Multigrid Methods for Isogeometric Analysis with THB-Splines

Clemens Hofreither[a], Bert Jüttler[b], Gábor Kiss[c], Walter Zulehner[a]

[a] *Institute of Computational Mathematics*
Johannes Kepler University Linz, Altenberger Str. 69, 4040 Linz, Austria
[b] *Institute of Applied Geometry*
Johannes Kepler University Linz, Altenberger Str. 69, 4040 Linz, Austria
[c] *Doctoral Program "Computational Mathematics"*
Johannes Kepler University Linz, Altenberger Str. 69, 4040 Linz, Austria

Abstract

We propose a geometric multigrid algorithm for the solution of the linear systems which arise when using hierarchical spline spaces for isogeometric discretizations of elliptic partial differential equations. In particular, we focus on the use of hierarchical B-spline (HB) and truncated hierarchical B-spline (THB) bases to implement adaptively refined isogeometric discretizations. We describe an approach for constructing the nested sequences of hierarchical spline spaces which are required for our geometric multigrid solver. Furthermore, we give an algorithm for the computation of the so-called prolongation matrices which map coarser-space functions to their finer-space representations for both HB- and THB-spline bases.

In several numerical experiments, we study two-dimensional Poisson problems with singularities and employ a posteriori error estimators for the adaptive refinement of the hierarchical spline spaces. Using standard multigrid smoothers, we observe that using THB-spline bases consistently confers a dramatic advantage in terms of iteration numbers over the use of HB-spline bases, in particular for higher spline degrees. When used as a preconditioner for Conjugate Gradient iteration, the proposed multigrid method exhibits practical iteration numbers for two-dimensional problems up to the tested case of fourth-degree splines.

Keywords: isogeometric analysis, multigrid, hierarchical splines, truncated hierarchical B-splines, adaptive refinement

1. Introduction

Isogeometric analysis (IgA), which was introduced by Hughes [21], is a novel framework for numerical simulation that aims at the close integration of geometry representations from Computer Aided Geometric Design (CAD) into the analysis process, thereby avoiding the otherwise necessary and expensive data exchange between the finite element analysis (FEA) and CAD software. However, the tensor product B-spline and NURBS spaces which are most commonly used in geometric design and IgA do not provide the possibility of local refinement. Due to their construction, any refinement propagates over the entire domain. To address this issue, other spline spaces which permit local adaptive refinement have been utilized in IgA. These include T-splines [1, 10, 29], hierarchical B-splines [2, 27, 28, 31], LR (locally refined) B-splines in [22] and polynomial splines over hierarchical T-meshes [26, 32].

In the present work, we focus on the hierarchical approach to the adaptive refinement of B-splines, which can be traced back to [12]. The key idea of the construction of the hierarchical basis, which was introduced by Kraft [25], consists in a suitable selection of standard tensor-product B-spline functions from a sequence of nested B-spline bases, guided by a sequence of nested subdomains.

Email addresses: chofreither@numa.uni-linz.ac.at (Clemens Hofreither), bert.juettler@jku.at (Bert Jüttler), gabor.kiss@dk-compmath.jku.at (Gábor Kiss), zulehner@numa.uni-linz.ac.at (Walter Zulehner)

June 19, 2017

The hierarchical B-spline (HB-spline) basis inherits several desirable properties of B-splines such as linear independence, local support and non-negativity. However, it does not possess the partition of unity property and is only weakly stable with respect to the supremum norm.

To overcome these drawbacks, the *truncated hierarchical B-spline* (THB-spline) basis has been recently introduced in [15]. The additional truncation step, which is applied during the construction of the THB-spline basis, ensures that the basis possesses the partition of unity property. In addition, it also improves the stability properties of the basis, see [16], and significantly reduces the overlap between the basis functions from different refinement levels.

For the efficient solution of the large, sparse linear systems arising in isogeometric discretizations, multigrid methods are an attractive approach. Multigrid solvers for uniformly refined tensor product spline spaces have been studied by several researchers. In particular, geometric multigrid has been considered in [13, 20, 18, 19], and an additive multilevel scheme was proposed in [5]. A symbol-based approach has been taken in [7, 8, 9]. Finally, a local refinement strategy based on full multigrid ideas has been developed in [6].

In the present work, we investigate geometric multigrid solvers for isogeometric discretizations based on adaptively refined hierarchical spline spaces. In particular, we describe how to construct nested sequences of hierarchical spline spaces starting from a coarse tensor product spline space, or from an initial coarse basis used to describe the geometry which may already be hierarchical. We also address the topic of how to compute the necessary prolongation matrices between nested hierarchical spline spaces in both the HB- and the THB-spline bases.

As the smoother for the multigrid algorithm, we use a standard Gauss-Seidel method throughout. It is known by now that classical multigrid smoothers such as Gauss-Seidel, when used in the IgA setting, do not yield methods which are robust in the spline degree [19]. The development of p-robust multigrid methods for IgA is an active research topic even in the case of tensor product bases. The most promising approaches thus far appear to be mass-based smoothers ([20] and a forthcoming publication) as well as symbol-based methods [7, 8, 9]. However, neither of these can be applied to the case of hierarchical spline spaces in a straightforward way. This justifies us in using a classical multigrid smoother in the present work.

We perform several numerical examples and compare the performance of V-cycle multigrid for discretizations based on HB-spline bases and on THB-spline bases. Our numerical experiments indicate that using THB-spline bases leads to significantly reduced iteration numbers in the multigrid algorithm compared to using HB-spline bases, especially for higher spline degrees. In order to mitigate the high iteration numbers which occur for higher spline degrees in some cases due to the use of a Gauss-Seidel smoother, we also propose a global Conjugate Gradient (CG) iteration preconditioned with one V-cycle. In many cases, this significantly reduces the overall iteration numbers and leads to a practical method.

The remainder of the paper is structured as follows. In Section 2, we recall some preliminaries on hierarchical spline spaces, (T)HB-spline bases, and IgA. In Section 3, we describe the construction of nested hierarchical spline spaces and a multigrid algorithm over such nested spaces. In Section 4, we give several numerical experiments for adaptively refined discretizations of the Poisson equation. In Section 5, we end with some concluding remarks.

2. Preliminaries

2.1. Bases for hierarchical spline spaces

We consider a sequence

$$S^0 \subset S^1 \subset \ldots \subset S^{\Lambda-1} \subset S^\Lambda$$

of nested tensor-product spline spaces of degree $\mathbf{p} = (p_1, \ldots, p_d)$ over the parameter domain $D = (0,1)^d$, where every space $S^\lambda, \lambda = 0, \ldots, \Lambda$ is spanned by normalized B-splines $\mathcal{B}^\lambda$ and the positive integer Λ specifies the number of levels. (We use Greek characters for the levels in the spline hierarchy in order to keep the Latin ones for the multigrid levels.) Additionally let

$$\mathbf{\Omega} = (\Omega^\lambda)_{\lambda \in \mathbb{N}_0}$$

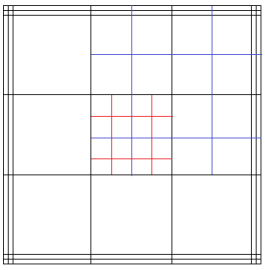


Figure 1: Subdomain structure $\mathbf{\Omega}$ with 3 hierarchical levels $\Omega_0, \Omega_1, \Omega_2$ highlighted by black, blue and red, respectively.

be a sequence of inversely nested subdomains satisfying

$$D = \Omega^0 \supseteq \Omega^1 \supseteq \ldots \supseteq \Omega^\Lambda = \emptyset. \tag{1}$$

The first subdomain Ω^0 is the entire parameter domain D. The higher level subdomains represent the regions that are selected for refinement to the level λ.

Each tensor-product spline space S^λ is defined by d knot vectors, one for each coordinate direction in the parameter space. These knot vectors are also nested, since this implies the required inclusions of the spline spaces. The knot hyperplanes (knot lines for $d = 2$) determine a segmentation of the domain into cells, and we assume that each subdomain is a collection of cells of the same level. Figure 1 shows such a hierarchy in the bivariate case for degree $\mathbf{p} = (2, 2)$, as indicated by the three-fold boundary knots, with three levels of refinement.

