Weighted Isogeometric Collocation based on Spline Projectors

Alessandro Giust[a,*], Bert Jüttler[b]

[a]*Doctoral Program Computational Mathematics, Johannes Kepler University, Linz, Austria*
[b]*Institute of Applied Geometry, Johannes Kepler University, Linz, Austria*

Abstract

In this paper, we propose a novel version of weighted isogeometric collocation that is especially suited for adaptive THB-spline refinement.

It is well known that the choice of the collocation nodes is crucial to ensure stability and good approximation properties, especially when adaptive refinement is performed. In order to address this issue, we make use of a particular class of locally supported quasi-interpolant schemes to propose the new method of Weighted Isogeometric Collocation based on Spline Projectors (WICSP).

We show that WICSP performs well in the case of tensor-product spline discretizations, both with respect to the rate of convergence and computational complexity. In particular, we observe experimentally an optimal rate of convergence for odd degree basis functions and obtain a dimension-independent computational complexity $\mathcal{O}(np)$ for matrix-free applications, similar to other approaches such as weighted quadrature [8] and clustered collocation [30].

We explore how these results extend to the case of adaptively refined THB-spline discretizations. We observe that WICSP is compatible with THB-spline refinement, exhibiting good accuracy and low computational costs. In fact, we get a complexity of $\mathcal{O}(np^{2d})$ for matrix assembly and $\mathcal{O}(np^d)$ for the matrix-free case. This compares well with the available methods for isogeometric THB-spline discretizations.

Keywords: isogeometric analysis, collocation, THB-splines, quasi-interpolant, spline projector

1. Introduction

Within the framework of Isogeometric Analysis [20], collocation is a particularly simple and efficient method to solve PDEs on complex domains. In contrast to Galerkin–based approaches, which transform the original problem to its weak formulation, it directly solves the strong form of the equation via interpolation at prescribed collocation nodes. This eliminates the need of performing numerical integration, but requires discretization spaces with higher regularity, which are naturally provided by the isogeometric framework. Indeed, spline basis functions are a natural choice for this type of discretization, since their smoothness can be easily tuned according to the specific differential operator, thus allowing to solve higher order equations by simply controlling the regularity of the bases.

Isogeometric collocation has been an active research topic especially because of its computational efficiency. Numerical quadrature in Galerkin–based IGA entails high computational costs of the matrix assembly routines, and – although remarkable progress has been made in the last years (see [8, 21, 32] and the references cited therein) – it still constitutes a relevant problem. Isogeometric collocation allows for a much faster assembly of the system matrices, especially if basis functions of high degree are used.

*Corresponding author
Email addresses: alessandro.giust@dk-compmath.jku.at (Alessandro Giust), bert.juettler@jku.at (Bert Jüttler)

Despite its evident efficiency, the approximation power of isogeometric collocation is still not completely understood. It has been observed that the choice of the collocation nodes affects both the accuracy and the stability of the method. Typical choices of the collocation nodes include the use of the Greville abscissas of spline bases [3], and of Demko abscissas [10]. However, already in the case of second order elliptic PDEs, it has been noted that these standard choices lead to sub-optimal approximation properties. In fact, the L^2 error converges as $\mathcal{O}(h^{p-1})$ for odd and as $\mathcal{O}(h^p)$ for even degree.

More recently, the adoption of superconvergent points as collocation nodes turned out to be a game-changer concerning the approximation properties of isogeometric collocation. These points constitute an approximation of the Cauchy-Galerkin points (i.e., the points where the Galerkin residual is zero), and their estimated location can be computed numerically through superconvergence theory [18]. Since the number of superconvergent points exceeds the total number of degrees of freedom, a least-square approach has been adopted in [1] to determine the approximate solution. This choice leads to optimal approximation properties for odd degree basis functions but increases the computational costs.

Montardini et al. [30] present a selection strategy in order to determine a suitable subset of the superconvergent points that matches the dimension of the discretization space. Again, optimal approximation power for odd degree basis functions was observed experimentally, but now with lower computational costs. In [42], the authors show how to construct a suitable basis transformation that allows to recover superconvergence properties using the Greville points at the price of introducing additional computational costs.

Due to its simplicity and computational efficiency, isogeometric collocation has been applied successfully to a wide variety of practical problems, ranging from elastostatics [4], shell problems and beams [34] to phase-field modeling [19]. Moreover, the use of non-standard spline basis functions has been investigated [27], and tailored solvers have been developed [11]. Finally, an extension to multi-patch spline discretizations was recently proposed in [24].

In order to deal with complex problems, it has been observed that constructions for discretization spaces that support local refinement are crucial for efficient and accurate numerical simulations. In the context of isogeometric analysis, the developments focused initially on the use of T-splines [5, 12] and were later extended to hierarchical B-splines [37, 41]. Further research was devoted to the application of PHT-splines [31, 43] and LR-splines [23], while the use of truncated hierarchical (TH) B-splines was studied in [14].

While the use of splines that support adaptive refinement in Galerkin-based isogeometric analysis is relatively well understood, only rather few results exist for the case of isogeometric collocation. In their seminal paper about collocation methods, Schillinger et al. [38] introduced the weighted isogeometric collocation approach and showed it to be suitable for the case of adaptive hierarchical NURBS discretizations. This technique permits to avoid problems of linear dependence or instability by considering weighted sums of point evaluations. The collocation nodes are derived by combining Greville abscissas of different levels, and the weights are found based on the B-spline refinement coefficients. The use of T-splines was studied in [9] and was employed to solve second- and fourth-order boundary-value problems. In [22] the authors used PHT-splines to perform adaptive numerical solutions introducing an hybrid Galerkin-collocation method able to cope with multi-patch structures.

Similar to the case of Galerkin-based isogeometric analysis, the use of adaptive spline refinement possesses great potential for improving the efficiency of collocation-based numerical simulation. Further progress can be expected with the help of new approaches for the selection of the collocation nodes (and weights in the case of weighted collocation).

We will focus on THB-splines, since they possess several advantageous properties that make them well suited for geometric modeling and simulation. In particular, they possess the partition of unity property, which is not provided by the standard hierarchical construction. Furthermore, they exhibit strong stability [16], algebraic completeness [29] and — compared to standard hierarchical B-splines – better conditioning and higher sparsity of the resulting system matrices.

For THB-spline-based isogeometric discretizations, we make use of a certain class of locally supported quasi-interpolation (QI) operators in order to establish the new Weighted Isogeometric

Collocation method based on Spline Projectors (WICSP). As indicated by this name, our approach defines the set of collocation nodes and weights through QI schemes that possess the additional property of reproducing spline functions.

The use of these QI operators allows to obtain a unifying weighted collocation framework that encompasses both THB-splines and tensor-product splines. The use of quasi-interpolation to perform collocation with quadratic splines was originally introduced in [13]. While that article considered a QI operator that is not a spline projector and focuses on uniform refinement of low degree splines, it still showed promising convergence properties that match the approximation power of the quadratic splines.

In the present paper, we connect these observations with the recently introduced framework of weighted isogeometric collocation of Schillinger et al. [38] and employ a particular class of locally supported spline projectors to propose the new method of Weighted Isogeometric Collocation based on Spline Projectors (WICSP). This allows us to consider isogeometric discretizations of PDEs, using splines of any degree. Moreover, we explore how these results can be extended to the case of adaptively refined THB-spline discretizations.

The remainder of the paper is organized as follows. First, Section 2 presents the general framework of weighted isogeometric collocation. Subsequently, we introduce our new discretization operator based on spline projectors in Section 3. The next two sections, which present numerical results and analyze the computational complexity, are devoted to tensor-product B-splines and to THB-splines, respectively. Finally, we conclude the paper with a discussion of possible future developments.

2. The weighted collocation framework

We present a general framework for isogeometric collocation. Moreover, we introduce the discretization operator later employed in WICSP.

2.1. Model problem and discretization

We consider a simple boundary value problem defined in $\Omega \subset \mathbb{R}^d$

$$\begin{aligned} Du - f &= 0 \quad u \in V_0 \\ u|_{\partial\Omega} &= 0 \end{aligned}$$

where V_0 is a suitable infinite dimensional space such that $u = 0$ on $\partial\Omega$ and D is a linear operator. For example, choosing $Du = -\Delta u$ leads to the Poisson equation. This includes the case of the pull-back of an equation to the parameter space which is needed to perform isogeometric simulations. In this situation, the operator D takes the form

$$Du = \sum_{t \in \mathcal{T}} \Phi_t \, D_t u \tag{1}$$

with elementary differential operators (i.e., derivatives) D_t and certain coefficient functions Φ_t that depend on the geometry mapping and on derivatives thereof. The index t varies in a finite index set $\mathcal{T}$, that does not depend on the discretization.

We perform a space discretization by introducing a finite-dimensional space $V_{0,h} \subset V_0$, which is spanned by the basis

$$B = \{\beta_j : j \in \mathcal{J}\}$$

with some index set

$$\mathcal{J} = \{j : 0 < j \leq n\}.$$

Clearly we have $|\mathcal{J}| = n = \dim V_{0,h}$. Our aim is to approximate the exact solution by an element of $V_{0,h}$, which is therefore expressed as a linear combination of the basis functions

$$u_h = \sum_{j \in \mathcal{J}} u_j \beta_j,$$

with coefficients $\{u_j\}_{j\in\mathcal{J}}$. In particular, we are interested in tensor-product spline spaces and their generalization to hierarchical splines, which are spanned by HB- and THB-splines. The appendix recalls the fundamental notions and definition concerning these spline functions.

Furthermore, we need to perform a discretization of the equations. This is achieved by invoking the linear *discretization operator*

$$\Lambda : C^r(\Omega) \to \mathbb{R}^m,$$

which is applied to the space discretization $Du_h - f$. The non–negative integer r denotes the smoothness which is needed to apply Λ. For instance, in the case of second order PDEs, we have $r = 0$ for standard Galerkin discretizations and $r = 1$ for collocation. The individual components of the operator will be denoted by

$$\lambda_i : C^r(\Omega) \to \mathbb{R}, \quad i = 1, \dots, m.$$

We arrive at the linear equations

$$\sum_{j\in\mathcal{J}} \lambda_i(D\beta_j)u_j = \lambda_i(f), \quad i = 1, \dots, m$$

which are solved for the coefficients u_j, either directly (if $m = n$) or in the least-squares sense (if $m > n$). We will focus on the first case, since the latter approach tends to generate matrices with a high condition number, cf. [1].

