Kjetil André Johannessen

Isogeometric Analysis Using Locally Refined B-splines

Thesis for the degree of Philosophiae Doctor

Trondheim, December 2014

Norwegian University of Science and Technology
Faculty of Information Technology, Mathematics
and Electrical Engineering
Department of Mathematical Sciences

NTNU – Trondheim
Norwegian University of
Science and Technology

NTNU
Norwegian University of Science and Technology

Thesis for the degree of Philosophiae Doctor

Faculty of Information Technology, Mathematics and Electrical Engineering
Department of Mathematical Sciences

ISBN 978-82-326-0668-9 (printed ver.)
ISBN 978-82-326-0669-6 (electronic ver.)
ISSN 1503-8181

Doctoral theses at NTNU, 2014:376

Printed by NTNU-trykk

To my family, near and far.

Preface

It has been said that the essence of mathematics is not to make simple things complicated, but to make complicated things simple. During my years pursuing a PhD I have come to embrace this statement. Too much research fades away into the void, simply due to the fact that it is presented in a way that is too complicated. Algorithms or mathematical theories, which might be novel and true, simply vanish out of existence after the initial researcher stops working on them, since it is too hard for new young researchers to carry on the torch. Much of my work, have thus involved trying to simplify existing methodologies and making them more available to the scientific public. Working in a paper form, this is by no means an easy task. Too often you are confronted with either specifying too much or too little resulting in confusion either way. It is a fine line to walk and I hope that at the end of my work, I have conveyed some of the ideas to a new and broader audience.

It is a pleasure to acknowledge the generous help of a number of people in the preparation of this work. First of all, I am in debt to Trond Kvamsdal who has been my advisor throughout this thesis. Without his constant feedback, advice, inspiration and guidance, this thesis would never have seen the light of day. Also a large thanks to Thomas J.R. Hughes, Mike Scott, Yuri Bazilevs and the isogeometric community which have not only been great sources of inspiration, but have given valuable feedback at multiple conferences. I would also like to take this opportunity to acknowledge SINTEF ICT, and especially Tor Dokken, Vibeke Skytt and Kjell Fredrik Pettersen. You have been an unparalleled resource in anything related to spline theory. I would like to thank to my fellow researchers Mukesh Kumar, Arne Morten Kvarving, Knut Nordanger, Knut Morten Okstad, Siv Bente Raknes, Filippo Remonato and Annette Stahl for countless hours of both work and fun. Last, but not least, I would like to thank my family Linda, Victoria and Fredrik for their continued support and understanding during late working hours. Love does not make the world go 'round, it's what makes the ride worthwhile.

Contents

Introduction

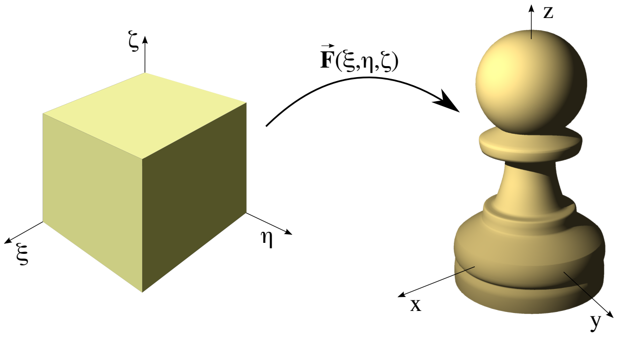


Figure 1: NURBS allow for a convenient geometric mapping from a simple parametric domain (ξ, η, ζ) to a complex physical domain (x, y, z).

1 A brief history of Isogeometry

Computer Aided Design (CAD) and Finite Element Analysis (FEA) are essential technologies in modern product development. However, the interoperability of these technologies is severely disturbed by inconsistencies in the mathematical approaches used. The main reason for inconsistencies is that the technologies evolved in different communities with the focus on improving disjoint stages in product development processes, and taking little heed on relations to other stages. Efficient feedback from analysis to CAD and refinement of the analysis model are essential for computer-based design optimization and virtual product development. The current lack of efficient interoperability of CAD and FEA makes refinement and adaptation of the analysis model cumbersome, slow and expensive.

The new paradigm of Isogeometric Analysis, which was introduced by Hughes *et al.* [33], demonstrates that much is to be gained with respect to efficiency, quality and accuracy in analysis by replacing traditional Finite Elements by volumetric NURBS elements.

The term *isoparametric* methods has already been established and was based on the idea that one should use the same basis for the unknown field variables and the geometry. What was different when Hughes introduced the word *isogeometry* was that it would be the geometry that would dictate what basis should be used. This was in contrast to previous paradigms where one had a convenient discretization of the field variables, and tried creating an appropriate geometry in that same basis. For isogeometry this is turned around and we would have a convenient geometry discretization and from this create the basis for the field variables.

The idea was quickly embraced and got widespread attention. Isogeometry has been applied to a variety of problems of engineering interests, such as flow simulations [1, 12, 13, 19, 22, 27, 25, 26], electromagnetic problems [20, 21, 39], structural engineering [2, 38, 28] and biomechanics [55, 11].

It was quickly discovered that the smooth spline functions offered far more than simply convenience. They proved more accurate per degree of freedom, allowed for the discretization of high-order differential equations [15, 14] in addition to providing stable solutions, even for high order [10].

While initially isogeometry was considered equivalent to using non-uniform rational B-splines (NURBS) as basis in the finite element method, later years have seen the term grow beyond this. It has been applied to collocation methods [3, 44], multigrid methods [30] and finite volume methods [32].

However for all its strengths and advantages, NURBS have some flaws that make them not flexible enough as a common basis for future CAD and FEA. They are defined by patchwise tensor product. This means that they are in general not watertight, lack local refinement and do not accommodate extraordinary points.

2 A brief history of local refinement

T-splines are a recently developed generalization of NURBS [48, 47], they were introduced to cure the above geometric limitations and to generate local refinements in the mesh. It is interesting to note that the work on T-splines was initially a CAD endeavor and these introductory papers were published before isogeometry had become a word in 2005 [33]. As a CAD technology, T-splines had a few shortcomings which made them inconvenient for FEA, such as linear dependence [18] and refinement propagation [24]. In light of the strict requirements of isogeometric analysis, a new sub-class of T-splines: analysis-suitable (AS) T-splines [46, 37] emerged, which is a significant step towards more versatility.

T-splines were however not the first technology to attempt local refinement for smooth spline functions. In 1988 Forsey and Bartels introduced the Hierarchically refined B-splines [29], which have seen a rejuvenation in later years with applications in isogeometry [52, 51, 16, 42, 43]. Giannelli *et al.* [31, 35] published later a generalization on these, called Truncated Hierarchical B-splines which recaptured several properties of NURBS that Hierarchical B-splines had lost such as partition of unity and strong stability.

We believe that the recently proposed locally refined LR B-splines by Dokken *et al.* [23] may have the potential to form an alternative framework for future interoperable CAD and FEA systems. The new approach directly operates on the spline spaces, and in this way a broad spectrum of piecewise spline functions may be obtained. LR B-splines consist of smooth, piecewise polynomial basis functions that constitute a partition of unity. Among other advanced features they may facilitate local h-refinement. Since this class of splines is rich and versatile, it may break new ground and seems to be attractive as foundation for integrating CAD and FEA on one computational platform.

3 A brief history of error estimation

Since 1970s several strategies have been developed to estimate the discretization error of Finite Element (FE) solution. A first posteriori error estimates were introduced by Babuska and Rheinboldt in 1978, see [5, 4]. Since then many different estimation procedures have been introduced. The existing techniques to obtain an energy estimates may be classified into two main categories:

- *Residual based estimates*: The approximate FE solution does not satisfy the governing partial differential equation. This lack of fulfillment is called the residual and the error can be estimated by solving local problems where the load functions are given by the local residuals.

- *Recovery-based estimates*: These estimates employ a projection technique in order to recover a post-processed quantity (usually the stresses) from the FE solution. The error is then estimated by taking the difference between the recovered solution and the FE solution.

The use of a posteriori error estimator in isogeometric analysis is still in its infancy. To the best of our knowledge only few work has been done in this direction, see [17, 24, 34, 36, 45, 49, 51, 52, 54, 53]. The authors in [24] used the idea of hierarchical bases with bubble functions approach of Bank and Smith [9] to design a posteriori error estimator for T-splines, which was also used in [17, 50]. But their performance was less satisfactory due to the needed saturation assumption as noted on page 41 of [34]. Another simple idea of explicit residual based error estimator has been explored in [34, 49, 52, 54, 53]. They require the computation of constants in Clement-type interpolation operators. Such constant are mesh (element) dependent and often incomputable for general element shape. A global constant can overestimates the local constants, and thus the exact error. Recently, a functional-type a posteriori error estimate for isogeometric discretization is presented in [36]. These type of error estimate, which was introduced in [40, 41] on functional grounds (including integral identity and functional analysis arguments) are applicable for any conforming and non-conforming discretizations and known to provide a guaranteed and computable error bounds. But the hindrance in their popularity is due to high cost of computations which are based on solving a global minimization problem (Majorant minimization problem) in H(div) spaces. In [36], authors made an attempt to to reduce the cost of computations for tensorial spline spaces but the same idea of cost reduction need further study in adaptive isogeometric analysis. In this article we explore another approach to design a posteriori error estimate in setting of Zienkiewicz-Zhu [56] where the improved gradient obtained from recovery procedure is used instead of exact gradient of solution. The recovery based estimators are very popular in engineering community because of their simple implementation and as they provide good effectivity indices. In an extensive study on the quality of different a posteriori error estimates belonging to first two categories above (residual based vs. recovery based), Babuska and co-workers in [7, 6, 8]; conclude that the Superconvergent Patch Recovery (SPR) technique developed by Zienkiewicz and Zhu [57, 58] is the most robust estimator for the class of smooth solutions approximated on patch-wise uniform grids of linear or quadratic elements. In this thesis, we develop recovery based error estimates for isogeometric discretization and verify their effectiveness for quadratic B-splines and quadratic LR B-splines elements in adaptive isogeometric analysis.

4 Outline of the Thesis

The thesis is divided into 5 parts: one introduction and 4 individual papers, all submitted for publication, or accepted in the journal of Computer Methods in Applied Mechanics and Engineering. In the introduction we will briefly introduce the core concepts of B-splines and locally refined B-splines. The LR B-spline software used in the preparation of this work have been made publicly available as open source. This is discussed in the appendix. The introduction is meant to give a very short summary, while the papers contain the necessary details.

- The first paper investigates the use of LR B-splines in finite element method.
- The second paper highlights similarities and differences with a similar technology: the hierarchical B-splines.
- The third paper develops an a posteriori error estimator for isogeometric analysis.
- The fourth paper discusses the use of local refinement and compatible discretization for Stokes problems.

5 B-splines

Consider a knot vector of non-decreasing knots $\{\xi_i\}_{i=1}^{n+p+1}$. This partitions the parametric domain into elements with a given smoothness across each knot. We can construct a basis on the domain $[\xi_{p+1}, \xi_{n+1}]$ by piecewise smooth polynomials using the Cox-de Boor recursion formula (1)

$$\begin{aligned} N_{i,p}(\xi) &= \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} N_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} N_{i-1,p-1}(\xi) \\ N_{i,0}(\xi) &= \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{else} \end{cases} \end{aligned} \tag{1}$$

where, by slight abuse of notation, we define that $\frac{0}{0} := 0$ and p is the polynomial degree of the basis. It is customary (but not required) that the knot vector is *open*, that is the first $p + 1$ entires are equal as well as the last $p + 1$ entires are equal. In Figure 2 we show an example of a basis constructed on a uniform open knot vector. The knot vector holds all the information of the basis constructed. In particular, the following holds true

- the B-splines N_i are polynomial and C^∞ in between the knots
- the B-splines are C^{p-m} at the knots, where m is the knot multiplicity
- the B-splines are non-negative everywhere
- they satisfy the partition of unity, i.e. $\sum N_i = 1$
- each B-spline is depends on exactly $p + 2$ knots.

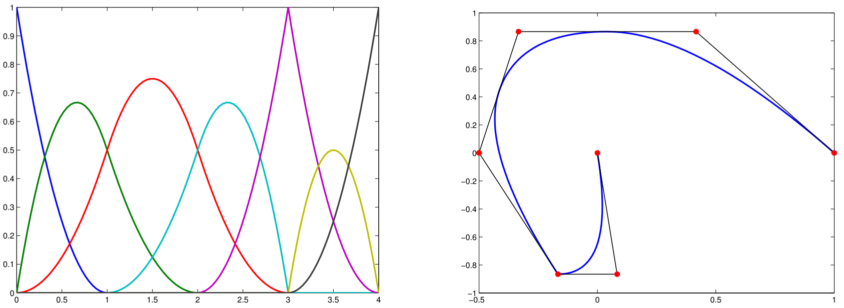
It is the last point, which will allow us to define a local knot vector corresponding to each function, and this observation will be utilized below to introduce LR B-splines.

It is possible to create a parametric curve using all B-splines created by a single knot vector Ξ by multiplicating with control points $\boldsymbol{c}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}$

$$\boldsymbol{x}(\xi) = \sum_{i=1}^{n} N_i(\xi) \boldsymbol{c}_i$$

Moving to surfaces we define the set of bivariate basis functions as the tensor product of two one-dimensional knot vectors Ξ and Ψ, and a surface can then be represented as

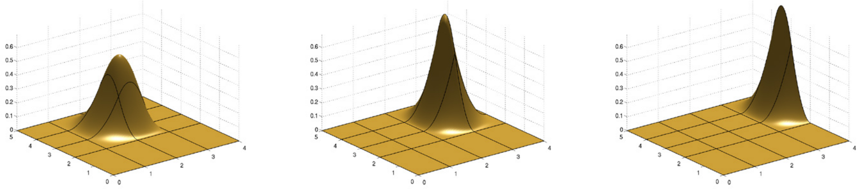
$$\begin{aligned} \boldsymbol{x}(\xi, \eta) &= \sum_{j=1}^{m} \sum_{i=1}^{n} B_{ij}(\xi, \eta) \boldsymbol{c}_{ij} \\ &= \sum_{j=1}^{m} \sum_{i=1}^{n} N_i(\xi) N_j(\eta) \boldsymbol{c}_{ij} \end{aligned}$$



(a) All basis functions generated by the knot vector Ξ

(b) Parametric curve (blue) given by control points, here shown in red

Figure 2: All quadratic basis functions generated by the knot $\Xi = [0,0,0,1,2,3,3,4,4,4]$. By multiplication with a set of control points, we are able to construct a parametric curve.



(a) $B_{4,4}(\xi,\eta)$

(b) $B_{5,4}(\xi,\eta)$

(c) $B_{7,4}(\xi,\eta)$

Figure 3: Three bivariate functions. By combining *all* functions corresponding to the knot vector $\Xi = [0,0,0,1,2,3,3,4,4,4]$ and $\Psi = [0,0,0,1,3,4,5,5,5]$ we are creating a tensor product basis

where $N_i(\xi)$ is created by a knot vector Ξ and $N_j(\eta)$ is created by a knot vector Ψ. By combining all N_i with all N_j we are effectively creating a tensor product. While traditionally the word "B-splines" refers to the collective set of all functions $\{B_{ij}\}$, or the mapped geometry $\boldsymbol{x}(\xi,\eta)$, we will here also use it to address a single function.

To construct *rational* B-splines (NURBS), we introduce a rational weight w corresponding to each B-spline and let the rational functions be defined as $R(\xi,\eta)$ such that

$$R_{ij}(\xi,\eta) = \frac{B_{ij}(\xi,\eta)}{\sum_{\hat{i}}\sum_{\hat{j}} B_{\hat{i},\hat{j}}(\xi,\eta)w_{\hat{i}\hat{j}}}w_{ij}$$

Note that it is always possible to create a rational representation R of any B-spline discretization B by adding rational weights. This work will focus on the creation of B based on local refinement and the extension to rational functions is considered straightforward.

6 Finite Element Method

Assume as a model problem that we are going to solve the Poisson equation

$$\begin{aligned} \nabla^2 u &= f, \quad \text{in } \Omega \\ u &= 0, \quad \text{on } \partial\Omega \end{aligned}$$

by multiplying with a test function v, and integrating over the domain Ω, we arrive at the weak form: Find $u \in X$ such that

$$\begin{aligned} a(u,v) &= b(v) \\ a(u,v) &= \int_\Omega \nabla u \nabla v \, dA \\ b(v) &= \int_\Omega f v \, dA. \end{aligned}$$

We then choose a finite-dimensional subspace $X_h \subset X$ and reformulate this as find $u_h \in X_h$ such that

$$a(u_h, v_h) = b(v_h), \quad \forall v_h \in X_h.$$

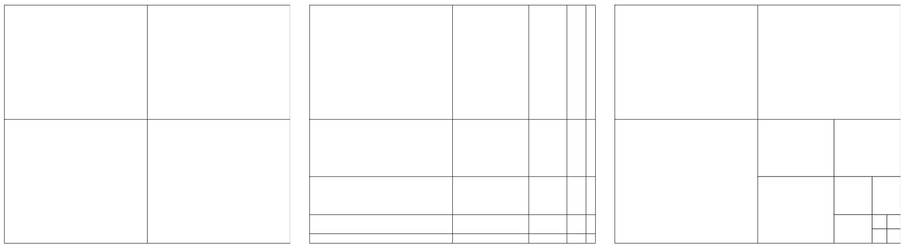
It can be shown that this is equivalent to solving the linear system of equations

$$\begin{aligned} Au &= b, \quad \text{where} \\ A_{ij} &= \int_\Omega \nabla N_i \nabla N_j \, dA \\ b_i &= \int_\Omega f N_i \, dA \end{aligned}$$

and $X_h = \text{span}\{N_i\}$. The most notable thing with this framework is the choice of space X_h. By creating this using NURBS, we are entering the realm of isogeometry. We will, in this work, however create it as a generalization on the NURBS, and construct X_h by using Locally Refined B-splines.

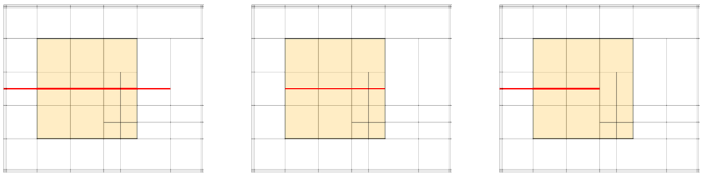
7 LR B-splines

Traditional NURBS and B-splines are as stated above constructed by tensor products. Locally refined B-splines are a way to alleviate this restriction



(a) Initial mesh (b) Tensor product refinement (c) Truly local refinement

Figure 4: For tensorial meshes, all lines expand the entire parametric domain. Locally refined B-splines allow us to terminate these mesh lines prior to this and create a much more localized refinement



(a) Line traversing B (b) Line traversing B (c) Line not traversing B

Figure 5: Traversing the support of a basis function. The refinement algorithm simply checks the rectangular support of basis functions, and if these are traversed by the straight lines through the parametric domain.

and allows for the construction of a basis on a more general mesh which may include T-joints.

When talking about LR B-splines, we usually distinguish between the mesh $\mathcal{M}$ and the set of B-splines $\mathcal{S}$. The mesh is represented by the set of all lines; vertical and horizontal. The function space $\mathcal{S}$ is represented by the B-splines themselves, which are uniquely determined by their local knot vectors of length $p + 2$. The refinement algorithm is the interplay in between these two entities and is categorized by two operations: traversing and splitting.

A line in the mesh $\mathcal{M}$ is said to **traverse** a B-spline B_i if it passes through its support, and all of its support.

See Figure 5 for examples on traversing meshlines.

A knot-line is said to **exist** in a B-spline $B_{\Xi,\Psi}$ if its (constant) knot value is represented in Ξ for vertical lines or Ψ for horizontal lines.

A B-spline B_i can be **split** at the knot ξ (or η), producing two new B-splines B_1 and B_2. When inserting the two new B-splines into the existing space $\mathcal{S}$, these will be updated if they exist already, or we create a new entry if they do not exist.

Algorithm 1 Refinement algorithm

```
 1: Insert new line E
 2: for every B-spline $B_i \in \mathcal{S}$ do
 3:    if E traverse $B_i$ and E does not exist in $B_i$ then
 4:       split $B_i$
 5:    end if
 6: end for
 7: for every newly created B-spline $B_j$ from line 4 or 10 do
 8:    for every existing line $E \in \mathcal{M}$ do
 9:       if E traverse $B_j$ and E does not exist in $B_j$ then
10:          split $B_j$
11:       end if
12:    end for
13: end for
```

Locally Refined B-splines is a constructive algorithm to locally refine B-splines by progressively splitting single functions into smaller ones. It consists in all its simplicity by checking if lines traverse rectangular support, which triggers the creation of new functions.

8 Compatible discretizations and spline derivatives

LR B-splines generates a smooth basis over a given mesh. It is often interesting to look at *derivatives* of these functions and spaces of derivatives. This has applications in both error estimation and finite element analysis. Once such application is for the steady stokes equation given by

$$\begin{aligned} -\mu\nabla^2\boldsymbol{u} + \nabla p &= \boldsymbol{f} \quad \text{in } \Omega \\ \nabla \cdot \boldsymbol{u} &= 0 \quad \text{in } \Omega. \end{aligned}$$

In this equation we have that the velocity $\boldsymbol{u}$ is given by one more derivative than the pressure p. To create a compatible discretization we let p live in the "derivative space" of $\boldsymbol{u}$. Formally, this can be described as letting the following De Rham complex be exact

$$\mathbb{R} \to X_h^0 \xrightarrow{\mathbf{rot}} X_h^1 \xrightarrow{\text{div}} X_h^2 \to 0 \tag{2}$$

and letting the discrete solutions $\boldsymbol{u}_h \in X_h^1$ and $p_h \in X_h^2$.

For one-dimensional problems derivative spaces are fairly predictive. Assume we have a knot vector $\Xi = [0,0,0,0,1,2,3,3,4,4,4,4]$ which generates a set of cubic basis functions. These will be C^1 at $\xi = 3$ and C^2 at all other internal knots. Differentiating this lowers the polynomial degree by one and the continuity by one and we are left with a quadratic basis which is C^0 at $\xi = 3$ and C^1 at all other knots. A good guess for the knot vector would thus be $\Xi = [0,0,0,1,2,3,3,4,4,4]$ and indeed it can be shown that this is the case. By removing the first and last knot from the knot vector in an univariate basis, we are able to create the derivative space.

For 2D this becomes slightly more involved. The same trick may however be applied for a tensor product basis. Let a basis $\mathcal{S} = \text{span}\{B_i\}$ be spanned by the global knot vectors Ξ and Ψ such that

$$\begin{aligned}
\Xi &= [\xi_1, \xi_2, \xi_3, ..., \xi_{n+p-1}, \xi_{n+p}, \xi_{n+p+1}] \\
\Xi' &= [\quad \xi_2, \xi_3, ..., \xi_{n+p-1}, \xi_{n+p} \qquad] \\
\Psi &= [\eta_1, \eta_2, \eta_3, ..., \eta_{n+p-1}, \eta_{n+p}, \eta_{m+q+1}] \\
\Psi' &= [\quad \eta_2, \eta_3, ..., \eta_{n+p-1}, \eta_{n+p} \qquad]
\end{aligned}$$

and let $\partial_x \mathcal{S} = \text{span}\{\frac{\partial B_i}{\partial x}\}$ and $\partial_y \mathcal{S} = \text{span}\{\frac{\partial B_i}{\partial y}\}$. It can be shown that $\partial_x \mathcal{S}$ is constructed by the knot vectors Ξ' and Ψ, while $\partial_y \mathcal{S}$ is constructed by Ξ and Ψ'. Note that the smaller knot vectors always create a smaller derivative space. This is however only the case for tensorial meshes.

For locally refined meshes, the derivative space might actually grow larger than the initial space $\mathcal{S}$. This might seem counterintuitive at first, but it is linked to the fact that differentiating decreases the polynomial degree which reduces the size of the space, but also reduces continuities which *increases* the size of the space. Note that in the case of (2) we are not only considering derivatives, we are moving between spaces taking the divergence or $\mathbf{rot}$.

We will in the last paper investigate this further and show that it is possible to construct a set of basis for X_0, X_1 and X_2 such that (2) is exact.

9 Summary of Papers

9.1 Paper I: Isogeometric analysis using LR B-splines

This is the first paper written on the use of LR B-splines as a basis for finite element analysis. It gives an in-depth introduction into the fundamentals of LR B-splines, which is mostly a restatement of the paper by Dokken *et al.*[23] written using simpler linguistic terms. While the refinement algorithm states what to do when a new line is inserted, it does not tell you which lines to insert. The paper proposes different refinement strategies and investigates their performance on actual differential equations. It summarizes with extensive numerical tests, showing that LR B-splines perform optimally on problems with local features or singularities.

9.2 Paper II: On the similarities and differences between Classical Hierarchical, Truncated Hierarchical and LR B-splines

Since the first paper suggests that the structured mesh refinement is a strong candidate for iterative refinement, this raises the natural question of the difference between Hierarchical B-splines and structured mesh refinement in LR B-splines. Indeed, they share a great deal of properties, and to the untrained eye, they might look identical. The paper tries to create a unified notation to highlight some of the similarities and differences which appear in these two technologies. It concludes by observing that they both contain different functions, and span different functions spaces on identical meshes. The conditioning numbers of mass and stiffness matrices are shown to be up to twice the size for Classical Hierarchical B-splines.

9.3 Paper III: Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis

In order to produce adaptive meshes in finite element analysis we are dependent on an indicator to drive the refinement. For academic test problems, we often have an exact solution, but this is not true in general. This paper investigates a posteriori error estimators for the smooth basis produced by LR B-splines. It is shown that there exist super-convergent points which display higher accuracy than the global solution, and moreover that these points are computable. Based on a patch-recovery technique this allows us to recover a better representation of the computed derivative, which in turn produces a very accurate error estimator.

9.4 Paper IV: Divergence-conforming discretization for Stokes problem on locally refined meshes using LR B-splines

This paper investigates the relation between different continuities and polynomial degrees over the same locally refined mesh. It is shown that by simply altering the polynomial degree over the same mesh, we are able to construct a H-div compatible discretization satisfying a discrete de Rham complex. The elegance of using the *same* mesh to build all the different basis is striking and all results from the tensor product case follow directly. We show that the solution is pointwise divergent-free, stable and shows optimal convergence rates for problems with local features of singularities.

Bibliography

[1] I. Akkerman et al. “The role of continuity in residual-based variational multiscale modeling of turbulence”. In: *Comput. Mech.* 41.3 (2008), pp. 371–378.

[2] F. Auricchio et al. “A fully “locking-free” isogeometric approach for plane linear elasticity problems: a stream function formulation”. In: *Comput. Methods Appl. Mech. Engrg.* 197.1-4 (2007), pp. 160–172.

[3] F. Auricchio et al. “Isogeometric collocation for elastostatics and explicit dynamics”. In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012). Higher Order Finite Element and Isogeometric Methods, pp. 2–14.

[4] I. Babuška and W. C. Rheinboldt. “A posteriori error estimates for the finite element method”. In: *Internat. J. Numer. Methods Eng.* 12 (1978), pp. 1597–1615.

[5] I. Babuška and W. C. Rheinboldt. “Error estimates for adaptive finite element computations”. In: *SIAM J. Numer. Anal.* 15.4 (1978), pp. 736–754.

[6] I. Babuška, T. Strouboulis, and C. S. Upadhyay. “A model study of the quality of a posteriori error estimators for finite element solutions of linear elliptic problems, with particular reference to the behavior near the boundary”. In: *Internat. J. Numer. Methods Engrg.* 40.14 (1997), pp. 2521–2577.

[7] I. Babuška, T. Strouboulis, and C. S. Upadhyay. “A model study of the quality of a posteriori error estimators for linear elliptic problems. Error estimation in the interior of patchwise uniform grids of triangles”. In: *Comput. Methods Appl. Mech. Engrg.* 114.3-4 (1994), pp. 307–378.

[8] I. Babuška et al. “Validation of a posteriori error estimators by numerical approach”. In: *Internat. J. Numer. Methods Engrg.* 37.7 (1994), pp. 1073–1123.

[9] R. E. Bank and R. K. Smith. “A posteriori error estimates based on hierarchical bases”. In: *SIAM J. Numer. Anal.* 30.4 (1993), pp. 921–935.

[10] Y. Bazilevs et al. “Isogeometric analysis: approximation, stability and error estimates for h-refined meshes”. In: *Mathematical Models and Methods in Applied Sciences* 16 (2006), pp. 1031–1090.

[11] Y. Bazilevs et al. “Isogeometric Fluid–structure Interaction Analysis with Applications to Arterial Blood Flow”. English. In: *Computational Mechanics* 38.4–5 (2006), pp. 310–322.

[12] Y. Bazilevs et al. “Isogeometric fluid-structure interaction: theory, algorithms, and computations”. In: *Comput. Mech.* 43.1 (2008), pp. 3–37.

[13] Y. Bazilevs et al. “Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes”. In: *Comput. Methods Appl. Mech. Engrg.* 199.13-16 (2010), pp. 780–790.

[14] M. J. Borden et al. “A higher-order phase-field model for brittle fracture: formulation and analysis within the isogeometric analysis framework”. In: *Comput. Methods Appl. Mech. Engrg.* 273 (2014), pp. 100–118.

[15] M. Borden et al. “A phase-field description of dynamic brittle fracture”. In: *Computer Methods in Applied Mechanics and Engineering* 217–220.0 (2012), pp. 77 –95.

[16] P. Bornemann and F. Cirak. “A subdivision-based implementation of the hierarchical B-spline finite element method”. In: *Computer Methods in Applied Mechanics and Engineering* (2012).

[17] A. Buffa, D. Cho, and M. Kumar. “Characterization of T-splines with reduced continuity order on T-meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 201-204.0 (2012), pp. 112–126.

[18] A. Buffa, D. Cho, and G. Sangalli. “Linear independence of the T-spline blending functions associated with some particular T-meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 199.23-24 (2010), pp. 1437 –1445.