The hierarchical spline space induced by the sequence $\mathbf{\Omega}$ is then given by

$$\mathcal{H}(\mathbf{\Omega}) := \text{span} \bigcup_{\lambda=0,\ldots,\Lambda-1} \{\beta^\lambda \in \mathcal{B}^\lambda : \text{supp}\beta^\lambda \subseteq \Omega^\lambda\}.$$

The hierarchical basis $\mathcal{K}$, which has been introduced by Kraft [25], with respect to a given subdomain hierarchy $\mathbf{\Omega}$ is obtained by selecting the *active* B-splines from all levels. A basis function $\beta^\lambda \in \mathcal{B}^\lambda$ is said to be active if

$$\Omega^{\lambda+1} \not\supseteq \text{supp}\beta^\lambda \subseteq \Omega^\lambda,$$

thus

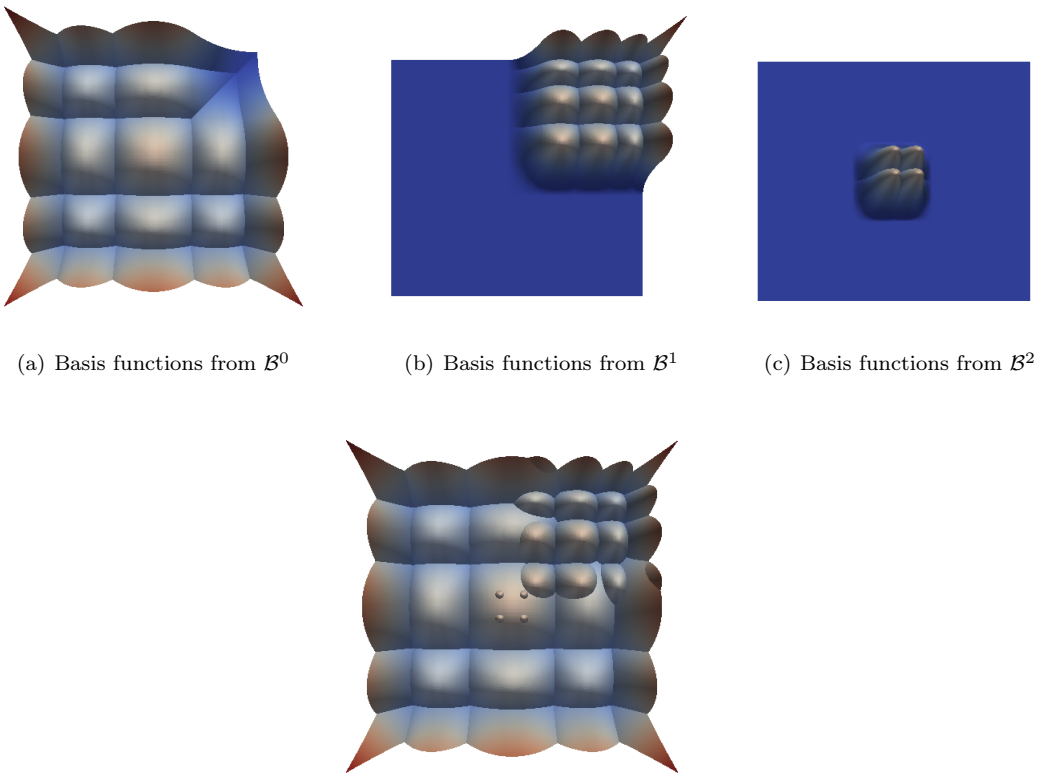
$$\mathcal{K} = \bigcup_{\lambda=0,\ldots,\Lambda-1} \{\beta^\lambda \in \mathcal{B}^\lambda : \Omega^{\lambda+1} \not\supseteq \text{supp}\beta^\lambda \subseteq \Omega^\lambda\}.$$

We have $\mathcal{H}(\mathbf{\Omega}) = \text{span}(\mathcal{K})$. The basis functions collected in $\mathcal{K}$ will be denoted as HB-splines. Figure 2 shows the construction of the HB-splines for the domain hierarchy shown in Figure 1.

The HB-splines inherit some of the main properties of the B-splines such as non-negativity, local support and linear independence, as shown in [31]. The partition of unity property and the convex hull property, however, are not preserved during the construction. Nevertheless, they may be recovered, under certain assumptions on the subdomain hierarchy, by a suitable scaling of the basis functions, see [31].

A different approach to restore these properties which does not require any assumptions about the subdomain hierarchy consists in using the THB-splines. In addition, this construction decreases the support overlaps between basis functions from different levels, which will be beneficial for using the multigrid framework, as observed below.

The construction of the THB-splines $\mathcal{T}$ is based on the refinability of B-splines. In fact, any B-spline $\beta \in \mathcal{B}^\lambda$ of level λ can be represented as a linear combination of the B-splines of level $\lambda + 1$ with supports contained in $\text{supp}\,\beta$, using only non-negative coefficients. Consequently, if a function β of level λ is not active since it satisfies $\text{supp}\,\beta \subseteq \Omega^{\lambda+1}$, then $\mathcal{K}$ contains functions of higher levels that allow to represent it.



(a) Basis functions from $\mathcal{B}^0$ (b) Basis functions from $\mathcal{B}^1$ (c) Basis functions from $\mathcal{B}^2$

(d) Hierarchical basis

Figure 2: The contribution of basis functions from levels 0, 1 and 2 to the hierarchical basis $\mathcal{K}$ defined over the subdomain structure shown in Figure 1.

The truncation of a function $\tau \in S^\lambda$ is defined with the help of its representation

$$\tau = \sum_{\beta \in \mathcal{B}^{\lambda+1}} c_\beta^{\lambda+1}(\tau)\beta, \qquad c_\beta^{\lambda+1} \in \mathbb{R}.$$

with respect to the basis $\mathcal{B}^{\lambda+1}$. In order to truncate the function, we simply eliminate the contributions of all basis functions that are selected at the higher level

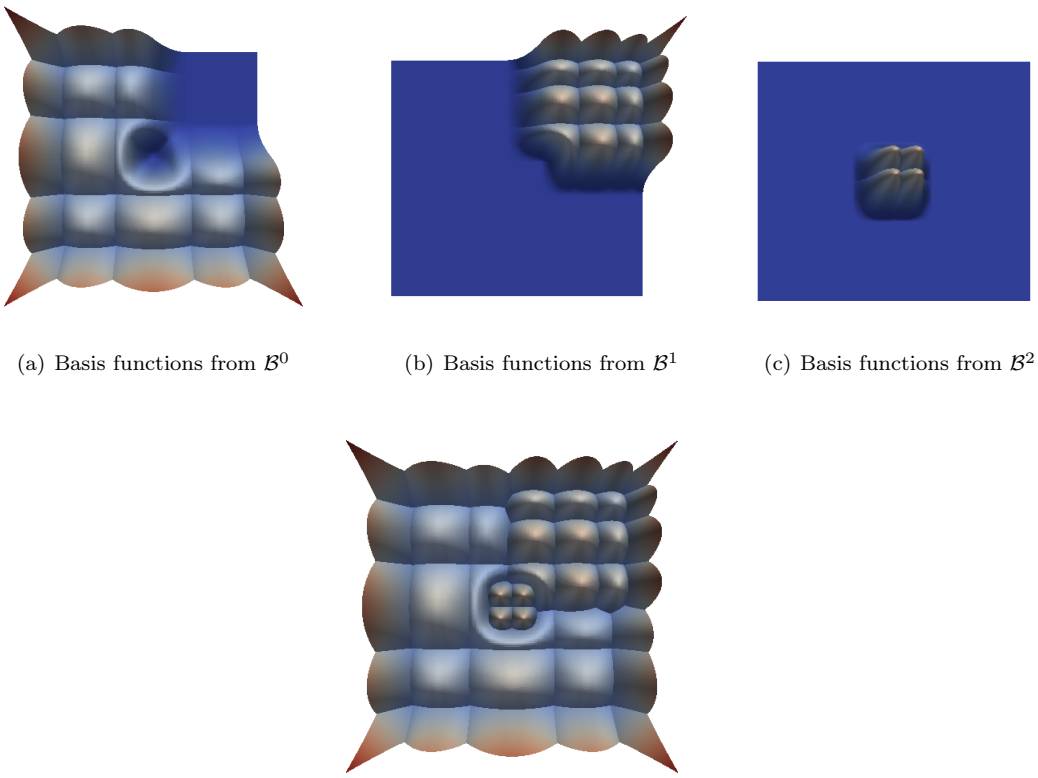
$$\operatorname{trunc}^{\lambda+1}\tau = \sum_{\beta \in \mathcal{B}^{\lambda+1}, \operatorname{supp}\beta \not\subseteq \Omega^{\lambda+1}} c_\beta^{\lambda+1}(\tau)\,\beta.$$

It may happen that an omitted function $\beta \in \mathcal{B}^{\lambda+1}$ is not active, i.e., not contained in $\mathcal{K}$. This is the case whenever $\operatorname{supp}\beta \subseteq \Omega^{\lambda+2}$. Nevertheless, this function is still contained in the span of all active functions due to the refinability of B-splines.