2.2. Discretization operators

Several possibilities for the choice of the operator Λ are available:

- In the **Galerkin** approach, one chooses

  $$\lambda_i^{\mathrm{G}}(g) = \int_\Omega g(x)\beta_i(x)\,\mathrm{d}x$$

  The integral is then often re-written via integration by parts, which leads to the corresponding weak form of the problem. This reduces the required order of differentiability for the trial functions. A similar approach, using test and trial functions that belong to different spaces, can be used to derive the Petrov-Galerkin method.

- An algebraically equivalent system of equations is obtained when considering any other basis $\Gamma = \{\gamma_j : j \in \mathcal{J}\}$ of $V_{0,h}$. Among the many bases of the of the discretization space, the dual basis with respect to the L^2 inner product is characterized by the conditions

  $$\langle \beta_i, \gamma_j \rangle = \int_\Omega \beta_i(x)\gamma_j(x)\mathrm{d}x = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

  It is closely related to the best L^2 approximation of $g \in C^r(\Omega)$ by an element of $V_{0,h}$, which takes the form

  $$\sum_{j\in\mathcal{J}} \langle \gamma_j, g\rangle \beta_j \ .$$

  For this basis we get the operator components

  $$\lambda_i^{\mathrm{DG}}(g) = \langle \gamma_i, g\rangle = \int_\Omega g(x)\gamma_i(x)\mathrm{d}x$$

  which we will denote as the **dual Galerkin** functionals. Clearly, using these functionals for the discretization is more of theoretical interest, since the dual basis functions are globally supported (thus leading to dense matrices) in general.

- An isogeometric **collocation**-based discretization is generated by choosing the functionals

$$\lambda_i^{\mathrm{IC}}(g) = \int_\Omega g(x)\delta(x-\xi_i)\mathrm{d}x = g(\xi_i)$$

where δ is the Dirac delta function and ξ_i denotes the i-th collocation node. The resulting system of equations then simplifies to point evaluations,

$$\sum_{j\in\mathcal{J}} D\beta_j(\xi_i)\, u_j = f(\xi_i), \quad i = 1,\dots,m$$

The choice of the collocation nodes is crucial for the stability and the approximation properties of the discretization.

- A similar approach results in the **weighted isogeometric collocation** (WIC) discretization. In this case we have

$$\lambda_i^{\mathrm{WIC}}(g) = \int_\Omega g(x) \sum_{k\in\mathcal{K}_i} w_{ik}\delta(x-\xi_k)\mathrm{d}x = \sum_{k\in\mathcal{K}_i} w_{ik}\, g(\xi_k)$$

where we use several collocation nodes ξ_k for each functional and a weighted sum with certain weights w_{ik}. The index k varies in an associated index set $\mathcal{K}_i$. The entries of the final linear system are weighted sums of point evaluations

$$\sum_{j\in\mathcal{J}} \big(\sum_{k\in\mathcal{K}_i} w_{ik}\, D\beta_j(\xi_k)\big)u_j = \sum_{k\in\mathcal{K}_i} w_{ik}\, f(\xi_k), \quad i = 1,\dots,m$$

As in the previous case, the choice of the collocation nodes and weights influences the properties of the method.

- We are interested in a special case of the latter framework. One may choose the components of the linear operator such that

$$\lambda_i^{\mathrm{WICSP}}(g) = \lambda_i^{\mathrm{DG}}(g)$$

holds for all $g \in V_{0,h}$. In other words, *we choose a weighted collocation-type discretization operator, which coincides with the dual Galerkin discretization operator on the discretization space.* This can be achieved with the help of a suitable spline projector, and thus we will will refer to it as *Weighted Isogeometric Collocation based on Spline Projectors* (WICSP).

3. The spline projector-based discretization operator

We present the QI scheme used in WICSP. We focus on its properties, such as accuracy and complexity, and discuss the construction of the grid of collocation nodes. Furthermore, we present some numerical examples which motivate our choice.

3.1. Quasi-interpolants

In order to find the functionals $\lambda_j^{\mathrm{WICSP}}$, we employ locally supported quasi-interpolant (QI) schemes, which are expressed as

$$Sg = \sum_{j\in\mathcal{J}} \lambda_j(g)\beta_j$$

where the coefficient functionals $\lambda_j(g)$ are typically computed through linear combinations of point evaluations. QI schemes turn out to be efficient methods to perform approximations of given functions using splines. In fact, they possess a low computational complexity and provide optimal approximation power under suitable assumptions. These include the property of reproducing

polynomials of degree p. QI schemes for univariate splines (that extend naturally to the tensor-product case) that possess this property were studied by Sablonnière in several papers, see [35] and the references cited therein. We are interested in QI schemes that possess the additional property of being *spline projectors*, since this appears to have a beneficial effect concerning the accuracy of the approximation. The following example motivates this additional requirement:

Example 1. We use the Sablonnière quasi-interpolation operator (denoted as P_h since it reproduces polynomials) to approximate $f(x) = \sin(5\pi x)$ on $\Omega = [0, 1]$ by splines with knots in $h\mathbb{Z}$ and compare the results with the fitting-based QI (denoted as S_h since it reproduces splines) from [17], which will be described in more detail in the remainder of this Section. Figure 1 reports the maximum error on $[0, 1]$ (left) and at knots and knot-span midpoints (right). While both schemes achieve the same optimal rate (order $p + 1$) of convergence, the spline projector leads to more accurate results, especially for larger values of the degree p. Moreover, it appears to create superconvergence at specific points for even degrees.

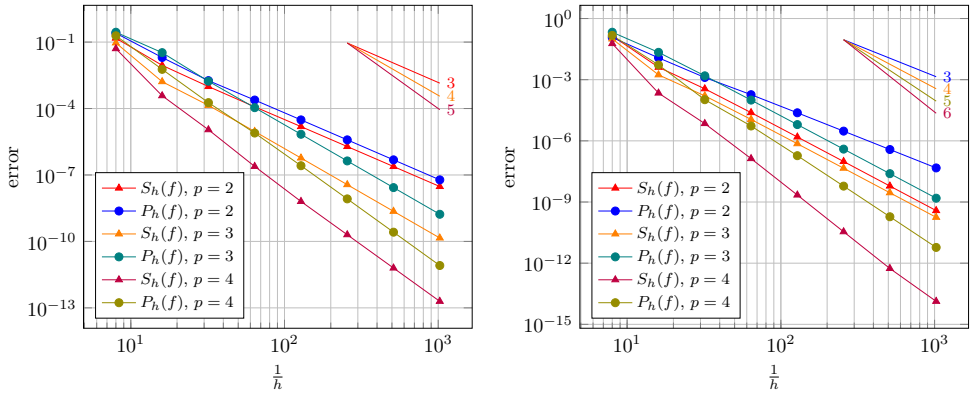

Figure 1: Max error in $[0, 1]$ (left) and max error at knots and knots-span midpoints for S_h and P_h (right).

In the case of *hierarchical* B-splines, spline projectors were investigated already in [25], based on a recursive construction. The case of THB-splines was first studied in [40], introducing a scheme that required $\mathcal{O}(p^d)$ evaluations and $\mathcal{O}(p^d)$ floating point operations per degree of freedom. The follow-up paper [39] improved the efficiency by sacrificing the spline projector property.

More recently, a fitting-based construction of spline projectors, requiring only $\mathcal{O}(1)$ function evaluations (and $\mathcal{O}(p^d)$ floating point operations) per degree of freedom, has been derived in [17] and was experimentally shown to give more accurate results. This construction was shown to be general and suitable for (T)HB-splines including tensor-product splines.

We briefly recall these results, in order to make this paper self-contained. We restrict ourselves to the case of admissible hierarchical meshes of class 2 [7]. A similar construction can be performed also for a non-graded mesh but would incur a higher computational cost. It has been shown that mesh grading ensures better conditioning and higher sparsity of the resulting system matrices.

3.2. The grid of weighted collocation nodes

The construction is based on a *grid of weighted collocation nodes*

$$\{\eta_k \,:\, k \in \mathcal{K}\}$$

with indices taken from a suitable index set $\mathcal{K}$, where we perform evaluations. We distinguish between two cases:

- Case 1 – Tensor-product splines: Here we make use of superconvergent points. It has been noted that the use of those points results in improved approximation properties of collocation based technologies [1], [30]. These points are an approximation of the Cauchy-Galerkin points

(i.e., the points where the Galerkin residual is zero). Estimates of their location are reported in Table 1. These points naturally define a regular grid, see Figure 2. We note that in the case of basis functions of even degree, that grid is obtained simply via one round of dyadic subdivision.

- Case 2 – Hierarchical splines: We construct the grid subdividing each active cell either once or twice via dyadic subdivision, and the resulting vertices form the nodes, see Figure 3 for an example. An active cell of level ℓ is subdivided twice if the support of at least one of the $(p+1)^d$ TPB-splines of level ℓ that are non-zero on it contains an active cell of level $\ell+1$, and only once otherwise. Note that this grid contains the Greville points for uniform (i.e., tensor-product) meshes, and it is then even identical to the grid of superconvergent points for even degrees.

Table 1: Estimated location of the superconvergent points in $[-1,1]$

Degree	SC. points
$p=3$	$\pm\frac{1}{\sqrt{3}}$
$p=4$	$-1,0,1$
$p=5$	$\pm\frac{\sqrt{225-30\sqrt{30}}}{15}$
$p=6$	$-1,0,1$

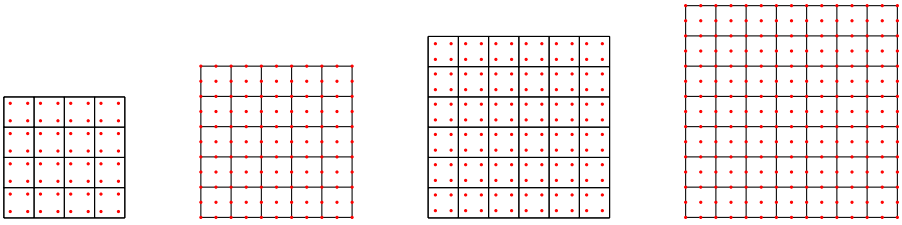

Figure 2: From left to right, superconvergent grid for B-splines of degree $p=3,4,5,6$.