[19] A. Buffa, C. de Falco, and G. Sangalli. “IsoGeometric Analysis: Stable elements for the 2D Stokes equation”. In: *International Journal for Numerical Methods in Fluids* 65.11-12 (2011), pp. 1407–1422.

[20] A. Buffa, G. Sangalli, and R. Vázquez. “Isogeometric analysis in electromagnetics: B-splines approximation”. In: *Comput. Methods Appl. Mech. Engrg.* 199.17-20 (2010), pp. 1143–1152.

[21] A. Buffa, G. Sangalli, and R. Vázquez. “Isogeometric methods for computational electromagnetics: B-spline and T-spline discretizations”. In: *J. Comput. Phys.* 257.part B (2014), pp. 1291–1320.

[22] K. Chang, T. J. R. Hughes, and V. M. Calo. “Isogeometric variational multiscale large-eddy simulation of fully-developed turbulent flow over a wavy wall”. In: *Comput. & Fluids* 68 (2012), pp. 94–104.

[23] T. Dokken, T. Lyche, and K. Pettersen. “Polynomial splines over locally refined box-partitions”. In: *Comput. Aided Geom. Des.* 30.3 (Mar. 2013), pp. 331–356.

[24] M. R. Dörfel, B. Jüttler, and B. Simeon. “Adaptive isogeometric analysis by local h-refinement with T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 264 –275.

[25] J. A. Evans and T. J. R. Hughes. “Isogeometric divergence-conforming B-splines for the Darcy-Stokes-Brinkman equations”. In: *Math. Models Methods Appl. Sci.* 23.4 (2013), pp. 671–741.

[26] J. A. Evans and T. J. R. Hughes. “Isogeometric divergence-conforming B-splines for the steady Navier-Stokes equations”. In: *Math. Models Methods Appl. Sci.* 23.8 (2013), pp. 1421–1478.

[27] J. Evans. “Divergence-free B-spline Discretizations for Viscous Incompressible Flows”. PhD thesis. The University of Texas at Austin, 2011.

[28] P. Fischer et al. “Isogeometric analysis of 2D gradient elasticity”. English. In: *Computational Mechanics* 47.3 (2011), pp. 325–334.

[29] D. Forsey and R. Bartels. “Hierarchical B-spline refinement”. In: *ACM SIGGRAPH Computer Graphics* 22.4 (1988), pp. 205–212.

[30] Gahalaut K.P.S., Kraus J.K., and Tomar S.K. “Multigrid methods for isogeometric discretization”. In: *Computer Methods in Applied Mechanics and Engineering* 253.0 (2013), pp. 413–425.

[31] C. Giannelli, B. Jüttler, and H. Speleers. “THB-splines: The Truncated basis for hierarchical splines”. In: *Computer Aided Geometric Design* 29.7 (2012), pp. 485 –498.

[32] C. Heinrich, B. Simeon, and S. Boschert. “A finite volume method on NURBS geometries and its application in isogeometric fluid–structure interaction”. In: *Mathematics and Computers in Simulation* 82.9 (2012), pp. 1645–1666.

[33] T. Hughes, J. Cottrell, and Y. Bazilevs. “Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement”. In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195.

[34] K. A. Johannessen. *An adaptive isogeometric finite element analysis.* Master thesis, Norwegian University of Science and Technology, Norway. 2009.

[35] G. Kiss, C. Giannelli, and B. Jüttler. *Algorithms and Data Structures for Truncated Hierarchical B-splines.* Tech. rep. 14. Johannes Kepler University, 2012.

[36] S. K. Kleiss and S. Tomar. "Guaranteed and sharp a posteriori error estimates in isogeometric analysis". In: *Preprint, arXiv:1304.7712* (2013).

[37] X. Li and M. Scott. "Analysis-suitable T-splines: Characterization, refineability, and approximation". In: *Mathematical Models and Methods in Applied Sciences* 24.06 (2014), pp. 1141–1164. eprint: `http://www.worldscientific.com/doi/pdf/10.1142/S0218202513500796`.

[38] N. Nguyen-Thanh et al. "Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids". In: *Computer Methods in Applied Mechanics and Engineering* 200.21–22 (2011), pp. 1892–1908.

[39] A. Ratnani and E. Sonnendrücker. "An arbitrary high-order spline finite element solver for the time domain Maxwell equations". In: *J. Sci. Comput.* 51.1 (2012), pp. 87–106.

[40] S. I. Repin. "A posteriori error estimates for approximate solutions to variational problems with strongly convex functionals". In: *J. Math. Sci. (New York)* 97.4 (1999). Problems of mathematical physics and function theory, pp. 4311–4328.

[41] S. I. Repin. "A posteriori error estimation for variational problems with uniformly convex functionals". In: *Math. Comp.* 69.230 (2000), pp. 481–500.

[42] D. Schillinger and E. Rank. "An unfitted hp-adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry". In: *Computer Methods in Applied Mechanics and Engineering* 200.47 - 48 (2011), pp. 3358 –3380.

[43] D. Schillinger et al. "An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces". In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012), pp. 116 –150.

[44] D. Schillinger et al. “Isogeometric collocation: Cost comparison with Galerkin methods and extension to adaptive hierarchical NURBS discretizations”. In: *Computer Methods in Applied Mechanics and Engineering* 267.0 (2013), pp. 170 –232.

[45] M. A. Scott, D. C. Thomas, and E. J. Evans. “Isogeometric spline forests”. In: *Comput. Methods Appl. Mech. Engrg.* 269 (2014), pp. 222–264.

[46] M. Scott et al. “Local refinement of analysis-suitable T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 213 (2012), pp. 206–222.

[47] T. Sederberg et al. “T-spline simplification and local refinement”. In: *ACM Transactions on Graphics* 23.3 (2004), pp. 276–283.

[48] T. Sederberg et al. “T-splines and T-NURCCs”. In: *ACM Transactions on Graphics* 22.3 (2003), pp. 477–484.

[49] L. Tian, F. Chen, and Q. Du. “Adaptive finite element methods for elliptic equations over hierarchical T-meshes”. In: *J. Comput. Appl. Math.* 236.5 (2011), pp. 878–891.

[50] A.-V. Vuong et al. “A hierarchical approach to adaptive local refinement in isogeometric analysis”. In: *Comput. Methods Appl. Mech. Engrg.* 200.49-52 (2011), pp. 3554–3567.

[51] A. Vuong et al. “A hierarchical approach to adaptive local refinement in isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 200.49-52 (2011), pp. 3554–3567.

[52] P. Wang et al. “Adaptive isogeometric analysis using rational PHT-splines”. In: *Computer-Aided Design* 43 (2011), pp. 1438–1448.

[53] G. Xu et al. “Optimal analysis-aware parameterization of computational domain in 3D isogeometric analysis”. In: *Comput.-Aided Des.* 45.4 (2013), pp. 812–821.

[54] G. Xu et al. “Parameterization of computational domain in isogeometric analysis: methods and comparison”. In: *Comput. Methods Appl. Mech. Engrg.* 200.23-24 (2011), pp. 2021–2031.

[55] Y. Zhang et al. “Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow”. In: *Comput. Methods Appl. Mech. Engrg.* 196.29-30 (2007), pp. 2943–2959.

[56] O. C. Zienkiewicz and J. Z. Zhu. “A simple error estimator and adaptive procedure for practical engineering analysis”. In: *Internat. J. Numer. Methods Engrg.* 24.2 (1987), pp. 337–357.

[57] O. C. Zienkiewicz and J. Z. Zhu. “The superconvergent patch recovery and a posteriori error estimates. I. The recovery technique”. In: *Internat. J. Numer. Methods Engrg.* 33.7 (1992), pp. 1331–1364.

[58] O. C. Zienkiewicz and J. Z. Zhu. “The superconvergent patch recovery and a posteriori error estimates. II. Error estimates and adaptivity”. In: *Internat. J. Numer. Methods Engrg.* 33.7 (1992), pp. 1365–1382.

Paper I

Isogeometric analysis using LR B-splines

Kjetil André Johannessen, Trond Kvamsdal and
Tor Dokken

Published in *Computer Methods in Applied Mechanics and
Engineering* 02/2014; 269(1):471-514

Isogeometric Analysis Using LR B-splines

Kjetil André Johannessen, Trond Kvamsdal, and Tor Dokken
Department of Mathematical Sciences
Norwegian University of Science and Technology, Trondheim, Norway
Department of Applied Mathematics, SINTEF ICT, Norway
e–mail: Kjetil.Johannessen@math.ntnu.no, Trond.Kvamsdal@math.ntnu.no, and
Tor.Dokken@sintef.no

Abstract

The recently proposed locally refined B-splines, denoted LR B-splines, by Dokken *et al.* [6] may have the potential to be a framework for isogeometric analysis to enable future interoperable computer aided design and finite element analysis. In this paper, we propose local refinement strategies for adaptive isogeometric analysis using LR B-splines and investigate its performance by doing numerical tests on well known benchmark cases. The theory behind LR B-spline is not presented in full details, but the main conceptual ingredients are explained and illustrated by a number of examples.

1 Introduction

1.1 Background

Computer Aided Design (CAD) and Finite Element Analysis (FEA) are essential technologies in modern product development. However, the interoperability of these technologies is severely disturbed by inconsistencies in the mathematical approaches used. The main reason for inconsistencies is that the technologies evolved in different communities with the focus on improving disjoint stages in product development processes, and taking little heed on relations to other stages. Efficient feedback from analysis to CAD and refinement of the analysis model are essential for computer-based design optimization and virtual product development. The current lack of efficient interoperability of CAD and FEA makes refinement and adaptation of the analysis model cumbersome, slow and expensive.

The new paradigm of Isogeometric Analysis, which was introduced by Hughes *et al.* [11], demonstrates that much is to be gained with respect to efficiency, quality and accuracy in analysis by replacing traditional Finite Elements by volumetric NURBS elements.

NURBS are not flexible enough to be a common basis for future CAD and FEA merely due to some required properties in design and analysis such as locally refineable, accommodate extraordinary points, and trimless option. T-splines are a recently developed generalization of NURBS [2], [7],

[20], they were introduced to cure the above geometric limitations and to generate local refinements in the mesh. In context of isogeometric analysis, a new sub-class of T-splines as analysis-suitable (AS) T-splines [19] have emerged, which is a significant step towards more versatility. Recently there has also been published works related to hierarchical refinement of splines introduced by Forsey and Bartels [8]; see [22], [21], [9], [4], [16], and [17].

We believe that the recently proposed locally refined LR B-splines by Dokken *et al.* [6] may have the potential to form an alternative framework for future interoperable CAD and FEA systems. The new approach directly operates on the spline spaces, and in this way a broad spectrum of piecewise spline functions may be obtained. LR B-splines consist of smooth, piecewise polynomial basis functions that constitute a partition of unity. Among other advanced features they may facilitate local h-refinement. Since this class of splines is rich and versatile, it may break new ground and seems to be attractive as foundation for integrating CAD and FEA on one computational platform.

Our long term vision is to create a radically new computational platform with powerful and versatile refinement and adaptation procedures based on the concept of LR B-splines. Downward compatibility to existing NURBS-based models and the synergy of CAD and FEA expertise in each development stage will be essential and, at the same time, promote the broad acceptance and dissemination in both academia and the software industry.

In any finite element analysis of real world problems, it is of great importance that the quality of the computed solution may be determined. However, the assessment of the quality of a computed solution is challenging, both mathematically and computationally. Thus traditionally, the quality of the solution is assessed manually by the scientist or engineer doing the simulation, but this is unreliable. Numerical simulation of many industrial problems in civil, mechanical and naval industry often require large computational resources. It is therefore of utmost importance that computational resources are used as efficiently as possible to make new results readily available and to expand the realm of which processes may be simulated. We thus identify reliability and efficiency as two challenges in simulation based engineering.

These two challenges may be addressed by error estimation combined with adaptive refinements. A lot of research has been performed on error estimation and adaptive mesh refinement, see e.g. (Ainsworth and Oden, 2000 [1]). However, adaptive methods are not yet an industrial tool, partly because the need for a link to traditional CAD-system makes this difficult in industrial practice. Here, the use of an isogeometric analysis framework may facilitate more widespread adoption of this technology in industry, as adaptive mesh refinement does not require any further communication with the CAD system.

1.2 Aim and outline of the paper

The aim of this paper is to present local refinement strategies using LR B-splines and investigate its performance in adaptive isogeometric analysis by means of showing numerical results on well known benchmark examples.

The paper is organized as follows:

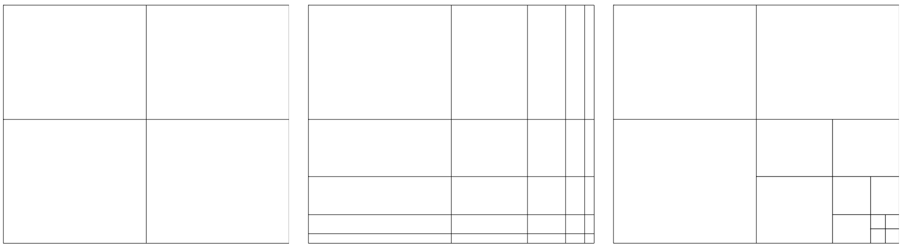
In Section 2, we stated the preliminaries definitions of *B-splines* and *meshes* to illustrate the local refinement of B-spline using knot insertion. Then the basic important ingredients to understand *LR B-splines* concept such as *LR-mesh*, *LR B-spline space*, and *meshline extension* are given. Our aim here is to fix the notations, for a detailed mathematical description related to LR B-splines we refer the reader to Dokken *et al.* [6].

In Section 3, we give a brief introduction to the finite element method and the need for adaptive refinement in real world problems. The main characteristics of isogeometric finite element methods using B-splines (or NURBS) and LR B-splines is presented. Further we describe a general approach, that suits LR B-splines, to perform local h-refinement in adaptive isogeometric finite element method.

Section 4 is devoted to illustrate the local refinement strategies using LR B-splines. A more general discussion on different options for local refinement is given. Then we presented three specific local refinement strategies which we shall investigate in the numerical examples section. At the last the conceptual similarities between adaptive refinement in classical FEM *versus* isogeometric methods using LR B-splines (for $p = 1$ and 2) are given.

Numerical experiments are performed in Section 5. The aim of this section is to illustrate the performance of the local refinement strategies of Section 4. In particular, we investigate whether adaptive refinement using LR B-splines achieves optimal convergence rate, in terms of better accuracy per degrees of freedom (dofs) compared to the uniform refinement case, for non-smooth elliptic problems. For the purpose we consider one synthetic case of refinement along the diagonal and elliptic PDEs with known solutions.

We end this paper by giving some conclusion upon our findings in Section 6.



(a) Initial mesh (b) Tensor product refinement (c) Truly local refinement

Figure 1: Lack of local refinement of tensor B-splines.

2 Spline theory

The problem with traditional B-splines and NURBS is that they are formulated as tensor products of univariate B-splines. This means that refinement in one of the univariate B-splines will cause the insertion of an entire new row or column of knots in the bivariate spline space. As an example of refinement around a local point is achieved which also refine the other area of mesh. This is illustrated in Figure 1, where we have recursively refined the lower right corner. Ideally we do not want to insert any knot in the upper right and lower left part of the mesh, but with B-splines and NURBS, this is unavoidable. Thus to achieve truly local refinement we need some new structure to the mesh which is not based on global tensor products. This is what T-splines, Hierarchical B-splines, and LR B-splines address. T-splines were first introduced by Sederberg *et al.* [20] and have, like NURBS, primarily been used in computer aided design (CAD). In recent years T-splines have, however, been introduced to isogeometric analysis [3, 7, 18, 19].

In the following subsections we will present the LR B-splines, first we establishing a vocabulary that contains several definitions in section 2.1 and then we discuss the algorithms in section 2.2 followed by some properties of LR splines in Section 2.3

2.1 LR-splines

We start the introduction by describing the local knot vectors. From elementary B-spline theory we know that a knot vector of size $n+p+1$ will generate n linearly independent basis functions of degree p. Usually this knot vector is required to start and end with a knot of multiplicity $p+1$, ensuring at least $p+1$ basis functions to be generated. If we ignore this restriction, it is clear that we can generate a single basis function using a knot vector of size $p+2$. The purpose of open knot vectors (knot vectors

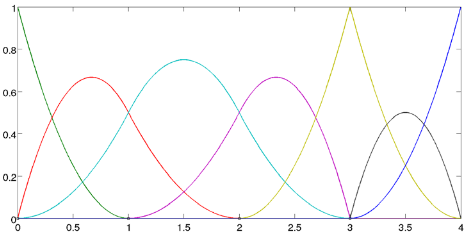


Figure 2: All quadratic basis functions generated by the knot $\Xi = [0,0,0,1,2,3,3,4,4,4]$. Each individual basis function can be described using a local knot vector of 4 knots each $(p+2)$.

with knots of multiplicity $p+1$ at the start and end) is only to ensure interpolating end points which is advantageous in a number of ways, for instance to simplify the handling of Dirichlet boundary conditions. From the evaluation algorithms of B-splines, it follows that every single basis function will depend on not more than $p+2$ knots, each basis function using different knots. For instance, consider a set of quadratic basis function from the knot vector Ξ. We then have

$$\begin{array}{rcl}
\Xi &=& [0,0,0,1,2,3,3,4,4,4] \\
\Xi_1 &=& [0,0,0,1\;] \\
\Xi_2 &=& [0,0,1,2\;] \\
\Xi_3 &=& [0,1,2,3\;] \\
\Xi_4 &=& [1,2,3,3\;] \\
\Xi_5 &=& [2,3,3,4\;] \\
\Xi_6 &=& [3,3,4,4\;] \\
\Xi_7 &=& [3,4,4,4],
\end{array} \tag{1}$$

where the seven basis functions will be separately generated by the *local knot vectors* $\Xi_1, ..., \Xi_7$. One might add here that we will not need the entire set of basis functions, and remove a subset of these, keeping only the ones we are interested in. Even though it might be instructive to look at local basis functions as a subsequence of a global knot vector, this is of little practical value. Instead we will not require any global knot vector Ξ, but rather create the local knot vectors Ξ_i in a different manner. The concept local knot vectors is important for LR B-splines as they are used as the building blocks. We have illustrated the basis functions given by Equation (1) in Figure 2. Using local knot vectors, we define a single B-spline function as

Definition 1. A **B-spline** $B(\boldsymbol{\xi})$ of degrees $\boldsymbol{p}$ is a separable function B :

$\mathbb{R}^n \to \mathbb{R}$

$$B_{\boldsymbol{\Xi}}(\boldsymbol{\xi}) = \prod_{i=1}^{n} B_{\Xi^i}(\xi^i) \tag{2}$$

defined by the n nondecreasing local knot vectors $\Xi^i \in \mathbb{R}^{p_i+2}$ and the degrees p_i, where each $B_{\Xi^i}(\xi^i)$ are univariate B-spline functions of degree p_i over the knot vector Ξ^i.

Note that the degree is implicitly defined by the number of knots in each local knot vector.

Definition 2. The **parametric coordinate space** of dimension 1 2 and 3 is denoted using the greek letters ξ, η and ζ and is related in (2) as

$$(\xi^1, \xi^2, \xi^3) = (\xi, \eta, \zeta) \tag{3}$$

with the corresponding knot vectors begin denoted as $\Xi, \mathcal{H}, \mathcal{Z}$ such that

$$(\Xi^1, \Xi^2, \Xi^3) = (\Xi, \mathcal{H}, \mathcal{Z}). \tag{4}$$

For any B-spline in higher dimension than 3 it is custom to use index notation. The univariate, bivariate and trivariate cases are as following

$$\begin{array}{rlll} B_{\boldsymbol{\Xi}}(\xi^1) & = B_{\boldsymbol{\Xi}}(\xi) & = B_{\Xi^1}(\xi) & = B_{\Xi}(\xi) \\ B_{\boldsymbol{\Xi}}(\xi^1, \xi^2) & = B_{\boldsymbol{\Xi}}(\xi, \eta) & = B_{\Xi^1}(\xi) B_{\Xi^2}(\eta) & = B_{\Xi}(\xi) B_{\mathcal{H}}(\eta) \\ B_{\boldsymbol{\Xi}}(\xi^1, \xi^2, \xi^3) & = B_{\boldsymbol{\Xi}}(\xi, \eta, \zeta) & = B_{\Xi^1}(\xi) B_{\Xi^2}(\eta) B_{\Xi^3}(\zeta) & = B_{\Xi}(\xi) B_{\mathcal{H}}(\eta) B_{\mathcal{Z}}(\zeta). \end{array}$$

We will in the remainder of the text regard bivariate B-splines unless otherwise stated and use the short hand notation

$$B[\xi_0 \xi_1 ... \xi_{p+1}; \eta_0 \eta_1 ... \eta_{p+1}] := B_{\Xi}(\xi) B_{\mathcal{H}}(\eta), \tag{5}$$

where the local knot vectors are known (integers), i.e. $B[0123; 00145]$ for $\Xi^1 = [0, 1, 2, 3], \Xi^2 = [0, 0, 1, 4, 5]$. This particular B-spline would be of polynomial degree $p_1 = 2$ and $p_2 = 3$ due to the number of elements in the local knot vectors.

Also note that we are distinguishing between subscripts and superscripts on the local knot vectors as the former refers to the index in a *set* of B-splines while the latter is the parametric dimension. Consider the set of biquadratic B-splines

$$\{B[0123; 0012], B[2345; 2245], B[1255; 0112]\} = \{B_{\boldsymbol{\Xi}_1}, B_{\boldsymbol{\Xi}_2}, B_{\boldsymbol{\Xi}_3}\},$$

where

$$\begin{array}{rcl} B_{\boldsymbol{\Xi}_1}(\xi^1, \xi^2) & = & B_{\Xi_1^1}(\xi^1) B_{\Xi_1^2}(\xi^2) \\ B_{\boldsymbol{\Xi}_2}(\xi^1, \xi^2) & = & B_{\Xi_2^1}(\xi^1) B_{\Xi_2^2}(\xi^2) \\ B_{\boldsymbol{\Xi}_3}(\xi^1, \xi^2) & = & B_{\Xi_3^1}(\xi^1) B_{\Xi_3^2}(\xi^2) \end{array}$$

and

$$\begin{array}{rclrcl} \Xi_1^1 & = & [0,1,2,3] & \Xi_1^2 & = & [0,0,1,2] \\ \Xi_2^1 & = & [2,3,4,5] & \Xi_2^2 & = & [2,2,4,5] \\ \Xi_3^1 & = & [1,2,5,5] & \Xi_3^2 & = & [0,1,1,2]. \end{array}$$

Definition 3. A **weighted B-spline** is defined as

$$B_{\mathbf{\Xi}}^{\gamma}(\boldsymbol{\xi}) = \gamma \prod_{i=1}^{n} B_{\Xi^i}(\xi^i),$$

where $\gamma \in (0,1]$.

The weighted B-spline is simply a B-spline multiplied by a scalar weight γ. This is to ensure that LR B-splines maintain the partition of unity property, and should not be confused with the rational weights w which is common in NURBS (non-uniform rational B-splines). For simplicity, we will denote both weighted and non-weighted B-splines as B and assume that it is clear from the context if it is one or the other.

Definition 4. A **Box Mesh** or T-mesh is a partitioning of a two-dimensional rectangular domain $[\xi_0, \xi_n] \times [\eta_0, \eta_n]$ into smaller rectangles by horizontal and vertical lines.

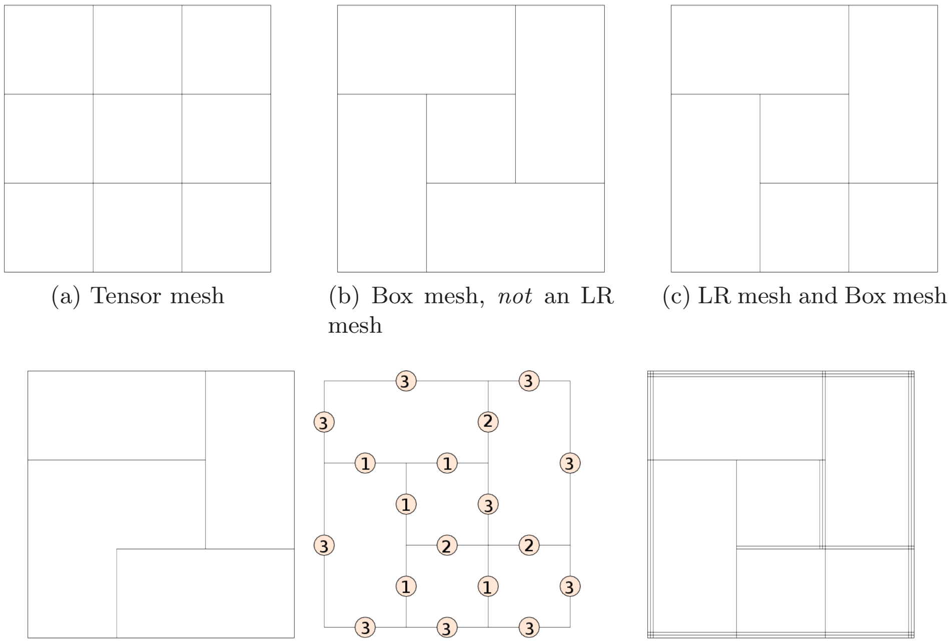
Definition 5. A **Tensor Mesh** is a Box Mesh where there are no T-joints, i.e. all horizontal and vertical lines span the entire length $[\xi_0, \xi_n]$ or $[\eta_0, \eta_n]$.

Definition 6. An **LR-Mesh** $\mathcal{M}_n$ is a Box Mesh which is the result from a series of single line insertions $\{\varepsilon_i\}_{i=1}^n$ from a initial tensor mesh $\mathcal{M}_0$, i.e. $\mathcal{M}_n \supset \mathcal{M}_{n-1} \supset ... \supset \mathcal{M}_1 \supset \mathcal{M}_0$ and each intermediate state $\mathcal{M}_{i+1} = \{\mathcal{M}_i \cup \varepsilon_i\}$ is a also a Box Mesh.

In other words, it must be possible to create the mesh by inserting one line at a time, where these lines never stop in the center of an element (knot span). See Figure 3 for examples of the different meshes.

Definition 7. A Box Mesh, Tensor Mesh or LR-Mesh **with multiplicities** is a Mesh where each line segment has a corresponding integer value n, called the line multiplicity. Each multiplicity must satisfy $0 < n \leq p$, where p is the polynomial degree (in ξ-direction for vertical lines and in η-direction for horizontal lines).

Note that it is possible to create a C^{-1}-basis if using knot lines of multiplicity $n = p$.



(a) Tensor mesh

(b) Box mesh, *not* an LR mesh

(c) LR mesh and Box mesh

(d) Not an LR-mesh, nor a box mesh

(e) LR mesh with multiplicities

(f) Alternative way of drawing (e)

Figure 3: Note that there is no way to create the box mesh (b) from single line insertions (starting at tensor mesh) where every intermediate state is also a box mesh. This is a prerequisite for all LR meshes.

Definition 8. The **support** of a (weighted) B-spline $B : \mathbb{R}^2 \to \mathbb{R}$

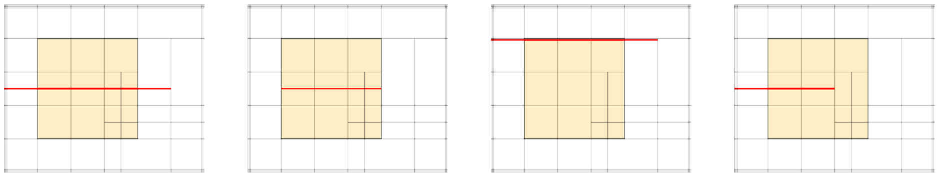
$$\begin{aligned} B(\xi, \eta) &= \gamma B_{\Xi}(\xi) B_{\mathcal{H}}(\eta) \\ \Xi &= [\xi_0, \xi_1, ..., \xi_{p_1+1}] \\ \mathcal{H} &= [\eta_0, \eta_1, ..., \eta_{p_2+1}] \end{aligned} \tag{6}$$

is the closure of all points where it takes nonzero value, i.e. $(\xi, \eta) \in [\xi_0, \xi_{p_1+1}] \times [\eta_0, \eta_{p_2+1}]$.

Definition 9. A meshline ε is said to **traverse** the support of a (weighted) B-spline $B : \mathbb{R}^2 \to \mathbb{R}$ (see (6)) if

- a horizontal line $\varepsilon = [\xi_0^*, \xi_1^*] \times \eta^*$ satisfies

$$\xi_0^* \leq \xi_0, \quad \xi_{p_1+1} \leq \xi_1^*$$
$$\eta_0 \leq \eta^* \leq \eta_{p_2+1},$$



(a) Line traversing the interior of B

(b) Line traversing the interior of B

(c) Line traversing the edge of B

(d) Line neither traversing the interior nor the edge of B

Figure 4: Traversing the support of a basis function. Note that we distinguish between traversing the edge and the interior of the support of B.

- a vertical line $\varepsilon = \xi^* \times [\eta_0^*, \eta_1^*]$ satisfies

$$\xi_0 \leq \xi^* \leq \xi_{p_1+1}$$
$$\eta_0^* \leq \eta_0, \quad \eta_{p_2+1} \leq \eta_1^*.$$

A horizontal line is said to traverse **the interior** if $\eta_0 < \eta^* < \eta_{p_2+1}$ and traverse **the edge** if $\eta_0 = \eta^*$ or $\eta_{p_2+1} = \eta^*$. Similarly for vertical lines it is said to traverse the interior if $\xi_0 < \xi^* < \xi_{p_1+1}$ and traverse the edge if $\xi_0 = \xi^*$ or $\xi_{p_1+1} = \xi^*$.

See Figure 4 for examples on traversing meshlines.

