Adaptive IsoGeometric and ImmersoGeometric analysis using truncated hierarchical B-splines
applications in steady heat conduction and steady viscous flow

van Zuijlen, P.H.

Award date:
2019

EINDHOVEN UNIVERSITY OF TECHNOLOGY

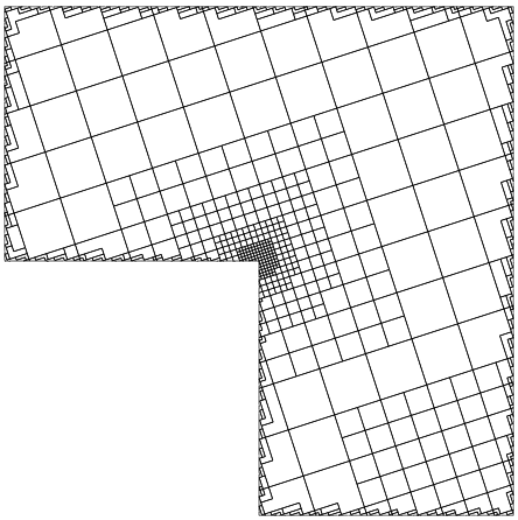
Department of Mechanical Engineering
Energy Technology Group

Adaptive IsoGeometric and ImmersoGeometric analysis using truncated hierarchical B-splines

Applications in steady heat conduction and steady viscous flow

Thesis supervisors:
Dr. Ir. C.V. Verhoosel
Dr. Ir. H. Hoang

Written by:
P.H. van Zuijlen



Eindhoven, April 2, 2019

Abstract - This work presents the framework of hierarchical refinement and error estimation based adaptivity in the IsoGeometric and ImmersoGeometric settings. The concept of truncated hierarchical B-splines is employed and a detailed study on its implementation aspects is conducted. An efficient algorithm for the construction of the truncated hierarchical basis is provided. Residual-based and goal-oriented error estimation are used to elaborate local error estimators, which allow for adaptively optimizing the mesh. These adaptivity methods are tested on benchmark applications in steady heat conduction and steady viscous flow. The adaptive refinement framework is successfully applied in the context of the ImmersoGeometric setting. Numerical experiments are presented to demonstrate the capabilities of the adaptive immersed framework.

Keywords - Truncated hierarchical B-splines, Error estimation & adaptivity, IsoGeometric analysis, ImmersoGeometric analysis, Finite cell method

Contents

1 Introduction

Numerical approximation methods, such as the Finite Element Method (FEM) or IsoGeometric Analysis (IGA), are based on calculating the approximation of a solution subject to a physical problem. Evidently it is desired to find a sufficiently accurate approximated solution within an acceptable computational timescale. Adaptive methods have been developed to improve the approximated solution by refining the mesh, while not rigorously increasing the computational effort. Methods based on a posteriori error approximation are regarded herein to construct local error estimators based on the approximation results. These local error indicators are used to adaptively refine the mesh, allowing an optimal relation between simulation quality and computational cost.

The residual-based adaptivity method is a commonly used adaptivity method [1,18]. In this method, the residual is used to evaluated local error estimators which identify error sensitive elements in the mesh. As an alternative adaptivity method, goal-oriented adaptivity [1–3] is employed. The elaboration of the local error estimators is based on a chosen quantity of interest, rather than the residual. This allows for optimizing the mesh with respect to the quantity of interest. In this work, both error estimation methods are employed based on steady heat conduction and steady viscous flow problems. The adaptive setting using these proposed error estimation methods is studied based on benchmark cases.

In this work the computational framework of IsoGeometric Analysis (IGA) [4] is used. The original idea of IGA is to use the same spline-based discretization space for the conforming mesh representation of the geometry as for the analysis of the solution. In contrast to classical finite element methods, the time consuming process of constructing the analysis-suitable geometry and the mesh does not exist. In this work, a function basis consisting of B-splines is considered. To fit the locally refined structure, truncated hierarchical B-splines [5] are used. The construction of the truncated hierarchical basis is elaborated and its performance is compared to the standard hierarchical basis [6]. Implementation aspects to efficiently compute the basis are presented.

The construction of the conforming mesh representation of a complex geometry can be an intricate process. To this end, IGA has been combined with the Finite Cell method [7–9]. The central idea is to capture the complex domain within a simple embedding domain. The geometry is resolved by an advanced recursively constructed integration scheme. The combination of immersed methods and IsoGeometric analysis, referred to as ImmersoGeometric analysis, forms a natural framework for capturing the complex geometry. Volumetric analysis of a broad class of geometry parametrizations is enabled by this framework.

Applications of ImmersoGeometric analysis in the context of incompressible flow problems have been studied by Hoang et. al. [10, 11]. In References [10, 11] stabilization methods are presented to obtain oscillation-free solutions for the velocity and pressure fields. The main objective of this work is to enhance the stabilized ImmersoGeometric analysis framework with error estimation and adaptivity functionality. Implementation aspects for this framework are studied and numerical examples are presented to test the proposed methodology.

This thesis is outlined as follows: The principle of the truncated hierarchical basis is elaborated in Section 2, followed by a numerical comparison between the truncated hierarchical B-splines and the standard hierarchical B-splines. The residual-based and goal-oriented error estimation methods regarding the steady heat conduction problems are elaborated in Section 3. In Section 4 the elaboration of these error estimation methods is done for steady viscous flow problems. In Section 5 the adaptivity enhanced stabilized ImmersoGeometric setting is elaborated and its performance is demonstrated with numerical examples.

2 Truncated hierarchical B-splines

In this section, the construction of truncated hierarchical splines is discussed after a brief explanation of the fundamentals of B-splines. To this end, a brief explanation on the construction of B-splines is given. Thereafter the mechanism to extend the B-spline basis to a hierarchical structure is elaborated. A numerical study is conducted to compare the truncated hierarchical B-splines (THB-splines) with the hierarchical B-splines (HB-splines).

2.1 B-splines

B-splines are constructed using an underlying knot vector [4]. The knot vector is a set of non-decreasing coordinates or knots in a one dimensional parametric domain $\Xi = \{\xi_1, \xi_2, ..., \xi_{p+n+1}\}$, where ξ denotes the coordinate of the parametric domain, p is the polynomial degree of the B-splines considered and n is the number of B-splines. The B-splines are constructed with the Cox-de Boor recursion formula, defined as:

$$
\begin{aligned}
&N_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i+1,p-1}(\xi) \\
&\text{with} \\
&N_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad i = 1, 2, ..., p+n+1
\end{aligned}
\tag{2.1}
$$

Note that the number of elements that support a B-spline depends on the degree p. The overlap of the B-splines is increased with an increasing degree. The continuity of the B-splines is determined by the knot repetition. Without the occurrence of knot repetitions C^{p-1}-continuous B-splines are obtained. C^{-1}-continuous B-splines are obtained at the domain boundary where the knot value appears $p+1$ times and C^0-continuity is obtained on the interface between different patches. B-splines form a partition of unity, meaning $\sum_i N_i = 1$.

Tensor product B-splines are used for multidimensional parametric spaces, such as surfaces and volumes. The tensor product of the knot vectors in multiple dimensions forms a patch, whereas their knot spans form the elements. The mesh $\mathcal{Q}$ is formed by one or more patches. The set of all tensor product B-splines defined on the uniform mesh is referred to as the uniform basis $\mathcal{B} = \{N_1, N_2, ..., N_n\}$.

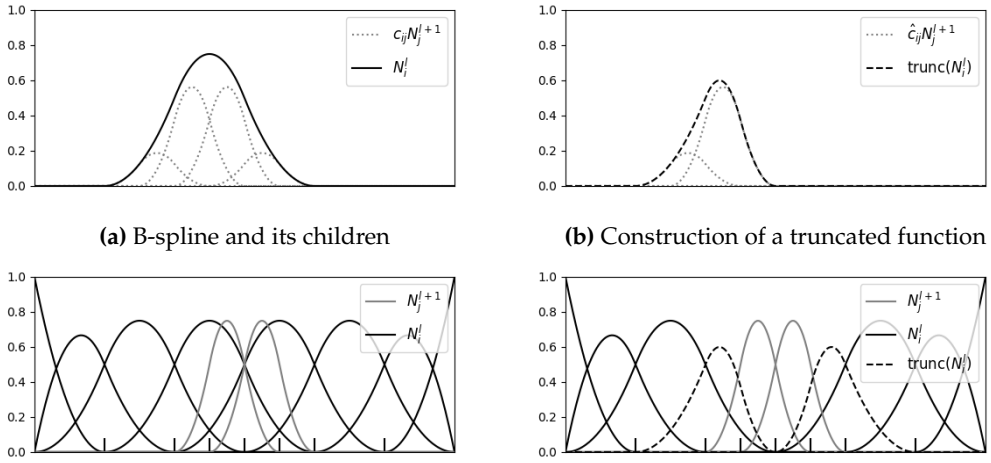
Consider $\mathcal{Q}^l$ as the mesh $\mathcal{Q}$ after applying $l \in \mathbb{N}_0$ uniform refinements, with a corresponding set of uniform B-splines $N_i^l \in \mathcal{B}^l$. Because of the nested nature of different refinement levels, B-splines of level l can be constructed as a linear combination of B-splines of level $l+1$ [6]

$$
N_i^l = \sum_j c_{ij} N_j^{l+1}, \tag{2.2}
$$

with non-negative coefficients $c_{ij} \in \mathbb{R}_{\geq 0}$. The B-splines N_j^{l+1} for which $c_{ij} \neq 0$ are referred to as the children of N_i^l. An example of a function N_i^l with its children N_j^{l+1} is shown in Figure 1a.

2.2 Hierarchical refinement

The mesh, resulting from the tensor product of the knot vectors, can be refined by means of knot insertion, which is done by inserting new knots in one of the knot vectors. For multidimensional cases, this insertion causes refinements over the full length of the mesh, as is illustrated in Figure 2a. Therefore knot insertion is suboptimal for refining specific parts of the domain. Local hierarchical refinement is considered a more appropriate method, which is done by bi-sectioning targeted elements into 2^d elements, where d indicates the dimension of the mesh.



(a) B-spline and its children

(b) Construction of a truncated function

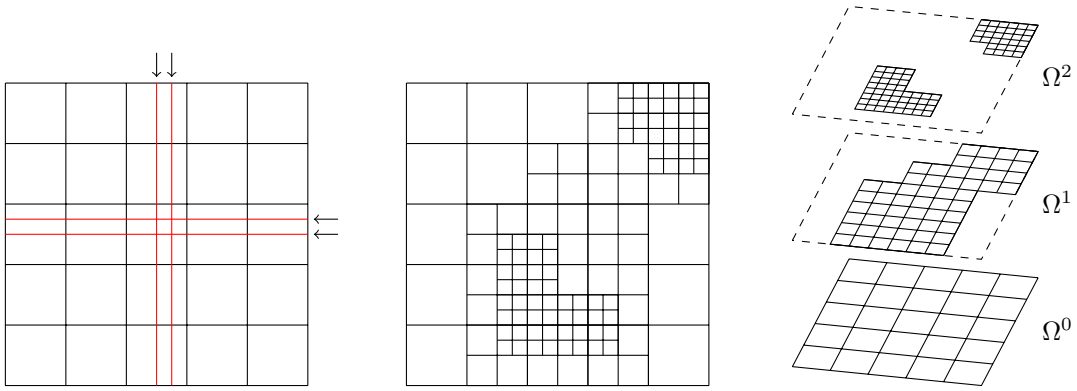
(c) Hierarchical B-spline basis

(d) Truncated hierarchical B-spline basis

Figure 1: In **(a)** the linear combination of B-splines N_j^{l+1} that constructs N_i^l is shown. The two most right children are present in the hierarchical basis in **(c)**, causing a duplication. These two functions are excluded when creating truncated B-splines **(b)**. The truncated hierarchical basis is shown in **(d)**.

Let $\Omega \subset \mathbb{R}^d$ be the domain on which the mesh is defined. For each level l, a subdomain of Ω is defined as the union of all elements of that level and all finer levels, with the finest level defined as N. An example of a hierarchical mesh and its subdomains is shown in Figure 2b and 2c. All subdomains are a subset of a coarser level subdomain, such that:

$$\Omega = \Omega^1 \supseteq \Omega^2 \supseteq ... \supseteq \Omega^N \tag{2.3}$$



(a) Knot insertion

(b) Hierarchical mesh $\mathcal{Q}$

(c) Subdomains of $\mathcal{Q}$

Figure 2: In **(a)** an example of knot insertion is shown. It can be seen that knot insertion affects the global mesh. Figure **(b)** shows an example of a hierarchical mesh $\mathcal{Q}$, with its subdomains Ω^l shown in **(c)**.

Because the underlying tensor product knot vector structure is lost in the hierarchical mesh, the basis cannot be build by regular B-splines as defined in Section 2.1. A possible way of constructing the basis is by hierarchical B-splines (HB-splines) [6]. To construct the HB-splines, functions from uniform B-spline bases of different levels $\mathcal{B}^l$ are selected. B-splines with their support completely within it's corresponding subdomain Ω^l, but not completely within the subdomain of one finer level Ω^{l+1} are placed in the hierarchical basis. This construction can be formalized as:

$$\left[\begin{array}{ll} \mathcal{H} & = \bigcup_{l=0}^{N} \mathcal{H}^l \\ \text{with} & \\ \mathcal{H}^l & = \{N_i^l \,|\, \text{supp}(N_i^l) \subseteq \Omega^l,\, \text{supp}(N_i^l) \not\subseteq \Omega^{l+1}\} \qquad l = \{1, ..., N-1\} \\ \mathcal{H}^N & = \{N_i^N \,|\, \text{supp}(N_i^N) \subseteq \Omega^N\} \end{array}\right. \tag{2.4}$$

One disadvantage of the hierarchical basis is that the partition of unity is not preserved, due to the overlap of basis functions of different levels, as can be seen in Figure 1c. Some finer level functions N_j^{l+1} are placed in the hierarchical basis, while they are also the children of coarser functions N_i^l, according to Equation (2.2).

The duplication of these functions causes the loss of partition of unity. Partition of unity is an important property in CAD, since it enables affine transformations to the geometry. Consider X_i to be a set of control points, such that a geometry $s(\xi)$ is constructed by $s(\xi) = \sum_i X_i N_i(\xi)$. Let $\mathbf{T}$ be a transformation matrix and $\mathbf{b}$ a translation vector. Only if partition of unity is preserved by the set of B-splines, affine transformations can be applied on the geometry s as well as the control points X_i, such that:

$$s^*(\xi) = \mathbf{T}s(\xi) + \mathbf{b} \qquad \xLeftrightarrow{\sum_i N_i(\xi)=1} \qquad s^*(\xi) = \sum_i (\mathbf{T}X_i + \mathbf{b}) N_i(\xi) \tag{2.5}$$

Because affine transformation are commonly performed in CAD, partition of unity is beneficial property from this perspective. The mechanism of building THB-splines is based on excluding the contribution of the children in N_i^l if they are already in the basis. This way, the support of the coarse B-splines is reduced and partition of unity is preserved [5] (Figure 1d). This makes THB-splines suitable for analysis as well as for CAD. The exclusion of basis function children is done by defining constants $\hat{c}^{l+1}$, such that:

$$\hat{c}_{ij}^{l+1} = \begin{cases} c_{ij} & \text{if supp}(N_j^{l+1}) \not\subseteq \Omega^{l+1} \qquad N_j^{l+1} \in \mathcal{B}^{l+1} \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

The recursive truncation operation is then defined as:

$$\text{trunc}(N_i^l) = \sum_j \hat{c}_{ij} \text{trunc}(N_j^{l+1}) \tag{2.7}$$

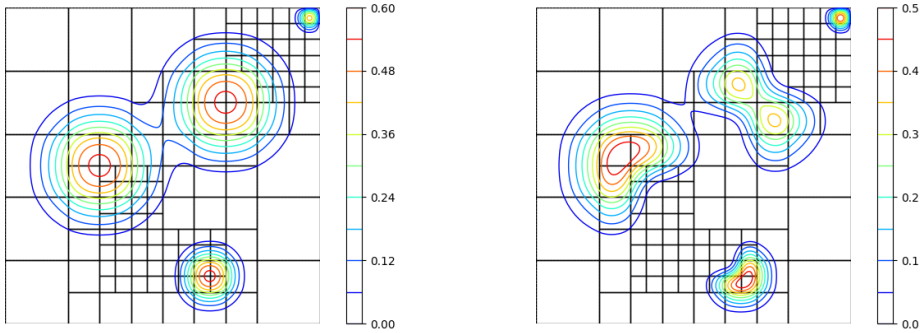
This operation is visualized in Figure 1b. The functions used to form $\text{trunc}(N_i^l)$ should be truncated functions of their own. This way the duplication of B-splines over all finer levels is avoided. Note that for functions of the highest level truncation has no effect:

$$\text{trunc}(N_i^N) = N_i^N \tag{2.8}$$

The truncated hierarchical basis $\mathcal{T}$ is constructed in a similar way as the hierarchical basis $\mathcal{H}$, however truncated B-splines are placed in the basis instead of regular B-splines:

$$\left[\begin{array}{ll} \mathcal{T} & = \bigcup_{l=0}^{N} \mathcal{T}^l \\ \text{with} & \\ \mathcal{T}^l & = \{\text{trunc}(N_i^l) \,|\, \text{supp}(N_i^l) \subseteq \Omega^l,\, \text{supp}(N_i^l) \not\subseteq \Omega^{l+1}\} \qquad l = \{1, ..., N-1\} \\ \mathcal{T}^N & = \{N_i^N \,|\, \text{supp}(N_i^N) \subseteq \Omega^N\} \end{array}\right. \tag{2.9}$$

In figure 1d the multi-level univariate truncated hierarchical basis is shown. The dotted lines indicate which functions are affected by the truncation operation. In Figure 3 four quadratic tensor product B-splines in the hierarchical basis and the truncated hierarchical basis are plotted. The function in the upper right corner is identical for both bases. The support of the rest of the functions is reduced in the truncated basis, following the structure of the mesh.



(a) Hierarchical B-splines

(b) Truncated hierarchical B-splines

Figure 3: The height lines of four tensor product HB-splines **(a)** and corresponding tensor product THB-splines **(b)** are shown. The shape of the THB-splines is affected by the hierarchical structure of the mesh.

2.3 Algorithm implementation aspects

In this section, implementation aspects of the algorithm for the construction of the truncated hierarchical basis and the hierarchical basis are discussed. Different methods for building the basis are explained in [12, 13]. The algorithm developed in the context of this work has been implemented in the Nutils finite element library [14]. In this section, some important implementation aspects are discussed.

2.3.1 Defining the basis

After the mesh has been refined, the first step of constructing the basis is to define which functions are placed in the basis. This selection is based on the overlap between the support of B-splines N_i^l and the subdomains $\Omega^l \subset \Omega$, as can be seen in Equations (2.4) and (2.9). A possible approach is to construct all uniform bases $\mathcal{B}^l$ for $l = 1, 2, ..., N$ and to evaluate the support of all B-splines in these bases on the subdomains Ω^l. The number of B-splines in basis $\mathcal{B}^l$, denoted by n^l, is approximated by $n^l \sim m \cdot 2^{dl}$, where m is the number of elements in the initial mesh $\mathcal{Q}$ and d is the dimension. Hence, the number of operations needed, denoted by order $O(n^N)$, is dependent on the maximum level N. It can be seen that for high refinement levels, this approach is computationally expensive, because the number of operations scales exponentially with the refinement level.