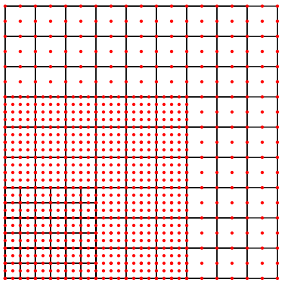

Figure 3: Example of mesh and corresponding grid of evaluation nodes for THB-splines of degree $p=3$.

3.3. Defining the functional via local least-squares approximation

For each basis function β_j we obtain the functional λ_j^{WICSP} as follows:

1. We define the local domain[1] $\Omega_j = \operatorname{supp}\tilde{\beta}_j$, where $\tilde{\beta}_j$ denotes the mother (i.e., un-truncated) HB-spline of the THB-spline β_j. In particular, we use $\beta_j = \tilde{\beta}_j$ for tensor-product splines.

[1]The assumption of graded meshes implies that in each domain Ω_j at most two hierarchical levels are present.

2. The corresponding local hierarchical spline space V_j is defined by restricting the mesh of $V_{0,h}$ to Ω_j, with the associated locally defined truncated hierarchical basis $B_j = \{\hat{\beta}_\ell : \ell \in \mathcal{L}_j\}$. This basis includes β_j by construction.
3. We select a local grid of evaluation nodes $(\eta_k)_{k\in\mathcal{K}_j}$, $\mathcal{K}_j \subset \mathcal{K}$ is a subset of the full index set that we introduced for the evaluation nodes. In order to reduce the total number of function evaluations, we choose them from the global set of evaluation nodes, trying to re-use them as often as possible by employing index sets $\mathcal{K}_j$ that are non-disjoint. Again we distinguish between two cases:
   - Case 1 – Tensor-product splines: We use all the grid points within the support of the function. (We use enlarged supports near the domain boundaries.)
   - Case 2 – Hierarchical splines: The support of the mother HB-spline consists of $(p+1)^d$ cells of level ℓ. We use the grid points obtained by applying dyadic subdivision twice (if next finer level is present within the support) or once (otherwise). This results in $(4p+5)^d$ and $(2p+3)^d$ nodes in the first and second case, respectively. In the first case, the number of nodes can be decreased to $(3p+2)^d$ to obtain the *reduced point grid*, where the centrally located $p+3$ nodes per direction are placed with half the density of the p nodes near the boundaries, see Figure 4.
4. We determine the functional $\lambda_j^{\mathrm{WICSP}}$ approximating the function g on Ω_j via discrete L^2 fitting. More precisely, we solve the equations

$$\sum_{\ell\in\mathcal{L}_j} c_\ell \hat{\beta}_\ell(\eta_k) = g(\eta_k) \ , k \in \mathcal{K}_j \ ,$$

   in the least-squares sense, which results in the coefficients

$$(c_\ell)_{\ell\in\mathcal{L}_j} = (A^T A)^{-1} A^T f$$

   with

$$A = (\hat{\beta}_\ell(\eta_k))_{\ell\in\mathcal{L}_j, k\in\mathcal{K}_j}, \qquad f = (g(\eta_k))_{k\in\mathcal{K}_j}.$$

   The selection of the evaluation nodes ensures unisolvency. The functional $\lambda_j^{\mathrm{WICSP}}$ is finally determined by setting $\lambda_j^{\mathrm{WICSP}} = c_\ell$, where $\hat{\beta}_\ell = \beta_j$. More precisely, we pick the row of the matrix $(A^T A)^{-1} A^T$ that corresponds to β_j and multiply it with the vector f.

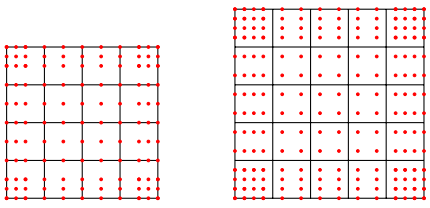

Figure 4: Reduced grid of nodes for degree $p = 3$ (left) and $p = 4$ (right).

We analyze the computational cost per coefficient of the projection. The domain Ω_j consists of active cells of at most two levels due to the mesh grading assumptions. The number of possible combinations of active cells is bounded by $2^{(p+1)^d}$, which is a very pessimistic upper bound.

Moreover, the grid of nodes contains $\mathcal{O}(p^d)$ points. This holds for both the grids obtained by dyadic refinement and for the superconvergent one, which is employed in the tensor-product case.

Finally, we have to evaluate the effort to perform the local discrete L^2 fitting. We may assume that all the (not more than $2^{(p+1)^d}$) possible rows of the matrices $(A^T A)^{-1} A^T$ have been precomputed and stored in a look-up table. Finding the right entry via binary search requires at most

$$\log 2^{(p+1)^d} = \mathcal{O}(p^d)$$

operations. The unknown coefficient is calculated performing the inner product between the precomputed row vector of the matrix $(A^TA)^{-1}A$ (which is a generalized inverse matrix of A) and the column vector that collects the values of g on the grid of evaluation nodes.

We conclude that the projection step requires $\mathcal{O}(p^d)$ flops. More precisely, *the number of multiplications – and also the number of additions – is equal to the number of evaluation nodes.*

3.4. Use of QIs for discretizing ODEs

We apply WICSP and Sablonnière's quasi-interpolation operator to the Poisson equation

$$-u''(x) = f \quad \text{on} \quad \Omega = [0,1]\ .$$

We consider the exact solution $u(x) = \sin(5\pi x)$ and impose the Dirichlet boundary conditions by simply interpolating the values at the boundary.

Figure 5 reports the resulting convergence rates for degrees $p = 3, \dots 6$. The following observations are in order:

- Both methods perform equally well for low degrees ($p \leq 4$) on dyadic grids, while the spline projector-based discretization gives more accurate results on the superconvergent grid of points for $p = 3$.
- The discretization via weighted collocation based on Sablonniére's quasi-interpolation operators essentially fails for larger degrees, $p \geq 5$. This is due to the fact that the total number of collocation nodes is less than the dimension of the spline space.
- The discretization via WICSP also works for larger degrees. The use of the superconvergent grid of points for odd degrees gives more accurate results.

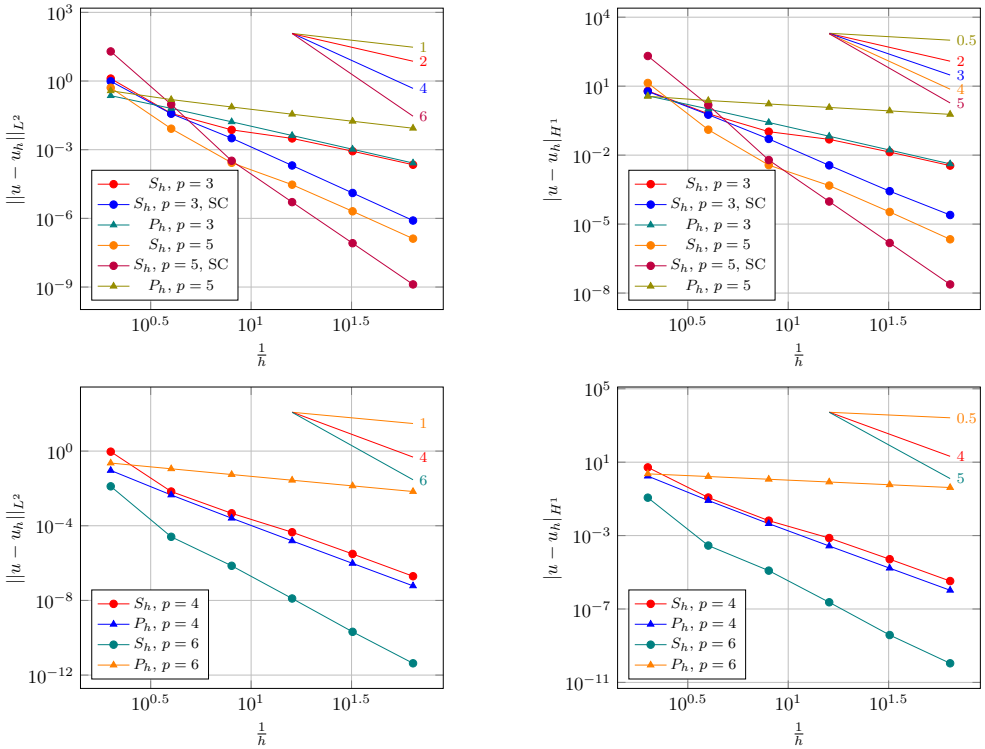

Figure 5: Convergence rate for the 1D Poisson equation using S_h and P_h for different degrees p.

Remark. The bandwidth of the resulting stiffness matrix also depends on the discretization method. WICSP, Sablonnière's method and Galerkin projection gives bandwidth $2p + 1$, while standard collocation leads to matrices with bandwidth $p + 1$.

4. Tensor-product spline discretization

We test our method performing different experiments on the domain Ω represented in Figure 6. The geometry is a quarter of circular ring represented in terms of NURBS. We have carried out the entire implementation in C++ with the use of the `G+Smo` library [28]. The experiments are run in a Linux workstation equipped with an Intel Xeon W3680 CPU, 3.33 GHz with 24 GB of memory.

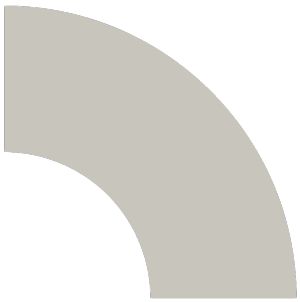

Figure 6: Domain Ω used for the numerical experiments.

4.1. Computational results

Convergence and approximation power. First, we analyze the approximation power when tensor-product splines are used. We verify the convergence behaviour of our method solving the Poisson equation on the domain Ω with known exact solution

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

and with source function $f(x, y) = 8\pi^2 \sin(2\pi x) \sin(2\pi y)$ computed accordingly. We impose the Dirichlet boundary conditions by restricting the exact solution to the boundary curves of Ω.

We want to verify the influence of the choice of the collocation points on the approximation power. We investigate the two discussed choices of point grids. The first one consists in the regular grid of points obtained by one round of dyadic subdivision, as already mentioned the Greville points form a subset of our grid. The second grid is constructed choosing the superconvergent (SC) points with respect to the degree of the basis.