Definition 10. A (weighted) B-spline $B : \mathbb{R}^2 \to \mathbb{R}$ (see (6)) has **minimal support** on a LR Mesh $\mathcal{M}$ if

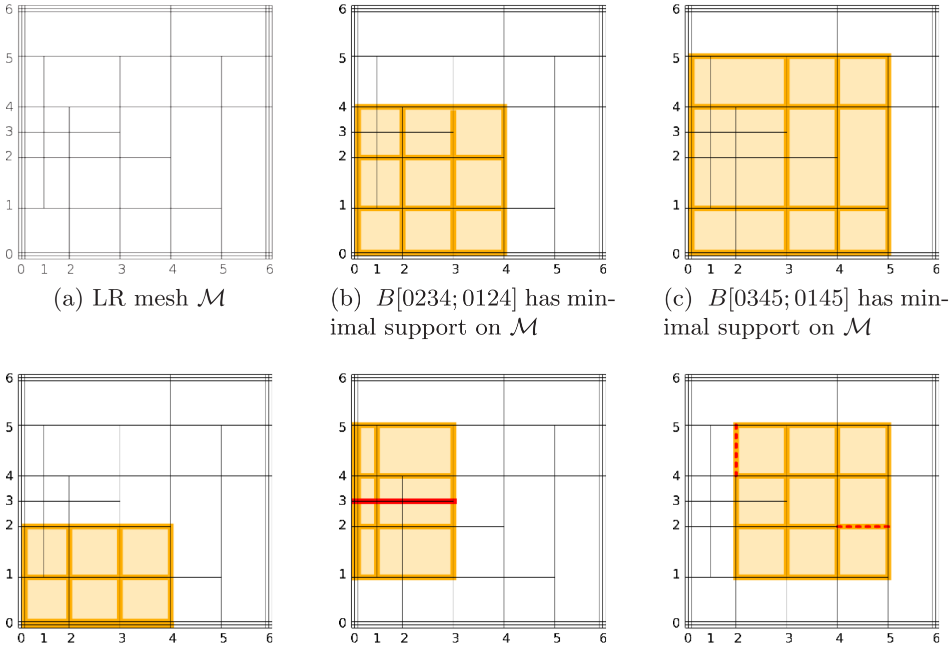
1. for every horizontal line $\varepsilon = [\xi_0^*, \xi_1^*] \times \eta^*$ of multiplicity n in the mesh $\mathcal{M}$ that traverses the support of B, there exist
$$\begin{cases} n \text{ unique } i \text{ such that } \eta_i = \eta^* & \text{, if } \varepsilon \text{ traverses the interior of } B \\ \text{an } i \text{ such that } \eta_i = \eta^* & \text{, if } \varepsilon \text{ traverses the edge of } B \end{cases}$$

2. for every vertical line $\varepsilon = \xi^* \times [\eta_0^*, \eta_1^*]$ of multiplicity n in the mesh $\mathcal{M}$ that traverses the support of B, there exist
$$\begin{cases} n \text{ unique } i \text{ such that } \xi_i = \xi^* & \text{, if } \varepsilon \text{ traverses the interior of } B \\ \text{an } i \text{ such that } \xi_i = \xi^* & \text{, if } \varepsilon \text{ traverses the edge of } B \end{cases}$$

See Figure 5 for examples on minimal support.

Definition 11. Let $\mathcal{M}$ be an LR-mesh with multiplicities. A function $B : \mathbb{R}^2 \to \mathbb{R}$ is called an **LR B-spline** on $\mathcal{M}$ if

1. $B^{\gamma}_{\mathbf{\Xi}}(\boldsymbol{\xi}) = \gamma B_{\Xi}(\xi) B_{\mathcal{H}}(\eta)$ is a weighted B-spline where all knot lines (and the knot line multiplicities) in Ξ and $\mathcal{H}$ is also in $\mathcal{M}$.



(a) LR mesh $\mathcal{M}$

(b) $B[0234; 0124]$ has minimal support on $\mathcal{M}$

(c) $B[0345; 0145]$ has minimal support on $\mathcal{M}$

(d) $B[0234; 0012]$ has minimal support on $\mathcal{M}$

(e) $B[0013; 1245]$ has **not** minimal support on $\mathcal{M}$ due to the meshline at $\eta = 3$

(f) $B[2345; 1245]$ has minimal support on $\mathcal{M}$, but is not an LR B-spline on $\mathcal{M}$ as the two highlighted lines are missing from $\mathcal{M}$

Figure 5: Minimal support ensures that every meshline traversing the support of a B-spline should appear in the local knot vector. Being an LR B-spline ensures the converse: that every line in the knot vector appears in the mesh $\mathcal{M}$

2. B has minimal support on $\mathcal{M}$.

Definition 12. A **meshline extension** ε on an LR mesh $\mathcal{M}_n$ is either

- a new meshline,
- an elongation of an existing meshline,
- a joining of two existing meshlines or
- increasing the multiplicity of an existing line

which causes one or more of the LR B-splines on $\mathcal{M}_n$ to not have minimal support on $\mathcal{M}_{n+1}$.

2.2 Refining LR B-splines

For local refinement, we again turn to existing spline theory. Tensor product B-splines form a subset of the LR B-splines and they obey some of the same core refinement ideas (globally not locally). From tensor product B-spline theory we know that one might insert extra knots to enrich the basis without changing the geometric description. This comes from the fact that we have available the relation between B-splines in the old coarse spline space and in the new enriched spline space. For instance if we want to insert the knot $\hat{\xi}$ into the knot vector Ξ between the knots ξ_{i-1} and ξ_i, then the relation is given by

$$B_\Xi(\xi) = \alpha_1 B_{\Xi_1}(\xi) + \alpha_2 B_{\Xi_2}(\xi), \tag{7}$$

where

$$\begin{aligned}
\alpha_1 &= \begin{cases} 1, & \xi_{p+1} \leq \hat{\xi} \leq \xi_{p+2} \\ \frac{\hat{\xi}-\xi_1}{\xi_{p+1}-\xi_1}, & \xi_1 \leq \hat{\xi} \leq \xi_{p+1} \end{cases} \\
\alpha_2 &= \begin{cases} \frac{\xi_{p+2}-\hat{\xi}}{\xi_{p+2}-\xi_2}, & \xi_2 \leq \hat{\xi} \leq \xi_{p+2} \\ 1, & \xi_1 \leq \hat{\xi} \leq \xi_2 \end{cases}
\end{aligned} \tag{8}$$

and the knot vectors are

$$\begin{aligned}
\Xi &= [\xi_1, \xi_2, \ldots \xi_{i-1}, \quad\; \xi_i, \ldots \xi_{p+1}, \xi_{p+2}] \\
\Xi_1 &= [\xi_1, \xi_2, \ldots \xi_{i-1}, \boldsymbol{\hat{\xi}}, \xi_i, \ldots \xi_{p+1} \qquad] \\
\Xi_2 &= [\qquad \xi_2, \ldots \xi_{i-1}, \boldsymbol{\hat{\xi}}, \xi_i, \ldots \xi_{p+1}, \xi_{p+2}].
\end{aligned}$$

Note that the insertion of the knot $\hat{\xi}$ into Ξ yields a knot vector of size $p + 3$, meaning that it is generating two B-splines. These two B-splines are the one being described by the local knot vectors Ξ_1 and Ξ_2, both of size $p + 2$.

Let us look at an example using this technique. Say we want to insert $\hat{\xi} = \frac{3}{2}$ into the B-spline $\Xi_3 = [0, 1, 2, 3]$. This would give us $\alpha_1 = \alpha_2 = \frac{3}{4}$ and the three functions are plotted in Figure 6. If one were to insert the knot $\hat{\xi} = \frac{3}{2}$ into the *set* of B-splines in Figure 2, then this will require two more functions to be split, namely the function $\Xi_2 = [0, 0, 1, 2]$ and $\Xi_4 = [1, 2, 3, 3]$. All the three splitting shown in Figure 6–7 will then take place. This insertion will replace three old B-splines with four new linearly independent B-splines (see the knot vectors in the figure legend to identify the four distinctive new B-splines).

Bivariate functions are refined in one parametric direction at a time. By pairing two local knot vectors, one for each of the parametric directions we are able to create a bivariate B-spline. For instance if we have the knot vector Ξ in the first parametric direction, and $\mathcal{H}$ in the second, we will have the B-spline $B_{\Xi,\mathcal{H}}(\xi, \eta) = B_\Xi(\xi) B_\mathcal{H}(\eta)$.

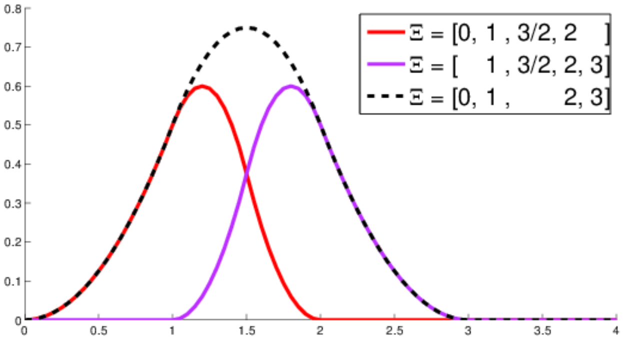
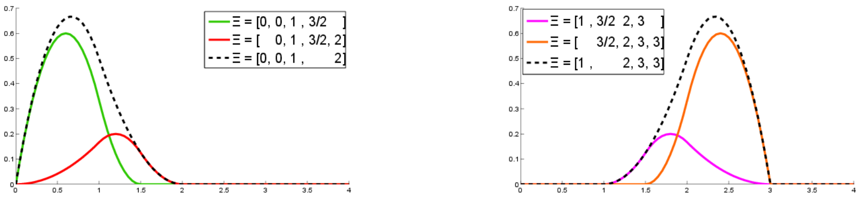


Figure 6: Splitting the B-spline $\Xi = [0, 1, 2, 3]$ into two separate B-splines by inserting the knot $\frac{3}{2}$.



(a) Inserting $\xi = \frac{3}{2}$ in $\Xi = (0, 0, 1, 2)$.

(b) Inserting $\xi = \frac{3}{2}$ in $\Xi = (1, 2, 3, 3)$.

Figure 7: Displaying function splitting in the case that $\hat{\xi}$ is not at the knotvector center.

By using the splitting algorithm in Equation (7) for the 2D case when splitting in one direction, we obtain:

$$\begin{aligned} B_{\boldsymbol{\Xi}}(\xi, \eta) &= B_{\Xi}(\xi) B_{\mathcal{H}}(\eta) \\ &= \left(\alpha_1 B_{\Xi_1}(\xi) + \alpha_2 B_{\Xi_2}(\xi)\right) B_{\mathcal{H}}(\eta) \\ &= \alpha_1 B_{\boldsymbol{\Xi}_1}(\xi, \eta) + \alpha_2 B_{\boldsymbol{\Xi}_2}(\xi, \eta). \end{aligned} \tag{9}$$

For weighted B-splines, this becomes

$$\begin{aligned} B^{\gamma}_{\boldsymbol{\Xi}}(\xi, \eta) &= \gamma B_{\boldsymbol{\Xi}}(\xi, \eta) \\ &= \gamma \left(\alpha_1 B_{\Xi_1}(\xi) + \alpha_2 B_{\Xi_2}(\xi)\right) B_{\mathcal{H}}(\eta) \\ &= B^{\gamma_1}_{\boldsymbol{\Xi}_1}(\xi, \eta) + B^{\gamma_2}_{\boldsymbol{\Xi}_2}(\xi, \eta), \end{aligned}$$

where

$$\begin{aligned} \gamma_1 &= \alpha_1 \gamma \\ \gamma_2 &= \alpha_2 \gamma. \end{aligned}$$

We now have everything we need to formulate the refinement rules. This will be implemented by keeping track of the mesh $\mathcal{M}_n$ and the spline space $\mathcal{S}_n$. Note that we do not need to keep track of the refinement history $\mathcal{M}_i, \quad i = 1...n-1$, we only need to store the current state. For each B-spline $B^{\gamma_i}_{\boldsymbol{\Xi}_i}$ we store the following:

- $\Xi_i \in \mathbb{R}^{p+2}$ - the local knot vector in the first parametric direction
- $\mathcal{H}_i \in \mathbb{R}^{p+2}$ - the local knot vector in the second parametric direction
- $\gamma_i \in \mathbb{R}$ - the scaling weight
- $\boldsymbol{c}_i \in \mathbb{R}^d$ - control points in d-dimensional space.

Through the refinement we aim at two points: keeping the partition of unity and leaving the geometric mapping unchanged, i.e. $\sum_i \gamma_i B_i = 1$ and $\boldsymbol{f} = \sum_i \gamma_i B_i \boldsymbol{c}_i$ for all levels of refinement.

Assuming a meshline extension is inserted, the refinement process is characterized by two steps.

- **Step 1:** Split any B-spline which support is traversed by the *new* meshline - update the weights and control points
- **Step 2:** For all new B-splines, check if their support is completely traversed by any *existing* meshline

We will here describe these two steps in detail.

When to split a B-spline

A B-spline may need to be split at either of the two refinement steps. In Step 1 we test *every* B-spline against *one* meshline. In Step 2 we test every *newly created* B-spline against *all* existing meshlines. This is just a conceptual understanding of the process. In a computational realization of this technique, both of these searches could be done locally.

A B-spline is split whenever a meshline is traversing the interior of that B-spline (see Definition 9).

In the refinement process we will at multiple stages perform checks to see if one particular function is split by one particular meshline. The algorithm is in essence testing B-splines against meshlines, one for one, and splitting every function that satisfies the splitting criterion. The rest is just formulating which B-splines are going to be checked against which meshlines. Note that in the case of a meshline extension being an elongation of an existing meshline or a joining of two existing ones, then we will use the full length of the meshline to flag B-splines for splitting. Thus, in the case of an elongation, we will be using the union of the old line with the elongation and use this combined length when testing if lines are traversing B-splines.

How to split a B-spline

The splitting itself is done through the use of Equation (8) and (9). Let us assume that the function B_i will be split and the result is the functions B_1 and B_2 with corresponding α_1 and α_2. We will now have to make sure that we keep the geometric mapping unchanged and preserves the partition of unity. There are two cases which can arise:

- The new function (B_1 or B_2) already exist in our spline space (due to previous splitting).
- The new function is not present and must be added.

In the latter case of the function not already existing, we will need to create it. We simply add it to our list of B-spline and give it weight and control point equal to it's parent function, i.e. $\gamma_1 := \alpha_1\gamma_i$ and $\boldsymbol{c}_1 := \boldsymbol{c}_i$. We then proceed to add it to the list of newly created functions which will be subsequently tested for splitting in Step 2. In the former case of the function already being present, we simply update the weight and control point and continue with the refinement process. In this case, the control point will be given as $\boldsymbol{c}_1 := (\boldsymbol{c}_1\gamma_1 + \boldsymbol{c}_i\gamma_i\alpha_1)/(\gamma_1 + \gamma_i\alpha_1)$ and the weight will be given by $\gamma_1 := \gamma_1 + \gamma_i\alpha_1$. Finally, we remove the old function from our list of B-splines. This is illustrated in Algorithm 2 where we have assumed that the inserted knot is in the Ξ-vector (the $\mathcal{H}$-case being completely analogue). Note that we are keeping the unrefined knot vector $\mathcal{H}_i$ unchanged in line 6-7, as it is apparent in Equation (9). We are also storing all newly created B-splines in $\mathcal{S}_{new}$ as these will be required in Step 2 of the refinement algorithm.

LR spline definition

We define an LR spline as an application of the refinement algorithm.

Definition 13. An **LR spline** $\mathcal{L}$ is a pair $(\mathcal{M}_n, \mathcal{S})$, where $\mathcal{M}_n$ is an LR mesh and $\mathcal{S}$ is a set of LR B-splines on $\mathcal{M}_n$, and

- for each intermediate step $\mathcal{M}_{i+1} = \{\mathcal{M}_i \cup \varepsilon_i\}$ the new line ε_i is a meshline extension
- $\mathcal{S} = \left\{B_{\boldsymbol{\Xi}_i}(\boldsymbol{\xi})\right\}_{i=1}^{m}$ is the set of all LR B-splines on $\mathcal{M}_n$ resulting from Algorithm 3.

We note that, there is no backwards dependence on the mesh, meaning that while the index in $\mathcal{M}_n$ seems to suggest that the LR spline is a sequence of meshes, it is enough that there exist one possible sequence. After we have constructed the set of LR B-splines on $\mathcal{M}_n$, it is safe to discard any link to the previous mesh $\mathcal{M}_{n-1}$.

Algorithm 2 Local ξ-split

```
parameters: $\hat{\xi}$                 {new knot}
            $B_i$                       {B-spline to be split ($B_i \in \mathcal{S}$)}
            $\mathcal{S}$               {Spline space}
            $\mathcal{S}_{new}$         {Functions not present in $\mathcal{S}$}
calculate $(\alpha_1, \alpha_2)$ from (8)
$\Xi \leftarrow \mathrm{SORT}(\Xi \cup \hat{\xi})$
$\Xi_1 \leftarrow [\xi_1, ..., \xi_{p+2}]$
$\Xi_2 \leftarrow [\xi_2, ..., \xi_{p+3}]$
$\mathcal{H}_1 \leftarrow \mathcal{H}_i$
$\mathcal{H}_2 \leftarrow \mathcal{H}_i$
if $(\Xi_1, \mathcal{H}_1) \in \mathcal{S}$ then
    $\boldsymbol{c}_1 \leftarrow (\boldsymbol{c}_1\gamma_1 + \boldsymbol{c}_i\gamma_i\alpha_1) / (\gamma_1 + \alpha_1\gamma_i)$
    $\gamma_1 \leftarrow \gamma_1 + \alpha_1\gamma_i$
else
    $\boldsymbol{c}_1 \leftarrow \boldsymbol{c}_i$
    $\gamma_1 \leftarrow \alpha_1\gamma_i$
    add $B_1$ to $\mathcal{S}_{new}$
end if
if $(\Xi_2, \mathcal{H}_2) \in \mathcal{S}$ then
    $\boldsymbol{c}_2 \leftarrow (\boldsymbol{c}_2\gamma_2 + \boldsymbol{c}_i\gamma_i\alpha_2) / (\gamma_2 + \alpha_2\gamma_i)$
    $\gamma_2 \leftarrow \gamma_2 + \alpha_2\gamma_i$
else
    $\boldsymbol{c}_2 \leftarrow \boldsymbol{c}_i$
    $\gamma_2 \leftarrow \alpha_2\gamma_i$
    add $B_2$ to $\mathcal{S}_{new}$
end if
remove $B_i$ from $\mathcal{S}$
```

Algorithm 3 LR B-spline refinement

```
                 $\mathcal{S}$   {Spline space}
parameters:      $\mathcal{M}$   {LR mesh}
                 $\mathcal{E}$   {Meshline extension}
for every B-spline $B_i \in \mathcal{S}$ do
  if $\mathcal{E}$ splits $B_i$ then
    perform split according to Algorithm 2
  end if
end for
for every B-spline $B_i \in \mathcal{S}_{new}$ do
  for every existing edge $\mathcal{E}_j \in \mathcal{M}$ do
    if $\mathcal{E}_j$ splits $B_i$ then
      perform split according to Algorithm 2
      {note that this may enlarge $\mathcal{S}_{new}$ further}
    end if
  end for
end for
```

Further, it does not matter if it is possible to make the mesh $\mathcal{M}_n$ in multiple ways. Indeed *any* ordering of the meshline insertions will produce the exact same end function space $\mathcal{S}$. See Section 2.3. As such, for any given LR mesh $\mathcal{M}_n$, the set of LR B-splines $\mathcal{S}$ is unique.

While it is possible to define LR splines by using non-weighted B-splines, as done in [6], we will here only consider weighted ones as to maintain the partition of unity which is important in finite element methods.

Definition 14. The **cardinality** of an LR spline $\mathcal{L} = (\mathcal{M}_n, \mathcal{S})$, where $\mathcal{S} = \left\{ B^{\gamma}_{\mathbf{\Xi}_i}(\boldsymbol{\xi}) \right\}_{i=1}^{m}$ is the number of B-splines in the set $\mathcal{S}$, and is denoted

$$|\mathcal{L}| = m. \tag{10}$$

Example

As an example we look at the insertion of two local knot lines in the tensor product mesh given by the global knot vectors $\Xi = \mathcal{H} = [0, 0, 0, 1, 2, 4, 5, 6, 6, 6]$. We first introduce the line spanning $(\xi, \eta) \in (3, 1) \rightarrow (3, 5)$, see Figure 8. The line will split the three B-splines illustrated in Figure 9. We calculate the corresponding α-values from Equation (8) and get

$$\begin{array}{llll} B[0124;1245]= & B[0123;1245]+\frac{1}{3}B[1234;1245] & & \\ B[1245;1245]= & & \frac{2}{3}B[1234;1245]+\frac{2}{3}B[2345;1245] & \\ B[2456;1245]= & & & \frac{1}{3}B[2345;1245]+B[3456;1245] \end{array} \tag{11}$$

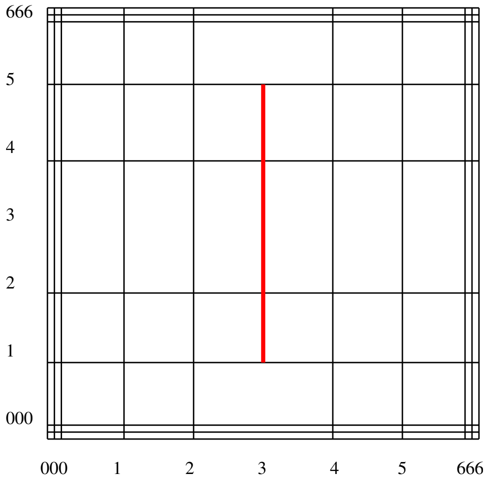
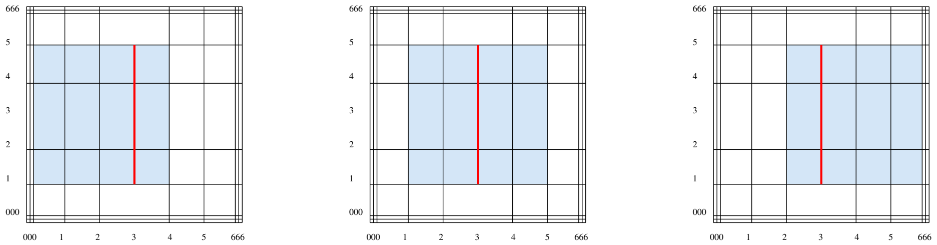


Figure 8: Inserting a local (vertical) meshline into a tensor product mesh.



(a) $B[0124;1245] = B[0123;1245] + \frac{1}{3}B[1234;1245]$

(b) $B[1245;1245] = \frac{2}{3}B[1234;1245] + \frac{2}{3}B[2345;1245]$

(c) $B[2456;1245] = \frac{1}{3}B[2345;1245] + B[3456;1245]$

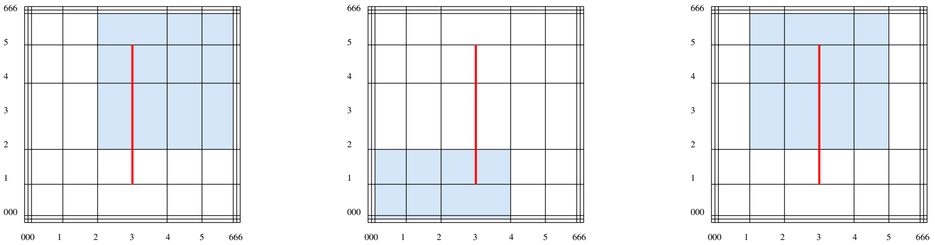
Figure 9: B-splines split by the new meshline.

Updating these splits sequentially, we get the following. Let the numerical indices $i = 1,2,3,4$ denote the new B-splines and alphabetical indices $i = a,b,c$ denote the old basis, i.e.

$$\begin{array}{rcl} B_1 & = & B[0123;1245] \\ B_2 & = & B[1234;1245] \\ B_3 & = & B[2345;1245] \\ B_4 & = & B[3456;1245] \\ B_a & = & B[0124;1245] \\ B_b & = & B[1245;1245] \\ B_c & = & B[2456;1245] \end{array}$$

	Splitting B_a		Splitting B_b		Splitting B_c	
	γ_i	$\boldsymbol{c}_i$	γ_i	$\boldsymbol{c}_i$	γ_i	$\boldsymbol{c}_i$
B_1	1	$\boldsymbol{c}_a$	1	$\boldsymbol{c}_a$	1	$\boldsymbol{c}_a$
B_2	1/3	$\boldsymbol{c}_a$	1	$\frac{1}{3}\boldsymbol{c}_a + \frac{2}{3}\boldsymbol{c}_b$	1	$\frac{1}{3}\boldsymbol{c}_a + \frac{2}{3}\boldsymbol{c}_b$
B_3			2/3	$\boldsymbol{c}_b$	1	$\frac{2}{3}\boldsymbol{c}_b + \frac{1}{3}\boldsymbol{c}_c$
B_4					1	$\boldsymbol{c}_c$

Table 1: Numerical values for weights and control points as Algorithm 3 iterates to insert the meshline in Figure 8.



(a) $B[24565; 2456]$ (b) $B[0124; 0012]$ (c) $B[1245; 2456]$

Figure 10: B-splines *not* split by the new meshline.

Note that at the tensor product case, all weights γ will be equal to one. After the first split of the old function B_a, we establish the new functions B_1 and B_2. Their weights will simply be the α-values 1 and 1/3 and the control points will remain unchanged $\boldsymbol{c}_1 = \boldsymbol{c}_2 = \boldsymbol{c}_a$. Splitting the second function in Equation (11) B_b will cause one of the results B_2 to be already present, so we update the corresponding weights and control point according to line 9 - 10 in Algorithm 2. The process is shown in Table 2 and the numerical values are tabulated sequentially according to whenever each of the B-splines B_a, B_b and B_c are being split.

We would now proceed to Step 2, and test every new B-spline B_1, B_2, B_3 and B_4 against all previously inserted meshlines, but as this is the first inserted line, this is unnecessary. Some of the supports of the unrefined B-splines are depicted in Figure 10.

Next we insert another line, this time spanning $(\xi, \eta) \in (1, 3) \to (5, 3)$ as shown in Figure 11. We first iterate through **Step 1** of the refinement. Here, 4 B-splines will be completely traversed by the new meshline as illustrated in Figure 12. We keep our old convention of numbering the old B-splines by alphabetical letters $i = a, b, c, d$ and the new B-splines by numerical numbers $i = 1, 2, \ldots$.

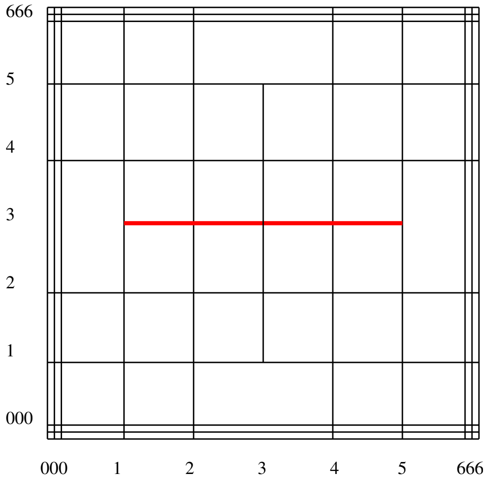
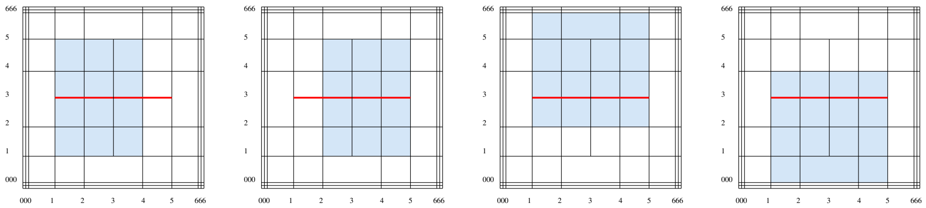


Figure 11: Inserting another local (horizontal) meshline.



(a) Basis B_a split, $B[1234; 1245] = \frac{2}{3}B[1234; 1234] + \frac{2}{3}B[1234; 2345]$

(b) Basis B_b split, $B[2345; 1245] = \frac{1}{3}B[2345; 1234] + B[2345; 2345]$

(c) Basis B_c split, $B[1245; 2456] = B[1245; 2345] + \frac{1}{3}B[1245; 3456]$

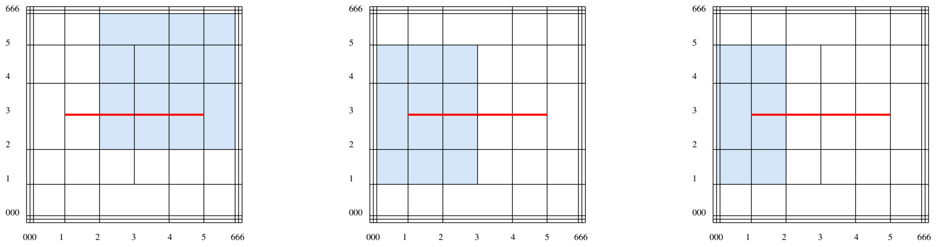
(d) Basis B_d split, $B[1245; 0124] = \frac{1}{3}B[1245; 0123] + B[1245; 1234]$

Figure 12: B-splines split by the new meshline.

$$\begin{aligned}
B[1234; 1245] &= \frac{2}{3}B[1234; 1234] + \frac{2}{3}B[1234; 2345] \\
B[2345; 1245] &= \frac{2}{3}B[2345; 1234] + \frac{2}{3}B[2345; 2345] \\
B[1245; 2456] &= \frac{1}{3}B[1245; 2345] + B[1245; 3456] \\
B[1245; 0124] &= B[1245; 0123] + \frac{1}{3}B[1245; 1234]
\end{aligned}$$

	Step 1		**Step 2**			
	Splitting B_a, B_b, B_c and B_d		Splitting B_5		Splitting B_8	
	γ_i	$\boldsymbol{c}_i$	γ_i	$\boldsymbol{c}_i$	γ_i	$\boldsymbol{c}_i$
B_1	$2/3$	$\boldsymbol{c}_a$	$2/3$	$\boldsymbol{c}_a$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_a + 2\boldsymbol{c}_d)$
B_2	$2/3$	$\boldsymbol{c}_a$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_a + 2\boldsymbol{c}_c)$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_a + 2\boldsymbol{c}_c)$
B_3	$2/3$	$\boldsymbol{c}_b$	$2/3$	$\boldsymbol{c}_b$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_b + 2\boldsymbol{c}_d)$
B_4	$2/3$	$\boldsymbol{c}_b$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_b + 2\boldsymbol{c}_c)$	$8/9$	$\frac{1}{8}(6\boldsymbol{c}_b + 2\boldsymbol{c}_c)$
B_5	$1/3$	$\boldsymbol{c}_c$	<Remove B_5>			
B_6	1	$\boldsymbol{c}_c$	1	$\boldsymbol{c}_c$	1	$\boldsymbol{c}_c$
B_7	1	$\boldsymbol{c}_d$	1	$\boldsymbol{c}_d$	1	$\boldsymbol{c}_d$
B_8	$1/3$	$\boldsymbol{c}_d$	$1/3$	$\boldsymbol{c}_d$	<Remove B_8>	

Table 2: Numerical values of weights and control points as Algorithm 3 iterates to insert the meshline in Figure 11.



