We consider both stiffness matrices and residual vectors, taking into account the total number of quadrature/collocation points as well as the floating point operations required to evaluate a collocation and a quadrature point in different problem classes. The operation counts demonstrate that, compared with the Galerkin methods, IGA-C considerably reduces the computing cost. Second, we examine key indicators, such as bandwidth and

cost of matrix-vector products, to characterize the efficiency of direct and iterative solvers. They consistently indicate a superior solver performance for IGA-C over IGA-G. Third, we quantify the cost of IGA-C, IGA-G and FEA-G to solve a series of representative benchmark problems in 3D, considering different combinations of smooth and "rough" solutions for scalar and vector fields. We compare the different methods with respect to accuracy vs. the number of degrees of freedom as well as accuracy vs. the total computing time. With respect to the number of degrees of freedom, IGA-G is several orders of magnitude more accurate than IGA-C and FEA-G. With respect to accuracy vs. computing time, IGA-C is superior to both IGA-G and FEA-G. The latter manifests the potential of isogeometric collocation as a fast and accurate IGA technology.

1.3. Local adaptivity in isogeometric collocation with hierarchical refinement of NURBS

In the second part of the paper, we explore the use of hierarchically refined NURBS basis functions for isogeometric collocation. Hierarchical refinement of NURBS has recently gained increasing attention as a viable pathway to local refinement of NURBS parameterizations for use in IGA [61–64]. The technology relies on the principle of B-spline subdivision [65, 66], which makes it possible to reliably control linear independence throughout the refinement process. In addition, the maximum smoothness of NURBS is maintained, which is essential for use in collocation. Since hierarchical B-splines rely on a local tensor product structure, they can be easily generalized to arbitrary dimensions and facilitate automation of the refinement process [63, 64]. A hierarchical organization of a basis can be directly transferred into tree data structures [67–69], which allow for a straightforward implementation with manageable coding effort.

However, at first sight, using a hierarchically refined NURBS basis in collocation seems not straightforward, since collocation points derived from the Greville abscissae of different hierarchical levels are generally not distinct, which is necessary for the linear independence of the system matrix. To overcome this problem, we introduce the concept of weighted collocation for NURBS basis functions. Instead of generating each discrete collocation equation from the evaluation of the PDE at a single collocation point, we use a weighted average of PDE evaluations taken at several collocation points. Weighted IGA collocation can be consistently derived from the two-scale relation of B-spline subdivision and allows the presence of coincident collocation points. To preserve the core advantage of a minimum number of point evaluations, we restrict the use of weighted collocation to the transition regions, where NURBS basis functions of different hierarchical levels overlap, and continue to collocate at single Greville abscissae in the rest of the domain, where coincident collocation

points cannot occur. A further simplification of the method can be achieved by exploiting the idea of truncated hierarchical NURBS [63, 70] in the transition regions. The validity and effectiveness of the resulting adaptive IGA collocation scheme is illustrated by a number of numerical examples in one, two and three dimensions. In particular, we demonstrate that, if we use the weighting scheme locally, the ratio between the number of collocation points and the number of degrees of freedom always remains close to one.

1.4. Structure and organization of the paper

The present work is organized as follows: Section 2 provides a brief review of NURBS based isogeometric analysis and the derivation of IGA collocation from the method of weighted residuals. Section 3 presents the comparative study that discusses various aspects related to the computational efficiency of IGA-C, IGA-G and FEA-G. Section 4 gives a brief introduction to hierarchical refinement of NURBS basis functions. Section 5 introduces the concept of weighted isogeometric collocation. Section 6 derives the adaptive isogeometric collocation method, combining the advantages of standard IGA collocation, weighted collocation and hierarchical refinement. Section 7 presents a range of numerical examples, illustrating the effectiveness of adaptive IGA collocation. Section 8 summarizes our most important points and motivates future research in IGA collocation.

2. NURBS-based isogeometric collocation

We start with a concise introduction to isogeometric collocation methods in the spirit of [45, 46]. After a brief review of B-spline and NURBS basis function technology, we derive the discrete collocation equations, following the method of weighted residuals. We clarify the main features of collocation by comparison with the well-known Galerkin method. Finally, we summarize some key technical aspects of isogeometric collocation.

2.1. B-spline and NURBS basis functions

In the following, we outline some technical aspects of B-spline and NURBS bases for IGA. Readers interested in more details are referred to Piegl and Tiller [71], Cohen et al. [72], Rogers [73] or Farin [74], who provide in-depth reviews of the underlying geometric concepts and algorithms.

2.1.1. Univariate B-splines

A B-spline basis of degree p is formed from a sequence of knots called a knot vector $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$, where $\xi_1 \leq \xi_2 \leq \ldots \leq \xi_{n+p+1}$ and $\xi \in \mathbb{R}$ is called a knot. A

univariate B-spline basis function $N_{i,p}(\xi)$ is defined using a recurrence relation, starting with the piecewise constant ($p = 0$) basis function

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

For $p > 0$, the basis function is defined using the Cox-de Boor recursion formula

$$N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \tag{2}$$

If a knot has multiplicity k, the smoothness of the B-spline basis is C^{p-k} at that location. Fig. 1a illustrates a B-spline basis of polynomial degree $p = 3$ and knot vector $\Xi = \{0, 0, 0, 0, 1, 2, 3, 4, 4, 4, 4\}$, where knots at the beginning and the end are repeated $p+1$ times to make the basis interpolatory (in this case, the knot vector is said to be open).

Having constructed the corresponding basis functions, we can build a B-spline curve in d_s dimensions by a linear combination of basis functions

$$\boldsymbol{C}(\xi) = \sum_{i=1}^{n} \boldsymbol{P}_i \, N_{i,p}(\xi) \tag{3}$$

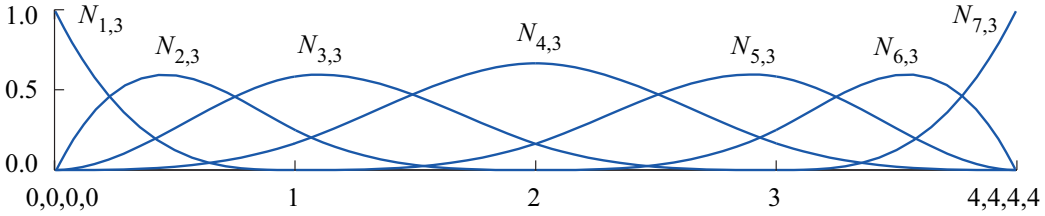
where coefficients $\boldsymbol{P}_i \in \mathbb{R}^{d_s}$ are called control points. Piecewise linear interpolation of the control points defines the control polygon. An example generated from the B-spline basis shown in Fig. 1a is provided in Fig. 1b. Note that the curve is of continuity class C^2.

2.1.2. Multivariate B-splines

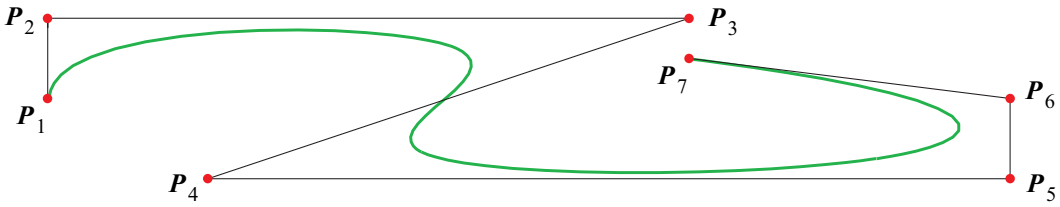
Multivariate B-splines are a tensor product generalization of univariate B-splines. We use d_s and d_p to denote the dimension of the physical and parameter spaces, respectively. Multivariate B-spline basis functions are generated from d_p univariate knot vectors

$$\Xi^{\ell} = \{\xi_1^{\ell}, \xi_2^{\ell}, ..., \xi_{n_\ell + p_\ell + 1}^{\ell}\} \tag{4}$$

where $\ell = 1, \ldots, d_p$, p_ℓ indicates the polynomial degree along parametric direction ℓ, and n_ℓ is the associated number of basis functions. The resulting univariate B-spline basis functions in each direction ℓ can then be denoted by N_{i_ℓ,p_ℓ}^{ℓ}, from which multivariate basis functions



(a) *Cubic B-spline patch with interpolatory ends.*



(b) *B-spline curve generated from the above basis using control points* $\boldsymbol{P}_i$.

Figure 1: *Example of cubic B-spline basis functions and a corresponding B-spline curve in 2D.*

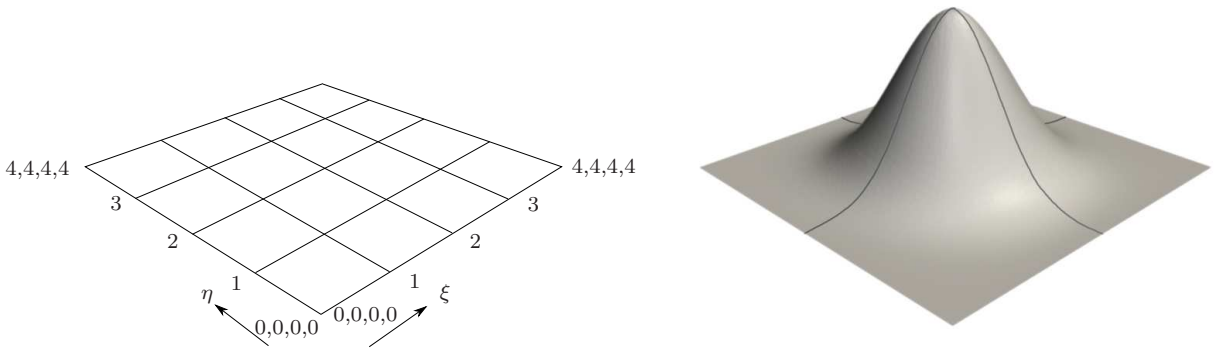
$B_{\boldsymbol{i},\boldsymbol{p}}(\boldsymbol{\xi})$ can be constructed as

$$B_{\boldsymbol{i},\boldsymbol{p}}\left(\boldsymbol{\xi}\right) = \prod_{\ell=1}^{d_p} N^{\ell}_{i_\ell,p_\ell}(\xi^\ell) \tag{5}$$

Multi-index $\boldsymbol{i} = \{i_1, \ldots, i_{d_p}\}$ denotes the position in the tensor product structure, $\boldsymbol{p} = \{p_1, \ldots, p_{d_p}\}$ indicates the polynomial degree, and $\boldsymbol{\xi} = \{\xi^1, \ldots, \xi^{d_p}\}$ are the parametric coordinates in each parametric direction ℓ. A bivariate parametric space and B-spline basis function are shown in Figs. 2a and 2b, respectively. B-spline surfaces ($d_p = 2$) and solids ($d_p = 3$) are a linear combination of multivariate B-spline basis functions and control points in the form

$$\boldsymbol{S}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}} \boldsymbol{P}_{\boldsymbol{i}}\, B_{\boldsymbol{i},\boldsymbol{p}}\left(\boldsymbol{\xi}\right) \tag{6}$$

where the sum is taken over all combinations of multi-index $\boldsymbol{i}$. In the multivariate case, the control points $\boldsymbol{P}_{\boldsymbol{i}} \in \mathbb{R}^{d_s}$ form the so-called control mesh.



(a) *Tensor product structure of open knot vectors in the parameter space.*

(b) *Cubic basis function generated from* Ξ=$\{0, 1, 2, 3, 4\}$ *in* ξ- *and* η-*directions.*

Figure 2: *Bivariate cubic knot spans and a corresponding uniform B-spline basis function.*

2.1.3. Non-uniform rational B-splines

NURBS can be obtained through a projective transformation of a corresponding B-spline in $\mathbb{R}^{d_s+1}$. Univariate NURBS basis functions $R_{i,p}(\xi)$ are given by

$$R_{i,p}\left(\xi\right) = \frac{w_i N_{i,p}(\xi)}{\sum_{j=1}^{n} w_j N_{j,p}(\xi)} \tag{7}$$

where $N_{i,p}(\xi)$ are polynomial B-spline basis functions and w_i are weights. Multivariate NURBS basis functions are formed as

$$R_{\boldsymbol{i},\boldsymbol{p}}\left(\boldsymbol{\xi}\right) = \frac{w_{\boldsymbol{i}} B_{\boldsymbol{i},\boldsymbol{p}}(\boldsymbol{\xi})}{\sum_{\boldsymbol{j}} w_{\boldsymbol{j}} B_{\boldsymbol{j},\boldsymbol{p}}(\boldsymbol{\xi})} \tag{8}$$

NURBS curves, surfaces and solids are then defined as

$$\boldsymbol{S}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}} \boldsymbol{P}_{\boldsymbol{i}}\, R_{\boldsymbol{i},\boldsymbol{p}}\left(\boldsymbol{\xi}\right) \tag{9}$$

Suitable control points and weights for arbitrarily complex geometries can be derived with and exported from CAGD tools such as Rhino [4, 75].

2.2. The variational background of collocation

The collocation method can be considered as a member of the family of numerical schemes based on the method of weighted residuals [2, 76–79]. In what follows, we focus on the boundary value problem associated with steady advective-diffusive transport, defined in

strong form as

$$\mathscr{L}(u) \equiv \boldsymbol{a} \cdot \nabla u - \nabla \cdot (D \nabla u) = f \qquad \text{in } \Omega \tag{10a}$$

$$u = u_D \qquad \text{on } \Gamma_D \tag{10b}$$

$$\boldsymbol{n} \cdot D \nabla u = h \qquad \text{on } \Gamma_N \tag{10c}$$

where $\mathscr{L}$ denotes the advection-diffusion operator, $u(\boldsymbol{x})$ is the scalar unknown, $\boldsymbol{a}$ is the velocity, D is the diffusion coefficient and f is a source term. The function u_D specifies the solution of u on the Dirichlet boundary Γ_D, while function h specifies the normal diffusive flux on the Neumann boundary Γ_N. The unit outward normal along Γ is denoted by $\boldsymbol{n}$. The principles outlined in the following equivalently apply to other PDE systems as well, e.g. elasticity [46].

2.2.1. The method of weighted residuals

The method of weighted residuals (MWR) considers approximations u^* to the exact solution u of the form

$$u^* = \tilde{u}_D(\boldsymbol{x}) + \sum_{i=1}^{n} N_i(\boldsymbol{x})\, c_i \tag{11}$$

The function $\tilde{u}_D$ in Eq. (11) is viewed as an extension of the prescribed boundary condition, that is, it is defined on Ω and satisfies the Dirichlet boundary condition Eq. (10b) when evaluated on Γ_D. The remainder of Eq. (11) is chosen from a n-dimensional finite subspace $S_n = \text{span}(N_1(\boldsymbol{x}), ..., N_n(\boldsymbol{x}))$, spanned by the linearly independent basis functions N_i, and exactly satisfies the zero Dirichlet boundary a priori. The corresponding unknown coefficients c_i are determined in such a way that the residuals, which are obtained by the substitution of u^* into Eqs. (10a) and (10c), are zero in an average sense as follows

$$\int_\Omega (\mathscr{L}(u^*) - f)\ \omega_\Omega\ d\Omega \ + \ \int_{\Gamma_N} (\boldsymbol{n} \cdot D \nabla u^* - h)\ \omega_\Gamma\ d\Gamma \ = \ 0 \tag{12}$$

Functions $\omega_\Omega(\boldsymbol{x})$ and $\omega_\Gamma(\boldsymbol{x})$ are test functions that are defined over the domain Ω and the Neumann boundary Γ_N, respectively. Note that we do not need to consider the residual that emanates from Eq. (10b), since Dirichlet boundary conditions are exactly satisfied a priori by Eq. (11). Equation 12 constitutes the weighted-residual or weak form of the boundary value problem Eqs. (10).

2.2.2. Collocation

In the collocation method, test functions $\omega_\Omega(\boldsymbol{x})$ and $\omega_\Gamma(\boldsymbol{x})$ are selected as two sets of Dirac δ functions, which can be formally constructed as the limit of a sequence of smooth functions with compact support that converge to a distribution [45, 46], satisfying the so-called *sifting property*

$$\int_\Omega g_\Omega(\boldsymbol{x})\ \delta_\Omega(\boldsymbol{x}-\boldsymbol{x}_i)\ d\Omega \;=\; g_\Omega(\boldsymbol{x}_i) \tag{13}$$

$$\int_\Omega g_\Gamma(\boldsymbol{x})\ \delta_\Gamma(\boldsymbol{x}-\boldsymbol{x}_i)\ d\Omega \;=\; g_\Gamma(\boldsymbol{x}_i) \tag{14}$$

provided that g_Ω is a continuous function about the point $\boldsymbol{x}_i \in \Omega$ and g_Γ is a continuous function on the boundary about the point $\boldsymbol{x}_i \in \Gamma$.

In collocation, the Dirac δ test functions are defined at k interior points in Ω with coordinates $\boldsymbol{x}_i, i = 1, \ldots, k$, and $n-k$ boundary points on Γ_N with coordinates $\boldsymbol{x}_i, i = k+1, \ldots, n$, and read

$$\omega_\Omega = \sum_{i=1}^{k} \delta_\Omega(\boldsymbol{x}-\boldsymbol{x}_i)\ \hat{c}_i \tag{15}$$

$$\omega_\Gamma = \sum_{i=k+1}^{n} \delta_\Gamma(\boldsymbol{x}-\boldsymbol{x}_i)\ \hat{c}_i \tag{16}$$

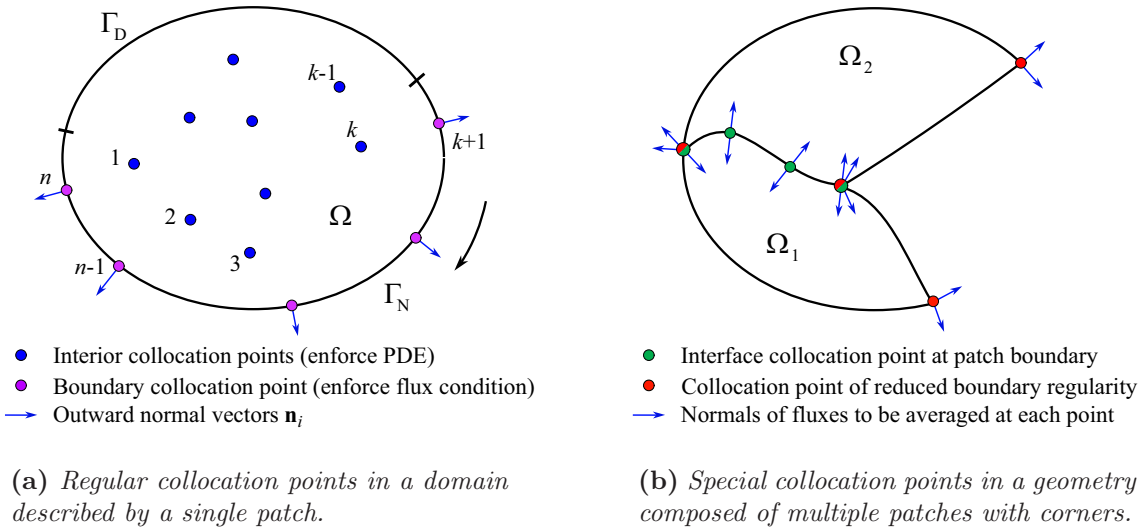
where n denotes the total number of basis functions. The locations of the Dirac δ functions are called collocation points and are illustrated in Fig. 3a. Substitution of Eqs. (15) and (16) into the weak form Eq. (12) yields

$$\sum_{i=1}^{k} \hat{c}_i \left(\mathscr{L}\left[\ u_D(\boldsymbol{x}_i) + \sum_{j=1}^{k} N_j(\boldsymbol{x}_i)\ c_j\ \right] - f(\boldsymbol{x}_i) \right) + \sum_{i=k+1}^{n} \hat{c}_i \left(\boldsymbol{n}_i \cdot D \sum_{j=k+1}^{n} \nabla N_j(\boldsymbol{x}_i)\ c_j - h(\boldsymbol{x}_i) \right) = 0 \tag{17}$$

In this step, the integrals are naturally eliminated due to the sifting property, Eqs. (13) and (14), of the Dirac δ test functions. Collocation thus amounts to satisfying the strong form of the residual at the collocation points.

Since coefficients $\hat{c}_i$ are arbitrary, Eq. (17) yields a system of linear algebraic equations with unknowns c_j. The elements of the system matrix $\boldsymbol{K}$ and load vector $\boldsymbol{F}$ are defined as

$$K_{ij} = \begin{cases} \mathscr{L}\Big(N_j(\boldsymbol{x}_i) \Big), & \text{for } 1 \le i \le k \\ \boldsymbol{n}_i \cdot D \nabla N_j(\boldsymbol{x}_i), & \text{for } k+1 \le i \le n \end{cases} \tag{18}$$



(a) *Regular collocation points in a domain described by a single patch.*

(b) *Special collocation points in a geometry composed of multiple patches with corners.*

Figure 3: *IGA collocation: Schematic outline of different collocation points.*

$$F_i = \begin{cases} -\mathscr{L}\left(u_D(\boldsymbol{x}_i)\right) + f(\boldsymbol{x}_i), & \text{for } 1 \leq i \leq k \\ -\boldsymbol{n}_i \cdot D\nabla u_D(\boldsymbol{x}_i) + h(\boldsymbol{x}_i), & \text{for } k+1 \leq i \leq n \end{cases} \tag{19}$$

The system can be solved for the unknown coefficients c_j, which define the approximation u^* of Eq. (11). It should be noted that the system matrix of Eq. (18) is generally not symmetric.

Due to the evaluation of the differential operators of the PDE, collocation requires basis functions N_i with certain smoothness properties, so that higher derivatives are well-defined in the vicinity of each collocation point. In our numerical examples, we need to evaluate the second-order advection-diffusion operator $\mathscr{L}$ of Eq. (10), so that basis functions are required, which are at least C^2 at all interior collocation points and at least C^1 on the Neumann boundary. We note that there are many forms and instantiations of collocation methods, e.g. subdomain collocation [76, 79, 80], radial boundary collocation [81–83], or meshfree point collocation methods [84, 85].

2.2.3. Galerkin

To clarify the variational concept of collocation, it is instructive to compare it with the derivation of the Galerkin method [2, 79, 86, 87]. The Galerkin method adopts the same formal approximation u^* of the solution, i.e. Eq. (11). In contrast to (point) collocation, which uses a set of Dirac δ functions, the Galerkin method chooses test functions ω_Ω and ω_Γ, which are zero at Γ_D, but can otherwise be represented by the same basis functions as

the approximation of the solution in Eq. (11)

$$\omega_\Omega = \omega_\Gamma = \sum_{i=1}^{n} N_i(\boldsymbol{x})\ \hat{c}_i \tag{20}$$

The Galerkin solution is thus determined by forcing the residual to be orthogonal to each basis function ω_i in the following sense

$$\int_\Omega \Big(\mathscr{L}\left[u^*(\boldsymbol{x})\right] - f(\boldsymbol{x}) \Big) \omega_i \ d\Omega \ + \ \int_{\Gamma_N} \ \Big(\boldsymbol{n} \cdot D\nabla u^*(\boldsymbol{x}) \ - h(\boldsymbol{x}) \Big) \omega_i \ d\Gamma = 0 \tag{21}$$

This form still requires at least C^1-continuous basis functions or additional terms that handle the jumps in the derivatives of C^0-continuous basis functions. Applying integration by parts and the divergence theorem to the diffusion term, we obtain the standard well-known variational formulation of the Galerkin method [88], which is valid for C^0-continuous basis functions. The components of the system matrix K_{ij} and the load vector F_i are derived as

$$K_{ij} \ = \ \int_\Omega N_i \Big(\boldsymbol{a} \cdot \nabla N_j \Big) d\Omega \ + \ \int_\Omega \nabla N_i \cdot \Big(D \nabla N_j \Big) d\Omega \tag{22}$$

$$F_i \ = \ \int_\Omega N_i f \ d\Omega \ + \ \int_{\Gamma_N} N_i h \ d\Gamma \tag{23}$$

A comparison with the corresponding terms of the collocation method in Eq. (18) and (19) reveals two major technical differences between the two methods. First, due to integration by parts in Eq. (21), Galerkin reduces the minimum continuity required by the basis functions N_i to C^0, which opens the door to standard C^0-continuous FEA technology. In collocation, however, the required smoothness corresponds to the highest differential operator in the strong form of the PDE, which calls for a minimum continuity of C^2 in the present case. Note that C^1 is sufficient for collocation points that are not located at the knots.

Second, due to the finite support of the Galerkin test functions, integrals are not eliminated from the discretized variational statement as they are in the collocation method due to the sifting property of the Dirac δ test functions. This requires their evaluation by numerical quadrature rules of the form

$$\int_\Omega g(\boldsymbol{x}) \ d\Omega \ = \ \sum_k g(\boldsymbol{x}_k) w_k \tag{24}$$

which replace the continuous integral by several point evaluations multiplied by correspond-

ing weights w_k in each element. Consequently, the computational effort of evaluating the Galerkin matrix and load vector is considerably increased as compared to collocation, which operates with the optimum of only one point evaluation for each basis function N_i.

2.3. Isogeometric collocation

Isogeometric collocation emanates from the combination of isogeometric basis function technology and the collocation method as described in Sections 2.1 and 2.2, respectively. Since smooth B-splines and NURBS naturally comply with the continuity requirements of collocation, the use of IGA in a collocation framework suggests itself. Beyond the essential property of smoothness, IGA basis functions come with an apparatus of optimized algorithms that allow for a highly formalized and efficient implementation and can handle very complex geometries [71, 73, 74]. The collocation method brings to the table its robustness and computational efficiency in terms of a minimum of point evaluations. In the following, we briefly summarize the key technical aspects of IGA collocation:

- *Collocation at Greville abscissae:* The success of a collocation method predominantly relies on the choice of suitable collocation points. In the framework of B-spline based collocation methods, different sets of points have been proposed that lead to collocation schemes with different stability and convergence properties. Examples are orthogonal collocation on Gauss-type quadrature points [56, 89], the maxima of spline basis functions [57, 90], and the Demko points [45, 91]. In the present study, we use the Greville abscissae, which are reported to be a very good choice from a practical engineering point of view [45, 46, 59, 90]. Collocation points based on the Greville abscissae have been shown to provide the best possible rates of convergence, while they have been found to be unstable only for very unusual cases [45, 46]. A Greville abscissa in 1D can be easily computed from a knot vector $\Xi = \{\xi_1, \xi_2, \ldots, \xi_{n+p+1}\}$ as

$$\hat{\xi}_i = \frac{\xi_{i+1} + \ldots + \xi_{i+p}}{p}, \qquad i = 1, \ldots, n \tag{25}$$

  where n denotes the number of basis functions in the patch. Equation (25) automatically produces the optimal number of points, so that each point can be associated with one particular basis function. Furthermore, favorable properties also hold for higher dimensional patches, for which Greville points can be easily generated by taking the tensor product of the 1D Greville abscissae of each parametric direction.

- *Imposition of boundary conditions:* Boundary conditions directly follow from the variational formulation derived in Section 2.2. Dirichlet boundary conditions are satisfied strongly by incorporating them into the approximation u^* of Eq. (11). Therefore, Greville points located on Γ_D do not need to be taken into account. Neumann boundary conditions are imposed by the evaluation of the normal diffusive flux at the $n - k$ boundary collocation points on Γ_N, see Fig. 3a and Eqs. (17) to (19).

- *Multi-patch geometries:* Geometric parameterizations in CAGD are often composed of several conforming NURBS patches, where the smoothness of basis functions is reduced to C^0 along patch interfaces. C^0 continuity is the physically appropriate condition at material interfaces. It is shown in [46] that IGA collocation can handle multi-patch parameterizations by introducing Neumann-type flux conditions for collocation points on the interface, which can be handled in the same way as Neumann boundary conditions (see Fig. 3b). This allows the collocation equations of each patch to be constructed individually, and the global system can be completed by simply summing up the equations associated with interface collocation points shared by multiple patches. Neumann-type interface conditions for different multi-patch scenarios are tabulated in [46].

- *Reduced regularity of the geometric boundary:* The evaluation of flux conditions at boundary and interface collocation points involve a well-defined normal vector $\boldsymbol{n}$, which requires a regular boundary curve that is at least C^1. It is shown in [46] that collocation points that lie on boundary locations of reduced regularity such as corners or sharp edges can be treated as follows: The flux condition is evaluated several times for all normal vectors that are well-defined in the neighborhood of the critical collocation point. The final contribution to the system matrix is simply their average (see also Fig. 3b). Averaging rules for different irregular points are tabulated in [46].

Readers interested in a more detailed presentation and the theoretical background of these issues are referred to the fundamental contributions in [45, 46].

3. Comparison of isogeometric collocation with isogeometric Galerkin and C^0 finite element methods in terms of computational efficiency

In the following, we provide an extensive comparison of isogeometric collocation (IGA-C) with isogeometric Galerkin (IGA-G) and standard C^0 finite element methods (FEA-G) in terms of their computational efficiency. In this section, we will quantify the efficiency of a method by estimating the computational cost of its main algorithms that we measure in terms of the number of floating point operations (flops) involved as well as by computing times measured with our codes. When looking at flops, we adopt the corresponding operation counts as a suitable indicator of the actual computing time. The present section will focus on three main aspects: (a) Cost for the formation and assembly of stiffness matrices and residual vectors; (b) cost for the direct and iterative solution of systems of algebraic equations; and (c) accuracy in error norms vs. the total number of degrees of freedom as well as vs. the total computing time that includes formation/assembly, preconditioning and iterative solution. The results demonstrate the potential of IGA-C to achieve a specified level of accuracy with a computational effort that is orders of magnitude smaller than for IGA-G and FEA-G.

3.1. Cost for the formation and assembly of stiffness matrices and residual vectors

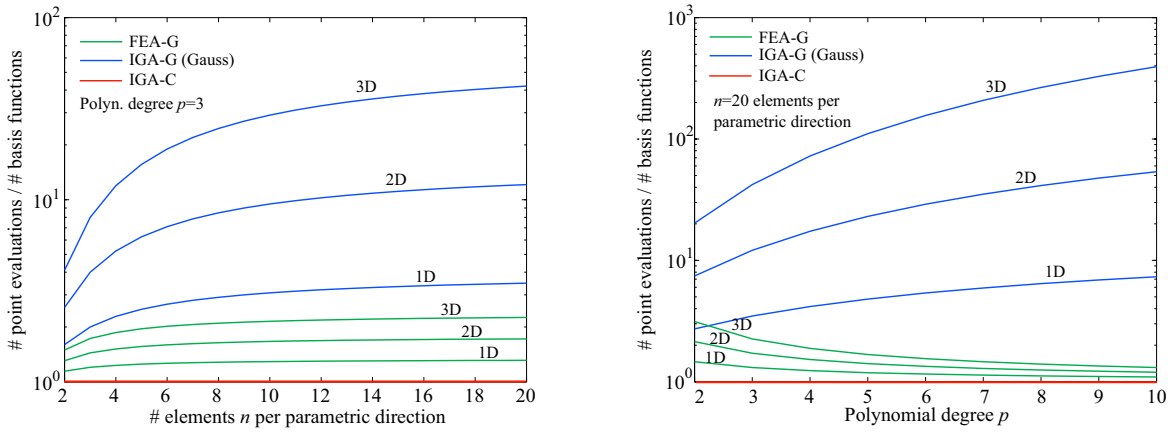
The cost required for the formation and assembly of stiffness matrices and residual vectors is governed by the number of quadrature/collocation points and the algorithmic operations required at each of these points. We consider model discretizations in one, two and three dimensions that are characterized by the polynomial degree p of the basis functions and the number of elements n in each parametric direction. For the sake of clarity and simplicity, we assume throughout this section that the model discretizations in 2D and 3D have the same number of elements n in each parametric direction. The spatial dimension of the model discretizations will be denoted by parameter d. We refer to the domains delineated by knot spans as Bézier elements, or knot span elements, or simply as elements. These terms are used synonymously. We also note that this terminology is consistent with the usual notion of a finite element in which case the (typically C^0) element boundaries are the knots, thus pertaining to spline-based methods as well as traditional finite element methods. For IGA-G and IGA-C, we employ C^{p-1}-continuous NURBS basis functions defined by a single patch [1, 2]. For FEA-G, we use elements based on Bernstein polynomials [37, 74, 92]. We note that many of our results and conclusions equivalently hold for other C^0 approximations, such as basis functions based on Lagrange [88] or integrated Legendre polynomials [93], since their functions have the same support and span the same space as Bernstein polynomials. It

	IGA-C	IGA-G (Gauss)	IGA-G (optimal)	FEA-G
Total #	$(n+p)^d$	$n^d(p+1)^d$	$(n+1)^d(p/2+1)^d$	$n^d(p+1)^d$
# per basis function	$\frac{(n+p)^d}{(n+p)^d}=1$	$\frac{n^d(p+1)^d}{(n+p)^d}$	$\frac{(n+1)^d(p/2+1)^d}{(n+p)^d}$	$\frac{n^d(p+1)^d}{(np+1)^d}$
# per basis function as $n \gg p$	1	$(p+1)^d$	$(p/2+1)^d$	$\left(\frac{p+1}{p}\right)^d \approx 1$

Table 1: *Number of quadrature/collocation points in the model discretizations.*

should be kept in mind that each element of a C^0-continuous discretization introduces more independent degrees of freedom than a corresponding knot span element of a C^{p-1}-continuous NURBS discretization. To avoid a potential bias, we will mostly consider operation counts per basis function or per degree of freedom.

In IGA-C, the cost of formation and assembly is dominated by the evaluation of interior collocation points, since the number of points in the interior of a spline patch is one order of magnitude larger then the number of boundary points on the Neumann boundary. In addition, the computation of interior points is more expensive, since it involves second derivatives. For the sake of simplicity and clarity, we assume in our counts that each collocation point is an interior point.



(a) *Constant polynomial degree p=3, increasing number of elements n.*

(b) *Constant number of elements n=20, increasing polynomial degree p.*

Figure 4: *Number of quadrature points per basis function in 1D, 2D and 3D. The curves for IGA-G can be reduced by factor $1/2^d$ by using the improved quadrature rules of [44].*

d	IGA-C	IGA-G	FEA-G
1	$35(p+1)+2$	$2(p+1)^2+14(p+1)$	$2(p+1)^2+4(p+1)$
2	$125(p+1)^2+37$	$4(p+1)^4+35(p+1)^2+4$	$4(p+1)^4+18(p+1)^2+4$
3	$304(p+1)^3+223$	$6(p+1)^6+65(p+1)^3+20$	$6(p+1)^6+41(p+1)^3+20$

Table 2: *Cost in flops for the formation and assembly of the local stiffness matrix at one quadrature point (IGA-G/FEA-G) and at one collocation point (IGA-C) in a scalar problem (Laplace). A detailed derivation is provided in Appendix A.2.*

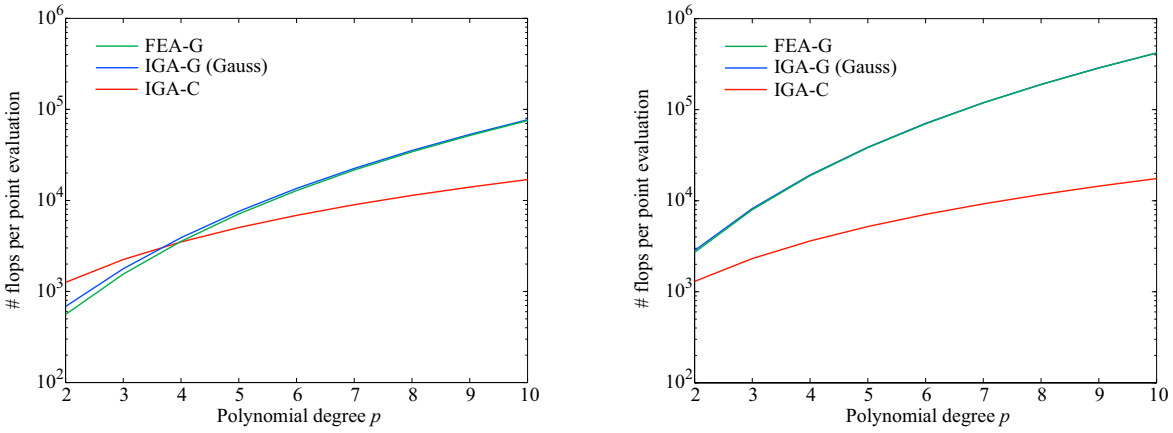
d	IGA-C	IGA-G	FEA-G
1	$36(p+1)+2$	$3(p+1)^2+14(p+1)$	$3(p+1)^2+4(p+1)$
2	$136(p+1)^2+37$	$24(p+1)^4+69(p+1)^2+4$	$24(p+1)^4+52(p+1)^2+4$
3	$323(p+1)^3+223$	$108(p+1)^6+278(p+1)^3+20$	$108(p+1)^6+254(p+1)^3+20$

Table 3: *Cost in flops for the formation and assembly of the local stiffness matrix at one quadrature point (IGA-G/FEA-G) and at one collocation point (IGA-C) in a vector problem (elasticity). A detailed derivation is provided in Appendix A.3.*

3.1.1. Number of quadrature/collocation points

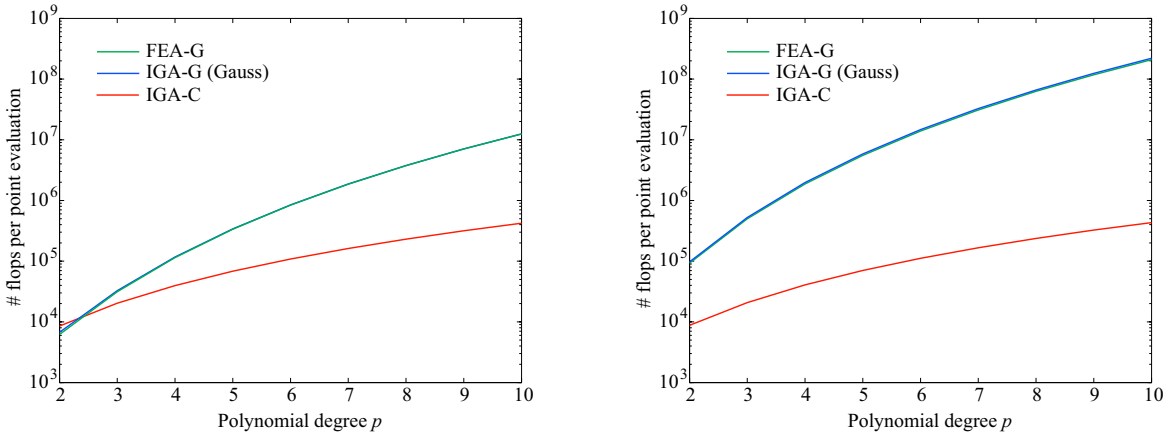
The number of quadrature/collocation points in the model discretizations depends on n, p and d. Corresponding counts are given in the first row of Table 1 for IGA-C, IGA-G and FEA-G. The counts for the Galerkin based methods are based on full Gauss quadrature [88], which is optimal for FEA-G. For IGA-G, we also report counts based on the nearly optimal quadrature rule recently given in [44]. The latter takes into account the inter-element continuity of smooth NURBS basis functions and leads to a reduction of point evaluations by a factor of about $1/2^d$.