Moreover, we analyze the behaviour with respect to the degree p of the basis, namely when B-splines of degree even or odd are used. The obtained convergence rate is reported in Figure 7 and 8 for degree odd and even respectively.

We obtain a different behaviour when discretizations of even or odd degree are considered.

- The use of the dyadic point grid leads to a convergence rate of order $p - 1$, both for the L^2 norm and H^1 semi-norm of the error if p is odd. However, the rate of convergence is always equal to p for even degree. This matches the known results for Greville point-based collocation [3].

- The use of superconvergent points gives better results. We observe a convergence rate of order $p + 1$ for the L^2 norm and p for the H^1 semi-norm of the error if p is odd. For even degree p, the grid of superconvergent points is equal to the dyadic grid, and we already noted that the rate of convergence is equal to p. Again, this is in agreement with the results presented in [1] and [30].

The results are summarized in Table 2.

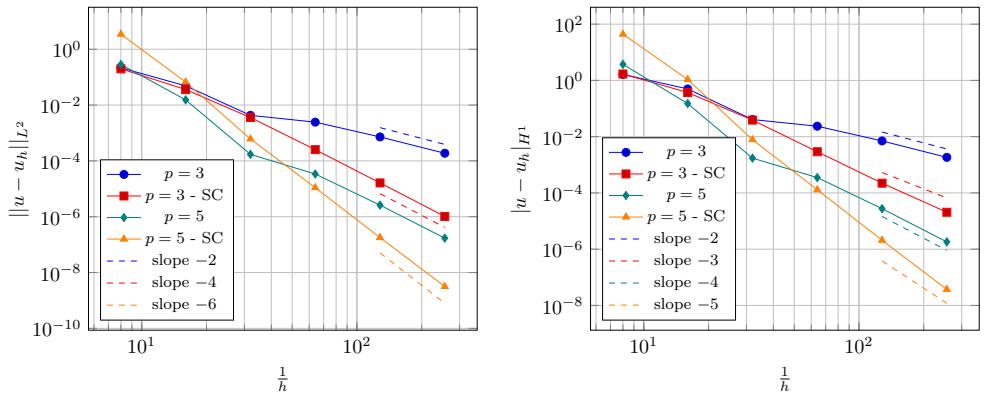

Figure 7: Convergence rate for odd degrees, both the for the dyadic grid and the grid of superconvergent (SC) points.

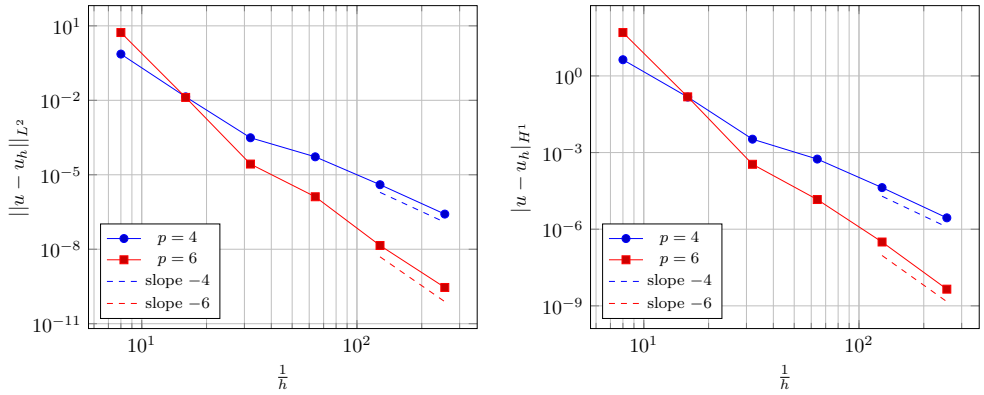

Figure 8: Convergence rate for even degrees, where the dyadic grid coincides with the grid of superconvergent points.

Comparison with Galerkin projection. We have verified that Galerkin-based isogeometric analysis and our weighted collocation approach on the superconvergent grid of points lead to the same approximation power for odd degree p. We now compare the accuracy of the two approaches.

Figure 9 reports the numerical errors obtained solving the Poisson equation with solution (2) on Ω. We observe that the Galerkin approach always leads to a higher accuracy, compared with our method. This is in line with the results presented in the literature about classical isogeometric collocation.

Condition number. We investigate numerically the condition number $\kappa(S)$ of the stiffness matrix for our method when the dyadic grid of collocation points is employed. We also compare the results with the ones obtained applying Greville collocation and Galerkin discretization. Again, we consider as a model problem the Poisson equation defined on Ω.

First, we analyze the growth of the condition number for WICSP with respect to h-refinement. In order to do so, we repeatedly perform uniform refinement and compute $\kappa(S)$ for different degrees

Table 2: Approximation power

	Point grid		SC. point grid	
	Odd p	Even p	Odd p	Even p
L^2	$p-1$	p	$p+1$	p
H^1	$p-1$	p	p	p

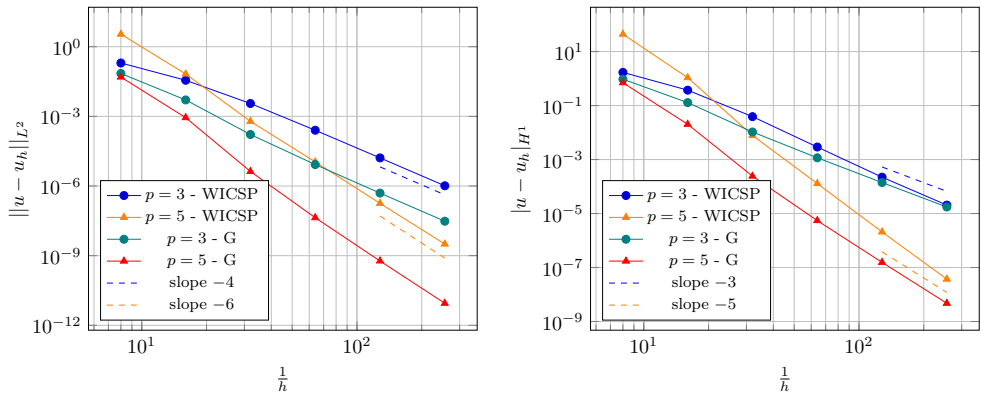

Figure 9: Numerical errors for isogeometric discretization of odd degree using WICSP and the Galerkin method.

p. The obtained results are presented in Figure 10 (left). We always observe experimentally a quadratic growth of the condition number for each degree p of the discretization.

Second, we investigate the dependence of $\kappa(S)$ under p-refinement and perform a comparison with the ones obtained using Greville collocation and Galerkin discretization. Therefore, we fix h and compute the condition number of the stiffness matrices arising from the application of WICSP, Greville collocation (C) and Galerkin discretization (G) for varying degrees p. The corresponding results are reported in Figure 10 (right). We observe that WICSP and Greville collocation appear to give rise to better conditioning of the resulting system matrices for high degree p, compared to the Galerkin discretization. Furthermore, WICSP always results in slightly higher condition numbers, compared to Greville collocation.

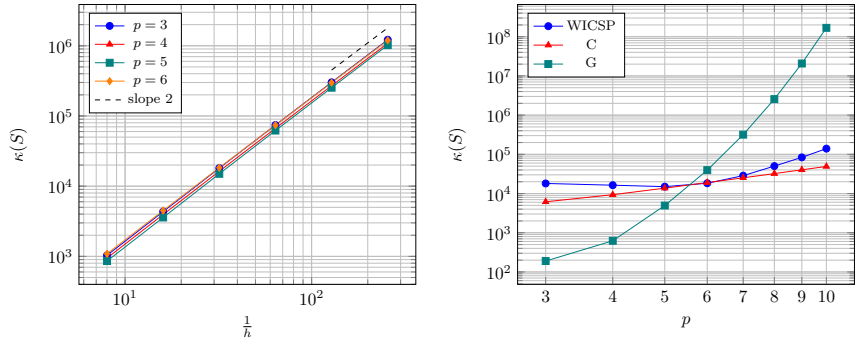

Figure 10: Left: Condition number of the stiffness matrix created by WICSP for various values of h for different degrees p. Right: Condition number for several values of p for WICSP, collocation at Greville points (C) and for Galerkin-based discretization (G).

4.2. Complexity of the matrix assembly

We now analyze the costs of the matrix assembly process of our approach. First, we discuss the case of dimension $d = 2$. We compute the elements of the system matrix

$$S_{ij} = \int_\Omega D\beta_j(x) \sum_{k \in \mathcal{K}_i} w_{ik} \delta(x - \xi_k) \mathrm{d}\Omega = \sum_{k \in \mathcal{K}_i} w_{ik}\, D\beta_j(\xi_k)$$

where the operator D takes the form $D = \sum_{t \in \mathcal{T}} \Phi_t D_t$, see Eq. (1). The coefficients w_{ik} are given by the spline projector (i.e., by the row of the matrix $(A^T A)^{-1} A^T$ that corresponds to the coefficient of β_i). They have a tensor-product structure $w_{ik} = w^1_{i_1 k_1} w^2_{i_2 k_2}$.

More precisely, we assemble the matrix

$$S_{ij} = \sum_{t\in\mathcal{T}} \sum_{k\in\mathcal{K}_i} w_{ik}\Phi_t(\xi_k)\, D_t\beta_j(\xi_k) \ .$$

The assembly procedure can be performed efficiently by exploiting the tensor-product structure of the discretization. We invoke a sum factorization technique that was already adopted in [2], [6] and [32].