(a) $B[2456; 2456]$ (b) $B[0123; 1245]$ (c) $B[0012; 1245]$

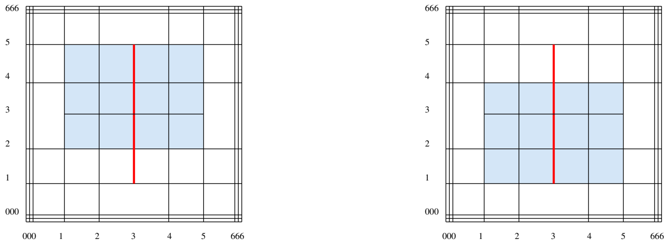
Figure 13: B-splines *not* split by the new meshline.

or more compact

$$
\begin{aligned}
B_a &= \frac{2}{3}B_1 + \frac{2}{3}B_2 \\
B_b &= \frac{2}{3}B_3 + \frac{2}{3}B_4 \\
B_c &= \frac{1}{3}B_5 + B_6 \\
B_d &= B_7 + \frac{1}{3}B_8
\end{aligned}
$$

We note that none of the new B-splines on the right hand side are equal, so all will be considered as new functions with corresponding weights and control points set by line 12 - 13 in Algorithm 2. Again, we show some of the B-splines in the vicinity of the newest meshline that are not splitted, see Figure 13.

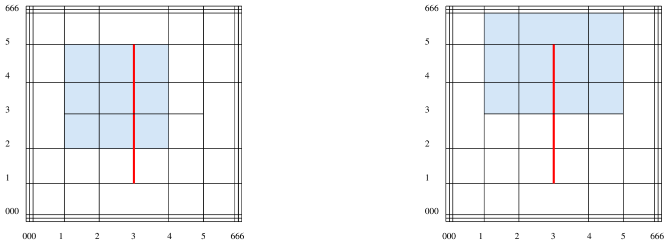
We now proceed to **Step 2** of Algorithm 3. There are 8 new B-splines and all of these would have to be checked against the previous line and see if they



(a) B-spline $B_5 = B[1245; 2345]$ split by the old meshline

(b) B-spline $B_8 = B[1245; 1234]$ split by the old meshline

Figure 14: *New* B-splines split by an *old* meshline in Step 2 of the refinement algorithm.



(a) B-spline $B_2 = B[1234; 2345]$ not split by the old meshline (knotline already present)

(b) B-spline $B_6 = B[1245; 3456]$ not split by the old meshline

Figure 15: New B-splines unchanged by the existing meshline in Step 2 of the refinement algorithm.

are to be further split. As it turns out, two of the new functions are now completely traversed by the previous line since their support have decreased with the knot insertion. These are B_5 and B_8 as depicted in Figure 14

$$\begin{aligned} B[1245; 2345] &= \frac{2}{3}B[1234; 2345] + \frac{2}{3}B[2345; 2345] \\ B[1245; 1234] &= \frac{2}{3}B[1234; 1234] + \frac{2}{2}B[2345; 1234] \end{aligned}$$

or

$$\begin{aligned} B_5 &= \frac{2}{3}B_2 + \frac{2}{3}B_4 \\ B_8 &= \frac{2}{3}B_1 + \frac{2}{2}B_3 \end{aligned}$$

2.3 LR spline properties

Consider a LR spline $(\mathcal{M}_n, \mathcal{S})$. Then

1. $\sum_{i=1}^m \gamma_i B_i(\boldsymbol{\xi}) = 1$, i.e. the LR B-splines form a partition of unity.

2. $(\mathcal{M}_{i+1}, \mathcal{S}_{i+1}) \supset (\mathcal{M}_i, \mathcal{S}_i)$, i.e. the LR spline is nested.

3. If there exists two meshline insertions lists $\{\varepsilon_0, \varepsilon_1, ..., \varepsilon_{n-1}\}$ and $\{\tilde{\varepsilon}_0, ..., \tilde{\varepsilon}_{n-1}\}$ such that $\mathcal{M}_{i+1} = \{\mathcal{M}_i \cup \varepsilon_i\}$, $\tilde{\mathcal{M}}_{i+1} = \{\tilde{\mathcal{M}}_i \cup \tilde{\varepsilon}_i\}$ and the final mesh is equal $\mathcal{M}_n = \tilde{\mathcal{M}}_n$, then the spline space is equal $\mathcal{S}_n = \tilde{\mathcal{S}}_n$, i.e. the LR spline refinement is order independent.

4. $\mathcal{S} = \{B_i(\boldsymbol{\xi})\}_{i=1}^m$ does in general not form a linearly independent set.

Partition of unity

The set of LR B-splines form a partition of unity, i.e.

$$\sum_{i=1}^{m} \gamma_i B_i(\boldsymbol{\xi}) = 1 \tag{12}$$

Proof: Since the refinement consists of repeated use of Algorithm 2, we will only show that the partition of unity is preserved through one step of this algorithm. The global result then follows from induction. Following our convention of enumerating old B-splines alphabetical and new B-splines numerical, let us name the three functions $\{a, 1, 2\}$ such that

$$B_a(\boldsymbol{\xi}) = \alpha_1 B_1(\boldsymbol{\xi}) + \alpha_2 B_2(\boldsymbol{\xi}) \tag{13}$$

There are three outcomes of the algorithm:

- B_1 and B_2 already exist in the spline space

- B_1, but not B_2 already exist in the spline space

- neither B_1 nor B_2 exist in the spline space.

We will here show that this holds for the first case, since the proof for the two other cases are completely analog. Assume that the partition of unity holds before splitting, i.e. (12), then

$$\begin{aligned}
\sum_{\substack{i=1\\ i\neq 1,2,a}}^{m} \gamma_i B_i(\boldsymbol{\xi}) + (\gamma_1 + \alpha_1\gamma_a)B_1(\boldsymbol{\xi}) + (\gamma_2 + \alpha_2\gamma_a)B_2(\boldsymbol{\xi}) &= \\
\sum_{\substack{i=1\\ i\neq 1,2,a}}^{m} \gamma_i B_i(\boldsymbol{\xi}) + \gamma_1 B_1(\boldsymbol{\xi}) + \gamma_2 B_2(\boldsymbol{\xi}) + \gamma_a(\alpha_1 B_1(\boldsymbol{\xi}) + \alpha_2 B_2(\boldsymbol{\xi})) &= \\
\sum_{\substack{i=1\\ i\neq 1,2,a}}^{m} \gamma_i B_i(\boldsymbol{\xi}) + \gamma_1 B_1(\boldsymbol{\xi}) + \gamma_2 B_2(\boldsymbol{\xi}) + \gamma_a B_a(\boldsymbol{\xi}) &= 1
\end{aligned}$$

Nested space

A spline space $\mathcal{S}_{i-1} \subset \mathcal{S}_i$ is said to be nested, if for any $f \in \mathcal{S}_{i-1}$ there exist an $\hat{f} \in \mathcal{S}_i$ such that $f = \hat{f}$. For LR splines the functions f and $\hat{f}$ can be represented by their control points as $f = \sum_{i=1}^{n} B_i c_i$ and $\hat{f} = \sum_{i=1}^{m} B_i \hat{c}_i$. In order to find the relation between an arbitrary $f = [\boldsymbol{c}]$ and $\hat{f} = [\hat{\boldsymbol{c}}]$, we need to find the relationship between their control points defining them.

As the refinement algorithm, is a repeated use of (7), which is a linear relation, we can formulate the relations between the set of old B-splines and the set of enriched B-splines as a matrix $C \in \mathbb{R}^{n\times m}$, satisfying

$$B_{old} = CB_{new}. \tag{14}$$

Hence given any $f = [\boldsymbol{c}]$, we can find $\hat{f} = [\hat{\boldsymbol{c}}]$ by $\hat{\boldsymbol{c}} = C^T\boldsymbol{c}$.

The LR meshes $\mathcal{M}$ are as such nested by construction.

Independence of meshline insertion order

LR splines are independent of the ordering at which the meshline extensions are inserted. That means if $\mathcal{L} = \{\mathcal{M}_n, \mathcal{S}_n\}$, $\hat{\mathcal{L}} = \{\hat{\mathcal{M}}_m, \hat{\mathcal{S}}_m\}$ and the meshes are equal $\mathcal{M}_n = \hat{\mathcal{M}}_m$, then $\mathcal{S}_n = \hat{\mathcal{S}}_m$. For this proof, see Dokken *et al.* [6].

Linear dependence

The LR splines are not linearly independent, in general. As an example for linearly dependent set of LR B-splines, see Figure 16. Here the linear dependence relation is given as:

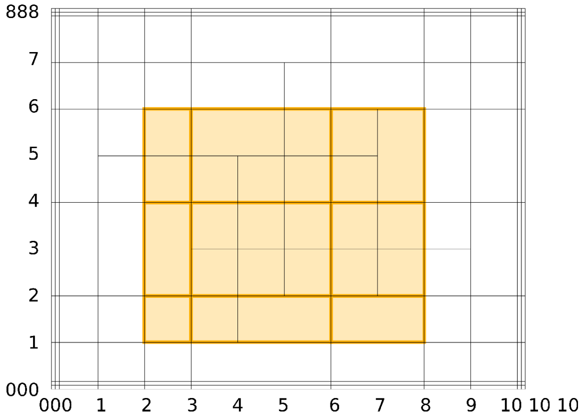


Figure 16: Example of a linearly dependent LR mesh using biquadratic B-splines. The shaded B-spline $B[2356;1246]$ is a linear combination of 7 smaller B-splines; their relation given in (15)

$$\begin{aligned}720 \cdot B[2368;1246] \quad = \quad & 108 \cdot B[5678;2346] + 135 \cdot B[2356;2456] + \\ & 108 \cdot B[3567;3456] + 268 \cdot B[3456;2345] + \\ & 324 \cdot B[4567;2345] + 360 \cdot B[2346;1245] + \\ & 384 \cdot B[3468;1234]. \end{aligned} \tag{15}$$

2.4 Linear independence of LR splines

As shown above, one cannot guarantee that an arbitrary LR-mesh is producing a linearly independent set of functions, however there are several ways to ensure that the system of functions you get is in fact linearly independent. We will here briefly describe three methods, but for full details we reference the work of Dokken et.al [6].

Hand-in-hand principle

Mourrain [15] presented a formula independent of choice of basis for the dimensionality of a spline space over a T-mesh. This result is generalized in [6] to also address general multiplicities and any dimension. Used in the bivariate setting it provides a topological equation based on the polynomial degrees, the elements, the edges and their multiplicities and the vertices. By observing the change of these components, we are able to predict the dimension increase of the spline space for classes of meshline insertions.

Definition 15. A **primitive meshline extension** is a meshline extension which increases the dimension of the spline space by one.

In particular we note that any meshline extension of the following type

- inserting a new meshline spanning p elements,
- elongation of a meshline by one element and
- increasing the multiplicity of a meshline of length p

are primitive.

Proposition 1. *Let $\mathcal{L} = \{\mathcal{M}, \mathcal{S}\}$ be a refinement of $\hat{\mathcal{L}} = \{\hat{\mathcal{M}}, \hat{\mathcal{S}}\}$, where $\mathcal{M} = \{\hat{\mathcal{M}} \cup \varepsilon\}$. If ε is a primitive meshline extension on $\hat{\mathcal{M}}$ and $\hat{\mathcal{S}}$ is linearly independent then $\mathcal{S}$ is linearly independent on $\mathcal{M}$ if and only if*

$$|\mathcal{L}| = \left|\hat{\mathcal{L}}\right| + 1. \tag{16}$$

Proof: We know from the dimension formula formula that the dimension of the function space should be equal to one greater than the dimension of the old space. What is left to show is that the LR Spline space does in fact increase in size, i.e. the new function is not a linear combination of the existing ones. This can be seen from a continuity perspective. Any meshline extension will decrease the continuity of the basis at the point where the mesh is extended and at least one B-spline will in the splitting scheme include this new knot within its support. Thus $\hat{\mathcal{S}} \subset \mathcal{S}$, with strict inclusion and the theorem is proved.

Take note that many meshline extensions can be formulated as a series of primitive meshline extensions. One such example is the insertion of a new meshline of length n which can be formulated as one new meshline of length p and $p - n$ meshline extensions of length one. One can as such carefully go hand-in-hand and ensure that the LR spline at all stages of the refinement is coinciding with the theoretical dimension proved by Mourrain.

Peeling algorithm

Another option is by the peeling algorithm and the notion of local linear independence.

Definition 16. An **element** $\mathcal{R} = (\xi_1, \xi_2) \times (\eta_1, \eta_2)$ in a box mesh is an open set where no horizontal or vertical lines cross.

Note that elements on LR splines means that all B-splines are C^∞ on $\mathcal{R}$ since all reduced continuity appears across meshlines.

Definition 17. An element $\mathcal{R}$ in an LR spline is said to be **locally linearly independent** if there exist no choice of coefficients $\{c_i\}$ such that

$$\sum_{i\in\mathcal{S}_\mathcal{R}} c_iB_i(\boldsymbol{\xi}) = 0, \quad \forall\boldsymbol{\xi}\in\mathcal{R} \tag{17}$$

except for the trivial solution $c_i = 0, \forall i$. Here $\mathcal{S}_\mathcal{R}$ denotes the set of all B-splines with support on the element $\mathcal{R}$.

Since all B-splines are polynomials when restricted to one particular element, it is clear that an element is locally linearly independent if and only if the set $\mathcal{S}_\mathcal{R}$ consists of exactly $(p+1)(q+1)$ B-splines, where p and q is the polynomial degree of the LR spline.

Algorithm 4 Peeling algorithm

```
parameters:  $\mathcal{S}$        {Spline space}
             $\Omega$        {Parametric domain}
             $\mathcal{S}_{LD}$     {possible linearly dependent B-splines}
             $\Omega_{LD}$     {areas of possible linear dependence}
$\mathcal{S}_{LD} \leftarrow \mathcal{S}$
$\Omega_{LD} \leftarrow \Omega$
for every element $\mathcal{R} \in \Omega$ do
  if $\mathcal{R}$ is locally linearly independent then
    $\Omega_{LD} \leftarrow \Omega_{LD} \setminus \mathcal{R}$
    for every B-spline $B_i$ with support on $\mathcal{R}$ do
      $\mathcal{S}_{LD} \leftarrow \mathcal{S}_{LD} \setminus B_i$
    end for
  end if
end for
while $\Omega_{LD}$ or $\mathcal{S}_{LD}$ changed do
  for  all elements $\mathcal{R} \in \Omega_{LD}$  do
    if $\mathcal{R}$ has support of exactly one B-spline $B_i \in \mathcal{S}_{LD}$ then
      $\Omega_{LD} \leftarrow \Omega_{LD} \setminus \mathcal{R}$
      $\mathcal{S}_{LD} \leftarrow \mathcal{S}_{LD} \setminus B_i$
    end if
  end for
end while
```

The Peeling algorithm is given in Algorithm 4. Here, we keep track of two things: The set of B-splines that may appear in a (global) linear dependence relation

$$\sum_i c_iB_i(\boldsymbol{\xi}) = 0 \tag{18}$$

and the set of possible areas where this may occur. Line 2-11 is just initialization where we remove all locally linearly independent elements and the B-splines with support on these. The next lines comes from the realization that (18) may never contain only one term. There have to be at least two B-splines with coefficients $c_i \neq 0$ for a nontrivial relation to exist. We may then in line 14-16 remove this B-spline and the element from possible linear combinations. The removal of B_i in line 16 will in turn decrease the number of B-splines with support on several rectangles which may cause the if-statement in line 14 to trigger more. As such, one may peel away B-spline after B-spline until one of two things happen: There are no B-splines left in the set $\mathcal{S}_{LD}$ and $\mathcal{S}$ is proven linearly independent or all elements $\mathcal{R} \in \mathcal{M}_{LD}$ have support of two or more B-splines. In the latter case there *may* exist a linear combination, and further investigation is required.

The tensor expansion

From any LR spline $\mathcal{L} = \{\mathcal{M}, \mathcal{S}\}$ expand the LR mesh $\mathcal{M}$ to a full tensor mesh. This will create a map from the LR spline basis to the tensor product basis and can be described as (14). Since we know that the tensor product B-splines are linearly independent, the LR spline basis $\mathcal{S}$ will also be linearly independent if and only if the matrix C has full rank [14].

In a computational realization of these methods it is possible to describe the matrix C using rational numbers under the assumption that the initial tensor product mesh $\mathcal{M}_0$ consisted of integer or rational knots. This is due to the fact that all refinements are done by halving each knot interval, and all splitting of B-splines, results in rational expressions seen in (8). It is then possible to compute the rank of the matrix using *exact* arithmetics, and this is what is done in the work within this paper. For a more computational efficient implementation of this method, consider using integers modulus some high Mersenne prime, for instance $p = 2^{31} - 1$. This gives a faster, more robust method at the cost of the very unlikely event that one of the matrix entries by chance becomes a multiple of p and the method produces the wrong result.

Choice of methodology

The tensor expansion for checking for linear independence does not scale to well with increasing problem sizes, but it has the advantage of always working for all meshes and continuities. The hand-in-hand principle has the drawback that it disallows a few refinements and the peeling algorithm, while necessary, it is not sufficient for linearly dependent meshes, meaning that it can prove that an LR spline is linearly independent, but it cannot prove it linearly dependent.

It is of course possible to combine these techniques, where one could for instance narrow the possible areas of linear dependence down to only a subset of the mesh using the peeling algorithm and proceeding with tensor expansions in only these areas for full verification.

For all numerical experiments presented in this paper, we tested for linear independence using the tensor expansion method, and no linearly dependent cases were discovered when using the full span and the structured mesh refinement techniques of Section 4 (below).

In fact we conjecture that by virtue of the particular choice of refinement scheme (full span and structured mesh) you will always be in a subset of the linearly independent LR Splines, similar to the subclass of T-splines which are the analysis suitable T-splines [14].

3 Isogeometric analysis

3.1 The Galerkin finite element method

The variational formulation

Many problems in science and engineering can be addressed by solving a variational problem. Given a Hilbert space $\mathcal{V}$, a continuous, coercive bilinear form $a(\cdot,\cdot)$ and a continuous linear functional $l \in \mathcal{V}^*$, where $\mathcal{V}^*$ is the dual space to $\mathcal{V}$, the variational formulation is defined by: find $u \in \mathcal{V}$ such that such that

$$a(u,v) = l(v) \quad \forall v \in \mathcal{V}. \tag{19}$$

The existence and uniqueness of the solution to this continuous problem is guaranteed by the Lax-Milgram theorem. The Galerkin Finite Element (FE) approximation to this variational problem may then be given as follow: Given a finite subspace $\mathcal{V}_h \subset \mathcal{V}$ and $l \in \mathcal{V}^*$, find $u_h \in \mathcal{V}_h$ such that

$$a(u_h,v_h) = l(v_h) \quad \forall v_h \in \mathcal{V}_h. \tag{20}$$

A priori error estimates

For cases when the bilinear form $a(\cdot,\cdot)$ is selfadjoint the FE-solution u_h is the optimal approximation to the analytical solution u as measured in the "a-norm" (often denoted "energy-norm" symbolized with $_E$):

$$||u - u_h||_E = \sqrt{a(u-u_h, u-u_h)}. \tag{21}$$

If the analytical solution (of a variational problem involving first order differentiation) is sufficiently smooth, i.e. $u \in H^{p+1}$, and the FE mesh $\mathcal{M}_0$ is regular and quasi-uniform, the error in the approximate FE-solution on a family of uniformly refined meshes $\{\mathcal{M}_k\}$, is bounded by

$$||u - u_h||_E \leq Ch^p||u||_{H^{p+1}}, \tag{22}$$

where C is some problem-dependent constant, h is the characteristic size of the finite elements, p denotes the highest degree of a complete polynomial in the FE basis and $||u||_{H^{p+1}}$ denotes the Sobolev norm of order $p+1$.

For problems where the solution is not sufficiently smooth, $u \notin H^{p+1}$, e.g. problems with singularities within the solution domain or on its boundary, we have the error bound

$$||u - u_h||_E \leq Ch^\alpha||u||_{H^{\alpha+1}}, \tag{23}$$

where the value of the non-negative real parameter α depends on how the family of meshes $\{\mathcal{M}_k\}$ are created. Assume that λ is a real number characterizing the strength of the singularity. For a sequence of uniformly, or nearly uniformly, refined meshes we then have

$$\alpha = \min\{p, \lambda\}. \tag{24}$$

Thus, when $\lambda < p$ the rate of convergence is limited by the strength of the singularity, and not to the polynomial order.

Adaptive mesh refinement (AMR)

For classical FEM the main method for obtaining an optimal grid for minimizing the global energy error (a-norm), has been to do adaptive grid refinement with the aim of obtaining an (quasi) uniform element error distribution. This approach has shown to be effective in order to eliminate any "pollution" from singularities in the domain or at the boundary as well as achieving optimal convergence rates for problems involving rough right hand sides. Thus, by means of adaptive mesh refinement we may achieve

$$\alpha = p \tag{25}$$

An important step in such adaptive refinement processes is *a posterirori* error estimation that provides a reliable element error distribution, see Ainsworth and Oden [1]. To achieve (quasi) uniform element error distribution we may subdivide those elements that have an element error that is above the average, or we may restrict ourself to subdivide those with the β percent elements with largest error contribution.

In classical FEM, the traditional way of refining a quadrilateral element is by subdivision, i.e. inserting a cross to obtain four new elements. If the aspect ratio (width to length ratio) is undesirable large, one may extend the algorithm to inserts only a single line, splitting the element into two new elements. This way of adaptive refinement give raise to so-called “hanging nodes” for which there are several techniques to reestablish the appropriate C^0- continuity.

3.2 The isogeometric finite element method

Spline spaces

In isogeometric FE methods we introduce splines as basis functions. The most common spline bases are tensor, either defined by B-splines or NURBS. Herein we will use tensor-product B-splines as well as locally refined B-splines denoted LR B-splines. Given the global knot vectors

$\Xi = [\xi_1, \xi_2, ..., \xi_{m+p+1}]$ and $\mathcal{H} = [\eta_1, \eta_2, ..., \eta_{n+q+1}]$ and an $m \times n$ grid of control points $\mathbf{C}_{i,j}$, then a tensor 2D B-splines surface of polynomial order p in x-direction and q in y-direction may be written as follows:

$$\mathbf{F}(\xi, \eta) = \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbf{C}_{i,j}\, B_{p,\Xi_i}(\xi) \cdot B_{q,\mathcal{H}_j}(\eta) \tag{26}$$

Here, the local knot vectors Ξ_i and $\mathcal{H}_j$ are defined as a subsequence of the corresponding global knot vectors Ξ and $\mathcal{H}$, respectively. The function space we use is given by

$$\mathcal{S}_{p,q} = \text{span}\,\{B_{p,i}(\xi) \otimes B_{q,j}(\eta)\}_{i=1,j=1}^{m,n} \tag{27}$$

For LR B-splines, these will instead be defined over a single running global index i using the local knot vectors Ξ_i and $\mathcal{H}_i$ by

$$\mathbf{F}(\xi, \eta) = \sum_{i=1}^{n_{dof}} \gamma_i \mathbf{C}_i B_{p,\Xi_i}(\xi) \cdot B_{q,\mathcal{H}_i}(\eta) \tag{28}$$

where the regularity is given by the local knot vectors and γ_i is the weighting factors needed to obtained partition of unity, see the Section 2. Let the function space spanned by LR B-spline basis functions be denoted by

$$\mathcal{L}_{p,q} = \text{span}\left\{B_{\mathbf{\Xi}_i}(\xi, \eta)\right\}_{i=1}^{n} \tag{29}$$

A proper function space for the FE trial functions v_h and the FE test functions w_h to achieve a compatible FE space is as follows:

$$\mathcal{V}_h(\Omega) = \left\{w_h \in \mathcal{V}(\Omega) \mid w_h(\mathbf{F}^{-1}(x_1, x_2)) \in \mathcal{L}_{p,q}(\xi, \eta)\right\} \tag{30}$$

where the coordinate mapping F is assumed to be an onto and invertible mapping between the parameter domain denoted by $\hat{\Omega}$ and the true domain Ω, i.e. for any $(x_1, x_2) \in \Omega$ there exist $(\xi^*, \eta^*) \in \hat{\Omega}$ such that $(x_1, x_2) = F(\xi^*, \eta^*)$.

Convergence rates for splines

Our model problem herein is the Poisson problem for which we have that:

$$||u - u_h||_E = \sqrt{a(u - u_h, u - u_h)} = (\nabla(u - u_h), \nabla(u - u_h)) = |u - u_h|_{H^1}. \tag{31}$$

Thus, for our model problem the a-norm is equal to the H^1 semi-norm. The a priori convergence estimate given in Equation (22) is proven to hold for tensor B-splines (or NURBS), see Bazilevs *et al.* [2]. We conjecture that it also holds for the LR B-splines that we are using herein. Note that we are not actively using the a priori convergence rates in our adaptive refinement strategies. However, we will in the numerical studies compare the obtained convergence rates towards the ones given by Equation (22).

Adaptive refinement of isogeometric finite elements

As shown in Section 3.1 by proper adaptive mesh refinement (AMR) we may utilize the full power of higher order methods and that is highly relevant for Isogeometric FE-methods. One important application of LR B-splines is to use them as an enabling technology for achieving optimal convergence order, i.e. accurate and efficient FE-models. Herein, our aim is to demonstrate and test the performance obtain by adaptive refinement using LR B-splines. Thus, we have chosen to investigate this by solving benchmark problems with known analytical solution — the exact error is thus computable.

The refinement algorithm chosen herein is based on increasing our solution space by $\beta \cdot n_{dof}$ new degrees of freedom for each iteration, where β is a prescribed growth parameter. This is achieved by continued refinement of the elements having the greatest elemental error contribution, ρ_e, to the global relative error, ρ

$$\rho_e = \frac{||u - u_h||_{E(\Omega_e)}}{||u||_{E(\Omega_e)}} \quad \text{and} \quad \rho = \frac{||u - u_h||_{E(\Omega)}}{||u||_{E(\Omega)}} = \sqrt{\sum_e^{n_{el}} \rho_e^2} \tag{32}$$

Typical we choose $5\% \leq \beta \leq 20\%$. Small values for β gives more accurate refinement process, whereas larger values result in fewer refinement steps.

Regarding adaptive refinement of isogeometric finite elements there have been some recent attempts using T-splines and hierarchical B-splines, see [5, 7, 21, 22]. It is common to mark those elements (knot-span) with highest energy error and subdivide into into four new elements by inserting a cross (ignoring large aspect ratio elements for now) through the element center.

As discussed in the previous section, the length of the crossing LR meshlines will have to be of a certain length in order to actually split a basis function properly. The actual length is depending on the surrounding topology of the mesh, and may split some neighboring elements into two new elements. In order to do a splitting we will need to compute the length of the new knot lines to ensure a proper meshline extension. This can be extracted by the element to basis function correspondence, which lists all basis functions which have support on each particular element. This is already available as it is needed in the assembly of the stiffness matrix, so no extra computations are required. Moreover, this eliminates the need for expensive topological searches.

4 Adaptive mesh refinement using LR B-splines

4.1 LR spline refinements

Although Dokken *et al.* [6] describe how to manipulate the LR B-splines when inserting knot lines, it is still up to the implementer to choose exactly which knot lines to use for refinement purposes. The inserted knot lines must at least entirely split an existing B-spline, which puts a minimum length requirement on it. This is to ensure a proper meshline extension which causes a B-spline in a state of having not minimal support, according to Definition 12. We have several options available when doing the refinement. Not only the length and the position of the knot lines, but also their multiplicity as splines in general open for duplicate knots. We will in the numerical examples investigate how much impact these choices made in the refinement process have on the properties of the resulting LR B-splines space.

As a starting point for the refinement algorithm we have a list of element error contributions ρ_e, see (32). This may be an estimated error based on some a posterior error estimator, or exact error in case of a known exact solution. For all results presented here, we are using the latter. A straightforward implementation would be to refine the β percent elements with largest error contribution. However, this will not suffice for our purposes, since we will be comparing different refinement schemes. To ensure a proper growth of the solution space $\mathcal{S}$ we propose to continuously insert new knot lines according to some algorithm until we have β percent more B-splines in our spline space. This will ensure that different refinement schemes are comparable.

To see an example of the converse, consider inserting a new knot in a 1D univariate set of B-splines. Using multiplicity 1, this will increase your spline space by 1, while using multiplicity p will cause p new B-splines, even though the number of elements refined is the same in both cases.

Definition 18. The **refinement parameter** β of an iterative scheme is defined such that two LR splines $\mathcal{L}_i$ and $\mathcal{L}_{i-1}$ satisfy

$$\begin{array}{rcl} \mathcal{L}_{i-1} & \subset & \mathcal{L}_i \\ (1+\beta)\,|\mathcal{L}_{i-1}| & \leq & |\mathcal{L}_i| \end{array}$$

Which simply states that $\mathcal{L}_i$ should be a refinement of $\mathcal{L}_{i-1}$ and the number of B-splines (not elements) should grow by at least β percent each iteration.