We define the THB-splines $\mathcal{T}$ by applying the truncation repeatedly to all functions in $\mathcal{K}$,

$$\mathcal{T} = \bigcup_{\lambda=0,\dots,\Lambda-1} \{\operatorname{trunc}^{\Lambda-1}(\operatorname{trunc}^{\Lambda-2}\dots(\operatorname{trunc}^{\lambda+1}\beta^\lambda)\dots) : \beta^\lambda \in \mathcal{K} \cap \mathcal{B}^\lambda\}.$$

In addition to preserving the properties of non-negativity, local support and linear independence, the THB-splines also form a partition of unity [15]. Moreover, the *hierarchical spline spaces* spanned by $\mathcal{K}$ and $\mathcal{T}$ are



(a) Basis functions from $\mathcal{B}^0$ (b) Basis functions from $\mathcal{B}^1$ (c) Basis functions from $\mathcal{B}^2$

(d) Hierarchical basis

Figure 3: The contribution of basis functions from levels 0, 1 and 2 to the THB-splines $\mathcal{T}$ defined over the subdomain structure shown in Figure 1. The effect of truncation is clearly visible in the central area of the domain.

identical,

$$\operatorname{span} \mathcal{K} = \operatorname{span} \mathcal{T} = \mathcal{H}(\mathbf{\Omega}). \tag{2}$$

Figure 3 shows the THB-splines for the domain hierarchy in Figure 1, which is another basis for the space spanned by the HB-splines in Figure 2.

2.2. Isogeometric analysis

Let

$$G : D \to G(D) \subset \mathbb{R}^d$$

denote a bijective geometry mapping which maps the parameter domain D to the computational domain $P = G(D)$ on which we wish to solve a boundary value problem. In IgA, this mapping is typically given in terms of spline basis functions, $G \in (S^0)^d$, but its concrete form is irrelevant to our discussion here.

Let $\mathcal{V}_h \subset H_0^1(G(D))$ denote a finite-dimensional space over the computational domain. In IgA, this discretization space is obtained by first fixing a family of basis functions $\{\psi_i : D \to \mathbb{R}\}$ over the parameter domain D and then mapping them into the computational domain via $\varphi_i(x) := \psi_i(G^{-1}(x))$. Originally, Hughes et al. [21] proposed tensor product B-spline and NURBS bases for $\{\psi_i\}$, and these are still the most popular choices when uniform refinement of the discretization space is desired. In the present work, we instead use HB-spline and THB-spline bases as described in Section 2.1 in order to facilitate local and

adaptive refinement of the discretization space. Thus, we obtain the isogeometric discretization space based on hierarchical splines,

$$\mathcal{V}_h = \{u \circ G^{-1} : u \in \mathcal{H}(\mathbf{\Omega})\} = \operatorname{span}\{\mathcal{K} \circ G^{-1}\} = \operatorname{span}\{\mathcal{T} \circ G^{-1}\}. \tag{3}$$

We now consider an IgA discretization of the Poisson equation with pure Dirichlet boundary conditions: find $u_h \in \mathcal{V}_h$ such that

$$a(u_h, v_h) = \langle F, v_h \rangle \qquad \forall v_h \in \mathcal{V}_h \tag{4}$$

with the bilinear form and linear functions, respectively,

$$a(u, v) = \int_P \nabla u \cdot \nabla v \, dx, \qquad \langle F, v \rangle = \int_P f v \, dx - a(g, v).$$

Here $g \in H^1(\Omega)$ is a suitable extension of the given Dirichlet data. This discretization results in a large, sparse, symmetric positive definite linear system. Our aim is to investigate geometric multigrid solvers for this system.

3. Multigrid Methods for Adaptively Refined Discretizations

We recall the abstract framework of multigrid methods and describe its realization for hierarchical spline spaces in the isogeometric framework. Special attention is paid to the construction of the prolongation matrices and the creation of the hierarchy by adaptive refinement.

3.1. An abstract multigrid framework

We recall the construction of a geometric multigrid solver in an abstract setting. We provide only an outline here and refer to the literature [4, 17, 30] for further reading. In later sections, we will flesh out the components of the multigrid solver for our particular setting.

Let $\mathcal{V}_0 \subset H_0^1(G(D))$ denote a coarse discretization space over our computational domain. By a suitable refinement process, we obtain a sequence of nested spaces

$$\mathcal{V}_0 \subset \mathcal{V}_1 \subset \mathcal{V}_2 \subset \ldots \subset \mathcal{V}_L. \tag{5}$$

On each level $\ell \in \{0, \ldots, L\}$, we denote the number of degrees of freedom by $n_\ell = \dim \mathcal{V}_\ell$. Given a variational formulation of a boundary value problem, we derive a Galerkin discretization in $\mathcal{V}_\ell$, resulting in a linear system

$$A_\ell u_\ell = f_\ell,$$

where $A_\ell \in \mathbb{R}^{n_\ell \times n_\ell}$ is the stiffness matrix, $f_\ell \in \mathbb{R}^{n_\ell}$ is the load vector, and $u_\ell \in \mathbb{R}^{n_\ell}$ is the coefficient vector of the discrete solution with respect to a chosen basis for $\mathcal{V}_\ell$.

Our aim is to solve the fine-space equation ($\ell = L$) in an efficient manner. The core idea of multigrid is to use an iterative procedure which efficiently reduces (or "smoothes") the high-frequency components of the error, while the low-frequency components of the error are treated by projecting the residual equation into a coarser space. Some of the low-frequency components in the fine space turn into high-frequency components when viewed in the coarser space and can again be efficiently smoothed there. This idea is applied recursively all the way to the coarsest level $\mathcal{V}_0$, where the linear system is small enough to be solved by a direct method.

Since the spaces are nested, an arbitrary function in $\mathcal{V}_\ell$ can always be represented on any finer level $\mathcal{V}_k$, $k > \ell$. Let $P_\ell \in \mathbb{R}^{n_\ell \times n_{\ell-1}}$ denote the *prolongation matrix* which maps the coefficients of a function represented in the basis of $\mathcal{V}_{\ell-1}$ to the coefficients of the same function represented in the basis of $\mathcal{V}_\ell$. In other words, P_ℓ realizes the canonical embedding

$$\mathcal{V}_{\ell-1} \to \mathcal{V}_\ell : \quad v \mapsto v$$

with respect to the particular choice of bases. The transpose of the prolongation matrix $P_\ell^\top$ is called the *restriction matrix* and maps from the fine to the coarse space. It is easy to show that, when using Galerkin discretizations, the coarse-space stiffness matrix can be obtained via $A_{\ell-1} = P_\ell^\top A_\ell P_\ell$.

As an important ingredient in the multigrid algorithm, we need so-called *smoothers*. On each level except the coarsest, $\ell \in \{1, \dots, L\}$, we choose a smoother $S_\ell \in \mathbb{R}^{n_\ell \times n_\ell}$. This matrix must be easier to invert than the stiffness matrix A_ℓ. Standard choices are the Jacobi smoother, where S_ℓ is chosen as a scalar multiple of the diagonal of A_ℓ, and the Gauss-Seidel smoother, where S_ℓ is chosen as the lower triangular part of A_ℓ. We then define the smoothing operator

$$\mathcal{S}_\ell(v, b) := v + S_\ell^{-1}(b - A_\ell v)$$

for arbitrary vectors and right-hand sides $v, b \in \mathbb{R}^{n_\ell}$. Analogously, we define the transposed smoothing operator $\mathcal{S}_\ell^\top(v, b)$ where the matrix S_ℓ is replaced by its transpose $S_\ell^\top$ in the definition.