Focusing on the bivariate case, we introduce the multi-index notation $i = (i_1 i_2)$, $j = (j_1 j_2)$, $k = (k_1 k_2)$ and consider the separate components of the operator $D_t = D_t^1 D_t^2$ in the two parametric directions and rewrite the matrix element as

$$S_{(i_1 i_2)(j_1 j_2)} = \sum_{t\in\mathcal{T}} \Bigg(\sum_{k_1\in\mathcal{K}_{i_1}^1} D_t^1\beta_{j_1}^1(\xi_{k_1}^1)\, w_{i_1 k_1}^1 \underbrace{\sum_{k_2\in\mathcal{K}_{i_2}^2} D_t^2\beta_{j_2}^2(\xi_{k_2}^2)\, w_{i_2 k_2}^2\, \Phi_t(\xi_{k_1}^1,\xi_{k_2}^2)}_{X_{t,i_1 i_2 j_2 k_1}} \Bigg) \ .$$

We analyze the complexity, focusing on a single instance of t:

- First, we focus on the evaluation of $X_{t,i_1 i_2 j_2 k_1}$. This calculation requires to loop through all the $\mathcal{O}(n_1 n_2) = \mathcal{O}(n)$ possible indices i_1 and i_2 of basis functions. For each pair of indices, we have to consider $2p+1$ overlapping basis functions with index j_2, and $\mathcal{O}(p^2)$ collocation nodes with indices $k_1 \in \mathcal{K}_{i_1}^1$ and $k_2 \in \mathcal{K}_{i_2}^2$. Since the computational effort for each such 6-tuple of indices is constant, we arrive at total costs of $\mathcal{O}(np^3)$ flops.
- We are now ready to evaluate the sum with respect to k_1, which requires the execution of two loops with $\mathcal{O}(n_1 n_2)$ index pairs (i_1, i_2). Similar to the inner sum, we consider the overlapping basis functions with indices j_1 and j_2 and the evaluation nodes in the first direction with indices k_1. All these calculations lead to costs of $\mathcal{O}(p^3)$ flops for each pair (i_1, i_2).
- Finally we note the size of $|\mathcal{T}| = \mathcal{O}(1)$ is discretization-independent, thus we conclude that the overall complexity to assemble the resulting system matrix results equal to $\mathcal{O}(np^3)$.

An analogous analysis can be carried out in higher dimensions, again exploiting the tensor-product structure of the discretization. In general, the computational costs are equal to $\mathcal{O}(np^{d+1})$ flops. We note that the costs of our approach are in line with the most advanced techniques for matrix assembly in Galerkin-based isogeometric analysis with tensor-product splines [8, 32]. However, the resulting costs are higher than the one of the standard collocation-based approach. This is due to the fact that the spline projector approach requires more collocation nodes compared to the standard isogeometric collocation method.

4.3. Matrix-free application

The approximate numerical solution of PDEs is typically obtained solving large and sparse linear systems of equations. Iterative solvers are preferred over more memory-consuming direct methods. Moreover, it has been observed that these methods only require the calculation of matrix-vector products without the need of explicitly storing the system matrix that arises from the discretization of a PDE. This has been recently exploited in [8] to define efficient isogeometric matrix-free solvers. Here we explore the applicability of our collocation method in a matrix-free framework.

We analyze the computational costs of the matrix-vector multiplication

$$v_i = (S_{ij}u)_i = \sum_j S_{ij}u_j = \sum_j \left(\sum_{t\in\mathcal{T}} \sum_{k\in\mathcal{K}_i} w_{ik}\Phi_t(\xi_k)\, D_t\beta_j(\xi_k) \right) u_j \ .$$

In order to make use of the sum-factorization approach, we rewrite the sum as

$$v_{(i_1 i_2)} = \sum_{k_1 \in \mathcal{K}^1_{i_1}} w^1_{i_1 k_1} \underbrace{\sum_{k_2 \in \mathcal{K}^2_{i_2}} w^2_{i_2 k_2} \sum_{t \in \mathcal{T}} \Phi_t(\xi^1_{k_1}, \xi^2_{k_2}) \underbrace{\sum_{j_1} D^1_t \beta^1_{j_1}(\xi^1_{k_1}) \underbrace{\sum_{j_2} D^2_t \beta^2_{j_2}(\xi^2_{k_2}) u_{(j_1 j_2)}}_{x_{t,k_2 j_1}}}_{y_{t,k_1 k_2}}}_{z_{k_1 i_2}} .$$

The calculation is performed recursively:

- The evaluation of the first (innermost) term $x_{t,k_2 j_1}$ requires to loop through the $\mathcal{O}(n_1 n_2) = \mathcal{O}(n)$ indices k_2 and j_1. The loop with index j_2 needs to visit only $\mathcal{O}(p)$ basis functions. We already noted that $|\mathcal{T}| = \mathcal{O}(1)$. Therefore, the costs to compute the first term are equal to $\mathcal{O}(np)$.
- The evaluation of the second term $y_{t,k_1 k_2}$ requires to visit $\mathcal{O}(n_1 n_2) = \mathcal{O}(n)$ indices k_1 and k_2 and $\mathcal{O}(p)$ basis functions with index j_1. This results in total costs of $\mathcal{O}(np)$ flops.
- The third term $z_{k_1 i_2}$ is computed by looping through all the $\mathcal{O}(n_1 n_2)$ indices k_1 and k_2 and $\mathcal{O}(p)$ overlapping basis functions i_2. Again, the costs are equal to $\mathcal{O}(np)$.
- Finally, we compute $v_{i_1 i_2}$ with $\mathcal{O}(np)$ operations via two loops through $\mathcal{O}(n_1 n_2)$ instances of the indices i_1 and i_2. For each pair, we have to consider only $\mathcal{O}(p)$ collocation nodes k_1, which leads to the total complexity of this step.

While we analyzed the behaviour of the matrix-vector product in the bivariate case, the analysis carries over to higher dimensions. We conclude that the computational complexity of our method in the matrix-free context is equal to $\mathcal{O}(np)$, thus it is dimension-independent. Similar results are available for standard isogeometric collocation and for Galerkin projection via weighted quadrature [36].

4.4. Summary and comparison with other methods

We summarize three main observations regarding the spline projector-based weighted collocation for tensor-product splines, cf. Figure 11:

- It gives the optimal rate of convergence for odd degrees (red disk), similar to standard Galerkin methods, least-squares and clustered collocation, and Galerkin via weighted quadrature.
- It gives the dimension-independent complexity $\mathcal{O}(np)$ for the matrix-free application like Greville collocation, Galerkin via weighted quadrature, and clustered collocation.
- It can be realized without using the least-squares approach (that would lead to higher condition numbers).

Among the six approaches that are identified in the figure, our method belongs to the core group that possess these three advantageous properties.

5. Adaptive THB-spline discretization

While it also performs well in the tensor-product case, the true advantage of our approach relies in its application to adaptive discretization strategies with THB-splines. We will show that the spline projector-based collocation admits a natural generalization to the case of THB-splines. Moreover, we show that the computational complexity of the assembly compares well with Galerkin-based discretizations.

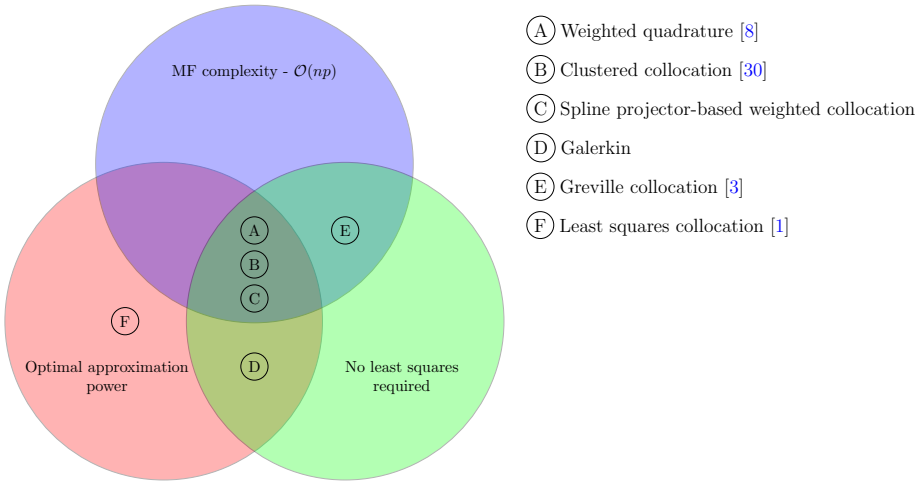

Figure 11: Properties of the different methods.

5.1. Computational results

Convergence and approximation power. We verify the suitability of our method when adaptive refinement with THB-splines is performed. More precisely, we will show that the WICSP method is directly applicable to THB-spline discretizations and it preserves the advantages of using the adaptive refinement strategy.

As before we consider the Poisson equation on the domain Ω (see again Figure 6), but this time with the known exact solution

$$u(x,y) = e^{-10^3\left((x-1)^2+(y-1)^2\right)} \ . \tag{3}$$

The solution exhibits a sharp exponential peak centered at $(1,1)$. This makes it an appropriate test case to evaluate the performance of the adaptive refinement strategy.

We use basis functions which are at least C^1 continuous. Thus we can adopt the a posteriori error estimate

$$\eta_E^2 = h_E^2 ||f - Du_h||^2_{L^2(E)}$$

defined on an active cell (or element) $E \subset \Omega$, where h_k denotes the diameter of the selected element E. This error estimate, although based on the weak form of the problem, performed well in all our numerical experiments. The study of error estimators for isogeometric collocation is still a topic of ongoing research, see e.g. [22].

We mark the active cell E for refinement if η_E exceeds a suitable threshold Θ

$$\eta_E \geq \Theta$$

There are several strategies to select the threshold Θ. We employ the "relative threshold" approach discussed in [14]. We control the percentage of active cells that will be marked in each refinement step by choosing Θ such that

$$|\{E : \eta_E > \Theta\}| \approx (1-\psi)|\{E\}|$$

The factor $1-\psi$ identifies the percentage of active cells that will be marked for refinement. In our experiments we set $\psi = 0.85$.

Finally, we refine by adopting a suitable mesh grading strategy, in order to ensure the admissibility of the resulting mesh.

Figure 12 reports the L^2 and H^1 norms of the error obtained by using WICSP for THB- and tensor-product splines of degree $p = 3$ and $p = 4$, where we employed the super-convergent collocation nodes in the latter case. As to be expected, we always observe a superior performance

of the adaptive refinement strategy compared to the uniform refinement case. In fact, we always obtain a comparable accuracy with a significant lower number of degrees of freedom. This is due to the fact that in the adaptive refinement case the refinement is correctly concentrated around the location of the peak of the exact solution $u(x, y)$ where the error results larger. The use of tensor-product refinement leads to over-refinement in areas of the domain where the error is already negligible. An example of the obtained solution and the corresponding hierarchical mesh is depicted in Figure 13.