Assume we have a LR-mesh as given in Figure 17 and we want to refine the element $T_e = [2,4]\times[1,2]$. To find out what are appropriate LR meshline extensions, we list all B-splines with support on this element and these are

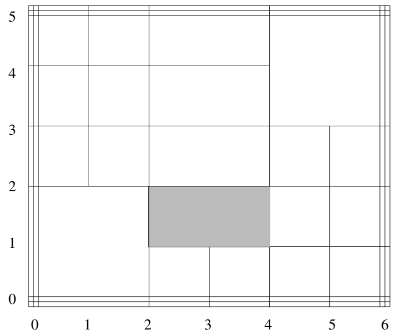


Figure 17: LR-mesh with an element marked for refinement

Table 3: B-splines with support on $[2,4] \times [1,2]$ in Figure 17 - 18.

i	Ξ_i	$\times$	$\mathcal{H}_i$
1	[0024]	$\times$	[0002]
2	[0245]	$\times$	[0002]
3	[2456]	$\times$	[0012]
4	[0024]	$\times$	[0023]
5	[0245]	$\times$	[0023]
6	[2456]	$\times$	[0123]
7	[0024]	$\times$	[0234]
8	[0246]	$\times$	[0235]
9	[2466]	$\times$	[1235]

shown in Figure 18 and tabulated in Table 3. An obvious choice when choosing the refinement line, is to make sure that it is refining *every* B-spline over that element. This can simply be done by making it run from the smallest ξ-knot value to the largest, and likewise in the other parametric direction. In this particular case one would then have a ξ-line going from $\xi = 0$ to $\xi = 6$ with constant $\eta = 1.5$ such that it passes through the center of the element. For the other parametric line, this would then have to run from $\eta = 0$ to $\eta = 5$ through $\xi = 3$. This is the "safe" way of refining, as one will have little reason to prioritize refining some B-splines over others given the limited information available.

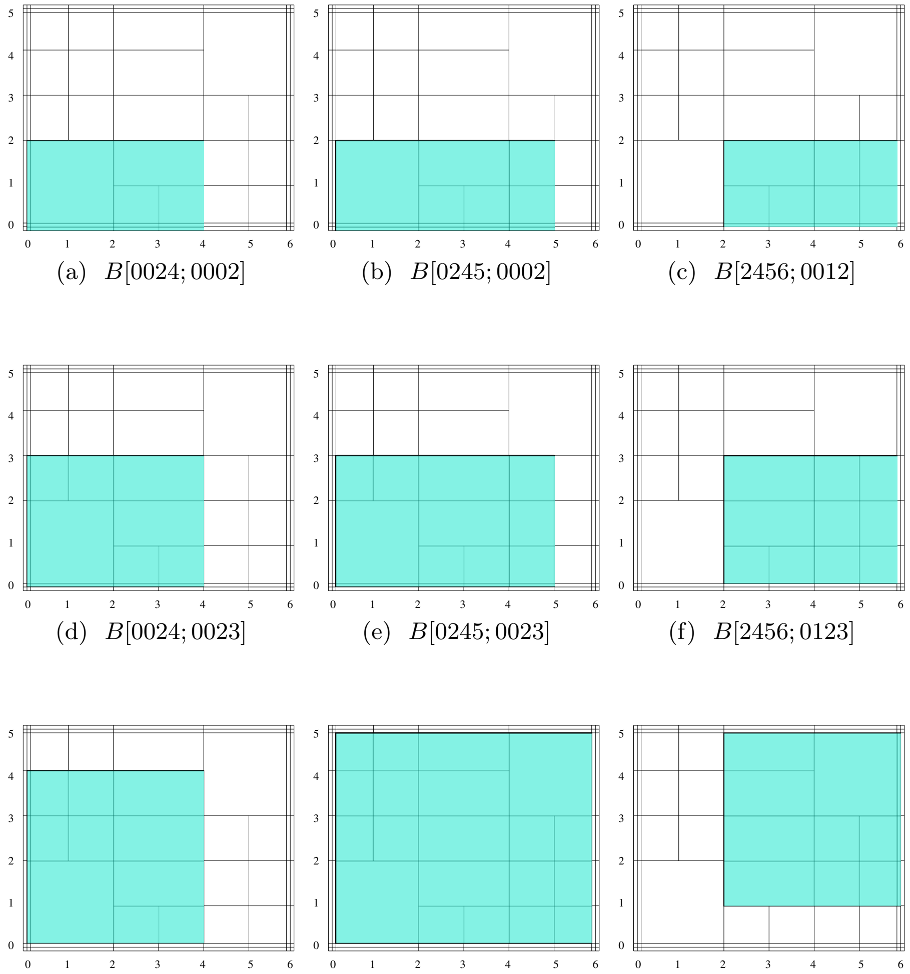
However another obvious option also comes to mind. Since we are doing *local* refinement, it is natural to want the refinement lines to be as local as possible. One might argue that we should insert the smallest possible line, while still being long enough that it actually splits an entire B-spline. This information can directly be computed from the local B-spline list in Table 3 as one can iterate through the list and choose the smallest knot vector and choose this. Note that this comes with the loss of uniqueness, since both Ξ_1 and Ξ_3 has parametric ξ-length 4 in the above example. We now must decide on whether to use Ξ_1 and insert $[0, 4]$ or to use Ξ_3 and insert $[2, 6]$. Of course this will have implications on the resulting spline space, but without more knowledge of the underlying problem, it is not possible to say which one is advantageous over the other. We thus propose to pick a *random* function of all the available with smallest parametric support. Notice again we don't need any topological information about the surrounding mesh topology to insert the knot lines. We are simply extracting every information we need locally, and the resulting refinement will not yield any more than what is extracted here.

As all generalizations of B-spline spaces, LR B-splines do also allow for duplicate knots. Duplicate knots work just in the same way as they do for regular tensor product B-splines, in which the B-splines all have continuity C^{p-m} where p is the spline polynomial degree and m is the knot multiplicity. We may choose to insert not regular knot lines, but also knot lines of higher multiplicity. This is obviously a good idea if we actually *want* the lower continuity for instance if we either are doing geometry modelling or if we know something about the underlying problem. However this is not the only reason to include the double knot lines. As will become apparent later, this will help us keep the refinement more localized and reduce the propagation effects.

So to sum up: we have several design parameters when choosing a refinement scheme. We may choose

- which B-splines with support on the element to refine
- the location of the element split
- the multiplicity of the inserted knot lines

the second point refers to the fact that one doesn't necessary need to create a cross through the element *center* as described in the example above. There is nothing preventing us from inserting two lines in both directions through each element, placing them one third from the edge each, effectively splitting the element into 9. For this discussion, we will restrict ourself to inserting crosses through the element centers. These are all illustrated in Figure 19 where we refine a shaded element using different techniques.

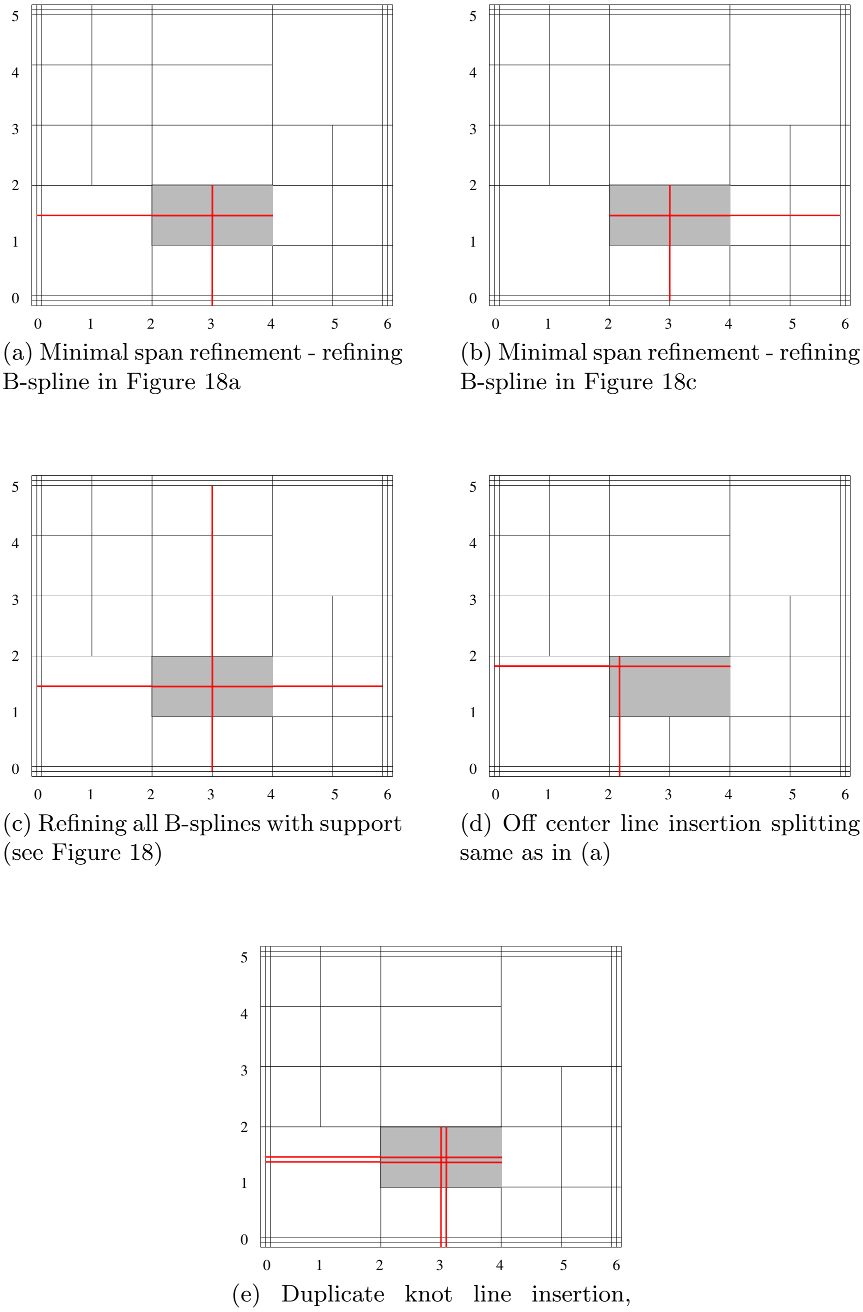


(a) $B[0024; 0002]$ (b) $B[0245; 0002]$ (c) $B[2456; 0012]$

(d) $B[0024; 0023]$ (e) $B[0245; 0023]$ (f) $B[2456; 0123]$

(g) $B[0024; 0234]$ (h) $B[0246; 0235]$ (i) $B[2466; 1235]$

Figure 18: All B-splines with support on the element $[2, 4] \times [1, 2]$.



(a) Minimal span refinement - refining B-spline in Figure 18a

(b) Minimal span refinement - refining B-spline in Figure 18c

(c) Refining all B-splines with support (see Figure 18)

(d) Off center line insertion splitting same as in (a)

(e) Duplicate knot line insertion, splitting the same as in (a)

Figure 19: Different choices for refining the shaded element.

4.2 Local refinement strategies for LR B-splines

We will here present three different local refinement strategies that will be used in our numerical examples. The starting point for all of these is the assumption that we have identified a set of elements which needs refinement and proceed to refine these using one of three strategies. The goal is to split the marked element (knot span) into four new elements by inserting a cross. However, as already discussed, this cross cannot be limited to only spanning the marked element.

Full span

Our strategy here is to refine *every* B-spline with support on the marked element. The inserted meshline in the ξ-direction will then have to span from the minimum ξ-knot to the maximum ξ-knot of all functions with support on the marked element. Likewise for the meshline in the η-direction. The exact length of the two inserted meshlines are extracted by using the list of element to B-splines correspondence.

This strategy will make sure that all B-splines with support on the marked element (knot span) are treated equally and all of them will be split by the refinement. However, the drawback of this strategy is that one get a somewhat large footprint. Moreover, the neighbouring elements will be split by a single line which will in essence double their aspect ratio. This is depicted in Figure 20a where one can clearly see the rectangular shaped neighbouring elements arising from this strategy.

Minimum span

This refinement strategy inserts a cross through the marked element center with the aim of making the refinement footprint as small as possible. Thus, we want the inserted meshlines to be as short as possible, but still splitting at least one B-spline. From the list of element to B-spline correspondence we may deduct which B-splines having the smallest ξ- and η-support. Note that this comes with the loss of uniqueness as there may be several B-splines with the same length of the ξ-span, but with different local origin, i.e. for the B-spline i and j we may have $\xi^i_{p+1} - \xi^i_0 = \xi^j_{p+1} - \xi^j_0$ but $\xi^i_0 \neq \xi^j_0$ (see Figure 18a - 18c). This local refinement strategy is depicted in Figure 20b where we have chosen to split only one of the several available B-splines. Due to the (in general) lack of uniqueness of which B-splines to split, we will herein propose to do a *random* choice of which B-spline to refine. This will cause properties such as symmetry to be lost.

Structured mesh

The idea of refining elements is a legacy from the finite element method where every inserted vertex would correspond to an additional degree of freedom. With LR B-splines this is not the case and as seen from the two previous schemes the required length of the inserted meshlines may vary from element to element. Another way of refining LR B-splines is identifying *B-splines* which needs to be refined as opposed to which *elements*. In the case of the synthetic diagonal refinement problem presented later, these are easily extracted as all functions along the diagonal that satisfy $\xi_i = \eta_i$ for $i = 0, ..., p+1$. However, for general isogeometric finite element computations we need a criteria to identify which B-spline to be refined. We propose the following definition

Definition 19. The **B-spline error** is the sum of element error over all supported elements, i.e.

$$\|e\|^2_{\mathcal{M}(N_i)} = \sum_{K \in \mathcal{M}(N_i)} \|e\|^2_K \tag{33}$$

where $\mathcal{M}(N_i)$ is the set of elements on which the B-spline N_i is nonzero, and $\|e\|_K$ is the usual element error measured in energy norm,

$$\|e\|^2_K = a(u - u_h, u - u_h)_{\Omega_K} \tag{34}$$

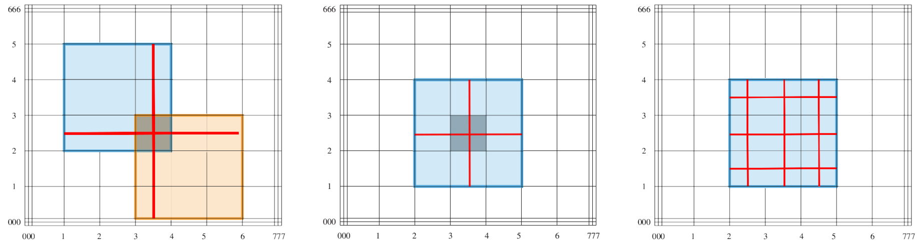
Once the B-splines which are subject to refinement are identified, we proceed to refine these by inserting a net of knot lines halving the largest supported knot intervals as shown in Figure 21.

Regularity

We note that just like tensor product B-splines, LR B-splines also allow for duplicate knots. The effects of duplicate knots is twofold. Firstly it reduces the regularity, such that a knot of multiplicity m will give rise to a C^{p-m} function across that knot, where p is the polynomial degree. In addition to the decreased regularity, one also decreases the support of the function. This will in turn diminish the propagation effects of the refinement. We will investigate refinement using different regularities.

4.3 Hanging nodes in FEM versus LR B-splines

Adaptive refinement of classical quadrilateral (Lagrange) FE has been achieved by means of many different approaches, e.g: Subdivision of marked

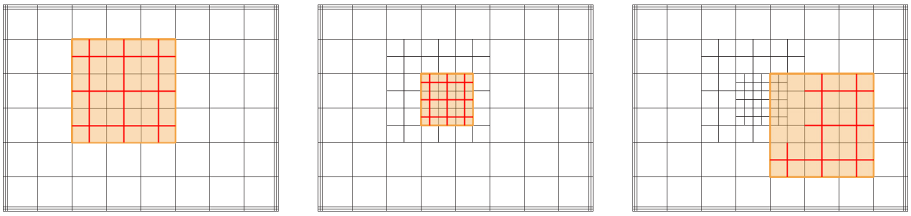


(a) Full span - split all functions on one element, here only two of all the nine functions with support on this element is depicted

(b) Min span - split one random functions on one element, note that for odd order splines (or by a poor random pick), the symmetry would be lost

(c) Structured Mesh - split all knot spans on one B-spline, notice that no bad aspect ratio elements are created

Figure 20: The ideas behind the different refinement strategies, here illustrated on a quadratic tensor product mesh. Notice the fundamental difference in that 20a - 20b is refining an element, while 20c is refining a B-spline.



(a) Iteration 1

(b) Iteration 2

(c) Iteration 3

Figure 21: Three iterations of an example structured mesh refinement. Notice that we at each iteration halve the largest supported elements. A selection of LR B-splines over the mesh from iteration 3 is depicted in Figure 22

- patch of elements into smaller elements with transition zones to contain C^0 continuity
- elements into four new elements using multipoint constraints to contain C^0 continuity
- elements into four new elements using transition elements to contain C^0 continuity

To give some insight into the developed local refinement strategy using LR B-splines we will below illustrate how it for $p = 1$ compares to the the concept of using transition elements for adaptive refinement of FE grid. The comparison is chosen for a basic refinement case and we would like to emphasize that adaptive refinement using LR B-splines is in general more versatile than the concept of using transition elements.

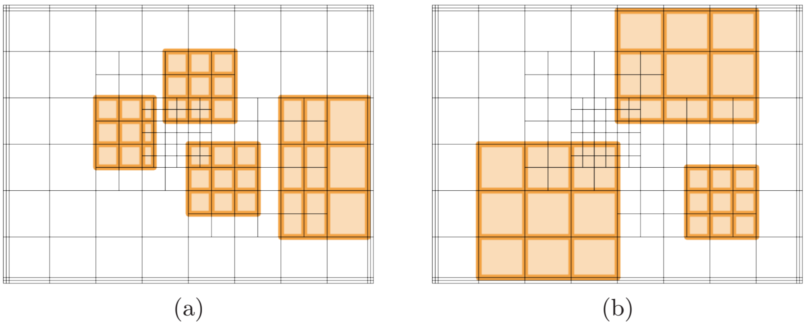


Figure 22: Some example quadratic LR B-splines over the LR mesh from Figure 21c

Refinement for $p = 1$

Assume that we want to divide the centre FE element in the grid shown in Figure 23 in two elements by a horizontal split[1]. The concept of using transition elements then implies that we to the left and right of the centre element introduce 5-noded transition elements, see Hughes [10]. In Figure 24 we have displayed the element nodal shape functions for three of the five nodes. For node 1 and 4 the nodal shape functions for the 5-noded transition element are identical to those for node 1 and 4 for the standard bilinear 4-noded quadrilateral. However, for node 2 and 3 we have to modify the element nodal shape functions compared to the 4-noded in order to achieve that the shape function in node i evaluated at node j is either equal to 1 if $i = j$ or equal to 0 if $i \neq j$. Finally the element shape function for the new inserted node 5 is as displayed in Figure 24c. It is easy to verify that C^0-continuity is attained for the refined FE grid displayed in Figure 24.
To achieve the same refinement using LR B-splines we would insert a horizontal meshline as displayed in Figure 25. The nodal basis functions corresponding to those in Figure 24a–24c are displayed in the Figures 26–28, respectively. The leftmost column is showing an alternative way of plotting LR B-splines and is to be understood in the following way. Each continuity reduction line i.e. the LR meshlines, is plotted. For each B-spline we plot an ellipse with center at the Greville point (average value of the local knot vector) and with a size corresponding to the size of the support of that particular B-spline. Furthermore, we have shaded the ellipse for the particular B-spline which is shown, as well as the support of that function.

The discrete function space after refinement is identical for the FE and LR B-spline case. The only difference is that we in the LR B-spline case

[1] A horizonal split is chosen for simplicity instead of a cross, but the this comparison apply for a cross as well.

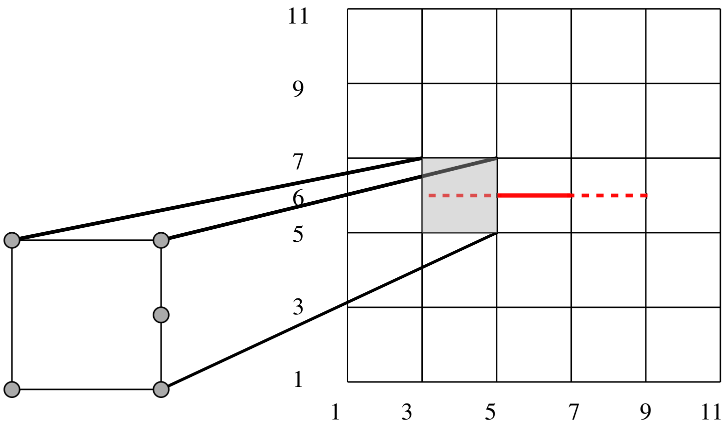
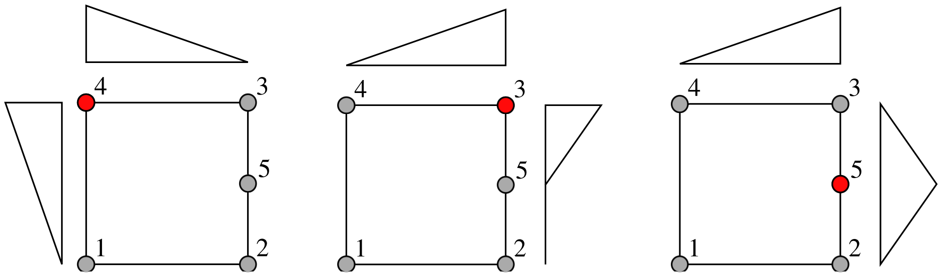


Figure 23: The transition element concept: Horizontal split of the centre element into two new elements.

get three more elements, whereas for the FE case we only get one new element. However, notice that we in Figure 23 have stipulated the transition elements to indicate that one should treat them as two elements when performing numerical integration. The reason being that the element nodal shape functions 2, 3, and 5, are not infinitely smooth across the stipulated line (they are only C^0). Thus, in practice we need to do the same amount of work related to numerical integration for both the FE and LR B-spline case.



(a) Element nodal shape function for element node 4 is unchanged

(b) Element nodal shape function for node 3 is modified

(c) Element nodal shape function for the inserted node 5

Figure 24: The transition element concept: The 5-noded transition element and its element nodal shape functions.

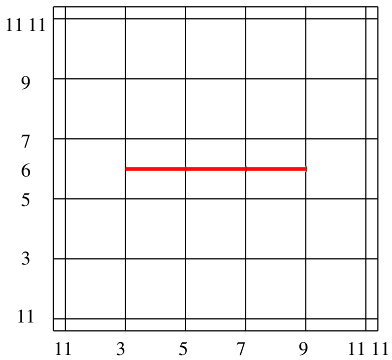
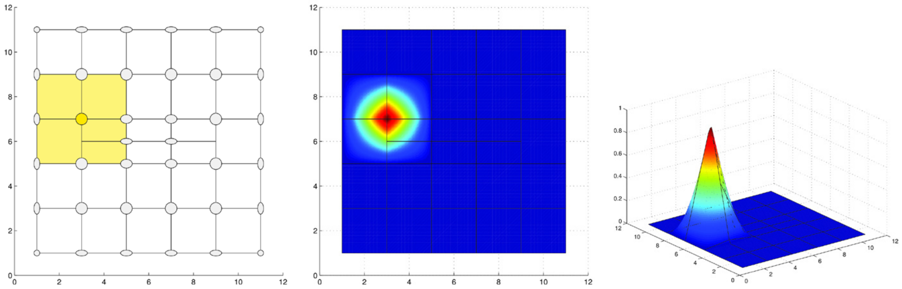


Figure 25: LR B-spline refinement: Inserting a horizontal meshline in order to to split the centre element into two new elements.

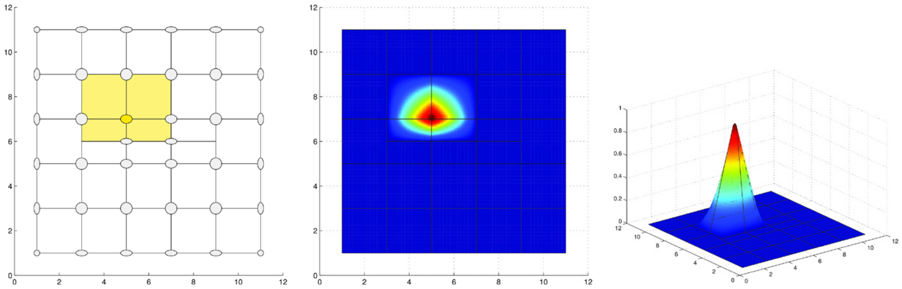


(a) Function space and support

(b) Top view

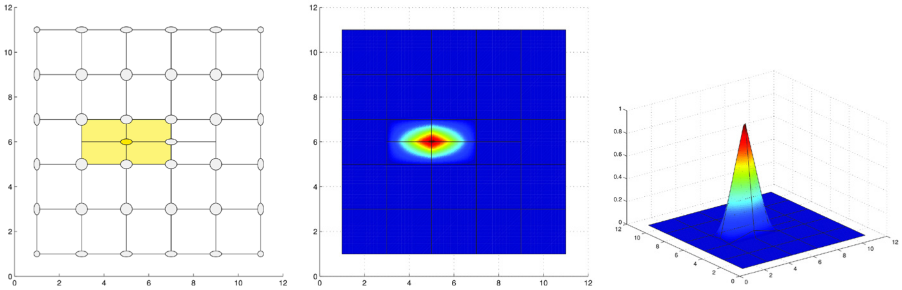
(c) Perspective view

Figure 26: B-spline $B[135; 579]$.



(a) Function space and support (b) Top view (c) Perspective view

Figure 27: B-spline $B[357;679]$.



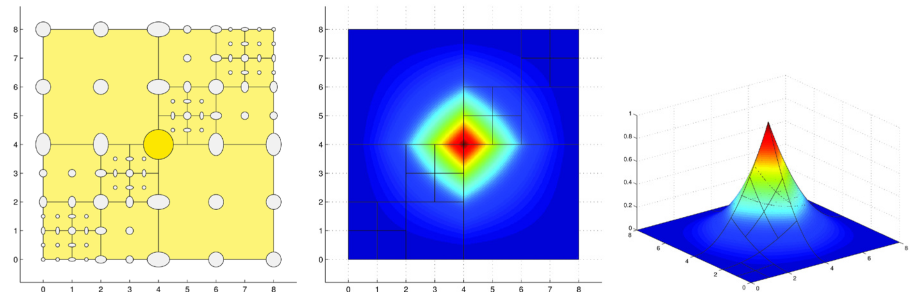
(a) Function space and support (b) Top view (c) Perspective view

Figure 28: B-spline $B[357;567]$.

C^0-Refinement for $p = 2$

To make a comparison between classical Lagrange functions and LR splines with $p = 2$, we must consider C^0 elements. This is perfectly possible by using double knot lines, which is done here. The example mesh is taken from the diagonal refinement case for $p = 2$ and $m = 2$ which is going to be discussed in the Section 5.2. We see that the Greville points for $p = 2$ do in fact line up with the traditional way of drawing Lagrangian biquadratic finite element nodes. We have the usual 9 nodes for each element, provided that there are no hanging nodes nearby. Note however that the basis functions themselves are different. LR B-splines are non-negative, while Lagrange functions do take negative values. Nevertheless, they have the same support, and we see that around the hanging nodes, the basis functions vanish. This is equivalent to setting multipoint-constraints on these nodes, effectively removing them as a degree of freedom. This is can be seen in Figure 29 - 33, where several of the B-splines are shown. Also note that there is no upper bound on the number of hanging nodes on a single element, as several elements have 2 hanging nodes in this example.

It is interesting to see that one might recreate the Lagrangian function space, also with hanging nodes. However it is important to note that this is something that we in general will not try to do. Doing this causes us to loose the smoothness which is characteristic of all spline spaces, and this is a property which we would like to preserve.