A 'lazy' approach is developed in the context of this thesis, using the structure of the hierarchical mesh and Cox-de-Boor algorithm to determine which B-splines have support on each element in the hierarchical mesh. This is done in an element-wise way, such that only $O(m)$ operations are needed. Note that, in contrast to n^N, m does not exponentially increase with refinement depth l.

Two different sets of functions are distinguished, called active $\mathcal{A}$ and passive $\mathcal{P}$ B-splines. The active B-splines are placed in the truncated hierarchical basis or the hierarchical basis. The passive B-splines are used as children for coarser functions that overlap with the regarded subdomain, which are truncated in the truncated hierarchical basis. These sets are defined as:

$$\mathcal{A}^l = \{N_i^l \,|\, \text{supp}(N_i^l) \subseteq \Omega^l, \text{supp}(N_i^l) \not\subseteq \Omega^{l+1}\} \tag{2.10a}$$

$$\mathcal{P}^l = \{N_i^l \,|\, \text{supp}(N_i^l) \cap \Omega^l \neq \emptyset, \text{supp}(N_i^l) \cap \Omega^{l-1} \neq \emptyset\} \tag{2.10b}$$

In the next step, the construction of the basis is done. A comparison between a global construction approach and an element-wise approach is made.

2.3.2 Global construction approach

By collecting the coefficients c_{ij}, corresponding to $N_i^l \in \mathcal{B}^l$ and $N_j^{l+1} \in \mathcal{B}^{l+1}$, in a matrix $\mathbf{R}$, the global execution of Equation (2.7) can be defined as:

$$\mathcal{B}^l = \mathbf{R}_l^{l+1} \cdot \mathcal{B}^{l+1} \tag{2.11}$$

The matrix over multiple levels is found by $\mathbf{R}_{l-1}^{l+1} = \mathbf{R}_{l-1}^{l} \cdot \mathbf{R}_{l}^{l+1}$. The global multi-level matrix $\mathbf{ML}$ can be constructed by making a selection of the rows of each matrix $\mathbf{R}_i^N$ for $i = 1, 2, ..., N$ with maximum refinement level N, where the rows correspond to the active B-splines in all sets $\mathcal{A}^l$. The hierarchical basis is then found by:

$$\mathcal{H} = \mathbf{ML} \cdot \mathcal{B}^N \tag{2.12}$$

Note that the truncated hierarchical basis is found by using the coefficients $\hat{c}_{ij}$ instead of c_{ij} for the assembly of $\mathbf{ML}$. The number of operations to be executed in this algorithm is dominated by the number of B-splines in the finest basis $\mathcal{B}^N$, and thus of order $O(n^N)$. Practically this restricts this type of algorithm to compute cases with a moderate refinement level, because the amount of operations grows exponentially.

2.3.3 Element-wise construction approach

In this approach, the algorithm consists of a loop over all elements $K \in \mathcal{Q}$, where m is the total number of elements in $\mathcal{Q}$. The hierarchical level of the element K is denoted by l_K. On each element the set of polynomials, based on the Cox-de-Boor algorithm, is evaluated. A linear combination of these polynomials is used to construct the parts of the B-splines $N_i^{l_K}$ that are supported by the element K. All active and passive B-splines from $\mathcal{A}^{l_K}$ and $\mathcal{P}^{l_K}$ that are supported by the element are constructed.

In a loop going through all coarser levels, starting at $l_K - 1$, it is checked whether any coarser B-splines in $\mathcal{A}^l$ and $\mathcal{P}^l$ overlap the element. If this is the case, these B-splines are constructed by the linear combination of their children, following Equation (2.2) for the hierarchical basis and (2.7) for the truncated hierarchical basis. Note that in the latter case, the passive B-splines on that element are updated with their truncated version as well. This allows a correct truncation over multiple levels.

Because of the element-wise structure of this algorithm, the total number of operations scales with $O(m)$. Therefore this algorithm is computationally much more beneficial than the global construction approach. In Appendix A the code implemented in nutils v5.0a0 [14] is given, which is based on the element-wise construction approach. Some comments are provided to elaborate the code.

2.4 Bases performance analysis

In this section, the performance of the HB-splines is compared to the THB-splines. The considered performance measures are the computational efficiency and the stability. These measures are quantified by the sparsity and the condition number related to the matrices constructed with the bases. The analysis is based on the mass matrix $\mathbf{M}$ and the stiffness matrix $\mathbf{K}$, defined as:

$$M_{ij} = \int_\Omega N_i N_j \, d\Omega \qquad K_{ij} = \int_\Omega \nabla N_i \nabla N_j \, d\Omega \qquad i, j = 1, 2, ..., n\,, \tag{2.13}$$

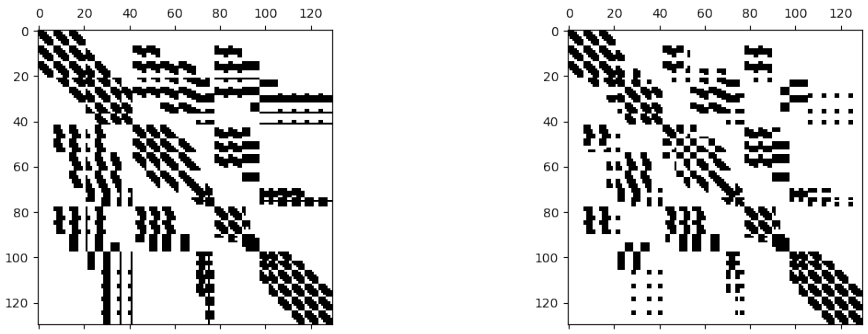
taking numerical integration into account. The matrices constructed with the truncated hierarchical basis and the standard hierarchical basis are refered to as $\mathbf{M}^{\mathcal{T}}$, $\mathbf{K}^{\mathcal{T}}$ and $\mathbf{M}^{\mathcal{H}}$, $\mathbf{K}^{\mathcal{H}}$ respectively. For more results on the similarities and differences of the different B-splines, the reader is referred to Johannessen et. al. [15].

2.4.1 Sparsity

The sparsity of a matrix is related to the computational effort needed to solve the system of equations. Solving a dense matrix requires more system memory and more operations than solving a sparse matrix. The sparsity is measured by the number of non-zero entries in the matrix, which is identical for matrices $\mathbf{M}$ and $\mathbf{K}$. The sparsity patterns for both bases resulting from the example mesh from Figure 2b are shown in Figure 4. An entry in the matrix is non-zero if the support of both functions overlap, such that:

$$\text{supp}(N_i) \cap \text{supp}(N_j) \neq \emptyset \tag{2.14}$$

Truncated B-splines have a smaller support than their original version. This directly results in a decrease of non-zero entries in the matrices $\mathbf{M}$ and $\mathbf{K}$. The sparsity of the truncated hierarchical basis and the hierarchical basis resulting from the example mesh shown in Figure 2b are given in Table 1. From these results it can be seen that the matrices constructed with the truncated hierarchical bases are sparser than for the hierarchical basis and are therefore computationally less expensive to compute. From these results it can also be seen that the difference in sparsity between the two methods decreases with a higher polynomial order p. Due to the significant increase of support of B-splines of higher polynomial orders, the effect of the truncation on the sparsity is less dominant in these cases.



(a) Hierarchical basis

(b) Truncated hierarchical basis

Figure 4: Sparsity patterns of the mass and stiffness matrix using hierarchical basis and truncated hierarchical basis with quadratic B-splines. It can be seen that the pattern of the hierarchical basis is denser than the truncated hierarchical basis. The black blocks indicate non-zero entries in the matrix.

Degree	Truncated hierarchical basis	Hierarchical basis	Ratio
1	1078	1472	0.73
2	3320	4342	0.76
3	5611	6397	0.88
4	7775	8113	0.96

Table 1: The number of non-zero elements in the mass and stiffness matrices associated with the mesh from Figure 2b for both bases are presented with different degrees. The matrices corresponding to the truncated hierarchical basis have a lower number of non-zero entries.

2.4.2 Stability

The condition number of a matrix is related to the numerical stability of the linear system. This quantity evaluates the sensitivity of the solution of the system to perturbations in the matrix. A matrix with a low condition number is less sensitive to perturbations than a matrix with a high condition number, and therefore generally more stable. With λ_{max} and λ_{min} defined as the maximum and minimum eigenvalues of a symmetric positive definite matrix $\mathbf{A}$, the condition number is calculated as:

$$\kappa(\mathbf{A}) = \frac{\lambda_{max}}{\lambda_{min}} \tag{2.15}$$

The lowerbound of λ_{max} and the upperbound of λ_{min} are related to the maximal and minimal diagonal values of matrix $\mathbf{A}$ respectively, such that:

$$\begin{aligned} \lambda_{max} &\gtrsim \max(A_{ii}) \\ \lambda_{min} &\lesssim \min(A_{ii}) \end{aligned} \tag{2.16}$$

The mass matrix $\mathbf{M}$ is known to always be positive definite, while the stiffness matrix $\mathbf{K}$ is positive semi-definite, due to the C^{-1}-continuity on the boundary of the domain. A symmetric positive definite stiffness matrix is obtained when the basis functions which comply to this C^{-1}-continuity on the boundary are omitted, meaning they are not included in the construction of the matrix in Equation (2.13). In the example bases shown in Figure 1c and 1d, the most left and most right B-spline would be omitted.

The diagonal values of the mass matrix are calculated with $M_{ii} = \int_\Omega N_i^2 \, d\Omega$. When assuming that the maximum diagonal value is related to the coarsest basis function in the basis N_i^1, while the minimum value is related to the finest basis function $N_i^N = \text{trunc}(N_i^N)$, it can be shown that:

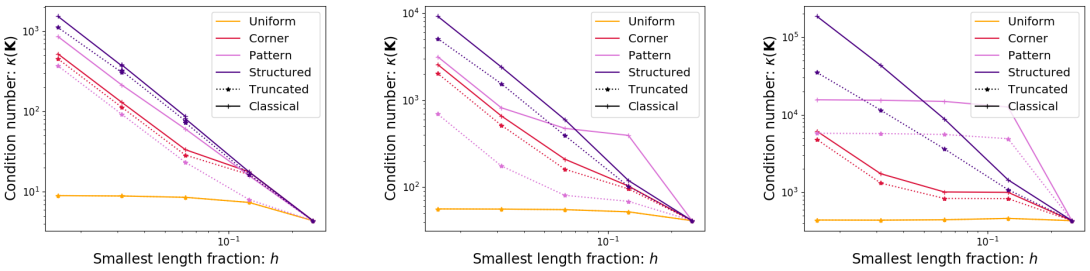
$$\kappa(\mathbf{M}^{\mathcal{T}}) \sim \frac{\int_\Omega \text{trunc}(N_i^1)^2 \, d\Omega}{\int_\Omega (N_i^N)^2 \, d\Omega} \qquad \kappa(\mathbf{M}^{\mathcal{H}}) \sim \frac{\int_\Omega (N_i^1)^2 \, d\Omega}{\int_\Omega (N_i^N)^2 \, d\Omega} \tag{2.17}$$

By definition of the truncation operation it holds that $\int_\Omega N_i^2 \, d\Omega \geq \int_\Omega \text{trunc}(N_i)^2 \, d\Omega$, because the truncated B-splines are constructed with a reduced number of finer B-splines and have a reduces support. From this definition it can be inferred that:

$$\kappa(\mathbf{M}^{\mathcal{T}}) \lesssim \kappa(\mathbf{M}^{\mathcal{H}}) \tag{2.18}$$

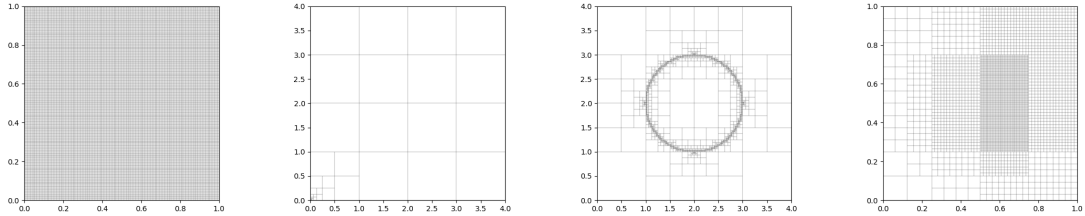
Note that this proof is not applicable in a general sense, but should be considered as a guideline. This implies that the condition number of the mass matrix is generally lower for the truncated hierarchical basis than for the hierarchical basis. This statement is supported by results shown in Figure 5, where the condition number for different meshes under refinement is shown for a varying polynomial degree. Different mesh refinement patterns are used, as shown in Figure 6. For all of the refinement patterns, except for the uniform refinement, it can be seen that the condition number of the truncated hierarchical basis is lower than that of the hierarchical basis. For uniform refinement, it holds that $\kappa(\mathbf{M}^{\mathcal{T}}) = \kappa(\mathbf{M}^{\mathcal{H}})$, because the bases are identical.

The estimate (2.18) presented for the mass matrix does not apply to the stiffness matrix $\mathbf{K}$. The results for the condition numbers under refinement are shown in Figure 7. From these results it can be seen that no clear distinction between the truncated hierarchical basis and the hierarchical basis can be made. For a low polynomial degree, the standard hierarchical basis has a lower condition number, while the truncated hierarchical basis has a lower condition number for higher polynomial degree.
From the performance analysis, it is concluded that the THB-splines are generally preferred over HB-splines with respect to sparsity. Due to the reduction of the support of truncated B-splines, their corresponding sparsity is decreased. The stability study shows that the condition number of the mass matrix is generally better when using THB-splines. For the stiffness matrix, no clear preference



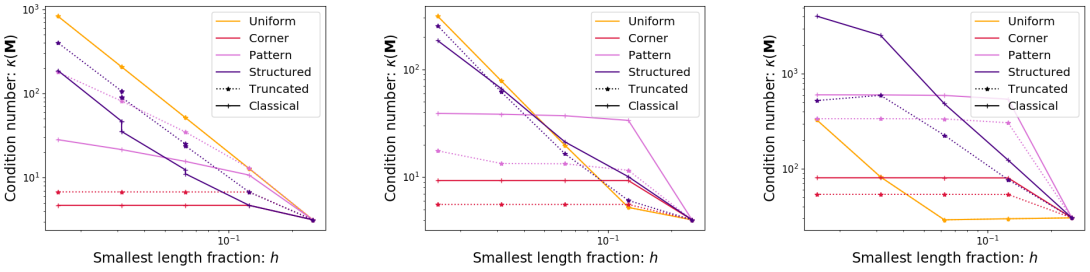
(a) Polynomial degree $p = 1$ **(b)** Polynomial degree $p = 2$ **(c)** Polynomial degree $p = 3$

Figure 5: Condition numbers of the mass matrix $\mathbf{M}$ for different refinements



(a) Uniform **(b)** Corner **(c)** Pattern **(d)** Structured

Figure 6: Resulting meshes for different refinements. For each strategy, the maximum hierarchical level is $N = 5$



(a) Polynomial degree $p = 1$ **(b)** Polynomial degree $p = 2$ **(c)** Polynomial degree $p = 3$

Figure 7: Condition numbers of the stiffness matrix $\mathbf{K}$ for different refinements

can be shown. Finally it is to be noted that partition of unity is only preserved in the truncated hierarchical basis. THB-splines allow for affine transformations, making them the preferred basis for local refinements from the CAD perspective.

3 Adaptivity methods for steady heat conduction

In the previous section the construction of a truncated hierarchical B-splines is evaluated, which are used on locally refined meshes. A natural choice of locally refining the mesh is by using adaptivity methods. The aim of adaptivity methods is to obtain a sufficiently accurate solution with the minimum amount of computational effort. These methods are based on finding local information about the error, such that refinements can be carried out only in the relevant parts of the mesh. This local information is expressed in the form of indicators η_K corresponding to an element $K \in \mathcal{Q}$.

In this work, two different methods are evaluated. The residual-based adaptivity method uses the residual to estimate the local contributions to the error of the solution. Goal-oriented adaptivity seeks to optimize the mesh according to a chosen quantity of interest, rather than the error of the solution itself.

3.1 Problem formulation

Both methods are here demonstrated in the context of steady heat conduction. The steady heat conduction is defined by the he Laplace equations. Let $\Omega \subset \mathbb{R}^d$ be the domain with boundary $\partial\Omega$. The Neumann boundary Γ_N and Dirichlet boundary Γ_D are defined such that $\overline{\Gamma_N} \cup \overline{\Gamma_D} = \partial\Omega$ and $\Gamma_N \cap \Gamma_D = \emptyset$. Given a source term f, Neumann data g and Dirichlet data u_D, the temperature u is defined by the equations:

$$\begin{aligned} \Delta u &= f && \text{in } \Omega \\ \nabla u \cdot \boldsymbol{n} &= g && \text{on } \Gamma_N \\ u &= u_D && \text{on } \Gamma_D \end{aligned} \tag{3.1}$$

where the normal vector $\boldsymbol{n}$ points out of the domain. This set of equations is referred to as the strong form of the problem. The elaboration of the equations is done with the simplification given that $f = 0$. The weak form of the the Laplace problem, defined on Ω, is given as:

$$(W) := \begin{cases} \text{Find } u \in \mathcal{U}, \text{ such that:} \\ \int_\Omega \nabla u \cdot \nabla v \, d\Omega = \int_{\Gamma_N} g v \, d\Gamma \qquad \forall v \in \mathcal{V} \end{cases} \tag{3.2}$$

with $\mathcal{U} = \{u \,|\, u \in H^1(\Omega), u = u_D \text{ on } \Gamma_D\}$ and $\mathcal{V} = \{v \,|\, v \in H^1(\Omega), w = 0 \text{ on } \Gamma_D\}$. The coercive bi-linear operator $\mathcal{B}(u,v) = \int_\Omega \nabla u \cdot \nabla v \, d\Omega$ and linear operator $\mathcal{F}(v) = \int_{\Gamma_N} g v \, d\Gamma$ are defined corresponding to this problem. By adapting a Galerkin discretization, the discretized trial and test functions are respectively defined as:

$$\begin{aligned} u^h &\in \mathcal{U}^h = \text{span}(\{N_i^p\}_{i=1}^n) \cap \mathcal{U} \\ v^h &\in \mathcal{V}^h = \text{span}(\{N_i^p\}_{i=1}^n) \cap \mathcal{V} \end{aligned} \tag{3.3}$$

The superscript h denotes the association with the mesh $\mathcal{Q}$, characterized with its elements length scale h and p is the polynomial degree of the B-splines. The discretized weak formulation is written as:

$$(G) := \begin{cases} \text{Find } u^h \in \mathcal{U}^h \text{ such that} \\ \mathcal{B}(u^h, v^h) = \mathcal{F}(v^h) \qquad \forall v^h \in \mathcal{V}^h \end{cases} \tag{3.4}$$

3.2 Residual-based error estimation

In residual-based error estimation, the residual is used to approximate the error $u - u^h$. Using coercivity and the symmetry of the bilinear form $\mathcal{B}(u,v)$, it can be shown that there exists a constant c such that the upper-bound of the of the error is defined as:

$$\|u - u^h\|_{\mathcal{U}} \leq c \frac{\mathcal{B}(u - u^h, v)}{\|v\|_{\mathcal{V}}} \qquad \forall v^h \in \mathcal{V}^h \tag{3.5}$$