The iterative procedure $v_j \mapsto v_{j+1} := \mathcal{S}_\ell(v_j, b)$ must converge to the solution of the linear system $A_\ell v = b$. Typically, it does so very slowly. However, to obtain an efficient multigrid algorithm it is only required that the smoothing iteration efficiently reduces the components of the error which can not be represented in the coarser space $\mathcal{V}_{\ell-1}$, i.e., the high-frequency error.

We now have the components in place to define the multigrid algorithm in a recursive fashion. On the coarsest level, we define the multigrid operator as the exact solution of the coarse-space problem, i.e.,

$$\mathcal{MG}_0(v, b) := A_0^{-1} b.$$

On every level $\ell > 0$, we define the V-cycle multigrid operator $\mathcal{MG}_\ell(v, b)$ by the sequence of operations

$$\begin{aligned} v^{(1)} &:= \mathcal{S}_\ell(v, b), \\ v^{(2)} &:= v^{(1)} + P_\ell \, \mathcal{MG}_{\ell-1}(0, P_\ell^\top (b - A_\ell v^{(1)})), \\ \mathcal{MG}_\ell(v, b) := v^{(3)} &:= \mathcal{S}_\ell^\top (v^{(2)}, b). \end{aligned}$$

These three steps are called pre-smoothing, coarse-grid correction, and post-smoothing, respectively. It is possible to apply several steps of the smoothing operator $\mathcal{S}_\ell$ in sequence, however in this work we always use exactly one pre- and one post-smoothing step.

The V-cycle iteration for the fine-grid problem is given, for an arbitrary initial guess $u_0 \in \mathbb{R}^{n_L}$, by the procedure

$$u_{j+1} := \mathcal{MG}_L(u_j, f_L) \qquad \forall j = 0, 1, 2, \dots$$

It is called the V-cycle because in each iteration, the method starts with some fine-level approximation $v_j \in \mathbb{R}^{n_L}$, proceeds by smoothing and restriction through all levels to the coarsest space, and then by prolongation and smoothing back to the finest space, obtaining a new approximation $v_{j+1} \in \mathbb{R}^{n_L}$. This pattern can be visualized as a V-shape as in Figure 4.

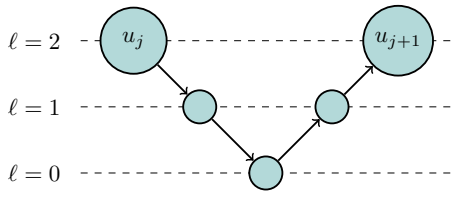


Figure 4: Visualization of one iteration of the multigrid V-cycle algorithm with three nested spaces. Circles correspond to smoothing operations or, on the coarsest level $\ell = 0$, the exact solve. Downward arrows correspond to restriction and upward arrows to prolongation operations.

3.2. Nested sequences of hierarchical spline spaces

The sequence of nested spaces (5) will be generated by performing an isogeometric discretization (with a given geometry mapping) based on nested hierarchical spline spaces (2), which are spanned by (T)HB-splines. More precisely, we use *several* instances of the hierarchical spline spaces $\mathcal{H}(\mathbf{\Omega})$ as in (2) which are constructed in a way which ensures that they are nested. The following lemma, which has been derived in [31], will be essential for this construction.

Lemma 1 ([31])**.** *We consider the two sequences* $\mathbf{\Omega}$ *and* $\hat{\mathbf{\Omega}}$ *of nested subdomains*

$$\Omega^0 \supseteq \Omega^1 \supseteq \ldots \supseteq \Omega^\Lambda \quad \text{and} \quad \hat{\Omega}^0 \supseteq \hat{\Omega}^1 \supseteq \ldots \supseteq \hat{\Omega}^\Lambda$$

and the hierarchical spline spaces $\mathcal{H}(\mathbf{\Omega})$ *and* $\mathcal{H}(\hat{\mathbf{\Omega}})$ *defined by them. If* $\Omega^\lambda \subseteq \hat{\Omega}^\lambda$ *for every* $\lambda = 0, \ldots, \Lambda$*, then*

$$\mathcal{H}(\mathbf{\Omega}) \subseteq \mathcal{H}(\hat{\mathbf{\Omega}}).$$

In other words, enlarging all subdomains of a given subdomain hierarchy $\mathbf{\Omega}$ while preserving the property that they are nested leads to a superspace of the hierarchical spline space. Clearly, this result is inherited by the spaces used for the isogeometric discretizations (3), since these spaces are obtained simply by composing the hierarchical spline spaces with the inverse of the geometry mappings.

In order to use this result in the multigrid framework, we consider $L+1$ sequences $\mathbf{\Omega}_\ell$ of subdomains,

$$\mathbf{\Omega}_\ell = (\Omega_\ell^\lambda)_{\lambda \in \mathbb{N}_0} \qquad \text{for } \ell = 0, \ldots, L,$$

where each of them satisfies the assumption (1). Each sequence $\mathbf{\Omega}_\ell$ induces a hierarchical spline space $\mathcal{H}(\mathbf{\Omega}_\ell)$ as defined in Section 2.1, as well as a set of HB-spline and THB-spline basis functions, which we denote by $\mathcal{K}_\ell$ and $\mathcal{T}_\ell$, respectively. The idea is now to choose the multigrid spaces as hierarchical spline spaces, transformed by the geometry mapping as in Section 2.2,

$$\mathcal{V}_\ell := \{u \circ G^{-1} : u \in \mathcal{H}(\mathbf{\Omega}_\ell)\} = \operatorname{span}\{\mathcal{K}_\ell \circ G^{-1}\} = \operatorname{span}\{\mathcal{T}_\ell \circ G^{-1}\} \qquad \text{for } \ell = 0, \ldots, L.$$

Under the additional assumption

$$\Omega_{\ell-1}^\lambda \subseteq \Omega_\ell^\lambda \qquad \forall \lambda \in \mathbb{N}_0,\ \ell \in \mathbb{N}, \tag{6}$$

Lemma 1 yields the nestedness of the multigrid spaces,

$$\mathcal{V}_0 \subseteq \ldots \subseteq \mathcal{V}_\ell \subseteq \ldots \subseteq \mathcal{V}_L,$$

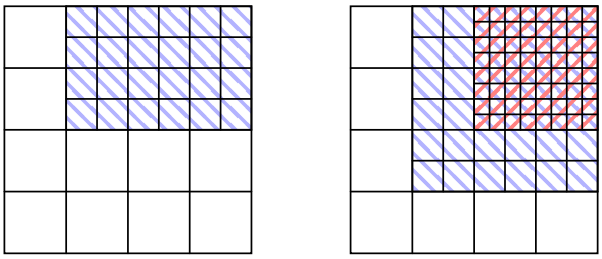
which we have assumed in the construction of our multigrid method in Section 3.1. Figure 5 illustrates an example of two subdomain hierarchies that satisfy the above conditions and therefore induce two nested hierarchical spline spaces.

Combining assumption (6) with the fact that the subdomains within each hierarchy are also nested (1), we can sketch the following diagram for the relationships between the subdomains:

$$\begin{array}{lccccccc}
D = & \Omega_0^0 & \supseteq \ldots \supseteq & \Omega_0^\lambda & \supseteq & \Omega_0^{\lambda+1} & \supseteq \ldots \supseteq & \Omega_0^\Lambda & = \emptyset \\
& \| & & \cap| & & \cap| & & \| & \\
& \vdots & & \vdots & & \vdots & & \vdots & \\
& \| & & \cap| & & \cap| & & \| & \\
D = & \Omega_\ell^0 & \supseteq \ldots \supseteq & \Omega_\ell^\lambda & \supseteq & \Omega_\ell^{\lambda+1} & \supseteq \ldots \supseteq & \Omega_\ell^\Lambda & = \emptyset \\
& \| & & \cap| & & \cap| & & \| & \\
& \vdots & & \vdots & & \vdots & & \vdots & \\
& \| & & \cap| & & \cap| & & \| & \\
D = & \Omega_L^0 & \supseteq \ldots \supseteq & \Omega_L^\lambda & \supseteq & \Omega_L^{\lambda+1} & \supseteq \ldots \supseteq & \Omega_L^\Lambda & = \emptyset
\end{array}$$

Each row of this diagram corresponds to a hierarchical spline space and therefore to a multigrid level $\mathcal{V}_\ell$. Each column corresponds to a level λ of the underlying hierarchical spline space construction.