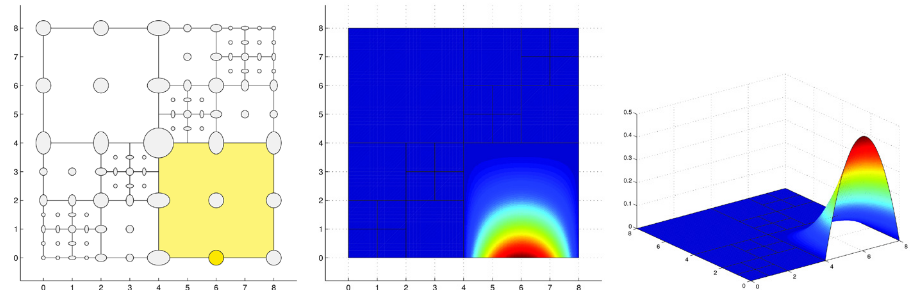


(a) Function space and support

(b) Top view

(c) Perspective view

Figure 29: B-spline $B[0448; 0448]$

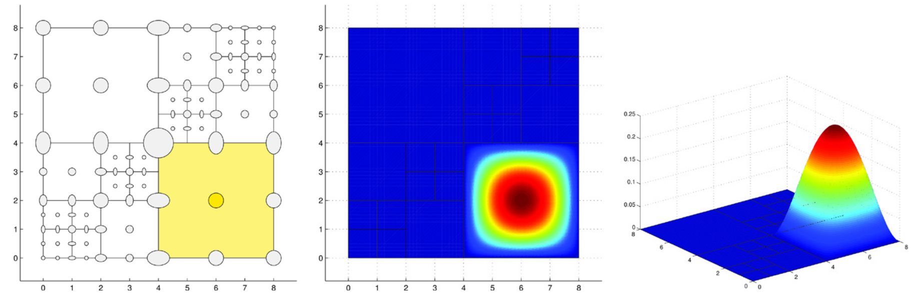


(a) Function space and support

(b) Top view

(c) Perspective view

Figure 30: B-spline $B[4488; 0004]$

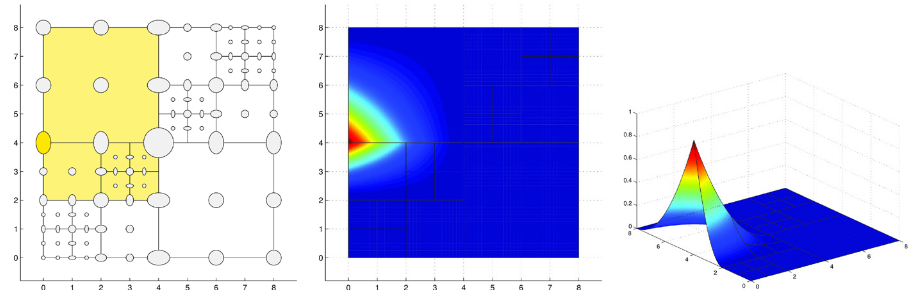


(a) function space and support

(b) Top view

(c) Perspective view

Figure 31: B-spline $B[4488; 0044]$

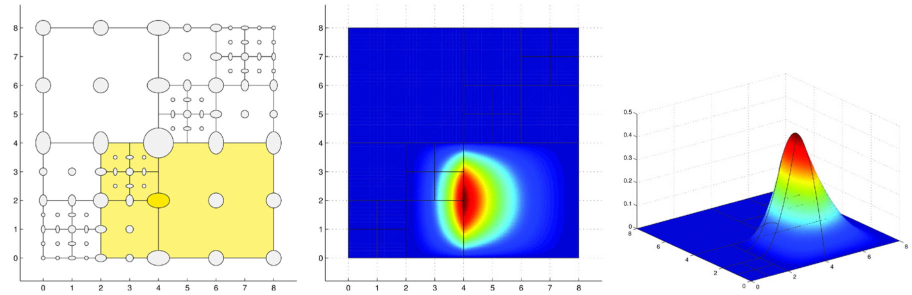


(a) function space and support

(b) Top view

(c) Perspective view

Figure 32: B-spline $B[0224; 2448]$



(a) function space and support

(b) Top view

(c) Perspective view

Figure 33: B-spline $B[2448; 0044]$

5 Numerical Results

5.1 Preliminaries

To demonstrate the performance of adaptive refinement using LR B-splines, we study one synthetic case and two Poisson type problems with known analytic solutions. The synthetic case denoted *Diagonal Refinement* is chosen as it illustrates very well the spreading effect of the refinement schemes and has been addressed by other researchers in the isogeometric community ([5, 7, 13]). The first Poisson example denoted *L-shape* is chosen as it has a point singularity at the boundary that causes reduced convergence rate when performing uniform refinement. Whereas the next Poisson problem *Interior Layer* has a rough right hand side that impose a sharp layer along a circular arc in the interior of the domain. The asymptotic convergence rate is here suboptimal for uniform refinement until the uniform element size h becomes smaller than a threshold given by the width of the interior layer.

The aim of the numerical experiments herein is to investigate whether adaptive refinement using LR B-splines achieves optimal convergence rate for non-smooth problems such that it gives better accuracy per dof compared to uniform refinement. Thus, the adaptive strategy is based on refining a prescribed portion of the the elements, i.e. $\beta \cdot n_{\text{el}}$ having the greatest elemental contribution, ρ_e to the global error, ρ in order to achieve uniform element error distribution. Furthermore, we want to investigate the sensitivity in accuracy and convergence rates towards relevant parameters e.g. polynomial order p, regularity C^r and local refinement strategies.

All the cases are analysed with LR B-splines of polynomial order $p = 2, 3, 4$. We have performed the tests with different regularity, C^r, were $0 \leq r \leq p - 1$, obtained by using multiple knot lines. The multiplicity $m = 1$ corresponds to maximum regularity $r = p - m = p - 1$ whereas $m = p$ corresponds to minimum regularity $r = p - m = 0$. We have used two different local refinement strategies denoted *full span* and *structured mesh*.

For the synthetic case *Diagonal Refinement* we present the following results:

- Refined grids: Representative examples of refined grids
- Tables showing the number of dofs and elements for each refinement step
- Graphs showing the relation between number of elements and number of dofs

For the two Poisson cases (*L-shape* and *Interior Layer*) we present the following results:

- Convergence plot: Log of relative error vs log of number of degrees of freedoms (n_{dof})
- Refined grids: Representative examples of refined grids
- Error distribution: Elemental contribution, ρ_e, to the total error ρ
- Root mean square error of the element error distribution

The exact error $e = u - u_h$ is measured in the energy norm (a-norm) $||e||_E$ as given in Equation (21). Let $||e||_{E(\Omega)}$ and $||e||_{E(\Omega_e)}$ be the global and element error, respectively. Then we define the root mean square of exact element error:

$$||e||_{\text{RMS}} = \left(\frac{1}{n_{\text{el}}} \sum_{e=1}^{n_{\text{el}}} (||e||_{E(\Omega_e)} - ||e||_{\text{avg}})^2 \right)^{1/2} / ||e||_{\text{avg}} \qquad (35)$$

where the average exact element error is defined as

$$||e||_{\text{avg}} = \frac{1}{n_{\text{el}}} \sum_{e=1}^{n_{\text{el}}} ||e||_{E(\Omega_e)} \qquad (36)$$

The quantity root mean square of exact element error given in Equation (35) measures the deviation from an uniform element error distribution. For uniform element error distribution we have $||e||_{\text{RMS}} = 0$. Thus, we refer to *asymptotically optimal mesh refinement* (see Kvamsdal and Okstad [12]) as a sequence of meshes satisfying

$$\lim_{h \to 0} ||e||_{\text{RMS}} = 0 \qquad (37)$$

5.2 Diagonal refinement

As an introductory example we look at the diagonal refinement. This example highlights some of the problems that local refinement strategies face since the request for refinement is in conflict with the parametrization direction. With the parametrization being parallel to the coordinate axes, and the diagonal 45 degrees on this, one will have to refine both parametric directions equally. Dörfel *et al.* [7] showed that under T-spline refinement this could provoke a worst-case scenario where the mesh lines propagate through the entire domain.

We will here present three different refinement strategies for this particular problem and analyse the resulting spline space resulting from these. The starting point for all of these is the assumption that we have identified a set of elements which needs refinement (here: the diagonal elements) and proceed to refine these using one of three strategies.

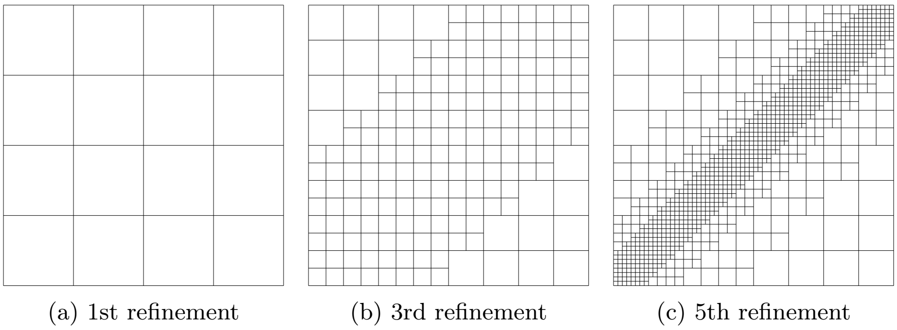

(a) 1st refinement (b) 3rd refinement (c) 5th refinement

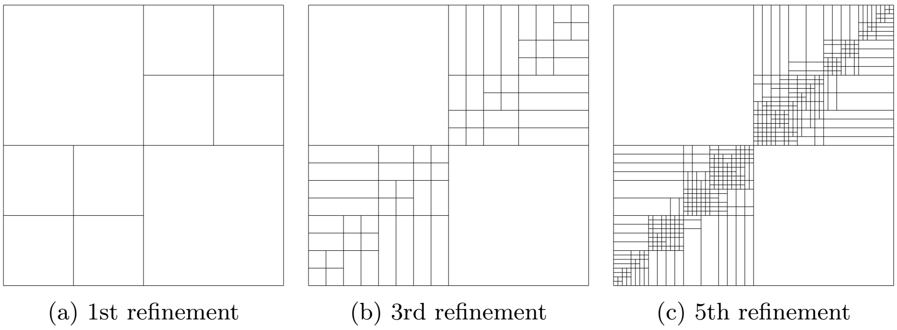
Figure 34: The Diagonal Refinement problem: *Full span* and *structured mesh* refinement strategy using single knot lines and bicubic B-splines. Note that in the special case of diagonal refinement, these strategies coincide completely

As discussed already, every inserted mesh line must at least span the support of at least one basis function. Thus it is in general not possible to only insert a single cross through one element when refining. If the inserted knot lines are limited to that element, then they will in general not be long enough to traverse the entire support of a basis function.

Results

Several refinement strategies was tested to see their performance on this benchmark test for local refinement. The setup was using $p = 3$ in each parametric direction and trying to refine the elements along the diagonal. No a priori knowledge on the problem was used as the input was just a given set of elements to be refined on a general LR B-spline. The first strategy that we tested was the *full span* strategy which for all tagged elements, chooses to refine all B-splines with support on that element. To accomplish this, we need a rather long mesh line in both ξ- and η-direction. Thus the characteristic propagation effect is rather large as can be seen in Figure 34 where several steps of the refinement process have been illustrated. Even if the propagation is clearly apparent, the refinement is still very much contained in a band along the diagonal and global refinement is avoided.

The diagonal test case is very exceptional in the sense that bad aspect ratio elements that are characteristic of the *full span* refinement are all canceled out by the next level on the diagonal. Thus the *full span* strategy produces identical meshes as the *structured mesh* strategy. However, the differences between these two strategies become apparent when they are used in an adaptive refinement process as shown in the next two subsections.



(a) 1st refinement (b) 3rd refinement (c) 5th refinement

Figure 35: The Diagonal Refinement problem: *Minimum span* refinement strategy using single knot lines.

The final option is the *minimum span* refinement. Due to the fact that this picks a random function we introduce stochastic effects in our refinement strategy and things such as symmetry is in general lost. It does however turn out to be more local than in the previous two cases. The results of a series of iterations using this refinement strategy is plotted in Figure 35. Note that after the first refinement, not a single line is added to the top left and bottom right portion of the mesh. This is due to the interpolatory basis functions at the edge which only span one element. Since the algorithm compares meshline lengths, it will always favor crossing zero-span elements such as the ones located at the edges.

Although the latter refinement strategy does indeed reduce the effect of propagation, it is still apparent. Inspired by similar results for T-splines [5], we now test duplicate knot lines. The effect of splitting the elements using both double and triple knot lines is here shown in Figure 36 and 37. The results are quite promising as the triple knot line insertion removes *all* propagation into neighbouring elements. It is kept perfectly local and results in a very good mesh. One thing to keep in mind though when inserting duplicate knots is that each B-spline is splitted into more than two new B-splines. This results in a larger growth of the total number of B-splines than when using single knot lines. So even if the mesh seems tighter, or more compact, Figure 37 contains more B-splines than the mesh in Figure 35.

The corresponding *index* mesh to Figure 36-37 is given in Figure 38-39. Notice the high aspect ratio of some of the elements in Figure 35 and to a lesser degree in 36. This is a side effect which happens when inserted knot lines are traversing neighbouring elements and is not restricted to the element being refined. Of course, it is possible to combat this effect by recursively inserting more lines to compensate for this aspect ratio, but that would be in contrast with what we are trying to achieve here, which is to

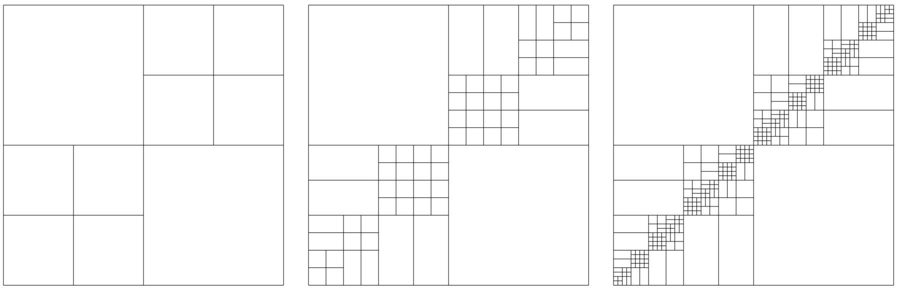

(a) 1st refinement (b) 3rd refinement (c) 5th refinement

Figure 36: The Diagonal Refinement problem: *Minimum span* refinement using double knot lines.

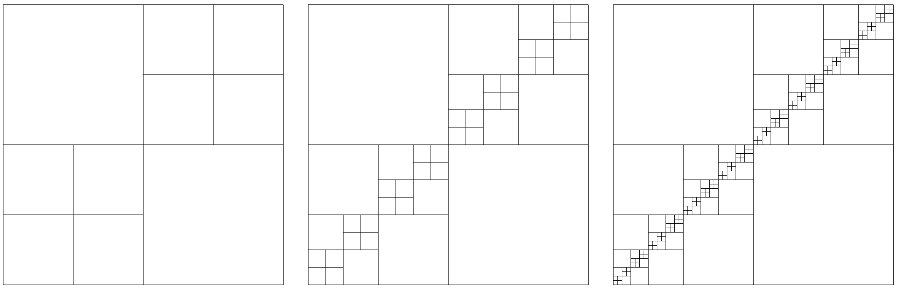

(a) 1st refinement (b) 3rd refinement (c) 5th refinement

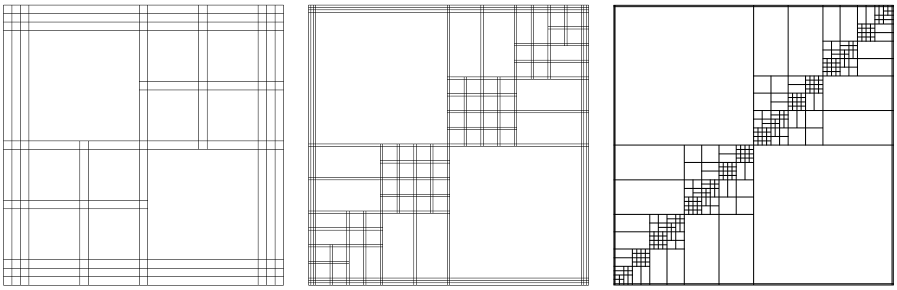
Figure 37: The Diagonal Refinement problem: *Minimum span* refinement using triple knot lines.

keep the refinement as local as possible.

Degrees of freedom versus elements

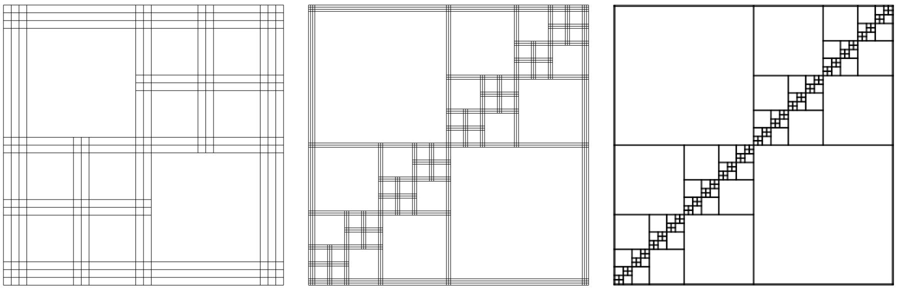
From the numerical experiments we observe that the number of basis functions (i.e. dofs) versus number of elements varies significantly with the regularity. If we compare the mesh using single knot line refinement (C^2-refinement) in Figure 35 with the mesh using triple knot lines refinement (or C^0-refinement) in Figure 37 we get the numbers displayed in the Table 5. The most refined C^0-mesh is containing approximately 5 times as many degrees of freedom and *less* than half the number of elements when compared with the most refined C^2-mesh.

We may take a deeper look into exactly how much this effect is apparent by plotting the number of basis functions and elements for our diagonal refinement case. We compare the minimum span refinement using single,



(a) 1st refinement (b) 3rd refinement (c) 5th refinement

Figure 38: The Diagonal Refinement problem: Index domain for *minimum refinement* using double knot lines.



(a) 1st refinement (b) 3rd refinement (c) 5th refinement

Figure 39: The Diagonal Refinement problem: Index domain for *minimum refinement* using triple knot lines.

double and triple knot lines. The results are given in Table 5. We see a clear tendency that the C^0 refinement keeps above 7:1 ratio between the number of degrees of freedom and the number of elements, whereas for the C^2-refinement the ratio is dropping below 1:1 for the most refined grid.

For any univariate C^r-regular B-spline basis we note that n_{el} elements gives

$$n_{\text{dof}} = n_{\text{el}}(p-r) + r - 1 \tag{38}$$

where p is the polynomial order and the knot multiplicity is $m = p - r$. For a tensor product spline with n_{el}^2 elements it is clear that this gives $n_{dof}^2 = ((p-r)n_{\text{el}})^2 + \mathcal{O}(n_{\text{el}})$. For our particular (*minimum span*) case we have $p = 3$ and $r = 0, 1, 2$ and it seems reasonable that n_{dof} is approximately 8 times larger than n_{el} for C^0 and a factor 0.6 for C^2 at the most refined grid. We see that LR B-splines shows a somewhat similar growth of basis function to elements as regular tensor product B-splines does. This is shown

Table 4: The Diagonal Refinement problem: Number of elements vs degrees of freedom using the *full span* strategy.

refinement count	C^2-elements	C^2-DOFs	C^1-elements	C^1-DOFs	C^0-elements	C^0-DOFs
1	4	25	4	36	4	49
2	16	49	16	100	16	169
3	64	121	46	220	46	439
4	196	253	112	452	112	1009
5	496	505	250	908	250	2179
6	1132	997	532	1812	532	4549
7	2440	1969	1102	3612	1102	9319

Table 5: The Diagonal Refinement problem: Number of elements vs degrees of freedom using the *minimum span* strategy.

refinement count	C^2-elements	C^2-DOFs	C^1-elements	C^1-DOFs	C^0-elements	C^0-DOFs
1	4	25	4	36	4	49
2	10	31	10	60	10	103
3	26	47	26	124	22	199
4	66	87	66	248	46	379
5	198	191	150	484	94	727
6	506	364	325	956	190	1411
7	1215	747	682	1904	382	2767

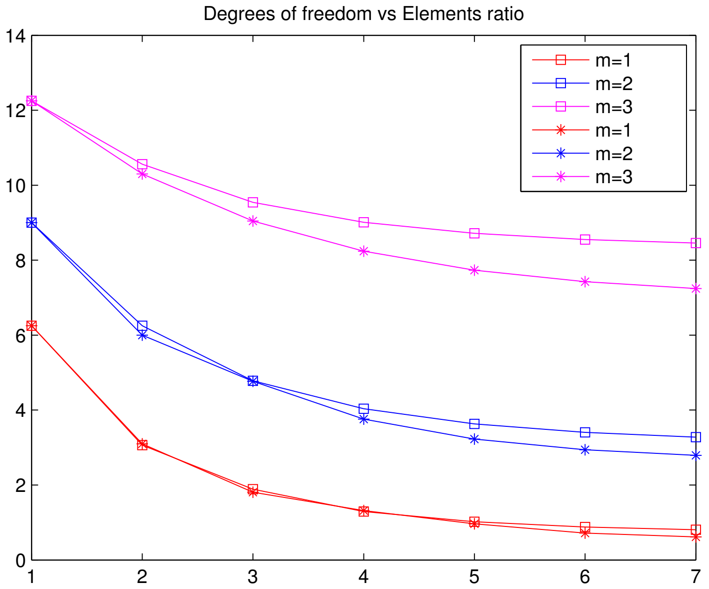
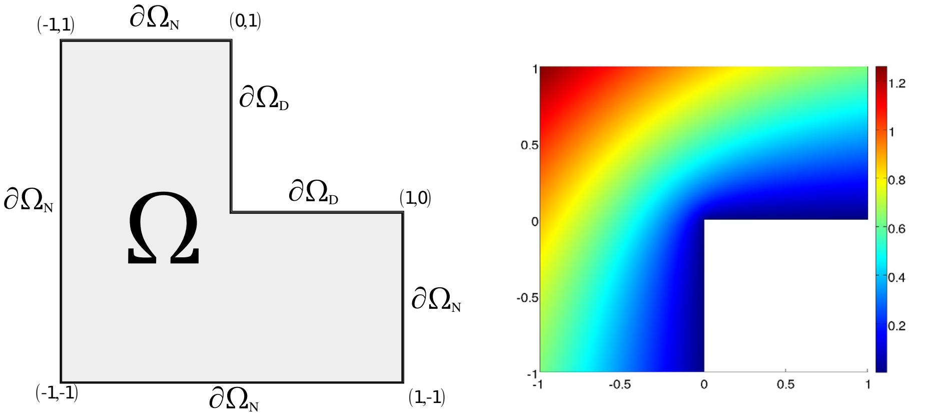


Figure 40: The Diagonal Refinement problem: Ratio of degrees of freedom versus elements using both *full span* (squares) and *minimum span* (stars) local refinement strategy. For tensor product bicubic splines, we have the asymptotic limit of 9 for $m = 3$, 4 for $m = 2$ and 1 for $m = 1$, see Equation (38).

in Figure 40 where the tabulated values are plotted.
For the solution of stationary problems such as the ones considered in this paper this doesn't have too many implications. However, for the solution of a time-dependent non-linear elasticity problem, where the global coefficient (stiffness) matrix must be assembled for each iteration, this might be a drawback. The cost of numerical integration (by Gaussian quadrature) is dominated by the number of Gaussian integration points and hence the number of elements. Due to this huge discrepancy between the number of dofs and the number of elements, one might argue that measuring convergence rates and running time should no longer be plotted as a function of dofs, but rather as a function of elements. For our purposes however we note that the bottleneck is still the solution of the linear system of equations, and hence we keep the convergence plots with dof along the x-axis.



(a) The domain Ω and the boundary conditions

(b) The exact solution, see Equation (40)

Figure 41: The L-shape problem: A Poisson problem with a singularity point on the boundary.

5.3 L-shape

Problem definition

The problem consist of solving the stationary heat equation, or Laplace equation $\nabla^2 u = 0$ on a L-shaped domain $\Omega = [-1,1]^2 \setminus [0,1]^2$ with appropriate boundary conditions, see Figure 41.

$$\begin{array}{rlll} \nabla^2 u & = 0 & \text{in} & \Omega \\ u & = 0 & \text{on} & \partial\Omega_D \\ \frac{\partial u}{\partial n} & = g & \text{on} & \partial\Omega_N \end{array} \tag{39}$$

with $g(x,y)$ given by the exact solution at the Neumann edge and $\boldsymbol{n}$ being an outward unit normal. It can be shown that

$$u_{ex}(r,\theta) = r^{2/3} \sin\left(\frac{2\theta+\pi}{3}\right) \tag{40}$$

is a solution to the Laplace equation $\nabla^2 u = 0$, and this is what we will be using as our analytical solution. The generation of g is straightforward from u_{ex} but is not given as a simple expression and the details are omitted here. The homogeneous Dirichlet boundary condition is given at $y = 0, x \in [0,1]$ and $x = 0, y \in [0,1]$, while all other edges are given with Neumann conditions (see Figure 41a). Note that the exact solution, which is pictured in Figure 41b exhibit a singularity at the origin. The function has a sharp edge at that point, and the derivative is not well defined there.

In Figure 42 we see that the convergence for uniform mesh refinement is limited by the strength of the singularity, i.e. the convergence rate is equal to $-q/2 = -1/3$. For problems where the solution is not sufficiently smooth, $u \notin H^{p+1}$, as the L-shape with a singular point on its boundary, we do not obtain optimal convergence. In particular, the use of high order polynomials is then inefficient.

Results

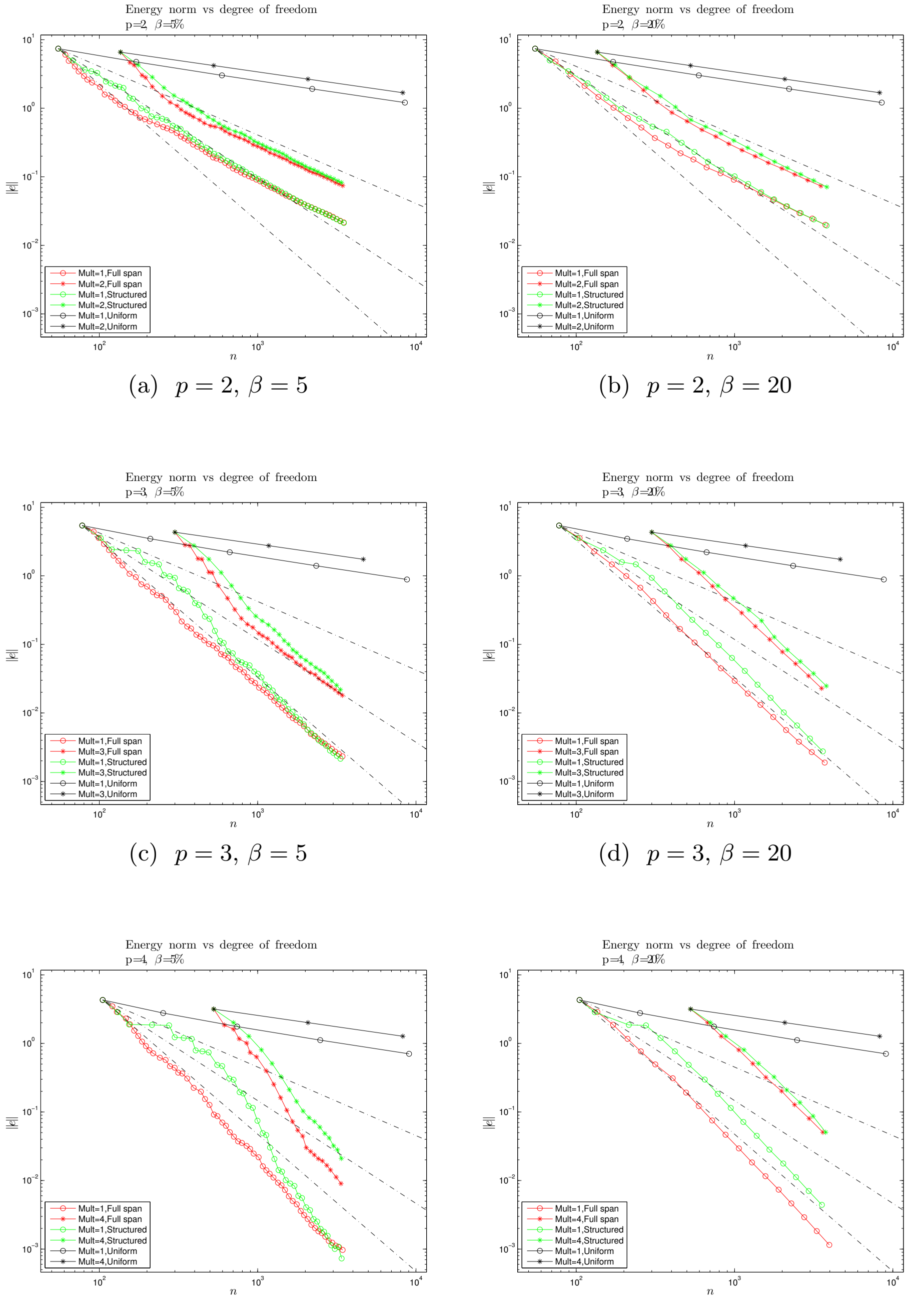
In Figure 42 we show the results obtained by adaptive refinement using LR B-splines. The results are displayed using different polynomial order $p = \{2, 3, 4\}$, portion of refinement $\beta = \{5, 20\}$, multiplicity/regularity $m = \{0, p-1\}$ and local refinement scheme.

The main observation is that we achieve optimal asymptotic convergence rate (valid for (quasi) uniform element error distribution), i.e. $-q/2 = -p/2 = \{-1, -3/2, -2\}$ for $p = \{2, 3, 4\}$, respectively.

We clearly see that high regularity (i.e. low multiplicity) is efficient when we compare relative error versus number of degrees of freedom. At first glance this seems odd, as the L-shape is one of the benchmarks for demonstrating the need for local refinement. Moreover, in the above example we concluded that the "perfect" local refinement was the one which introduced $p-1$ multiple knot lines as this did not propagate at all. However, introducing double knot lines will split each basis function into *three* new functions, as opposed to inserting a single knot line for splitting it into two. For triple knot lines this will of course split each function into four new ones. This means that the multiple knot line insertion actually gives a faster growth of the degrees of freedom. Furthermore, we see that the convergence results are more sensitive to the tested variation of local refinement strategies for higher polynomial order p and higher percentage of elements added in each refinement β. Notice that for $(p, m) = (4, 1)$ the *structured mesh* refinement gives a higher error than for the *full span* refinement strategy. Furthermore, from our experiments we saw that for $\beta = 50$ we did not always obtain optimal convergence rate, i.e. the value of β should not be chosen too large.

The resulting grids are different for the different local refinement strategies. This is illustrated in the Figures 43–45 where we have displayed the effect on the refined grid using different local refinement strategies. As can be seen in the two Figures 43 and 44 the *full span* method have more elements with high aspect ratios than the *structured mesh* method, but the latter one give a more widespread stepwise uniform refinement towards the singularity point. However, when terminating at 3300 dofs, the two methods produce quite similar global energy error. In general, the two different refinement strategies are both able to refine sharply around the origin where the singularity appears.

In Figure 46 we have displayed the root-mean square of the exact element error (in % measuring the deviation from uniform error distribution) versus n_{dof} obtained for uniform refinement using B-splines and adaptive refinement using LR B-splines. We see immediately that for the uniform refinement using B-splines the root mean square for the error distribution increases with number of uniform refinements. This is as expected as the error in the vicinity of the singularity point will be more and more dominant with uniform refinement. The highly non-uniform error distribution is consistent with the observed reduced convergence order in Figure 42. For the adaptive refined grids we see that for $p = 2$ the *rms* of the error distribution reduces in the first refinement steps and than becomes more or less constant $rms = 5 - 8 \cdot 10^{-1}$. As discussed in the paragraph above the lack of sufficient refinement around the singularity point prevent the *rms* to approach to zero. However, the values obtained may be classified as quasi-uniform, i.e. the *rms* is bounded when increasing n_{dof}. For $p = 3$ we see that the *rms*-values are a bit higher (around 1) and more differences between the different local refinement strategies. The results for $p = 4$ is even more spread and in particular for $\beta = 20$ we observe that the *rms*-values are slightly non-decreasing. This is consistent with the observation made above, i.e. that for $p = 4$ we get noticeable higher error for $\beta = 20$ than for $\beta = 5$, see Figure 42 f). Notice that the local refinement strategy *full span* have the lowest *rms*-value in all cases! Furthermore, that low multiplicity (i.e. $m = 1$) gives lower *rms*-values for *full span* than for high multiplicity, but this is not always the case for the *structured mesh* method.