Here $\mathcal{B}(u - u^h, v)$ is defined as the residual, and by using Equation (3.2), it is rewritten as:

$$\mathcal{B}(u - u^h, v) = \mathcal{B}(u, v) - \mathcal{B}(u^h, v) = \mathcal{F}(v) - \mathcal{B}(u^h, v) = \mathcal{R}^h(v) \tag{3.6}$$

When evaluating the residual using the discretized test space $\mathcal{V}^h$, it can be shown with that the residual is zero. This property is referred to as Galerkin orthogonality, and can be proven by using the discrete weak form (3.4):

$$\mathcal{R}^h(v^h) = \mathcal{F}(v^h) - \mathcal{B}(u^h, v^h) = \mathcal{F}(v^h) - \mathcal{F}(v^h) = 0 \qquad \forall v^h \in \mathcal{V}^h \tag{3.7}$$

The residual $\mathcal{R}^h(v)$ cannot be computed directly, because of the unknown test function v. A local representation of the residual needs to be found, independent of the test functions. To this end, the interpolation operator of Clément et. al. [16] is defined as $\Pi^h : \mathcal{V} \to \mathcal{V}^h$. The interpolation of v is subject to the Galerkin orthogonality, such that $\mathcal{R}^h(v) = \mathcal{R}^h(v - \Pi^h v)$. The residual is evaluated over all elements $K \in \mathcal{Q}$, each with their interface ∂K in order to find localized information.

$$\mathcal{R}^h(v - \Pi^h v) = \sum_{K \in \mathcal{Q}} \Big[\int_{\partial K \cap \Gamma_N} g(v - \Pi^h v)\, d\Gamma - \int_K \nabla u^h \cdot \nabla (v - \Pi^h v)\, d\Omega \Big] \tag{3.8}$$

To find local terms that converge under the refinement of an element, the strong form is recovered by integration by parts. An upperbound of the residual is found by taking the absolute value of each part. This is done to prevent different terms from cancelling each other out:

$$\begin{aligned} \mathcal{R}^h(v - \Pi^h v) \leq \sum_{K \in \mathcal{Q}} \Big[& \int_{\partial K \cap \Gamma_N} \big| \underbrace{(g - \nabla u^h \cdot \boldsymbol{n})}_{\epsilon_{bound}} (v - \Pi^h v) \big|\, d\Gamma \\ & + \int_K \big| \underbrace{\Delta u^h}_{\epsilon_{int}} (v - \Pi^h v) \big|\, d\Omega + \int_{\partial K \setminus \Gamma} \big| \underbrace{[\![-\nabla u^h \cdot \boldsymbol{n}]\!]}_{\epsilon_{jump}} (v - \Pi^h v) \big|\, d\Gamma \Big] \end{aligned} \tag{3.9}$$

Here $[\![\nabla u^h \cdot \boldsymbol{n}]\!]$ denotes the jump of the normal derivative of u^h over the element interface $\partial K \cap \partial K'$ with $K' \in Q$ and $K' \neq K$. Note that in IGA this term is only non-zero over the patch interface if the polynomial degree is quadratic or higher, i.e. $p > 1$.

In Equation (3.9) the local residue terms ϵ_{bound}, ϵ_{int} and ϵ_{jump} are distinguished. These terms assess the approximation error locally. With $\| \cdot \|_{L^2(\Omega)}$ defined as the L^2-norm over Ω, the Cauchy-Schwarz inequality is applied to find separate norms for the local residual terms and $(v - \Pi^h v)$:

$$\begin{aligned} \mathcal{R}^h(v - \Pi^h v) \leq \sum_{K \in \mathcal{Q}} \Big[& \|\epsilon_{bound}\|_{L^2(\partial K \cap \Gamma_N)} \|(v - \Pi^h v)\|_{L^2(\partial K \cap \Gamma_N)} + \|\epsilon_{int}\|_{L^2(K)} \|(v - \Pi^h v)\|_{L^2(K)} \\ & + \|\epsilon_{jump}\|_{L^2(\partial K \setminus \Gamma)} \|(v - \Pi^h v)\|_{L^2(\partial K \setminus \Gamma)} \Big] \end{aligned} \tag{3.10}$$

By using the interpolant operators of Clément, it is shown in Ref. [16] that the following inequality holds for some constants $c_1, c_2 \in \mathbb{R}_{>0}$:

$$\begin{aligned} \|v - \Pi^h v\|_{L^2(K)} &\leq c_1 h_K \|v\|_{H^1(\widetilde{K})} \\ \|v - \Pi^h v\|_{L^2(\partial K)} &\leq c_2 h_K^{\frac{1}{2}} \|v\|_{H^1(\widetilde{K})} \end{aligned} \tag{3.11}$$

where $\widetilde{K}$ is a subdomain containing element K and all neighbouring elements. This inequality is used to introduce the length scale of the elements h_K to the localized formulation in (3.10).

$$\begin{aligned}\mathcal{R}^h(v-\Pi^h v) \leq c_3 \sum_{K\in\mathcal{Q}} \Big[\Big(& h_K^{\frac{1}{2}} \|\epsilon_{bound}\|_{L^2(\partial K\cap\Gamma_N)} + h_K \|\epsilon_{int}\|_{L^2(K)} \\ & + h_K^{\frac{1}{2}} \|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)} \Big) \|v\|_{H^1(\tilde{K})} \Big] \\ \leq c_4 \Big(\sum_{K\in\mathcal{Q}} \Big[& h_K^{\frac{1}{2}} \|\epsilon_{bound}\|_{L^2(\partial K\cap\Gamma_N)} + h_K \|\epsilon_{int}\|_{L^2(K)} \\ & + h_K^{\frac{1}{2}} \|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)} \Big]^2 \Big)^{\frac{1}{2}} \Big(\sum_{K\in\mathcal{Q}} \|v\|^2_{H^1(\tilde{K})} \Big)^{\frac{1}{2}}\end{aligned} \tag{3.12}$$

The sum of the local norms of v can be rewritten as the global norm, with a constant c_5.

$$\Big(\sum_{K\in\mathcal{Q}} \|v\|^2_{H^1(\tilde{K})} \Big)^{\frac{1}{2}} = c_5 \|v\|_{\mathcal{V}} \tag{3.13}$$

Substitution of the inequality of Equation (3.12) into Equation (3.5) with Equation (3.6) gives the upper-bound of the error expressed in the localized indicators, such that for a constant C:

$$\|u-u^h\|_{\mathcal{U}} \leq C \Big(\sum_{K\in\mathcal{Q}} \eta_K^2 \Big)^{\frac{1}{2}} \tag{3.14}$$

with

$$\eta_K = h_K^{\frac{1}{2}} \|(g-\nabla u^h\cdot\boldsymbol{n})\|_{L^2(\partial K\cap\Gamma_N)} + h_K \|\Delta u^h\|_{L^2(K)} + h_K^{\frac{1}{2}} \|[\![\nabla u^h\cdot\boldsymbol{n}]\!]\|_{L^2(\partial K\setminus\Gamma)} \tag{3.15}$$

The residual-based error indicators are assembled by different terms, corresponding to the error on the boundary conditions, the internal error and the error on the jump over elements respectively. The indicators give an accurate description of the local obedience of the approximate solution to the strong form equations.

3.3 Goal-oriented error estimation

Goal-oriented adaptivity is based on finding indicators related to the error in a chosen goal quantity $\mathcal{G}(u)$, instead of a norm of the approximation error $u-u^h$. An auxiliary problem, called the dual problem (D), is solved. Its solution z, element of test space $\mathcal{V}$ and often referred to as the dual solution or influence function, expresses the sensitivity of the problem goal quantity $\mathcal{G}(u)$. The dual problem is defined as:

$$(D) := \begin{cases} \text{Find } z\in\mathcal{V}, \text{ such that} \\ \mathcal{B}(v,z) = \mathcal{G}(v) \qquad \forall v\in\mathcal{V}, \end{cases} \tag{3.16}$$

An important property of the dual problem is that it is related to the primal problem by means of the primal-dual equivalence, expressed as:

$$\mathcal{G}(u) = \mathcal{B}(u,z) = \mathcal{F}(z) \tag{3.17}$$

In the event of a fully Dirichlet constrained problem, no Neumann data exists and $\mathcal{F}(v)=0$. From the primal-dual equivalence (3.17) it can be seen that in this case $\mathcal{G}(u)=0$, meaning there is no free choice for a goal quantity and goal-oriented adaptivity is not a suitable adaptivity method.

From the equivalence it can be shown that the residual evaluated over the dual solution is related to the error in the goal quantity.

$$\mathcal{G}(u)-\mathcal{G}(u^h) = \mathcal{B}(u,z)-\mathcal{B}(u^h,z) = \mathcal{F}(z)-\mathcal{B}(u^h,z) = \mathcal{R}^h(z) \tag{3.18}$$

Just as for the primal problem, the dual solution is found by solving the discretized dual problem. It can however be seen that when evaluating the dual residual, taking $z^h\in\mathcal{V}^h$ is a poor choice for

the discretization, because the residual would be zero due to the Galerkin orthogonality (Equation (3.7)). The discretized function space of the dual solution should be 'richer' in order to obtain a proper estimate. In Dédé et. al. 2012 [17], dual space enrichment by order elevation and uniformly refinement are compared. Herein it is concluded that order elevation is more efficient, while its approximation quality is comparable to uniformly refined dual spaces. Additionally, the order elevated dual problem can be evaluated on the same mesh $\mathcal{Q}$ as the primal problem, which is beneficial when evaluating mesh dependent quantities. The enriched dual space is defined as:

$$\mathcal{V}^{h,p+1} = \text{span}(\{N_i^{p+1}\}_{i=1}^n) \cap \mathcal{V} \tag{3.19}$$

Note that the enriched dual space is C^p-continuous, in contrast to the C^{p-1}-continuous primal space. After solving the primal and dual problem, the dual residual $\mathcal{R}^h(z^h)$ can directly be calculated. This is in contrast to the residual $\mathcal{R}^h(v)$ in Equation (3.6), which contains the unknown test function v. The dual residual is rewritten as a sum over all elements, such that local residual terms are found:

$$\mathcal{R}^h(z^h) = \sum_{K\in\mathcal{Q}} \Big[\int_{\partial K\cap\Gamma_N} \underbrace{g z^h}_{A}\, d\Gamma - \int_K \underbrace{\nabla u^h \cdot \nabla z^h}_{B}\, d\Omega \Big] \tag{3.20}$$

These local residual terms however give a poor representation of the error induced by these elements, because part A and B are not related to a local approximation inaccuracy. To find terms which express the local error more precisely, integration by parts is used to evaluate an alternative form, similar to the strong formulation:

$$\mathcal{R}^h(z^h) = \sum_{K\in\mathcal{Q}} \Big[\int_{\partial K\cap\Gamma_N} \underbrace{(g - \nabla u^h \cdot \boldsymbol{n})}_{\epsilon_{bound}} z^h + \int_K \underbrace{\Delta u^h}_{\epsilon_{int}} z^h + \int_{\partial K\setminus\Gamma} \underbrace{[\![-\nabla u^h \cdot \boldsymbol{n}]\!]}_{\epsilon_{jump}} z^h \Big] \tag{3.21}$$

Note that according to the strong form of the Laplace problem (3.1), it holds that $g - \nabla u \cdot \boldsymbol{n} = 0$ and $\Delta u = 0$. The residue terms ϵ_{bound}, ϵ_{int} and ϵ_{jump} measure the inaccuracy of the approximate solution u^h evaluated on the strong formulation. Therefore these terms evaluated over element K contain information of the local contribution to the error.

To find a term that locally represents the error in z^h, the interpolation operator I^h is introduced as an L^2-projection operator, such that $I^h : \mathcal{V}^{h,p+1} \to \mathcal{V}^{h,p}$. By Galerkin orthogonality it holds that $\mathcal{R}^h(I^h z) = 0$. This property is used to reformulate the dual residual as $\mathcal{R}^h(z^h - I^h z)$. The term $z^h - I^h z$ evaluates the error between z^h and its projection on a basis with a lower polynomial degree. Because the projection I^h becomes more precise with a decreasing element size, the error converges under mesh refinement. Because the operator I^h is constructed with a linear least squares projection, this error representation is lost when integrating over the elements. This is visualized in Figure 8. To prevent the loss of information on the error, the integration over the elements should be done using the absolute value. Therefore an upperbound of the residual $\mathcal{R}^h(z^h - I^h z)$ is found by:

$$\mathcal{R}^h(z^h - I^h z) \leq \sum_{K\in\mathcal{Q}} \Big[\int_{\partial K\cap\Gamma_N} \big|\epsilon_{bound}(z^h - I^h z)\big| + \int_K \big|\epsilon_{int}(z^h - I^h z)\big| + \int_{\partial K\setminus\Gamma} \big|\epsilon_{jump}(z^h - I^h z)\big| \Big] \tag{3.22}$$

	$\int_K (f - I^h f)$	$\int_K \lvert f - I^h f\rvert$
K_1	-0.00544	0.09697
K_2	0.01806	0.02197
K_3	-0.00338	0.00587

Table 2: While the mismatch between $f(x)$ and $I^h f(x)$ is largest in element K_1, this mismatch is not expressed in the first column. By taking the absolute value, this mismatch is clearly seen.

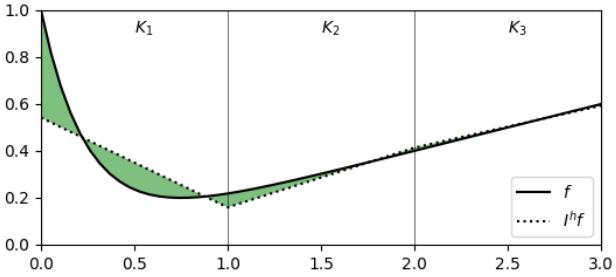


Figure 8: The error between $f(x)$ and its projection $I^h f(x)$, where $f(x) = e^{-4x} + \frac{1}{5}x$. The green areas indicate the difference between the two functions. The error and absolute error are given in Table 2.

By applying Cauchy-Schwarz [1], separated L^2-norms of the local residual terms and the dual terms are found. Due to this separation, the indicators are constructed with a part that is related to the primal solution and a part related to the dual solution.

$$\mathcal{G}(u) - \mathcal{G}(u^h) = \mathcal{R}^h(z^h - I^h z) \leq \sum_{K \in \mathcal{Q}} \eta_K \tag{3.23}$$

with

$$\begin{aligned} \eta_K = &\|(g - \nabla u^h \cdot \boldsymbol{n})\|_{L^2(\partial K \cap \Gamma_N)} \|z^h - I^h z\|_{L^2(\partial K \cap \Gamma_N)} + \|\Delta u^h\|_{L^2(K)} \|z^h - I^h z\|_{L^2(K)} \\ &+ \|[\![\nabla u^h \cdot \boldsymbol{n}]\!]\|_{L^2(\partial K \setminus \Gamma)} \|z^h - I^h z\|_{L^2(\partial K \setminus \Gamma)} \end{aligned} \tag{3.24}$$

Compared to the residual-based indicators from Equation (3.15), the element size h_K is replaced with a term related to the error in the dual solution. These terms add extra weight to the indicators, based on the error in the dual solution. This way the influence of the element to the error in the goal-quantity is included in the indicators. Additionally, the inequality holds without an unknown constant C.

3.4 Numerical example: Laplace on an L-shaped domain

To evaluate the effectivity of both adaptivity methods, a numerical experiment, based on a Laplace case defined on an L-shaped domain with a known solution [2], is conducted. The sharp corner in the domain induces a singularity, which will cause most of the approximation error. The convergence of the exact error $\|u - u^h\|$ and the exact error in the quantity of interest $\mathcal{G}(u) - \mathcal{G}(u^h)$ are studied under refinement using residual-based and goal-oriented error estimation.

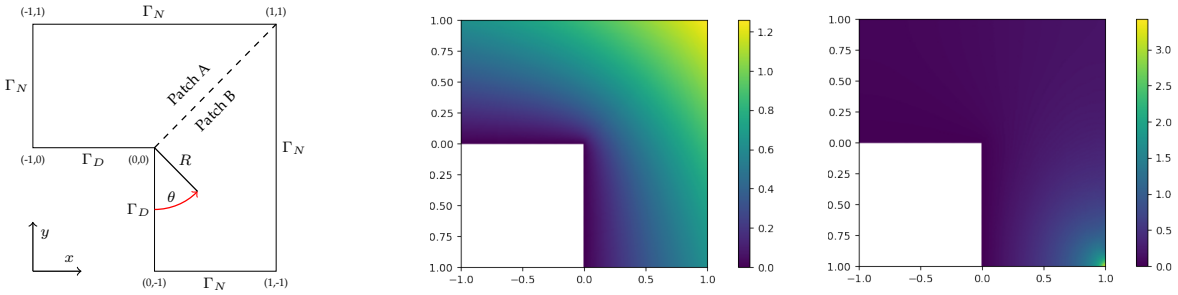
3.4.1 Problem formulation

Let Ω be a L-shaped domain with the dimensions defined in Figure 9a. The mesh $\mathcal{Q}$ is constructed with two patches, with the patches interface between (0,0) and (1,1). By defining polar coordinates $\theta = \tan(\frac{y}{x}) \in [0, 1\frac{1}{2}\pi]$ and $R = (x^2 + y^2)^{\frac{1}{2}} \in [0, \sqrt{2}]$, the exact solution is given as:

$$u(x, y) = R^{\frac{2}{3}} \sin\left(\frac{2}{3}\theta\right) \tag{3.25}$$

The exact solution is shown in Figure 9b. The Laplace equations given in Equation (3.1) are utilized, with on the Dirichlet boundary $u_D = 0$ and the Neumann data g constructed according to the exact solution.

[1] In this work the Cauchy-Schwarz inequality is used to find separate L^2-norms for the local residual terms and dual terms. By Hölders inequality it can be seen that other sets of norms are possible, which might improve the quality of the indicators.



(a) L-shaped domain

(b) Exact solution $u(x, y)$

(c) Dual solution $z^h(x, y)$

Figure 9: In Figure **(a)** the aspects of the domain are shown. It can be seen that the conforming mesh consists of two patches. Figures **(b)** and **(c)** show the primal solution $u(x, y)$ and dual solution $z(x, y)$ respectively.

3.4.2 Results

The problem is solved according to the weak formulation as is defined in Equation (3.2) with quadratic B-splines. For residual-based adaptivity, the indicators as defined in Equation (3.15) are used. In Figure 12 an example of these indicators evaluated after the second refinement are shown.

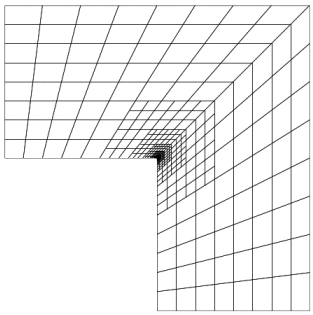
For each refinement step, all elements K for which $\eta_K \geq \lambda \max(\eta)$ are marked for refinement, where the marking fraction is chosen between $0 \leq \lambda \leq 1$. A maximum refinement level N_{max} is set to bound the number refinement of iterations. The recursive refinement algorithm is stopped if all elements for which $\eta_K \geq \lambda \max(\eta)$ are of the maximum refinement level N_{max}. The resulting mesh with $\lambda = 0.5$ and $N_{max} = 10$ is shown in Figure 10a.