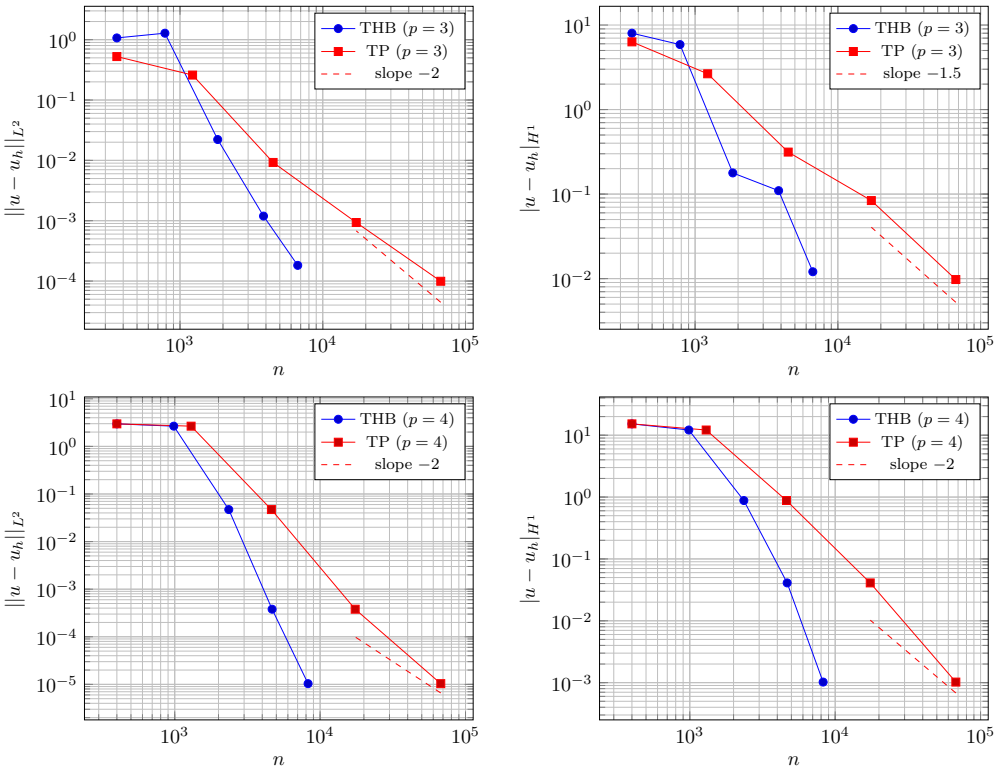

Figure 12: L^2 and H^1 errors of WICSP for THB- and tensor-product splines for $p = 3$ (top row) and $p = 4$ (bottom row).

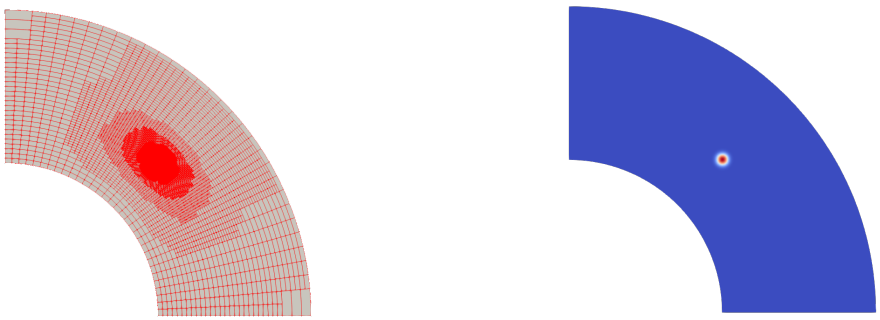

Figure 13: Mesh obtained via adaptive refinement (left) and corresponding solution u_h (right) for $p = 3$.

Comparison with Galerkin projection. In order to compare with the Galerkin discretization, we again consider the Poisson equation on Ω with the known exact solution (3). Figure 14 reports the results obtained when using tensor-product splines and THB-splines. We note again that the use

of the Galerkin approach always leads to a better accuracy (with higher computational costs for the matrix assembly) if compared with our weighted collocation approach. However, both THB-spline based methods (WICSP and Galerkin) perform significantly better than the corresponding tensor-product discretizations, due to the near-singular nature of the exact solution.

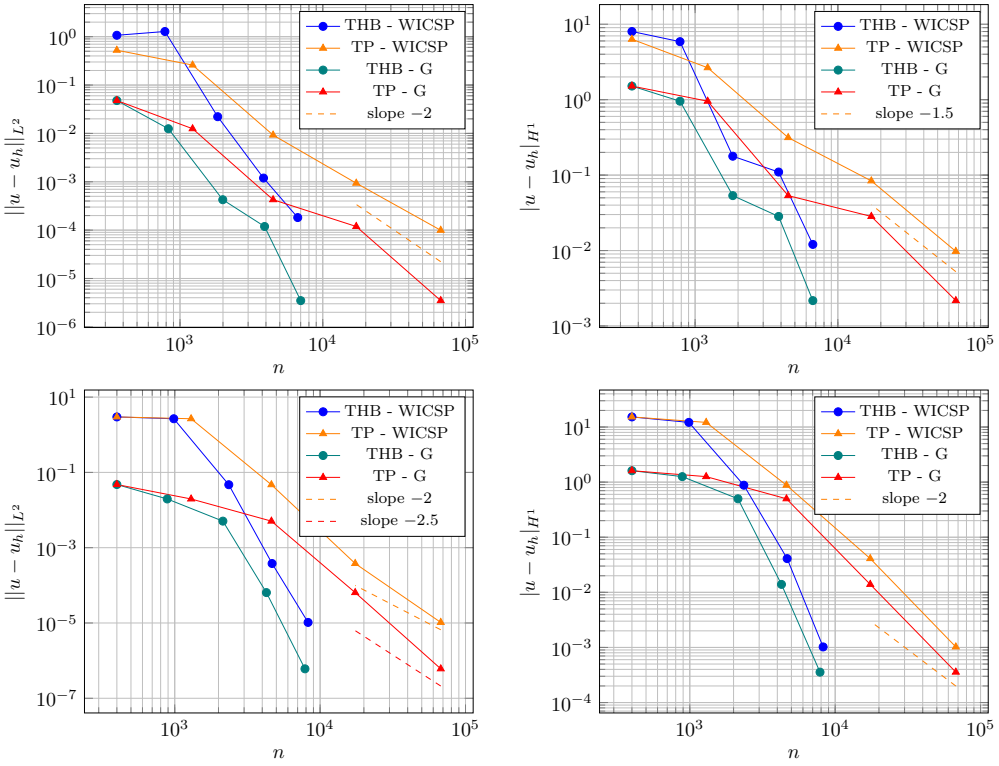

Figure 14: L^2 and H^1 errors of WICSP and Galerkin discretization for THB- and tensor-product splines for $p = 3$ (top row) and $p = 4$ (bottom row).

5.2. Complexity of the matrix assembly

We now analyze the costs of the matrix assembly process of our approach. We compare it with the use of Gaussian quadrature in the context of Galerkin projection, since this is currently the only viable approach to isogeometric analysis based on THB-splines.

We compute the elements of the system matrix

$$S_{ij} = \int_\Omega D\beta_j(x) \sum_{k \in \mathcal{K}_i} w_{ik} \delta(x - \xi_k) \mathrm{d}\Omega = \sum_{k \in \mathcal{K}_i} w_{ik}\, D\beta_j(\xi_k)$$

where the operator D again takes the form $D = \sum_{t \in \mathcal{T}} \Phi_t D_t$, see Eq. (1). The coefficients w_{ik} are given by the spline projector (i.e., by the row of the matrix $(A^T A)^{-1} A^T$ that corresponds to the coefficient of β_i). Neither these coefficients nor the basis functions possess a tensor-product structure, hence it is impossible to employ sum factorization. We perform the assembly via the straightforward Algorithm 1.

First, the matrix assembly routine requires a loop through all the $\mathcal{O}(n)$ basis functions. Second, we visit all the $\mathcal{O}(p^d)$ overlapping basis functions. Finally, we have to consider $\mathcal{O}(p^d)$ collocation nodes in the support intersection. We conclude that the computational complexity equals $\mathcal{O}(np^{2d})$.

The use of Gaussian quadrature in the case of Galerkin-based THB-spline discretizations entails computational costs of $\mathcal{O}(np^{3d})$ floating point operations. We therefore obtain a considerable advantage in terms of calculation time as p increases. This is due to the fact that our spline

Algorithm 1: Matrix assembly with THB-splines

```
// loop through basis functions
for i ← 1 to n do
    // loop through basis functions that overlap β_i
    for j : supp β_j ∩ supp β_i ≠ ∅ do
        // loop through collocation nodes in supp β_j ∩ supp β_i
        for k : ξ_k ∈ supp β_j ∩ supp β_i do
            // compute matrix entry
            S_ij = S_ij + w_ik Σ_{t∈T} Φ_t(ξ_k) D_t β_j(ξ_k)
        end
    end
end
```

projector-based approach utilizes a suitable point grid (with a p-independent number of evaluation nodes per active cell) as described in Section 3.

In order to verify experimentally the predicted computational complexity, we adopt the benchmark proposed in [33]. For each degree p we generate a hierarchical mesh consisting of four levels as follows: We initialize a uniform tensor-product mesh of $4p \times 4p$ active cells, which is then refined repeatedly by using the $2p \times 2p$ active cells of the bottom-left corner to obtain the next finer level. In this way we always obtain an admissible mesh. The meshes obtained for $p = 3$ and $p = 6$ are depicted in Figure 15.

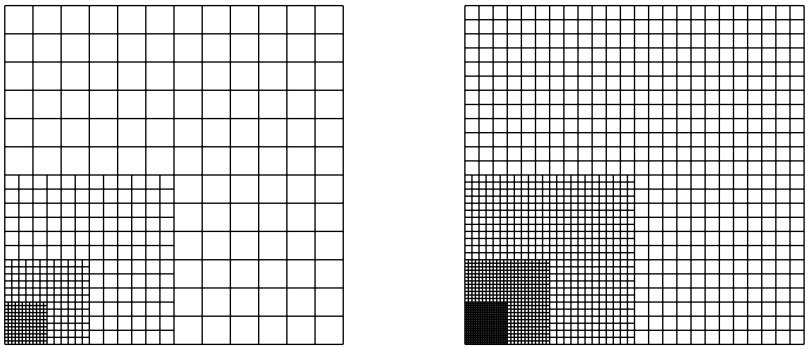

Figure 15: Mesh for $p = 3$ (left) and for $p = 6$ (right).