More significant for the efficiency of a method is the ratio between number of quadrature or collocation points and the number of basis functions, since this relates the number of point evaluations to the approximation power of the basis. Note that an equivalent measure is the number of point evaluations per control point or node of the IGA or FEA mesh, respectively. The corresponding counts are given in the second row of Table 1. In the third row, we state the asymptotic numbers for the case of $n \gg p$. We plot the number of quadrature/collocation points per basis function for increasing n and p in Fig. 4a and 4b, respectively. We observe that with respect to the optimum ratio of one obtained for IGA-C, the number of point evaluations per basis function is slightly higher in FEA-G, eventually converging to a value close to the optimum of one for a large n and p. For IGA-G (Gauss), however, the number of point evaluations per basis function is one to two orders of magnitude larger, asymptotically converging to values far away from one.



(a) *Scalar problem (Laplace).* **(b)** *Vector problem (elasticity).*

Figure 5: *2D case: Cost for the formation and assembly of the local stiffness matrix per collocation/quadrature point in flops.*



(a) *Scalar problem (Laplace).* **(b)** *Vector problem (elasticity).*

Figure 6: *3D case: Cost for the formation and assembly of the local stiffness matrix per collocation/quadrature point in flops.*

3.1.2. Cost of formation and assembly at one quadrature/collocation point

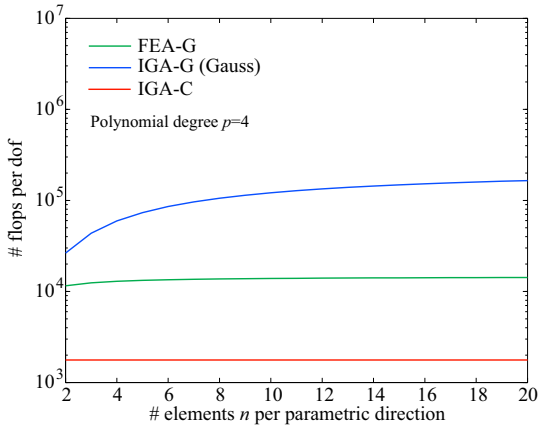
The small number of point evaluations is only one advantage of IGA-C. The second important aspect with respect to computational efficiency is the cost for the evaluation of local element arrays (e.g., stiffness matrices and residual vectors) at each quadrature/collocation point. In IGA-G and FEA-G, handling the local element arrays can be considered a two-step process of "form and assemble". The term formation refers to their construction by the algorithms in the element subroutines. The term assembly refers to the placement of

the local arrays in the global arrays by the assembly subroutine. In IGA-C, the local array resulting from one collocation point contains all entries of one row of the global array. It is therefore more efficient when at each collocation point the local subroutine directly operates on the corresponding row of the global matrix.

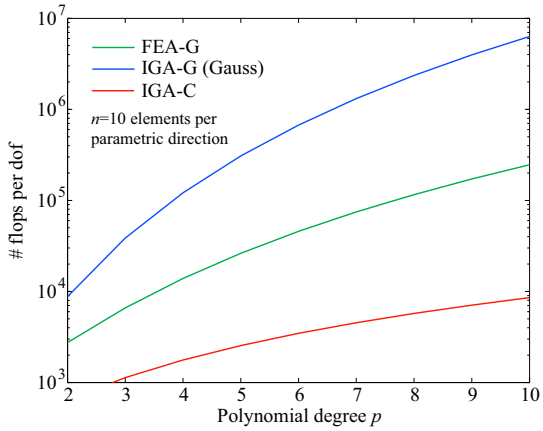
We will show in the following that in IGA-C the cost for the formation and assembly of the local stiffness matrix at each collocation/quadrature point is considerably smaller than in IGA-G and FEA-G. To illustrate that, we consider two model PDEs, i.e. the scalar Laplace equation and the vector equations of linear elasticity, and count the floating point operations (flops) required at one quadrature point in IGA-G and FEA-G and at one collocation point in IGA-C. We note that in this paper each multiplication and each addition is considered as one full floating point operation. We assume that IGA-C and IGA-G use NURBS to exactly represent the model geometry, and that FEA-G uses the finite element mesh for the approximation of the model geometry. Furthermore, we neglect the cost of all control structures and do not use the symmetry of the Galerkin matrices, since it does not hold for non-symmetric problems such as advection-diffusion. We note that the use of symmetry in Galerkin methods can decrease the operations required at each quadrature point, since matrix-matrix products can be reduced to the formation of the upper triangular part of the local stiffness matrix. However, the potential savings are less than half the operations at each quadrature point, since the upper triangular matrix contains more than half of the matrix entries and the expense for the computation of the basis functions and their gradients remains unchanged. Tables 2 and 3 report the corresponding operation counts per point evaluation for the case of the Laplace and elasticity problem, respectively. A detailed derivation of these relations can be found in Appendix A. The evaluation of operation counts for different polynomial degrees p is illustrated in Figs. 5 and 6 for the 2D and 3D cases, respectively.

d	IGA-C	IGA-G (Gauss)	FEA-G
1	$36(p+1)+2$	$\frac{n(p+1)}{(n+p)}\left(3(p+1)^2+14(p+1)\right)$	$\frac{n(p+1)}{(np+1)}\left(3(p+1)^2+4(p+1)\right)$
2	$\frac{136(p+1)^2+37}{2}$	$\frac{n^2(p+1)^2}{2(n+p)^2}\left(24(p+1)^4+69(p+1)^2\right)$	$\frac{n^2(p+1)^2}{2(np+1)^2}\left(24(p+1)^4+52(p+1)^2\right)$
3	$\frac{323(p+1)^3+223}{3}$	$\frac{n^3(p+1)^3}{3(n+p)^3}\left(108(p+1)^6+278(p+1)^3\right)$	$\frac{n^3(p+1)^3}{3(np+1)^3}\left(108(p+1)^6+254(p+1)^3\right)$

Table 4: *Total cost per degree of freedom in flops for the formation and assembly of the global stiffness matrix in elasticity.*

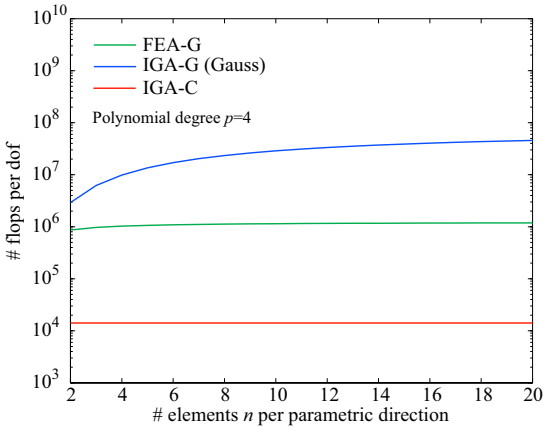


(a) *Constant polynomial degree p=4, increasing number of elements n.*

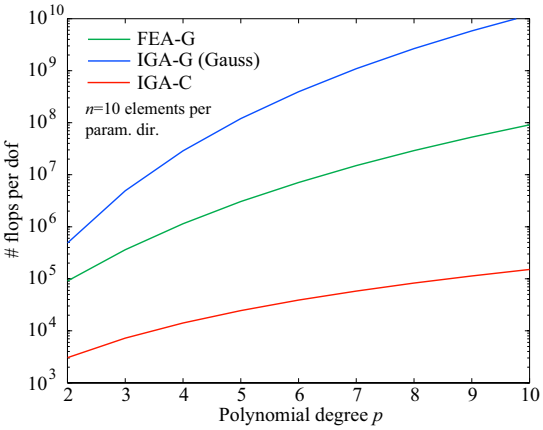


(b) *Constant number of elements n=10, increasing polynomial degree p.*

Figure 7: *2D elasticity: Cost per degree of freedom for the formation and assembly of the global stiffness matrix.*



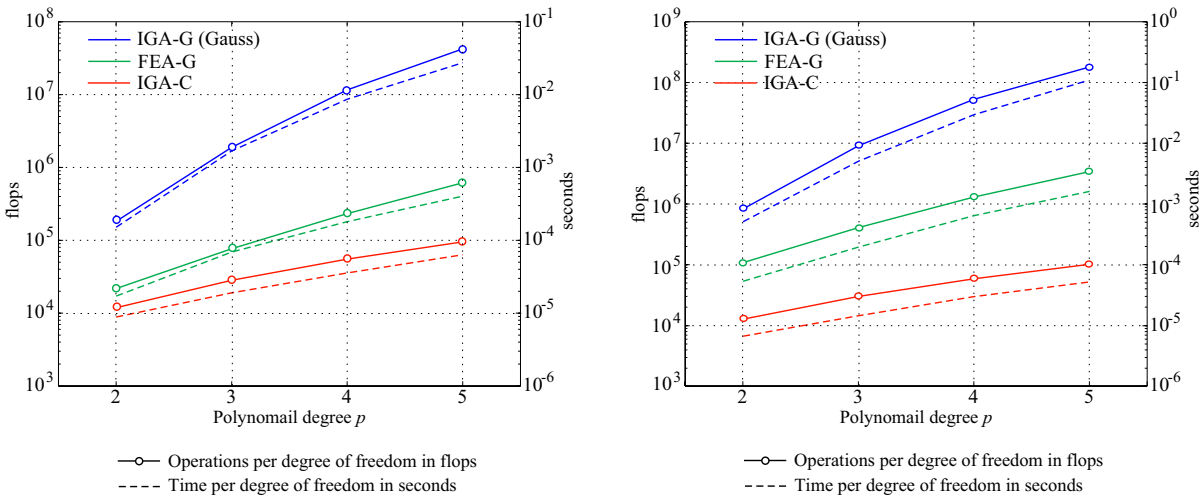
(a) *Constant polynomial degree p=4, increasing number of elements n.*



(b) *Constant number of elements n=10, increasing polynomial degree p.*

Figure 8: *3D elasticity: Cost per degree of freedom for the formation and assembly of the global stiffness matrix.*

We observe that the cost depends on the polynomial degree p of the basis functions and is of $\mathcal{O}(p^d)$ and of $\mathcal{O}(p^{2d})$ for IGA-C and IGA-G/FEA-G, respectively. A closer look at the tables given in Appendices A.2 and A.3 reveals that the parts of $\mathcal{O}(p^d)$ stem from the evaluation of the basis functions, which is more expensive in IGA-C due to the computation of the second order derivatives. Thus, the cost for collocation is practically invariant when we proceed from the scalar to the vector case (see Figs. 5 and 6), since the main expense stems



(a) *Laplace.* **(b)** *Elasticity.*

Figure 9: *Cost for formation and assembly of the global stiffness matrix per degree of freedom: Operation counts in flops vs. timings in seconds.*

from the computation of the basis functions and the cost for evaluating additional PDEs at the same collocation points is negligibly small. In Galerkin methods, the parts of $\mathcal{O}(p^{2d})$ arise from matrix-matrix products necessary for setting up the local stiffness matrix, which become increasingly expensive in the vector case (compare Figs. 5a/6a with 5b/6b). As a consequence, floating point operations at quadrature and collocation points are comparable for lower p and scalar problems. For larger p or in vector problems, however, the formation and assembly of local stiffness matrices at a quadrature point is orders of magnitude more expensive than the formation of a row of the global stiffness matrix at a collocation point.

3.1.3. Elasticity: Cost for formation and assembly of the global stiffness matrix

The full computational cost for the formation and assembly of global arrays can be obtained by multiplying the number of point evaluations with the expense required for one point evaluation itself. We first consider the formation and assembly of the global stiffness matrix in an elastic problem. In order to make the cost of IGA-C, IGA-G and FEA-G comparable, we normalize the total number of flops by the number of degrees of freedom of the corresponding model discretizations. The resulting costs per degree of freedom in flops are given in Table 4 and plotted in Figs. 7 and 8 for 2D and 3D cases, respectively, and clearly manifest the superiority of IGA-C. For all polynomial degrees p and mesh densities n considered, the cost of IGA-C is several orders of magnitude smaller than the cost required by IGA-G and FEA-G. For example, looking at the asymptotic costs for the quartic

3D discretizations shown in Fig. 8a, we observe that IGA-C requires almost two orders of magnitude fewer flops than FEA-G and about three and a half orders of magnitude fewer flops than IGA-G.

To make these relations more tangible, we transfer them to timings: If the total time for the formation and assembly of the global stiffness matrix of a given size takes one second in IGA-C, it will take almost two minutes in FEA-G and almost 1.5 hours in IGA-G to evaluate a global matrix of the same size. Our experience with test computations fully confirm these operation counts and timings. However, we note that matrices of the same size have different approximation power in IGA-C, IGA-G and FEA-G. It is therefore important to also consider accuracy vs. computing time, for which we refer to Section 3.3. To further corroborate the validity of our operation counts, we test the correlation of the cost for the formation and assembly of the global stiffness matrix vs. corresponding timings taken from our code. For each method, we timed computing time for the formation and assembly of a global stiffness matrix with a target size of about 30,000 degrees of freedom for p=2 through 5, and normalized the result by the number of degrees of freedom. Figure 9 shows that the operation counts per degree of freedom derived for the formation and assembly of the Laplace and elasticity matrices closely follow the trend of the corresponding timings.

3.1.4. Elastodynamics: Cost for an explicit time step

We examine the cost of one time step in an explicit Newmark predictor/multicorrector scheme [2, 46, 88]. For simplicity, we assume that the problem is linear elastic with constant density and no damping and that two explicit passes are sufficient. In this case, the lumped mass matrix can be computed beforehand, stored in a vector, and used throughout all time steps. We therefore assume that the lumped mass matrix is known. In each corrector step,

1. Predictor step: Compute $\boldsymbol{a}, \boldsymbol{u}$	6 GVOs
2. Compute residual vector: $\Delta \boldsymbol{F} = \boldsymbol{F}_{ext} - \boldsymbol{M}\boldsymbol{a} - \boldsymbol{K}\boldsymbol{u}$	1 FGR
3. Solve explicit system with lumped mass: $\boldsymbol{M}^* \Delta \boldsymbol{a} = \Delta \boldsymbol{F}$	1 GVO
4. First corrector step: Update $\boldsymbol{a} \mathrel{+}= \Delta \boldsymbol{a}$	1 GVO
5. Update residual with consistent mass: $\Delta \boldsymbol{F} \mathrel{-}= \boldsymbol{M} \Delta \boldsymbol{a}$	1 UGR
6. Solve explicit system with lumped mass: $\boldsymbol{M}^* \Delta \boldsymbol{a} = \Delta \boldsymbol{F}$	1 GVO
7. Second corrector step: Update $\boldsymbol{a} \mathrel{+}= \Delta \boldsymbol{a}$	1 GVO
8. Compute norm	4 GVOs
In total	14 GVOs + FGR + UGR

Table 5: *Operations for an explicit time step with two corrector passes. GVO, FGR and UGR denote a global vector operation, the formation of the global residual vector from local entities and the update of the global residual, respectively (see Table 6). +=/−= denote "add assignment" operators.*

d	IGA-C	IGA-G	FEA-G
	1. Global vector operation (GVO) (subtract/scalar multiply/etc.):		
1	$(n+p)$	$(n+p)$	$(np+1)$
2	$2(n+p)^2$	$2(n+p)^2$	$2(np+1)^2$
3	$3(n+p)^3$	$3(n+p)^3$	$3(np+1)^3$
	2. Formation and assembly of the global residual (FGR) (see Appendix A.4 for details):		
1	$(12(p+1)+12)(n+p)$	$n(23(p+1)^2+6(p+1))$	$n(13(p+1)^2+6(p+1))$
2	$(42(p+1)^2+119)(n+p)^2$	$n^2(57(p+1)^4+17(p+1)^2)$	$n^2(40(p+1)^4+17(p+1)^2)$
3	$(100(p+1)^3+573)(n+p)^3$	$n^3(104(p+1)^6+45(p+1)^3)$	$n^3(80(p+1)^6+45(p+1)^3)$
	3. Update of the global residual (UGR) (see step 4. of Appendix A.4):		
1	$(2(p+1)+2)(n+p)$	$n(5(p+1)^2+2(p+1))$	$n(5(p+1)+2(p+1))$
2	$(4(p+1)^2+4)(n+p)^2$	$n^2(10(p+1)^4+2(p+1)^2)$	$n^2(10(p+1)^4+2(p+1)^2)$
3	$(6(p+1)^3+6)(n+p)^3$	$n^3(15(p+1)^6+2(p+1)^3)$	$n^3(15(p+1)^6+2(p+1)^3)$

Table 6: *Number of floating point operations (flops) required for a global vector operation (GVO), formation and assembly of the global residual (FGR) and its update (UGR).*

the global residual vector is assembled by summing up the local contributions computed at each collocation point (IGA-C) and at all quadrature points of each element (IGA-G and FEA-G). This eliminates the need to store global vectors, and is therefore widely used in explicit codes such as LS-DYNA [94]. It requires the computation of local inertial, external and internal force vectors in the first corrector pass and the update of acceleration contributions to the residual on subsequent passes. For linear elasticity, the operations required for one explicit time step are summarized in Table 5.

The cost in flops for a global vector operation such as subtraction or scalar multiplication corresponds to the number of degrees of freedom in the model discretization under consideration. The cost for the formation and update of the residual vector per collocation/quadrature point are derived in Appendix A.4. For IGA-C we use an optimized algorithm to evaluate displacements and accelerations (see Algorithm 2 in Appendix A) that considerably reduces the number of linear system solves required for the computation of second order derivatives. For IGA-G and FEA-G we assume optimized linear algebra routines that avoid operations on zero entries of local matrices (see Appendix A.4). We multiply the flops per point evaluation given in Appendix A.4 with the total number of collocation/quadrature points to obtain the total cost in flops for the formation and update of the global residual vector (see Table 6). On this basis, we can compute the total number of flops per degree of freedom that are required for a two pass time step in the Newmark predictor/multicorrector scheme, using the information in Tables 1, 5 and 6. The resulting counts are summarized in Table 7.

In Figs. 10 and 11, we illustrate the time step cost per degree of freedom for 2D and

d	IGA-C	IGA-G	FEA-G
1	$14(p+1)+28$	$\frac{n(28(p+1)^2+8(p+1))}{(n+p)}+14$	$\frac{n(18(p+1)^2+8(p+1))}{np+1}+14$
2	$23(p+1)^2+76$	$\frac{n^2(67(p+1)^4+19(p+1)^2)}{2(n+p)^2}+14$	$\frac{n^2(50(p+1)^4+19(p+1)^2)}{2(np+1)^2}+14$
3	$35(p+1)^3+207$	$\frac{n^3(119(p+1)^6+47(p+1)^3)}{3(n+p)^3}+14$	$\frac{n^3(95(p+1)^6+47(p+1)^3)}{3(np+1)^3}+14$

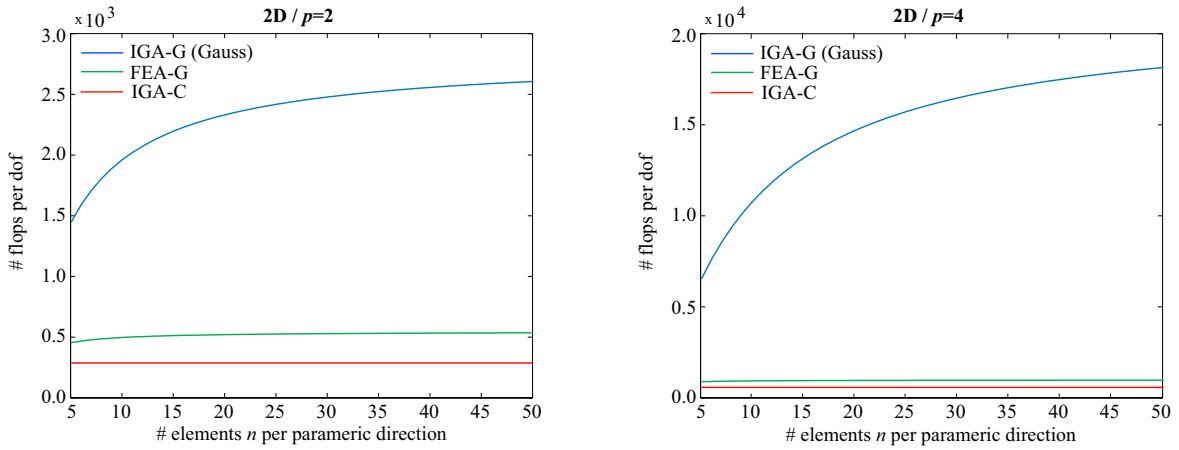
Table 7: *Total number of flops per degree of freedom for an explicit time step (2 corrector passes).*

3D model discretizations with quadratic and quartic basis functions. The combination of a minimum number of point evaluations with an inexpensive evaluation per point leads again to a significant reduction of the cost in IGA-C as compared to IGA-G. We observe in Fig. 11 that for 3D discretizations, the cost of IGA-C is between one and two orders of magnitude smaller than the cost of IGA-G. In FEA-G, the cost per quadrature point is slightly smaller than the cost per collocation point in IGA-C (see Appendix A.4). However, due to the optimal ratio of point evaluations to basis functions in collocation, the cost per degree of freedom is distinctly larger in FEA-G than in IGA-C. We note that if the point is to gain maximal speed, one could replace NURBS with B-splines. This would reduce the cost of IGA-C in 3D by an additional 15% with respect to what is shown here, and would further extend the lead of IGA-C with respect to FEA-G. We believe that in typical application fields of explicit dynamics such as car crash or metal forming, one would make many simplifications to minimize cost, and the B-splines simplification for analysis of a NURBS model would be probably first and foremost.

In addition, we emphasize that the cost of explicit dynamics also depends largely on the size of the critical time step. It has been shown in previous works (see for example [15]) that IGA allows for much larger stable time steps than FEA-G. Moreover, we remark that the high modes in FEA-G are notoriously ill-behaved (see for example [2, 8, 95]), which is not the case for IGA. On this basis, we can expect IGA-C to be significantly less expensive than FEA-G in an explicit elastodynamics setting.

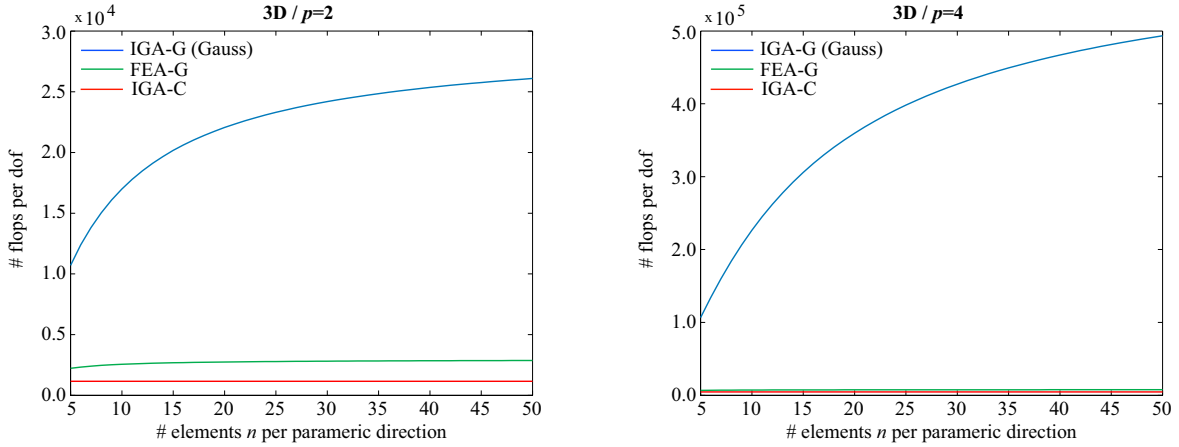
3.2. Cost of direct and iterative solvers

In the next step, we provide estimates of the computational cost necessary for the solution of discrete systems of algebraic equations. We do not focus on a specific solver implementation, but examine two key indicators, i.e. the bandwidth of the system matrix and the cost of global matrix-vector products, that characterize the computational efficiency of direct and iterative solvers, respectively. To this end, we consider a scalar problem (e.g. Laplace)



(a) *Quadratics in 2D.* **(b)** *Quartics in 2D.*

Figure 10: *2D linear elastodynamics: Cost per degree of freedom for an explicit time step with 2 corrector passes.*



(a) *Quadratics in 3D.* **(b)** *Quartics in 3D.*

Figure 11: *3D linear elastodynamics: Cost per degree of freedom for an explicit time step with 2 corrector passes.*

and corresponding model discretizations in 1D, 2D, and 3D. We assume a lexicographical ordering of the basis functions according to the parametric directions of the NURBS patch and the FEA mesh. To illustrate the principle of a lexicographical ordering scheme, we consider 3D tensor-product basis functions $\phi_{ijk} = N_i(\xi)\, N_j(\eta)\, N_k(\zeta)$. They are composed of one-dimensional components N_i, N_j and N_k in each parametric direction that are numbered consecutively from $i = 1 \dots n_\xi$, $j = 1 \dots n_\eta$ and $k = 1 \dots n_\zeta$, respectively, where n_ξ, n_η and n_ζ denote the total number of basis functions in each parametric direction. The

d	IGA-C (even p)	IGA-C (odd p)	IGA-G
1	$p/2$	$(p-1)/2$	p
2	$\frac{p\,(1+(n+p))}{2}$	$\frac{(p-1)(1+(n+p))}{2}$	$p(1+(n+p))$
3	$\frac{p(1+(n+p)+(n+p)^2)}{2}$	$\frac{(p-1)(1+(n+p)+(n+p)^2)}{2}$	$p(1+(n+p)+(n+p)^2)$

Table 8: *Lower/upper bandwidth (k_1/k_2) of the system matrix for a scalar problem. Due to the symmetry of the lexicographical ordering, $k_1 = k_2$ in the present case.*

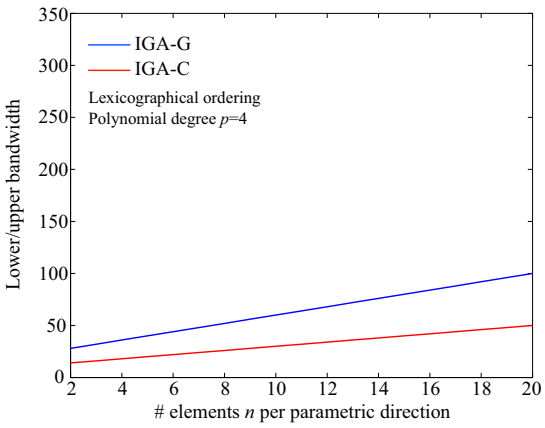
lexicographical number n_{bf} of a particular basis function ϕ_{ijk} can then be found as

$$n_{bf} = (k-1)n_\xi n_\eta + (j-1)n_\xi + i \tag{26}$$

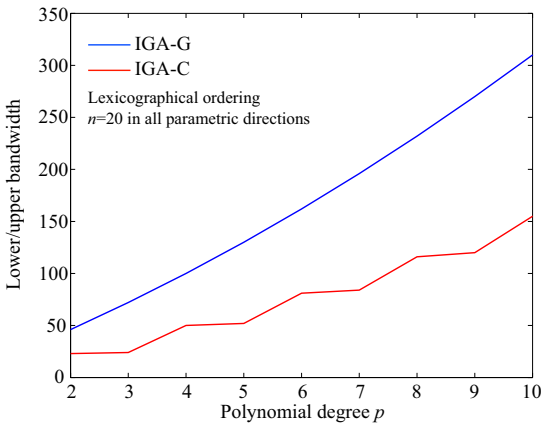
3.2.1. Bandwidth

System matrices in IGA-C, IGA-G and FEA-G are typically sparse matrices, in which the non-zero elements are restricted to a small band along the main diagonal of the matrix. The non-zero band of a matrix $A = [a_{ij}]$ is characterized by the lower and upper bandwidth, which are the smallest integers k_1 and k_2 such that $a_{ij} = 0$ for $i - j > k_1$ and $j - i > k_2$, respectively, and the bandwidth of A, which is $k_1 + k_2 + 1$ [96–98]. The classical LU decomposition factorizes a generally non-symmetric matrix as the product of a lower triangular matrix L and an upper triangular matrix U, from which the solution is found by forward elimination and back substitution. Classical operation counts for LU factorization of sparse banded matrices are of $\mathcal{O}(2\,n_{eq}\,k_1\,k_2)$ [96], where n_{eq} denotes the number of equations in the systems. While implementations of direct solvers greatly vary, the underlying main algorithm corresponds still to LU factorization and therefore, its trend applies to the best existing direct solvers [40]. The square of the bandwidth in the classical operation count indicates its important role for the performance of direct solvers.

Discretizations with smooth spline basis functions that exhibit maximum C^{p-1} continuity show a homogeneous bandwidth throughout the matrix. In this case it has been confirmed that the classical counts for LU factorization yield very accurate estimates that are close to the cost of modern sparse direct solver implementations [99]. Discretizations that arise from C^0 finite elements show a multi-block structure. For this case, more advanced direct techniques such as multi-frontal algorithms and static condensation can be used that require only a fraction of the classical operation counts, in particular if a high polynomial degree p is considered. This was illustrated recently by the study of Collier et al. [40] that compares

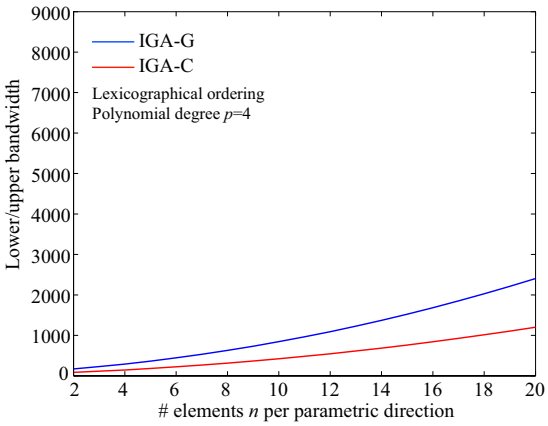


(a) *Constant polynomial degree p=4, increasing number of elements n.*

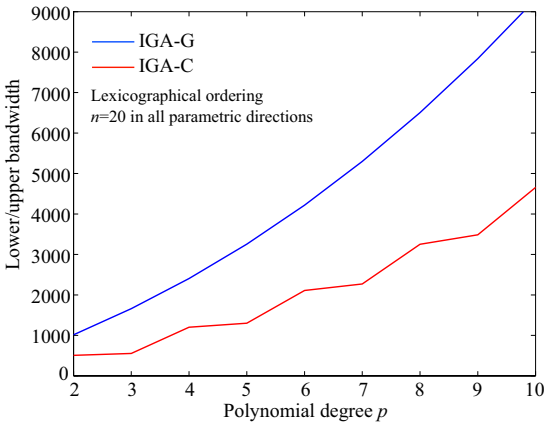


(b) *Constant number of elements n=10, increasing polynomial degree p.*

Figure 12: *Lower/upper bandwidth of the global stiffness matrix resulting from a 2D NURBS discretization of a scalar problem.*



(a) *Constant polynomial degree p=4, increasing number of elements n.*



(b) *Constant number of elements n=10, increasing polynomial degree p.*

Figure 13: *Lower/upper bandwidth of the global stiffness matrix resulting from a 3D NURBS discretization of a scalar problem.*

the performance of direct solvers for FEA-G and IGA-G.

Table 8 shows the lower/upper bandwidth in global stiffness matrices that result from the discretization of a scalar Laplace problem by IGA-C and IGA-G. Due to the symmetry of the lexicographical ordering of the spline basis functions, lower and upper bandwidths are equal. For IGA-C, the Greville abscissae yield collocation points that are located in the center of a knot span for even p and directly at a knot for odd p, so that the number of non-

IGA-C (even p)	IGA-C (odd p)	IGA-G	FEA-G (average)
$(p+1)^d$	p^d	$(2p+1)^d$	$(p+2)^d$

Table 9: *Population (number of non-zero entries) in each row of the global stiffness matrix that results from a discretization of a scalar problem.*

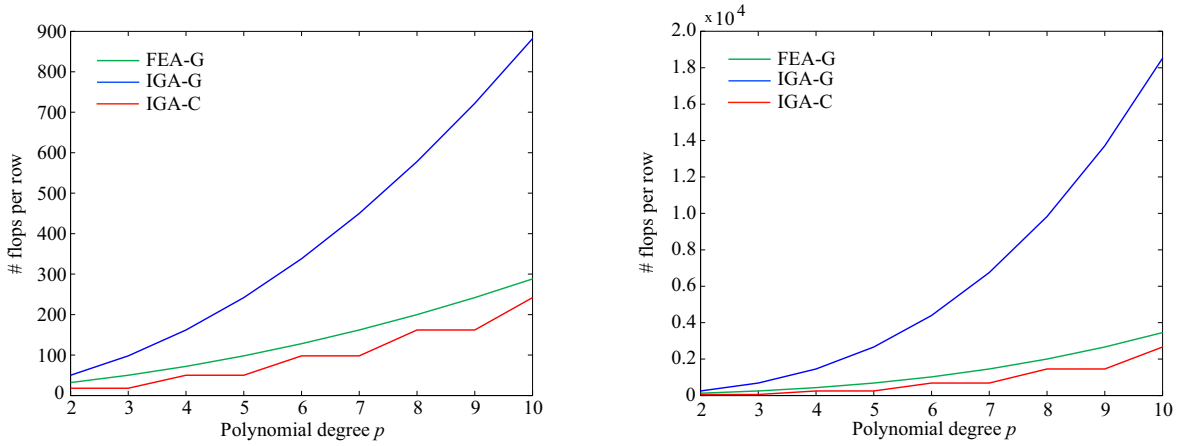
zero basis functions at a collocation point does not change for even and the next higher odd p. Therefore, spline basis functions of even polynomial degree lead to the same bandwidth as those of odd p. The lower/upper bandwidths for 2D and 3D NURBS discretizations with lexicographical ordering are illustrated in Figs. 12 and 13. We observe that IGA-G leads to bandwidths that are twice as large as those of IGA-C. Based on the classical operation counts for LU factorization, we can therefore expect a better performance of direct solvers for IGA-C than for IGA-G.

3.2.2. Cost of matrix-vector products

The performance of iterative solvers mainly depends on the cost of global matrix-vector products, which constitute the main expense during an iteration, and the conditioning of the system, which is of key importance for convergence with a minimum number of iterations [97]. Auricchio et al. [45] analyzed the eigenspectrum of collocation matrices and showed that their conditioning is comparable to those resulting from IGA-G. In particular, the upper part of the discrete eigenspectra in IGA-C and IGA-G is better behaved than in FEA-G, which results in better conditioned discrete systems. Despite the importance of conditioning, here we simply focus on the cost of matrix-vector products.

Table 9 shows the population of the global stiffness matrix, i.e. the number of non-zero entries in each row, for IGA-C, IGA-G and FEA-G. Due to the homogeneous structure of spline basis functions, the number of their interactions in IGA-C and IGA-G is constant except for the non-uniform basis functions at patch boundaries that involve repeated knots. However, their influence is negligible when discretizations of large n are considered. For IGA-C, one can again differentiate between collocation with basis functions of even and odd polynomial degree p due to the location of the collocation points. In FEA-G, basis functions can be classified into vertex, edge, face and interior functions, each of which interacts with a different amount of neighboring basis functions [41]. Using Table 3.1 given in [41], we compute the average population in FEA-G to be $(p+2)^d$.

For computing the matrix-vector product, one floating point operation is required for multiplication of each non-zero matrix entry with the corresponding vector component and one floating point operation for addition to the result. Using the data of Table 10, the



(a) *2D case.* **(b)** *3D case.*

Figure 14: *Cost of matrix-vector product per row (i.e. per degree of freedom). It constitutes a suitable indicator for the performance of iterative solvers.*

number of flops for the complete matrix-vector product operation can thus be computed by taking twice the average number of non-zero entries per row times the total number of rows in the matrix. Figures 14a and 14b plot the resulting cost per row (i.e. per degree of freedom) for IGA-C, IGA-G and FEA-G. We observe that IGA-C significantly reduces the cost for a matrix-vector product as compared to IGA-G, and is even slightly less expensive than FEA-G. Thus, all else being equal, we can expect that iterative solvers would perform considerably better for IGA-C than for IGA-G. This is what will be observed in the next section for timings taken from a preconditioned GMRES solver that is applied to solve systems of equations obtained with IGA-C, IGA-G and FEA-G.

3.3. Cost vs. accuracy

Finally, we assess IGA-C, IGA-G and FEA-G in terms of the computational cost required to achieve a specified level of accuracy. As a measure of accuracy, we use the relative error in the L^2 norm and the H^1 semi-norm. As a measure of cost, we use the total number of degrees of freedom as well as the serial computing time on a single processor. While the former is a good indicator for the approximation power and convergence properties of a method, the true computing time required to achieve a specified level of accuracy is the decisive question from a practical point of view.