(a) $p = 2,\ \beta = 5$

(b) $p = 2,\ \beta = 20$

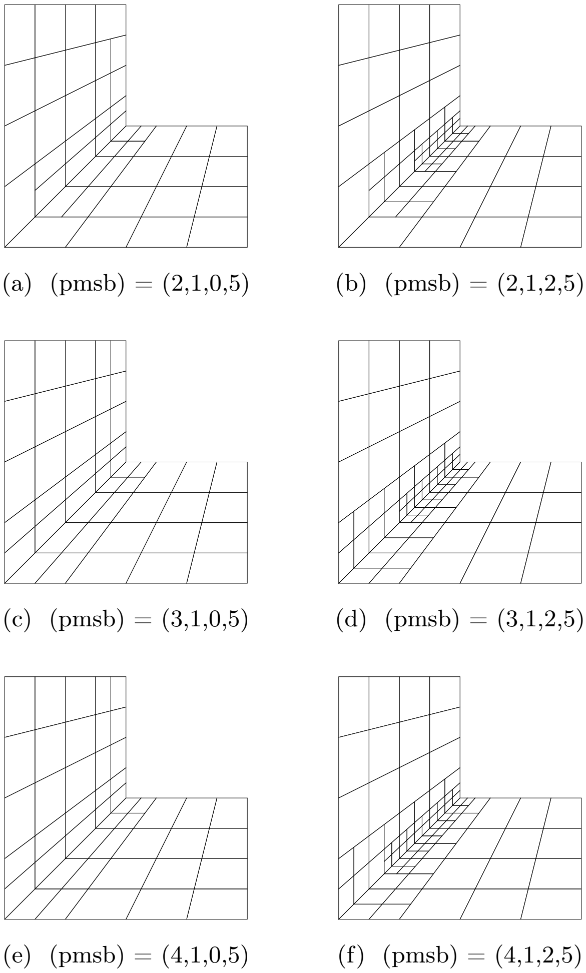
(c) $p = 3,\ \beta = 5$

(d) $p = 3,\ \beta = 20$

(e) $p = 4,\ \beta = 5$

(f) $p = 4,\ \beta = 20$

Figure 42: The L-shape problem: Relative global errors (in %) (measured in the a-norm) versus n_{dof} obtained for uniform refinement using B-splines and adaptive refinements using LR B-splines. The dotted lines are the sub-optimal convergence rate $\mathcal{O}(n_{\text{dof}}^{-1/3})$ valid for (quasi) uniform refinement and the optimal asymptotic convergence rates (valid for (quasi) uniform element error distribution) $\mathcal{O}(n_{\text{dof}}^{-1})$, $\mathcal{O}(n_{\text{dof}}^{-3/2})$, $\mathcal{O}(n_{\text{dof}}^{-2})$ for $p = \{2, 3, 4\}$, respectively.



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

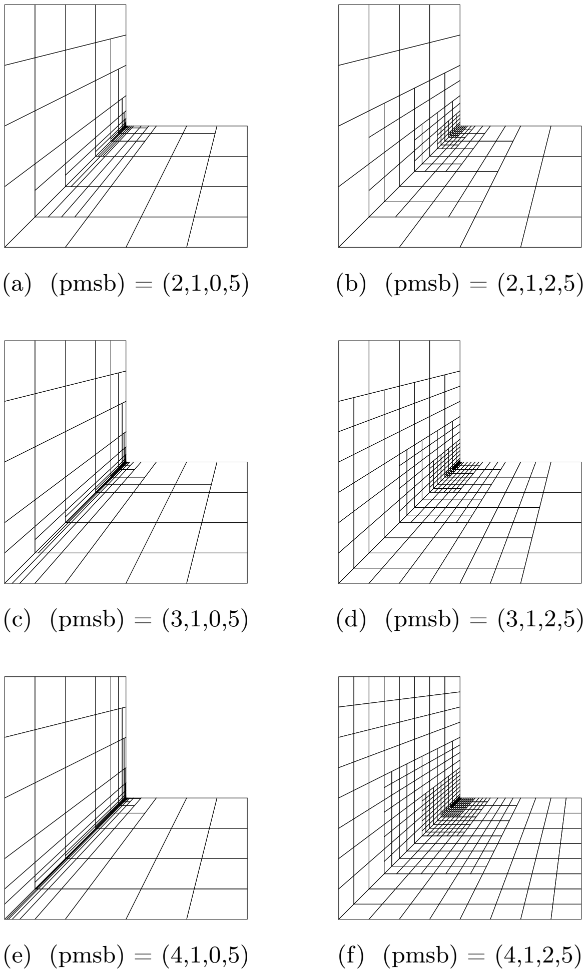
(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 43: The L-shape problem: The 3rd adaptively refined grid $\mathcal{M}_3$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (notice that β is denoted b in the subfigure captions).



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

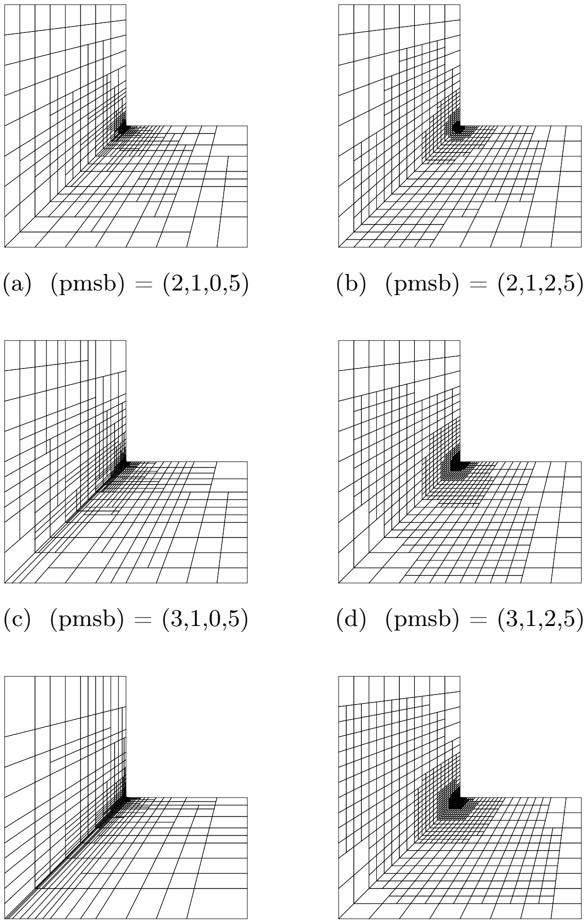
(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 44: The L-shape problem: The 12th adaptively refined grid $\mathcal{M}_{12}$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (notice that β is denoted b in the subfigure captions).



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

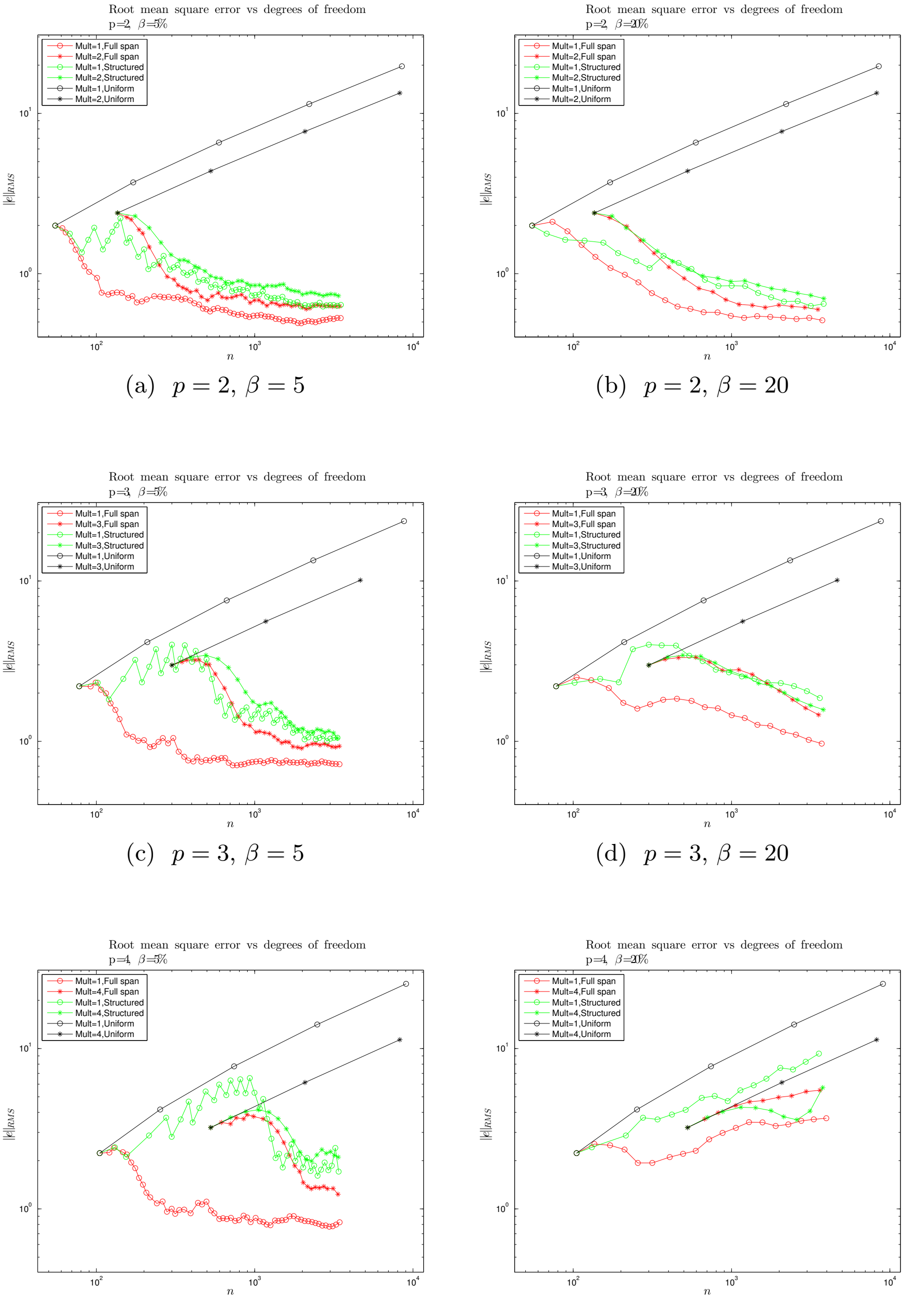
(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 45: The L-shape problem: The final adaptively refined grid $\mathcal{M}_n$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (notice that β is denoted b in the subfigure captions).



(a) $p = 2,\ \beta = 5$

(b) $p = 2,\ \beta = 20$

(c) $p = 3,\ \beta = 5$

(d) $p = 3,\ \beta = 20$

(e) $p = 4,\ \beta = 5$

(f) $p = 4,\ \beta = 20$

Figure 46: The L-shape problem: Root-mean square of exact element error (in %) (measuring the deviation from uniform error distribution) versus n_{dof} obtained for uniform refinement using B-splines and adaptive refinement using LR B-splines. Results displayed for $p = \{2, 3, 4\}$ (from top to bottom) and $\beta = \{5, 20\}$ (left to right).

5.4 Interior layer

Problem definition

The next test problem is a Poisson problem on a unit square with a sharp interior layer due to a highly varying right hand side (volumetric forcing). The problem is given as

$$\begin{aligned} \nabla^2 u &= f(x,y) && \text{in} && \Omega \\ u &= u_D(x,y) && \text{on} && \partial\Omega_D \\ \tfrac{\partial u}{\partial n} &= g(x,y) && \text{on} && \partial\Omega_N \end{aligned} \tag{41}$$

and has an exact solution given by

$$u(x,y) = \arctan\left(S(\sqrt{(x-1.25)^2+(y+0.25)^2} - \frac{\pi}{3})\right). \tag{42}$$

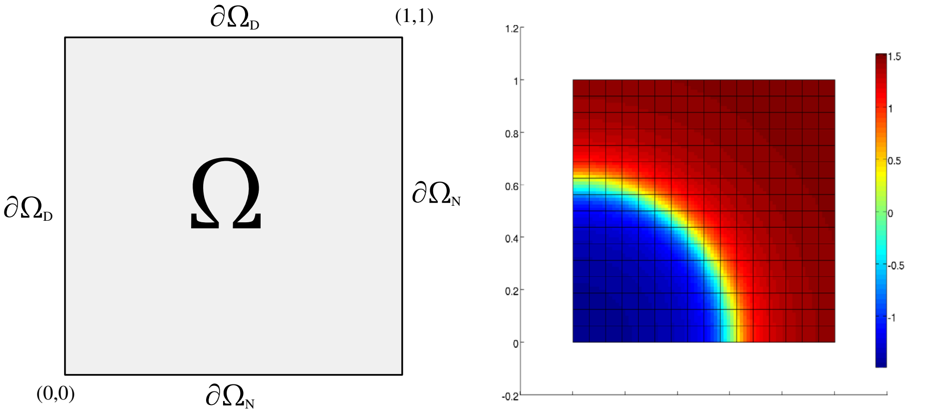
Note that the right hand side $f(x,y)$ is generated by taking the Laplacian (∇^2) of the analytical solution given in Equation (42) and similarly $g(x,y)$ is found by taking normal derivative, i.e. $\frac{\partial u}{\partial n}$, of the analytical solution. The analytical solution depicted in Figure 47b displays a "front"-type of behavior where the solution is rapidly changing across a circular band through the domain. This problem is mathematically smooth i.e. $u \in H^{p+1}(\Omega)$ for any finite p. However, due to the highly varying right hand side we may only expect optimal convergence order when the element size h is less than a given threshold that depends on the sharpness/bandwith of the interior layer. Hence, we may expect suboptimal convergence rate for uniform mesh refinement when the mesh is not fine enough.

In Figure 48 we see that the convergence for uniform mesh refinement is limited by the low regularity of the right hand side, i.e. the convergence rate is equal to $-q/2 = -1/2$. However, we see that for refined grids with small enough element size $h \lesssim 1/40$ (i.e. $n_{\text{dof}} \gtrsim 1600$) we obtain optimal convergence order. Thus, our refinement goal is here to resolve the interior layer as adequately as possible in order to obtain optimal convergence order, in an adaptive grid refinement process towards a global solution of a certain accuracy measured in the a-norm.

Results

In Figure 48 we show the results obtained by adaptive refinement using LR B-splines. The results are displayed using different polynomial order $p = \{2,3,4\}$, portion of refinement $\beta = \{5,10,20\}$, multiplicity $m = \{0, p-1\}$ and local refinement strategy.

The main observation is that we achieve optimal convergence rate, after some refinements, i.e. $-q/2 = -p/2 = \{-1, -3/2, -2\}$ for $p = \{2,3,4\}$,



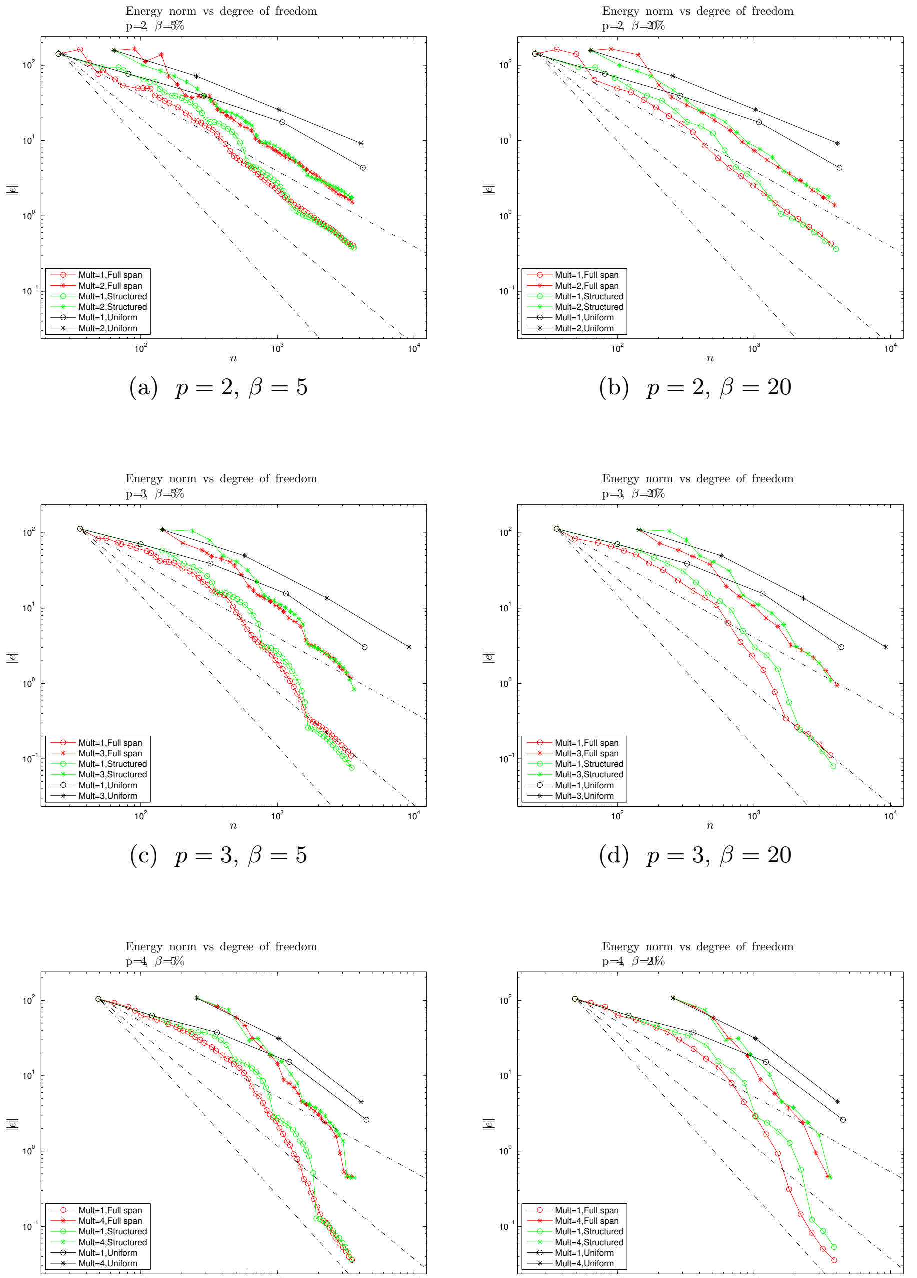
(a) The domain Ω and the boundary conditions (b) The exact solution, see Equation (42)

Figure 47: The Interior Layer problem: A Poisson problem with rough right hand side.

respectively. However, we see that for high polynomial order ($p = \{3, 4\}$) we need more refinements than for low order to obtain optimal convergence rate. Furthermore, for $p = 4$ we observe some “extra” refinement along the Dirichlet boundary due to the fact that $u \notin S^h$. Compared to uniform refinement the errors for adaptive refined meshes using LR B-splines are about 10 times lower. The sharper the interior layer, the more pronounced this error difference will become.

We clearly see that high regularity (i.e. low multiplicity) is efficient when we compare relative error versus number of degrees of freedom. Furthermore, we see that the convergence results are not sensitive to variation of β, whereas local refinement strategies have some influence for high polynomial order.

The resulting grids are different for the different local refinement strategies. This is illustrated in the Figures 49–51 where we have displayed the effect on the refined grid using different local refinement strategies. As seen, the LR B-splines makes it possible to refine sharply in the vicinity of the interior layer. Furthermore, we may see from the two Figures 49 and 50 that the *full span* method have more elements with high aspect ratios than the *structured mesh* method, whereas the latter one gives a more widespread uniform refinement on subdomains along the interior layer. In particular, the differences are pronounced at the $\mathcal{M}_3$ grid. However, at about $n_{dof} =$ 3000 the two methods produce quite similar global energy error. Notice, that the final grid for $p = 4$ shows extra refinement along the Dirichlet part of the boundary due to the fact that the inhomogeneous Dirichlet boundary conditions are approximated.



(a) $p = 2,\ \beta = 5$

(b) $p = 2,\ \beta = 20$

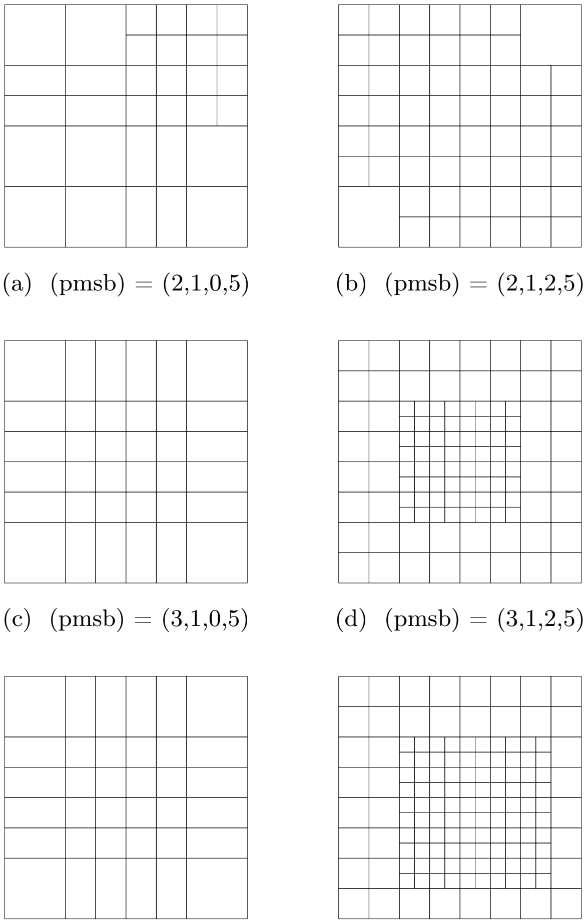
(c) $p = 3,\ \beta = 5$

(d) $p = 3,\ \beta = 20$

(e) $p = 4,\ \beta = 5$

(f) $p = 4,\ \beta = 20$

Figure 48: The Interior Layer problem: Relative global errors (in %) (measured in the a-norm) versus n_{dof} obtained for uniform refinement using B-splines and adaptive refinements using LR B-splines. The dotted lines are the optimal asymptotic convergence rates (valid for (quasi) uniform element error distribution) $\mathcal{O}(n_{\text{dof}}^{-1})$, $\mathcal{O}(n_{\text{dof}}^{-3/2})$, $\mathcal{O}(n_{\text{dof}}^{-2})$ for $p = \{2, 3, 4\}$, respectively.



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

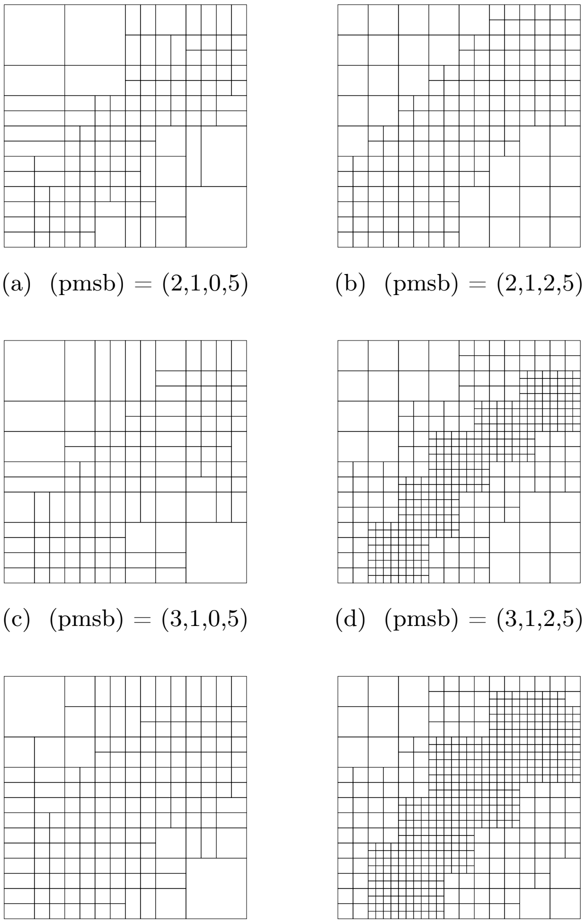
(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 49: The Interior Layer problem: The 3rd adaptively refined grid $\mathcal{M}_3$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, , 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (β is denoted b in the subfigure captions).



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

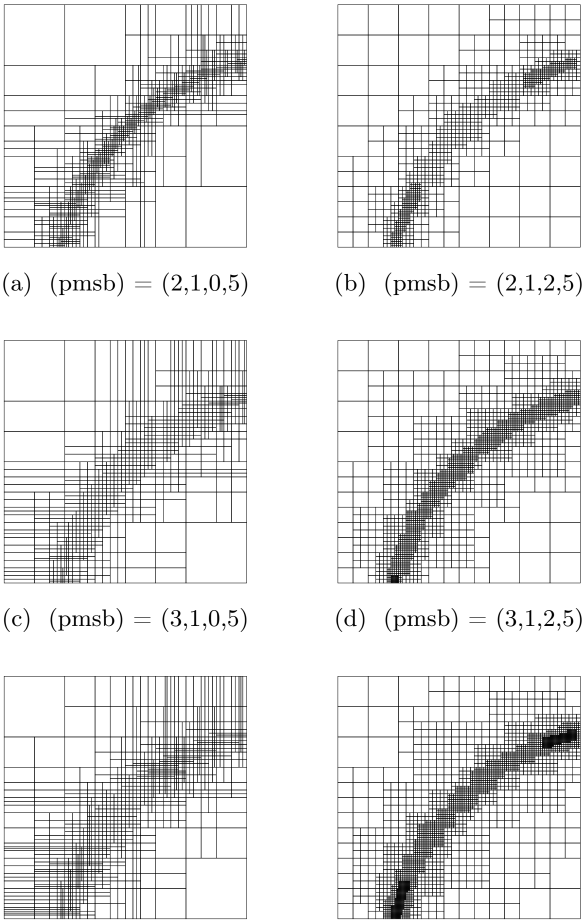
(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 50: The Interior Layer problem: The 12th adaptively refined grid $\mathcal{M}_{12}$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (β is denoted b in the subfigure captions).



(a) (pmsb) = (2,1,0,5)

(b) (pmsb) = (2,1,2,5)

(c) (pmsb) = (3,1,0,5)

(d) (pmsb) = (3,1,2,5)

(e) (pmsb) = (4,1,0,5)

(f) (pmsb) = (4,1,2,5)

Figure 51: The Interior Layer problem: The final adaptively refined grid $\mathcal{M}_n$ obtained by using LR B-splines with different polynomial degrees $p = \{2, 3, 4\}$ and local refinement strategies, but same multiplicity $m = 1$ and $\beta = 5$ (β is denoted b in the subfigure captions).

6 Conclusions

In this paper we have investigated adaptive refinement in isogeometric analysis using LR B-splines. Traditional tensor product B-splines lack the ability of local refinement which is needed in order to achieve optimal convergence order in real world applications. In particular, higher order isogeometric methods based on tensor product B-splines are not able to exploit the full potential offered by isogeometric analysis when applied to problems involving singularities or rough right hand sides.

Herein, the newly developed LR B-splines have been applied as adaptive refinement in isogeometric analysis. Different local refinement strategies has been proposed and implemented in the object oriented code *IFEM*.

We have performed an extensive set of numerical tests to investigate the performance of using LR B-splines to achieve accurate results measured in energy norm (a-norm) and optimal convergence rates for the classical benchmark tests L-shape and Interior Layer. The results are very good and we achieved optimal convergence rates for both test cases, and the sensitivity towards different choices of local refinement strategies and prescribed portion of refinement was moderate. Furthermore, high regularity gives less error versus degrees of freedoms compared to low regularity (for a given polynomial order) in all cases.

We conjecture that the application of the full span and structured mesh refinement strategies, both generates a subclass of LR B-splines that are linearly independent, omitting the need for linear independence testing. No linearly dependent mesh has been discovered while using these strategies. The proof for this is left as a topic for future investigation.

We think the LR B-splines may serve well as a framework for adaptive isogeometric methods as they are both versatile and manageable with regards to development of general purpose finite element software. The framework open new vistas for interoperable CAD and FEA systems, and more research and developments should be pursued to fully exploit these possibilities.

Acknowledgement

The authors acknowledge the financial support from the Norwegian Research Council and the industrial partners of the ICADA-project; Ceetron, Det Norske Veritas and Statoil. They also acknowledge the support from the other co-workers in the ICADA-project. We thank in particular Dr Mukesh Kumar for many fruitful discussions and suggestions that have improved the manuscript.

Bibliography

[1] M. Ainsworth and J. Oden. *A Posterori Error Estimation in Finite Element Analysis*. 1st. Wiley-Interscience, 2000.

[2] Y. Bazilevs et al. "Isogeometric analysis: approximation, stability and error estimates for h-refined meshes". In: *Mathematical Models and Methods in Applied Sciences* 16 (2006), pp. 1031–1090.

[3] Y. Bazilevs et al. "Isogeometric analysis using T-splines". In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 229–263.

[4] P. Bornemann and F. Cirak. "A subdivision-based implementation of the hierarchical B-spline finite element method". In: *Computer Methods in Applied Mechanics and Engineering* (2012).

[5] A. Buffa, D. Cho, and M. Kumar. "Characterization of T-splines with reduced continuity order on T-meshes". In: *Computer Methods in Applied Mechanics and Engineering* 201-204.0 (2012), pp. 112–126.

[6] T. Dokken, T. Lyche, and K. Pettersen. "Polynomial splines over locally refined box-partitions". In: *Comput. Aided Geom. Des.* 30.3 (Mar. 2013), pp. 331–356.

[7] M. R. Dörfel, B. Jüttler, and B. Simeon. "Adaptive isogeometric analysis by local h-refinement with T-splines". In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 264 –275.