Several comments on this construction of nested hierarchical spline spaces are in order:



$\boldsymbol{\Omega}_1 = (D, \Omega_1^1)$ $\boldsymbol{\Omega}_2 = (D, \Omega_2^1, \Omega_2^2)$

Figure 5: Two subdomain hierarchies over $D = (0,1)^2$ which satisfy (1) and (6). The subdomains Ω_ℓ^1 are shown in blue, and Ω_2^2 is shown in red. Due to (1), each hierarchy $\boldsymbol{\Omega}_\ell$ induces a hierarchical spline space $\mathcal{H}(\boldsymbol{\Omega}_\ell)$ as defined in Section 2.1. Due to (6), the hierarchical spline spaces are nested, $\mathcal{H}(\boldsymbol{\Omega}_1) \subset \mathcal{H}(\boldsymbol{\Omega}_2)$.

- In the standard situation, the sequences $\boldsymbol{\Omega}_\ell$ are constructed by adaptively refining a given uniform discretization consisting of only one level, see Section 3.4. Typically this means that one refinement level is added in every step. Thus we have $\Omega_\ell^\lambda = \emptyset$ whenever $\lambda > \ell$, and the diagram shown above becomes lower triangular.

  Note that each refinement step $\ell - 1 \to \ell$ not only adds an extra level, but may also enlarge the previously introduced subdomains, hence the sets in (6) are generally not equal for $\lambda < \ell$. This is analogous to the situation in standard adaptive finite element methods where an element that has been refined in an earlier step still may be further refined in a later step. To illustrate this, we refer the reader forward to Figure 8, where the meshes for successive multigrid levels ℓ in an actual example are shown. It can be clearly observed that each new multigrid level ℓ not only introduces a new, finer spline level $\lambda = \ell$, but also enlarges refinement areas for levels $\lambda < \ell$, i.e., $\Omega_{\ell-1}^\lambda \subseteq \Omega_\ell^\lambda$. The construction described above ensures that $\mathcal{V}_{\ell-1} \subseteq \mathcal{V}_\ell$.

- The construction of the multigrid hierarchy also supports geometry mappings which use hierarchical spline spaces, such as the mappings constructed in [11]. In this case, even the coarsest domain hierarchy may be fully populated, i.e., all subdomains are possibly non-empty, except for Ω_0^Λ.

- An alternative multigrid hierarchy can be constructed by considering only the subdomain sequence for the largest level L, i.e., the last row in the diagram. HB-splines $\mathcal{K}_L^\mu$ (and similarly THB-splines $\mathcal{T}_L^\mu$) spanning nested discretization spaces are obtained simply by replacing each subdomain Ω_L^λ with the empty set for $\lambda > \mu$, again producing a triangular diagram. Except for the discretization at the coarsest level, which is the same by definition, this approach gives larger dimensions of the discretization spaces in the multigrid hierarchy. Moreover, it does not comply with the isoparametric principle at all levels if the geometry mapping uses hierarchical splines.

3.3. Computation of the prolongation matrix

In order to describe the computation of the prolongation matrix, which is required for the multigrid solver, we assume that the B-splines within each basis $\mathcal{B}^\lambda$ are identified by indices, $\mathcal{B}^\lambda = \{\beta_i^\lambda \,:\, 1 \le i \le m^\lambda\}$, where m^λ is the number of tensor-product B-splines of level λ in the spline hierarchy. Given a fixed level ℓ of the multigrid hierarchy, the active B-splines (which are selected for inclusion in $\mathcal{K}_\ell$) are identified by indicator functions

$$\chi_\ell^\lambda(i) = \begin{cases} 1 & \text{if } \beta_i^\lambda \text{ is active in } \mathcal{K}_\ell, \\ 0 & \text{otherwise,} \end{cases} \qquad \lambda = 0, \dots, \Lambda.$$

These functions are represented by using a sparse data structure in our implementation, cf. [23]. We also use them to create an index set for the basis functions in $\mathcal{K}_\ell$ and $\mathcal{T}_\ell$,

$$\mathcal{I}_\ell = \bigcup_{\lambda=0}^{\Lambda} \{\lambda\} \times \mathcal{I}_\ell^\lambda, \qquad \mathcal{I}_\ell^\lambda = \{i : \chi_\ell^\lambda(i) = 1, 1 \le i \le m^\lambda\}, \tag{7}$$

where $\mathcal{I}_\ell^\lambda$ is the index set of active B-spline functions from the hierarchical level λ. The overall index set $\mathcal{I}_\ell$ of the multigrid hierarchy ℓ is sorted lexicographically with respect to λ and i.

Given two spline hierarchies for adjacent levels $\ell-1$ and ℓ in the multigrid hierarchy, the algorithm PROLONG computes the associated prolongation matrix for $\mathcal{K}_{\ell-1}$ and $\mathcal{T}_{\ell-1}$:

Algorithm PROLONG($\mathcal{I}_{\ell-1}$, $\mathcal{I}_\ell$, $(R^\lambda)_{\lambda=1,\dots,\Lambda}$) {

// $\mathcal{I}_{\ell-1}$ and $\mathcal{I}_\ell$: index sets of active basis functions of level $\ell-1$ and ℓ in the multigrid hierarchy, respectively. The index sets for each level λ in the spline hierarchy are $\mathcal{I}_{\ell-1}^\lambda$ and $\mathcal{I}_\ell^\lambda$, see (7)
// $(R^\lambda)_{\lambda=1,\dots,\Lambda}$: prolongation matrices between $\mathcal{B}^{\lambda-1}$ and $\mathcal{B}^\lambda$, precomputed using knot insertion.
// The algorithm returns the prolongation matrix P_ℓ.

create null matrix P_ℓ of size $|\mathcal{I}_{\ell-1}| \times |\mathcal{I}_\ell|$

for all $(\lambda, i) \in \mathcal{I}_{\ell-1}$ do {

create null vector M^λ of size m^λ
$M^\lambda[i] := 1$
for ν from λ to $N-1$ do {
for all $k \in \mathcal{I}_\ell^\nu$ do {
if $\nu > \lambda$ { // this line THB-splines only
for all $k \in \mathcal{I}_{\ell-1}^\nu$ do { // this line THB-splines only
$M^\nu[k] := 0$ } } // this line THB-splines only
$P_\ell[(\lambda, i), (\nu, k)] := M^\nu[k]$
$M^\nu[k] := 0$ // this line HB-splines only
}
$M^{\nu+1} := R^{\nu+1} M^\nu$ } }

RETURN(P_ℓ) }

For HB-splines, the algorithm iteratively computes the representation of each B-spline in $\mathcal{K}_{\ell-1}$ with respect to the new basis $\mathcal{K}_\ell$. This is performed by visiting all levels ν of the spline hierarchy. In each level we eliminate the contributions of the functions which are present at the next level ℓ of the multigrid hierarchy. Subsequently, the remainder is represented with respect to the next level $\nu+1$ of the spline hierarchy. Clearly, the iteration can be aborted if all coefficients in $M^{\nu+1}$ are zero.

The computation for THB-splines is based on the preservation of coefficients, see [16, Section 5]. We apply truncation to compute the representation at all levels and to determine the coefficients with respect to the THB-splines at level ℓ of the multigrid hierarchy. In order to speed up the computation, we determine, for each index $(\lambda, i) \in \mathcal{I}_{\ell-1}$, the highest level

$$\max\{\lambda' \,:\, \operatorname{supp} \beta_i^\lambda \cap \Omega_\ell^{\lambda'} \neq \emptyset\}$$

that can lead to non-zero entries in the prolongation matrix P_ℓ. The iteration is aborted when this level is reached.

3.4. Construction by adaptive refinement

In practice we usually construct the sequence of nested subdomains $\mathbf{\Omega}_\ell$, and therefore the sequence of associated discretization spaces, by adaptive refinement based on an a posteriori error estimator. We first describe a single step of our refinement strategy.