3.3.1. A set of scalar and vector problems with smooth and "rough" solutions

We examine smooth and "rough" solutions of scalar and vector boundary value problems in three dimensions. As a representative scalar problem, we consider Poisson's equation in

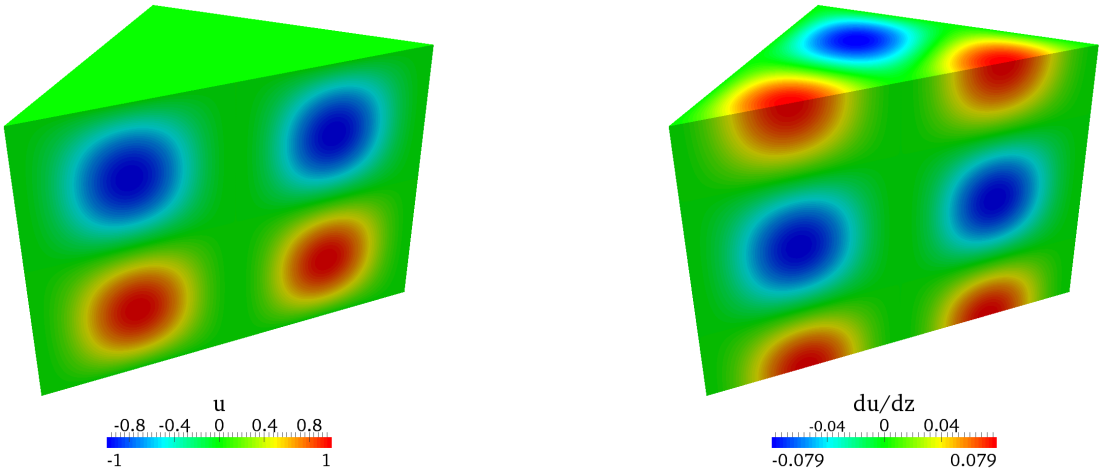


Figure 15: *Smooth solution of Poisson's problem (left) and its derivative with respect to the vertical direction (right).*

the framework of the following boundary value problem

$$-\Delta u = f \qquad \text{in } \Omega \tag{27a}$$

$$u = u_D \qquad \text{on } \Gamma_D \tag{27b}$$

$$\nabla u \cdot \boldsymbol{n} = h \qquad \text{on } \Gamma_N \tag{27c}$$

for which we assume exact smooth and rough solutions defined over the cube $\Omega = [0, 1]^3$. The corresponding smooth solution reads

$$u = \sin(2\pi x)\ \sin(2\pi y)\ \sin(2\pi z) \tag{28}$$

In accordance with Eq. (28), we can assume homogeneous Dirichlet boundary conditions over all surfaces of the cube. The exact smooth solution field and one of its first derivatives are plotted in Fig. 15. Insertion of Eq. (28) into the PDE Eq. (27a) yields the exact source term

$$f = 12\pi^2 \sin(2\pi x)\ \sin(2\pi y)\ \sin(2\pi z) \tag{29}$$

As a rough solution to Poisson's problem Eq. (27), we assume

$$u = xyz\left((x-1)^2 + (y-1)^2 + (z-1)^2\right)^{\frac{1}{4}} \tag{30}$$

Due to its exponent smaller than one, its derivatives exhibit a singularity in the corner $\{x, y, z\} = \{1, 1, 1\}$. We plot the exact solution field as well as one of its derivatives in

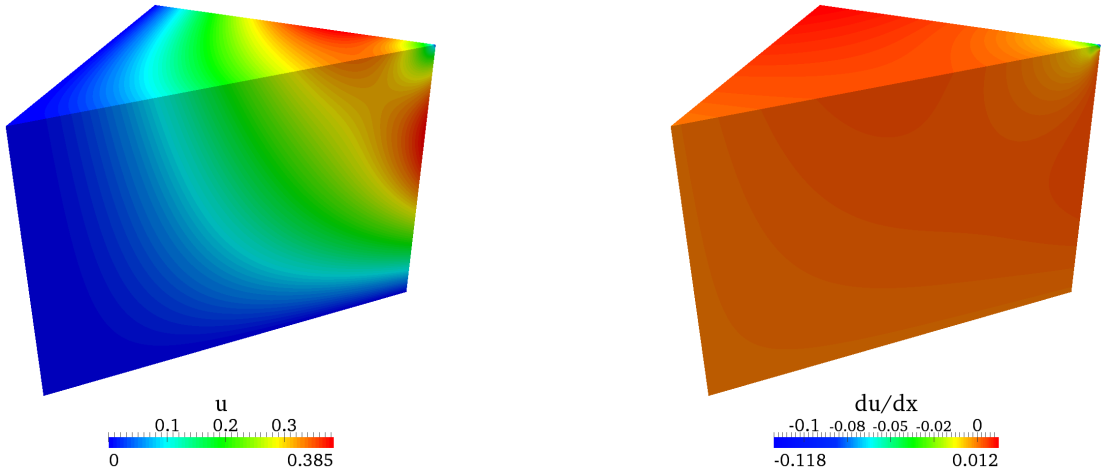


Figure 16: *Rough solution of Poisson's problem (left) and its derivative with respect to x (right). The coloring on the right is scaled to best illustrate the singularity in the corner.*

Fig. 16. Insertion of Eq. (30) into the PDE Eq. (27a) yields the exact source term

$$f = -\frac{15xyz - 4xz - 4xy - 4yz}{4\left((x-1)^2 + (y-1)^2 + (z-1)^2\right)^{\frac{3}{4}}} \tag{31}$$

We assume the following boundary conditions that are compatible to the exact solution Eq. (30). At the surfaces $x = 0$, $y = 0$ and $z = 0$ and at the corner $\{x, y, z\} = \{1, 1, 1\}$, we can again apply homogeneous Dirichlet constraints, while at the other surfaces we impose the following regular Neumann boundary conditions

$$\text{At } x = 1: \qquad h = yz\left((y-1)^2 + (z-1)^2\right)^{\frac{1}{4}} \tag{32a}$$

$$\text{At } y = 1: \qquad h = xz\left((x-1)^2 + (z-1)^2\right)^{\frac{1}{4}} \tag{32b}$$

$$\text{At } z = 1: \qquad h = xy\left((x-1)^2 + (y-1)^2\right)^{\frac{1}{4}} \tag{32c}$$

As a representative vector problem, we consider the PDE system of linear elasticity (see for example [88, 100]). To derive a set of exact smooth and rough solutions, we follow the same procedure as described for the scalar case of Poisson's problem.[1] In the scope of the present paper, we restrict ourselves to outline the main points. For the smooth and rough

[1]We encourage interested readers to contact the corresponding author, if they find the following information too sparse to fully reconstruct the derivation.

case, respectively, we assume the following two sets of displacement fields

$$u = v = w = \sin(2\pi x)\ \sin(2\pi y)\ \sin(2\pi z) \tag{33}$$

$$u = v = w = xyz\left((x-1)^2 + (y-1)^2 + (z-1)^2\right)^{\frac{1}{4}} \tag{34}$$

We choose the same solution fields for all displacement components in order to limit the number of terms of the resulting analytical quantities. Inserting Eqs. (33) and (34) into Navier's equations of elasticity [100] yields the body forces f_x, f_y and f_z for the smooth and rough cases, respectively. For the smooth case, compatibility with Eq. (33) allows the imposition of homogeneous Dirichlet boundary conditions. For the rough case, we need to partially impose Neumann boundary conditions at the surfaces $x = 1$, $y = 1$ and $z = 1$ that can be derived by inserting Eq. (34) in the strain-displacement and constitutive relations. For the computations, we assumed Young's modulus E=1 and Poisson's ratio ν=0.3.

We discretize the 3D cube by a structured grid whose elements either use C^{p-1} NURBS for IGA-C and IGA-G or C^0 Bernstein polynomials for FEA-G. For IGA-G, we use full Gauss quadrature in each element. All three methods are implemented within the same code, so that the only difference in the implementation of the three methods is in the formation and assembly of the stiffness matrix. The resulting system of equations is solved iteratively by a GMRES solver after preconditioning the stiffness matrix based on incomplete LU factorizations with zero fill-ins. The GMRES solver and the preconditioner are provided by Sandia's Trilinos packages AztecOO and Ifpack, respectively [101]. The timings[2] include the formation and assembly of the stiffness matrix and load vector, the preconditioning of the system of equations and its solution by the GMRES solver, but exclude all pre- and post-processing steps such as the computation of error norms. We consider four different polynomial degrees of the basis functions from quadratics (p=2) up to quintics (p=5). For each problem and each p, we first increase the number of degrees of freedom by uniform mesh refinement from about 200 to about 200,000 in each method, and record the relative errors in the L^2 norm and H^1 semi-norm. The convergence results with respect to the number of degrees of freedom are shown in Figs. 17, 19, 21, 23 for the smooth Poisson and elasticity problems, and in Figs. 25, 27, 29 and 31 for the rough Poisson and elasticity problems. For each problem and each p, we then increase the computing times by uniform mesh refinement from about 1 second to about 1,000 seconds, and record the relative errors. The L^2 norm and H^1 semi-norm. To ensure the reliability of the timings, we do not consider

[2]Using a single thread on a Intel(R) Xeon(R) W5590 @ 3.33GHz with 70 GB of RAM

Figure 17:

Smooth 3D Poisson:

L^2 error vs. degrees of freedom

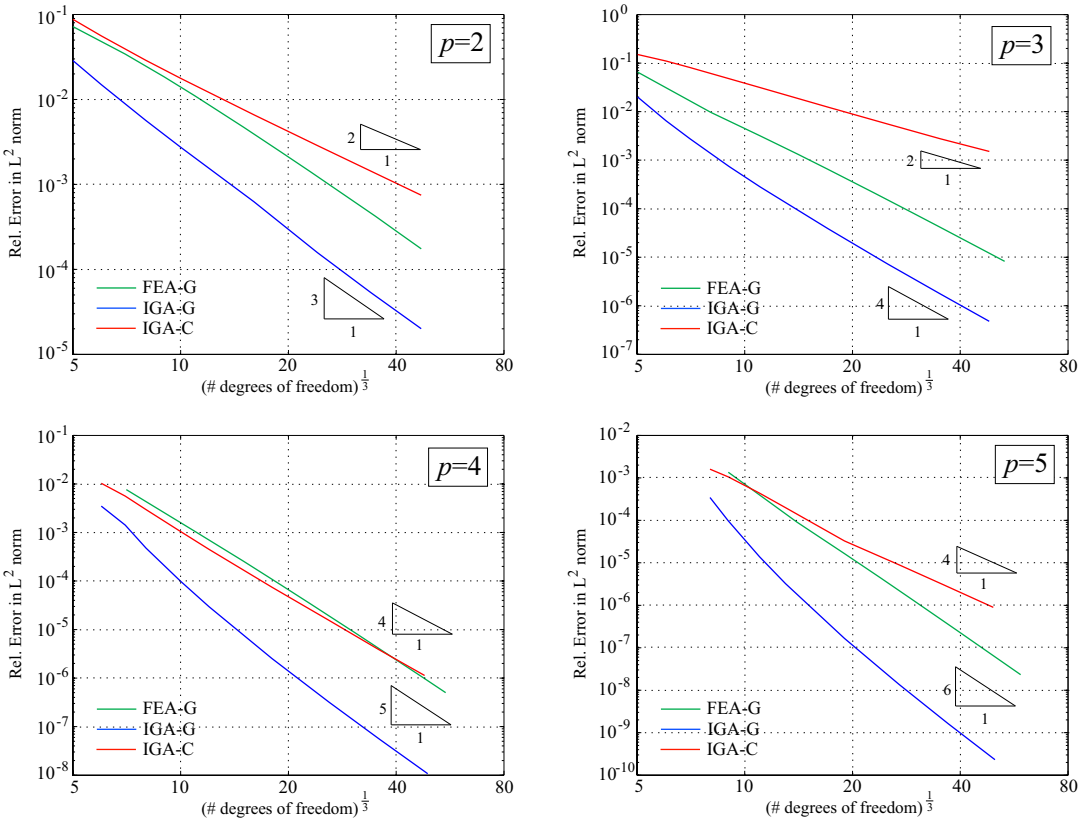


Figure 18:

Smooth 3D Poisson:

L^2 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

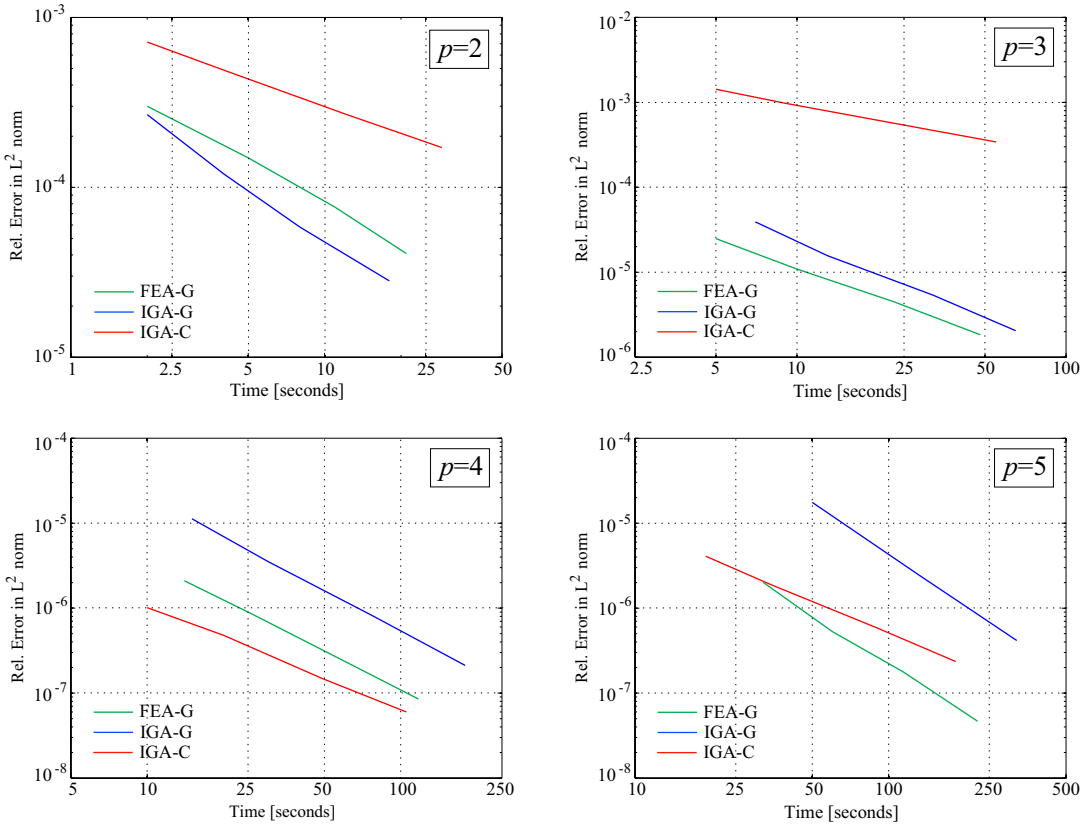


Figure 19:

Smooth 3D Poisson:

H^1 error vs. degrees of freedom

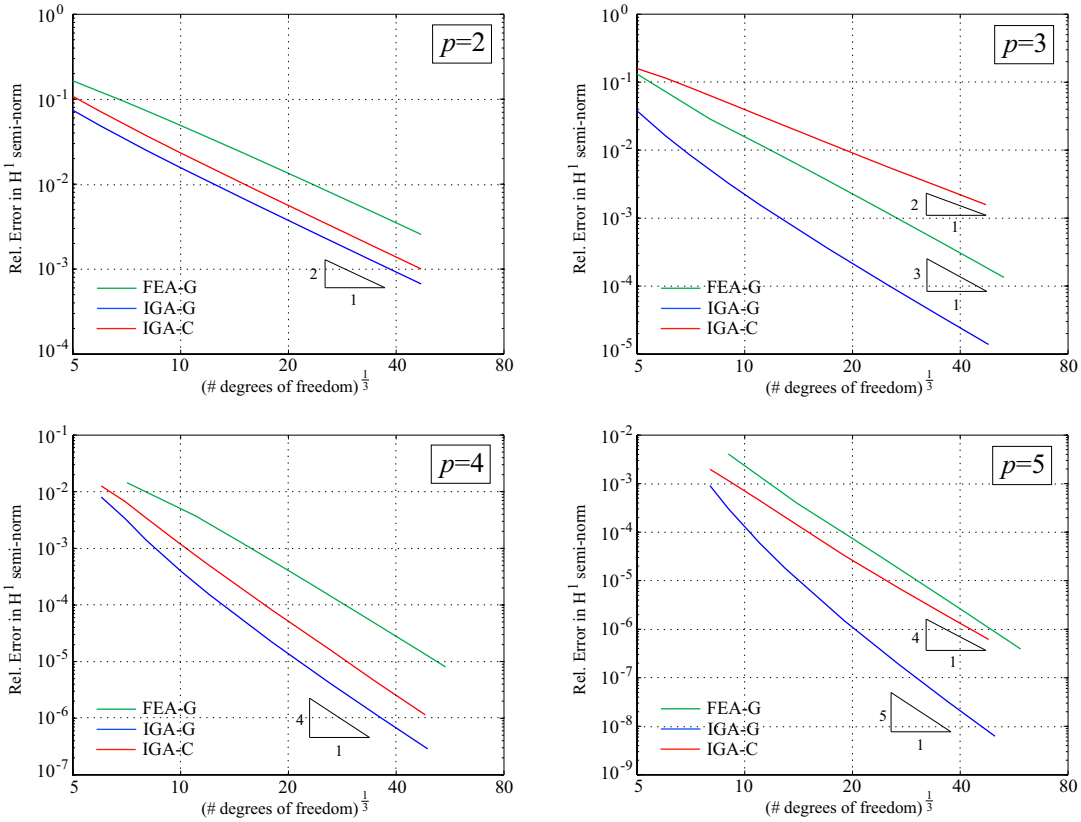


Figure 20:

Smooth 3D Poisson:

H^1 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

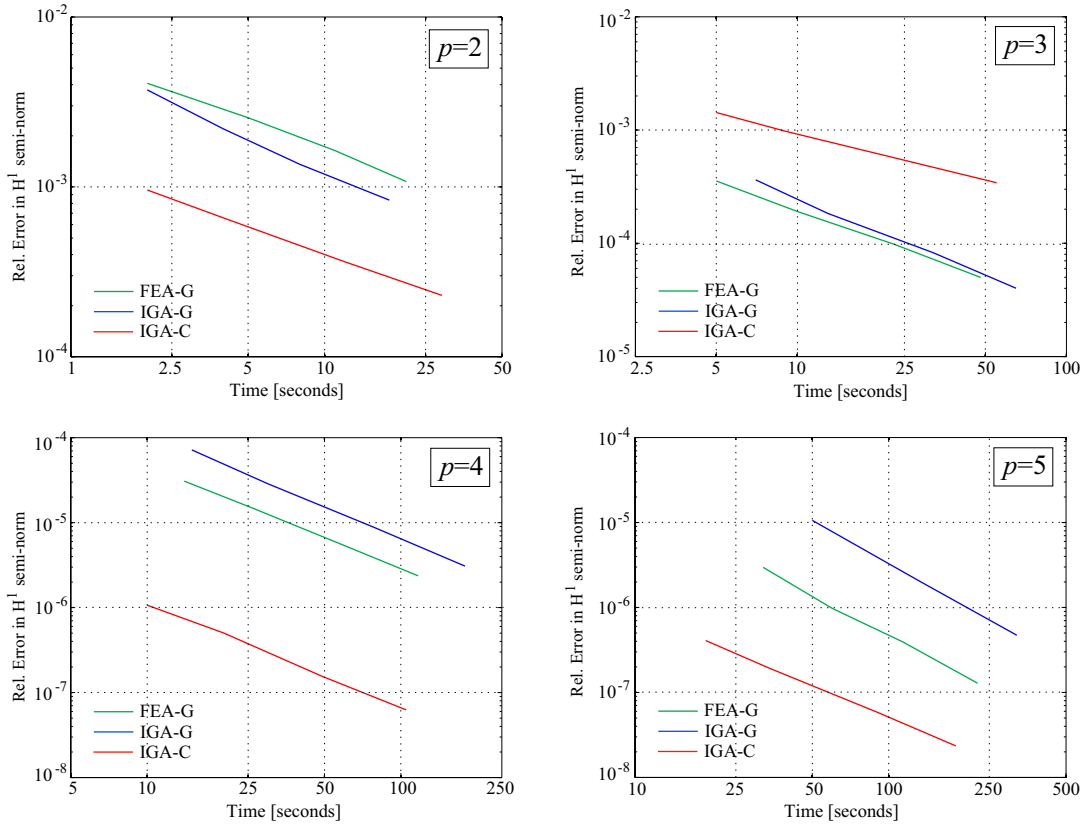


Figure 21:

Smooth 3D elasticity:

L^2 error vs. degrees of freedom

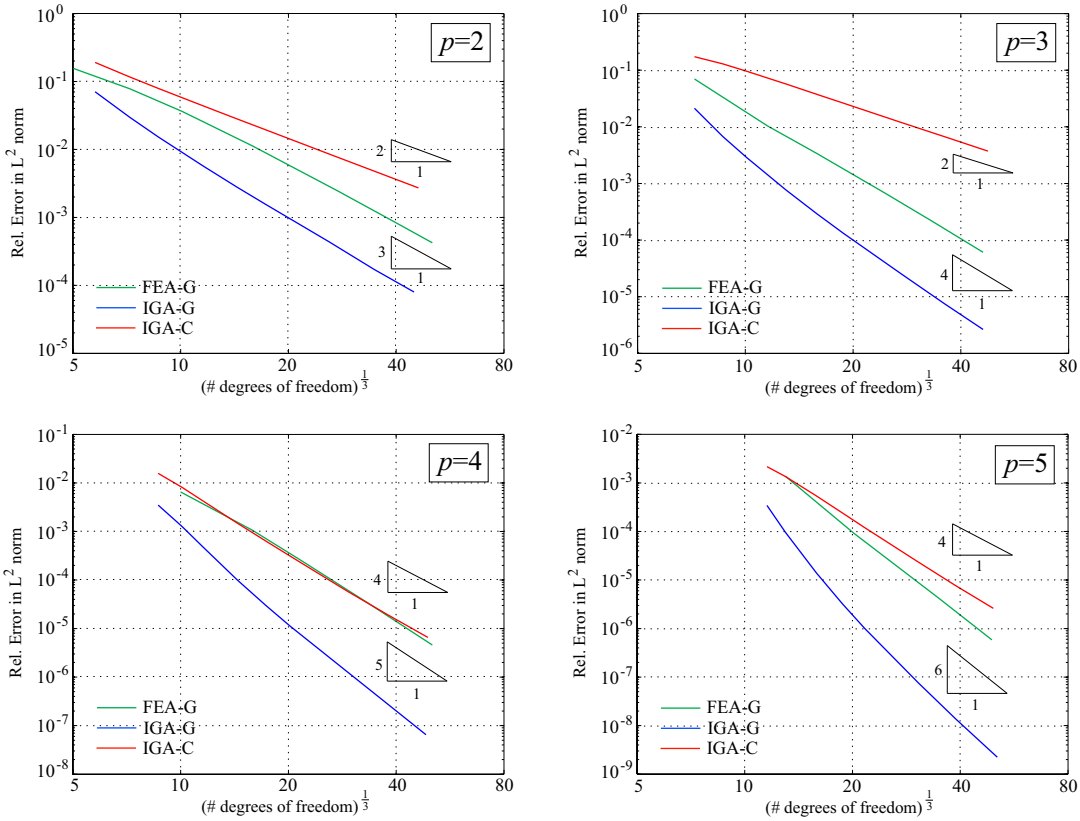


Figure 22:

Smooth 3D elasticity:

L^2 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

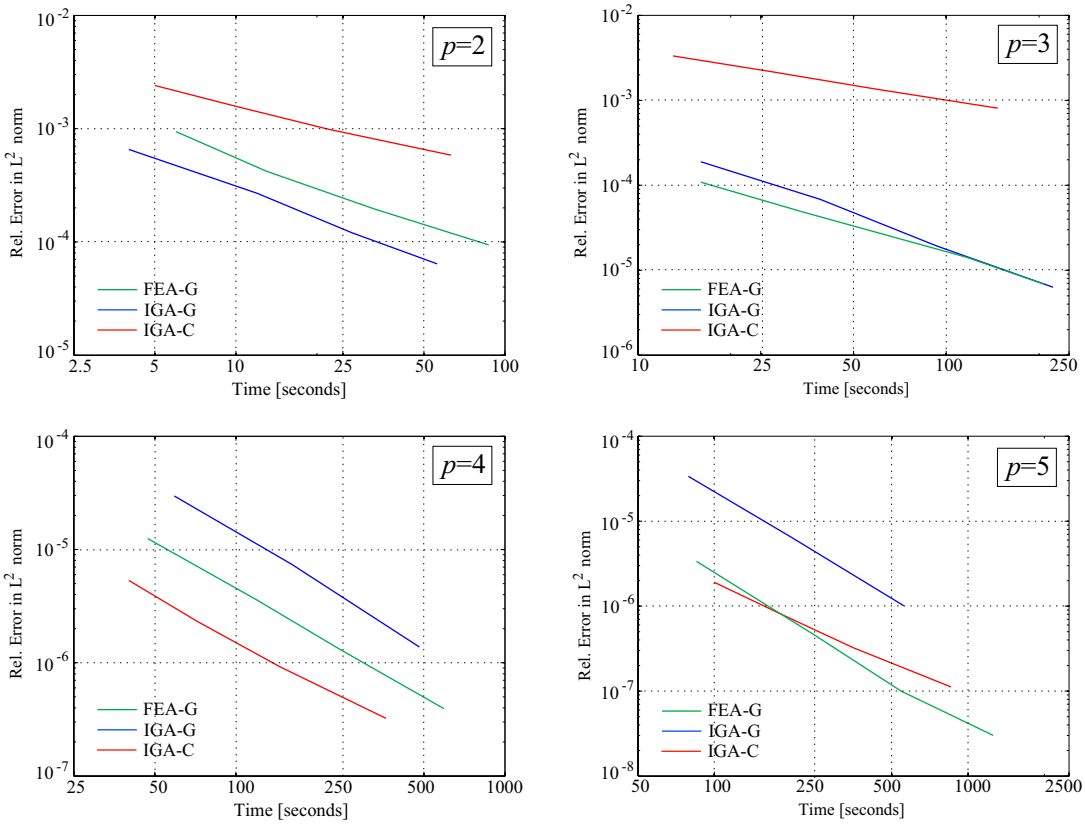


Figure 23:

Smooth 3D elasticity:

H^1 error vs. degrees of freedom

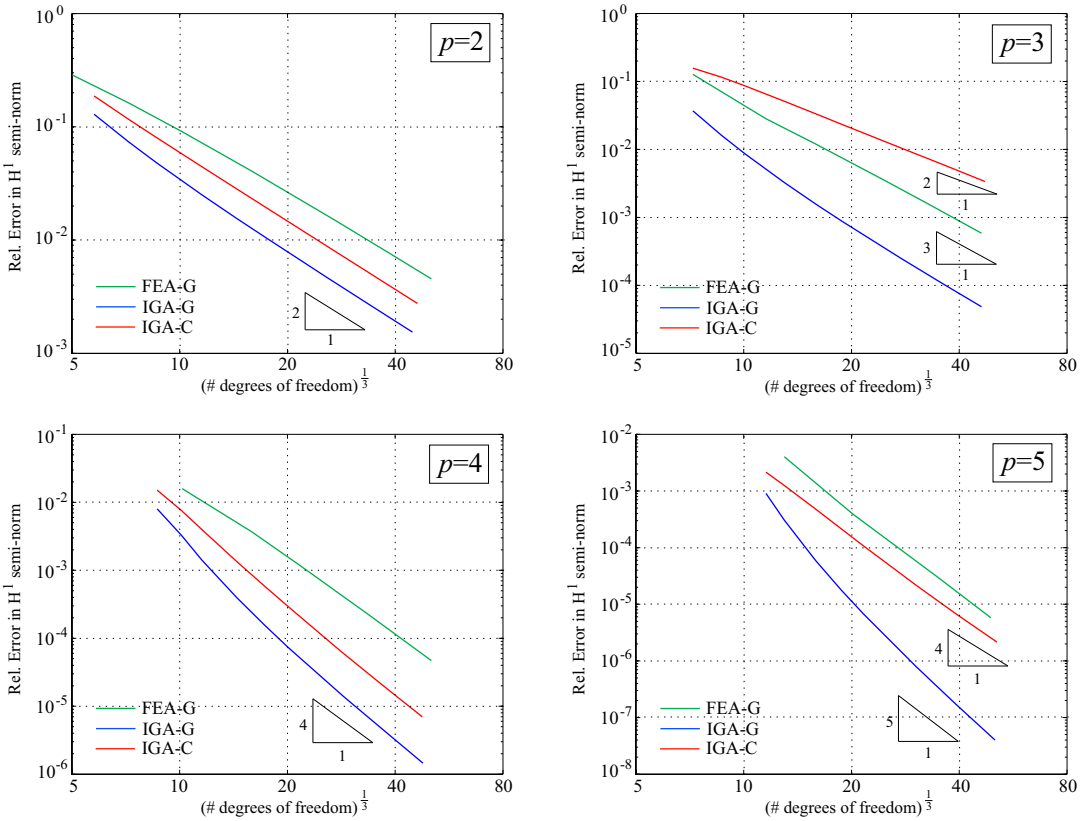
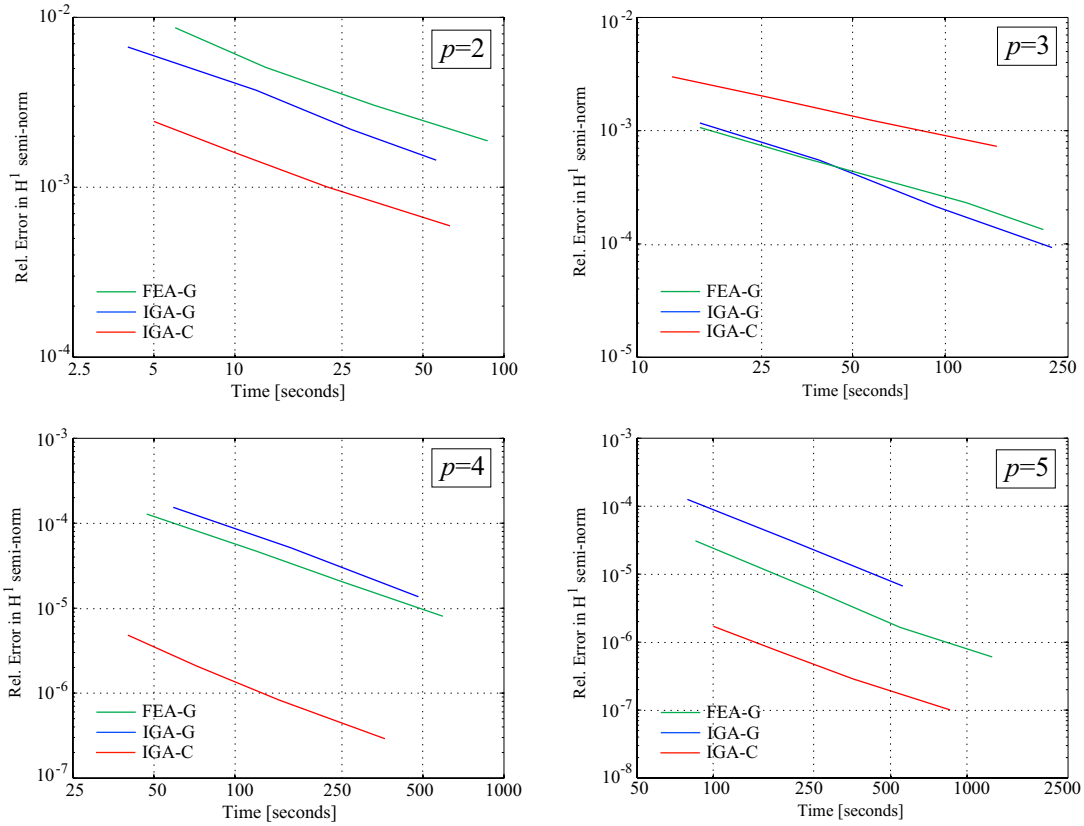


Figure 24:

Smooth 3D elasticity:

H^1 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)



overall computing times below one second. The corresponding equation systems range in size between 1,000 and 100,000 degrees of freedom for IGA-G, between 25,000 and 400,000 degrees of freedom for FEA-G and between 100,000 and 1,000,000 degrees of freedom for IGA-C. The convergence results with respect to computing time are shown in Figs. 18, 20, 22 and 24 for the smooth Poisson and elasticity problems, and in Figs. 26, 28, 30 and 32 for the rough Poisson and elasticity problems. Each figure compares the corresponding performance of IGA-C (red curve), IGA-G (blue curve) and FEA-G (green curve).

3.3.2. Theoretical analysis of isogeometric collocation

Before discussing the results of the convergence study, we briefly recall the following error estimate that holds for IGA-G and FEA-G [93, 102, 103]

$$\|u - \hat{u}\|_s \leq C\, h^{k-s}\, \|u\|_k \tag{35}$$

where $\hat{u}$ is the approximation to the true solution u, and C is a constant. In the cases s=0 and s=1, $\|\cdot\|_0$ and $\|\cdot\|_1$ denote the L^2 norm and the H^1 norm, respectively. The corresponding exponent to the mesh size h denotes the rate of convergence, whose optimal values of $\mathcal{O}(p+1)$ in L^2 and $\mathcal{O}(p)$ in H^1 occur when k=p+1.

Unfortunately, an abstract mathematical framework that allows a thorough numerical analysis of collocation methods has not yet been established. A thorough theoretical analysis of IGA-C including proofs of stability, convergence and error estimates in the sense of Eq. (35) is only available for the one-dimensional case [45]. For higher-dimensional spline spaces, theoretical studies have been accomplished only for some special cases [87, 104, 105]. As a consequence, convergence results for 2D and 3D NURBS discretizations are currently available only based on numerical studies [45, 46]. IGA collocation with basis functions of polynomial degree p and continuity C^{p-1} was observed numerically to converge with a rate of $\mathcal{O}(p)$ for the error in the L^2 norm and H^1 semi-norm, if the polynomial degree p is even, and with a rate of $\mathcal{O}(p-1)$ for the error in the L^2 norm and H^1 semi-norm, if the polynomial degree p is odd. For collocation, we refer to these convergence rates in the L^2 norm and H^1 semi-norm as the *best possible* rates, since we do not expect to achieve higher rates. We note that IGA collocation with basis functions of general polynomial degree p and continuity C^{p-1} converges optimally to the exact solution in the $W^{2,\infty}$ norm in the strict sense of Eq. (35) [45].