First, we verify the linear growth of the computational cost with respect to the number of degrees of freedom. Every active cell of the meshes obtained by the previous procedure is repeatedly refined (4 times in total) via dyadic subdivision in order to obtain a progressively increasing number of degrees of freedom. Equivalently, this can be seen as increasing the levels of all subdomains by $1, 2, 3, 4$. We then run our code and measure the time needed to assemble the resulting system matrix varying n for different values of p. The obtained results are reported in Figure 16 (left). We observe a linear dependence of the assembly time with respect to the total number of degrees of freedom. This is coherent with the analysis.

Next, we test the behaviour of the assembly time against the degree p and perform a comparison with the one obtained via Gauss quadrature, which is implemented in the `G+Smo` library. For the latter, we use $(p+1)^d$ nodes per active cell. This choice has been theoretically proven to guarantee optimal approximation power and it is typically adopted in matrix assembly routines. We consider again the meshes generated as described above and measure the assembly time for various values of the degree p. Figure 16 (right) presents the obtained experimental results.

We observe that the rate of growth of the computational time tends to $4 = 2d$ for our approach,

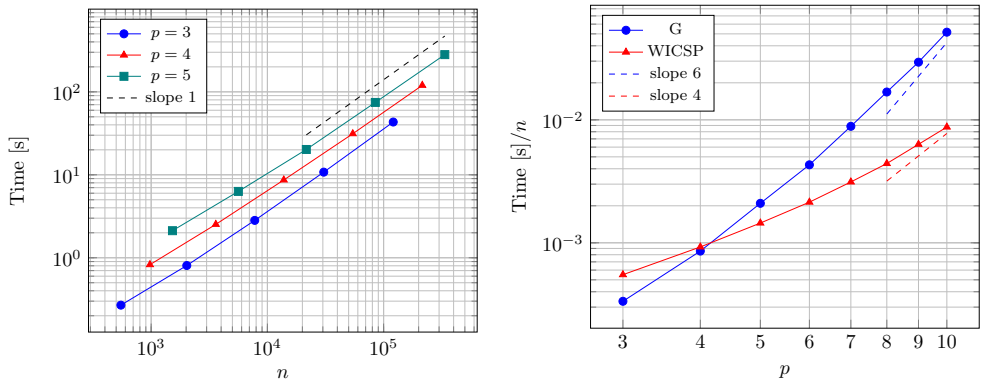

Figure 16: Matrix assembly time with respect to n for different degrees p (left) and matrix assembly time with respect to p (right).

while Gauss quadrature gives $6 = 3d$ as expected. However, we also note that our method is slightly slower than Gauss quadrature for low degrees. We attribute this fact to the extremely optimized implementation provided within the `G+Smo` library.

In order to further understand this discrepancy, we count the total number of visits of the evaluation nodes for WICSP and for Gauss quadrature that are needed to assemble the system matrix. More precisely, we count these visits for various values of the degree p, and for the sequence of multi-level-meshes generated by the above procedure, see Figure 15, and for THB meshes with only a single level (which are indeed tensor-product meshes). For simplicity, we did not employ the reduced point grid in the case of WICSP, even though this would have reduced the total number of visits.

The obtained results are reported in Figure 17. Two observations are in order:

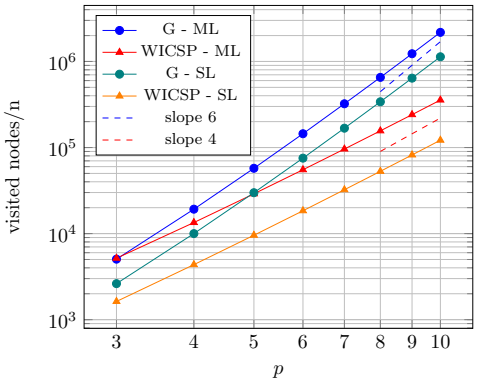

Figure 17: Number of visits of the evaluation nodes during the matrix assembly procedure for THB-splines on multi-level (ML) and single-level (SL) meshes.

- We obtain a similar number of visits of the evaluation nodes already for degree $p = 3$ when THB-splines are used.
- The advantage of using the WICSP method becomes more pronounced for meshes that are closer to the tensor-product case. This is due to the fact that number of evaluation nodes for WICSP attains its minimum (i.e. $(2p + 3)^d$) in this situation.

We conclude that the WICSP method possesses computational advantages with respect to matrix assembly already for relatively low degrees.

5.3. Matrix-free application

We discuss the design of efficient matrix-free solvers for isogeometric analysis with THB-splines in the context of the WICSP method. As already mentioned, sum-factorization approaches are not viable when THB-splines are used and therefore a different assembly algorithm is required to deal with the lack of tensor-product structure.

In order to define a matrix-free solver, we need to compute the matrix-vector product

$$v_i = (S_{ij}u)_i = \sum_j \left(\sum_{t\in\mathcal{T}} \sum_{k\in\mathcal{K}_i} w_{ik}\Phi_t(\xi_k)\, D_t\beta_j(\xi_k) \right) u_j = \sum_{k\in\mathcal{K}_i} w_{ik} \underbrace{\sum_{t\in\mathcal{T}} \sum_j \Phi_t(\xi_k)\, D_t\beta_j(\xi_k) u_j}_{v_{ik}}$$

The algorithm used to perform the multiplication is presented in Algorithm 2.

Algorithm 2: Evaluating a matrix-vector product with THB-splines

```
// precomputation of point values
// loop through collocation nodes
for k ← 1 to |K| do
    // loop through basis functions active on ξ_k
    for j : β_j(ξ_k) ≠ 0 do
        // compute point value
        v_ik = v_ik + Σ_{t∈T} Φ_t(ξ_k) D_t β_j(ξ_k) u_j
    end
end
// compute the vector v
// loop through basis functions
for i ← 1 to n do
    // loop through collocation nodes in supp β_i
    for k : ξ_k ∈ supp β_i do
        // compute vector entry
        v_i = v_i + w_ik v_ik
    end
end
```

We analyze the computational costs and compare them with Gauss quadrature:

- A preprocessing step is needed to store the relevant quantities later used to compute the vector components. This requires to first loop through all the collocation nodes. We have already seen how our spline projector approach permits to define a degree independent set of $\mathcal{O}(n)$ nodes. We then need to consider all the basis functions active on the selected node. The mesh grading assumption allows to conclude that only $\mathcal{O}(p^d)$ basis functions are active on each point. The total cost of the preprocessing step is therefore equal to $\mathcal{O}(np^d)$.
- After these preparations, we are ready to perform the matrix-vector product. In order to do so, we need to loop through all the $\mathcal{O}(n)$ basis functions and visit the $\mathcal{O}(p^d)$ collocation nodes in the support of the selected basis function. We conclude that *the total computational cost of the matrix-free approach via WICSP is equal to* $\mathcal{O}(np^d)$.
- The use of Gaussian quadrature in a matrix-free framework with THB-splines would lead to a higher cost of $\mathcal{O}(np^{2d})$ floating point operations, due to the fact that the total number of Gauss nodes depends on the degree of the discretization.

A detailed analysis reveals that the break-even point for WICSP vs. Gauss quadrature is attained for similar degrees as in the matrix assembly case, but the speedup factor of WICSP grows now much faster with p.

5.4. Summary

We have shown that the WICSP is particularly well suited for performing adaptive refinement using THB-splines:

- It supports the use of adaptive spline refinement, leading to better results with fewer degrees of freedom compared to the case of uniform refinement.
- It is beneficial for the fast assembly of the matrices arising in isogeometric discretizations. In this case the use of Galerkin based isogeometric analysis relies exclusively on Gauss quadrature routines to assemble the resulting system matrix. This typically leads to a high computational cost. On the contrary, our approach exhibits a considerably lower complexity equal to $\mathcal{O}(np^{2d})$. We have also performed different experiments to validate numerically how these advantages apply to discretizations with relatively low degrees.
- Finally, we have shown that our method gives rise to fast matrix-free solvers for THB-splines with computational complexity equal to $\mathcal{O}(np^d)$.

6. Conclusion

We have presented a novel weighted isogeometric collocation method based on spline projectors suitable for both tensor-product and THB-splines discretizations. In particular, we have observed how the resulting method gives rise to efficient solvers for isogeometric analysis and carried out numerous experiments to characterize its properties in terms of accuracy and complexity.

Regarding potential future work, we identify two areas of research that deserve further attention.

First, we plan to generalize our implementation to deal with more complex problems. We intend to extend our method to 3D, possibly multi-patch, geometries which are typically of interest in numerical simulation. The use of Neumann boundary conditions needs to be investigated as well. The case of knot vectors with internal knots that possess a multiplicity higher than one is not covered by the paper and may be a topic for future research. In fact, we expect that the advantages of our method we have observed in this paper carry over and become even more pronounced for higher dimensions. Moreover, we want to apply our approach to solve other PDEs, such as elasticity problems or other higher order equations. In both cases, isogeometric collocation has already been identified as a competitive approach [26, 34]. Finally, we intend to improve our code by exploring the possibility of a parallel implementation, which would further decrease the computing times.

Second, it is known that the theoretical analysis of convergence and stability properties of collocation methods is still an open problem, and this has not been addressed in this work. Additional research is required to fully understand these properties and confirm the observed experimental evidence. The equivalence of WICSP and dual Galerkin on the discretization space might be helpful in this respect.

Acknowledgements

The research was funded by the Austrian Science Fund (FWF): W1214-N15, project DK3. The support is gratefully acknowledged.

Appendix: (Truncated) Hierarchical B-splines

The definition of hierarchical B-splines on a bounded domain $\Omega^0 \subset \mathbb{R}^d$ is based on a finite sequence of d-variate *tensor product spline spaces* V^ℓ, $\ell = 0, \dots, N$, which are assumed to be nested, $V^\ell \subset V^{\ell+1}$, where the upper index ℓ specifies the *level*. The spline space of level ℓ is spanned by uniform *tensor-product B-splines* (TPB-splines) of degree p with knots $2^{-\ell}\mathbb{Z}$ in all

variables. The polynomial pieces of spline functions of level ℓ are obtained by considering their restrictions to the subsets of $2^{-\ell}(\mathbb{Z}^d + [0,1]^d)$, which we call *cells of level* ℓ.