Given a previously computed discrete solution u_h, we compute the residual a posteriori error estimator for IgA introduced in [14]. On each cell K, the estimator η_K is defined by

$$\eta_K^2 := h_K^2 \| f - (-\Delta u_h|_K) \|_{L_2(K)}^2,$$

where h_K denotes the diameter of the cell K. Note that compared to classical residual error estimators from the finite element literature (cf. Braess [3]), the jump components of the estimator across cell interfaces vanish due to our use of trial spaces which are at least C^1-continuous. Based on this estimator, we refine those cells K which satisfy some thresholding criterion, $\eta_K \geq \theta$, where the threshold θ is chosen according to a relative thresholding strategy; see [14] for details.

We now describe the overall adaptive solution procedure. To this end, we start with a coarse tensor product B-spline discretization defined over the parameter domain. In our notation, this means that the initial hierarchy

$$\mathbf{\Omega}_0 = (D)$$

contains only a single domain, namely, the entire parameter domain. Let $\mathcal{V}_0 = \mathcal{H}(\mathbf{\Omega}_0)$ denote the hierarchical spline space defined over $\mathbf{\Omega}_0$, which is simply a tensor product spline space. Over this coarse space, we solve the discretized boundary value problem (4) and obtain a discrete solution u_0.

We now compute the estimators η_K based on u_0 and apply the adaptive refinement strategy described above, which results in the marking of some cells in D for refinement. The marked cells are collected in Ω_1^1, and we obtain a new hierarchy

$$\mathbf{\Omega}_1 = (D, \Omega_1^1)$$

which induces a refined hierarchical spline space $\mathcal{V}_1 = \mathcal{H}(\mathbf{\Omega}_1) \supseteq \mathcal{V}_0$. We now solve (4) in $\mathcal{V}_1$ and obtain a new discrete solution u_1. We repeat the process of a posteriori error estimation and adaptive refinement, adding at most one new level to the hierarchy at each iteration, so that after the k-th step we obtain a hierarchy

$$\mathbf{\Omega}_k = (D, \Omega_k^1, \ldots, \Omega_k^k)$$

and associated hierarchical spline space $\mathcal{V}_k = \mathcal{H}(\mathbf{\Omega}_k)$. Note that the refinement procedure may not only add an additional level to the hierarchical spline space, but also enlarge the subdomains associated to the previous levels, as discussed in Section 3.2. We terminate this process once we are satisfied with the accuracy of the obtained solution u_k.

It is clear that the same procedure can be performed when the initial hierarchy $\mathbf{\Omega}_0$ is non-trivial, for instance because the geometry map is already given in terms of a hierarchical basis. In our examples, however, we always start from a tensor product space.

3.5. Application of multigrid

The nested sequence of discretization spaces $\mathcal{V}_0 \subset \mathcal{V}_1 \subset \ldots \subset \mathcal{V}_k$ constructed in Section 3.4 allows us to apply the abstract multigrid framework described in Section 3.1 to the solution of the discretized problem in $\mathcal{V}_k$, where $k = L$ is the number of levels in the multigrid hierarchy. For the computation of the transfer matrices P_ℓ which describe the embedding of $\mathcal{V}_{\ell-1}$ in $\mathcal{V}_\ell$, we use the procedure described in Section 3.3.

In the present work, we use a simple Gauss-Seidel smoother, i.e., the smoothing matrix S_ℓ used in level ℓ is chosen as the lower triangular part of the stiffness matrix A_ℓ on the same level. It is understood by now that this simple smoothing strategy does not always result in optimal multigrid solvers in the isogeometric setting, in particular for higher spline degrees and space dimensions [19]. However, the design of optimal smoothers for IgA is still a field of active research (see, e.g., [18, 8]), and we therefore stick to the simple choice of the Gauss-Seidel smoother.

4. Numerical experiments

In this section, we present experimental results for three model problems. Example 1 uses a synthetically generated sequence of meshes generated *a priori*, while Examples 2 and 3 use adaptive refinement by *a posteriori* error estimation for problems with nonsmooth solutions.

In all examples, we compare iteration numbers for THB-spline basis functions with those for HB-spline basis functions. This is a simple change of basis which leaves the discretization spaces unchanged. In each example we test V-cycle iteration with one pre- and one post-smoothing Gauss-Seidel step on each level, as well as Conjugate Gradient (CG) iteration preconditioned with one such V-cycle. The stopping criterion in

all cases is the reduction of the Euclidean norm of the initial residual by a factor of 10^{-8}. Since all examples exhibit a similar trend, we discuss them collectively in some more detail in Section 5, the conclusion.

4.1. Example 1

On the square $P = (-1, 1)^2$, we construct a hierarchy of THB-spline spaces by adaptively fitting the function

$$u(x, y) = \exp(52\sqrt{(10x - 2)^2 + (10y - 3)^2})^{-1}$$

starting from a coarse tensor product spline space. The applied fitting method is described in details in [24]. An example of the mesh resulting after several fitting steps is shown in Figure 6.

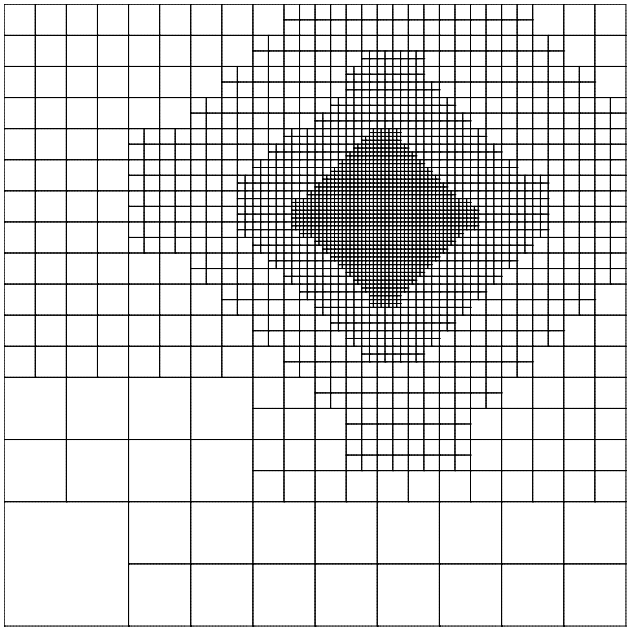


Figure 6: THB-spline supports after five steps of adaptive fitting with quadratic THB-splines (Example 1)

We then solve the Poisson equation

$$-\Delta u = f \quad \text{in } P, \qquad u|_{\partial P} = g,$$

where the right-hand side f and Dirichlet data g are chosen according to the exact solution u given above. We use the sequence of nested THB-spline spaces generated by the fitting process as the grids in our multigrid method.

Note that in this example, the mesh is refined *a priori* in such a way as to be able to represent the exact solution u well. This mimics the situation of an ideal (but impractical) error estimator. This example was included in order to separate the performance study of the multigrid method from the quality of a practical error estimator. The behavior of the estimator influences not only the approximation quality of the obtained solution, but also the sequence of multigrid spaces and therefore to a certain degree the convergence of the algorithm. Therefore, this test case serves to illustrate that the later examples, which we will perform with true *a posteriori* error estimators, are not strongly dependent on the particular choice of estimator.

The iteration numbers are reported in Table 1. Here, the upper part of the table gives the number of V-cycle iterations needed to reduce the ℓ_2-norm of the initial residual by 10^{-8}. For the lower part of the table, instead of V-cycle iteration, we perform Conjugate Gradient (CG) iteration preconditioned with one V-cycle.

V-cycle iteration

	$p=2$			$p=3$			$p=4$		
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	49			64			81		
1	109	11	27	169	43	43	196	167	167
2	365	15	48	475	90	554	528	891	12443
3	829	12	31	1174	75	441	1272	628	13126
4	1487	13	29	2266	81	414	2509	404	10771
5	2317	12	29	3580	75	408	4136	419	9091

CG iteration preconditioned with one V-cycle

	$p=2$			$p=3$			$p=4$		
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	49			64			81		
1	109	7	9	169	15	15	196	28	28
2	365	8	14	475	18	29	528	38	74
3	829	8	12	1174	19	46	1272	57	229
4	1487	8	12	2266	20	48	2509	50	229
5	2317	8	12	3580	19	45	4136	47	212

Table 1: Iteration numbers for Example 1

Overall, we observe that when using THB-splines, the number of iterations is significantly smaller than for HB-splines, in particular for higher spline degrees. Furthermore, the use of preconditioned CG iteration further reduces the iteration numbers, more pronouncedly so for the cases with higher degrees, without significantly increasing the computational cost of one cycle.