Figure 25:

Rough 3D Poisson:

L^2 error vs. degrees of freedom

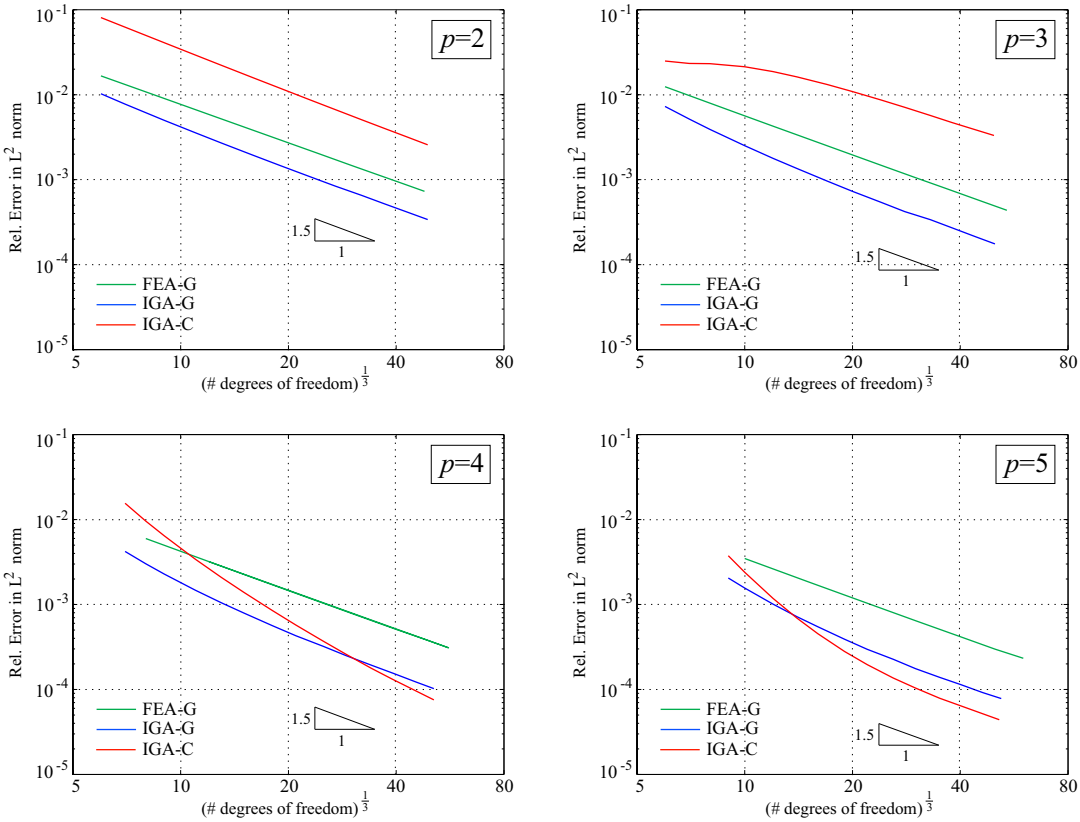


Figure 26:

Rough 3D Poisson:

L^2 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

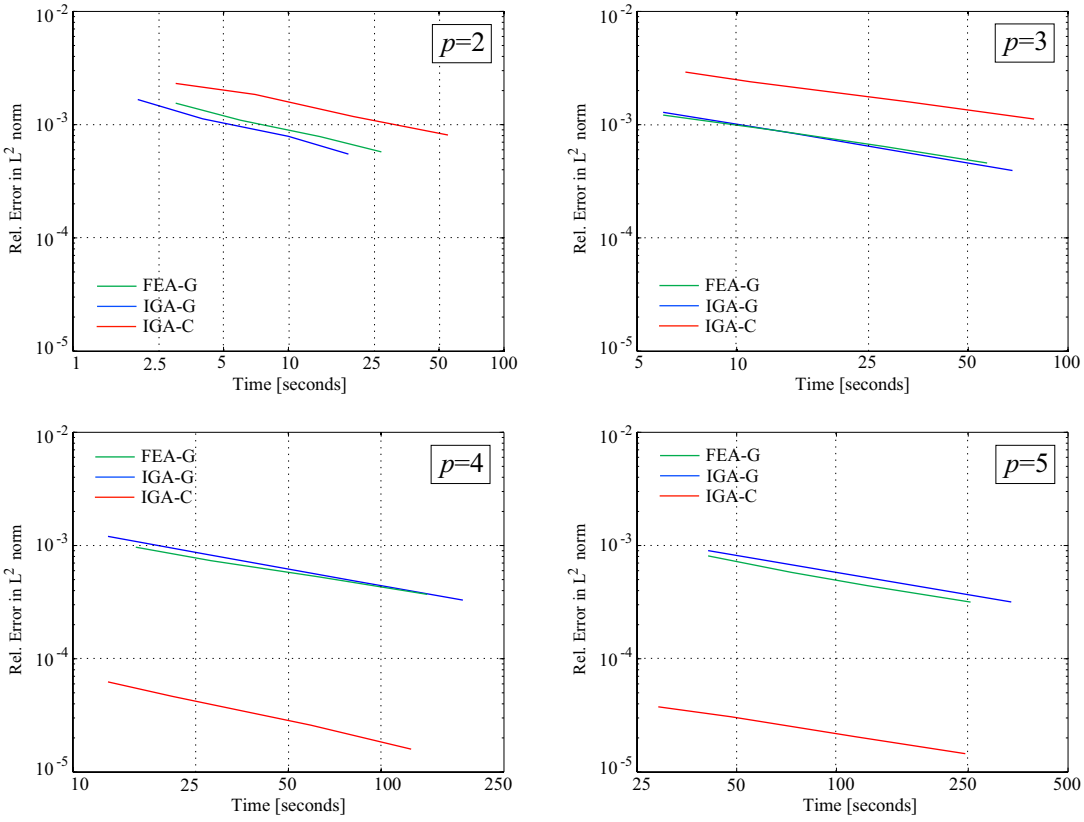


Figure 27:

Rough 3D Poisson:

H^1 error vs. degrees of freedom

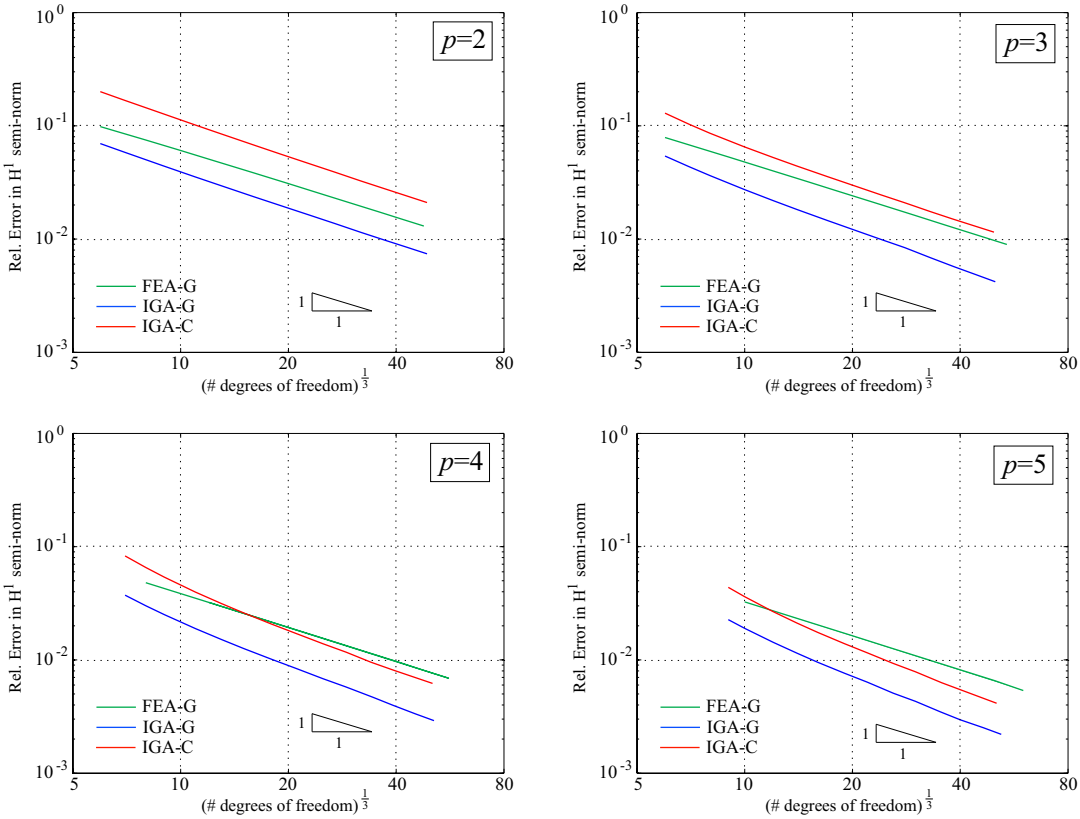


Figure 28:

Rough 3D Poisson:

H^1 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

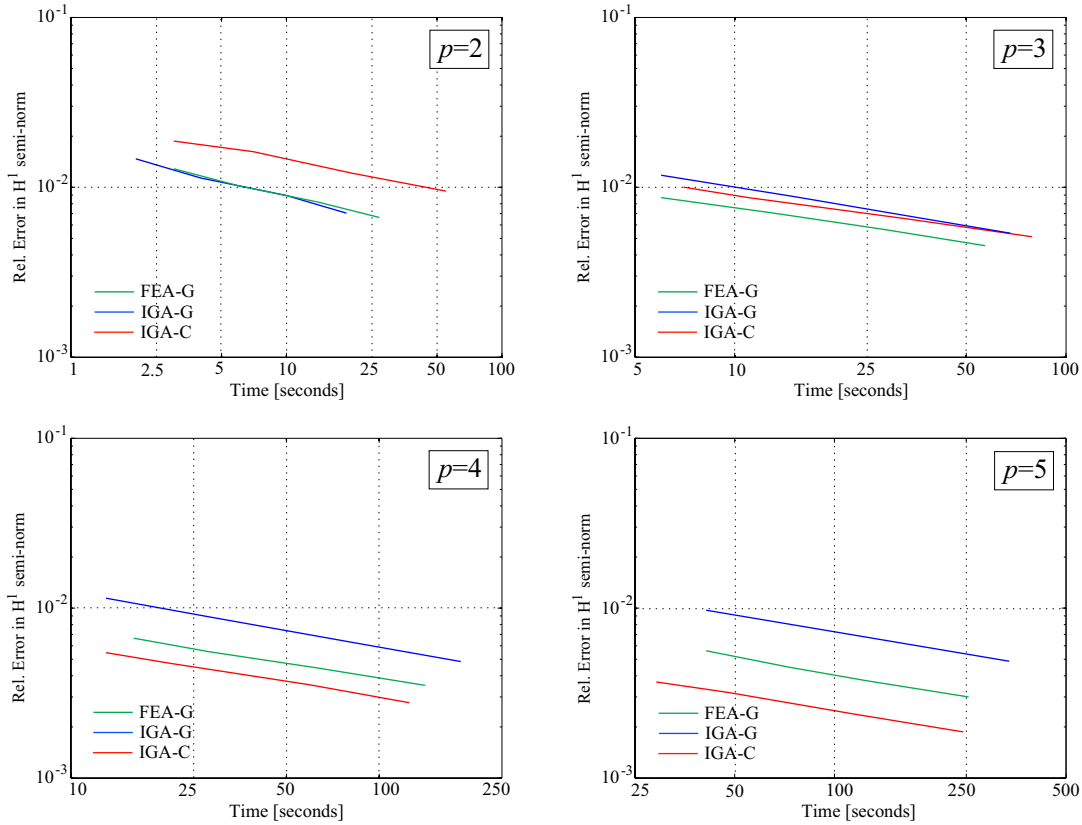


Figure 29:

Rough 3D elasticity:

L^2 error vs. degrees of freedom

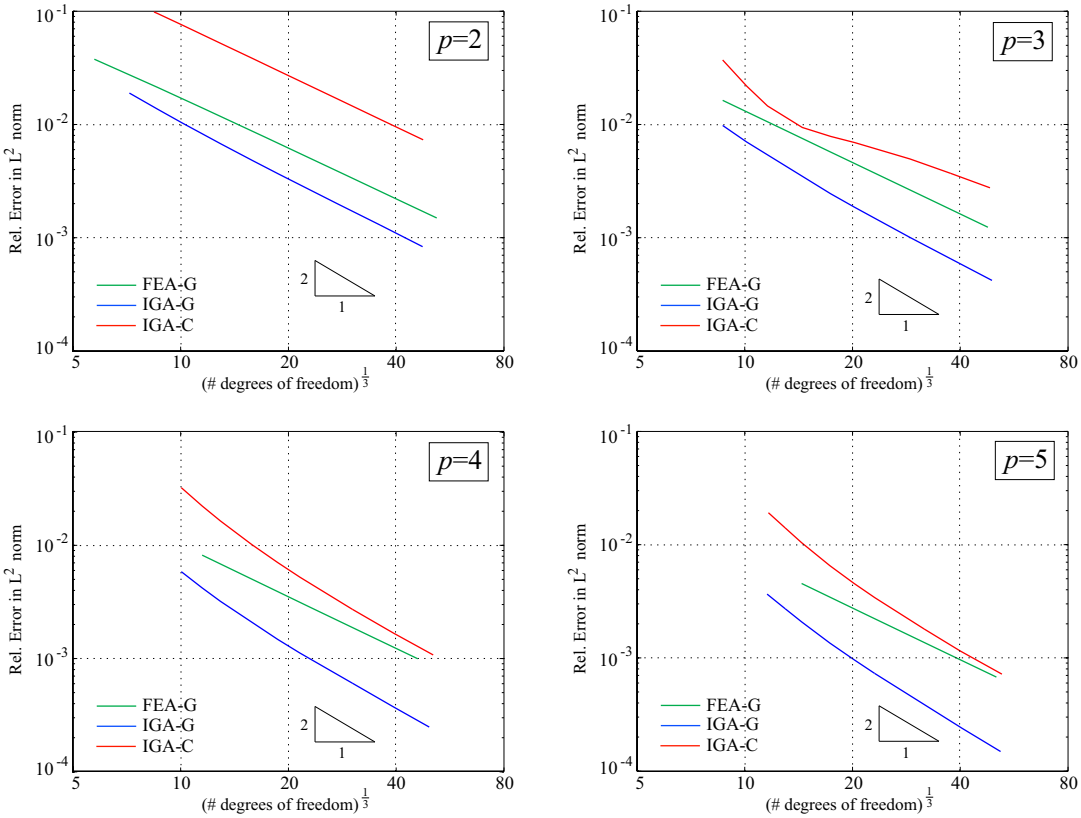


Figure 30:

Rough 3D elasticity:

L^2 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)

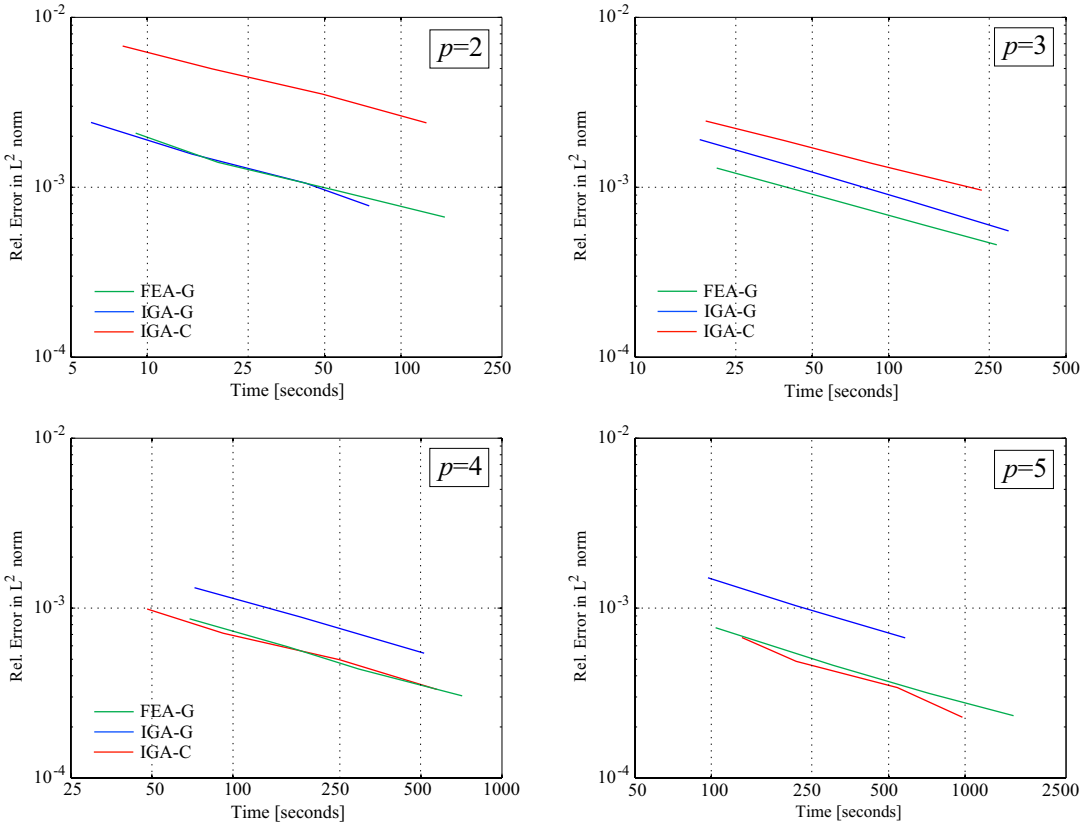


Figure 31:

Rough 3D elasticity:

H^1 error vs. degrees of freedom

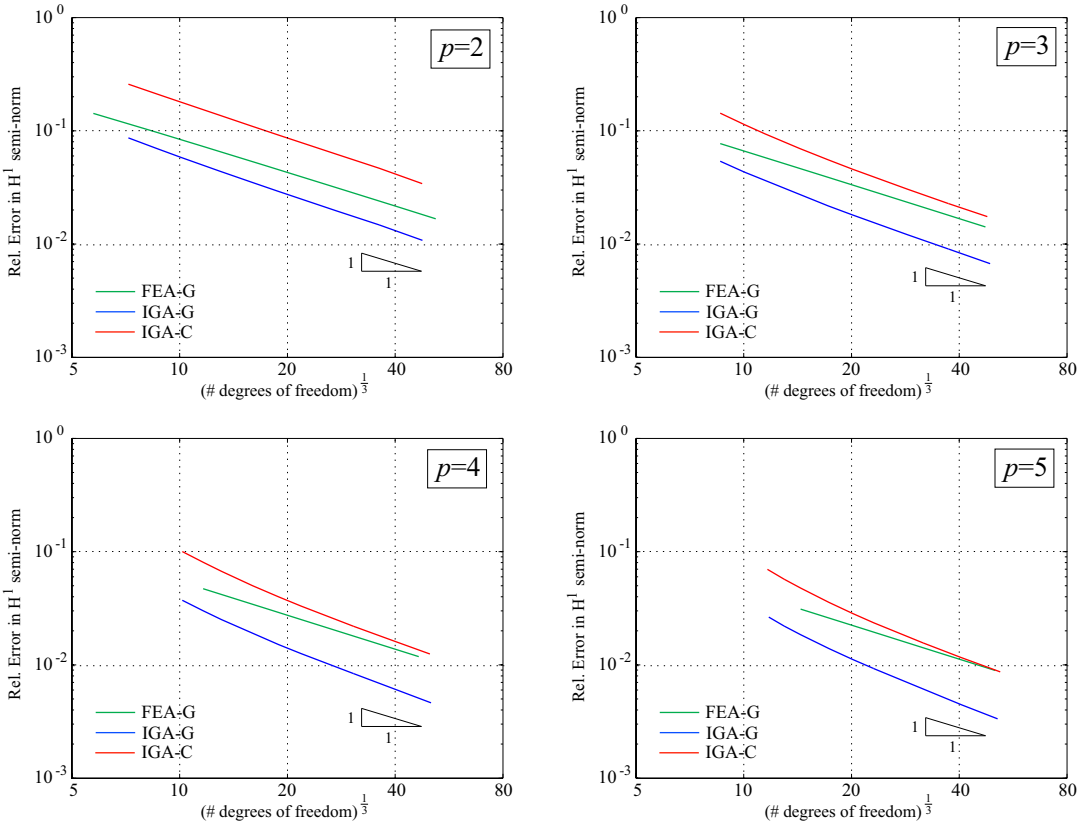
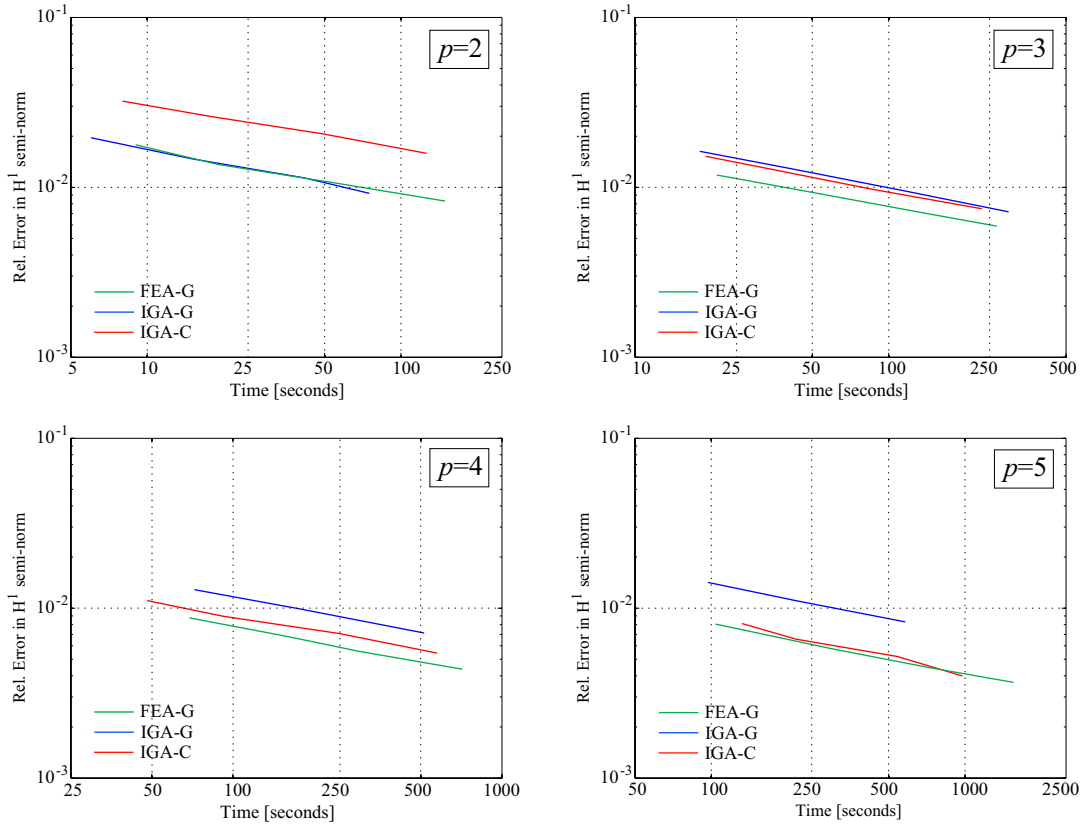


Figure 32:

Rough 3D elasticity:

H^1 error vs. time

(Matrix formation and assembly, preconditioning, solution with an iterative solver)



3.3.3. Accuracy vs. the number of degrees of freedom

The plots showing error norms vs. the number of degrees of freedom allow us to examine and compare the convergence properties of IGA-C, IGA-G and FEA-G. To establish a link between the mesh size h being a 1D measure and the number of degrees of freedom emanating from 3D discretizations, we take the cube root of the latter. Focusing on the smooth scalar and vector problems first (see Figs. 17, 19, 21, 23), we observe that both IGA-G and FEA-G achieve optimal rates of convergence in both the L^2 and H^1 norms for all polynomial degrees considered. For p=4 and p=5, IGA-G can be observed to be slightly superconvergent with asymptotic rates that exceed the theoretical optimum by up to 10%. With respect to the accuracy per degree of freedom, IGA-G is by far more accurate than FEA-G. The absolute difference in error between the corresponding asymptotic curves amounts to up to 2.5 orders of magnitude in the L^2 norm and 1.5 orders of magnitude in the H^1 semi-norm. For IGA-C, we observe that the best possible rates in both norms are achieved in all cases. Due to its lower rates of convergence, we would expect that asymptotically IGA-C would always lag behind IGA-G and FEA-G. Assuming an even polynomial degree of the basis functions, we can observe that for the error in the H^1 semi-norm IGA-C is fully competitive with both IGA-G and FEA-G with respect to the accuracy per degree of freedom. From an engineering point of view, this can be considered a significant benefit, since critical quantities of engineering interest are often derived from the first derivatives of the basis functions, e.g. stresses in elasticity. In general, we observe that for the higher order cases p=4 and p=5 the IGA-C error curves in both the L^2 and H^1 error norms are surprisingly close to the FEA-G curves in the examined range of degrees of freedom.

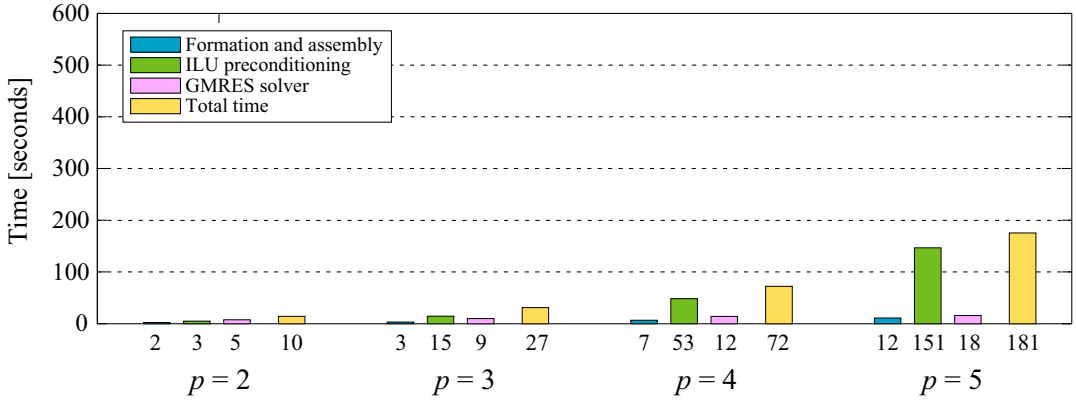
We then turn to the results for the rough scalar and rough vector problems, whose accuracy per degree of freedom is shown in Figs. 25, 27, 29 and 31. As predicted by theory [93, 102], the error estimate of Eq. (35) does not hold due to the corner singularity in the derivatives, which results in considerably lower rates of convergence in all methods. We observe that all methods show the same asymptotic rates in the L^2 and H^1 error norms. IGA-G exhibits the best accuracy per degree of freedom through almost all plots. FEA-G always lags behind IGA-G. For the vector problem, it seems that 100,000 degrees of freedom are not sufficient for FEA-G to reach the full asymptotic rate. In the L^2 case, IGA-C using quadratic and cubic basis functions is less accurate per degree of freedom than IGA-G and FEA-G. In the Poisson example, however, IGA-C using quartic and quintic basis functions is the most accurate in the L^2 norm, even leaving behind IGA-G. The reason for this phenomenon is not yet clear. With respect to the error in the H^1 error norm, IGA-C

using quartic and quintic basis functions is competitive with FEA-G.

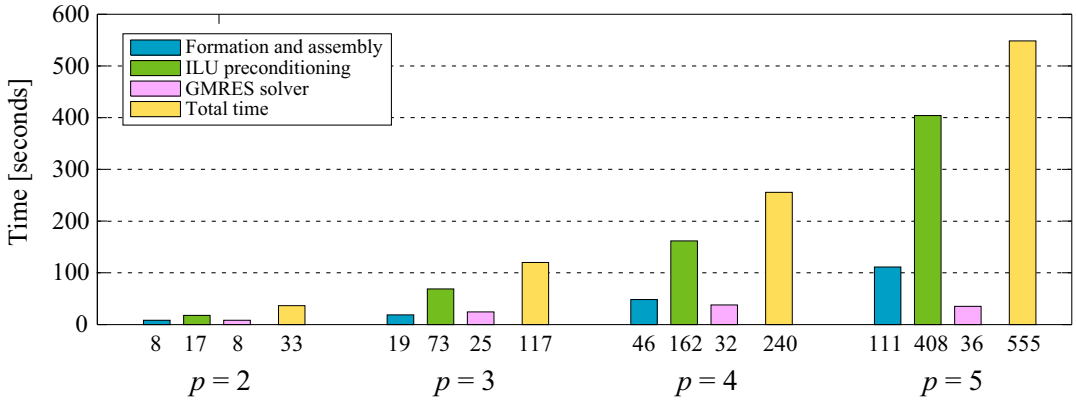
3.3.4. Accuracy vs. computing time

The groups of plots relating the error to the corresponding serial computing time allow us to estimate, which of the three methods will be the fastest to achieve a specified level of accuracy. This question is of fundamental importance and largely determines the potential of a numerical method for the use in engineering applications. Focusing first on the results for smooth problems shown in Figs. 18, 20, 22 and 24, we observe that if we choose an even polynomial degree p, IGA-C is generally orders of magnitude faster than both IGA-G and FEA-G. This is in particular true for the error in the H^1 semi-norm, where the convergence rates of IGA-C are equal to those of IGA-G and FEA-G for the case of even p. The superiority of IGA-C is best illustrated in Fig. 24, where with p=4 IGA-C achieves an error level of 10^{-5} in the H^1 semi-norm in less than 20 seconds, whereas both IGA-G and FEA-G require more than 500 seconds to reach that accuracy. Collocation loses some of its dominance when applied with odd polynomial degrees, since the difference in convergence rates compared with the Galerkin methods increases (see Section 3.2). This particularly holds for p=3, where IGA-C performs worse than IGA-G and FEA-G. The reason for this is that IGA-G and FEA-G achieve optimal rates of 4 and 3 in the L^2 and H^1 error norms, while the best possible rate in IGA-C is only 2 for both cases. For p=5, this effect is already considerably reduced, with IGA-C performing comparably with FEA-G in the L^2 case and being clearly the fastest method in the H^1 case. It can be expected that for odd polynomial degrees higher than 5, IGA-C will increasingly dominate over IGA-G and FEA-G. Comparing the results for the Poisson and elasticity problems, we note that the difference in computing time between IGA-C and the Galerkin methods increases considerably in the vector case. This is mainly due to the larger cost for matrix-matrix products in IGA-G and FEA-G, while the cost of IGA-C is virtually invariant to the number of unknowns per node (see Section 3.2 and Appendices A.2, A.3). Looking at the results for the rough problems shown in Figs. 26, 28, 30 and 32, we observe that the situation changes with respect to the smooth case. In general, we can say that IGA-C is fastest for the higher polynomial degrees p=4 and 5, while IGA-G or FEA-G are faster for p=2 and 3. The pronounced difference between even and odd polynomial degrees vanishes, since neither the Galerkin methods nor collocation achieve optimal or best possible rates of convergence.

Figures 33a and 33b show the relative computing times spent for the formation and assembly of the stiffness matrix and load vector, the ILU(0) preconditioning of the system and the iterative solution with the GMRES solver in IGA-C and FEA-G. We observe that



(a) *Isogeometric collocation (IGA-C)*



(b) *Standard C^0 finite elements (FEA-G)*

Figure 33: *Relative timings for the formation and assembly of the stiffness matrix and load vector, the ILU(0) preconditioning and the solution with the GMRES solver in IGA-C and FEA-G for the solution of the smooth 3D elasticity problem with 250,000 degrees of freedom.*

for IGA-C the main expense is clearly the preconditioning of the system. In FEA-G, the relative cost of formation and assembly is higher than in IGA-C. Since the cost of the ILU(0) preconditioner is proportional to the number of non-zero entries per row of the stiffness matrix, the cost of preconditioning of a stiffness matrix of the same size is smaller for IGA-C than for FEA-G.

3.3.5. From operation counts to computing times on modern multi-core machines

We believe that the superior performance of isogeometric collocation demonstrated in Section 3.1 and 3.2 in terms of floating point operations and in Section 3.3 in terms of timings on a single thread is only a conservative estimate of what is really possible in parallel computations on modern multi-core machines. One important aspect is that the bottleneck

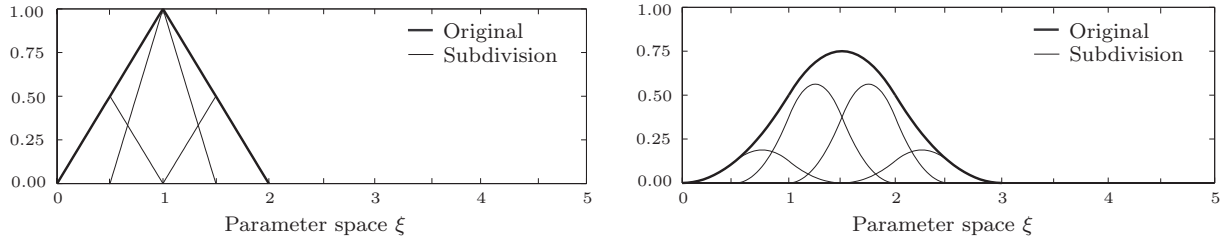
on modern machines lies with memory usage and access rather than flops. This is especially true for three-dimensional discretizations of high polynomial degrees where one works with very large element stiffness matrices. One necessarily cannot store the local stiffness matrix on lower-level caches, and the consequence is a large proportion of cache misses and accesses to higher levels of memory. With collocation, we completely circumvent this issue, since we do not have element stiffness matrices. Instead, we just work with one row of such a matrix. This drastically reduces memory usage and access when compared with Galerkin methods. Another important aspect is the potential of collocation in terms of parallel computing, in particular with respect to reduced parallel communication. When one assembles an element stiffness matrix into the global system, one has to send rows of the element stiffness matrix to the correct processor. One hopes that most of the time, the correct processor is the one where the element stiffness matrix was computed, but it is impossible to ensure this occurs all of the time. There will always be some matrix assembly communication cost associated with basis functions who have "support" over multiple processors. On the other hand, with collocation, one works with a single row and one can ensure it will be assembled into a row on the current processor. Therefore, there is no matrix assembly communication cost in collocation. The only communication cost one incurs with collocation is after a linear solve when one updates the solution coefficients. Moreover, typical large-scale simulations in computational mechanics, e.g. in the case of Newton iterations or implicit time stepping, require a large number of solutions of systems of equations with the same matrix structure. It is often the case that the preconditioner is computed only once at the beginning of a time step in nonlinear cases. Due to its sparse banded form, it can be efficiently stored for subsequent use. According to the relative timings given in Figure 33a, this significantly reduces the overall computing time for IGA-C in all subsequent solves. In FEA-G, the corresponding gain in efficiency is smaller, since the share of preconditioning in the total cost is smaller than in IGA-C. In IGA-G, we expect to see almost no gain in efficiency, since the total computing cost is completely dominated by the cost of formation and assembly, which need to be repeated before each solve. Based on this, we anticipate that efficient parallel implementations will put isogeometric collocation even further ahead of Galerkin methods than shown in Figs. 17 through 32 for large systems on modern multi-core machines.

3.4. A few rules of thumb

In summary, we present the following salient observations:

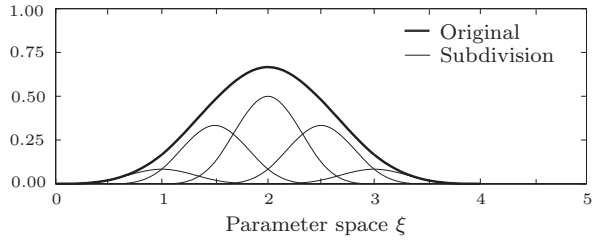
1. IGA-C is more efficient for even polynomial degrees than odd polynomial degrees.

2. Quadratic splines seem to play the same role within spline applications that linear finite elements have done historically in FEA-G.
3. For low order polynomial degrees, all methods seem to be equally viable.
4. IGA-C offers significant gains in efficiency compared with IGA-G and FEA-G. The relative efficiency increases with problem complexity in the following sense:
   - The gains are greater in 3D than in 2D.
   - The gains are greater for vector field problems than for scalar field problems.
   - The gains are greater for higher polynomial degrees.
5. In particular, higher order IGA-C (i.e., $p > 3$) offers the best accuracy-to-computing-time ratios, if one is interested in displacements *and* stresses.
6. Collocation potentially offers significant advantages for minimizing memory storage and access, and maximizing parallel performance.
7. IGA-C looks promising for explicit dynamics.
8. In singular problems, a globally uniform refinement strategy was adopted, which does not produce optimal convergence rates for any method. That being the case, we found the cost effectiveness comparable for all methods. This issue is further explored computationally in Section 7.2 using local refinement strategies.

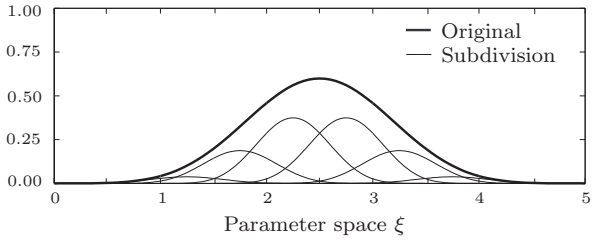


(a) *Linear B-spline.*

(b) *Quadratic B-spline.*



(c) *Cubic B-spline.*



(d) *Quartic B-spline.*

Figure 34: *Subdivision of a uniform coarse-scale B-spline into p+2 fine-scale B-splines of half the knot span width, illustrated for polynomial degrees p=1 through 4.*

4. Hierarchical refinement of NURBS

In the following, we briefly review B-spline subdivision and show how this concept can be employed to set up a hierarchical scheme for local refinement of B-spline and NURBS basis functions, which combines full analysis suitability and straightforward implementation.

4.1. Refinability of B-spline basis functions by subdivision

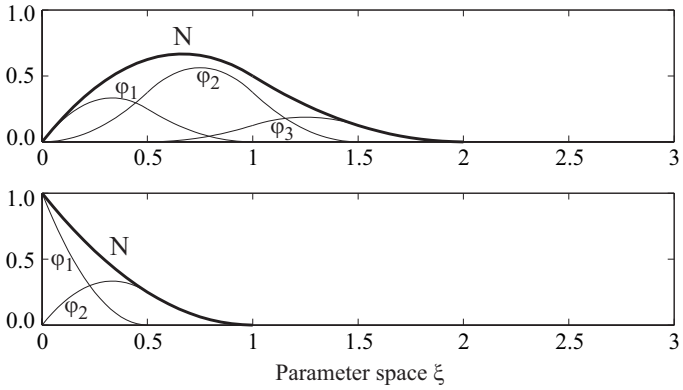
A remarkable property of B-splines is their natural refinement by subdivision. For a univariate uniform B-spline basis function N_p of polynomial degree p, the subdivision property leads to the following two-scale relation [65, 66, 106]

$$N_p(\xi) = 2^{-p} \sum_{j=0}^{p+1} \binom{p+1}{j} N_p(2\xi - j) \tag{36}$$

where the binomial coefficient is defined as

$$\binom{p+1}{j} = \frac{(p+1)!}{j!\,(p+1-j)!} \tag{37}$$

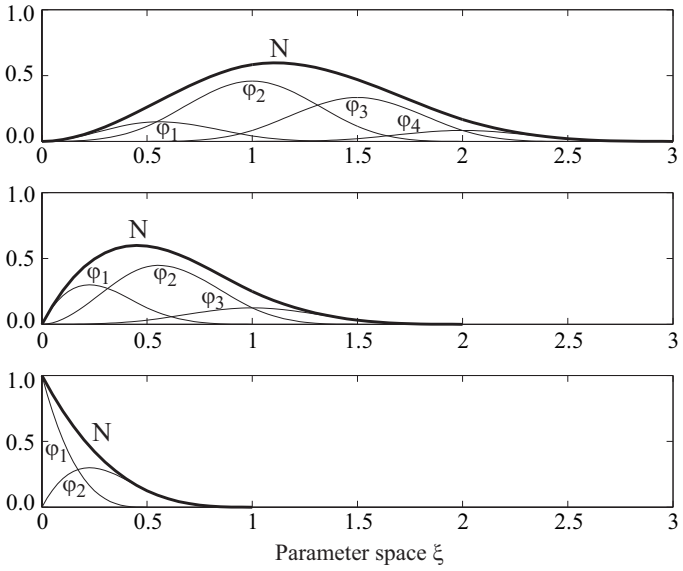
In other words, a B-spline can be expressed as a linear combination of contracted, translated and scaled copies of itself, as illustrated in Fig. 34. Note that Eq. (36) holds only for uniform



N: Ξ=$\{0,0,1,2\}$
φ_1: Ξ=$\{0,0,0.5,1\}$; s_1=0.5
φ_2: Ξ=$\{0,0.5,1,1.5\}$; s_2=0.75
φ_3: Ξ=$\{0.5,1,1.5,2\}$; s_3=0.25

N: Ξ=$\{0,0,0,1\}$
φ_1: Ξ=$\{0,0,0,0.5\}$; s_1=1.0
φ_2: Ξ=$\{0,0,0.5,1\}$; s_2=0.5

(a) *Subdivision rules for non-uniform B-splines of polynomial degree p=2.*



N: Ξ=$\{0,0,1,2,3\}$
φ_1: Ξ=$\{0,0,0.5,1,1.5\}$; s_1=0.25
φ_2: Ξ=$\{0,0.5,1,1.5,2\}$; s_2=0.6875
φ_3: Ξ=$\{0,0.5,1,1.5,2.5\}$; s_3=0.5
φ_4: Ξ=$\{0.5,1,1.5,2,3\}$; s_4=0.125

N: Ξ=$\{0,0,0,1,2\}$
φ_1: Ξ=$\{0,0,0,0.5,1\}$; s_1=0.5
φ_2: Ξ=$\{0,0,0.5,1,1.5\}$; s_2=0.75
φ_3: Ξ=$\{0,0.5,1,1.5,2\}$; s_3=0.1825

N: Ξ=$\{0,0,0,0,1\}$
φ_1: Ξ=$\{0,0,0,0,0.5\}$; s_1=1.0
φ_2: Ξ=$\{0,0,0,0.5,1\}$; s_2=0.5

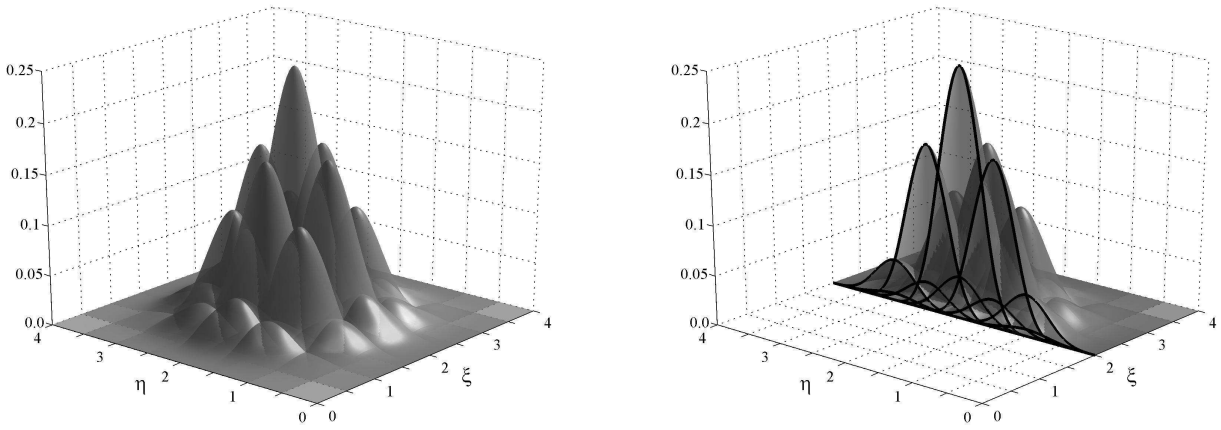
(b) *Subdivision rules for non-uniform B-splines of polynomial degree p=3.*

Figure 35: *Subdivision for quadratic and cubic B-splines resulting from open knot vectors. The symbols N, φ_i and s_i are used in the sense of the generalized subdivision rule Eq. (38).*

B-splines with distinct knots. For the case of non-uniform B-splines with repeated knots, we generalize Eq. (36) as

$$N = \sum_i \varphi_i \, s_i \tag{38}$$

The original B-spline function N is split into several new B-splines φ_i. These are multiplied by corresponding scaling factors s_i so that the generalized subdivision rule Eq. (38) holds. Corresponding knot vectors and scaling factors can be found in the literature [107, 108] or can be simply constructed by inspection on the basis of Eq. (38). In this work, we



(a) *Contracted, translated and scaled B-splines.* **(b)** *Cut along ξ=2.0.*

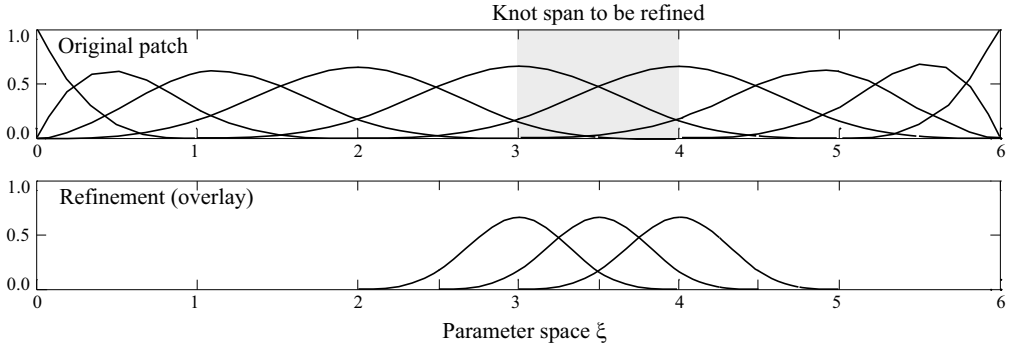
Figure 36: *Subdivision of the bivariate cubic B-spline shown in Fig. 3.*

use non-uniform subdivision to hierarchically refine boundary functions of a B-spline basis constructed from open knots vectors. Corresponding rules are given in Figs. 35a and 35b in terms of hierarchical functions, knot vectors and scaling factors for the two most common cases of p=2 and p=3, respectively.