Kraft [25] proposed to generate a multilevel spline basis by applying a selection procedure to the TPB-splines of all levels. His approach relies on a sequence of *subdomains* $\Omega^\ell \subset \mathbb{R}^d$, which are assumed to be inversely nested, $\Omega^\ell \supseteq \Omega^{\ell+1}$. We consider only level ℓ subdomains that are unions of finitely many (possibly overlapping) square blocks consisting of $\lceil \frac{p+1}{2} \rceil$ cells of level $\ell - 1$ per variable (also for $\ell = 0$). Cells of level ℓ are said to be *active* if they are contained in Ω^ℓ but not in $\Omega^{\ell+1}$. The union of active cells of level ℓ or higher covers the entire subdomain of level ℓ. The *mesh* is formed by all the active cells.

The basis of *hierarchical B-splines* (HB-splines) is obtained by collecting the TPB-splines of all levels ℓ, whose support is contained in Ω^ℓ and possesses at least one active cell of level ℓ. In addition, we assume *mesh grading* by requiring that the support of each level ℓ HB-spline contains cells of levels ℓ and $\ell + 1$ (but not higher) only.

HB-splines form a non-negative basis of the *hierarchical spline space*, which is spanned by them. The partition of unity is restored by the truncation mechanism [15]. Under the above assumption regarding mesh grading, one may find the *truncated hierarchical B-spline* (THB-spline) from a level ℓ HB-spline – which is called its *mother* – by considering its representation as the sum of $(p+2)^d$ suitably scaled TP-splines of level $\ell+1$, keeping only the contributions that take non-zero values on at least one active cell of level ℓ. The resulting THB-splines form another basis, which is a non-negative partition of unity, of the hierarchical spline space.

References

[1] Anitescu, C., Jia, Y., Zhang, Y., Rabczuk, T., 2015. An isogeometric collocation method using superconvergent points. Computer Methods in Applied Mechanics and Engineering 284, 1073 – 1097.

[2] Antolin, P., Buffa, A., Calabrò, F., Martinelli, M., Sangalli, G., 2015. Efficient matrix computation for tensor-product isogeometric analysis: The use of sum factorization. Computer Methods in Applied Mechanics and Engineering 285, 817 – 828.

[3] Auricchio, F., Beirão Da Veiga, L., Hughes, T., Reali, A., Sangalli, G., 2010. Isogeometric collocation methods. Mathematical Models and Methods in Applied Sciences 20, 2075 – 2107.

[4] Auricchio, F., Beirão da Veiga, L., Hughes, T., Reali, A., Sangalli, G., 2012. Isogeometric collocation for elastostatics and explicit dynamics. Computer Methods in Applied Mechanics and Engineering 249-252, 2–14.

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

[6] Bressan, A., Takacs, S., 2019. Sum factorization techniques in isogeometric analysis. Computer Methods in Applied Mechanics and Engineering 352, 437 – 460.

[7] Buffa, A., Giannelli, C., 2016. Adaptive isogeometric methods with hierarchical splines: Error estimator and convergence. Mathematical Models and Methods in Applied Sciences 26, 1–25.

[8] Calabrò, F., Sangalli, G., Tani, M., 2017. Fast formation of isogeometric Galerkin matrices by weighted quadratures. Computer Methods in Applied Mechanics and Engineering 316, 606 – 622.

[9] Casquero, H., Liu, L., Zhang, Y., Reali, A., Gomez, H., 2016. Isogeometric collocation using analysis-suitable T-splines of arbitrary degree. Computer Methods in Applied Mechanics and Engineering 301, 164–186.

[10] Demko, S., 1985. On the existence of interpolating projections onto spline spaces. Journal of Approximation Theory 43, 151–156.

[11] Donatelli, M., Garoni, C., Manni, C., Serra-Capizzano, S., Speleers, H., 2015. Robust and optimal multi-iterative techniques for IgA collocation linear systems. Computer Methods in Applied Mechanics and Engineering 284, 1120–1146.

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

[13] Foucher, F., Sablonnière, P., 2009. Quadratic spline quasi-interpolants and collocation methods. Mathematics and Computers in Simulations 79, 3455 – 3465.

[14] Giannelli, C., Jüttler, B., Kleiss, S.K., Mantzaflaris, A., Simeon, B., Špeh, J., 2016. THB-splines: An effective mathematical technology for adaptive refinement in geometric design and isogeometric analysis. Computer Methods in Applied Mechanics and Engineering 299, 337 – 365.

[15] Giannelli, C., Jüttler, B., Speleers, H., 2012. THB-splines: the truncated basis for hierarchical splines. Computer Aided Geometric Design 29, 485–498.

[16] Giannelli, C., Jüttler, B., Speleers, H., 2014. Strongly stable bases for adaptively refined multilevel spline spaces. Advances in Computational Mathematics 40, 459–490.

[17] Giust, A., Jüttler, B., Mantzaflaris, A., 2020. Local (T)HB-spline projectors via restricted hierarchical spline fitting. Computer Aided Geometric Design 80, 101865.

[18] Gomez, H., De Lorenzis, L., 2016. The variational collocation method. Computer Methods in Applied Mechanics and Engineering 309, 152–181.

[19] Gomez, H., Reali, A., Sangalli, G., 2014. Accurate, efficient, and (iso)geometrically flexible collocation methods for phase-field models. Journal of Computational Physics 262, 153–171.
[20] Hughes, T., Cottrell, J., Bazilevs, Y., 2005. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement. Computer Methods in Applied Mechanics and Engineering 194, 4135 – 4195.
[21] Hughes, T., Reali, A., Sangalli, G., 2010. Efficient quadrature for NURBS-based isogeometric analysis. Computer Methods in Applied Mechanics and Engineering 199, 301 – 313.
[22] Jia, Y., Anitescu, C., Zhang, Y., Rabczuk, T., 2019. An adaptive isogeometric analysis collocation method with a recovery-based error estimator. Computer Methods in Applied Mechanics and Engineering 345, 52–74.
[23] Johannessen, K.A., Kvamsdal, T., Dokken, T., 2014. Isogeometric analysis using LR B-splines. Computer Methods in Applied Mechanics and Engineering 269, 471–514.
[24] Kapl, M., Vitrih, V., 2020. Isogeometric collocation on planar multi-patch domains. Computer Methods in Applied Mechanics and Engineering 360, 112684.
[25] Kraft, R., 1998. Adaptive und linear unabhängige Multilevel B-Splines und ihre Anwendungen. Ph.D. thesis. Universität Stuttgart.
[26] Kruse, R., Nguyen-Thanh, N., De Lorenzis, L., Hughes, T., 2015. Isogeometric collocation for large deformation elasticity and frictional contact problems. Computer Methods in Applied Mechanics and Engineering 296, 73–112.
[27] Manni, C., Reali, A., Speleers, H., 2015. Isogeometric collocation methods with generalized B-splines. Computers and Mathematics with Applications 70, 1659–1675.
[28] Mantzaflaris, A., Giust, A., others (see website), 2020. G+Smo (Geometry plus Simulation modules) v0.8.1. github.com/gismo.
[29] Mokriš, D., Jüttler, B., Giannelli, C., 2014. On the completeness of hierarchical tensor-product B-splines. Journal of Computational and Applied Mathematics 271, 53 – 70.
[30] Montardini, M., Sangalli, G., Tamellini, L., 2017. Optimal-order isogeometric collocation at Galerkin superconvergent points. Computer Methods in Applied Mechanics and Engineering 316, 741 – 757.
[31] Nguyen-Thanh, N., Nguyen-Xuan, H., Bordas, S., Rabczuk, T., 2011. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. Computer Methods in Applied Mechanics and Engineering 200, 1892–1908.
[32] Pan, M., Jüttler, B., Giust, A., 2020. Fast formation of isogeometric Galerkin matrices via integration by interpolation and look-up. Computer Methods in Applied Mechanics and Engineering 113005.
[33] Pan, M., Jüttler, B., Mantzaflaris, A., 2021. Efficient matrix assembly in isogeometric analysis with hierarchical B-splines. Journal of Computational and Applied Mathematics 113278.
[34] Reali, A., Gomez, H., 2015. An isogeometric collocation approach for Bernoulli-Euler beams and Kirchhoff plates. Computer Methods in Applied Mechanics and Engineering 284, 623–636.
[35] Sablonnière, P., 2005. Recent progress on univariate and multivariate polynomial and spline quasi-interpolants, in: Mache, D.H., Szabados, J., de Bruin, M.G. (Eds.), Trends and Applications in Constructive Approximation, Birkhäuser, Basel. pp. 229–245.
[36] Sangalli, G., Tani, M., 2018. Matrix-free weighted quadrature for a computationally efficient isogeometric k-method. Computer Methods in Applied Mechanics and Engineering 338, 117 – 133.
[37] Schillinger, D., Dedè, L., Scott, M.A., Evans, J.A., Borden, M.J., Rank, E., Hughes, T.J., 2012. 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 249-252, 116–150.
[38] Schillinger, D., Evans, J., Reali, A., Scott, M., Hughes, T., 2013. Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations. Computer Methods in Applied Mechanics and Engineering 267, 170 – 232.
[39] Speleers, H., 2017. Hierarchical spline spaces: quasi-interpolants and local approximation estimates. Advances in Computational Mathematics 43, 235–255.
[40] Speleers, H., Manni, C., 2016. Effortless quasi-interpolation in hierarchical spaces. Numerische Mathematik 132, 155–184.
[41] Vuong, A.V., Giannelli, C., Jüttler, B., Simeon, B., 2011. A hierarchical approach to adaptive local refinement in isogeometric analysis. Computer Methods in Applied Mechanics and Engineering 200, 3554–3567.
[42] Wang, D., Qi, D., Li, X., 2021. Superconvergent isogeometric collocation method with Greville points. Computer Methods in Applied Mechanics and Engineering 377, 113689.
[43] Wang, P., Xu, J., Deng, J., Chen, F., 2011. Adaptive isogeometric analysis using rational PHT-splines. Computer-Aided Design 43, 1438–1448.