4.2. Example 2

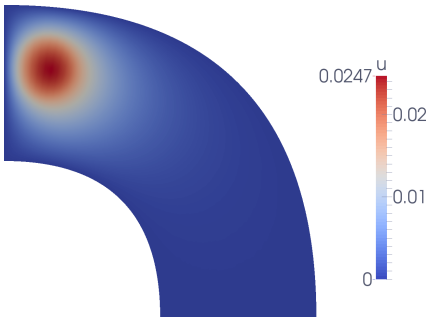


Figure 7: Solution field obtained in Example 2 with quadratic THB-splines.

We solve the Poisson equation

$$-\Delta u = f \quad \text{in } P, \qquad u|_{\partial P} = 0$$

with the source function

$$f(x,y) = \begin{cases} 1, & (x-0.25)^2 + (y-1.6)^2 < 0.2^2, \\ 0, & \text{otherwise.} \end{cases}$$

The domain P approximates a quarter annulus in the first quadrant with inner and outer radius 1 and 2, respectively, by means of quadratic tensor product B-splines.

Starting from a coarse tensor-product B-spline basis ($\ell = 0$), we construct a hierarchy of THB-spline spaces by means of adaptive refinement based on an *a posteriori* error estimator as described in Section 3.4.

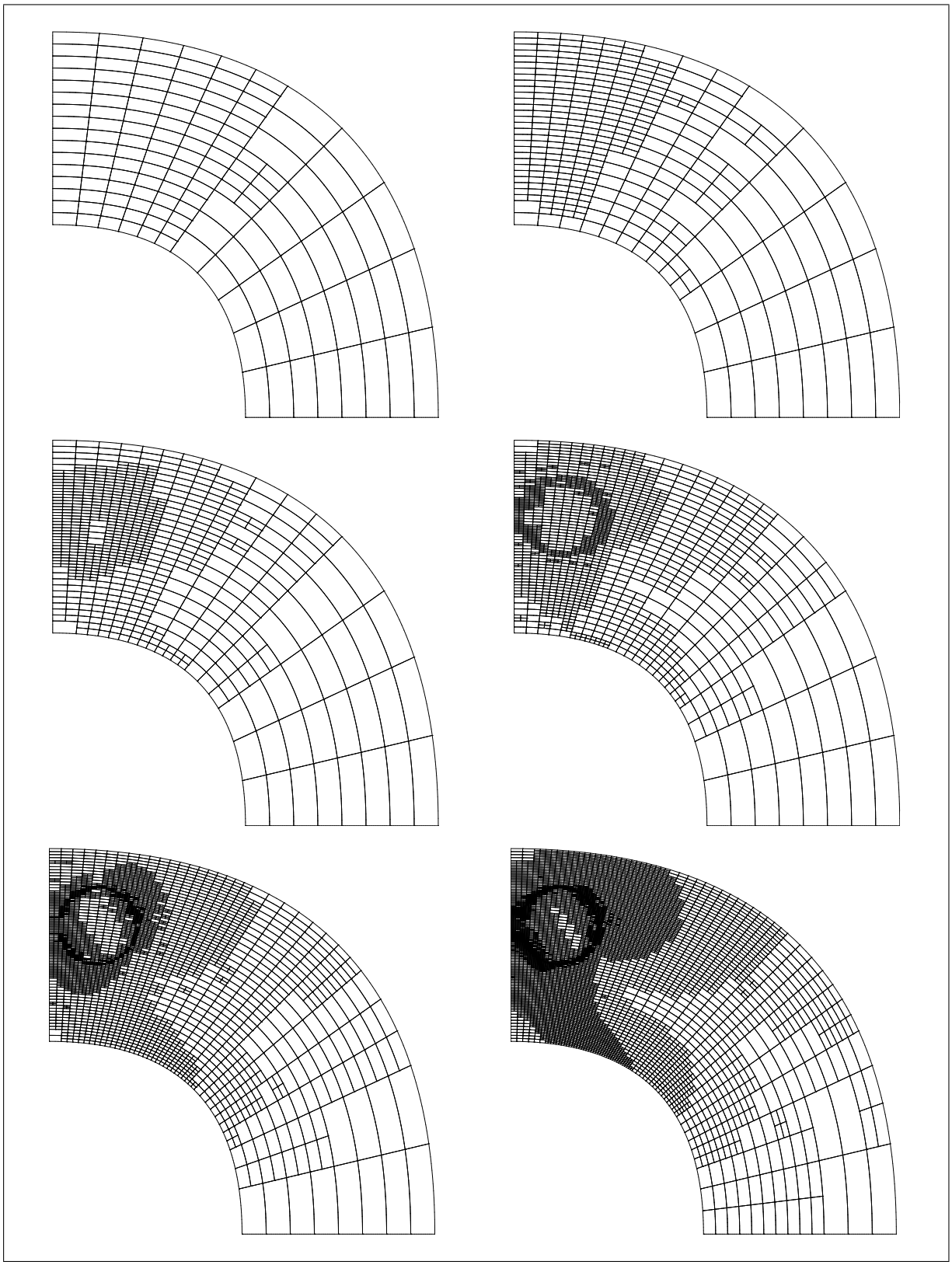


Figure 8: Meshes for different multigrid levels $\ell = 1, \ldots, 6$ in Example 2 using quadratic THB-splines.

	V-cycle iteration								
		$p=2$			$p=3$			$p=4$	
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	100			121			144		
1	182	15	35	195	70	253	226	217	4128
2	343	19	42	353	78	379	381	428	6577
3	701	25	57	668	136	1187	673	527	6387
4	1355	26	64	1326	200	742	1216	2109	26743
5	2822	23	98	2635	145	991	2312	1489	35034
	CG iteration preconditioned with one V-cycle								
		$p=2$			$p=3$			$p=4$	
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	100			121			144		
1	182	9	12	195	16	26	226	30	61
2	343	10	14	353	20	44	381	39	123
3	701	11	17	668	26	57	673	52	158
4	1355	11	18	1326	26	63	1216	84	297
5	2822	11	21	2635	27	70	2312	88	351

Table 2: Iteration numbers for Example 2

The meshes of the resulting hierarchical bases when using quadratic THB-splines are shown in Figure 8. A plot of the obtained solution is shown in Figure 7.

We then set up a multigrid method as described above which always has $\ell = 0$ as its coarse grid and solves the problem on some given level ℓ by V-cycle iteration.

The iteration numbers are reported in Table 2. Once again, when using THB-splines, the number of iterations is significantly smaller than for HB-splines, and CG iteration further reduces the numbers.

4.3. Example 3

We solve the classical benchmark problem for a singularity arising from a reentrant corner on the L-shape domain

$$-\Delta u = 0 \quad \text{in } P, \qquad \text{where } P = (-1,1)^2 \setminus [0,1]^2.$$

We choose pure Dirichlet boundary conditions according to the exact solution, given in polar coordinates,

$$g(r,\varphi) = r^{2/3}\sin((2\varphi - \pi)/3),$$

where the argument is assumed to have the range $\varphi \in [0, 2\pi)$. The geometry is represented by piecewise linear splines.

We then perform adaptive refinement using THB-splines as described in Example 2. An example of a THB-spline basis arising from adaptive refinement is shown in Figure 9.

The procedure is as in Example 2, and we report the resulting iteration numbers in Table 3, which again confirms the observation regarding the superior performance of THB-splines.

5. Conclusion

We have presented a multigrid solver for isogeometric discretizations of the Poisson equation on adaptively refined hierarchical spline spaces. We have tested the solver both on synthetic examples as well as on hierarchies which were generated by adaptive refinement where the exact solution has a singularity. In each example, we compared the iteration numbers when the hierarchical spline space is represented by standard HB-splines and by THB-splines.