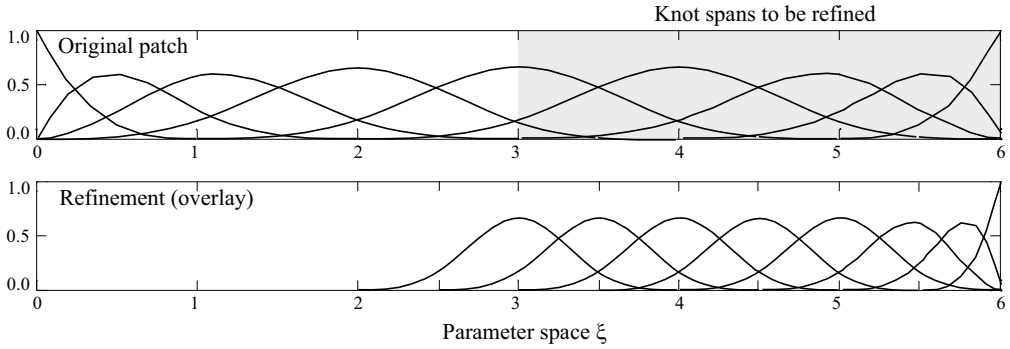
Due to their tensor product structure, the generalization of subdivision to multivariate B-splines is a straightforward extension of Eqs. (36) and (38) and can be written in the uniform case as

$$B_{\boldsymbol{p}}\left(\boldsymbol{\xi}\right) = \sum_{\boldsymbol{j}} \left(\prod_{\ell=1}^{d} 2^{-p_\ell} \binom{p_\ell + 1}{j_\ell} N_{p_\ell}(2\xi^\ell - j_\ell) \right) \tag{39}$$

Following Section 2.1.2, multi-indices $\boldsymbol{j}$=$\{j_1, \ldots, j_{d_p}\}$, $\boldsymbol{p}$=$\{p_1, \ldots, p_{d_p}\}$ and $\boldsymbol{\xi}$=$\{\xi^1, \ldots, \xi^{d_p}\}$ denote the position in the tensor product structure, the polynomial degree and the independent variables in each direction ℓ of the d_p-dimensional parameter space. Fig. 36 illustrates the new basis functions resulting from the multivariate two-scale relation Eq. (39) applied to the bivariate cubic B-spline of Fig. 2b. Analogous to Eq. (38), a generalization of Eq. (39) for non-uniform multi-variate B-splines may be easily constructed. The most widely known application of Eqs. (36), (38) and (39) is the development of highly efficient subdivision algorithms for the fast and accurate approximation of smooth surfaces by control meshes in computer graphics [65, 66, 107, 108].



(a) *Hierarchical refinement, inspired by the hp-d adaptive approach [61]. The combination of the original patch and three contracted B-splines yields the refined basis.*



(b) *Hierarchical refinement in the sense of the hp-d adaptive approach for the three rightmost knot span elements in a row. The overlay is generated by repeating the nucleus operation of Fig. 37a.*

Figure 37: *Two-level hierarchical refinement of a one-dimensional cubic B-spline patch.*

4.2. Construction of adaptive hierarchical approximation spaces

In the following, we will review a hierarchical scheme for local refinement of B-splines in one dimension, which combines concepts from B-spline subdivision, the *hp-d* adaptive approach [61, 109–111] and existing hierarchical refinement techniques for B-spline finite elements [62, 112–114] and standard nodal based FEA [115, 116]. We refer to our previous work in [63] for a more detailed presentation including an overview of corresponding algorithms.

4.2.1. Two-level hierarchical refinement for one element

As a first step, we define a nucleus operation, i.e. the refinement of one knot span element. Figure 37a exhibits a portion of a B-spline patch, where the element in the center is to be refined. We borrow the main idea of the *hp-d* adaptive approach, which was originally

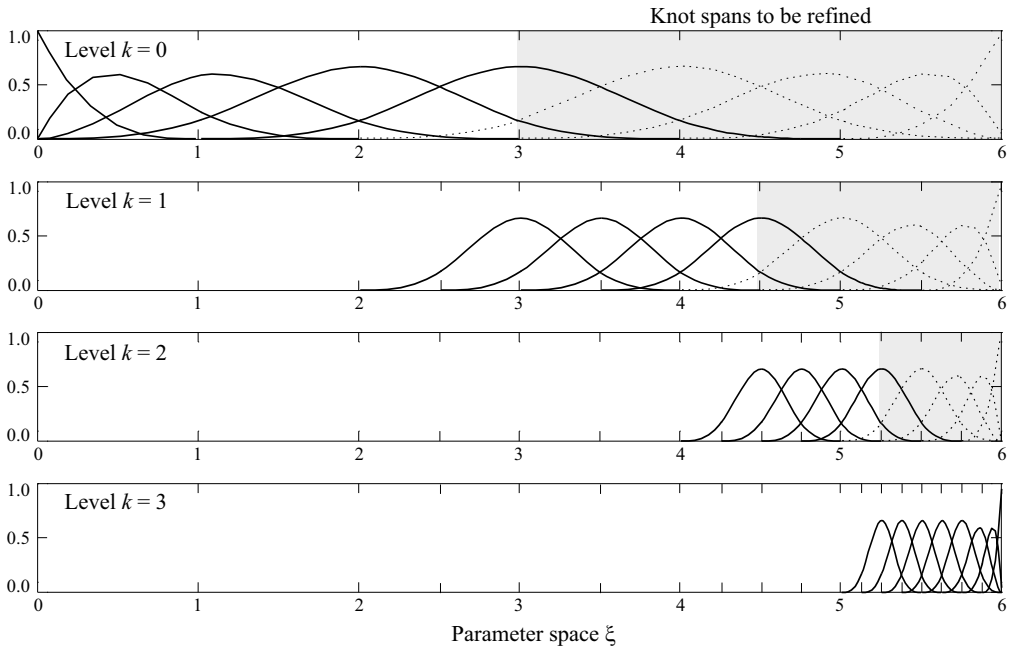


Figure 38: *Hierarchical multi-level refinement: B-splines of level k plotted in dotted line can be represented by a linear combination of B-splines of the next level k+1 according to the two-scale relation Eq. (36) and therefore need to be removed from the basis.*

introduced for the *p*-version of the FEM [110, 111] and successfully applied to B-spline bases in [61, 109]. In an *hp-d* sense, we add an overlay of three fine-scale B-splines of contracted knot span width to the original B-spline basis. At this point, no changes in the original coarse-scale basis functions are required, since we can infer from Eq. (36) that single fine-scale B-splines of contracted knot span width are linearly independent with respect to the original B-splines of full knot span width. The resulting basis is the combination of coarse-scale and fine-scale B-splines. In Fig. 37a, original and overlay basis functions are plotted on separate levels, which reflects the two-level hierarchy between the original basis and its refinement overlay. Furthermore, we do not change the amplitude of fine-scale B-splines, thus ignoring the presence of scaling factors s_i in Eq. (38).

4.2.2. Two-level hierarchical refinement for several elements

Let us proceed one step further to the refinement of several knot span elements in a row. Figure 37b illustrates the two-level hierarchical basis, which results from a repetition of the nucleus operation illustrated in Fig. 37a for the three rightmost elements in the patch. In particular, this procedure does not affect the higher-order smoothness of the refined basis, since the first $p-1$ derivatives of the hierarchical B-spline basis functions are zero at their

support boundaries. The specific refinement rule of Fig. 37a is valid for polynomial degree p=3, but can be easily transferred to B-spline bases of other polynomial degrees by looking for the minimum number of fine-scale B-splines per element, with which a complete row of fine-scale B-splines in the overlay level can be achieved, when several elements are refined.

4.2.3. Multi-level hierarchical refinement

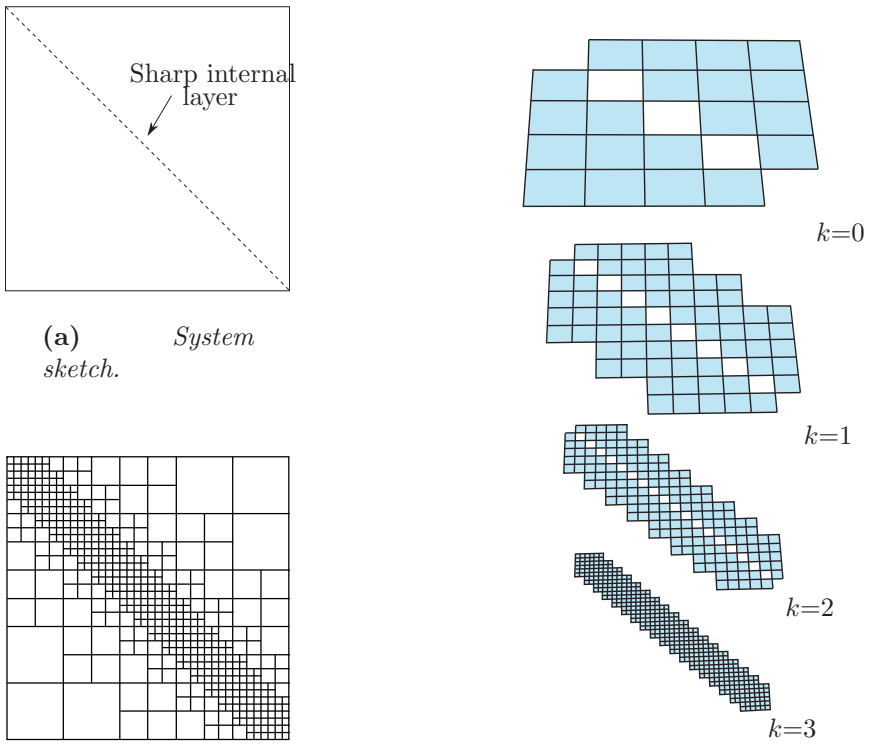
In order to increase the degree of local refinement, we proceed from the two-level hierarchy of a single refinement step to a general multi-level hierarchy, consisting of several overlay levels. Let us introduce the level counter k, where k=0 denotes the original B-spline patch. In each refinement step, the nucleus operation is applied to elements of the currently finest level k to produce a new overlay level k+1. Finer-scale B-splines of the new level k+1 are found by bisecting the knot span width with respect to level k. The multi-level refinement procedure is illustrated in Fig. 38, where the nucleus operation is successively applied to the three rightmost knot span elements of each level k. The resulting grid consists of a nested sequence of bisected knot span elements, and multiple hierarchical overlay levels of repeatedly contracted uniform B-splines. Note that for the refinement of boundary functions, the fine-scale B-splines are chosen according to the rules of Fig. 35b.

4.2.4. Recovering linear independence

In order to guarantee full analysis suitability of the hierarchically refined B-spline basis, we have to ensure its linear independence. Comparing the different levels in the hierarchy of Fig. 38, one can immediately observe that each overlay level k+1 consists of more than p+2 consecutive fine-scale B-splines. As a consequence, their linear combination is capable of representing some of the B-spline basis functions of the previous level k according to the two-scale relation Eq. (36). Therefore, we need to identify all B-spline basis functions that are a combination of fine-scale B-spline basis functions of the next level k+1 and remove them from the hierarchical basis. Furthermore, we need to ensure that any sequence of contracted B-splines on consecutive fine-scale knot spans is complete (see [63] for an instructive example). In Fig. 38, basis functions to be taken out are shown as dotted lines, while the final linear independent hierarchical B-spline basis consists of all basis functions shown as solid lines.

4.3. Generalization to multiple dimensions

The tensor product structure of multivariate B-splines permits a straightforward generalization of the one-dimensional hierarchical refinement concept presented in Section 4.2



(a) *System sketch.*

(b) *Hierarchical mesh.*

(c) *Multi-level structure of hierarchically contracted knot spans.*

Figure 39: *Multi-level hierarchical refinement of a quadratic B-spline patch along an internal layer.*

to multiple dimensions. In Fig. 39a, multivariate hierarchical refinement is illustrated for a bivariate quadratic B-spline patch, which is to be refined along its diagonal. Figure 39b shows the hierarchical mesh, which represents the global element structure. The corresponding multi-level knot spans, over which the hierarchical B-splines are defined, are illustrated in Fig. 39c. Note that for all higher levels the knot spans along the diagonal are empty, since all basis functions defined therein have been removed from the basis to preserve linear independence. In the collocation method, Dirichlet boundary conditions need to be imposed strongly. In 2D and 3D, non-zero Dirichlet boundary conditions can often be imposed easily, if the B-spline basis satisfies partition of unity at the Dirichlet boundary. This can be accomplished by keeping track of the scaling factors s_i of Eq. (38) during the refinement process [62] (see also Fig. 35). The strong imposition of complex functions can be achieved by a least squares fit of boundary basis functions [2, 117].

4.4. Generalization to NURBS

We derive subdivision rules for multivariate NURBS by inserting the two-scale relation of Eq. (39) into the construction rule for NURBS basis functions Eq. (8), which yields

$$R^{\boldsymbol{h}}_{\boldsymbol{i},\boldsymbol{p}}(\boldsymbol{\xi}) = \frac{w_{\boldsymbol{i}} \sum_{\boldsymbol{j}} \left(\prod_{\ell=1}^{d} 2^{-p_\ell} \binom{p_\ell+1}{j_\ell} N_{p_\ell}(2\xi^\ell - j_\ell) \right)}{\sum_{\boldsymbol{j}} w_{\boldsymbol{j}} B_{\boldsymbol{j},\boldsymbol{p}}(\boldsymbol{\xi})} \tag{40}$$

where the multi-index notation exactly follows the one introduced in Section 2.1.2 for multivariate B-splines. To efficiently accommodate NURBS in the hierarchical refinement process, we first separate Eq. (40) in a B-spline part (numerator) and a rational part (denominator), which we treat separately. In the numerator, we perform hierarchical refinement on the B-spline level, making full use of the concepts discussed in the previous paragraphs.

According to the isogeometric paradigm [1, 2], the geometry is described exactly by the original unrefined NURBS basis, so that geometry refinement is normally not required. Therefore, the denominator of Eq. (40) can always be computed with the original B-spline basis $B^0_{\boldsymbol{j},\boldsymbol{p}}(\boldsymbol{\xi})$

$$\mathrm{sum}(\boldsymbol{\xi}) = \sum_{\boldsymbol{j}} w_{\boldsymbol{j}} B^0_{\boldsymbol{j},\boldsymbol{p}}(\boldsymbol{\xi}) \tag{41}$$

where $w_{\boldsymbol{j}}$ and $\boldsymbol{P}_{\boldsymbol{j}}$ are the initial set of weights and control points. Note that we can additionally drop the weights $w_{\boldsymbol{i}}$ in the numerator of Eq. (40) for further simplification. Furthermore, the geometry mapping is computed throughout the hierarchical refinement procedure from the original unrefined NURBS basis

$$\boldsymbol{x}(\boldsymbol{\xi}) = \sum_{\boldsymbol{j}} \frac{w_{\boldsymbol{j}} B^0_{\boldsymbol{j},\boldsymbol{p}}(\boldsymbol{\xi})}{\mathrm{sum}(\boldsymbol{\xi})} \boldsymbol{P}_{\boldsymbol{j}} \tag{42}$$

Nonetheless, using the refined NURBS basis for enhancing the geometry representation would be of course possible [118, 119].

4.5. Efficient implementation of hierarchical refinement

A considerable advantage of hierarchical refinement in the present form is its straightforward and efficient implementation through quadtrees and octrees [67], which provide a natural way to decompose and organize spatial data according to different levels of complexity and offer fast access to relevant parts of a dataset [68, 69]. The quadtree concept shown in Fig. 40 illustrates the analogy between an adaptive hierarchical quadrilateral mesh and the two-dimensional tree. The tree is the fundamental entity, where each node or

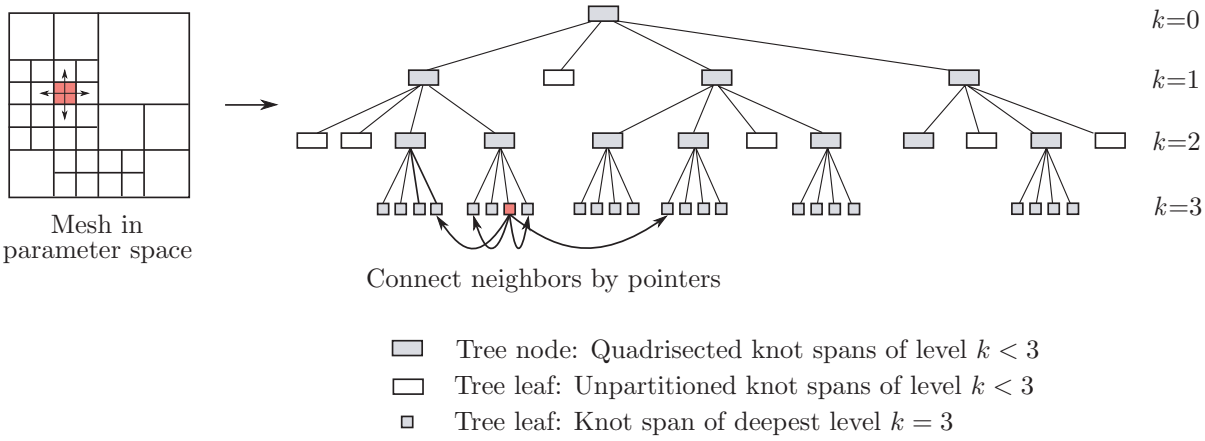


Figure 40: *Quadtree example illustrating the hierarchical data organization of part of an adaptive mesh. The neighboring relations within each hierarchical level are established by pointers, which are shown here for one element of the finest level (in red color).*

leaf holds all the information of the corresponding knot span on the respective hierarchical level. Additionally, each node or leaf can be equipped with pointers that connect it with all direct neighbors of the same hierarchical level (see Fig. 40), so that "horizontal" neighboring relations can be frequently checked with little computational effort. More details on implementation aspects and related algorithms can be found in [63].

In this context, we would like to point out that a different implementation approach has been recently proposed in [64], which is largely based on subdivision related concepts and algorithms developed in CAGD [65, 66, 107, 108]. The discrete interpretation of the two-scale relation is used to establish algebraic relations between the basis functions and their coefficients on different levels of the hierarchical mesh, which give rise to a subdivision projection technique. First, local element matrices and vectors are computed on a single level, using a fixed number of basis functions. During the subsequent assembly step, multiplication with subdivision matrices projects them to the correct levels of the hierarchical B-spline basis. Conceptually, subdivision projection is very similar to Bézier extraction [37, 38] and permits the integration of hierarchical B-splines into conventional finite element codes.

5. The concept of weighted isogeometric collocation

In the following, we first motivate the need for an isogeometric collocation scheme that can handle coincident collocation points on different levels of a hierarchical NURBS mesh. Subsequently, we derive the concept of weighted isogeometric collocation in a variational context and demonstrate its validity and numerical properties for standard single level NURBS

patches in one dimension.

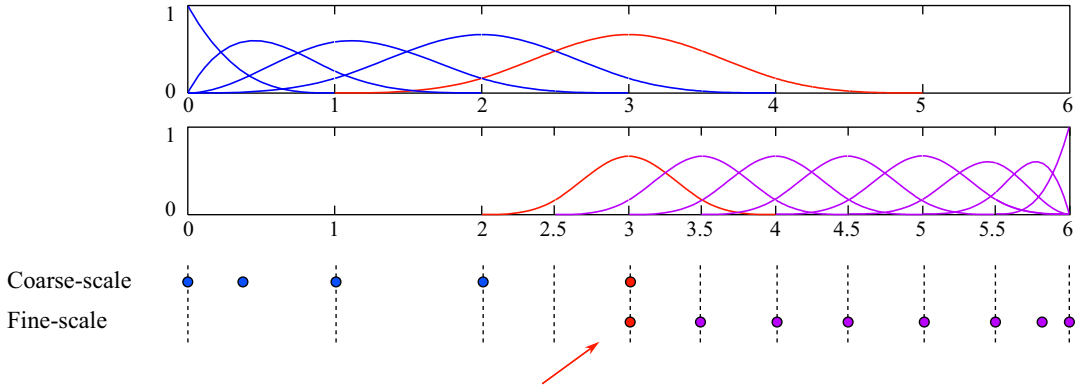
5.1. Motivation

We first attempt a straightforward combination of isogeometric collocation and hierarchical refinement of NURBS. Figure 41a shows a two-level cubic B-spline patch in one dimension. The corresponding collocation points are the Greville abscissae determined from the knot vector on each hierarchical level, which are shown along with the basis functions. Also in a hierarchical basis, the Greville abscissae automatically leads to the optimal number of collocation points, i.e. one point per basis function. However, two collocation points are coincident in the transition region, where the coarse and fine-scale basis functions overlap. Since each point evaluation results in a specific collocation equation according to Eqs. (17) through (19), two coincident points will produce the same equation, leading to a linearly dependent system. Figure 41b shows a two-level quartic B-spline patch, for which the collocation points based on the Greville abscissae of each hierarchical level are unique. However, these collocation points are not properly graded in regions, where basis functions of different hierarchical levels overlap. Numerical experiments reveal that this basis works for diffusion dominated problems, but becomes unstable for Péclet numbers higher than 50, which is attributed to the non-uniform accumulation of collocation points. For hierarchical refinement in the framework of the Galerkin method (see for example [62, 63, 114]), these problems do not occur. Each basis function is evaluated at several quadrature points, so that linear independence is properly reflected in the system. The problem of linear dependence in the context of hierarchical refinement and collocation can thus be interpreted as a lack of evaluation points in the transition regions, which leads to a loss of information concerning the hierarchical basis.

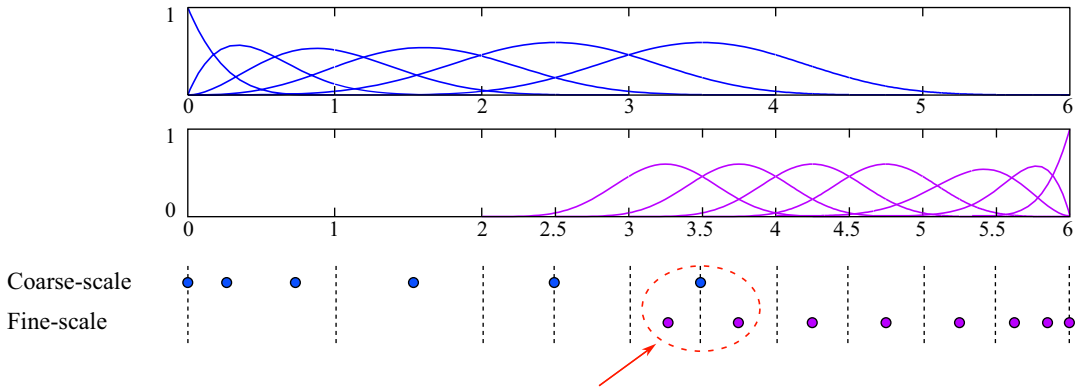
5.2. Variational background

Putting the issue of hierarchical refinement aside for a moment, we introduce the concept of weighted isogeometric collocation in a variational context. The basic idea of the modified collocation scheme is to achieve a compromise between the one point evaluation of standard collocation and the large number of point evaluations per basis function required by full Gauss quadrature of the integrals in the Galerkin method.

Weighted collocation in general can be derived by running through the same process as outlined for the standard scheme in Eqs. (12) through (19). The basis for its variational formulation is again the weighted residual form of the boundary value problem, Eq. (12), where the approximation of the solution field is achieved by Eq. (11). In contrast to standard



(a) *Cubic two-level patch: Coincident collocation points in the transition region.*



(b) *Quartic two-level patch: Accumulation of collocation points in the transition region.*

Figure 41: *Straightforward combination of hierarchical refinement and multi-level Greville abscissae.*

collocation, the summands of the test functions ω_Ω and ω_Γ are not chosen as individual Dirac δ functions, evaluated at one specific collocation point, but as sums of several weighted Dirac δ functions, evaluated at several collocation points

$$\omega_\Omega = \sum_{i=1}^{k} \left(\sum_a \delta(\boldsymbol{x} - \boldsymbol{x}_j)\ \alpha_a \right)\ \hat{c}_i \tag{43}$$

$$\omega_\Gamma = \sum_{i=k+1}^{n} \left(\sum_a \delta(\boldsymbol{x} - \boldsymbol{x}_j)\ \alpha_a \right)\ \hat{c}_i \tag{44}$$

where α_a is an individual weighting factor for each Dirac δ function.

Substitution of Eqs. (43) and (44) into the weak form of Eq. (12) again eliminates the

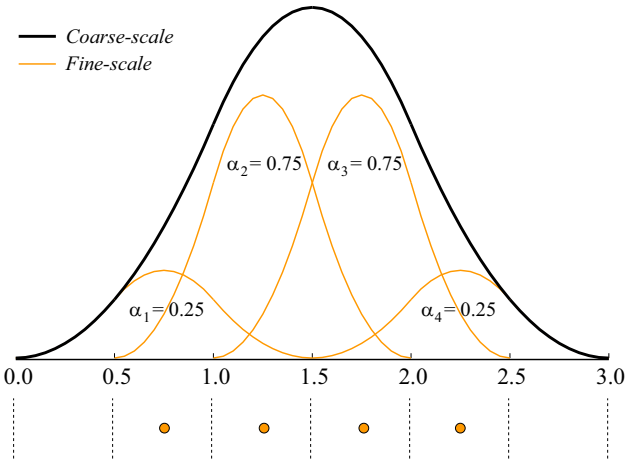


Figure 42: *The set of weighted collocation points and weighting factors α_a for a quadratic uniform B-spline.*

integrals using the sifting property Eqs. (13) and (14) of the Dirac δ functions, and yields

$$\sum_{i=1}^{k} \hat{c}_i \quad \sum_a \alpha_a \Big(\mathscr{L} \left[\, u_D(\boldsymbol{x}_i) + \sum_{j=1}^{k} N_j(\boldsymbol{x}_i) \, c_j \, \right] - f(\boldsymbol{x}_i) \Big) + \sum_{i=k+1}^{n} \hat{c}_i \quad \sum_a \alpha_a \Big(\boldsymbol{n}_i \cdot D \sum_{j=k+1}^{n} \nabla N_j(\boldsymbol{x}_i) \, c_j \; - h(\boldsymbol{x}_i) \Big) \;\; = \;\; 0 \quad (45)$$

Following from Eq. (45), the elements of system matrix $\boldsymbol{K}$ and load vector $\boldsymbol{F}$ are defined as

$$K_{ij} = \begin{cases} \sum_a \alpha_a \left[\mathscr{L} \left(N_j(\boldsymbol{x}_i) \right) \right], & \text{for } 1 \le i \le k \\ \sum_a \alpha_a \left[\boldsymbol{n}_i \cdot D \nabla N_j(\boldsymbol{x}_i) \right], & \text{for } k+1 \le i \le n \end{cases} \quad (46)$$

$$F_i = \begin{cases} -\sum_a \alpha_a \left[\mathscr{L} \left(u_D(\boldsymbol{x}_i) \right) + f(\boldsymbol{x}_i) \right], & \text{for } 1 \le i \le k \\ -\sum_a \alpha_a \left[\boldsymbol{n}_i \cdot D \nabla u_D(\boldsymbol{x}_i) + h(\boldsymbol{x}_i) \right], & \text{for } k+1 \le i \le n \end{cases} \quad (47)$$

A comparison of Eqs. (46) and (47) with Eqs. (18) and (19) confirms the interpretation of the current scheme as "weighted collocation", since the current entries can be derived by summing up the entries of the standard collocation matrix and vector, multiplied by a weighting factor α_a.

5.3. Collocating at the fine-scale Greville abscissae

In addition to using NURBS basis functions within the framework of the weighted collocation method, we need to come up with a suitable set of collocation points. In our opinion, a natural choice is the Greville abscissae generated from the fine-scale knot vector, obtained from a hierarchical split of the complete coarse-scale basis according to the two-scale relation Eq. (36). This choice combines a range of important advantages. First, the use of Greville abscissae as collocation points is in line with current practice in IGA collocation [45, 46]. In particular, they can be generated easily across hierarchical levels, and it is confirmed below that the crucial properties of best possible convergence and stability also transfer to weighted collocation. Second, considering its split in the sense of the two-scale relation, each coarse-scale basis function can be assigned to a clearly defined group of collocation points. This is illustrated in Fig. 42 for a uniform quadratic B-spline. Moreover, the split of the coarse-scale basis involves the scaling of the corresponding fine-scale basis functions (see Eqs. (36), (39) and Fig. 42), which gives rise to a natural choice of weighting factors α_a. Note that for boundary basis functions based on non-uniform B-splines, the generalized subdivision rule of Eq. (38) applies (see also Fig. 35). Finally, we anticipate here that the specific choice of collocation points based on the fine-scale Greville abscissae will be most useful, when accommodating hierarchical refinement in the weighted collocation concept.

The complete set of weighted collocation points is illustrated for a 1D quadratic B-spline patch in Fig. 43. The coloring clearly indicates the assignment of each group of collocation points to a specific coarse-scale basis function. In addition, we compare the collocation points of the weighting scheme to the standard collocation points derived from the Greville abscissae of the coarse-scale basis and to the quadrature points required for full integration in the Galerkin method. For the present example, we count 14 weighted collocation points as compared to eight standard collocation points and 18 quadrature points in the Galerkin method, which reflects the compromise between the two original approaches. It should be noted that coincident collocation points of the fine-scale Greville abscissae need to be evaluated only once and can then be assembled to the different equations after multiplication with the corresponding weighting factor. Hence, coincident points in the weighting scheme count as one point evaluation only. It should also be noted that in this example we do not mix interior and boundary test functions, and the boundary basis functions in 1D are not weighted (see Fig. 43).

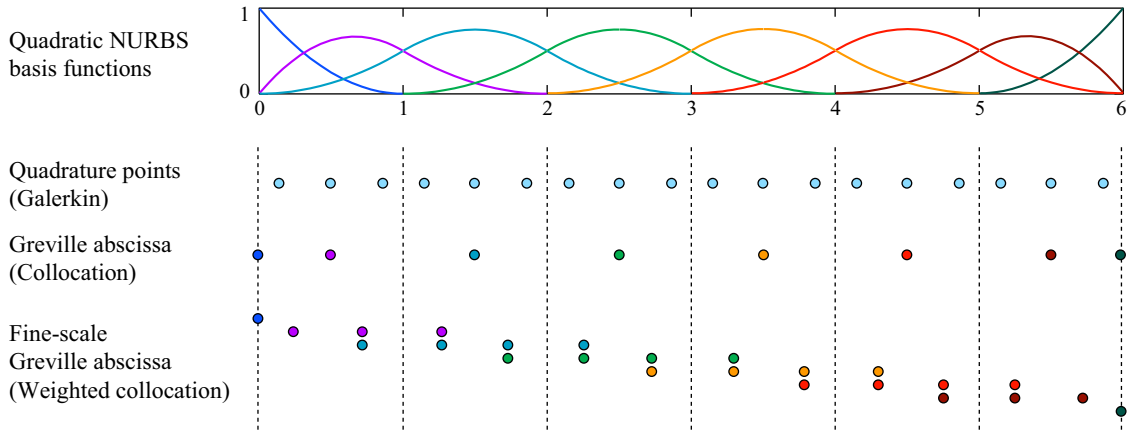


Figure 43: *Quadrature/collocation points for Galerkin, standard and weighted collocation methods in a quadratic single-level B-spline patch.*

5.4. A simple model problem in 1D

In the following, we test the weighted collocation method for single level B-spline patches in one dimension and explore its numerical properties, in particular best possible convergence rates and stability. As a test bed, we use the standard steady advection-diffusion equation and the following Dirichlet boundary conditions

$$\mathrm{Pe}\,\frac{\partial u}{\partial x} - L\,\frac{\partial^2 u}{\partial x^2} = 0 \tag{48a}$$

$$u(x=0) = 0; \quad u(x=L) = 1 \tag{48b}$$

Its solution characteristics are governed by the global Péclet number $\mathrm{Pe} = aL/D$, where parameters a, D and L are the velocity, the diffusion coefficient and the length of the domain, respectively. For increasing Pe, the exponential boundary layer at the right hand end of the domain steepens. An in-depth discussion of this problem and its exact solution can be found in [120, 121].

For the case of a moderate Péclet number Pe=10, the plots of Figs. 44 and 45 compare the rates of convergence of the error in the L^2 norm and H^1 semi-norm, obtained with weighted collocation, standard collocation and Galerkin for even polynomial degrees (quadratic, quartic) and odd polynomial degrees (cubic, quintic), respectively. The different methods use the same B-spline basis that is uniformly refined over the complete domain, and imply a change of point evaluations as shown in Fig. 43. The following observations can be made: The rates of convergence in L^2 and H^1 achieve the best possible rates in both collocation schemes ($\mathcal{O}(p)$ for even p, $\mathcal{O}(p-1)$ for odd p). Weighted collocation seems to

involve a lower error constant C (see Eq. (35)), which decreases the error level in comparison to standard collocation. However, it does not achieve an improvement of convergence rates over standard collocation, although the fact that weighted collocation passes more information to the system matrix might have raised some hope in that direction. Furthermore, the equivalence of convergence rates obtained with collocation and Galerkin in H^1 for even polynomial degrees becomes evident in Fig. 44b, where the convergence curves of weighted collocation and Galerkin coincide. For L^2 and odd polynomial degrees, Galerkin converges at a higher rate than both collocation schemes.

We also tested the stability of weighted isogeometric collocation. To this end, we raised the Péclet number Pe in steps of one order of magnitude from zero (pure diffusion) to 1,000 (strong advection domination) and varied the polynomial degree p from quadratics to degree eight. For all resulting combinations of Pe and p, we did not encounter any unstable solution behavior. However, we observed an oscillatory behavior for the case of advection domination, which disappeared when the basis was refined enough to capture the boundary layer on the right hand end. The issue of oscillations for high Péclet numbers is well known for standard Galerkin discretizations, and is usually addressed by consistent stabilization techniques [121, 122]. In the framework of isogeometric collocation, we performed some initial tests of collocation-point upwinding, for which a brief discussion and examples in 2D are given in Appendix B.

6. Adaptive isogeometric collocation in one dimension

As shown in the previous section, weighted isogeometric collocation using the fine-scale Greville abscissae as collocation points is stable and converges with the best possible rate. Moreover, it allows coincident collocation points, which is essential for the use of several levels of hierarchically refined NURBS basis functions. However, it requires more point evaluations than standard IGA collocation, which lessens the advantage of collocation in terms of computational efficiency.

6.1. Standard and weighted collocation across a hierarchy of meshes

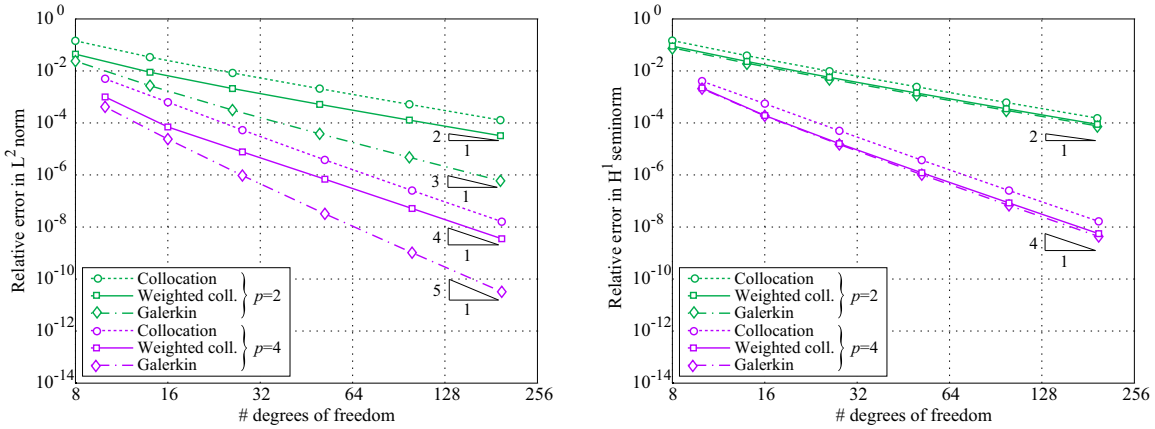
A resolution of this dilemma is a combination of standard isogeometric collocation, weighted collocation and hierarchical refinement of NURBS, performed in such a way that all relevant advantages can be maintained. Our basic idea, which we simply refer to in the following as *adaptive isogeometric collocation*, goes as follows:

Adaptive isogeometric collocation:

- Use hierarchical refinement of NURBS to establish an analysis suitable basis that adaptively resolves local features of the problem.
- Use collocation for each basis function, whose coarse-scale collocation point is located within or on the boundary of a knot span, which does not overlap with basis functions of finer-scale hierarchical levels. Overlapping with basis functions of coarser scales is allowed. The corresponding collocation point is derived from the Greville abscissae of the current hierarchical level k, where the basis function is defined.
- Use weighted collocation for each basis function, whose coarse-scale collocation point is located within a knot span that overlaps with basis functions of finer-scale hierarchical levels. The corresponding group of collocation points is derived from the fine-scale Greville abscissae of the next hierarchical level $k+1$.