According to the construction algorithm with which the truncated hierarchical B-splines are created, the refinement of a single element does not necessarily result in additional basis functions in the basis $\mathcal{T}$. Therefore not only the element that is marked is refined, but the total support of all basis functions that have support on element K is refined. This refinement strategy is used for all cases considered in this work.

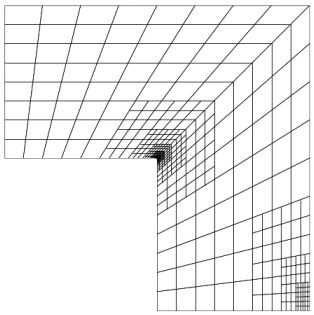
As a quantity of interest for the goal-oriented refinement, the value in the lower right corner $(1, -1)$ is defined, such that $\mathcal{G}(u) = u(1, -1)$. The dual problem is solved according to Equation (3.16). Note that the problem is solved with an order elevated function space, with cubic B-splines. The approximated dual solution $z^h(x, y)$ is shown in Figure 9c. The dual problem can be interpreted as a Laplace problem with a point source in the corner. The indicators are evaluated following Equation (3.24) and an example of the indicators after the second refinement is shown in Figure 12. The resulting mesh with $\lambda = 0.5$ and $N_{max} = 10$ is shown in Figure 10b.

The convergence behavior of the L^2-norm of the exact error and the error in the quantity of interest $\mathcal{G}(u) - \mathcal{G}(u^h)$ under refinement are shown in Figure 11. The convergence rates are estimated based on simulation data. Comparison of these rates to theoretical results is not considered here. This holds for the convergence rates in the remainder of this thesis. From Figure 11 it can be seen that both methods significantly improve the convergence rate compared to uniform refinement. Note that due to the singularity at the corner, uniform refinement is far from optimal.

An interesting result can be seen in Figure 11b, where it is shown that the convergence rate in the error of the goal quantity is higher for the residual-based refinement than for the goal-oriented refinement. This counter-intuitive result is caused by the domination of the singularity point at (0,0) on the approximation error in the whole domain. Applying refinements on this point does not only reduce the error on that point, but all over the domain. By means of the dual solution, extra weight is added to the goal-oriented indicators to refine in the corner (1,-1). In particular this case, this appears not to be the most optimal choice of refinement.

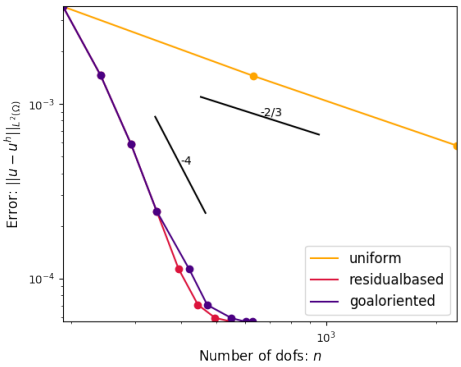


(a) Residual-based mesh after 9 refinements

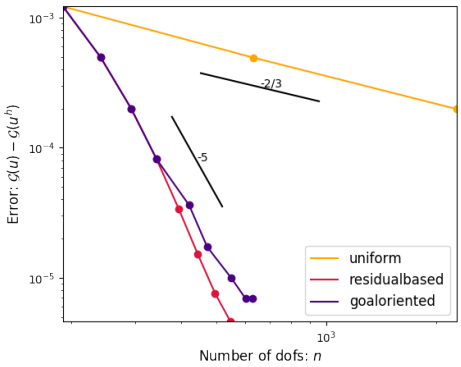


(b) Goal-oriented mesh after 11 refinements

Figure 10: Locally refined meshes from residual-based and goal-oriented adaptivity methods. For both meshes a marking fraction of $\lambda = 0.5$ is used.

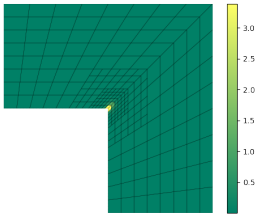


(a) L^2-norm error $\|u - u^h\|_{L^2(\Omega)}$

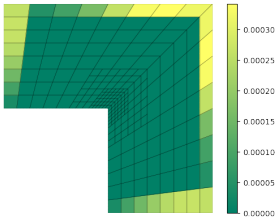


(b) Error in the quantity of interest $\mathcal{G}(u) - \mathcal{G}(u^h)$

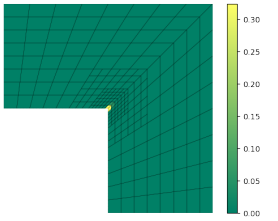
Figure 11: Convergence of the L^2-norm of the exact error in u and the error in the quantity of interest. For the discretization of u^h, quadratic B-splines are used.



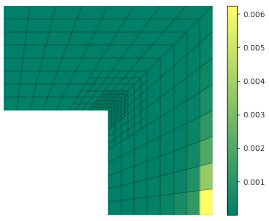
(a) $\|\epsilon_{int}\|_{L^2(K)}$



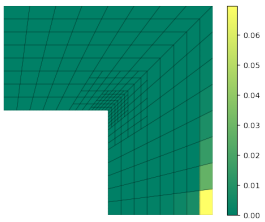
(b) $\|\epsilon_{bound}\|_{L^2(\partial K \cap \Gamma_N)}$



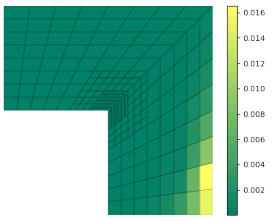
(c) $\|\epsilon_{jump}\|_{L^2(\partial K \setminus \Gamma)}$



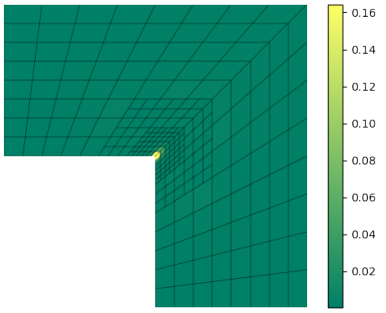
(d) $\|z^h - I^h z\|_{L^2(K)}$



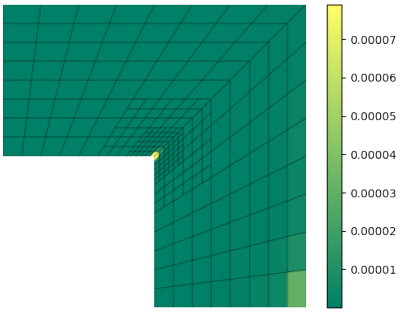
(e) $\|z^h - I^h z\|_{L^2(\partial K \cap \Gamma_N)}$



(f) $\|z^h - I^h z\|_{L^2(\partial K \setminus \Gamma)}$



(g) Residual-based indicators η_K



(h) Goal-oriented method indicators η_K

Figure 12: The indicators evaluated after the second refinement iteration are shown. The terms from **(a)**, **(b)** and **(c)** are used to construct the residual-based indicators as well as the goal-oriented indicators. Note that they are exactly the same at this point, only because the mesh is identical for both cases. These terms are multiplied with h_K or $h_K^{\frac{1}{2}}$ for the residual-based indicators, while they are multiplied element-wise with the terms from **(d)**, **(e)** and **(f)** respectively to compute the goal-oriented indicators. It can be seen that the residual-based indicators in **(g)** purely focus on the singularity point on (0,0), while the goal-oriented indicators in **(h)** also focus on the point of interest in (1,-1).

4 Steady viscous flow analysis

In this section, the adaptivity setting as defined in the previous section will be elaborated for the Stokes equations, simulating an incompressible steady viscous flow. The derivation of the residual-based indicators and goal-oriented indicators is elaborated.

4.1 Problem formulation

The flow in the domain $\Omega \in \mathbb{R}^d$ is described by the Stokes equations:

$$\begin{aligned} -\nabla \cdot (2\mu \nabla^s \boldsymbol{u}) + \nabla p &= \boldsymbol{f} \qquad \text{in } \Omega \\ \nabla \cdot \boldsymbol{u} &= 0 \qquad \text{in } \Omega \\ 2\mu \nabla^s \boldsymbol{u} \cdot \boldsymbol{n} - p\boldsymbol{n} &= \boldsymbol{g} \qquad \text{on } \Gamma_N \\ \boldsymbol{u} &= \boldsymbol{u}_D \qquad \text{on } \Gamma_D \end{aligned} \tag{4.1}$$

with velocity $\boldsymbol{u}$, pressure p, constant viscosity μ, body force $\boldsymbol{f}$ and Neumann data $\boldsymbol{g}$. The weak form of the Stokes problem is defined as:

$$(W) = \begin{cases} \text{Find } (\boldsymbol{u}, p) \in \mathcal{U}_{\boldsymbol{u}_D} \times \mathcal{V}, \text{ such that} \\ \int_\Omega \big(2\mu \nabla^s \boldsymbol{u} : \nabla^s \boldsymbol{v} - p \nabla \cdot \boldsymbol{v}\big)\, d\Omega = \int_\Omega \boldsymbol{f} \cdot \boldsymbol{v}\, d\Omega + \int_{\Gamma_N} \boldsymbol{g} \cdot \boldsymbol{v}\, d\Gamma \qquad \forall \boldsymbol{v} \in \mathcal{U}_0 \\ \int_\Omega \big(-q \nabla \cdot \boldsymbol{u}\big)\, d\Omega = 0 \qquad \forall q \in \mathcal{V} \end{cases} \tag{4.2}$$

with the function spaces defined as:

$$\mathcal{U}_{\boldsymbol{u}_D} = \{\boldsymbol{v} \in [H^1(\Omega)]^d : \boldsymbol{v} = \boldsymbol{u}_D \text{ on } \Gamma_D\} \qquad \mathcal{V} = \{q \in L^2(\Omega)\} \tag{4.3}$$

Let $(\boldsymbol{u}^h, p^h)$ and $(\boldsymbol{v}^h, q^h)$ be the discretized trial and test functions respectively, discretized by using Taylor-Hood elements for B-splines. The B-splines used to discretize the pressure field are of order $p-1$, while the B-splines of the velocity field are of order p. Both spaces are C^{p-2}-continuous in the case of Taylor Hood elements. The discretized function spaces are defined as:

$$\begin{aligned} \mathcal{U}^h_{\boldsymbol{u}_D} &= \text{span}(\{\boldsymbol{N}^p_i\}_{i=1}^n) \cap \mathcal{U}_{\boldsymbol{u}_D} \\ \mathcal{V}^h &= \text{span}(\{N^{p-1}_i\}_{i=1}^n) \cap \mathcal{V} \end{aligned} \tag{4.4}$$

The bilinear form $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v})(p, q)$ and linear operator $\mathcal{F}(\boldsymbol{v})$ are defined such that the discretized weak form can be written as:

$$(G) := \begin{cases} \text{Find } (\boldsymbol{u}^h, p^h) \in \mathcal{U}^h_{\boldsymbol{u}_D} \times \mathcal{V}^h, \text{ such that} \\ \mathcal{L}(\boldsymbol{u}^h, \boldsymbol{v}^h)(p^h, q^h) = \mathcal{F}(\boldsymbol{v}^h) \qquad \forall \boldsymbol{v}^h \in \mathcal{U}^h_0,\, \forall q^h \in \mathcal{V}^h \end{cases} \tag{4.5}$$

4.2 Residual-based adaptivity for Stokes

The elaboration of the residual-based error estimators for the Stokes problem is comparable to the elaboration for the Laplace problem, with the exception of the coupled solutions $(\boldsymbol{u}, p)$ instead of the single solution u. For more information on this elaboration, the reader is referred to [1] and [18]. Recalling the coercivity of the bilinear form $\mathcal{L}(\boldsymbol{u}, \boldsymbol{v})(p, q)$ and considering Galerkin orthogonality, the upper bound of the error in $(\boldsymbol{u}, p)$ is defined as:

$$\big(\|\boldsymbol{u}\|_{\mathcal{U}_g} + \|p\|_{\mathcal{V}}\big) \leq C \frac{\mathcal{L}(\boldsymbol{u} - \boldsymbol{u}^h, \boldsymbol{v})(p - p^h, q)}{\big(\|\boldsymbol{v}\|_{\mathcal{U}_0} + \|q\|_{\mathcal{V}}\big)} \tag{4.6}$$

The residual is defined as $\mathcal{R}^h(\boldsymbol{v}, q) = \mathcal{L}(\boldsymbol{u} - \boldsymbol{u}^h, \boldsymbol{v})(p - p^h, q)$. By evaluating the residual over the elements and deriving the strong formulation, the residual is written as:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{v},q) &= \int_\Omega \left(\boldsymbol{f}\cdot\boldsymbol{v} - 2\mu\nabla^s\boldsymbol{u}^h : \nabla^s\boldsymbol{v} + p^h\nabla\cdot\boldsymbol{v} + q\nabla\cdot\boldsymbol{u}^h\right) d\Omega + \int_{\Gamma_N} \boldsymbol{g}\cdot\boldsymbol{v}\, d\Gamma \\ &= \sum_{K\in\mathcal{Q}} \Big[\int_K \underbrace{\left(\boldsymbol{f} + \nabla\cdot(2\mu\nabla^s\boldsymbol{u}^h) - \nabla p^h\right)}_{\epsilon_{moment}}\cdot\boldsymbol{v}\, d\Omega - \int_K q\underbrace{\nabla\cdot\boldsymbol{u}^h}_{\epsilon_{incomp}}\, d\Omega \\ &\quad - \int_{\partial K\setminus\Gamma} \underbrace{[\![2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} - p\boldsymbol{n}]\!]}_{\epsilon_{jump}}\cdot\boldsymbol{v}\, d\Gamma + \int_{\partial K\cap\Gamma_N} \underbrace{\left(\boldsymbol{g} - 2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} + p^h\boldsymbol{n}\right)}_{\epsilon_{bound}}\cdot\boldsymbol{v}\, d\Gamma\Big]\end{aligned} \tag{4.7}$$

By using the interpolation operator $\Pi^h : \mathcal{U}_0 \to \mathcal{U}_0^h$, it holds that $\mathcal{R}^h(\Pi^h\boldsymbol{v},0) = 0$ due to Galerkin orthogonality. The Cauchy-Schwarz inequality is applied to the adapted residual $\mathcal{R}^h(\boldsymbol{v}-\Pi^h\boldsymbol{v},q)$, such that:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{v}-\Pi^h\boldsymbol{v},q) \le \sum_{K\in\mathcal{Q}} \Big[&\|\epsilon_{moment}\|_{L^2(K)}\|\boldsymbol{v}-\Pi^h\boldsymbol{v}\|_{L^2(K)} + \|\epsilon_{incomp}\|_{L^2(K)}\|q\|_{L^2(K)} \\ &+ \|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)}\|\boldsymbol{v}-\Pi^h\boldsymbol{v}\|_{L^2(\partial K\setminus\Gamma)} + \|\epsilon_{bound}\|_{L^2(K)}\|\boldsymbol{v}-\Pi^h\boldsymbol{v}\|_{L^2(\partial K\cap\Gamma_N)}\Big]\end{aligned} \tag{4.8}$$

The inequality defined in Equation (3.11), based on interpolation operators of Clément, is applied to get an upper bound of the parts containing the term $\boldsymbol{v}-\Pi^h\boldsymbol{v}$. With constants c_i it is shown that:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{v}-\Pi^h\boldsymbol{v},q) &\le \sum_{K\in\mathcal{Q}} \Big[c_1 h_K\|\epsilon_{moment}\|_{L^2(K)}\|\boldsymbol{v}\|_{H^1(\tilde{K})} + \|\epsilon_{incomp}\|_{L^2(K)}\|q\|_{L^2(K)} \\ &\quad + c_2 h_K^{\frac{1}{2}}\|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)}\|\boldsymbol{v}\|_{H^1(\tilde{K})} + c_3 h_K^{\frac{1}{2}}\|\epsilon_{bound}\|_{L^2(K)}\|\boldsymbol{v}\|_{H^1(\tilde{K})}\Big] \\ &\le \sum_{K\in\mathcal{Q}} c_4\Big[\big(h_K\|\epsilon_{moment}\|_{L^2(K)}\| + \|\epsilon_{incomp}\|_{L^2(K)} + h_K^{\frac{1}{2}}\|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)} \\ &\quad + h_K^{\frac{1}{2}}\|\epsilon_{bound}\|_{L^2(K)}\big)\big(\|\boldsymbol{v}\|_{H^1(\tilde{K})} + \|q\|_{L^2(K)}\big)\Big]\end{aligned} \tag{4.9}$$

By using the discrete Cauchy-Schwarz inequality on the latter inequality, the sum is split in a sum containing the terms related to the trail functions and a sum containing the test functions.

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{v}-\Pi^h\boldsymbol{v},q) \le &\Big(\sum_{K\in\mathcal{Q}} c_4\Big[h_K\|\epsilon_{moment}\|_{L^2(K)} + \|\epsilon_{incomp}\|_{L^2(K)} + h_K^{\frac{1}{2}}\|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)} \\ &+ h_K^{\frac{1}{2}}\|\epsilon_{bound}\|_{L^2(K)}\Big]^2\Big)^{\frac{1}{2}}\Big(\sum_{K\in\mathcal{Q}}\Big[\|\boldsymbol{v}\|_{H^1(\tilde{K})} + \|q\|_{L^2(K)}\Big]^2\Big)^{\frac{1}{2}}\end{aligned} \tag{4.10}$$

An upper bound for the term containing the test functions can be defined, such that with a constant c_5:

$$\Big(\sum_{K\in\mathcal{Q}}\Big[\|\boldsymbol{v}\|_{H^1(\tilde{K})} + \|q\|_{L^2(K)}\Big]^2\Big)^{\frac{1}{2}} \le c_5\big(\|\boldsymbol{v}\|_{\mathcal{U}_0} + \|q\|_{\mathcal{V}}\big) \tag{4.11}$$

Now by substitution of this inequality into Equation (4.10) it can be found that for a constant C:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{v}-\Pi^h\boldsymbol{v},q) \le C\Big(&\sum_{K\in\mathcal{Q}}\Big[h_K\|\epsilon_{moment}\|_{L^2(K)} + \|\epsilon_{incomp}\|_{L^2(K)} + h_K^{\frac{1}{2}}\|\epsilon_{jump}\|_{L^2(\partial K\setminus\Gamma)} \\ &+ h_K^{\frac{1}{2}}\|\epsilon_{bound}\|_{L^2(K)}\Big]^2\Big)^{\frac{1}{2}}\big(\|\boldsymbol{v}\|_{\mathcal{U}_0} + \|q\|_{\mathcal{V}}\big)\end{aligned} \tag{4.12}$$

The upper bound of the residual can be substituted in (4.6) to find the set of residual-based indicaters as:

$$\big(\|\boldsymbol{u}\|_{\mathcal{U}_g} + \|p\|_{\mathcal{V}}\big) \leq C\big(\sum_{K\in\mathcal{Q}} \eta_K^2\big)^{\frac{1}{2}} \tag{4.13}$$

with

$$\begin{aligned}\eta_K =& h_K\|\boldsymbol{f} + \nabla\cdot(2\mu\nabla^s\boldsymbol{u}^h) - \nabla p^h\|_{L^2(K)} + \|\nabla\cdot\boldsymbol{u}^h\|_{L^2(K)} \\ &+ h_K^{\frac{1}{2}}\|[\![2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} - p\boldsymbol{n}]\!]\|_{L^2(\partial K\setminus\Gamma)} + h_K^{\frac{1}{2}}\|\boldsymbol{g} - 2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} + p^h\boldsymbol{n}\|_{L^2(\partial K\cap\Gamma_N)}\end{aligned} \tag{4.14}$$

The indicators are constructed with terms that are related to the inaccuracy of the approximate solutions in obeying the strong form equations. Therefore they represent the error induced by the element on which they are evaluated.