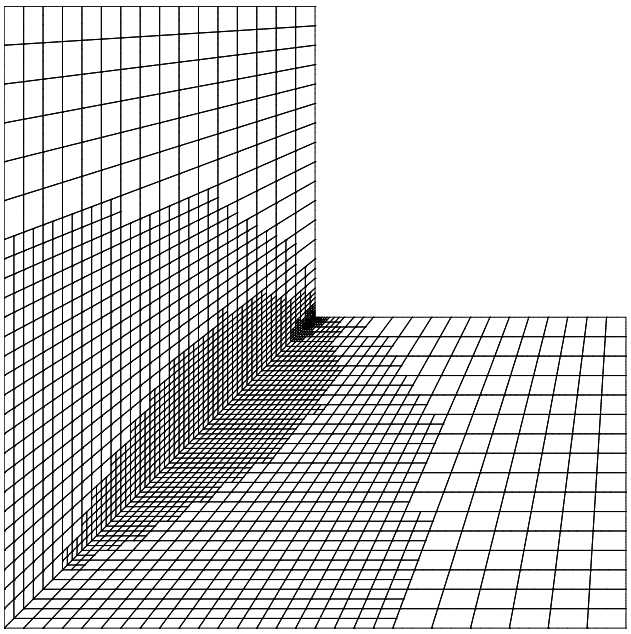


Figure 9: L-shape domain with cubic THB-splines after four adaptive refinement steps (2379 dofs)

	V-cycle iteration								
		$p=2$			$p=3$			$p=4$	
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	630			703			780		
1	810	18	33	909	71	297	982	392	3487
2	1163	16	35	1252	68	317	1332	370	3300
3	1631	18	33	1721	84	359	1783	557	4266
4	2338	17	41	2379	73	395	2458	446	5873
5	3327	18	32	3414	87	382	3514	615	6804
	CG iteration preconditioned with one V-cycle								
		$p=2$			$p=3$			$p=4$	
		THB	HB		THB	HB		THB	HB
ℓ	dofs	iter	iter	dofs	iter	iter	dofs	iter	iter
0	630			703			780		
1	810	9	12	909	19	36	982	41	98
2	1163	9	13	1252	19	39	1332	40	116
3	1631	9	13	1721	20	41	1783	50	146
4	2338	9	15	2379	20	43	2458	53	160
5	3327	9	13	3414	21	42	3514	62	190

Table 3: Iteration numbers for Example 3

In all examples, we found that the V-cycle iteration numbers to reduce the initial residual by a fixed factor were significantly smaller when using THB-splines compared to HB-splines. The difference in iteration numbers was especially dramatic for higher spline degrees.

For high spline degrees, the pure V-cycle iteration numbers are not satisfactory even when using THB-splines. This is consistent with the known fact that even in the case of tensor product B-splines, the Gauss-Seidel smoother does not perform well for high spline degrees [19]. Research into optimal and robust smoothers for isogeometric multigrid methods is ongoing, and we can hope that insights gained in the tensor product case will also result in better smoothers for the more complicated case of hierarchical splines.

As a remedy, we have used CG iteration preconditioned with one V-cycle. The numerical examples have shown that this usually results in a significant reduction of the iteration numbers, while having the same asymptotic computational complexity per cycle. The speedup is particularly dramatic for the cases with higher degrees and makes the iteration numbers very practical even for the case $p = 4$. The advantage of THB-splines over HB-splines remains clear also in the case of CG iteration.

Acknowledgments. This work was supported by the National Research Network "Geometry + Simulation" (NFN S117) and by the Doctoral Programme "Computational Mathematics" (W1214), both funded by the Austrian Science Fund (FWF).

References

[1] 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(5Ũ8):229–263, 2010.

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

[3] D. Braess. *Finite Elements: Theory, Fast Solvers, and Applications in Solid Mechanics*. Cambridge University Press, 2007.

[4] W. L. Briggs, V. E. Henson, and S. F. McCormick. *A Multigrid Tutorial*. Society for Industrial and Applied Mathematics, 2000.

[5] A. Buffa, H. Harbrecht, A. Kunoth, and G. Sangalli. BPX-preconditioning for isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 265:63–70, 2013.

[6] A. Chemin, T. Elguedj, and A. Gravouil. Isogeometric local h-refinement strategy based on multigrids. *Finite Elements in Analysis and Design*, 100:77–90, 2015.

[7] M. Donatelli, C. Garoni, C. Manni, S. Serra-Capizzano, and H. Speleers. Robust and optimal multi-iterative techniques for IgA Galerkin linear systems. *Computer Methods in Applied Mechanics and Engineering*, 284:230–264, 2014.

[8] M. Donatelli, C. Garoni, C. Manni, S. Serra-Capizzano, and H. Speleers. Symbol-based multigrid methods for Galerkin B-spline isogeometric analysis. *TW Reports*, 2014.

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

[10] M.R. Dörfel, B. Jüttler, and B. Simeon. Adaptive isogeometric analysis by local h-refinement with T-splines. *Computer Methods in Applied Mechanics and Engineering*, 199(5-8):264–275, 2010.

[11] A. Falini, J. Špeh, and B. Jüttler. Planar domain parameterization with THB-splines. *Computer Aided Geometric Design*, 35–36:95–108, 2015. Geometric Modeling and Processing 2015.

[12] David R. Forsey and Richard H. Bartels. Hierarchical B-spline refinement. *Computer Graphics (ACM SIGGRAPH)*, 22(4):205–212, 1988.

[13] K. P. S. Gahalaut, J. K. Kraus, and S. K. Tomar. Multigrid methods for isogeometric discretization. *Computer Methods in Applied Mechanics and Engineering*, 253:413–425, 2013.

[14] C. Giannelli, B. Jüttler, S. K. Kleiss, A. Mantzaflaris, B. Simeon, and J. Špeh. 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, 2016.

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

[16] C. Giannelli, B. Jüttler, and H. Speleers. Strongly stable bases for adaptively refined multilevel spline spaces. *Advances in Computational Mathematics*, 40(2):459–490, 2014.

[17] W. Hackbusch. *Multi-Grid Methods and Applications*, volume 4 of *Springer Series in Computational Mathematics*. Springer-Verlag, Berlin, Heidelberg, New York, Tokyo, 2003.

[18] C. Hofreither and W. Zulehner. Mass smoothers in geometric multigrid for isogeometric analysis. In J.-D. Boissonnat, A. Cohen, O. Gibaru, C. Gout, T. Lyche, M.-L. Mazure, and L. L. Schumaker, editors, *Curves and Surfaces*, volume 9213 of *Lecture Notes in Computer Science*, pages 272–279. Springer International Publishing, 2015.

[19] C. Hofreither and W. Zulehner. Spectral analysis of geometric multigrid methods for isogeometric analysis. In I. Dimov, S. Fidanova, and I. Lirkov, editors, *Numerical Methods and Applications*, volume 8962 of *Lecture Notes in Computer Science*, pages 123–129. Springer International Publishing, 2015.

[20] C. Hofreither and W. Zulehner. On full multigrid schemes for isogeometric analysis. In T. Dickopf, J. M. Gander, L. Halpern, R. Krause, and F. Luca Pavarino, editors, *Domain Decomposition Methods in Science and Engineering XXII*, pages 267–274. Springer International Publishing, 2016.

[21] 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(39-41):4135–4195, October 2005.

[22] K. A. Johannessen, T. Kvamsdal, and T. Dokken. Isogeometric analysis using LR B-splines. *Computer Methods in Applied Mechanics and Engineering*, 269:471–514, 2014.

[23] G. Kiss, C. Giannelli, and B. Jüttler. Algorithms and data structures for truncated hierarchical B-splines. In M. Floater et al., editors, *Mathematical Methods for Curves and Surfaces*, volume 8177 of *Lecture Notes in Computer Science*, pages 304–323. Springer, 2014.

[24] G. Kiss, C. Giannelli, U. Zore, B. Jüttler, D. Großmann, and J. Barner. Adaptive CAD model (re-)construction with THB-splines. *Graph. Models*, 76(5):273–288, 2014.

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

[26] N. Nguyen-Thanh, H. Nguyen-Xuan, S.P.A. Bordas, and T. Rabczuk. Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids. *Computer Methods in Applied Mechanics and Engineering*, 200(21-22):1892–1908, 2011.

[27] D. Schillinger, L. Dedé, 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*, 249:116–150, 2012.

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

[29] 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.

[30] U. Trottenberg, C.W. Oosterlee, and A. Schüller. *Multigrid*. Academic Press, 2001.

[31] 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*, 200:3554–3567, 2011.

[32] P. Wang, J. Xu, J. Deng, and F. Chen. Adaptive isogeometric analysis using rational PHT-splines. *Computer-Aided Design*, 43(11):1438–1448, 2011.