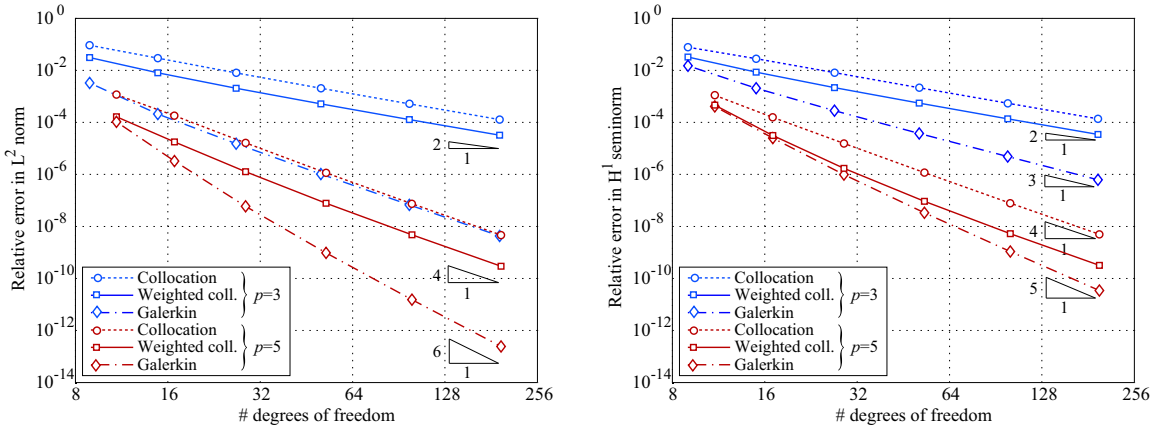
On the one hand, the application of the weighted collocation scheme in regions where basis functions of different hierarchical levels overlap allows coincident collocation points across hierarchical levels and thus avoids linear dependencies in the system matrix that occurred in the standard scheme (see Section 5.1). On the other hand, the application of the standard collocation scheme in single-level regions, which can be expected to cover the majority of the domain, effectively limits the number of point evaluations and thus preserves the fundamental advantage of collocation. Moreover, this strategy is consistent in the sense that it is generally valid irrespective of the polynomial degree and that it applies to any configuration of hierarchical levels. Furthermore, collocation points can be generated easily and implemented seamlessly within hierarchical refinement routines.

The idea of the restriction of weighted collocation to basis functions whose collocation points are located in regions of overlapping hierarchical levels is illustrated in Fig. 46 for a one dimensional multi-level patch of quadratic B-splines. The color of the collocation points indicates the group of B-splines to which they are assigned. Blue, purple and green collocation points correspond to the Greville abscissae of the current scale k, while red and orange points correspond to the Greville abscissae of the next finer scale $k+1$ according to the weighted collocation concept.



(a) *Error in* L^2. **(b)** *Error in* H^1.

Figure 44: *Weighted collocation: Convergence for uniform h-refinement of quadratic and quartic B-splines.*



(a) *Error in* L^2. **(b)** *Error in* H^1.

Figure 45: *Weighted collocation: Convergence for uniform h-refinement of cubic and quintic B-splines.*

6.2. Truncation of weighted collocation points

The method of adaptive isogeometric collocation can be further simplified by considering the concept of a truncated hierarchical basis, which was recently introduced by Giannelli et al. [70]. The truncation of a hierarchical B-spline basis recognizes that some fine-scale basis functions of level k are contained implicitly in some B-spline basis function of coarser scales, and eliminates them by subtracting finer-scale from coarser-scale basis functions. From a mathematical point of view, this procedure corresponds to a normalization of the

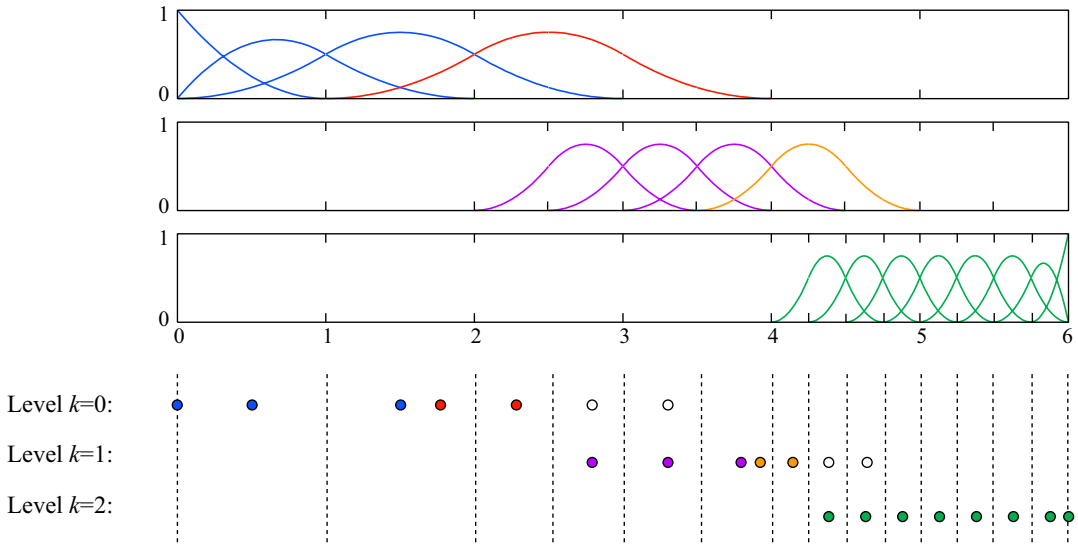


Figure 46: *Adaptive isogeometric collocation for a two-level quadratic B-spline patch.*

hierarchical basis.

For the present example of the one-dimensional hierarchical B-spline patch in Fig. 46, truncation can be illustrated as follows. We focus on the red basis function shown in Fig. 46 on level k=0 and its subdivision split into fine-scale B-splines, which are separately plotted in Fig. 47. As a consequence of the two scale relation Eq. (36), the two first basis functions of the next hierarchical level k=1 are equivalent to two fine-scale B-splines of the red basis function shown in Fig. 47. The multiplicity of B-splines on the two neighboring levels can be removed by subtracting the two fine-scale B-splines from the coarse-scale red function. Recalling that the choice of weighted collocation points is motivated by the fine-scale Greville abscissae (see Fig. 42), we immediately see that the number of weighted collocation points for the truncated basis function of Fig. 47 naturally decreases as a consequence of the removal of fine-scale B-splines. Thus, in analogy to the truncation of a basis function, the corresponding set of weighted collocation points can also be truncated.

Replacing original basis functions of a hierarchical basis by their truncated counterparts leads to a normalization of the hierarchical basis with increased sparsity and better conditioning of the corresponding system matrix [63, 70]. However, the normalized basis still spans the same space as the set of original basis functions and thus maintains exactly the same approximation power. We can therefore argue here that groups of weighted collocation points can be reduced according to a possible truncation, no matter whether we truncate the corresponding basis function in the hierarchical basis or not. This is illustrated in Fig. 46, where the white collocation points have been truncated, although the basis functions remain

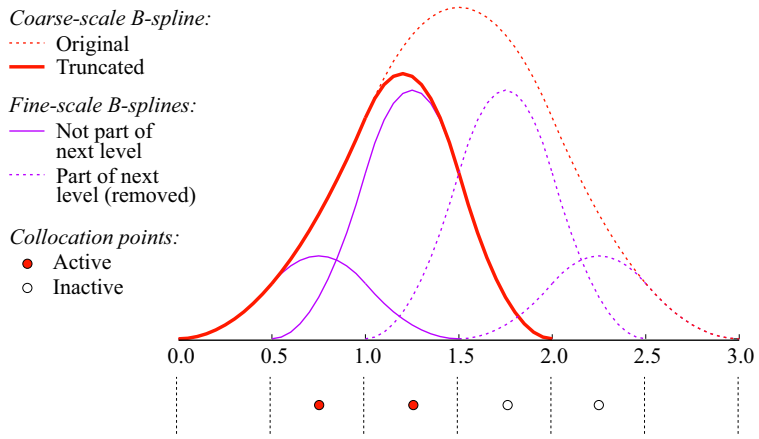
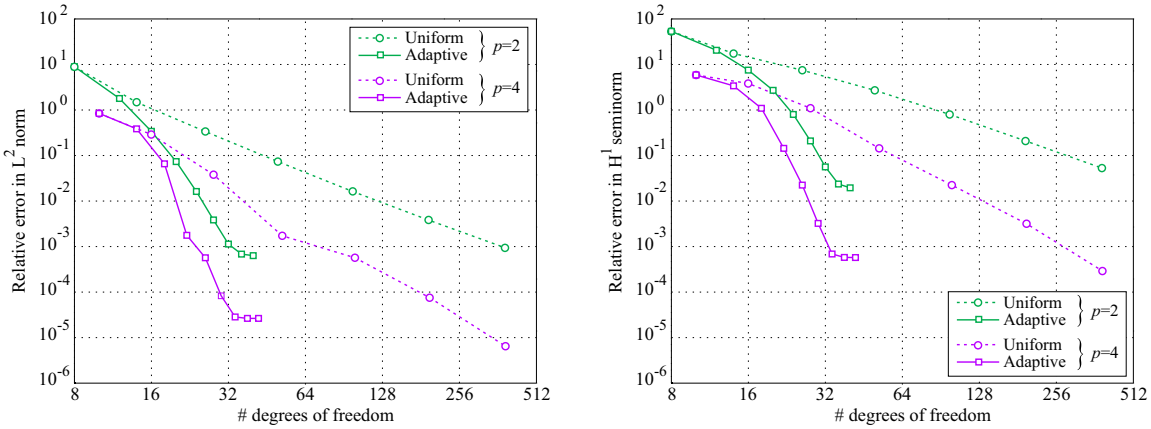


Figure 47: *The concept of truncation for hierarchical basis functions and weighted collocation points.*

the original B-splines. It should be noted that the concept of truncation constitutes a simplification of the adaptive IGA collocation, but does not affect its efficiency in terms of the number of point evaluations, since each truncated point corresponds to a coincident point on the next hierarchical level k+1.

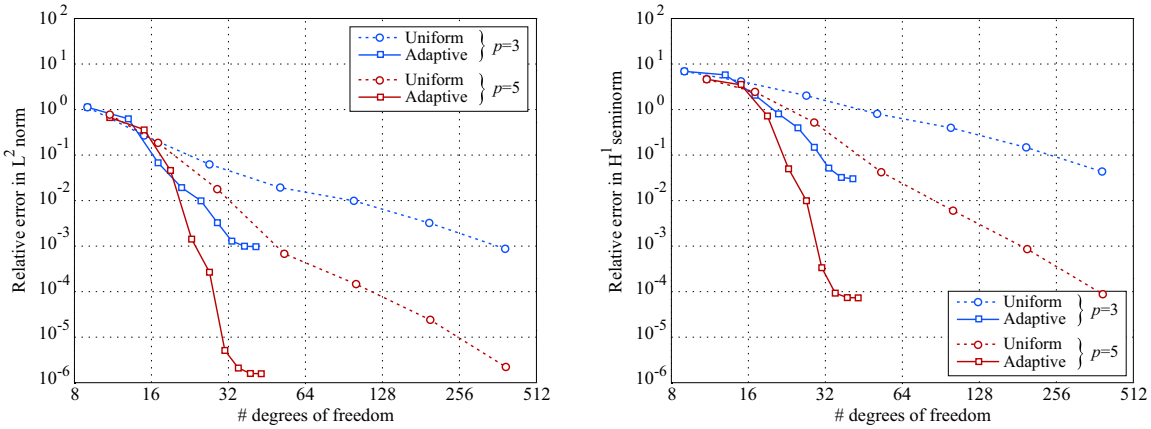
6.3. A simple model problem in 1D revisited

We test the efficiency of adaptive isogeometric collocation with the 1D steady advection-diffusion problem discussed in Section 5.4. A Péclet number of Pe=150 leads to a boundary layer at the right hand end of the 1D domain, which involves a very high gradient. We use a sequence of hierarchical bases in the sense of Fig. 46 to obtain an accurate solution. In each refinement step, we generate an additional hierarchical level by a bisection of the four rightmost elements. Figures 48 and 49 show the corresponding convergence of the error in the L^2 norm and H^1 semi-norm, as obtained with hierarchical B-splines of even polynomial degree (quadratic, quartic) and odd polynomial degree (cubic, quintic), respectively. Furthermore, they show the convergence obtained by uniform h-refinement of the corresponding B-spline patch that is based on a bisection of all elements over the complete domain in each refinement step. It can be observed that due to the local resolution of the boundary layer, adaptive isogeometric collocation achieves rates of convergence, which are far higher than those of uniform refinement. To arrive at the final error level in both the L^2 and H^1 cases, the hierarchical bases require about one order of magnitude fewer degrees of freedom than uniform h-refinement. After several adaptive refinement steps, the largest part of the error does not stem from the excessively refined right boundary anymore, so that the convergence rates of the adaptive solutions level off.



(a) *Error in* L^2. **(b)** *Error in* H^1.

Figure 48: *Adaptive isogeometric collocation: Convergence of quadratic and quartic B-splines.*



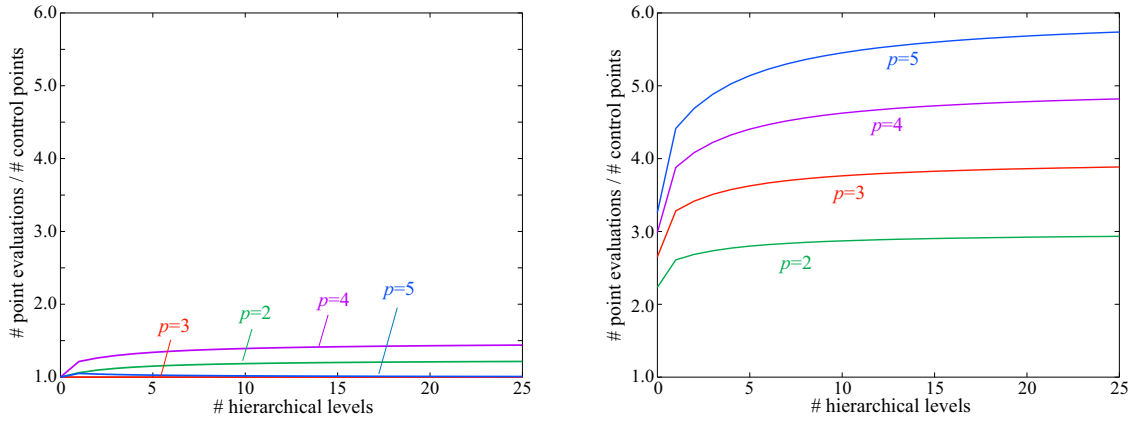
(a) *Error in* L^2. **(b)** *Error in* H^1.

Figure 49: *Adaptive isogeometric collocation: Convergence of cubic and quintic B-splines.*

6.4. Computational efficiency in 1D

The key motivation for the use of isogeometric collocation is its interpretation as a stable higher-order one-point quadrature scheme, which requires only one point evaluation per control point (i.e. node) [45, 46]. A suitable measure for the computational efficiency of a method in this sense is the ratio of the number of evaluations at quadrature or collocation points $\tilde{n}$ over the number of control points n_{cp} in the system

$$r_{eff} = \frac{\tilde{n}}{n_{cp}} \tag{49}$$



(a) *Adaptive IGA collocation.* **(b)** *Galerkin (Gauss quadrature).*

Figure 50: *Computational efficiency in terms of r_{eff}, i.e. the number of point evaluations per control point, for a one-dimensional hierarchical basis, constructed in the sense of Fig. 46.*

Standard IGA collocation is optimal, since it automatically leads to the minimum of r_{eff}=1.0. In adaptive isogeometric collocation, the ratio is larger than the optimum due to the use of weighted collocation. However, only a few basis functions are affected due to its restriction to the transition regions between hierarchical levels, so that r_{eff} always remains very close to the optimum of one. With respect to a Galerkin method, adaptive isogeometric collocation can still be characterized as a one-point quadrature scheme, whose computational cost in terms of point evaluations is considerably smaller than in a corresponding Galerkin scheme. Moreover, we have seen in Section 3.1 that the cost of one quadrature point in a Galerkin method is considerably larger than the cost of one collocation point. With respect to standard IGA collocation, we expect that the resolution of local features of adaptive IGA collocation leads to a sizable increase in the overall computational efficiency, in particular for large problems in two and three dimensions, which by far exceeds the little extra cost indicated by the slight deviation from the optimum r_{eff}=1.0.

We illustrate this statement by comparing the cost of point evaluations per control point for analysis with the hierarchically refined B-spline patch of Fig. 46. We consider adaptive IGA collocation and a Galerkin method [63] that uses full Gauss quadrature in each knot span. Figures 50a and 50b plot the ratio r_{eff} of Eq. (49) with increasing number of hierarchical levels for adaptive IGA collocation and Galerkin, respectively. The results confirm our initial argument. Despite the additional points due to weighted collocation in the transition regions between hierarchical levels, the ratio r_{eff} for adaptive IGA collocation stays

close to the optimum of one throughout all polynomial degrees p considered. In particular, it is considerably lower than the corresponding r_{eff} of full Gauss quadrature required by the Galerkin method. Note that the partial coincidence of coarse- and fine-scale collocation points of neighboring levels for odd polynomial degrees p leads to better ratios in comparison to even p, where the set of coarse- and fine-scale collocation points are completely distinct.

7. Adaptive isogeometric collocation in two and three dimensions

In the following, we demonstrate that the concept of adaptive isogeometric collocation that has been presented in the previous sections for the 1D case works equivalently well in higher dimensions. To this end, we examine 2D elliptic problems with smooth and rough solutions as well as advection-diffusion benchmarks in 2D and 3D. The results confirm that adaptive IGA collocation achieves the best possible rates of convergence in higher dimensions, works well for rough solutions, and considerably reduces the computational cost in terms of point evaluations in comparison to full Gauss quadrature of corresponding Galerkin discretizations.

7.1. Annular ring with a smooth solution

With the first numerical example, we demonstrate that adaptive isogeometric collocation achieves the best possible rates of convergence in higher dimensions. To this end, we consider the elliptic PDE

$$-\Delta u + u = f \qquad \text{in } \Omega \tag{50a}$$

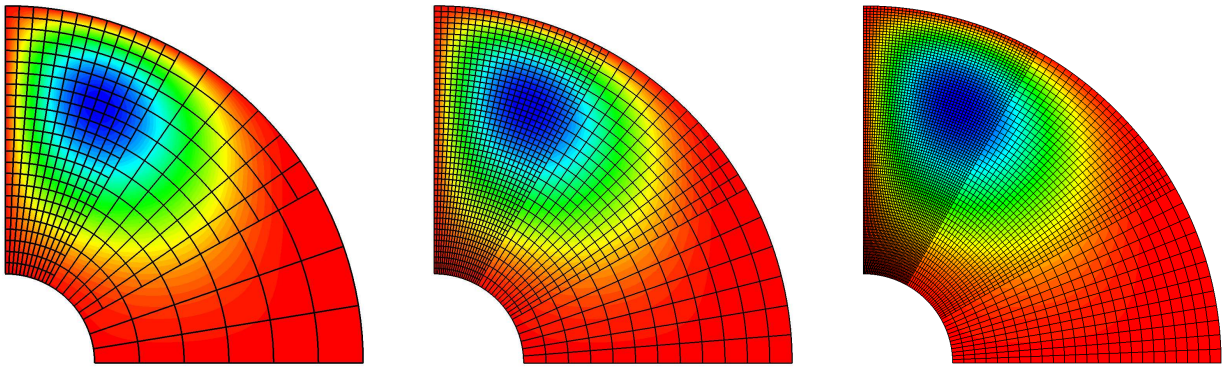
$$u = 0 \qquad \text{on } \partial\Omega \tag{50b}$$

The problem is defined over a quarter of an annular ring with inner radius R_{in}=1.0 and outer radius R_{out}=4.0. The quarter annulus is located within the positive quadrant of the Cartesian coordinate system x, y. The source term f is manufactured in such a way that the exact solution to the PDE over the quarter annulus reads

$$u = \phi^2(r^2 - 1)(r^2 - 16)\sin(x) \tag{51}$$

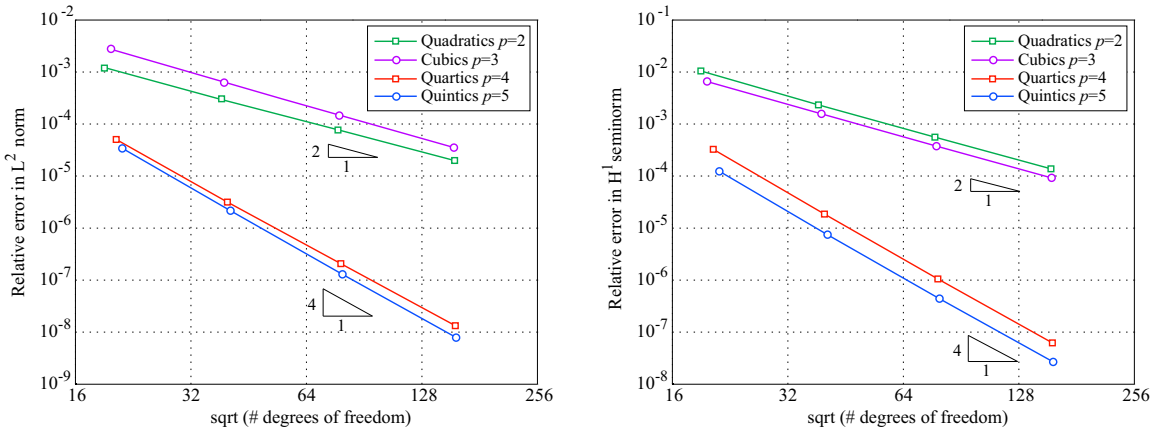
with polar coordinates $r = x^2 + y^2$ and $\phi = \arccos(x/r)$. The solution is illustrated by its numerical approximations plotted on the corresponding meshes in Fig. 51.

We start from a 5×9 single-level NURBS mesh, which represents the geometry of the quarter annular ring exactly. Note that throughout the hierarchical refinement, the geometry



(a) *Initial three-level mesh.* **(b)** 1^{st} *refinement step.* **(c)** 2^{nd} *refinement step.*

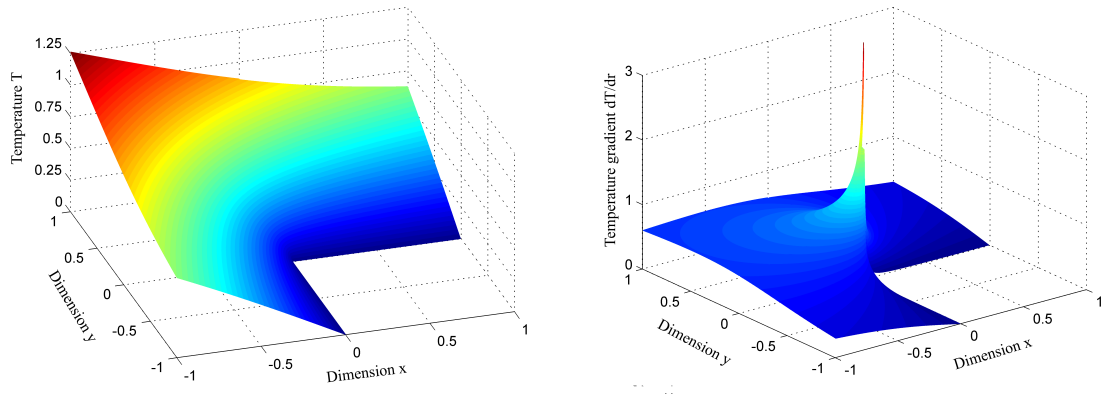
Figure 51: *Uniform refinement of a three-level hierarchical NURBS mesh for a quarter annulus. The contours show the approximation of the solution field for quadratic NURBS.*



(a) *Error in* L^2. **(b)** *Error in* H^1.

Figure 52: *Adaptive isogeometric collocation: Best possible rates of convergence for the 2D quarter annulus.*

is represented by the initial mesh in the sense of Eq. (42). This initial mesh is then graded towards the expected location of the highest gradients at the left hand boundary by two levels of hierarchically refined NURBS (see Fig. 51a). The three-level mesh of Fig. 51a is then refined uniformly, where in each refinement step all elements over the complete domain are quadrisected. The meshes after the first and the second refinement step are shown in Figs. 51b and 51c, respectively. The corresponding convergence of the relative error in the L^2 norm and H^1 semi-norm are plotted in Figs. 52a and 52b, respectively, for quadratic, cubic, quartic and quintic NURBS. The plots clearly show that adaptive isogeometric collocation



(a) *Smooth solution T Eq. (53).* **(b)** *Singularity in the derivative $\partial T/\partial r$.*

Figure 53: *Heat conduction over the L-shaped domain with reentrant corner.*

based on hierarchical refinement and weighted collocation leads to the best possible rates of convergence of $\mathcal{O}(p)$ for even and of $\mathcal{O}(p-1)$ for odd polynomial degrees p.

7.2. L-shaped domain with a "rough" solution

The second numerical example consists of a stationary heat conduction problem defined over an L-shaped domain $\Omega = [-1,1]^2 \backslash [0,1] \times [-1,0]$. It is governed by the Laplace equation

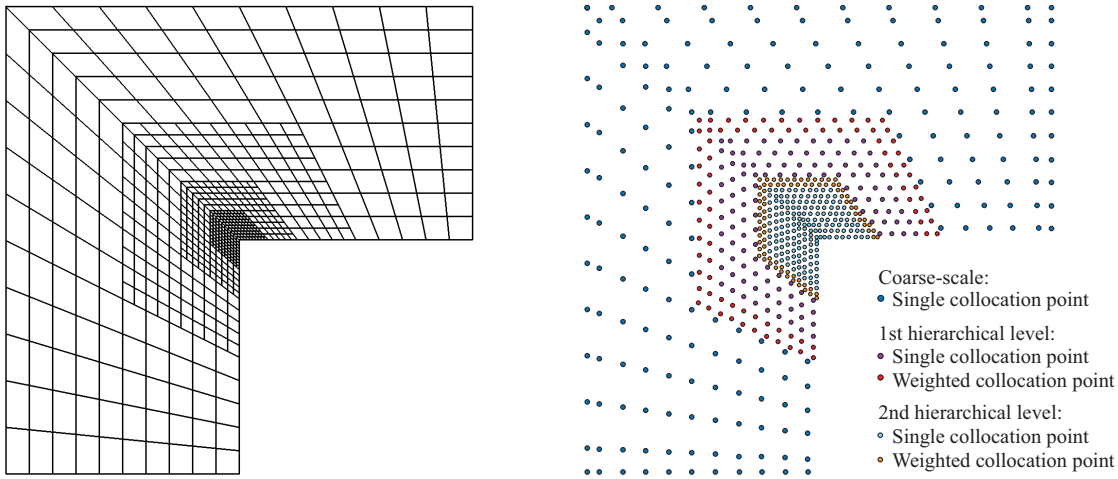
$$\Delta T = 0 \qquad \text{in } \Omega \tag{52}$$

with homogeneous Dirichlet boundary conditions on the reentrant edges $\Gamma_{D,1} = [0 \leq x \leq 1]$ and $\Gamma_{D,2} = [-1 \leq y \leq 0]$. On the rest of the boundaries, Neumann boundary conditions are imposed that satisfy the exact solution

$$T = r^{\frac{2}{3}} \sin\left(\frac{2\phi}{3}\right) \tag{53}$$

with polar coordinates $r = x^2 + y^2$ and $\phi = \arccos(x/r)$. The solution Eq. (53) can be characterized as "rough", since its gradient exhibits a singularity at the reentrant corner. The exact solution and its first derivative with respect to r are plotted in Figs. 53a and 53b.

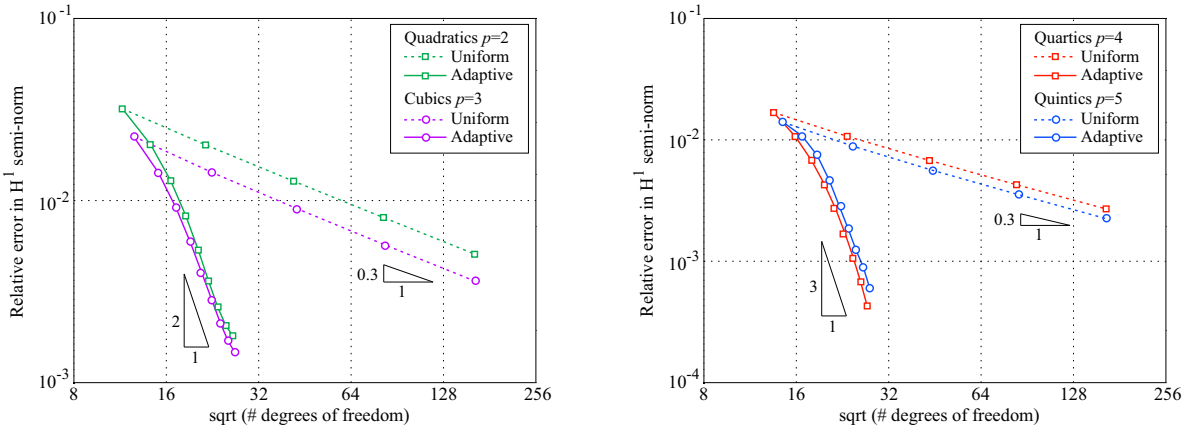
We discretize the L-shaped domain by two patches of 10×10 NURBS elements. We subsequently add an increasing number of hierarchical levels around the reentrant corner by splitting half of the NURBS elements on the currently finest hierarchical level in each parametric direction of both patches. Figure 54a shows the adaptive NURBS mesh for



(a) *Four-level hierarchical mesh refined towards the reentrant corner.*

(b) *Collocation points corresponding to the three-level quadratic basis.*

Figure 54: *Adaptive isogeometric collocation: Discretization and collocation points for the L-shaped domain.*



(a) *Error in H^1 for p=2,3.*

(b) *Error in H^1 for p=4,5.*

Figure 55: *Adaptive isogeometric collocation: Convergence in the presence of a corner singularity.*

the four-level case. Figure 54b plots the corresponding set of collocation points for the case of three-level quadratic NURBS. Its coloring refers to single collocation points of the standard collocation scheme, each of which can be attributed to one basis function, and to weighted collocation points in the transition regions of neighboring hierarchical levels. It can be clearly observed that the majority of points are standard collocation points for basis functions away from the transition regions that contribute towards an optimum ratio of one

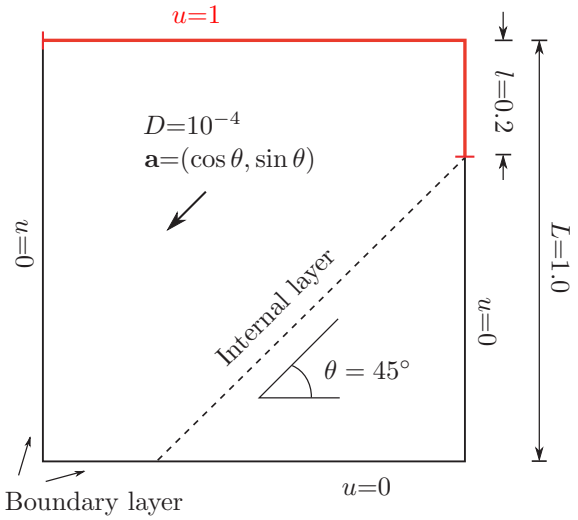


Figure 56: *Advection skew to the mesh in 2D: Problem definition.*

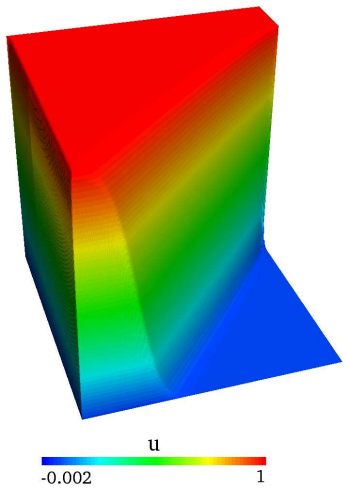


Figure 57: *Overkill solution computed on a mesh of 480×480 quadratic B-spline elements.*

point evaluation per degree of freedom.

For convenience, we carry out the analysis on one of the patches, using symmetry boundary conditions along the patch interface. The convergence of the relative error in the H^1 semi-norm is plotted in Figs. 55a and 55b for quadratic, cubic, quartic and quintic NURBS basis functions, respectively. It can be observed that adaptive isogeometric collocation using an increasing number of hierarchical levels improves the convergence rates by around one order of magnitude with respect to uniform refinement of the complete domain. The present numerical example thus confirms that adaptive IGA collocation works well for rough problems.

7.3. Advection skew to the mesh

For the remaining numerical examples, we return to the advection-diffusion PDE Eq. (10). A typical benchmark test [122, 123] for adaptive refinement is sketched in Fig. 56, which was examined in a Galerkin context for uniform k-refinement [2], local T-spline refinement [124, 125] and local hierarchical refinement of NURBS [63]. The velocity $\boldsymbol{a}$ is inclined to the mesh at 45° and the diffusivity D is chosen extremely small, so that the problem is dominated by advection, resulting in a very high global Péclet number of 10^4. Thus, we expect sharp interior and boundary layers, which require stable numerical techniques in addition to increased resolution to be accurately captured. A corresponding Galerkin overkill solution computed on a uniform mesh of 480×480 quadratic B-spline elements is shown in Fig. 57.

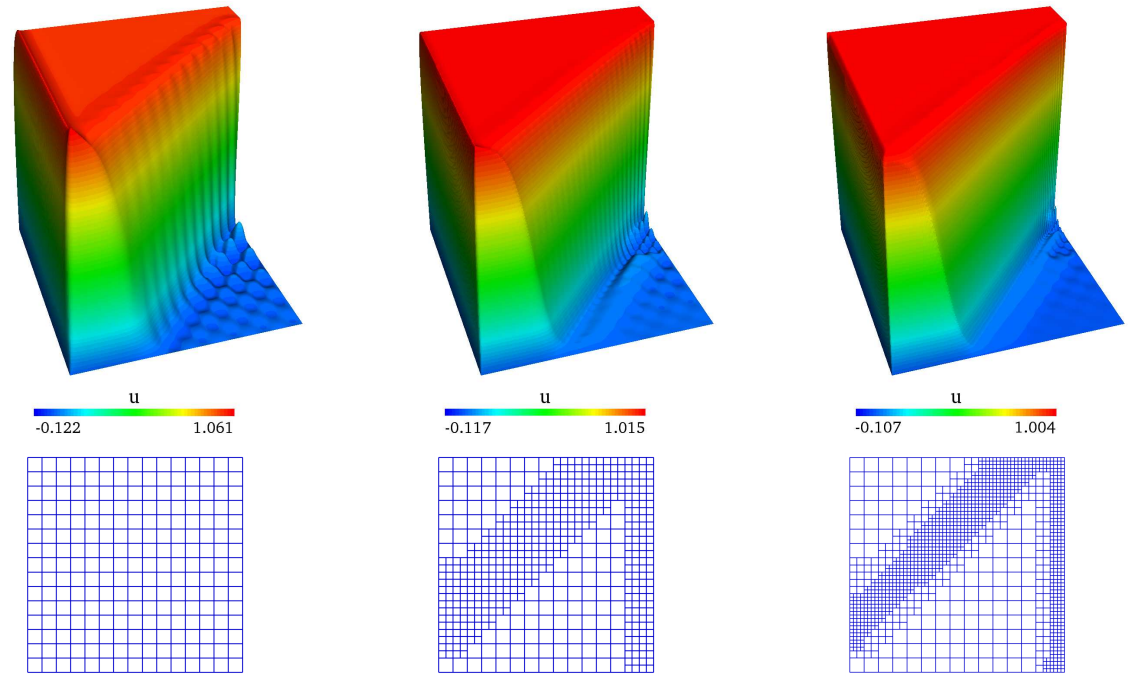
We investigate the adaptive resolution of the internal and boundary layers with the

present hierarchical refinement approach, starting from a 15×15 grid of quadratic B-splines. We satisfy boundary conditions strongly at the inflow and outflow boundaries. To this end, we keep track of the scaling factors s_i of Eq. (38) in the subdivision split of boundary functions in order to satisfy partition of unity at the non-zero inflow boundary. Furthermore, we use upwinding of the collocation points to prevent unphysical oscillations (see Appendix B). Employing an automatic refinement scheme based on a gradient based error indicator [63], we obtain a sequence of hierarchical meshes with corresponding solution fields shown in Figs. 58a through 58f.