4.3 Goal-oriented adaptivity for Stokes

In Section 3.3 the elaboration of the goal-oriented error indicators for the Laplace problem is explained. The elaboration for the Stokes problem is similar, with the difference that it is a coupled problem. Therefore the dual problem is a coupled problem as well, and the goal quantity should be chosen in such a way that the dual problem is well-defined. Other applications of goal-oriented adaptivity on the Navier-Stokes equations can be found in [19] and [20].

Let $\mathcal{G}(\boldsymbol{u},p)$ be the goal quantity to the Stokes problem. By using the symmetry of the bilinear form $\mathcal{L}(\boldsymbol{u},\boldsymbol{v})(p,q)$, the dual problem is solved to find the dual-velocity $\boldsymbol{z}$ and the dual-pressure s. The dual problem is formulated as:

$$(D) = \begin{cases} \text{Find } (\boldsymbol{z},s)\in\mathcal{U}_0\times\mathcal{V}, \text{ such that} \\ \mathcal{L}(\boldsymbol{v},\boldsymbol{z})(q,s) = \mathcal{G}(\boldsymbol{v},q) \qquad \forall\boldsymbol{v}\in\mathcal{U}_0, \quad \forall q\in\mathcal{V} \end{cases} \tag{4.15}$$

An important property of the dual problem is that the primal-dual equivalence holds, such that $\mathcal{G}(\boldsymbol{u},p) = \mathcal{L}(\boldsymbol{v},\boldsymbol{z})(q,s) = \mathcal{F}(\boldsymbol{z})$. This equivalence essentially states that the goal quantity is related to the primal and dual solution. From the weak problem definition, it can be shown that this equivalence holds:

$$\mathcal{G}(\boldsymbol{v},q) = \int_\Omega \big(-2\mu\nabla^s\boldsymbol{v} : \nabla^s\boldsymbol{z} + q\nabla\cdot\boldsymbol{z} + s\nabla\cdot\boldsymbol{v}\big)\,d\Omega = \int_\Omega \boldsymbol{f}\cdot\boldsymbol{z}\,d\Omega + \int_{\Gamma_N}\boldsymbol{g}\cdot\boldsymbol{z}\,d\Gamma \tag{4.16}$$

From the equivalence it can be seen that no goal quantity can be chosen if $\mathcal{F}(\boldsymbol{z}) = 0$, which is the case if there is no body force and the boundaries are purely Dirichlet, such that $\boldsymbol{f} = \boldsymbol{0}$ and $\boldsymbol{g} = \boldsymbol{0}$. For this case, residual-based adaptivity is suggested. The residual in the goal quantity is defined as $\mathcal{R}^h(\boldsymbol{z},s) = \mathcal{G}(\boldsymbol{u},p) - \mathcal{G}(\boldsymbol{u}^h,p^h)$. Order elevation is used to discretize the function spaces, such that $(\boldsymbol{z}^h,s^h)\in\mathcal{U}_0^{h,p+1}\times\mathcal{V}^{h,p+1}$. For the coupling of this mixed problem, Taylor-Hood elements are used. Let I^h be an L^2-projection operator. Due to Galerkin orthogonality, the residual can be equipped with $I^h\boldsymbol{z}$ and I^hs, such that:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{z}^h - I^h\boldsymbol{z}, s - I^hs) = \sum_{K\in\mathcal{Q}}\Big[\int_K \Big(\boldsymbol{f}\cdot(\boldsymbol{z}^h - I^h\boldsymbol{z}) - 2\mu\nabla^s\boldsymbol{u}^h : \nabla^s(\boldsymbol{z}^h - I^h\boldsymbol{z}) + p^h\nabla\cdot(\boldsymbol{z}^h - I^h\boldsymbol{z})\Big)\,d\Omega \\ + \int_K \nabla\cdot\boldsymbol{u}^h(s^h - I^hs)\,d\Omega + \int_{\partial K\cap\Gamma_N}\boldsymbol{g}\cdot(\boldsymbol{z} - I^h\boldsymbol{z})\,d\Gamma\Big]\end{aligned} \tag{4.17}$$

Integration by parts is used to evaluate terms that represent the inaccuracy in the approximation:

$$\begin{aligned}\mathcal{R}^h(\boldsymbol{z}^h - I^h\boldsymbol{z}, s - I^h s) = \sum_{K\in\mathcal{Q}} \Big[&\int_K \big(\boldsymbol{f} + \nabla\cdot(2\mu\nabla^s\boldsymbol{u}^h) - \nabla p^h\big)\cdot(\boldsymbol{z}^h - I^h\boldsymbol{z})\,d\Omega \\ &- \int_K \nabla\cdot\boldsymbol{u}^h(s^h - I^h s)\,d\Omega - \int_{\partial K\setminus\Gamma} [\![2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} - p\boldsymbol{n}]\!]\cdot(\boldsymbol{z}^h - I^h\boldsymbol{z})\,d\Gamma \\ &+ \int_{\partial K\cap\Gamma_N} \big(\boldsymbol{g} - 2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} + p^h\boldsymbol{n}\big)\cdot(\boldsymbol{z}^h - I^h\boldsymbol{z})\,d\Gamma\Big]\end{aligned} \tag{4.18}$$

By applying the Cauchy-Schwarz inequality, the goal-oriented indicators are found by:

$$\mathcal{R}^h(\boldsymbol{z}^h - I^h\boldsymbol{z}, s - I^h s) \leq \sum_{K\in\mathcal{Q}} \eta_K \tag{4.19}$$

with

$$\begin{aligned}\eta_K =& \|\boldsymbol{f} + \nabla\cdot(2\mu\nabla^s\boldsymbol{u}^h) - \nabla p^h\|_{L^2(K)}\|z^h - I^h z\|_{L^2(K)} + \|\nabla\cdot\boldsymbol{u}^h\|_{L^2(K)}\|s - I^h s\|_{L^2(K)} \\ &+ \|[\![2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} - p\boldsymbol{n}]\!]\|_{L^2(\partial K\setminus\Gamma)}\|z^h - I^h z\|_{L^2(\partial K\setminus\Gamma)} \\ &+ \|\boldsymbol{g} - 2\mu\nabla^s\boldsymbol{u}^h\cdot\boldsymbol{n} + p^h\boldsymbol{n}\|_{L^2(\partial K\cap\Gamma_N)}\|z^h - I^h z\|_{L^2(\partial K\cap\Gamma_N)}\end{aligned} \tag{4.20}$$

In these indicators, the same local residue terms as seen in the residual-based indicators from Equation (4.14) can be distinguished. They are weighted with a term related either to the error in the dual-velocity $\boldsymbol{z}$ or the error in the dual-pressure s.

4.4 Numerical example: Flow around a sharp corner

In this section, the results regarding a numerical experiment are presented. A benchmark showcase simulating steady incompressible flow around a corner is used [18]. The problem has a known exact solution. This case is used to test the error estimation indicators elaborated in Section 4.2 and 4.3.

4.4.1 Problem formulation

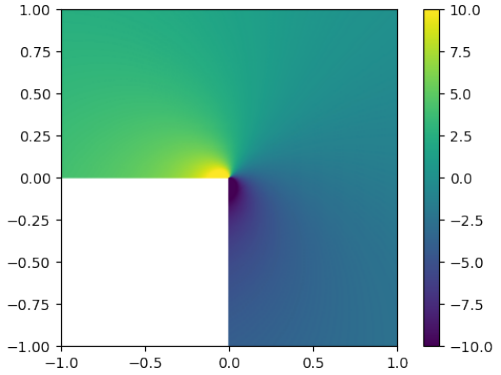
The adaptivity methods adjusted to the Stokes equations are demonstrated on an example regarding flow on the same L-shaped domain as seen in Figure (9a). The exact velocity and pressure are given by:

$$\begin{aligned} u_x &= \quad R^\alpha\big[\sin(\theta)\tfrac{\partial\psi}{\partial\theta} - (1+\alpha)\cos(\theta)\psi\big] \\ u_y &= -\quad R^\alpha\big[\cos(\theta)\tfrac{\partial\psi}{\partial\theta} + (1+\alpha)\sin(\theta)\psi\big] \end{aligned} \qquad p = -\frac{R^{\alpha-1}}{1-\alpha}\Big[(1+\alpha)^2\frac{\partial\psi}{\partial\theta} + \frac{\partial^3\psi}{\partial\theta^3}\Big] \tag{4.21}$$

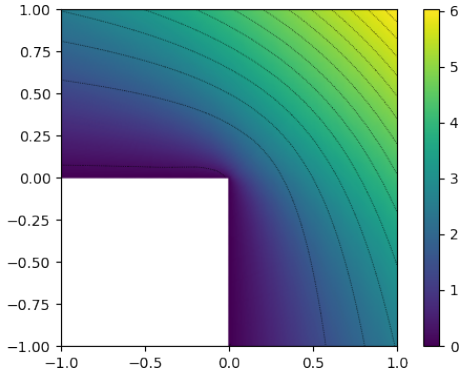
with constants $\alpha = 856399/1572864$ and $\omega = \frac{3}{2}\pi$ and with $\psi(\theta)$ defined as:

$$\begin{aligned}\psi(\theta) =& \frac{\cos(\alpha\omega)}{1+\alpha}\sin((1+\alpha)\theta) - \frac{\cos(\alpha\omega)}{1-\alpha}\sin((1-\alpha)\theta) \\ &+ \cos((1-\alpha)\theta) - \cos((1+\alpha)\theta)\end{aligned} \tag{4.22}$$

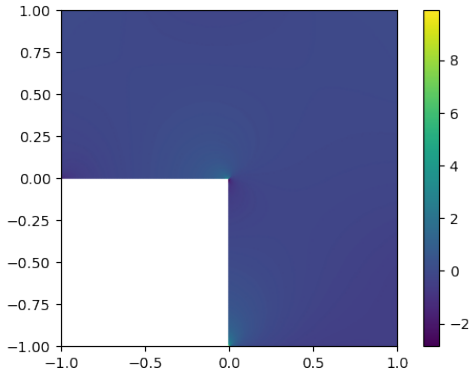
The pressure and velocity are illustrated in Figure 13. The problem is solved according to the weak problem formulation for Stokes, given in Equation (4.2) with the viscosity $\mu = 1$, no body force $\boldsymbol{f} = \boldsymbol{0}$, a no slip condition on Γ_D, such that $\boldsymbol{u}_D = \boldsymbol{0}$, and the Neumann data $\boldsymbol{g}$ on Γ_N defined with the known exact solution. As a goal quantity, the outflow on the bottom boundary $\Gamma_{\mathcal{G}} = \Gamma_N|_{y=-1}$ is taken, such that $\mathcal{G}(\boldsymbol{u}) = \int_{\Gamma_{\mathcal{G}}} \boldsymbol{n}\cdot\boldsymbol{u}\,d\Gamma$. The resulting dual solutions s and $\boldsymbol{z}$ are shown in Figure 13c and 13d respectively. The dual problem can be compared to a flow problem with a fixed traction pre-described to the lower boundary $\Gamma_{\mathcal{G}}$. On the rest of the Neumann boundaries $\boldsymbol{g} = \boldsymbol{0}$ is imposed. These boundary conditions enforce a flow through the lower boundary. The dual-pressure also has a clear singular point around the corner.



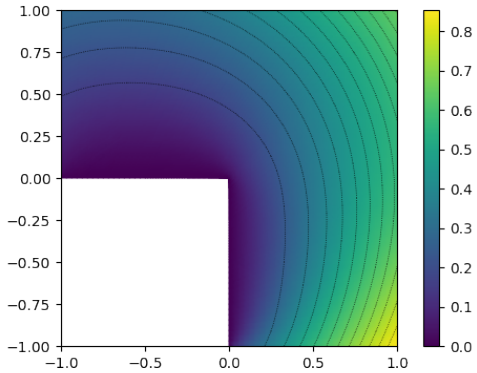
(a) Pressure p with $p_{min} = -10$ and $p_{max} = 10$



(b) Velocity $\boldsymbol{u}$ magnitude and streamlines



(c) Dual-pressure s



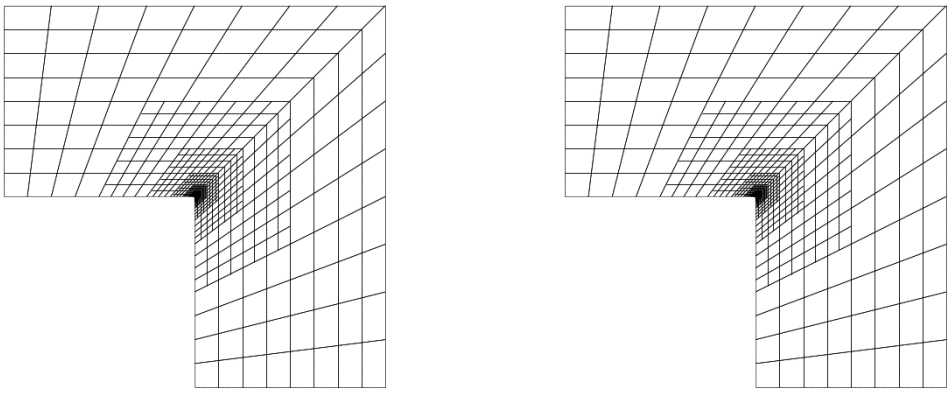
(d) Dual-velocity $\boldsymbol{z}$ magnitude and streamlines

Figure 13: The solutions to the Stokes problem on the L-shaped domain. Because of the singularity, the pressure goes to infinity in the corner. The pressure is plotted between -10 and 10. The dual solutions are created with the outflow in the bottom boundary as a quantity of interest. The dual pressure s has a singularity at the corner as well.

4.4.2 Results

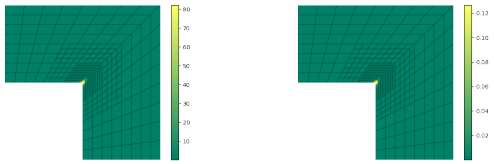
In Figures 14a and 14b, the mesh for both methods resulting from a sequence of 6 refinements with a refinement fraction of $\lambda = 0.5$ is shown. The singularity has a very dominant effect on the error in the solutions. For that reason, the residual-based method as well as the goal-oriented method merely refine the elements closest to the singularity. In Figure 15, the residual-based and the goal-oriented indicators evaluated after the second refinement are shown. From these Figures it can be seen that the term ϵ_{moment} contains very high values on the singularity and dictates the refinement around this point. Because this term is present in the residual-based indicators and the goal-oriented indicators, the resulting meshes are identical.

The convergence of the L^2-norm error in the pressure and velocity and the error in the goal quantity are shown in Figure 16. Note that the convergence behavior is the same for both refinement methods, because with every iteration the same elements are refined. From these figures it can be seen that the convergence rate of the L^2-norm error in the pressure and the velocity and the error in the quantity of interest are much higher for the adaptive refinement techniques, compared to the uniform refinement.



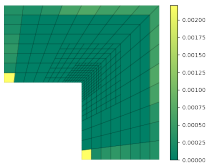
(a) Residual-based mesh after 6 refinements **(b)** Goal-oriented mesh after 6 refinements

Figure 14: Both meshes are created with 6 refinements and a refinement fraction of $\lambda = 0.5$. Because of the dominance of the singularity, both methods have exactly the same refinement pattern.

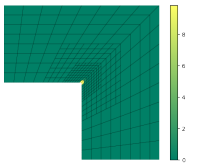


(a) $\|\epsilon_{moment}\|_{L^2(K)}$

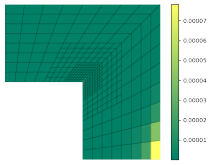
(b) $\|\epsilon_{incomp}\|_{L^2(K)}$



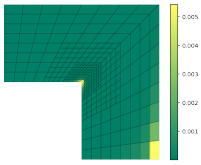
(c) $\|\epsilon_{bound}\|_{L^2(\partial K \cap \Gamma_N)}$



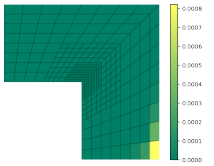
(d) $\|\epsilon_{jump}\|_{L^2(\partial K \setminus \Gamma)}$



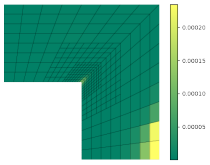
(e) $\|\boldsymbol{z}^h - I^h \boldsymbol{z}\|_{L^2(K)}$



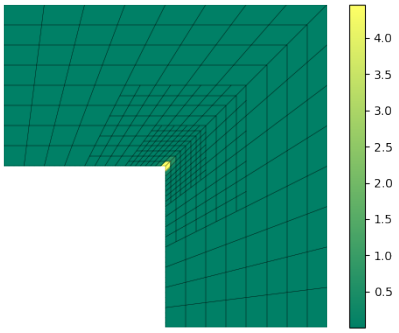
(f) $\|s^h - I^h s\|_{L^2(K)}$



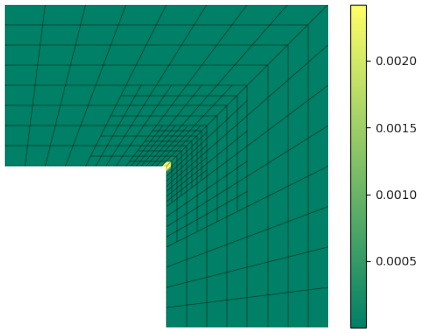
(g) $\|\boldsymbol{z}^h - I^h \boldsymbol{z}\|_{L^2(\partial K \cap \Gamma_N)}$



(h) $\|\boldsymbol{z}^h - I^h \boldsymbol{z}\|_{L^2(\partial K \setminus \Gamma)}$

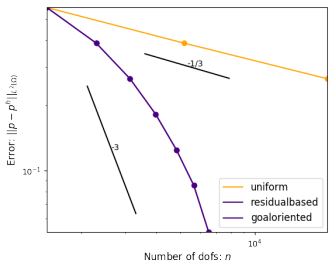


(i) Residual-based indicators

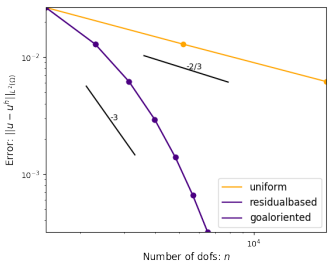


(j) Goal-oriented indicators

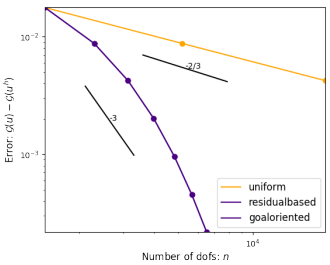
Figure 15: The indicators evaluated after the second refinement iteration are shown. The terms **(a)**, **(b)**, **(c)** and **(d)** are used to construct the residual-based indicators as well as the goal-oriented indicators. Note that they are exactly the same at this point, only because the mesh is identical for both cases. These terms are multiplied with h_K or $h_K^{\frac{1}{2}}$ for the residual-based indicators, while they are multiplied element-wise with **(e)**, **(f)**, **(g)** and **(h)** respectively to compute the goal-oriented indicators. It can be seen that the residual-based indicators **(g)** purely focus on the singularity point on (0,0), while the goal-oriented indicators **(h)** also focus on the point in (1,-1).