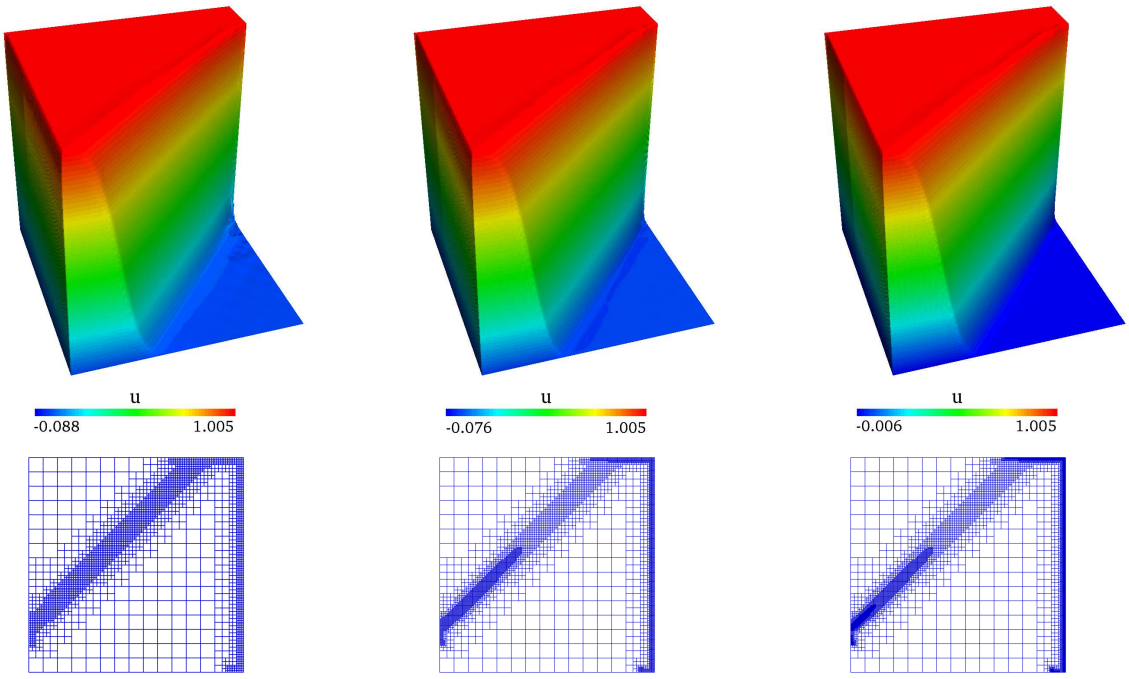
It can be observed that the refinement captures the location of the internal and the boundary layers very well. Despite the high Péclet number and the high degree of local refinement with a larger number of hierarchical levels, no stability or robustness issues in the adaptive isogeometric collocation scheme were encountered. We can observe some over- and undershooting of the adaptive solution along the internal layer as also reported for T-spline [124, 125] and hierarchical B-spline refinement [63] with the Galerkin method. We observe that five hierarchical refinement steps are required to get control over the undershooting close to the jump at the inflow boundary. While providing the same fine-scale element size around the internal and boundary layers, the finest adaptive mesh of Fig. 58f with 8,187 degrees of freedom requires, at a comparable level of accuracy, only about 3.5% of the degrees of freedom of the uniform overkill mesh of Fig. 57 with 230,400 degrees of freedom. Finally, we would like to point out the high quality of the refinement in terms of locality, as the hierarchical elements of the finest level show no propagation through the mesh.

7.4. Advection-diffusion in a rotating cylinder

With the next numerical example, we show that adaptive isogeometric collocation can be extended to 3D solid elements in a straightforward manner. We consider the advection-diffusion benchmark introduced in [63], which consists of a three-dimensional cylinder that rotates around its axis with tangential velocity $a_\theta = \omega r$ and radial velocity $a_r = 0$. At the same time, a flow of constant axial velocity a_z is assumed, which results in a helical plume of the concentration that emerges from the fixed local inflow boundary condition $u = 1$. A sketch of the problem is given in Fig. 59. The geometry of the cylinder is described exactly by four equal NURBS patches, each of which covers one quarter of the cylinder and consists of 9×9×50 quadratic NURBS elements in (r, θ, z)-directions, respectively. The geometry is represented throughout the refinement process by the initial mesh in the sense of Eq. (42). Dirichlet boundary conditions are prescribed strongly at the inflow and outflow boundaries at both ends, where at the non-zero inflow, we satisfy partition of unity of the boundary basis



(a) *Initial - 225 dofs.* **(b)** 1^{st} *step - 507 dofs.* **(c)** 2^{nd} *step - 1,180 dofs.*



(d) 3^{rd} *step - 2,649 dofs.* **(e)** 4^{th} *step - 4,984 dofs.* **(f)** 5^{th} *step - 8,187 dofs.*

Figure 58: *Sequence of hierarchical meshes and corresponding solution fields for the 2D benchmark problem dominated by advection skew to the mesh.*

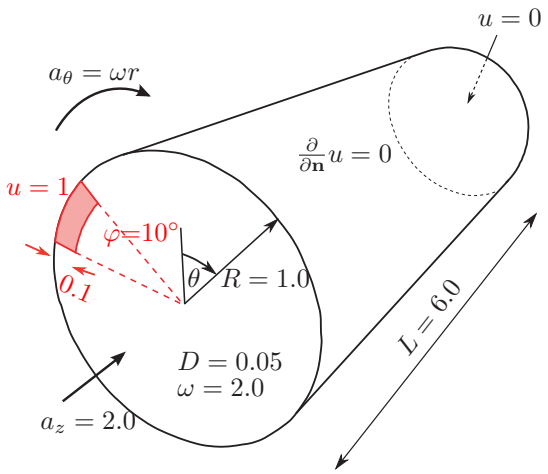


Figure 59: *Advection-diffusion in a rotating cylinder: System sketch.*

functions by taking into account the scaling factors s_i of Eq. (38) throughout the refinement process. At the radial boundary of the cylinder, we impose no-flux Neumann conditions.

To accurately resolve the boundary and internal layers along the plume, we apply an automatic adaptive refinement procedure based on a gradient-based error indicator [63]. Figures 60 and 62 show the resulting complete adaptive mesh with two levels of hierarchical NURBS as well as the initial mesh and the sets of finest elements after each refinement step. The corresponding solution obtained with the adaptive mesh of Fig. 60 is plotted in Fig. 61. We observe that the refinement accurately traces the steepest gradients of the concentration u. A uniform discretization that yields a plume resolution with the same small element size as in the adaptive mesh requires a globally refined mesh of 36×144×200 quadratic NURBS elements with 1,095,200 degrees of freedom, whereas the present adaptive mesh requires only 104,017 degrees of freedom.

7.5. Computational efficiency in higher dimensions

We emphasize again the key motivation for the use of isogeometric collocation, i.e. its interpretation as a stable higher-order one-point quadrature scheme with only one point evaluation per control point (i.e. node) [45, 46]. Measuring the computational efficiency by the ratio of the number of evaluations at collocation points over the number of control points in the system (see Eq. (49) in Section 6.4), we have an optimum of r_{eff}=1.0 for IGA collocation of a single-level NURBS basis. For adaptive IGA collocation, this ratio is increased due to weighted collocation, which takes into account more than one collocation point for some of the basis functions. However, due to its locally restricted application in transition regions between hierarchical levels, we demonstrated in Section 6.4 that for

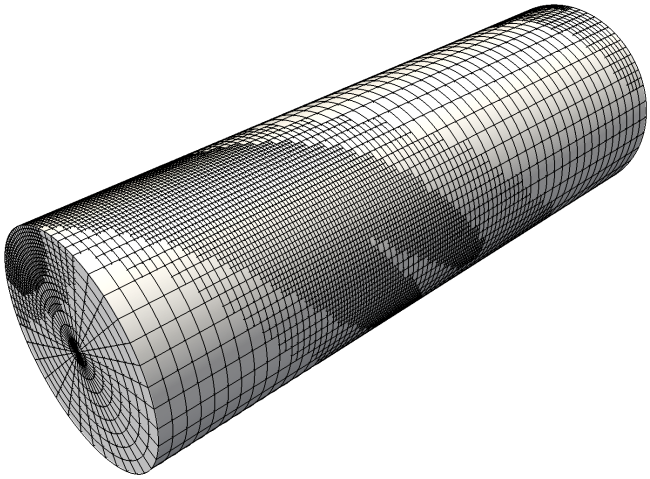


Figure 60: *Adaptive mesh with two levels of hierarchical quadratic NURBS.*

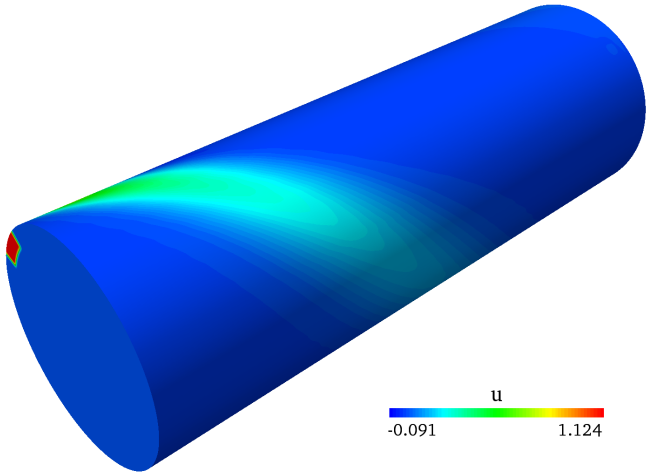
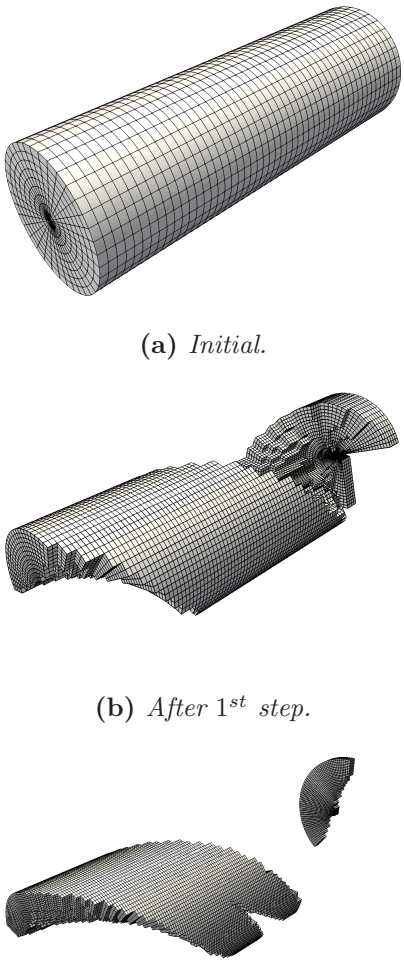


Figure 61: *Solution field obtained with the adaptive mesh of Fig. 60.*



(a) *Initial.*

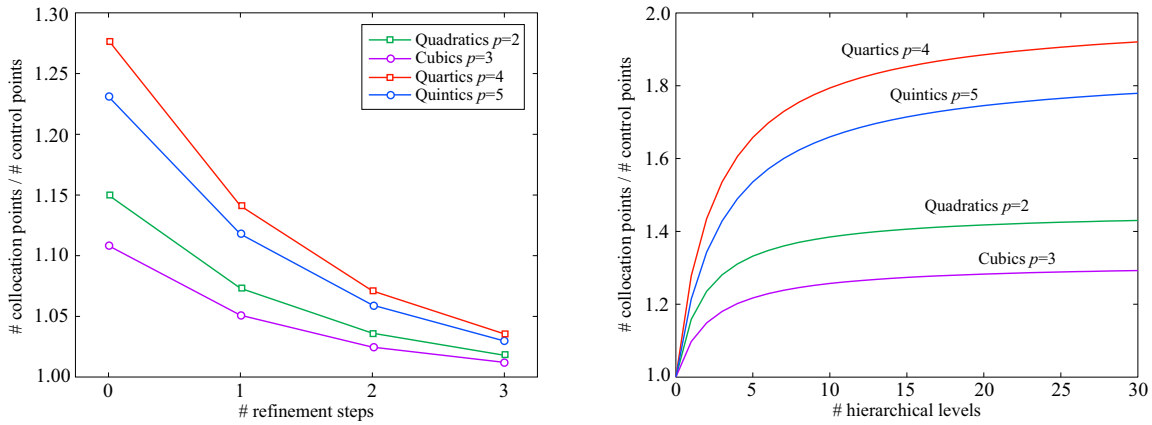
(b) *After* 1^{st} *step.*

(c) *After* 2^{nd} *step.*

Figure 62: *Finest elements at different steps of the adaptive procedure.*

the one-dimensional case the ratio r_{eff} always remains close to the optimum of one. In the following, we show that this argument equally holds for higher dimensions.

First, we compare the two different refinement strategies that we applied in Section 7.1 for the quarter annulus and in Section 7.2 for the L-shaped domain, respectively. The former starts with a fixed number of hierarchical levels in the initial mesh and applies uniform mesh refinement to the initial hierarchically graded mesh. The latter starts with an unrefined uniform mesh and locally increases the number of hierarchical levels, leading to a strong grading of elements from coarsest to finest scale. Figures 63a and 63b show that for



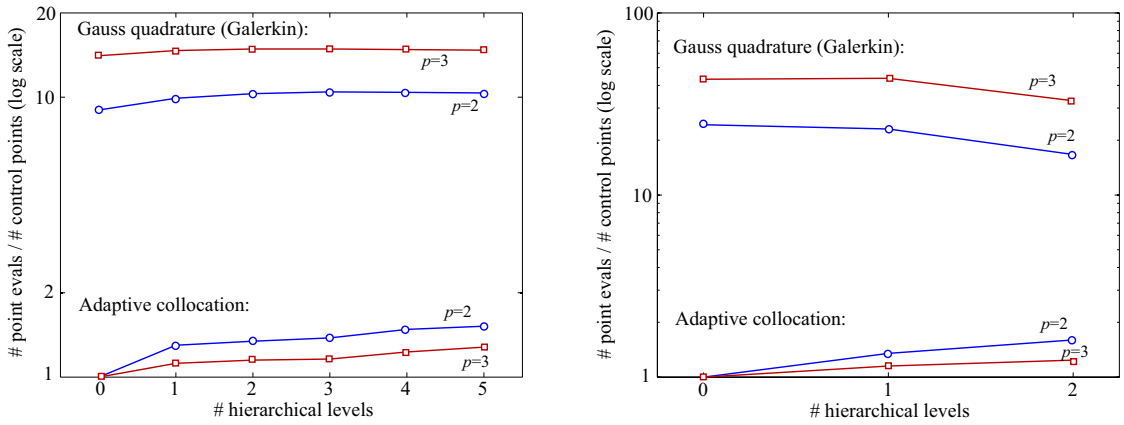
(a) *Quarter annulus of Fig. 51 (h-refinement of a hierarchically graded mesh): Each hierarchical level is uniformly refined, while the number of hierarchical levels is kept the same in each step.*

(b) *L-shaped domain of Fig. 54 (increase of local mesh grading): In each refinement step, an additional hierarchical level is added that increases the mesh density locally.*

Figure 63: *Computational efficiency r_{eff} of adaptive IGA collocation in terms of number of point evaluations per control point for two different refinement strategies.*

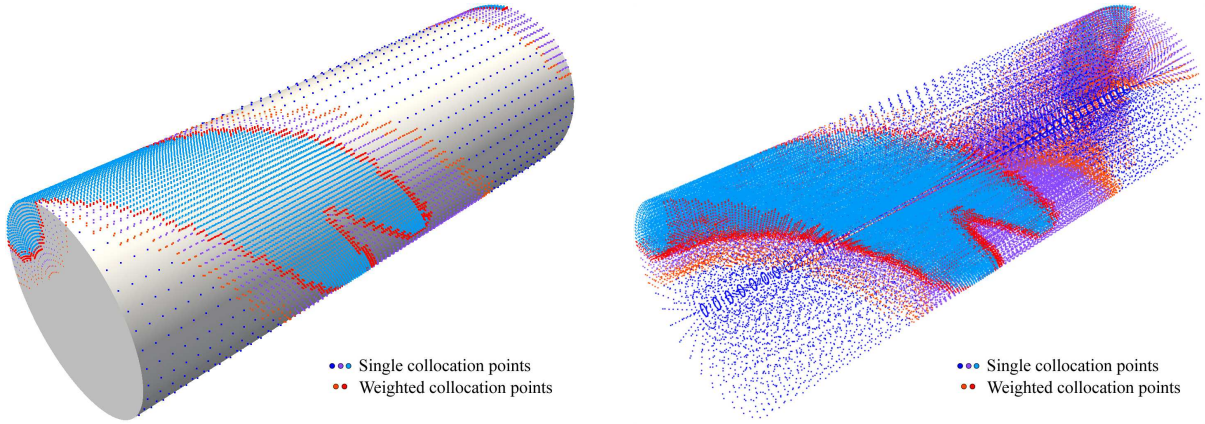
the quarter annulus, the ratio r_{eff} between point evaluations and control points converges quickly to the optimum r_{eff}=1.0, while for the L-shaped domain, the ratio r_{eff} converges towards values below 2.0 for all polynomial degrees p considered. We conclude that if a hierarchically graded mesh is uniformly refined, the ratio of point evaluations per degree of freedom recovers in the limit the optimal computational efficiency of standard single-level IGA collocation. If the grading of a mesh is increased by the addition of hierarchical levels, the ratio saturates to an asymptotic value acceptably "close" to its optimum r_{eff}=1.0.

Second, we focus on the 2D advection benchmark discussed in Section 7.3. Considering the topology of the area to be refined in each step, we can easily identify this as a sort of worst-case example. Due to the boundary and internal layers, refinement is required along lines rather than over an area, which is illustrated by the sequence of refined meshes in Figs. 58a through 58e. As a consequence, the hierarchical levels in each refinement step involve a large number of weighted collocation points due to the stretched transition regions, while the number of single-level collocation points emanating from the areas enclosed is comparatively small. Considering the refined meshes of Fig. 58, we construct corresponding quadratic and cubic basis functions and collocation points as well as full Gauss quadrature points required for Galerkin based IGA. Due to the unfavorable topology of refinement areas,



(a) *2D case (Section 7.3).* **(b)** *3D case (Section 7.4).*

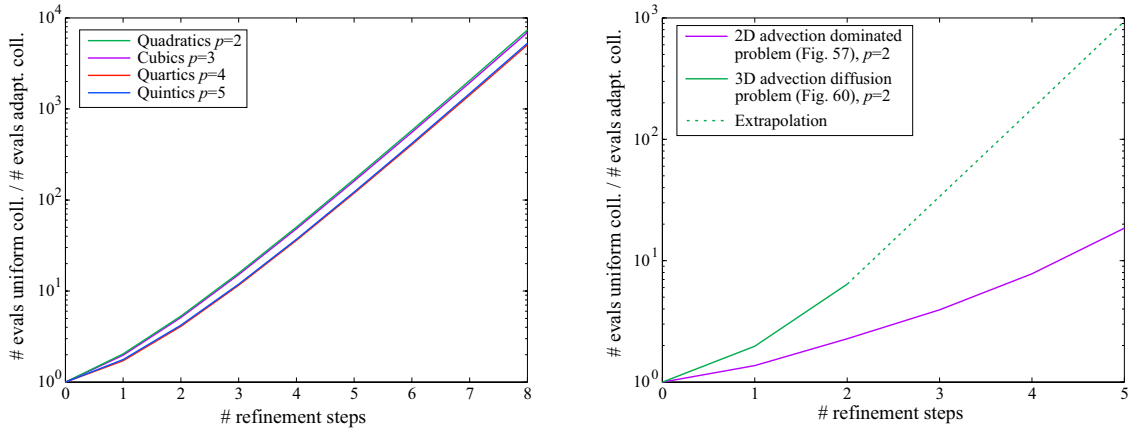
Figure 64: *Computational efficiency in terms of r_{eff} of adaptive collocation and of full Gauss quadrature required in the Galerkin method.*



(a) *Collocation points on the surface.* **(b)** *Cloud of all collocation points.*

Figure 65: *3D cylinder: Single collocation points uniquely assigned to one basis function and weighted collocation points for basis functions along the transition regions between hierarchical levels.*

the ratios r_{eff} plotted in Fig. 64a are higher than for the L-shaped domain, but still stay well below a value 2.0 for both p=2 and p=3. A comparison of the ratios r_{eff} obtained from collocation and full Gauss quadrature reveals that for the same adaptive NURBS basis in 2D, collocation cuts down the number of point evaluations per control point by about one order of magnitude. For the 3D case discussed in Section 7.4, the behavior is equivalent. Considering the refined meshes of Figs. 60 and 62, we again construct quadratic and cubic basis functions and compute corresponding collocation and Gauss quadrature points. Figure 64b illustrates



(a) *L-shaped domain (Section 7.2).* **(b)** *2D and 3D advection diffusion problems.*

Figure 66: *Comparison of uniform and adaptive collocation by the ratio of their point evaluations.*

the resulting computational efficiency in terms of the ratio r_{eff}, which is acceptably close to the optimum $r_{\mathit{eff}} = 1.0$ and one to two orders of magnitude smaller than that of full Gauss quadrature required by Galerkin based IGA. Figure 65 plots the collocation points for the adaptive mesh of Fig. 60. Comparing the size of the different sets of points, we see that the number of single collocation points by far outweighs the number of weighted collocation points, which explains its ratios r_{eff} being close to 1.0.

Finally, we briefly illustrate again the imperative requirement for local adaptivity in isogeometric collocation. To this end, we compute the relation of collocation points for a uniformly refined NURBS basis of the same fine-scale resolution with respect to collocation points for a hierarchically refined NURBS basis. The resulting number is a normalization with respect to adaptive collocation and thus tells us how many times more point evaluations are required by standard uniform collocation. Figure 66a shows the evolution of this number during the refinement of the L-shaped domain problem discussed in Section 7.2. For the finest resolution of the reentrant corner shown in Fig. 53, standard IGA collocation with a uniform NURBS basis requires 10,000 times (!) as many point evaluations as adaptive IGA collocation applied in the sense of Fig. 54. Figure 66b shows similar ratios for the advection diffusion examples discussed in Sections 7.3 and 7.4. In particular, we observe that the difference in point evaluations between uniform and adaptive hierarchical refinement further increases when we go from 2D to 3D problems.

8. Summary and conclusions

In this paper, we compared isogeometric collocation (IGA-C) with isogeometric Galerkin (IGA-G) and standard C^0 finite element methods (FEA-G) in terms of their computational efficiency. We first assessed the computational cost in floating point operations for the formation and assembly of stiffness matrices and residual vectors. The operation counts demonstrate that compared to IGA-G and FEA-G, IGA-C significantly reduces the computational cost. Second, we showed that in IGA-C, the bandwidth of the stiffness matrix and the cost of matrix-vector products are smaller than in IGA-G and FEA-G. These properties are important indicators for the performance of direct and iterative solvers, respectively. Third, we used a series of representative smooth and rough problems in 3D to numerically compare the different methods with respect to accuracy vs. the number of degrees of freedom as well as accuracy vs. the serial computing time on a single thread. We showed that IGA-C can be orders of magnitude faster than IGA-G and FEA-G to achieve a specified level of accuracy. We illustrated that IGA-C approximations using basis functions of even polynomial degree can be considered the sweet spot of collocation, since they offer the same convergence rate in H^1 with respect to the optimal rates of the Galerkin methods at a computational cost that is orders of magnitude smaller. We also observed that IGA-C approximations using basis functions of odd polynomial degree (in particular cubics) are not as efficient as those using basis functions of even degree, since they exhibit a less favorable ratio between best possible rates of convergence and computational cost. As a final thought, we discussed the significant potential of IGA-C in terms of efficient memory usage and parallelization that will put IGA-C even further ahead of Galerkin methods when computing time in parallel applications on modern multi-core machines is considered.

We then showed that local adaptivity in isogeometric collocation can be based on local hierarchical refinement of NURBS. In recent contributions on isogeometric collocation, the Greville abscissae derived from the knot vector of the NURBS basis have been shown to provide a suitable set of collocation points with favorable numerical properties. However, in the context of a hierarchical basis, the Greville abscissae of different hierarchical levels may generate coincident collocation points, which leads to the linear dependence of the equation system. To bypass this problem, we introduced the concept of weighted isogeometric collocation that derives each collocation equation from a group of collocation points, generated from the fine-scale Greville abscissae of the next hierarchical level. We showed that weighted IGA collocation leads to the best possible rates of convergence, however, at a higher computational cost than standard IGA collocation. We therefore applied weighted collocation

only in regions, where hierarchical levels overlap and coincident points are possible, and continued to collocate at single points in the rest of the domain. The resulting adaptive isogeometric collocation scheme combines several advantages. It reconciles collocation at the Greville abscissae with hierarchically refined NURBS basis functions, fully inheriting the favorable properties of both technologies. The key idea of local weighted collocation is acceptably simple and permits a straightforward implementation, in particular if hierarchical refinement routines already exist. Several numerical examples in one, two and three dimensions demonstrated that adaptive isogeometric collocation in this form converges with the best possible rates, while it preserves the key advantage in terms of a small number of point evaluations. In particular, we showed that the ratio between the number of point evaluations and the number of control points (i.e. nodes) in the system remains always close to the optimum of one and considerably below what is required in a corresponding Galerkin method. Moreover, adaptive IGA collocation demonstrated its robustness for "rough" and advection dominated problems, and no stability issues were encountered in the presence of a large number of hierarchical levels.

We believe that isogeometric collocation has the potential to offer a more efficient alternative to existing finite element technology. Some particularly promising research topics that need to be investigated are the appropriate treatment of boundary and patch interface conditions, fully nonlinear problems, locking free plate and shell formulations, three-dimensional solids and fluids, efficient parallel implementations, and large-scale industrial applications. There are many challenges and hurdles to be overcome, but the opportunity seems very significant. At the same time, we are convinced that more efficient quadrature schemes for isogeometric Galerkin methods are possible, and their investigation constitutes an important objective of future research in IGA.

Acknowledgments. D. Schillinger was supported by the German National Science Foundation (Deutsche Forschungsgemeinschaft DFG) under grant SCHI 1249/1-1. J.A. Evans, M.A. Scott, and T.J.R. Hughes were supported by grants from the Office of Naval Research (N00014-08-1-0992), the National Science Foundation (CMMI-01101007), and SINTEF (UTA10-000374), with the University of Texas at Austin. A. Reali was supported by the European Research Council through the FP7 Ideas Starting Grant n. 259229 *ISOBIO*, and by the Italian MIUR through the FIRB "Futuro in Ricerca" Grant n. RBFR08CZ0S.

Appendix A. Derivation of operation counts at a quadrature/collocation point

In the following, we provide details on the operation counts regarding the cost in floating point operations (flops) at each quadrature/collocation point for the formation and assembly of (a) the local stiffness matrix of a scalar problem (Laplace), (b) the local stiffness matrix of a vector problem (elasticity) and (c) the residual vector in an elastodynamics problem without damping. Operations required for the formation and assembly of local matrix and vector entities are separately listed in Appendices A.2, A.3, and A.4 for cases (a), (b) and (c), respectively. We note that in this paper each multiplication and each addition are considered as a single flop. We refer to isogeometric collocation, Galerkin based isogeometric analysis and C^0 finite element methods with abbreviations IGA-C, IGA-G and FEA-G, respectively. We neglect the cost of all control structures and do not use the symmetry of the Galerkin stiffness matrices, since this does not hold for non-symmetric problems such as advection-diffusion. We assume that IGA-C and IGA-G use NURBS to represent the geometry exactly and FEA-G uses the finite element mesh for the approximation of the geometry. For FEA-G, our counts are based on Bernstein polynomials which do not involve a rational mapping and whose cost is therefore comparable to standard C^0 basis functions. For IGA-C, we report operation counts for an interior collocation point that sees $(p+1)$ basis functions.

For the solution of the small systems of linear equations that occur during the computation of first and second derivatives in global coordinates, we assumed standard Gaussian elimination. Typically we need to solve many systems with the same coefficient matrix, but k different right hand sides. This requires only one forward elimination of the coefficient matrix and k forward eliminations and back substitutions of the right hand sides. The corresponding cost in flops can be computed with $2/3n^3 + 3/2kn^2 - (3k+4)/6n$, where n is the number of equations in the system.

The computation of univariate basis functions in local coordinates depends largely on function type and specific implementation. Therefore we neglect its cost, assuming it is small and comparable between methods. Alternatively, we could assume that univariate basis functions in local coordinates are precomputed for a typical element (IGA-G/FEA-G) or the required types and locations of collocation points, which actually many codes do (as for instance the code that we used for all computations shown in this paper). Operations related to the computation of multivariate basis functions and its derivatives in global coordinates are the same for the stiffness forms of case (a) and (b), and therefore reported only once in Appendix A.1.

For the residual forms in the elastodynamics case (c), we can optimize the evaluation

Data: Local coordinates $\{xi, eta\}$ of the current quadrature/collocation point;

Arrays of control point values $cpts(:,1), cpts(:,2)$ and weights $w(:,1)$ defining the geometry;

Functions $Bspline(*,i)$, $dBspline(*,i)$ and $ddBspline(*,i)$ providing the function values of the i^{th} univariate B-splines and its first/second derivatives in local coordinates;

Result: NURBS basis functions and its first/second derivatives in global coordinates ;

```
% 1. Form tensor products
for j=1:(p+1) do
    for i=1:(p+1) do
        B       ((j-1)*(p+1)+i)  =     BSpline(xi,i)  *     BSpline(eta,j);
        dBxi    ((j-1)*(p+1)+i)  =    dBSpline(xi,i)  *     BSpline(eta,j);
        dBeta   ((j-1)*(p+1)+i)  =     BSpline(xi,i)  *    dBSpline(eta,j);
        ddBxi   ((j-1)*(p+1)+i)  =   ddBSpline(xi,i)  *     BSpline(eta,j);
        ddBeta  ((j-1)*(p+1)+i)  =     BSpline(xi,i)  *   ddBSpline(eta,j);
        dBxieta ((j-1)*(p+1)+i)  =    dBSpline(xi,i)  *    dBSpline(eta,j);
    end
end

% 2. Multiply each B-spline function with corresponding weight
N     (1,:) =       B  .*  w(:,1);
dN    (1,:) =    dBxi  .*  w(:,1);
dN    (2,:) =   dBeta  .*  w(:,1);
ddN   (1,:) =   ddBxi  .*  w(:,1);
ddN   (2,:) = dBxieta  .*  w(:,1);
ddN   (3,:) =  ddBeta  .*  w(:,1);

% 3. Compute sums of B-spline functions
w_sum     = sum(N(1,:));
dw_xi     = sum(dN(1,:));
dw_eta    = sum(dN(2,:));
d2w_xi    = sum(ddN(1,:));
d2w_xieta = sum(ddN(2,:));
d2w_eta   = sum(ddN(3,:));

% 4. Compute NURBS basis functions and its first and second derivatives in local coordinates
ddN(1,:) = ddN(1,:)/w_sum - (2*dN(1,:)*dw_xi + N*d2w_xi)/w_sum^2 ...
           + 2*N*dw_xi^2/w_sum^3;
ddN(2,:) = ddN(2,:)/w_sum - (dN(1,:)*dw_eta + dN(2,:)*dw_xi ...
           +N*d2w_xieta)/w_sum^2 + 2*N*dw_xi*dw_eta/w_sum^3;
ddN(3,:) = ddN(3,:)/w_sum - (2*dN(2,:)*dw_eta + N*d2w_eta)/w_sum^2 ...
           +2*N*dw_eta^2/w_sum^3;
dN(1,:)  = dN(1,:)/w_sum - N*dw_xi/w_sum^2;
dN(2,:)  = dN(2,:)/w_sum - N*dw_eta/w_sum^2;
N        = N/w_sum;
```

continued on next page

continued from previous page

```
% 5. Compute Jacobian matrix
 dxdxi    =    [ dN(1,:)*cpts(:,1)     dN(2,:)*cpts(:,1);
                 dN(1,:)*cpts(:,2)     dN(2,:)*cpts(:,2)];

% 6. Set up the Hessian and the matrix of squared first derivatives
 d2xdxi2  =    [ddN(1,:)*cpts(:,1)    ddN(2,:)*cpts(:,1)    ddN(3,:)*cpts(:,1);
                 ddN(1,:)*cpts(:,2)    ddN(2,:)*cpts(:,2)    ddN(3,:)*cpts(:,2)];

 dxdxi2   =    [dxdxi(1,1)^2    dxdxi(1,1)*dxdxi(1,2)    dxdxi(1,2)^2;
2*dxdxi(1,1)*dxdxi(2,1)    dxdxi(1,1)*dxdxi(2,2)+dxdxi(1,2)*dxdxi(2,1)    2*dxdxi(1,2)*dxdxi(2,2);
                dxdxi(2,1)^2    dxdxi(2,1)*dxdxi(2,2)    dxdxi(2,2)^2];

% 7. Solve for first derivatives in global coordinates (using MATLAB's backslash operator)
dN = dxdxi'\dN;

% 8. Solve for second derivatives in global coordinates (using MATLAB's backslash operator)
ddN = dxdxi2'\(ddN - d2xdxi2'*dN);
```

Algorithm 1:
MATLAB code snippet 1 - Compute NURBS basis functions and their first and second derivatives in global coordinates. This routine is required for the formation of the global rows of a stiffness matrix at each collocation point in IGA collocation.

of displacements and accelerations in IGA-C, so that the number of linear system solves required for the computation of second order derivatives is considerably reduced. To help interested readers to retrace our counts, we additionally provide corresponding MATLAB routines. Algorithm 1 shows the computation of 2D NURBS basis functions and their first and second derivatives for the computation of the stiffness forms. Algorithm 2 shows the evaluation of displacements and accelerations in 2D NURBS based collocation.