(a) Error $\|p - p^h\|_{L^2(\Omega)}$



(b) Error $\|\boldsymbol{u} - \boldsymbol{u}^h\|_{L^2(\Omega)}$



(c) Error $\mathcal{G}(\boldsymbol{u}) - G(\boldsymbol{u}^h)$

Figure 16: The convergence behavior of the different refinement methods is shown. The mesh resulting from both methods is exactly the same after each refinement step, and so is the convergence of the error for both methods. The adaptivity methods significantly improve the convergence behavior of the errors compared to uniform refinement. The error in the velocity as well as in the pressure has an improved convergence compared to the uniform refinement. Quadratic B-splines are used to approximate the pressure and cubic B-splines for the velocity with Taylor-Hood elements.

5 Adaptivity in ImmersoGeometric analysis

In this section, the concepts of Section 3 and 4 are applied to a finite cell method [21]. The central idea of the finite cell method is to extend the physical domain with a simple embedding domain, on which a regular mesh is built. The geometry of the physical domain is recovered by a recursive bi-sectioning algorithm applied on the regular mesh. This method proves to be advantageous for geometrically complex physical domains, because no mesh conforming representation of the geometry is needed. The combination of the Finite Cell Method and IsoGeometric Analysis is referred to as ImmersoGeometric analysis [8,9].

In this section the details of the bi-sectioning algorithm are discussed. Thereafter implementation aspects for the ImmersoGeometric Analysis are elaborated and finally results on the Laplace and Stokes problem using the immersed framework are presented.

5.1 The immersed setting

Let $\Omega \in \mathbb{R}^d$ be a physical domain, extended by its ambient domain Ω_A. On this ambient domain, an ambient mesh $\mathcal{Q}_A$ with elements K is defined and a uniform B-spline basis $\mathcal{B}$ is constructed. For convenience, the ambient domain is chosen to be rectangular with square elements. From the ambient mesh, elements that do not intersect with the physical domain can be omitted. The B-splines from $\mathcal{B}$ that are fully supported by these omitted elements are not taken into consideration. The resulting mesh is defined as:

$$\mathcal{Q} := \{K \mid K \in \mathcal{Q}_A, K \cap \Omega \neq \emptyset\} \tag{5.1}$$

In Figure 17 an example of the immersed setting is shown.

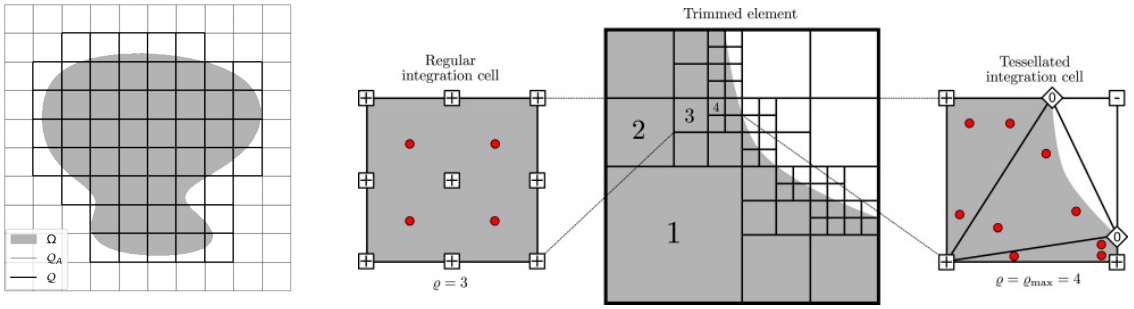
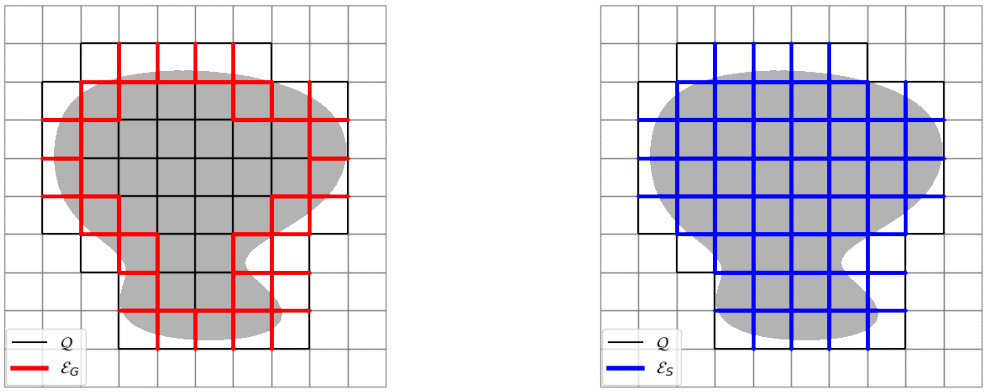


Figure 17: **(a)** Physical domain Ω embedded by Ω_A. **(b)** A schematic representation of the recursive bi-sectioning algorithm. Cut-elements are recursively refined until the finest recursion level ϱ_{max} is reached. In the tessellation procedure, triangular segments are made following $\partial\Omega$. Integration points inside of Ω are added to the integration scheme. (Source: Verhoosel et. al. 2015 [7])

The physical domain is represented by a set of integration points. Because standard quadrature rules would poorly evaluate the geometry on the elements that are intersected by the physical boundary $\partial\Omega$, an advanced numerical integration technique is used, based on bisectioning [7,8]. Let ϱ_{max} be a predefined maximum recursion depth level. The elements that are intersected by $\partial\Omega$ are recursively partitioned into equally-sized smaller elements, until the recursion depth level is reached. A tessellation procedure is applied on the finest level elements, sectioning the element in triangular segments. The set of integration points corresponding to these partitioned elements are used for the numerical integration scheme. The union of all bisected and partitioned elements interior to the element K are referred to as the cut-element K^{cut}. With this trimming algorithm, an integration scheme tailored to the physical boundary is construced. A schematic representation of the recursive

bi-sectioning algorithm is shown in Figure 17b. The boundary $\partial\Omega$ is represented by the faces of the triangular partitions resulting from the tessellation procedure. The set of boundary faces $\mathcal{E}$ is defined as:

$$\mathcal{E} := \{e \subset \partial\Omega \,|\, e = \text{int}(\partial K \cap \partial\Omega), K \in \mathcal{Q}\} \tag{5.2}$$



(a) Example of ghost mesh

(b) Example of skeleton mesh

Figure 18: In Figure **(a)** the ghost mesh are illustrated on the example domain of Figure 17. The skeleton mesh corresponding to this example are shown in **(b)**. It can be seen that the ghost mesh is a subset of the skeleton mesh.

5.2 Problem formulation in the immersed setting

The immersed setting discussed in the previous section is essentially different from the conforming setting that has been used for the problem formulation in Section 3 and 4. Because the basis functions are continuously defined over the physical boundary $\partial\Omega$, strong imposition of Dirichlet boundary conditions is intractable. In [22] a method for imposing Dirichlet boundaries weakly by using Nitsche's method is explained. An additional term is added to the left hand side of the weak problem, weakly enforcing the value of the solution on the Dirichlet boundary Γ_D. For the Laplace and the Stokes problem, the Nitsche boundary terms are defined respectively as:

$$\text{Laplace:} \quad \mathcal{N}(u^h, v^h) := -\int_{\Gamma_N} \left[\nabla u^h \cdot n v^h + \nabla v^h \cdot n(u^h - u_D)\right] d\Gamma + \sum_{e\in\mathcal{E}} \int_e \frac{\beta}{h}(u^h - u_D)v^h \, d\Gamma \tag{5.3}$$

$$\begin{aligned}\text{Stokes:} \quad \mathcal{N}(\boldsymbol{u}^h, \boldsymbol{v}^h, p^h, q^h) := &-\int_{\Gamma_D} \left[(\nabla^S \boldsymbol{u}^h \cdot \boldsymbol{n}) \cdot \boldsymbol{v}^h + (\nabla^S \boldsymbol{v}^h \cdot \boldsymbol{n}) \cdot (\boldsymbol{u}^h - \boldsymbol{u}_D)\right] d\Gamma \\ &+ \mu \sum_{e\in\mathcal{E}} \int_e \frac{\beta}{h}(\boldsymbol{u}^h - \boldsymbol{u}_D) \cdot \boldsymbol{v}^h \, d\Gamma + \int_{\Gamma_D} \left[p^h \boldsymbol{n} \cdot v^h + q^h \boldsymbol{n} \cdot (\boldsymbol{u}^h - \boldsymbol{u}_D)\right] d\Gamma\end{aligned} \tag{5.4}$$

Here β is the Nitsche parameter and h is the length fraction corresponding to the face e. The stability parameter β should be selected appropriately, being large enough to ensure stability, while not being too large to cause ill-conditioning or a reduction in accuracy. An obvious choice for the length fraction would be $h = \sqrt[d]{\text{meas}(K^{cut})}$, where $\text{meas}(K^{cut})$ denotes the length, area or volume of the cut-element. In Hansbo et. al. [23], it has been shown that this length fraction choice is suboptimal in the case of bad cut-elements, e.g. very small cut-elements or sliver cuts. As a result, instabilities may occur, caused by ill-conditioning of the problem [24].

A stable and optimal choice of h is proposed by Burman et. al. [25] and Hansbo et. al. [25] together with the introduction of the ghost-penalty technique. The length fraction is determined independently of the cut-element configuration by $h = \sqrt[d]{\text{meas}(K)}$. To allow the scaling of the Nitsche penalty term independent of the cut-element configuration, an additional ghost-penalty operator is utilized. The ghost-penalty operator controls the high derivative jumps over the interfaces of the elements which are intersected by the domain boundary $\partial\Omega$. In this contribution, B-splines of degree p with C^{p-1}-continuity are considered. The high derivative should be of order p. For the Laplace and the Stokes problem, the ghost-penalty operators are defined respectively as:

$$\text{Laplace:} \qquad s_{gh}(u^h, v^h) = \sum_{e \in \mathcal{E}^G} \int_e \tilde{\gamma} h_K^{2p-1} [\![\partial_n^p u^h]\!][\![\partial_n^p v^h]\!] \, d\Gamma \tag{5.5}$$

$$\text{Stokes:} \qquad s_{gh}(\boldsymbol{u}^h, \boldsymbol{v}^h) = \sum_{e \in \mathcal{E}^G} \int_e \tilde{\gamma} \mu h_K^{2p-1} [\![\partial_n^p \boldsymbol{u}^h]\!][\![\partial_n^p \boldsymbol{v}^h]\!] \, d\Gamma, \tag{5.6}$$

with polynomial degree p and $[\![\partial_n^p \cdot]\!]$ the p^{th} normal derivative jump over the interface $e \in \mathcal{E}^G$. The stability parameter $\tilde{\gamma}$ should be chosen such that the stability is ensured, while not being to dominant, which may cause ill-conditioning or an reduction in accuracy. The set of interfaces $\mathcal{E}^G$ on which this stabilization is applied, referred to as the ghost mesh, is defined as:

$$\mathcal{E}^G = \{\partial K \cap \partial K' \mid K, K' \in \mathcal{Q}, K \cap \partial\Omega \neq \emptyset, K \neq K'\} \tag{5.7}$$

An example of the ghost mesh is shown in Figure 18a.

An additional issue is encountered when considering the Stokes equations in the immersed setting. In the conforming setting, inf-sup stability is achieved by adopting velocity-pressure pairs, such as Taylor-Hood or Raviart-Thomas elements. In the immersed setting, this stability is not ensured due to the existence of the cut-elements, causing oscillations in the pressure field. In [11] the skeleton stabilization method for the Navier-Stokes equations in the ImmersoGeometric context has been proposed. The skeleton-penalty term is originally introduced in [26] in the conforming IGA setting. The purpose of this stabilization method is twofold; the issue regarding the inf-sup stability is solved, allowing the utilization of identical pairs of spaces for the velocity and pressure field. Additionally, the issue related to the bad cut-elements for the pressure field is solved by controlling the high order pressure jumps. The set of interfaces to which the stabilization is applied is referred to as the skeleton mesh, which is essentially formed by the inner interfaces of the mesh:

$$\mathcal{E}^S = \{\partial K \cap \partial K' \mid K, K' \in \mathcal{Q}, K \neq K'\} \tag{5.8}$$

Note that the skeleton mesh is a superset of the ghost mesh, such that $\mathcal{E}^G \subset \mathcal{E}^S$. An example of the skeleton is shown in Figure 18b. The skeleton stabilization term $s_{sk}(p^h, q^h)$ is defined as:

$$s_{sk}(p^h, q^h) = \sum_{e \in \mathcal{E}^S} \int_e \gamma \mu^{-1} h_K^{2p+1} [\![\partial_n^p p^h]\!][\![\partial_n^p q^h]\!] \, d\Gamma \tag{5.9}$$

The stabilization parameter γ is chosen in a similar way as the parameters β and $\tilde{\gamma}$. More information about the appropriate selection of the stability parameters $\tilde{\gamma}$ and γ can be found in [11]. The stabilized discretized weak formulations for the immersed setting are given by:

$$\text{Laplace:} \qquad \mathcal{B}(u^h, v^h) + \mathcal{N}(u^h, v^h) + s_{gh}(u^h, v^h) = \mathcal{F}(v^h) \tag{5.10}$$

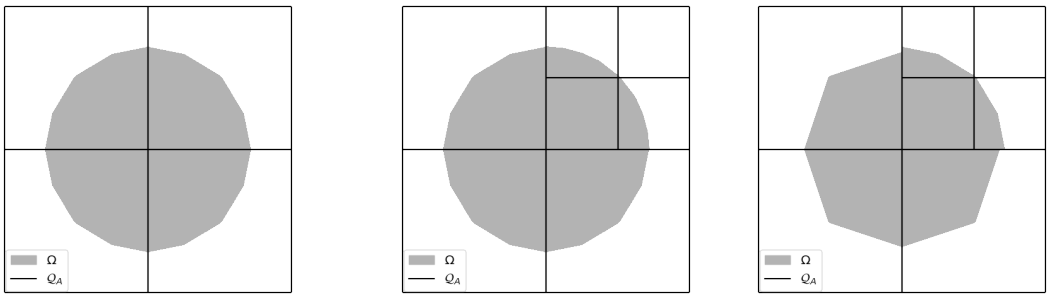
$$\text{Stokes:} \qquad \mathcal{L}(\boldsymbol{u}^h, \boldsymbol{v}^h)(p^h, q^h) + \mathcal{N}(\boldsymbol{u}^h, \boldsymbol{v}^h, p^h, q^h) + s_{gh}(\boldsymbol{u}^h, \boldsymbol{v}^h) = \mathcal{F}(\boldsymbol{v}^h) + s_{sk}(p^h, q^h) \tag{5.11}$$

5.3 Adaptive refinement in the immersed setting

In this section, implementation aspects needed for the adaptive refinement framework in the immersed setting are discussed.

5.3.1 Applying local refinement

Within the proposed immersed setting, refinements can be executed on the mesh $\mathcal{Q}$ before or after the trimming procedure. In both cases, the truncated hierarchical basis $\mathcal{T}$ can be constructed on the refined mesh. Additionally, applying refinements before the trimming procedure allows for locally improving the geometry reconstruction by the recursive bi-sectioning algorithm. Refining elements that are intersected by the boundary $\partial\Omega$ can be compared to locally increasing the maximum recursion depth ϱ_{max}. In Figure 19 the refinement of the ambient mesh before the trimming procedure is illustrated. Problems are encountered to match the tessellation procedure between elements of different levels. This is illustrated in Figure 19c. Note that this approach is computationally expensive, because the trimming procedure is to be repeated after every refinement iteration.



(a) **(b)** **(c)**

Figure 19: In **(a)**, the geometry representation of a circle with a 2x2 ambient mesh with $\varrho_{max} = 2$. In **(b)** it is shown that the geometry representation of the circle is improved due to the refinement of the ambient mesh. When taking a lower recursive refinement depth $\varrho_{max} = 1$, a problem with the matching of the tessellation procedure between different level elements is be seen in **(c)**.

In this work, refinement of the mesh Q after the trimming procedure is considered. The maximum refinement level is bounded by the maximum recursion depth $N \leq \varrho_{max}$, because the integration points collected in the recursive bi-sectioning algorithm are defined up to that level. To find suitable integration points for finer levels, the total trimming procedure needs to be repeated with a higher ϱ_{max}. Note that refinement of elements intersected by the boundary $\partial\Omega$ results in an enrichment of the truncated hierarchical basis, while the integration scheme remains intact.

5.3.2 Error estimation methods

Due to the Nitsche boundary imposition and the added stabilizing terms, the weak formulation for the Laplace and the Stokes problem in the immersed setting, given in Equation (5.10) and (5.11), differ from the formulations introduced in Section 3 and 4 with Equation (3.2) and (4.2). In this work, the assumption is made that the effect of these terms on the residual $\mathcal{R}^h(v)$ and the dual-residual $\mathcal{R}^h(z)$ is negligible. Therefore the indicators derived in Section 3 and 4 can be used as local error estimations for the adaptive immersed setting. Note that the B-spline basis defined on $\mathcal{Q}$ is C^{p-1}-continuous, because the total regular mesh can be captured with a single patch. When utilizing quadratic or higher order B-splines, the jump term can be omitted from the indicators.

To obtain a well-conditioned dual problem in the immersed setting, the Nitsche term and the ghost- and skeleton-penalty terms discussed in Section 5.2 are applied to the discrete dual form as well, which is formalized as:

$$\text{Laplace:} \qquad \mathcal{B}(v^h, z^h) + \mathcal{N}(v^h, z^h) + s_{gh}(v^h, z^h) = \mathcal{G}(v^h, q^h) \quad (5.12)$$

$$\text{Stokes:} \qquad \mathcal{L}(\boldsymbol{v}^h, \boldsymbol{z}^h)(q^h, s^h) + \mathcal{N}(\boldsymbol{v}^h, \boldsymbol{z}^h, q^h, s^h) + s_{gh}^{p+1}(\boldsymbol{v}^h, \boldsymbol{z}^h) = \mathcal{G}(\boldsymbol{v}^h, q^h) + s_{sk}^{p+1}(q^h, s^h) \quad (5.13)$$

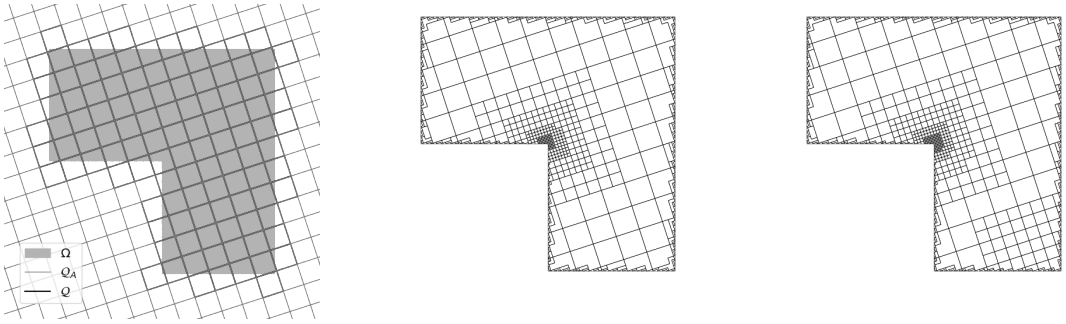
It is explicitly noted that the ghost-penalizing term $s_{gh}^{p+1}(\boldsymbol{v}^h, \boldsymbol{z}^h)$ and the skeleton-penalizing term $s_{sk}^{p+1}(q^h, s^h)$ should be evaluated using the polynomial degree of the dual problem $p+1$. The same ghost and skeleton can be reused for the evaluation of these terms in the dual problem[2]. The selection of suitable stabilization parameters $\tilde{\gamma}$ and γ depends on the polynomial order. It is advised to perform an additional selection procedure for these stability parameters for the dual problem.

5.4 Numerical experiments

Numerical experiments are conducted to evaluate the error estimation and adaptivity procedure within the ImmersoGeometric setting. The experiment considering the Laplace problem on an L-shaped domain, as elaborated in Chapter 3.4, is redefined on an immersed mesh. Similar results are obtained for this setting. Secondly an example of incompressible steady flow through a complex shaped channel is presented.

5.4.1 Numerical example: Laplace on an immersed L-shaped domain

For this numerical example, the physical domain Ω as described in Figure 9a is used. This domain is embedded by a squared ambient domain, which is rotated with an angle of 18 degrees to allowing an irregular cut-element pattern. The mesh $\mathcal{Q}$ resulting from omitting elements outside Ω is shown in Figure 20a. For the trimming procedure, a maximum recursion depth of $\varrho_{max} = 5$ is taken. The Laplace equations are solved by using the weak formulation for the immersed setting as derived in Equation (5.10). To select appropriate values for β and $\tilde{\gamma}$, the problem is solved with different values and the solution is compared to the exact solution. $\beta = 10$ and $\tilde{\gamma} = 10^{-3}$ are found to be suitable values.



(a) Immersed setting of the L-shaped domain

(b) Trimmed mesh after 4 residual-based refinements

(c) Trimmed mesh after 4 goal-oriented refinements

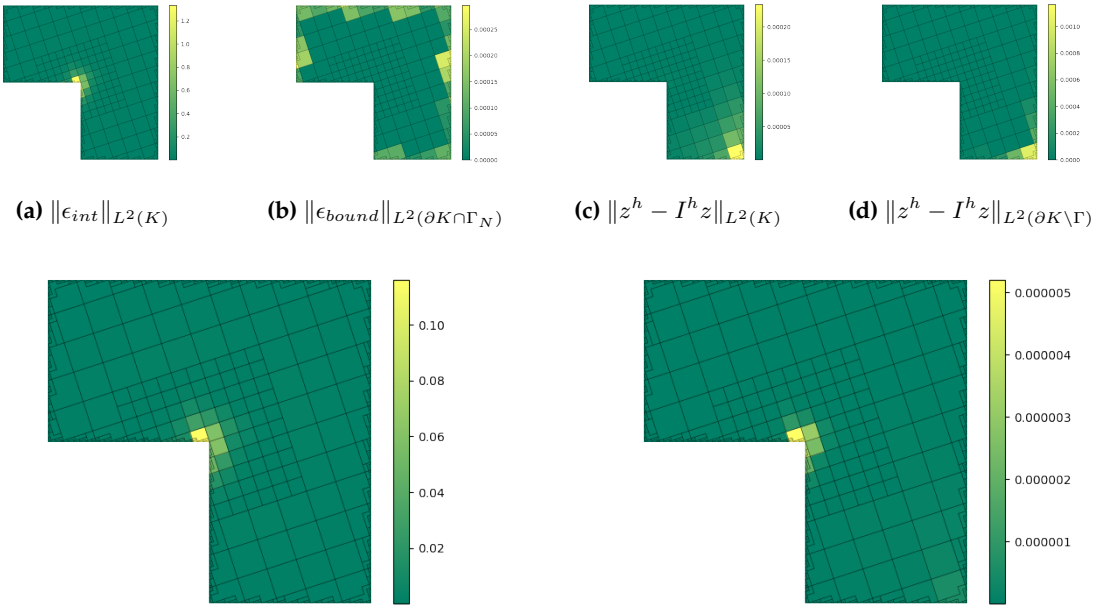
Figure 20: In Figure **(a)** the immersed setting for the L-shaped domain is shown. Refinements are applied to the trimmed mesh based on the residual-based and goal-oriented error estimation. The resulting meshes for both methods after 4 refinement iterations are shown in **(b)** and **(c)** respectively.

The indicators as derived in Equation (3.15) are used for the residual-based adaptivity. Because quadratic B-splines are used and the mesh consists of only one patch, the jump term can be omitted. In Figure 21 the indicators after one refinement are shown.

As a quantity of interest the value of u in the lower right corner is taken, such that $\mathcal{G}(u) = u(1, -1)$. The dual solution z is found by solving the dual problem as defined in Equation (5.10). Because the exact dual solution is unknown, the selection of $\tilde{\gamma}^{p+1}$ is based on minimizing the dual-residual $\mathcal{R}^h(z)$. A suitable value of $\tilde{\gamma}^{p+1} = 10^{-4}$ is found. The indicators as derived in Equation (3.24) are

[2]When considering the uniformly refined dual problem $\mathcal{V}^{h/2,p}$ instead of the order elevated dual problem $\mathcal{V}^{h,p+1}$, uniform refinement of the mesh $\mathcal{Q}$ after the trimming procedure is sufficient. The ghost $\mathcal{E}^G$ and the skeleton $\mathcal{E}^S$ need to be refined accordingly.

used for the goal-oriented adaptivity. Examples of these indicators are shown in Figure 21f.

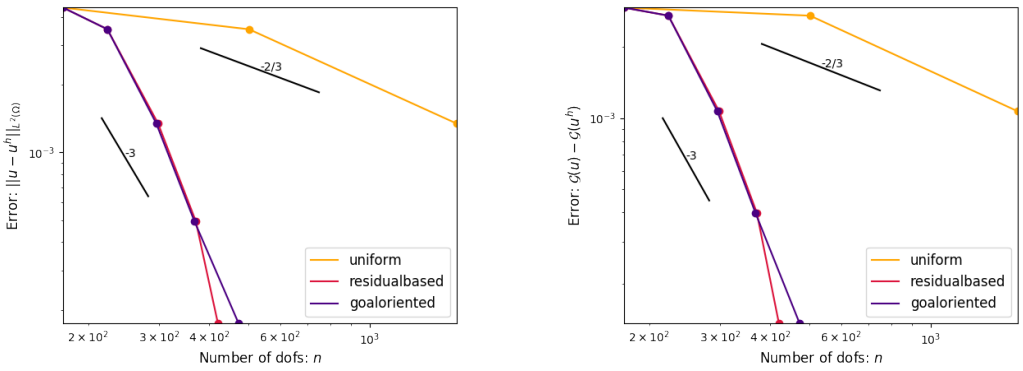


(a) $\|\epsilon_{int}\|_{L^2(K)}$ **(b)** $\|\epsilon_{bound}\|_{L^2(\partial K \cap \Gamma_N)}$ **(c)** $\|z^h - I^h z\|_{L^2(K)}$ **(d)** $\|z^h - I^h z\|_{L^2(\partial K \setminus \Gamma)}$

(e) Residual-based indicators **(f)** Goal-oriented indicators

Figure 21: The indicators evaluated after the first refinement iteration are shown. The terms from **(a)** and **(b)** are multiplied with h_K or $h_K^{\frac{1}{2}}$ respectively to construct the residual-based indicators. The terms in **(a)** and **(b)** are multiplied respectively with the terms in **(c)** and **(d)** to compute the goal-oriented indicators.

The resulting meshes for residual-based and goal-oriented adaptivity after four refinements are shown in Figure 20b and 20c. It can be seen that similar mesh structures are obtained for the immersed setting of the Laplace on the L-shaped domain as were obtained in the conforming setting. A clear focus on the singular point is visible for both error estimation methods. To study the effectiveness of the adaptive refinement, the convergence of the L^2-error and the error in the quantity of interest are shown in Figure 22. The convergence rates obtained for residual-based and goal-oriented adaptivity are very similar, which was also obtained for the conforming setting (Figure 11). The convergence rate obtained with uniform refinement is much lower.



(a) L^2-norm error $\|u - u^h\|_{L^2(\Omega)}$ **(b)** Error in the quantity of interest $\mathcal{G}(u) - \mathcal{G}(u^h)$

Figure 22: Convergence of the L^2-norm of the exact error in u and the error in the quantity of interest. For the discretization of u^h quadratic B-splines are used.

5.4.2 Numerical example: Stokes flow through a complex channel

For this numerical example, Stokes flow through a complex shaped channel is considered. The domain used for this example is shown in Figure 23. The Stokes equations as defined in Equation (4.1) are used to describe the flow through the channel. For this example the viscosity is taken to be $\mu = 1$ and there is no body force $\boldsymbol{f} = \boldsymbol{0}$. The trimmed cut-boundaries of the domain are defined as no slip Dirichlet boundaries, such that $\boldsymbol{u} = \boldsymbol{0}$ on Γ_D and the boundaries that intersect the ambient domain are Neumann boundaries. On the left Neumann boundary $\Gamma_N|_{x=0}$ a fixed traction $\boldsymbol{g} = -\boldsymbol{n}$ as inflow boundary is imposed and on the other outflow Neumann boundaries $\boldsymbol{g} = \boldsymbol{0}$ is imposed. The Stokes equations for this problem are reformulated as:

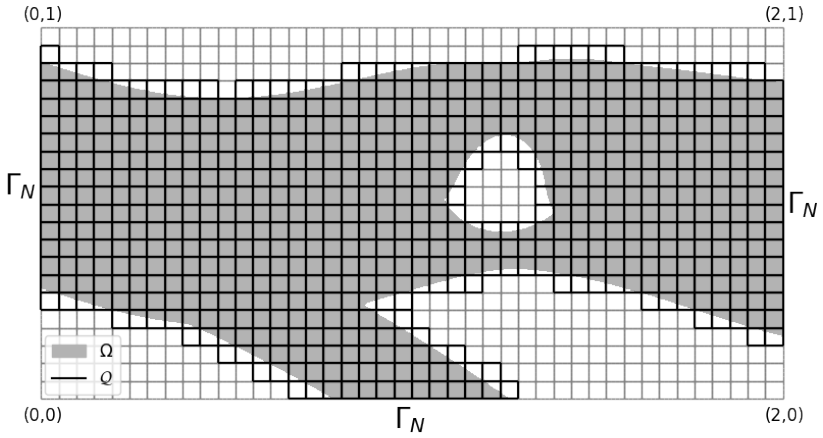
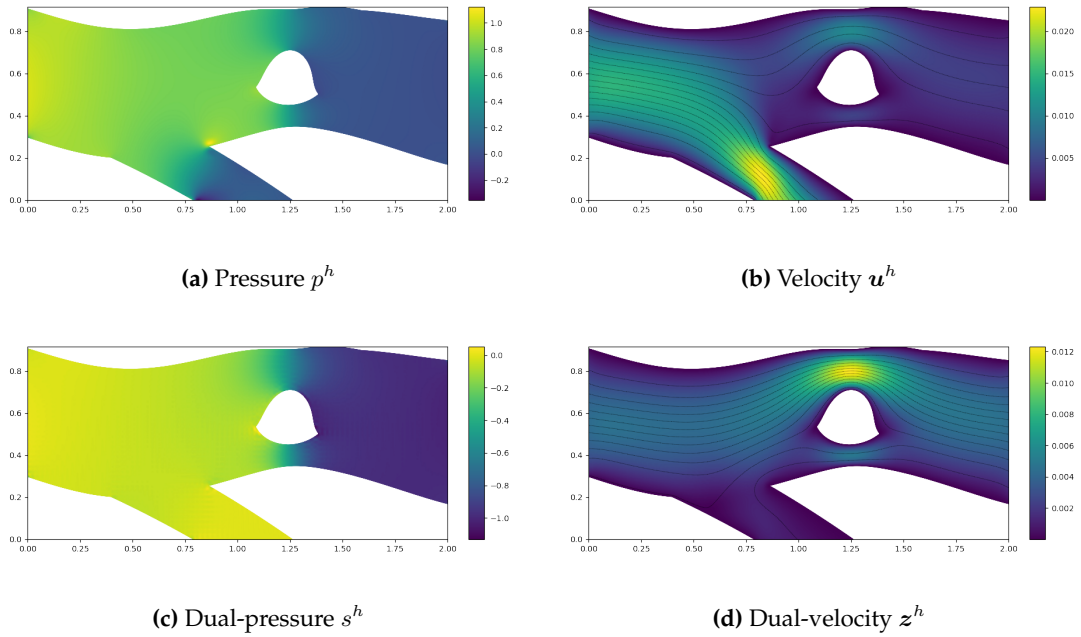


Figure 23: Complex channel domain Ω and the mesh $\mathcal{Q}$, constructed with 42x21 elements. The three boundaries that intersect the ambient boundary are the Neumann boundaries. The rest of the boundaries, which consist of all cut-element edges, are no-slip Dirichlet boundaries.

$$\begin{aligned} -\nabla \cdot (2\nabla^s \boldsymbol{u}) + \nabla p &= \boldsymbol{0} && \text{in } \Omega \\ \nabla \cdot \boldsymbol{u} &= 0 && \text{in } \Omega \\ 2\nabla^s \boldsymbol{u} \cdot \boldsymbol{n} - p\boldsymbol{n} &= -\boldsymbol{n} && \text{on } \Gamma_N|_{x=0} \\ \boldsymbol{u} &= 0 && \text{on } \Gamma_D \end{aligned} \tag{5.14}$$

The problem is solved by using the weak formulation for the immersed setting as derived in Equation (5.11). The selection of the parameters is based on the residual $\mathcal{R}^h(v)$ and the shape of the solutions. It has been found that $\beta = 100$, $\tilde{\gamma} = 10^{-4}$ and $\gamma = 10^{-2}$ are suitable values. The approximated pressure p^h and velocity $\boldsymbol{u}^h$ fields are shown in Figure 24.

The quantity of interest is defined as the flow through the right boundary is taken, such that



(a) Pressure p^h

(b) Velocity $\boldsymbol{u}^h$

(c) Dual-pressure s^h

(d) Dual-velocity $\boldsymbol{z}^h$

Figure 24: In **(a)** and **(b)** the pressure and velocity field are shown. In **(b)** the lines indicate the streamlines of the flow. In **(c)** and **(d)** the dual-pressure s^h and the dual-velocity $\boldsymbol{z}^h$ corresponding to the goal-quantity $\mathcal{G}(u) = \int_{\Gamma_\mathcal{G}} \boldsymbol{u} \cdot \boldsymbol{n} \, d\Gamma$ are shown.

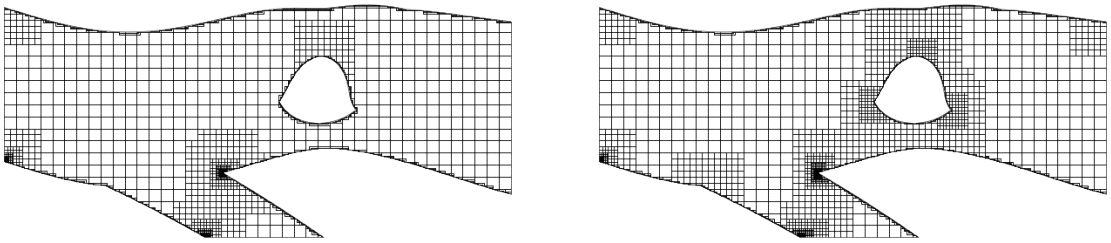
$\mathcal{G}(u) = \int_{\Gamma_\mathcal{G}} \boldsymbol{u} \cdot \boldsymbol{n} \, d\Gamma$ with $\Gamma_\mathcal{G} = \Gamma_N|_{x=2}$. The corresponding dual-pressure s^h and dual-velocity $\boldsymbol{z}^h$ are shown in Figures 24c and 24d respectively. These solutions are found by solving the dual problem with the weak formulation given in Equation (5.13), with tailored stabilization parameters $\tilde{\gamma} = 1$ and $\gamma = 10^{-5}$.

The residual-based and goal-oriented indicators are calculated with Equation (4.14) and (4.20) respectively. Refinements are carried out on the mesh shown in Figure 23 with a refinement fraction of $\lambda = 0.5$ and a maximum refinement level of $N = \varrho_{max} = 5$. The resulting meshes for both methods are shown in Figure 25.

From the refinement structure it can be seen that in both meshes, refinements are applied in the areas where the Neumann boundaries intersect with the Dirichlet boundaries. In these areas, singularities are induced because of the mismatch between the Neumann boundary and the Dirichlet boundary. This mismatch is caused by the no-slip boundary enforcing the inflow velocity in the tangential direction to be zero, which is in contrast with the imposition by the Neumann boundary. Refinements are also explicitly carried out on the sharp edge of the lower channel, which are caused by geometrical singularities comparable to the sharp corner in the example demonstrated in Section 4.4. In Figure 24a this singularity can be recognized in the pressure field.

A clear distinction between both refinement methods is observed around the obstacle. In the goal-oriented mesh, more refinements are applied around the obstacle than in residual-based mesh. In the primal problem, the largest portion of the flow is located between the inflow boundary $\Gamma_N|_{x=0}$ and the bottom outflow boundary $\Gamma_N|_{y=0}$. In the dual problem, the flow is enforced through the

righter outflow boundary $\Gamma_N|_{x=2}$, as seen in Figure 24d. The obstacle has much more impact on the flow pattern of the dual-velocity and therefore, a detailed mesh around the obstacle is required.



(a) Residual-based mesh **(b)** Goal-oriented mesh

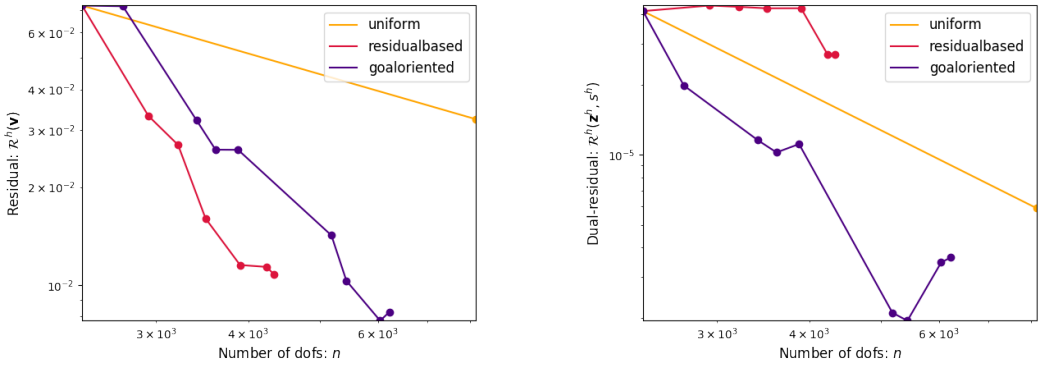
Figure 25: The residual-based mesh **(a)** and the goal-oriented mesh **(b)** after five refinement iterations are shown. A refinement fraction of $\lambda = 0.5$ is used.