For the elastodynamics case (c), we furthermore assume that the Jacobian matrix in IGA-G/FEA-G and the Jacobian and Hessian matrices in IGA-C are precomputed at each quadrature/collocation point. This requires the storage of a maximum of 27 doubles (for the case of a collocation point in 3D), but significantly reduces the computational effort for the formation of the residual vector. We also assume optimized linear algebra routines that avoid operations on zero entries of local matrices in IGA-G and FEA-G. In the tables of Appendix A, $\boldsymbol{B}$ denotes the strain-nodal displacement matrix and $\boldsymbol{H}$ is the matrix that maps nodal degrees of freedom to values at a collocation/quadrature point. In 3D, they

Data: Local coordinates $\{xi, eta\}$ of the current collocation point;

Arrays of coefficients *coefs_u*(:,1), *coefs_u*(:,2) and *coefs_a*(:,1), *coefs_a*(:,2) determining the current displacements and accelerations, respectively;

Weights $w(:,1)$ depending on the NURBS geometry;

Functions $Bspline(*,i)$, $dBspline(*,i)$ and $ddBspline(*,i)$ providing the function values of the i^{th} univariate B-splines and its first/second derivatives in local coordinates;

Result: The first/second derivatives of displacements **u** in global coordinates and accelerations **a**, computed with NURBS basis functions;

```
% 1. Form tensor products
for j=1:(p+1) do
    for i=1:(p+1) do
        B       ((j-1)*(p+1)+i)  =    BSpline(xi,i)  *    BSpline(eta,j);
        dBxi    ((j-1)*(p+1)+i)  =   dBSpline(xi,i)  *    BSpline(eta,j);
        dBeta   ((j-1)*(p+1)+i)  =    BSpline(xi,i)  *   dBSpline(eta,j);
        ddBxi   ((j-1)*(p+1)+i)  =  ddBSpline(xi,i)  *    BSpline(eta,j);
        ddBeta  ((j-1)*(p+1)+i)  =    BSpline(xi,i)  *  ddBSpline(eta,j);
        dBxieta ((j-1)*(p+1)+i)  =   dBSpline(xi,i)  *   dBSpline(eta,j);
    end
end

% 2. Multiply each B-spline function with corresponding weight
N     (1,:)  =      B  .*  w(:,1);
dN    (1,:)  =   dBxi  .*  w(:,1);
dN    (2,:)  =  dBeta  .*  w(:,1);
ddN   (1,:)  =  ddBxi  .*  w(:,1);
ddN   (2,:)  = dBxieta .*  w(:,1);
ddN   (3,:)  =  ddBeta .*  w(:,1);

% 3. Compute sums of weighted B-spline functions
w_sum      = sum(N(1,:));
dw_xi      = sum(dN(1,:));
dw_eta     = sum(dN(2,:));
d2w_xi     = sum(ddN(1,:));
d2w_xieta  = sum(ddN(2,:));
d2w_eta    = sum(ddN(3,:));

% 4. Derivatives of u in local coordinates multiplied with the sums of weights times basis functions
dwu_xi     = dN(1,:)   *  coefs_u(:,1:2);
dwu_eta    = dN(2,:)   *  coefs_u(:,1:2);
d2wu_xi    = ddN(1,:)  *  coefs_u(:,1:2);
d2wu_xieta = ddN(2,:)  *  coefs_u(:,1:2);
d2wu_eta   = ddN(3,:)  *  coefs_u(:,1:2);
```

continued on next page

continued from previous page

```
% 5. Divide to obtain derivatives of u in local coordinates
 du(1,1:2)    =   dwu_xi       /   dw_xi;
 du(2,1:2)    =   dwu_eta      /   dw_eta;
 d2u(1,1:2)   =   d2wu_xi      /   d2w_xi;
 d2u(2,1:2)   =   d2wu_xieta   /   d2w_xieta;
 d2u(3,1:2)   =   d2wu_eta     /   d2w_eta;

% Note: The Jacobian and the Hessian depend only on the initial geometry. We assume that both
% matrices dxdxi and d2xdxi2 are precomputed (see Algorithm 1) and stored at each collocation
% point. This is possible, since the corresponding memory consumption per point is negligible
% (10 doubles in 2D, 27 doubles in 3D).

% 6. Set up the remaining system matrix to determine second order derivatives of u
 dxdxi2   =   [dxdxi(1,1)^2   dxdxi(1,1)*dxdxi(1,2)   dxdxi(1,2)^2;
2*dxdxi(1,1)*dxdxi(2,1)   dxdxi(1,1)*dxdxi(2,2)+dxdxi(1,2)*dxdxi(2,1)   2*dxdxi(1,2)*dxdxi(2,2);
                dxdxi(2,1)^2   dxdxi(2,1)*dxdxi(2,2)   dxdxi(2,2)^2];

% 7. Solve for first derivatives of u in global coordinates (using MATLAB's backslash operator)
du = dxdxi'\du;

% 8. Solve for second derivatives of u in global coordinates (using MATLAB's backslash operator)
d2u = dxdxi2'\(d2u - d2xdxi2'*du);

% 9. Compute accelerations
a(1,:) = N(1,:) * coefs_a(:,1:2) / w_sum ;
```

Algorithm 2:
MATLAB code snippet 2 - Compute displacements and accelerations with 2D NURBS basis functions in the elastodynamics case. This routine is applied at each collocation point and minimizes the cost for the evaluation of second order derivatives.

show the following well-known structure per node A [88]

$$\boldsymbol{B}_A = \begin{bmatrix} N_{A,x} & 0 & 0 \\ 0 & N_{A,y} & 0 \\ 0 & 0 & N_{A,z} \\ 0 & N_{A,z} & N_{A,y} \\ N_{A,z} & 0 & N_{A,x} \\ N_{A,y} & N_{A,x} & 0 \end{bmatrix}, \qquad \boldsymbol{H}_A = \begin{bmatrix} N_A & 0 & 0 \\ 0 & N_A & 0 \\ 0 & 0 & N_A \end{bmatrix} \tag{A.1}$$

Appendix A.1. Flops to evaluate basis functions

d	IGA-C	IGA-G	FEA-G
	The computation of basis function values in parametric directions depends largely on function type and specific implementation. Therefore we neglect its cost, assuming it is small and comparable between methods.		
	1. Form tensor products:		
1	-	-	-
2	$6(p+1)^2$	$3(p+1)^2$	$3(p+1)^2$
3	$20(p+1)^3$	$8(p+1)^3$	$8(p+1)^3$
	2. Multiply each B-spline basis function with corresponding weight:		
1	$3(p+1)$	$2(p+1)$	-
2	$6(p+1)^2$	$3(p+1)^2$	-
3	$10(p+1)^3$	$4(p+1)^3$	-
	3. Compute sums of B-spline basis functions:		
1	$3(p+1)$	$2(p+1)$	-
2	$6(p+1)^2$	$3(p+1)^2$	-
3	$10(p+1)^3$	$4(p+1)^3$	-
	4. Compute NURBS basis functions and its derivatives with respect to local coordinates:		
	Function, first and second derivatives	Function and first derivatives	
1	$21(p+1)$	$6(p+1)$	-
2	$57(p+1)^2$	$11(p+1)^2$	-
3	$109(p+1)^3$	$16(p+1)^3$	-
	5. Compute Jacobian matrix:		
1	$2(p+1)$	$2(p+1)$	$2(p+1)$
2	$8(p+1)^2$	$8(p+1)^2$	$8(p+1)^2$
3	$18(p+1)^3$	$18(p+1)^3$	$18(p+1)^3$
	6. Compute Hessian and the matrix of squared first derivatives required for determining second order derivatives:		
1	$2(p+1)+2$	-	-
2	$12(p+1)^2+13$	-	-
3	$36(p+1)^3+63$	-	-

continued on next page

continued from previous page

	7. Solve for 1st derivatives:		
1	$(p+1)$	$(p+1)$	$(p+1)$
2	$5(p+1)^2+4$	$5(p+1)^2+4$	$5(p+1)^2+4$
3	$12(p+1)^3+20$	$12(p+1)^3+20$	$12(p+1)^3+20$
	8. Compute right hand side vectors and solve for 2nd derivatives:		
1	$3(p+1)$	-	-
2	$24(p+1)^2+20$	-	-
3	$87(p+1)^3+140$	-	-
	Total number of flops:		
1	$35(p+1)+2$	$13(p+1)$	$3(p+1)$
2	$124(p+1)^2+37$	$33(p+1)^2+4$	$16(p+1)^2+4$
3	$302(p+1)^3+223$	$62(p+1)^3+20$	$38(p+1)^3+20$

Appendix A.2. Flops to evaluate the local stiffness matrix in the Laplace problem

d	IGA-C	IGA-G	FEA-G
	1. Flops transferred from Appendix A.1:		
1	$35(p+1)+2$	$13(p+1)$	$3(p+1)$
2	$124(p+1)^2+37$	$33(p+1)^2+4$	$16(p+1)^2+4$
3	$302(p+1)^3+223$	$62(p+1)^3+20$	$38(p+1)^3+20$
	2. Set up local stiffness matrix:		
	No local stiffness matrix required.	Evaluate from right to left: $\boldsymbol{B}^T\boldsymbol{B}$ $\|J\|\omega$ ($\|J\|$ = Jacobian, ω = weight)	
1	-	$(p+1)^2+(p+1)$	$(p+1)^2+(p+1)$
2	-	$3(p+1)^4+2(p+1)^2$	$3(p+1)^4+2(p+1)^2$
3	-	$5(p+1)^6+3(p+1)^3$	$5(p+1)^6+3(p+1)^3$
	3. Evaluate Laplace operator on global level:	3. Add to local element stiffness matrix: (Final assembly to global matrix is neglected.)	
1	-	$(p+1)^2$	$(p+1)^2$
2	$(p+1)^2$	$(p+1)^4$	$(p+1)^4$
3	$2(p+1)^3$	$(p+1)^6$	$(p+1)^6$
	Total operations:		
1	$35(p+1)+2$	$2(p+1)^2+14(p+1)$	$2(p+1)^2+4(p+1)$
2	$125(p+1)^2+37$	$4(p+1)^4+35(p+1)^2+4$	$4(p+1)^4+18(p+1)^2+4$
3	$304(p+1)^3+223$	$6(p+1)^6+65(p+1)^3+20$	$6(p+1)^6+41(p+1)^3+20$

Appendix A.3. Flops to evaluate the local stiffness matrix in elasticity

d	IGA-C	IGA-G	FEA-G
	1. Flops transferred from Appendix A.1:		
1	$35(p+1)+2$	$13(p+1)$	$3(p+1)$
2	$124(p+1)^2+37$	$33(p+1)^2+4$	$16(p+1)^2+4$
3	$302(p+1)^3+223$	$62(p+1)^3+20$	$38(p+1)^3+20$
	2. Set up local stiffness matrix:		
	No local stiffness matrix required.	Evaluate from right to left: $\boldsymbol{B}^T\boldsymbol{D}\,\boldsymbol{B}\;\;|J|\omega$ ($\boldsymbol{B}$ = B-matrix, $\boldsymbol{D}$ = elasticity matrix, $\|J\|$ = Jacobian, ω = Gauss weight)	
1	-	$2(p+1)^2+(p+1)$	$2(p+1)^2+(p+1)$
2	-	$20(p+1)^4+36(p+1)^2$	$20(p+1)^4+36(p+1)^2$
3	-	$99(p+1)^6+216(p+1)^3$	$99(p+1)^6+216(p+1)^3$
	3. Evaluate Navier's eqs. on global level:	3. Add to local element stiffness matrix: (Final assembly to global matrix is neglected.)	
1	$(p+1)$	$(p+1)^2$	$(p+1)^2$
2	$12(p+1)^2$	$4(p+1)^4$	$4(p+1)^4$
3	$21(p+1)^3$	$9(p+1)^6$	$9(p+1)^6$
	Total operations:		
1	$36(p+1)+2$	$3(p+1)^2+14(p+1)$	$3(p+1)^2+4(p+1)$
2	$136(p+1)^2+37$	$24(p+1)^4+69(p+1)^2+4$	$24(p+1)^4+52(p+1)^2+4$
3	$323(p+1)^3+223$	$108(p+1)^6+278(p+1)^3+20$	$108(p+1)^6+254(p+1)^3+20$

Appendix A.4. Flops to evaluate the local residual vector in elastodynamics

d	IGA-C	IGA-G	FEA-G
	1a. Flops transferred from Appendix A.1: Assume that the Jacobian and the Hessian in IGA-C and the Jacobian in IGA-G/FEA-G are precomputed at each collocation/quadrature point.		
	Only steps 1. to 3. of Appendix A.1	Only steps 1. to 4. and 7. of Appendix A.1	
1	$6(p+1)$	$11(p+1)$	$(p+1)$
2	$18(p+1)^2$	$25(p+1)^2+4$	$8(p+1)^2+4$
3	$40(p+1)^3$	$44(p+1)^3+20$	$20(p+1)^3+20$
	1b. Compute first/second derivatives of $w\boldsymbol{u}$ w.r.t. local coordinates:		
1	$4(p+1)$	-	-
2	$20(p+1)^2$	-	-
3	$54(p+1)^3$	-	-
	1c. Compute first/second derivatives of $\boldsymbol{u}$ w.r.t. local coordinates and the matrix of squared first derivatives:		
1	$2+1=3$	-	-
2	$10+13=23$	-	-
3	$27+63=90$	-	-
	1d. Compute first/second derivatives of $\boldsymbol{u}$ w.r.t. global coordinates: Solve systems of eqs. according to steps 7. and 8. of Appendix A.1.		
1	$1+3=4$	-	-
2	$14+64=78$	-	-
3	$52+401=453$	-	-

continued on next page

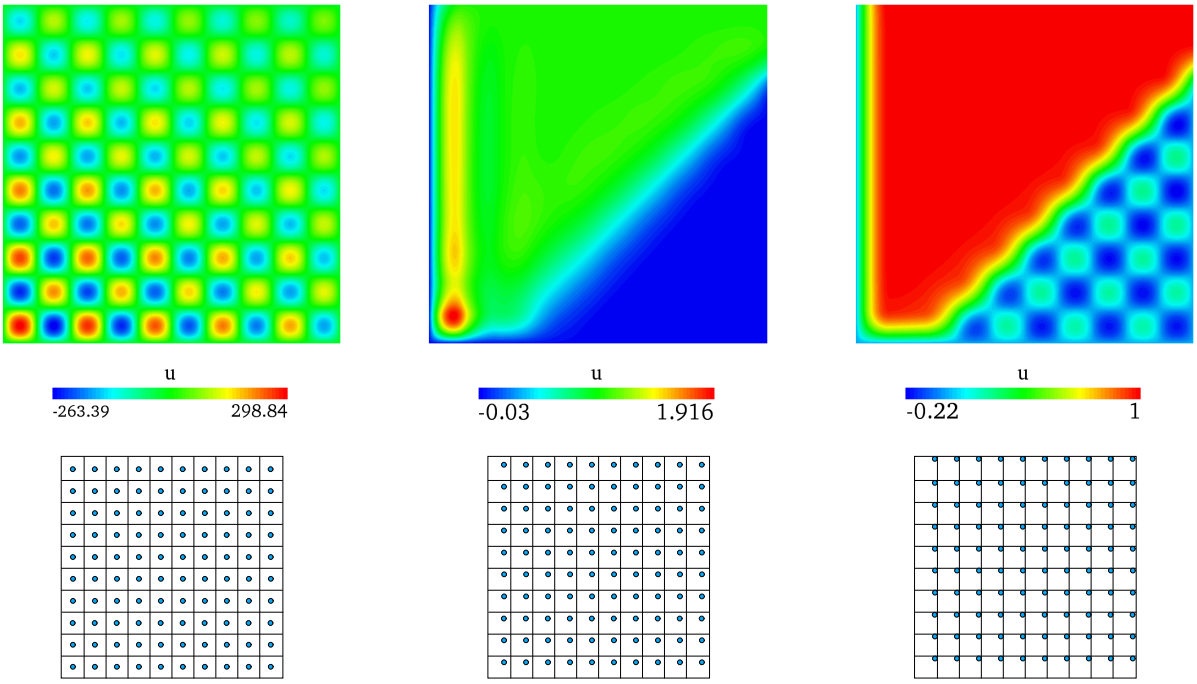
continued from previous page

	2. Evaluate external force and add to global residual:	2. Evaluate external force, add to element residual: From right to left: $\boldsymbol{H}^T \boldsymbol{f} \ \ \lvert J\rvert\omega$ ($\boldsymbol{f}$ = force vector, $\lvert J\rvert$ = Jacobian, ω = Gauss weight)	
1	1	$2(p+1)+2$	$2(p+1)+2$
2	2	$4(p+1)^2+3$	$4(p+1)^2+3$
3	3	$6(p+1)^3+4$	$6(p+1)^3+4$
	3. Evaluate Navier's eqs. with second derivatives of $\boldsymbol{u}$ and add to global residual:	3. Evaluate internal force, add to element residual: From right to left: $\boldsymbol{B}^T \boldsymbol{D}\ \boldsymbol{B}\ \boldsymbol{c} \ \ \lvert J\rvert\omega$ ($\boldsymbol{D}$ = elasticity matrix, $\boldsymbol{c}$ = local displ. vector)	
1	2	$5(p+1)+2$	$5(p+1)+2$
2	12	$18(p+1)^2+8$	$18(p+1)^2+8$
3	21	$39(p+1)^3+19$	$39(p+1)^3+19$
	4. Compute acceleration $\boldsymbol{a}$ and add $\rho\boldsymbol{a}$ to global residual vector:	4. Evaluate inertial force, add to element residual: From right to left: $\boldsymbol{H}^T \boldsymbol{H}\ \boldsymbol{k} \ \ \rho\lvert J\rvert\omega$ ($\boldsymbol{k}$ = local acceleration vector, ρ = density)	
1	$2(p+1)+2$	$5(p+1)+2$	$5(p+1)+2$
2	$4(p+1)^2+4$	$10(p+1)^2+2$	$10(p+1)^2+2$
3	$6(p+1)^3+6$	$15(p+1)^3+2$	$15(p+1)^3+2$
	No assembly required	Note: Cost of assembly of element residual to global residual is neglected.	
	Total operations:		
1	$12(p+1)+12$	$23(p+1)+6$	$13(p+1)+6$
2	$42(p+1)^2+119$	$57(p+1)^2+17$	$40(p+1)^2+17$
3	$100(p+1)^3+573$	$104(p+1)^3+45$	$80(p+1)^3+45$

Appendix B. Point upwinding in isogeometric collocation

For advection dominated operators, central difference type equations as they typically arise from Galerkin discretizations have long been known to lead to solutions that are corrupted by spurious oscillations [120–122]. It is natural that this also holds for IGA collocation, since the collocation points based on the Greville abscissae are located central to the corresponding B-spline basis functions. A common way to deal with this problem is the concept of upwinding [120–122]. In the context of IGA collocation, we can achieve an upwinding effect by simply shifting the collocation points upstream in the direction of the velocity. From a variational point of view, this corresponds to Dirac δ functions in the weighted residual form (see Section 2.2) that are shifted upstream. Since the modified Dirac δ functions are applied to all terms in the PDE, this formulation can be considered a consistent Petrov-Galerkin weighted residual method. In principle, this simple procedure is equivalent to shifting quadrature points upstream in a Galerkin context [126, 127] and has been successfully adapted to other collocation schemes based on different sets of collocation points (see for example [128–130]).

The validity and efficiency of point upwinding for IGA collocation is briefly illustrated by some results for the 2D advection dominated benchmark of Fig. 56, discretized with a 10×10 mesh of quadratic B-splines. Figure A.67a shows the unstable oscillatory solution resulting from collocation at the standard Greville abscissae. Figures A.67b and A.67c show the improvement of the solution for moderate and strong shifting of collocation points upstream. Although the general solution behavior is considerably improved by upwinding, it also entails spurious oscillations that can be observed at the lower-right part of the domain. Whereas the solution is completely free of oscillations and almost hitting the exact value of zero everywhere for moderate upwinding (see Fig. A.67b), a checker-board pattern appears for strong upwinding of collocation points (see Fig. A.67c). Note that the oscillations and moderate accuracy for values close to $u = 0$ in the plots of Figs. 57 and 58 are due to this phenomenon. For more complex problems with varying Péclet numbers, it would be useful to determine an explicit relation depending on the local Péclet number that provides the optimal distance a point needs to be shifted as a compromise between the beneficial effect of upwinding and the oscillatory side-effects of Fig. A.67c. Beyond the idea of collocation-point upwinding, we believe that it would be beneficial to investigate the development of a consistent stabilized IGA collocation technology that achieves analogous advantages as the SUPG method [122] for Galerkin based discretization methods.



(a) *No upwinding: Collocation points are located central to basis functions.*

(b) *Moderate upwinding: Shifted upstream by 1/4 of the knot span diagonal.*

(c) *Strong upwinding: Shifted upstream by almost 1/2 of the diagonal.*

Figure B.67: *2D benchmark of Fig. 56: The effect of point upwinding in single-level standard IGA collocation using quadratic B-splines.*

References

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

[2] J.A. Cottrell, T.J.R. Hughes, and Y. Bazilevs. *Isogeometric analysis: Towards Integration of CAD and FEA*. John Wiley & Sons, 2009.

[3] P. Kagan and A. Fischer. Integrated mechanically based CAE system using B-spline finite elements. *Computer Aided Design*, 32(8-9):539–552, 2000.

[4] R. Schmidt, J. Kiendl, K.U. Bletzinger, and R. Wüchner. Realization of an integrated structural design process: analysis-suitable geometric modelling and isogeometric analysis. *Computing and Visualization in Science*, 13(7):315–330, 2009.

[5] E. Cohen, T. Martin, R.M. Kirby, T. Lyche, and R.F. Riesenfeld. Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 199:334–356, 2010.

[6] J.A. Evans, Y. Bazilevs, I. Babuška, and T.J.R. Hughes. n-widths, sup-infs, and optimality ratios for the k-version of the isogeometric finite element method. *Computer Methods in Applied Mechanics and*

Engineering, 198(21–26):1726–1741, 2009.

[7] D. Großmann, B. Jüttler, H. Schlusnus, J. Barner, and A.H. Vuong. Isogeometric simulation of turbine blades for aircraft engines. *Computer Aided Geometric Design*, 29(7):519–531, 2012.

[8] J.A. Cottrell, A. Reali, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of structural vibrations. *Computer Methods in Applied Mechanics and Engineering*, 195:5257–5296, 2006.

[9] T. J. R. Hughes, A. Reali, and G. Sangalli. Duality and unified analysis of discrete approximations in structural dynamics and wave propagation: Comparison of p-method finite elements with k-method NURBS. *Computer Methods in Applied Mechanics and Engineering*, 197:4104–4124, 2008.

[10] T. Elguedj, Y. Bazilevs, V.M. Calo, and T.J.R. Hughes. $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticity using higher-order NURBS elements. *Computer Methods in Applied Mechanics and Engineering*, 197:2732–2762, 2008.

[11] R.L. Taylor. Isogeometric analysis of nearly incompressible solids. *International Journal for Numerical Methods in Engineering*, 87(1-5):273–288, 2011.

[12] J. Kiendl, K.U. Bletzinger, J. Linhard, and R. Wüchner. Isogeometric shell analysis with Kirchhoff-Love elements. *Computer Methods in Applied Mechanics and Engineering*, 198(49-52):3902–3914, 2009.

[13] D.J. Benson, Y. Bazilevs, M.C. Hsu, and Hughes T.J.R. Isogeometric shell analysis: The Reissner-Mindlin shell. *Computer Methods in Applied Mechanics and Engineering*, 199:276–289, 2010.

[14] R. Echter and M. Bischoff. Numerical efficiency, locking and unlocking of NURBS finite elements. *Computer Methods in Applied Mechanics and Engineering*, 199:374–382, 2010.

[15] D.J. Benson, S. Hartmann, Y. Bazilevs, M.C. Hsu, and T.J.R. Hughes. Blended Isogeometric Shells. *Computer Methods in Applied Mechanics and Engineering*, 255:133–146, 2013.

[16] Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and Y. Zhang. Isogeometric fluid-structure interaction: Theory, algorithms and computations. *Computational Mechanics*, 43:3–37, 2008.

[17] Y. Bazilevs, M.C. Hsu, J. Kiendl, R. Wüchner, and K.U. Bletzinger. 3D simulation of wind turbine rotors at full scale. Part II: Fluid-structure interaction. *International Journal of Numerical Methods in Fluids*, 65:236–253, 2011.

[18] Y. Bazilevs, M.C. Hsu, and M.A. Scott. Isogeometric fluid-structure interaction analysis with emphasis on non-matching discretizations, and with application to wind turbines. *Computer Methods in Applied Mechanics and Engineering*, 249–252:28–41, 2012.

[19] Y. Bazilevs, V.M. Calo, J. A. Cottrell, T.J.R. Hughes, A. Reali, and G. Scovazzi. Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows. *Computer Methods in Applied Mechanics and Engineering*, 197:173–201, 2007.

[20] I. Akkerman, Y. Bazilevs, V.M. Calo, T.J.R. Hughes, and S. Hulshoff. The role of continuity in residual-based variational multiscale modeling of turbulence. *Computational Mechanics*, 41:371–378, 2008.

[21] H. Gomez, V.M. Calo, Y. Bazilevs, and T.J.R. Hughes. Isogeometric analysis of the Cahn-Hilliard phase-field model. *Computer Methods in Applied Mechanics and Engineering*, 197:4333–4352, 2008.

[22] M.J. Borden, C.V. Verhoosel, M.A. Scott, Hughes T.J.R., and C.M. Landis. A phase-field description of dynamic brittle fracture. *Computer Methods in Applied Mechanics and Engineering*, 217–220:77–95, 2012.

[23] L. Dede', M.J. Borden, and T.J.R. Hughes. Isogeometric analysis for topology optimization with a phase field model. *Archives of Computational Methods in Engineering*, 19:427–465, 2012.

[24] J. Liu, L. Dede, J.A. Evans, M.J. Borden, and T.J.R. Hughes. Isogeometric analysis of the advective Cahn-Hilliard equation: Spinodal decomposition under shear flow. *ICES report 12-12*, The University of Texas at Austin, 2012.

[25] I. Temizer, P. Wriggers, and T.J.R. Hughes. Contact treatment in isogeometric analysis with NURBS. *Computer Methods in Applied Mechanics and Engineering*, 200:1100–1112, 2011.

[26] I. Temizer, P. Wriggers, and T.J.R. Hughes. Three-dimensional mortar-based frictional contact treatment in isogeometric analysis with NURBS. *Computer Methods in Applied Mechanics and Engineering*, 209–212:115–128, 2012.

[27] L. De Lorenzis, P. Wriggers, and G. Zavarise. A mortar formulation for 3D large deformation contact using NURBS-based isogeometric analysis and the augmented Lagrangian method. *Computational Mechanics*, 49(1):1–20, 2012.

[28] W.A. Wall, M.A. Frenzel, and C. Cyron. Isogeometric structural shape optimization. *Computer Methods in Applied Mechanics and Engineering*, 197:2976–2988, 2008.

[29] X. Qian and O. Sigmund. Isogeometric shape optimization of photonic crystals via coons patches. *Computer Methods in Applied Mechanics and Engineering*, 200:2237–2255, 2011.

[30] H.-J. Kim, Y.-D. Seo, and S.-K. Youn. Isogeometric analysis for trimmed CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 198:2982–2995, 2009.

[31] T. Rueberg and F. Cirak. Subdivision-stabilised immersed B-spline finite elements for moving boundary flows. *Computer Methods in Applied Mechanics and Engineering*, 209–212:266–283, 2012.

[32] D. Schillinger, M. Ruess, N. Zander, Y. Bazilevs, A. Düster, and E. Rank. Small and large deformation analysis with the p- and B-spline versions of the Finite Cell Method. *Computational Mechanics*, 50(4), 2012.

[33] E. Rank, M. Ruess, S. Kollmannsberger, D. Schillinger, and A. Düster. Geometric modeling, isogeometric analysis and the finite cell method. *Computer Methods in Applied Mechanics and Engineering*, 249-250:104–115, 2012.

[34] M. Ruess, D. Schillinger, Y. Bazilevs, V. Varduhn, and E. Rank. Weakly enforced essential boundary conditions for NURBS-embedded and trimmed NURBS geometries on the basis of the finite cell method. *International Journal for Numerical Methods in Engineering*, Submitted, 2013.

[35] R.N. Simpson, S.P.A. Bordas, J. Trevelyan, and T. Rabczuk. A two-dimensional isogeometric boundary element method for elastostatic analysis. *Computer Methods in Applied Mechanics and Engineering*, 209–212:87–100, 2012.

[36] M.A. Scott, R.N. Simpson, J.A. Evans, S. Lipton, S.P.A. Bordas, T.J.R. Hughes, and T.W. Sederberg. Isogeometric boundary element analysis using unstructured T-splines. *ICES report 12-23*, Univ. of Texas at Austin, 2012.

[37] M.J. Borden, M.A. Scott, J.A. Evans, and T.J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of NURBS. *International Journal for Numerical Methods in Engineering*, 87:15–47, 2011.

[38] M.A. Scott, M.J. Borden, C.V. Verhoosel, T.W. Sederberg, and T.J.R. Hughes. Isogeometric finite element data structures based on Bézier extraction of T-splines. *International Journal for Numerical*

Methods in Engineering, 88:126–156, 2011.

[39] M.A. Scott, X. Li, T.W. Sederberg, and T.J.R. Hughes. Local refinement of analysis-suitable T-splines. *Computer Methods in Applied Mechanics and Engineering*, 213–216:206–222, 2012.

[40] N. Collier, D. Pardo, L. Dalcin, M. Paszynski, and V.M. Calo. The cost of continuity: a study of the performance of isogeometric finite elements using direct solvers. *Computer Methods in Applied Mechanics and Engineering*, 213–216:353–361, 2012.

[41] N. Collier, D. Pardo, L. Dalcin, and V.M. Calo. The cost of continuity: performance of iterative solvers on isogeometric finite elements. *eprint arXiv:1206.2948*, 2012.

[42] S.K. Kleiss, C. Pechstein, B. Jüttler, and S. Tomar. IETI - Isogeometric Tearing and Interconnecting. *Computer Methods in Applied Mechanics and Engineering*, 247–248, 2012.

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

[44] F. Auricchio, F. Calabrò, T.J.R. 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*, 249–252:15–27, 2012.

[45] F. Auricchio, L. Beirao da Veiga, T.J.R. Hughes, A. Reali, and G. Sangalli. Isogeometric collocation methods. *Mathematical Models and Methods in Applied Sciences*, 20(11):2075–1077, 2010.

[46] F. Auricchio, L. Beirao da Veiga, T.J.R. Hughes, A. Reali, and G. Sangalli. Isogeometric collocation for elastostatics and explicit dynamics. *Computer Methods in Applied Mechanics and Engineering*, 249–252:2–14, 2012.

[47] M. Bischoff, W.A. Wall, K.-U. Bletzinger, and E. Ramm. Models and Finite Elements for Thin-walled Structures. In E. Stein, R. de Borst, and T.J.R. Hughes, editors, *Encyclopedia of Computational Mechanics*, volume 2, chapter 3, pages 59–137. John Wiley & Sons, 2004.

[48] L. Beirao da Veiga, C. Lovadina, and A. Reali. Avoiding shear locking for the timoshenko beam problem via isogeometric collocation methods. *Computer Methods in Applied Mechanics and Engineering*, 241–244:38–51, 2012.

[49] A.G. Kravchenko, P. Moin, and R.D. Moser. Zonal embedded grids for numerical simulation of wall-bounded turbulent flows. *Journal of Computational Physics*, 127:412–423, 1996.

[50] K. Shariff and R.D. Moser. Two-dimensional mesh embedding for B-spline methods. *Journal of Computational Physics*, 145:471–488, 1998.

[51] Y. Bazilevs and I. Akkerman. Large eddy simulation of turbulent Taylor-Couette flow using isogeometric analysis and residual-based variational multiscale method. *Journal of Computational Physics*, 229:3402–3414, 2010.

[52] Y. Bazilevs, C.M. Michler, V.M. Calo, and T.J.R. Hughes. Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly-enforced boundary conditions on unstretched meshes. *Computer Methods in Applied Mechanics and Engineering*, 199:780–790, 2010.

[53] A.G. Kravchenko, P. Moin, and K. Shariff. B-spline method and zonal grids for simulation of complex turbulent flows. *Journal of Computational Physics*, 151:757–789, 1999.

[54] J.A. Evans and T.J.R. Hughes. Isogeometric divergence-conforming B-splines for the steady Navier-Stokes equations. *Mathematical Models and Methods in Applied Sciences*, doi: 10.1142/S0218202513500139, 2012.

[55] J.A. Evans and T.J.R. Hughes. Isogeometric divergence-conforming B-splines for the unsteady Navier-Stokes equations. *Journal of Computational Physics*, doi: 10.1016/j.jcp.2013.01.006, 2013.

[56] B. Bialecki and G. Fairweather. Orthogonal spline collocation methods for partial differential equations. *Journal of Computational and Applied Mathematics*, 128:55–82, 2001.

[57] O. Botella. On a collocation B-spline method for the solution of the Navier-Stokes equations. *Computers & Fluids*, 31:397–420, 2002.

[58] O. Botella. A high-order mass-lumping procedure for B-spline collocation method with application to incompressible flow simulations. *International Journal for Numerical Methods in Fluids*, 41:1295–1318, 2003.

[59] R.W. Johnson. A B-spline collocation method for solving the incompressible Navier-Stokes equations using an ad hoc method: the boundary residual method. *Computers & Fluids*, 34:121–149, 2005.

[60] R.W. Johnson. Higher order B-spline collocation at the Greville abscissae. *Applied Numerical Mathematics*, 52:63–75, 2005.

[61] D. Schillinger and E. Rank. An unfitted *hp* adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry. *Computer Methods in Applied Mechanics and Engineering*, 200(47–48):3358–3380, 2011.

[62] 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(49–52):3554–3567, 2011.

[63] D. Schillinger, L. Dede', M.A. Scott, J.A. Evans, M.J. Borden, E. Rank, and T.J.R. Hughes. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering*, 249-250:116–150, 2012.

[64] B. Bornemann and F. Cirak. A subdivision-based implementation of the hierarchical b-spline finite element method. *Computer Methods in Applied Mechanics and Engineering*, accepted for publication, 2012.

[65] D. Zorin, P. Schröder, T. DeRose, L. Kobbelt, A. Levin, and W. Sweldens. Subdivision for modeling and animation. Technical report, 2000.

[66] J. Warren and H. Weimer. *Subdivision Methods for Geometric Design*. Morgan Kaufman Publishers, 2002.

[67] H. Samet. *Foundations of Multidimensional and Metric Data Structures*. Morgan Kaufmann Publishers, 2006.

[68] C. Burstedde, L.C. Wilcox, and O. Ghattas. p4est: Scalable algorithms for parallel adaptive mesh refinement on forests of octrees. *SIAM Journal on Scientific Computing*, 33(3):1103–1133, 2011.

[69] W. Bangerth, C. Burstedde, T. Heister, and M. Kronbichler. Algorithms and data structures for massively parallel generic adaptive finite element codes. *ACM Transactions on Mathematical Software*, 38(2):14, 2011.

[70] C. Giannelli, B. Jüttler, and H. Speleers. THB-splines: The truncated basis for hierarchical splines. *Computer Aided Geometric Design*, 29(7):485–498, 2012.

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

[72] E. Cohen, R.F. Riesenfeld, and G. Elber. *Geometric Modeling with Splines: An Introduction*. A K

Peters/CRC Press, 2001.

[73] D.F. Rogers. *An Introduction to NURBS with Historical Perspective*. Morgan Kaufmann Publishers, 2001.

[74] G. Farin. *Curves and surfaces for computer aided geometric design*. Morgan Kaufmann Publishers, 2002.

[75] *Rhinoceros — NURBS modeling for Windows*. http://www.rhino3d.com/, 2012.

[76] B.A. Finlayson and L.E. Scriven. The method of weighted residuals - a review. *Applied Mechanics Reviews*, 19:735–748, 1966.

[77] B.A. Finlayson. *The Method of Weighted Residuals and Variational Principles*. Academic Press, 1972.

[78] G.F. Pinder and A.M. Shapiro. A new collocation method for the solution of the convection-dominated transport equation. *Water resources research*, 15:1177–1182, 1979.

[79] G.F. Carey and J.T. Oden. *Finite Elements: a Second Course*. Prentice-Hall, 1983.

[80] J.S. Chen, L. Wang, H.-Y. Hu, and S.-W. Chi. Subdomain radial basis collocation method for heterogeneous media. *International Journal for Numerical Methods in Engineering*, 80:163–190, 2009.

[81] H.-Y. Hu, J.S. Chen, and W. Hu. Weighted radial basis collocation method for boundary value problems. *International Journal for Numerical Methods in Engineering*, 69:2736–2757, 2007.

[82] J.S. Chen, W. Hu, and H.-Y. Hu. Reproducing kernel enhanced local radial basis collocation method. *International Journal for Numerical Methods in Engineering*, 75:600–627, 2008.

[83] H.-Y. Hu, J.S. Chen, and W. Hu. Error analysis of collocation method based on reproducing kernel approximation. *Numerical Methods for Partial Differential Equations*, 27:554–580, 2011.

[84] N.R. Aluru. A point collocation method based on reproducing kernel approximations. *International Journal for Numerical Methods in Engineering*, 47:1083–1121, 2000.

[85] D.W. Kim and W.K. Liu. Maximum principle and convergence analysis for the meshfree point collocation method. *SIAM Journal on Numerical Analysis*, 44(2):515–539, 2006.

[86] R.D. Russell and J.M. Varah. A comparison of global methods for linear two-point boundary value problems. *Mathematics of Computation*, 29:1007–1019, 1975.

[87] P.M. Prenter. *Splines and Variational Methods*. Wiley, 1989.

[88] T.J.R. Hughes. *The Finite Element Method: Linear Static and Dynamic Finite Element Analysis*. Dover Publications, 2000.

[89] C. de Boor and B. Swartz. Collocation at Gaussian points. *SIAM Journal on Numerical Analysis*, 10:582–606, 1973.

[90] S. Jator and Z. Sinkala. A high oder B-spline collocation method for linear boundary value problems. *Applied Mathematics and Computation*, 191:100–116, 2007.

[91] S. Demko. On the existence of interpolation projectors onto spline spaces. *Journal of Approximation Theory*, 43:151–156, 1985.

[92] R.T. Farouki. The Bernstein polynomial basis: A centennial retrospective. *Computer Aided Geometric Design*, 29:379–419, 2012.

[93] B. Szabó and I. Babuška. *Finite Element Analysis*. Wiley, 1991.

[94] Livermore Software Technology Corporation. *LS-Dyna 971 R5 user's manual*.

[95] J.A. Evans and T.J.R. Hughes. Discrete spectrum analyses for various mixed discretizations of the Stokes eigenproblem. *Computational Mechanics*, 50(6), 2012.

[96] G.H. Golub and C. Van Loan. *Matrix Computations*. Johns Hopkins University Press, 1996.

[97] A. Quarteroni and A. Valli. *Numerical Approximation of Partial Differential Equations*. Springer, 2008.

[98] C.C. Christara and K.R. Jackson. Numerical methods. In G.L. Trigg, editor, *Mathematical Tools for Physicists*. Wiley, 2005.

[99] C. Ashcraft and R. Grimes. Personal communication.

[100] M.H. Sadd. *Elasticity. Theory, Applications, and Numerics*. Academic Press, 2009.

[101] Trilinos Version 11.0, Sandia National Laboratories, http://trilinos.sandia.gov, 2012.

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

[103] Y. Bazilevs, L.B. Da Veiga, J.A. Cottrell, T.J.R. Hughes, and G. Sangalli. Isogeometric analysis: Approximation, stability and error estimates for h-refined meshes. *Mathematical Models and Methods in Applied Sciences*, 16(7):1031–1090, 2006.

[104] D.N. Arnold and W. Wendland. On the asymptotic convergence of collocation methods. *Mathematics of Computation*, 41:349–381, 1983.

[105] D.N. Arnold and J. Saranen. On the asymptotic convergence of spline collocation methods for partial differential equations. *SIAM Journal on Numerical Analysis*, 21:459–472, 1984.

[106] C.K. Chui. *An introduction to wavelets*. Academic Press, 1992.

[107] J. Peters and U. Reif. *Subdivision surfaces*. Springer, 2008.

[108] M. Sabin. *Analysis and Design of Univariate Subdivision Schemes*. Springer, 2010.

[109] D. Schillinger, S. Kollmannsberger, R.-P. Mundani, and E. Rank. The finite cell method for geometrically nonlinear problems of solid mechanics. *IOP Conference Series: Material Science and Engineering*, 10:012170, 2010.

[110] E. Rank. Adaptive remeshing and h-p domain decomposition. *Computer Methods in Applied Mechanics and Engineering*, 101:299–313, 1992.

[111] D. Schillinger, A. Düster, and E. Rank. The hp-d adaptive finite cell method for geometrically nonlinear problems of solid mechanics. *International Journal for Numerical Methods in Engineering*, 89:1171–1202, 2012.

[112] D. Forsey and R.H. Bartels. Hierarchical B-spline refinement. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22(4):205–212, 1988.

[113] R. Kraft. Adaptive and linearly independent multilevel B-splines. In A.L. Méhauté, C. Rabut, and L.L. Schumaker, editors, *Surface Fitting and Multiresolution Methods*, pages 209–218. Vanderbilt University Press, 1997.

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

[115] H. Yserantant. On the multi-level splitting of finite element spaces. *Numerische Mathematik*, 49:379–412, 1986.

[116] P. Krysl, E. Grinspun, and P. Schröder. Natural hierarchical refinement for finite element methods. *International Journal for Numerical Methods in Engineering*, 56:1109–1124, 2003.

[117] S. Govindjee, J. Strain, T.J. Mitchell, and R. L. Taylor. Convergence of an efficient local least-squares fitting method for bases with compact support. *Computer Methods in Applied Mechanics and Engineering*, 213–216:84–92, 2012.

[118] W. Chen, Y. Cai, and J. Zheng. Generalized hierarchical NURBS for interactive shape modification. In *Proc. of the 7th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applations in Industry*, 2008.

[119] W. Chen, Y. Cai, and J. Zheng. Freeform-based form feature modeling using a hierarchical & multi-resolution NURBS method. In *Proc. of the 9th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applations in Industry*, 2010.

[120] O.C. Zienkiewicz and R.L. Taylor. *The Finite Element Method – Fluid Dynamics*, volume 3. Butterworth-Heinemann, 6th edition, 2005.

[121] J. Donea and A. Huerta. *Finite Element Methods for Flow Problems*. Wiley, 2003.

[122] A.N. Brooks and T.J.R. Hughes. Streamline upwind/Petrov-Galerkin formulations for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 32:199–259, 1982.

[123] Y. Bazilevs and T.J.R. Hughes. Weak imposition of dirichlet boundary conditions in fluid mechanics. *Computers & Fluids*, 36:12–26, 2007.

[124] Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, and T.W. Sederberg. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199:229–263, 2010.

[125] M.R. Dörfel, B. Simeon, and B. Jüttler. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199:264–275, 2010.

[126] T.J.R. Hughes. A simple scheme for developing upwind finite elements. *International Journal for Numerical Methods in Engineering*, 12:1359–1365, 1978.

[127] T.J.R. Hughes, W.K. Liu, and A. Brooks. Finite element analysis of incompressible viscous flows by the penalty function formulation. *Journal of Computational Physics*, 30:1–60, 1979.

[128] A.M. Shapiro and G.F. Pinder. Analysis of an upstream weighted collocation approximation to the transport equation. *Journal of Computational Physics*, 39:46–71, 1981.

[129] S. Adjerid, M. Aiffa, and J.E. Flaherty. Computational methods for singularly perturbed systems. In *AMS Proceedings of Symposia in Applied Mathematics*, pages 47–83. American Mathematical Society, 1998.

[130] D. Funaro and G. Pontrelli. Spline approximation of advection-diffusion problems using upwind type collocation nodes. *Journal of Computational and Applied Mathematics*, 110:141–153, 1998.