Because the exact solutions are unknown for this example, the convergence study is based on the residual and dual-residual, which are respectively calculated as:

$$\mathcal{R}^h(v) = \left| \int_\Omega (-\nabla \cdot (2\nabla^s \boldsymbol{u}^h) + \nabla p^h)\, d\Omega \right| \tag{5.15a}$$

$$\mathcal{R}^h(\boldsymbol{z}^h, s^h) = \left| \int_{\Gamma_N|_{x=0}} -\boldsymbol{n} \cdot \boldsymbol{z}^h \, d\Gamma - \int_\Omega (\nabla^s \boldsymbol{u}^h : \nabla^s \boldsymbol{z}^h - p^h \nabla \cdot \boldsymbol{z}^h - s^h \nabla \cdot \boldsymbol{u}^h)\, d\Omega \right| \tag{5.15b}$$

The convergence rates of these residuals evaluated during the refinement steps of residual-based refinement, goal-oriented refinement and uniform refinement are shown in Figure 26. From this figure it can be seen that both adaptive refinement methods accomplish a higher convergence rate of the residual $\mathcal{R}^h(v)$ than the uniform refinement. This is mainly achieved due to the fact that both methods construct a mesh with improved resolution around singularities. Additionally, it can be seen that the goal-oriented refinement method successfully increases the convergence rate with respect to the dual-residual $\mathcal{R}^h(\boldsymbol{z}^h, s^h)$, compared to residual-based refinement and goal-oriented refinement.



(a) Residual $\mathcal{R}^h(v)$ **(b)** Dual-residual $\mathcal{R}^h(\boldsymbol{z}^h, s^h)$

Figure 26: The convergence rate for residual evaluated with Equation (5.15a) and the dual-residual evalued with Equation (5.15b) are shown.

6 Conclusions

The main goal of this work is to enhance the ImmersoGeometric analysis framework with error estimation and adaptivity techniques. To achieve this goal, we have first studied spline technologies with local refinement capabilities. Truncated hierarchical B-splines are considered, and an analysis is conducted to compare the performance of the truncated hierarchical B-splines with standard hierarchical B-splines. From this analysis it is concluded that the matrices corresponding to the truncated hierarchical B-splines are generally sparser than those corresponding to the standard hierarchical B-splines. It is also demonstrated that generally system matrix condition numbers are smaller when using a truncated hierarchical basis. Since there are no substantial disadvantages to using truncated hierarchical B-splines, also not in terms of implementation, it is concluded that truncated hierarchical bases are the preferred local refinement technology to be considered in conjunction with IsoGeometric and ImmersoGeometric analysis. In the context of this thesis, a computationally efficient algorithm for the construction of THB-splines has been developed within the open source finite element library Nutils.

As an intermediate step toward error estimation and adaptivity for ImmersoGeometric analysis, we have first considered adaptive methods using THB-splines in the setting of (mesh conforming) IsoGeometric analysis. We have considered both residual-based and goal-oriented error estimation methods in the context of steady heat conduction and steady incompressible viscous flow problems. Error estimates and indicators have been derived for both adaptive methods. Numerical experiments have been performed on benchmark cases, from which it is concluded that both refinement methods effectively improve the error convergence rate in comparison to uniform refinements. Making a general distinction between the performance of both methods is intricate, since their performance is highly dependent on the problem under consideration. In the case of singular behavior of the primal problem, both adaptivity methods are observed to behave similarly, which we attribute to the fact that the singularities have a significant influence on the considered goal quantities. A significant difference between both adaptive methods can be encountered in the case that there are no singularities in the primal problem, or in cases where the goal quantity is selected such that it is hardly influenced by primal problem singularities. Such test cases have not been presented in this thesis and are recommended to be studied in more detail.

For the extension of the error estimation and adaptivity framework to the immersed setting, we have considered stabilized ImmersoGeometric analysis formulations. In the case of steady heat conduction problems we apply ghost-penalty stabilization in order to control the solution near immersed boundaries and in order to avoid ill conditioning related to the Nitsche boundary imposition. In the case of Stokes flow we have moreover considered skeleton-penalty stabilization to avoid the occurrence of pressure oscillations near boundaries and to stabilize equal order discretizations of the mixed formulation. The THB-spline based adaptivity framework considered in the conforming setting has been applied in this ImmersoGeometric analysis setting, where the most prominent modification pertains to the required stabilization of the dual problem. From the conducted numerical experiments we conclude that upon appropriate selection of the stabilization parameters, similar refinement patterns as for the conforming case are obtained for the L-shaped Laplace problem. The Stokes analysis of a complex shaped channel demonstrates that meaningful refinement patterns can be obtained for ImmersoGeometric viscous flow problems.

In this work only refinement of the ImmersoGeometric discretization space, and not of the geometry, is considered. This is mainly motivated by the currently available functionality in the employed finite element library. To optimally exploit the benefits of the adaptive ImmersoGeometric analysis framework, it is recommended, however, to consider local mesh refinement in combination with geometry refinement. Besides the evident need to extend the current implementation, it should be studied whether the presented error indicators remain suitable in combination with geometry refinement.

References

[1] A. Ern and J.-L. Guermond. *Theory and Practice of Finite Elements*. Applied Mathematical Sciences 159, Springer-Verlag, New York, 2004.

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

[3] J.T. Oden and S. Prudhomme. Goal-oriented error estimation and adaptivity for the finite element method. *Computers and Mathematics with Applications*, 41(5):735 – 756, 2001.

[4] 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):4135 – 4195, 2005.

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

[6] A.-V. Vuong, C. Giannelli, B. Jüttler, and B. Simeon. A hierarchical approach to adaptive local refinement in isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 200(49):3554 – 3567, 2011.

[7] C.V. Verhoosel, G.J. van Zwieten, B. van Rietbergen, and R. de Borst. Image-based goal-oriented adaptive isogeometric analysis with application to the micro-mechanical modeling of trabecular bone. *Computer Methods in Applied Mechanics and Engineering*, 284:138 – 164, 2015. Isogeometric Analysis Special Issue.

[8] E. Rank, M. Ruess, S. Kollmannsberger, D. Schillinger, and A. Düster. Geometric modeling, isogeometric analysis and the finite cell method. *Computer Methods in Applied Mechanics and Engineering*, 249-252:104 – 115, 2012. Higher Order Finite Element and Isogeometric Methods.

[9] D. Kamensky, M.-C. Hsu, D. Schillinger, J.A. Evans, A. Aggarwal, Y. Bazilevs, M.S. Sacks, and T.J.R. Hughes. An immersogeometric variational framework for fluid—structure interaction: Application to bioprosthetic heart valves. *Computer Methods in Applied Mechanics and Engineering*, 284:1005–1053, 2015.

[10] T. Hoang, C.V. Verhoosel, F. Auricchio, E.H. van Brummelen, and A. Reali. Skeleton-stabilized isogeometric analysis: High-regularity interior-penalty methods for incompressible viscous flow problems. *Computer Methods in Applied Mechanics and Engineering*, 337:324 – 351, 2018.

[11] T. Hoang, C.V. Verhoosel, F. Auricchio, E. Harald van Brummelen, and A. Reali. Skeleton-stabilized immersogeometric analysis for incompressible viscous flow problems. *Computer Methods in Applied Mechanics and Engineering*, 344:421 – 450, 2019.

[12] D. D'Angella, S. Kollmannsberger, E. Rank, and A. Reali. Multi-level bézier extraction for hierarchical local refinement of isogeometric analysis. *Computer Methods in Applied Mechanics and Engineering*, 328:147 – 174, 2018.

[13] E.M. Garau and R. Vázquez. Algorithms for the implementation of adaptive isogeometric methods using hierarchical b-splines. *Applied Numerical Mathematics*, 123:58 – 87, 2018.

[14] G.J. van Zwieten, van J. Zwieten, C.V. Verhoosel, E. Fonn, and W. Hoitinga. nutils/nutils v5.0a0.

[15] K. Johannessen, F. Remonato, and T. Kvamsdal. On the similarities and differences between classical hierarchical, truncated hierarchical and lr b-splines. *Computer Methods in Applied Mechanics and Engineering*, 291, 03 2015.

[16] P.h. Clément. Approximation by finite element functions using local regularization. *ESAIM: Mathematical Modelling and Numerical Analysis - Modélisation Mathématique et Analyse Numérique*, 9(R2):77–84, 1975.

[17] L. Dedè and H.A.F.A. Santos. B-spline goal-oriented error estimators for geometrically nonlinear rods. *Computational Mechanics*, 49(1):35–52, Jan 2012.

[18] R. Verfürth. *A Review of a Posteriori Error Estimation Adaptive Mesh-Refinement Techniques*, volume 50. 05 1996.

[19] R. Becker. An optimal-control approach to a posteriori error estimation for finite element discretizations of the navier-stokes equations. *East-West Journal of Numerical Mathematics*, 8, 01 2000.

[20] K. v.d. Zee, H. v. Brummelen, and R. Borst. Goal-oriented adaptivity for steady fluid-structure interaction. 03 2019.

[21] J.Parvizian, A. Düster, and E. Rank. Finite cell method. *Computational Mechanics*, 41(1):121–133, 2007.

[22] A. Embar, J. Dolbow, and I. Harari. Imposing Dirichlet boundary conditions with Nitsche's method and spline based finite elements. *International Journal for Numerical Methods in Engineering*, 83(7):877–898, 2010.

[23] P. Hansbo. Nitsche's method for interface problems in computational mechanics. *GAMM-Mitteilungen*, 28:183 – 206, 11 2005.

[24] F. de Prenter, C. Lehrenfeld, and A. Massing. A note on the stability parameter in nitsche's method for unfitted boundary value problems. *Computers and Mathematics with Applications*, 75(12):4322–4336, 6 2018.

[25] E. Burman and P. Hansbo. Fictitious domain finite element methods using cut elements: I. a stabilized lagrange multiplier method. *Computer Methods in Applied Mechanics and Engineering*, 199(41):2680 – 2686, 2010.

[26] T. Hoang, C.V. Verhoosel, F. Auricchio, E. Harald van Brummelen, and A. Reali. Mixed isogeometric finite cell methods for the stokes problem. *Computer Methods in Applied Mechanics and Engineering*, 316:400 – 423, 2017. Special Issue on Isogeometric Analysis: Progress and Challenges.

Appendices

A Truncated Hierarchical B-spline construction in nutils

This appendix contains the code that builds the hierarchical basis or the truncated hierarchical based, based on the given input, as it is implemented in the nutils v5.0a0 framework. This code is found in the function called topology.py on lines 2092-2238. This code is developed in collaboration with the company Evalf as part of the thesis.

The parts which are related to the algorithm implementation aspects explained in Section 2.3 are summarized here. Additionally, some parts that acquire some special attention are mentioned.

- 50-83 - In this part the definition of the basis is done, as is described in Section 2.3.1.
- 85-149 - In this part the construction of the basis is done, as is described in Section 2.3.3.
- 103-116 - Here the B-splines for the hierarchical basis are assembled.
- 118-145 - Here the B-splines for the truncated hierarchical basis are assembled. Line 120 indicates that the loop over the levels is done from fine to coarse.
- 139-145 - The coefficients $\hat{c}_{ij}$ are found by singular value decomposition. These values are stored in the projectcache improve the efficiency of the algorithm.

```
1   @log.withcontext
2   def basis(self, name, *args, truncation_tolerance=1e-15, **kwargs):
3     '''Create hierarchical basis.
4
5     A hierarchical basis is constructed from bases on different levels of
6     uniform refinement. Two different types of hierarchical bases are
7     supported:
8
9     1. Classical — Starting from the set of all basis functions originating
10    from all levels of uniform refinement, only those basis functions are
11    selected for which at least one supporting element is part of the
12    hierarchical topology.
13
14    2. Truncated — Like classical, but with basis functions modified such that
15    the area of support is reduced. An additional effect of this procedure is
16    that it restores partition of unity. The spanned function space remains
17    unchanged.
18
19    Truncation is based on linear combinations of basis functions, where fine
20    level basis functions are used to reduce the support of coarser level basis
21    functions. See `Giannelli et al. 2012`_ for more information on truncated
22    hierarchical refinement.
23
24    Args
25    ----
26    name : :class:`str`
27      Type of basis function as provided by the base topology, with prefix
28      ''h-'' (''h-std'', ''h-spline'') for a classical hierarchical basis and
29      prefix ''th-'' (''th-std'', ''th-spline'') for a truncated hierarchical
30      basis.
31    truncation_tolerance : :class:`float` (default 1e-15)
32      In order to benefit from the extra sparsity resulting from truncation,
33      vanishing polynomials need to be actively identified and removed from the
34      basis. The ''trunctation_tolerance'' offers control over this threshold.
35
36    Returns
37    -------
38    basis : :class:`nutils.function.Array`
39    '''
40
41    if name.startswith('h-'):
42      truncated = False
43      name = name[2:]
44    elif name.startswith('th-'):
45      truncated = True
46      name = name[3:]
47    else:
48      return super().basis(name, *args, **kwargs)
```

```
49
50      # 1. identify active (supported) and passive (unsupported) basis functions
51      ubases = []
52      ubasis_active = []
53      ubasis_passive = []
54      prev_transforms = None
55      prev_ielems = []
56      map_indices = []
57      for i, (topo, touchielems_i) in reversed(tuple(enumerate(zip(self.levels, self._indices_per_level)))):
58        with log.context('level {} ({:.0f}%)'.format(i, 100 * (i+.5) / len(self.levels))):
59
60          topo_index_with_tail = topo.transforms.index_with_tail
61          mapped_prev_ielems = [topo_index_with_tail(prev_transforms[j])[0] for j in prev_ielems]
62          map_indices.insert(0, dict(zip(prev_ielems, mapped_prev_ielems)))
63          nontouchielems_i = numpy.unique(numpy.array(mapped_prev_ielems, dtype=int))
64          prev_ielems = ielems_i = numpy.unique(numpy.concatenate([numpy.asarray(touchielems_i, dtype=int), \
65                  nontouchielems_i], axis=0))
66          prev_transforms = topo.transforms
67
68          basis_i = topo.basis(name, *args, **kwargs)
69          assert isinstance(basis_i, function.Basis)
70          ubases.insert(0, basis_i)
71          # Basis functions that have at least one touchelem in their support.
72          touchdofs_i = basis_i.get_dofs(touchielems_i)
73          # Basis functions with (partial) support in this hierarchical topology.
74          partsuppdofs_i = numpy.union1d(touchdofs_i, basis_i.get_dofs(numpy.setdiff1d(ielems_i, touchielems_i, \
75                  assume_unique=True)))
76          # Mask of basis functions in 'partsuppdofs_i' with strict support in this hierarchical topology.
77          partsuppdofs_supported_i = numpy.array([numeric.sorted_contains(ielems_i, basis_i.get_support(dof)).all() \
78                  for dof in partsuppdofs_i], dtype=bool)
79          ubasis_active.insert(0, numpy.intersect1d(touchdofs_i, partsuppdofs_i[partsuppdofs_supported_i], \
80                  assume_unique=True))
81          ubasis_passive.insert(0, partsuppdofs_i[~partsuppdofs_supported_i])
82
83      *offsets, ndofs = numpy.cumsum([0, *map(len, ubasis_active)])
84
85      # 2. construct hierarchical polynomials
86      hbasis_dofs = []
87      hbasis_coeffs = []
88      projectcache = {}
89
90      for ilevel, (level, indices) in enumerate(zip(self.levels, self._indices_per_level)):
91        for ilocal in indices:
92
93          hbasis_trans = level.transforms[ilocal]
94          tail = hbasis_trans[len(hbasis_trans)-ilevel:]
95          trans_dofs = []
96          trans_coeffs = []
97
98          local_indices = [ilocal]
99          for m in reversed(map_indices[:ilevel]):
100           ilocal = m[ilocal]
101           local_indices.insert(0, ilocal)
102
103         if not truncated: # classical hierarchical basis
104
105           for h, ilocal in enumerate(local_indices): # loop from coarse to fine
106             mydofs = ubases[h].get_dofs(ilocal)
107
108             imyactive = numeric.sorted_index(ubasis_active[h], mydofs, missing=-1)
109             myactive = numpy.greater_equal(imyactive, 0)
110             if myactive.any():
111               trans_dofs.append(offsets[h]+imyactive[myactive])
112               mypoly = ubases[h].get_coefficients(ilocal)
113               trans_coeffs.append(mypoly[myactive])
114
115             if h < len(tail):
116               trans_coeffs = [tail[h].transform_poly(c) for c in trans_coeffs]
117
118         else: # truncated hierarchical basis
119
120           for h, ilocal in reversed(tuple(enumerate(local_indices))): # loop from fine to coarse
121             mydofs = ubases[h].get_dofs(ilocal)
122             mypoly = ubases[h].get_coefficients(ilocal)
123
124             truncpoly = mypoly if h == len(tail) \
125               else numpy.tensordot(numpy.tensordot(tail[h].transform_poly(mypoly), project[...,mypassive], \
126                     self.ndims), truncpoly[mypassive], 1)
127
128             imyactive = numeric.sorted_index(ubasis_active[h], mydofs, missing=-1)
129             myactive = numpy.greater_equal(imyactive, 0) & numpy.greater(abs(truncpoly), \
130                     truncation_tolerance).any(axis=tuple(range(1,truncpoly.ndim)))
131             if myactive.any():
```

```
132                 trans_dofs.append(offsets[h]+imyactive[myactive])
133                 trans_coeffs.append(truncpoly[myactive])
134 
135               mypassive = numeric.sorted_contains(ubasis_passive[h], mydofs)
136               if not mypassive.any():
137                 break
138 
139               try: # construct least-squares projection matrix
140                 project = projectcache[mypoly]
141               except KeyError:
142                 P = mypoly.reshape(len(mypoly), -1)
143                 U, S, V = numpy.linalg.svd(P) # (U * S).dot(V[:len(S)]) == P
144                 project = (V.T[:,:len(S)] / S).dot(U.T).reshape(mypoly.shape[1:]+mypoly.shape[:1])
145                 projectcache[mypoly] = project
146 
147           # add the dofs and coefficients to the hierarchical basis
148           hbasis_dofs.append(numpy.concatenate(trans_dofs))
149           hbasis_coeffs.append(numeric.poly_concatenate(trans_coeffs))
150 
151       return function.PlainBasis(hbasis_coeffs, hbasis_dofs, ndofs, self.transforms)
```

B Code of Scientific Conduct for Master's thesis

Technische Universiteit Eindhoven University of Technology

Declaration concerning the TU/e Code of Scientific Conduct for the Master's thesis

I have read the TU/e Code of Scientific Conduct[i].

I hereby declare that my Master's thesis has been carried out in accordance with the rules of the TU/e Code of Scientific Conduct

Date

15 - 04 - 2019

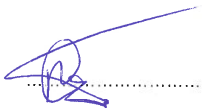
Name

Pieter van Zuijlen

ID-number

0829297

Signature



Submit the signed declaration to the student administration of your department.

[i] See: http://www.tue.nl/en/university/about-the-university/integrity/scientific-integrity/
The Netherlands Code of Conduct for Academic Practice of the VSNU can be found here also.
More information about scientific integrity is published on the websites of TU/e and VSNU

January 15 2016

Figure 27: Code of Scientific Conduct