[8] D. Forsey and R. Bartels. "Hierarchical B-spline refinement". In: *ACM SIGGRAPH Computer Graphics* 22.4 (1988), pp. 205–212.

[9] C. Giannelli, B. Jüttler, and H. Speleers. "THB-splines: The Truncated basis for hierarchical splines". In: *Computer Aided Geometric Design* 29.7 (2012), pp. 485 –498.

[10] T. Hughes. *The finite element method: linear static and dynamic finite element analysis*. Prentice-hall, 1987.

[11] T. Hughes, J. Cottrell, and Y. Bazilevs. “Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement”. In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195.

[12] T. Kvamsdal and K. Okstad. “Error estimation based on superconvergent patch recovery using statically admissible stress fields”. In: *International Journal for Numerical Methods in Engineering* 42.3 (1998), pp. 443–472.

[13] X. Li and M. Scott. *On the Nesting Behavior of T-splines.* Tech. rep. University of Texas at Austin, 2011.

[14] X. Li et al. “On linear independence of T-spline blending functions”. In: *Comput. Aided Geom. Des.* 29.1 (Jan. 2012), pp. 63–76.

[15] B. Mourrain. “On the dimension of spline spaces on planar T-subdivisions”. In: *Arxiv preprint arXiv:1011.1752* (Nov. 2010).

[16] D. Schillinger and E. Rank. “An unfitted hp-adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry”. In: *Computer Methods in Applied Mechanics and Engineering* 200.47 - 48 (2011), pp. 3358 –3380.

[17] D. Schillinger et al. “An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces”. In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012), pp. 116 –150.

[18] M. Scott et al. “Isogeometric finite element data structures based on Bézier extraction of T-splines”. In: *International Journal for Numerical Methods in Engineering* 88.2 (2011), pp. 126–156.

[19] M. Scott et al. “Local refinement of analysis-suitable T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 213 (2012), pp. 206–222.

[20] T. Sederberg et al. “T-splines and T-NURCCs”. In: *ACM Transactions on Graphics* 22.3 (2003), pp. 477–484.

[21] A. Vuong et al. “A hierarchical approach to adaptive local refinement in isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 200.49-52 (2011), pp. 3554–3567.

[22] P. Wang et al. “Adaptive isogeometric analysis using rational PHT-splines”. In: *Computer-Aided Design* 43 (2011), pp. 1438–1448.

Paper II

On the similarities and differences between Classical Hierarchical, Truncated Hierarchical and LR B-splines

Kjetil André Johannessen, Filippo Remonato and Trond Kvamsdal

Submitted to *Computer Methods in Applied Mechanics and Engineering*, 2014

On the similarities and differences between Classical Hierarchical, Truncated Hierarchical and LR B-splines

Kjetil André Johannessen, Filippo Remonato and Trond Kvamsdal

Department of Mathematical Sciences
Norwegian University of Science and Technology, Trondheim, Norway
e–mail: Kjetil.Johannessen@math.ntnu.no, Filippo.Remonato@math.ntnu.no,
Trond.Kvamsdal@math.ntnu.no

Abstract

Smooth spline functions such as B-splines and NURBS are already an established technology in the field of computer-aided design (CAD) and have in recent years been given a lot of attention from the computer-aided engineering (CAE) community. The advantages of local refinement are obvious for anyone working in either field, and as such, several approaches have been proposed. Among others, we find the three strategies *Classical Hierarchical B-splines*, *Truncated Hierarchical B-splines* and *Locally Refined B-splines*. We will in this paper present these three frameworks and highlight similarities and differences between them. In particular, we will look at the function space they span and the support of the basis functions. We will then analyse the corresponding stiffness and mass matrices in terms of sparsity patterns and conditioning numbers. We show that the basis in general do not span the same space, and that conditioning numbers are comparable. Moreover we show that the weighting needed by the Classical Hierarchical basis to maintain partition of unity has significant implications on the conditioning numbers.

1 Introduction

1.1 Background

Computer Aided Design (CAD) and Finite Element Analysis (FEA) are essential technologies in modern product development. However, the interoperability of these technologies is severely disturbed by differences in the mathematical approaches used. The main reason for inconsistencies is that the technologies evolved in different communities with the focus on improving disjoint stages in product development processes, and taking little heed on relations to other stages. Efficient feedback from analysis to CAD and refinement of the analysis model are essential for computer-based design optimization and virtual product development. The current lack of efficient interoperability of CAD and FEA makes refinement and adaptation of the analysis model cumbersome, slow and expensive.

In any finite element analysis of real world problems, it is of great importance that the quality of the computed solution may be determined. Furthermore, numerical simulation of many industrial problems in civil, mechanical and naval industry often require large computational resources. It is therefore of utmost importance that computational resources are used as efficiently as possible to make new results readily available and to expand the realm of which processes may be simulated. We thus identify reliability and efficiency as two challenges in simulation based engineering.

These two challenges may be addressed by a posteriori error estimation combined with adaptive refinements. A lot of research has been performed on error estimation and adaptive mesh refinement, see e.g. (Ainsworth and Oden, 2000 [1]) for an excellent overview. The senior author of the present paper have been working with error estimation and adaptivity for more than two decades (see e.g. [28], [35], [34], and [30]), and are well aware of the fact that adaptive methods are not yet an industrial tool, partly because the need for a link between the finite element program and traditional CAD-system. Here, the use of an isogeometric analysis framework may facilitate more widespread adoption of this technology in industry, as adaptive mesh refinement does not require any further communication with the CAD system.

The new paradigm of Isogeometric Analysis, which was introduced by Hughes *et al.* [21] (see also [10]), demonstrates that for smooth (enough) problems much is to be gained with respect to efficiency, quality and accuracy in analysis by replacing traditional Finite Elements by volumetric NURBS elements. However, a fundamental constraint of traditional NURBS is that they are (global) tensor product and lack the potential for local refinement. The need for local refinement has always been an issue, and several proposed solutions to local refinement has been derived such as T-splines [41],[40], Hierarichal B-splines [14],[26], Truncated Hierarichal B-splines [18] and Locally Refined B-splines [12]. The use of these techniques in CAD allows for more freedom since it is often enough that a forward mapping exist and can be efficiently manipulated or evaluated, and some conversion algorithms are available [48]. With their applications into isogeometric analysis [2], [3], [13], [38], [4], [45], [44], [47], [33], [46], [5], [37], [22], [23], [27], and [42] came new requirements of the basis. Linear independence [9], [8], [39], [43], [29], [19], [7], stability [17] and partition of unity [18] became center topics of research as isogeometric analysis is impractical or sometimes impossible without these properties. The research is ongoing for most of these basis and the community has yet to settle on a single technology which encompass every desired property without restrictions on the mesh.

With many different technologies addressing the same problem of local refinement, it is to be expected that there is overlap. We hope to shed some light on this topic by presenting some of these spline families, highlighting similarities and differences between them. In particular we will be looking at the Classical Hierarchical, the Truncated Hierarchical and the Locally Refined B-splines.

1.2 Aim and outline of the paper

The aim of this paper is to present several different local refinement strategies that currently exist. We will emphasize differences in both their mathematical and numerical properties.

The paper is organized as follows:

In Section 2, we give a brief introduction to the approximation theory. We describe the finite element method and least squares method with focus on derivation of the two matrices we will be looking at: the stiffness matrix A and the L_2 projection matrix M.

In Section 3, we define Hierarichal B-splines, Truncated Hierarichal B-splines and Locally Refined (LR) B-splines. Our aim is to provide a common framework and notation to better highlight their particular properties.

Numerical experiments are performed in Section 4. As our main measurement, we will be looking at conditioning number and sparsity pattern of the system matrices over several illustrative meshes.

In Section 5 we present some interesting observations and preliminary results for which more research is required.

We end this paper concluding upon our findings in Section 6.

2 Finite Element Theory

In this section we give a very short introduction to the finite element theory, with the only aim of presenting the quantities we will be using as our performance indicators. A thorough explanation of the theory behind finite elements can be found in many sources like [24, 36, 6, 20].

One of the first steps when applying a finite element method to a problem, is to derive its so called *variational formulation* and write the problem in a structure like: Find $u \in \mathcal{V}$ such that

$$a(u, v) = l(v) \quad \forall v \in \mathcal{V} \tag{1}$$

where $\mathcal{V}$ is a Hilber space, $a(\cdot\,,\cdot)$ is a continuous bilinear forms, and $l(v)$ is a continuous functional on the dual space of $\mathcal{V}$. The problem is also normally associated with some type of conditions, like boundary or initial conditions. The existence and uniqueness of solutions is then guaranteed by the Lax-Milgram theorem.

The Galerkin approach to this kind of problems consists of producing a finite-dimensional approximation $\mathcal{V}_h$ of the infinite-dimensional function space $\mathcal{V}$, and search for solutions $u_h \in \mathcal{V}_h$. Specifically, we have $\mathcal{V}_h \subset \mathcal{V}$, and the problem reads: Find $u_h \in \mathcal{V}_h$ such that

$$a(u_h, v_h) = l(v_h) \quad \forall v_h \in \mathcal{V}_h\,. \tag{2}$$

The space $\mathcal{V}_h$ is defined to be the span of selected basis functions $\{\varphi_1, \varphi_2, \ldots \varphi_n\}$. In the Isogeometric setting, these functions are chosen to be spline functions, for which we use the notation $\{B_1, B_2 \ldots B_n\}$.

We now give some classical examples to introduce the stiffness and mass matrices.

2.1 Poisson equation

Poisson equation is the classical model problem for Elliptic PDEs. It arises in several engineering problems like elastic membranes or magnetic fields and also appears as an important part of more complicated problems like Navier-Stokes. Given a domain $\Omega \subset \mathbb{R}^2$ and a continuous function $f : \Omega \to \mathbb{R}$, we want to find a function $u : \Omega \to \mathbb{R}$ such that

$$-\Delta u = f \quad \text{in } \Omega \tag{3}$$

and satisfies certain prescribed conditions on the boundary of the domain $\partial\Omega$. We typically have two types of boundary conditions, namely Dirichlet:

$$u = \bar{u} \text{ at } \partial\Omega_D \tag{4}$$

and Neumann:

$$\frac{\partial u}{\partial n} = \bar{t} \text{ at } \partial\Omega_N \tag{5}$$

Here $\bar{u}$ is prescribed boundary value of the unknown u along $\partial\Omega$, $\bar{t}$ the prescribed (Neumann) flux along $\partial\Omega$ and $\partial u/\partial n$ is the normal derivative of u, i.e. the directional derivative respect to the outward normal vector n. Furthermore, we assume $\partial\Omega_D \cup \partial\Omega_N = \Omega$ and $\partial\Omega_D \cap \partial\Omega_N = \emptyset$. In case of pure Neumann problem we introduce the following notation: $\Gamma = \partial\Omega_N = \partial\Omega$

We now multiply by a test function $v \in \mathcal{V}$ and integrate over the domain, and write (3) as

$$\int_\Omega \Delta u\, v \,\mathrm{d}A = \int_\Omega f\, v \,\mathrm{d}A \quad \forall v \in \mathcal{V}\,.$$

Using Green's formula and pure Neumann boundary condition we can rewrite the problem as: Find $u \in \mathcal{V}$ such that

$$\int_\Omega \nabla u\, \nabla v \,\mathrm{d}A = \int_\Omega f\, v \,\mathrm{d}A + \int_\Gamma \bar{t}\, v \,\mathrm{d}S \quad \forall v \in \mathcal{V} \tag{6}$$

The problem written as in (6) is the variational formulation for the Poisson's equation.

We now apply Galerkin's approach and choose $\mathcal{V}_h \subset \mathcal{V}$ where $\mathcal{V}_h = \mathrm{span}\{B_1, B_2, \ldots B_n\}$. Any $u_h \in \mathcal{V}_h$ can then be written as a linear combination of the basis functions

$$u_h = \sum_{j=1}^{n} B_j\, u_j$$

with $u_j \in \mathbb{R}$. Substituting this into (6) and systematically selecting $v = B_i, i = 1, \ldots, n$ allows us to write

$$\sum_{j=1}^{n} \int_\Omega \nabla B_i\, \nabla B_j \,\mathrm{d}A \;\, u_j = \int_\Omega f\, B_i \,\mathrm{d}A + \int_\Omega \bar{t} B_i \mathrm{d}A \quad \forall i = 1, \ldots, n$$

which is simply a linear system of equation of the form

$$A\boldsymbol{u} = \boldsymbol{b}$$

where

$$A_{i,j} = \int_\Omega \nabla B_i\, \nabla B_j \,\mathrm{d}A \tag{7}$$

$$\boldsymbol{u} = [u_1, u_2, \ldots u_n]^T$$

$$b_i = \int_\Omega f\, B_i \,\mathrm{d}A + \int_\Gamma \bar{t}\, B_i \,\mathrm{d}S\,. \tag{8}$$

Due to historical reasons, the matrix A is called *Stiffness Matrix* and the vector $\boldsymbol{b}$ is called *Load Vector*, a nomenclature common in structural/solid mechanics for which the Finite Element method was developed in the late 50's.

2.2 Least Squares fitting

Performing a least-square fit of a surface is often encountered as a geometrical problem. In this case, given a smooth continuous function $f : \Omega \to \mathbb{R}$, we are searching for a function $u_h \in \mathcal{V}_h$ such that $\|u_h - f\|_{L^2}$ is as small as possible. It is possible to show that the solution u_h is the L^2-projection of f and

$$u_h = \arg\min_{u \in \mathcal{V}_h} \|u - f\|_{L^2} \iff \int_\Omega u_h \, v_h \, \mathrm{d}A = \int_\Omega f \, v_h \, \mathrm{d}A \quad \forall v_h \in \mathcal{V}_h \,.$$

Applying the same procedure as in the Poisson's equation case, we can write the problem as a solution of a linear system of equations

$$M\boldsymbol{u} = \boldsymbol{b}$$

where now the matrix M is called *Mass Matrix* and is defined as

$$M_{i,j} = \int_\Omega B_i \, B_j \, \mathrm{d}A \,. \tag{9}$$

The load vector $\boldsymbol{b}$ is given by

$$b_i = \int_\Omega f \, B_i \, \mathrm{d}A \,.$$

2.3 Helmholtz equation

Helmholtz equation often arises in problems involving waves; in particular it is the time-independent version of the wave equation and is written as

$$\Delta u + k^2 \, u = f \,. \tag{10}$$

The solution u of Helmholtz equation represents the amplitude configuration of the wave in space, and k is the wavenumber.

It is easy now to see that the first term in (10) is the Laplacian operator we already encountered in the Poisson equation, while the second term is, besides the constant k, the same as in the least-squares fitting problem. Applying the same procedure as in the previous cases we are therefore able to rewrite the problem as searching for solutions of the linear system of equations

$$A\boldsymbol{u} + k^2 M\boldsymbol{u} = \boldsymbol{b} \Rightarrow \left(A + k^2 M\right) \boldsymbol{u} = \boldsymbol{b} \,.$$

As we have seen with the above examples, the matrices A and M play an important role in the solution of partial differential equations using a Galerkin approach. Simple elliptic problems may use only the stiffness matrix A; simple geometrical problems may use only the mass matrix M; while more complex or time-dependant problems use both.

The space $\mathcal{V}_h$ can be defined using several different types of basis functions. This choice is of great importance as it will dictate the properties of the solution space. Different types of basis functions will yield different system matrices and consequently this will affect the convergence rate of the numerical methods for solving linear systems of algebraic equations. For this reason, we will herein investigate the impact different classes of spline bases functions (presented in Section 3 below) have on important properties of these matrices as conditioning number, bandwidth, and sparsity.

3 Spline functions

In this section we present the theory of the three classes of spline functions considered in this paper: Classical Hierarchical, Truncated Hierarchical and LR B-splines. An effort has been made in order to unify the different concepts under a common framework of notations, to ease both the understanding and the comparison of the different technologies. We have included only the essentials and we refer the interested reader to the papers on which we based our studies for an in-depth introduction and details [12, 18, 22, 46].

3.1 Notation and common definitions

The Hierarchical (both Classical and Truncated) and the LR B-splines methodologies use quite different points of view when considering meshes and refinements. As such, different notations have been developed in the corresponding publications. We will in this paper use the following notation when we will need to address mesh-related quantities:

- ϵ for meshlines;
- Ω for domains, i.e. regions of the mesh (excluding mesh lines);
- V for full tensor product meshes;
- $\mathcal{M}$ for general meshes.

In particular, the Hierarchical setting focuses more on regions of the mesh and their underlying full tensor product meshes. For these reasons, Ω and V are often used in this context. The LR B-splines setting instead focuses more on meshlines and meshes as a whole. To provide a formal description of these different point of views we can write that a mesh $\mathcal{M}$ is seen as

$$\begin{aligned} \mathcal{M} &= \bigcup_l (\Omega^l \cap V^l) && \text{in the Hierarchical setting} \\ \mathcal{M} &= \bigcup_i \epsilon_i && \text{in the LR B-splines setting} \end{aligned}$$

where the index l denotes the Hierarchical level and i runs over all meshlines.

The notation we will use for basis functions is the following:

- $N \in \mathcal{N}$ for *uniform* (in the index domain) tensor product basis functions.
- $B \in \mathcal{B}$ for tensor product basis functions (possibly non-uniform).
- $H \in \mathcal{H}$ for Classical Hierarchical basis functions.

- $T \in \mathcal{T}$ for Truncated Hierarchical basis functions.
- $L \in \mathcal{L}$ for LR B-splines basis functions.

Of course, there exist cases where for some indices $N_i = B_i = H_i = T_i = L_i$, but we hope the different notation will ease the understanding of the technologies.

We have from elementary spline theory that a *knot vector* is a nondecreasing sequence of coordinates in the parameter space of the form $\Xi = [\xi_1, \xi_2, \dots, \xi_{n+p+1}]$, where each $\xi_i \in \mathbb{R}$ is called a *knot.* If the knot values are equidistant the knot vector is called *uniform*, and *non-uniform* otherwise. If the first and last knots have multiplicity $p+1$, the knot vector is called *open.* A knot vector comprising of $n+p+1$ knot values will generate n univariate linearly independent basis functions of degree p. We will focus our analysis on B-splines built from uniform, non-open knot vectors.

Corresponding to Ξ, we have the *index domain* $I = [1, \dots, n+p+1]$. The index domain is useful for considering non-uniform knots and also determine the support of functions. For uniform knot vectors Ξ we have $\Xi = \gamma I$ for some scaling factor $\gamma \in \mathbb{R}$, and this is what we will be working with in our examples. We would however like to stress that it is possible to generalize the same numerical tests using open or non-uniform knot vectors. For bivariate meshes, we consider the index domain to be the finest level tensor mesh, i.e. V^M, see Figure 3 for an example.

Definition 1. Given a knot vector $\Xi = [\xi_1, \xi_2, \dots, \xi_{n+p+1}]$ and a polynomial degree p, the n univariate basis functions $B_{1,p}, \dots, B_{n,p}$ are recursively defined in the following way:

$p = 0$:

$$B_{i,0}(\xi) = \begin{cases} 1 & \text{for } \xi_i \leqslant \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

$p > 0$:

$$B_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi) \tag{12}$$

The above definition is known as the *Cox-de Boor recursion formula.* From this definition it follows that each basis function depends only on $p+2$ knot values. For instance, for $p = 2$ the knot vectors $\Xi = [0, 1, 2, 3, 4, 5]$ will generate three basis functions corresponding to the *local* knot vectors $\Xi_1 = [0, 1, 2, 3]$, $\Xi_2 = [1, 2, 3, 4]$, and $\Xi_3 = [2, 3, 4, 5]$. Due to this, we will often refer to the basis functions using their local knot vectors. The notation will also be adjusted on a case-to-case basis depending on what we need to emphasize, and we will use $B_i = B_{\Xi_i}$ to keep track of the local knot vector on which the function is built.

From the univariate basis functions it is possible to define multivariate functions using the tensor product structure of B-splines:

Definition 2. A d-variate B-spline $B(\boldsymbol{\xi})$ of degrees $\boldsymbol{p} = [p_1, p_2, \dots, p_d]$ is a separable function $B : \mathbb{R}^d \longrightarrow \mathbb{R}$ defined as:

$$B_{\boldsymbol{\Xi}}(\boldsymbol{\xi}) = \prod_{i=1}^{d} B_{\Xi_i}(\xi_i)$$

where $\Xi_i \in \mathbb{R}^{p_i+1}$ is the local knot vector for the univariate basis function of degree p_i along the i-th parametric dimension.

Note that the polynomial degree is implicitly defined by the number of knots in the local knot vectors. In the bivariate setting, it is customary to denote the two parametric coordinates as ξ and η, and the corresponding polynomial orders as p and q. We denote a *tensor product basis* $\mathcal{B} = \{B_1, B_2, \dots, B_n\}$ as a basis of functions defined by taking a tensor product of 1D basis functions.

In the following we will construct the mesh such that the actual domain will be the unit square, i.e. the initial tensor product basis will form a partition of unity on $\Omega_0 = [0,1] \times [0,1]$. Since we will use uniform knot vectors, we will need to extend the mesh beyond Ω_0 trough the use of a *ghost domain* G. In this way the full parametric domain will be given by $G \cup \Omega_0$.

We will represent bivariate basis functions on the same plot through the use of anchors. A common choice for the coordinates of the anchor are the *Greville abscissae*. The Greville abscissae $(\bar{\xi}, \bar{\eta})$ corresponding to a basis function are defined as

$$\bar{\xi} = \frac{1}{p} \sum_{j=2}^{p+1} \xi_j, \quad \bar{\eta} = \frac{1}{q} \sum_{j=2}^{q+1} \eta_j \tag{13}$$

where ξ_j and η_j are the knot values in the local knot vector. One drawback of Greville abscissae, however, is that they work nicely only if the function under consideration has a rectangular support, like normal B-splines. As we will see, it is very natural for Truncated Hierarchical B-splines to have non-rectangular support. In this case we will use *area-averaged coordinates* defined as

$$\bar{\xi} = \frac{\sum\limits_{i:E_i \in \text{supp}T} A_i \, \xi_i^c}{\sum\limits_{i:E_i \in \text{supp}T} A_i}, \quad \bar{\eta} = \frac{\sum\limits_{i:E_i \in \text{supp}T} A_i \, \eta_i^c}{\sum\limits_{i:E_i \in \text{supp}T} A_i} \tag{14}$$

where the sum runs over all elements E_i in the support of the function, (ξ_i^c, η_i^c) are the coordinates of the centre of the element E_i and A_i is its area. The weighting by the area ensures that each element contributes in the right way to the resulting coordinate of the anchor. This method of calculating the coordinates returns the same result as the normal Greville abscissae in the case of rectangular support, while it allows to see the difference when the support is non-rectangular.

3.2 Hierarchical B-Splines

The application of the Hierarchical framework in Isogeometric Analysis is very well explained by Vuong et al. in [46], and Giannelli et al. in [18]. We will look at how an admissible mesh is constructed, and how the construction procedure defines a sequence of nested bounded domains linked to the different Hierarchical levels.

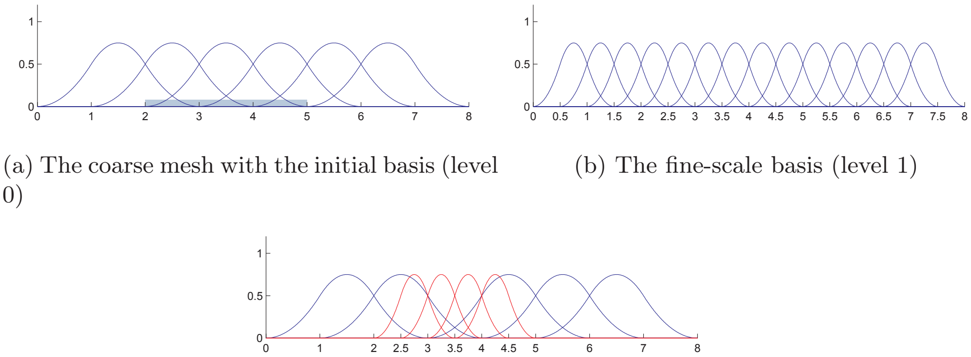
Introduction and general idea

The basic idea underlying Hierarchical B-splines is very simple, yet results in a good and flexible method to locally refine the mesh. A one-dimensional example for quadratic basis functions is illustrated in Figure 1: One portion of the initial level 0 mesh is selected for refinement. The coarse basis functions contained in that area are substituted by finer basis functions, and we thus obtain the Hierarchical basis.

As we can see from Figure 1 this basis does not constitute a partition of unity. This property is however easily recovered by weighting the functions appropriately (see Figure 2). In this case it is important to note that the region selected for refinement must contain at least the support of one coarse basis function. If this condition is not met, the weights associated with the finer basis functions will all be zero, and therefore no change will happen in the basis [46].

Figure 3 presents the Hierarchical approach on a simple 2D example with biquadratic basis functions. When a selected area of the mesh is refined, the knot spans are halved in each direction and this introduces one new level in the hierarchy. The basis functions from the previous level that are contained in the refined region are then substituted by the corresponding finer basis functions defined on the new knot spans. As in the univariate case, an appropriate weighting of the functions can be used to recover the partition of unity.

In the following we will focus mainly on the two dimensional case, and several examples will be presented.



(a) The coarse mesh with the initial basis (level 0)

(b) The fine-scale basis (level 1)

(c) The Hierarchical basis, constructed as a suitable combination of the coarse and fine scale basis functions

Figure 1: **Hierarchical Basis:** Construction of a univariate basis using quadratic basis functions. The highlighted area is selected for refinement, and the coarse functions contained therein are substituted by finer basis functions.

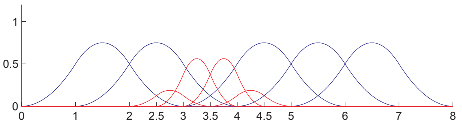


Figure 2: **Hierarchical Basis:** Weighting the fine scale functions appropriately ensures partition of unity for $\xi \in [2, 6]$.

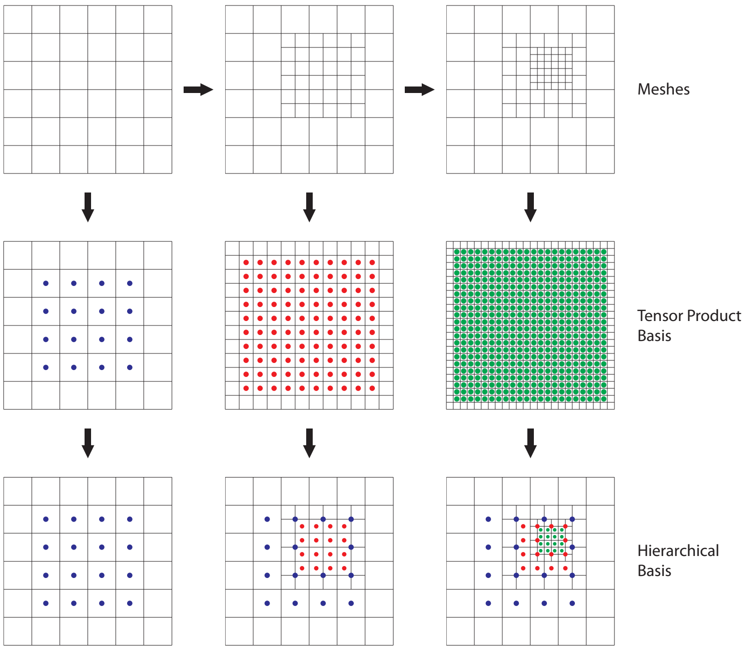


Figure 3: **2D Hierarchical Basis:** Example using biquadratic basis functions. **Upper row**: A two-step refinement is applied to the initial mesh displaying Ω^0, Ω^1 and Ω^2. **Middle row**: The tensor-product basis defined on the finest knot span available showing the functions $\mathcal{N}^l$ on the mesh V^l for all levels $l = 0, 1, 2$. From here we select the appropriate basis functions to include in the Hierarchical basis. **Lower row**: The actual Hierarchical basis defined on the refined mesh $\mathcal{M}$ above. At each step, the basis functions from the previous level that are contained in the refined region are substituted by the finer ones, showing the Hierarchical basis $\mathcal{H}$. The anchors are positioned using Greville Abscissae.

The Classical Hierarchical Basis

The construction of the mesh on which the Hierarchical basis is defined is a direct application of the idea presented above: starting from an initial, tensor product mesh V^0, some areas are selected for refinement. Once the areas have been selected, several new meshlines are introduced, halving the knot spans of the local knot vectors of all the functions contained therein.

A smart choice when it comes to substituting coarse basis functions with finer ones is exploiting the subdivision property of B-splines. A univariate B-spline N_Ξ defined on the knot vector $\Xi = [\xi_1, \xi_2 \dots \xi_{p+2}]$ can be expressed as a linear combination of scaled copies of itself through the following formula [38, 49] :

$$N_\Xi(\xi) = \sum_{i=1}^{p+2} 2^{-p} \binom{p+1}{i-1} N_\Xi(2\xi - \xi_i) \tag{15}$$

Note that $N_\Xi(2\xi - \xi_i) = N_{\Xi_i}$ where the new knot vector Ξ_i is constructed from Ξ halving all the knot spans and taking $p + 2$ subsequent knots. For example, for a quadratic basis function defined on $\Xi = [0, 1, 2, 3]$ we would have

$$\begin{aligned} \Xi_1 &= [0, 0.5, 1, 1.5] \quad & \Xi_2 &= [0.5, 1, 1.5, 2] \\ \Xi_3 &= [1, 1.5, 2, 2.5] \quad & \Xi_4 &= [1.5, 2, 2.5, 3] \end{aligned}$$

The relation given in Equation (15) is at the core of the Hierarchical refinement: It tells us which functions of level $l + 1$ we need to include in the basis when removing functions from level l, thus ensuring the nestedness of the Hierarchical spaces, and, if one wants to utilize the weighted basis, also explicitly gives the correct coefficients needed to maintain the partition of unity. Note that we will not use the weighted basis in the numerical examples of Section 4; this is to be consistent with the definitions given in the papers we used as references, [18, 22, 46].

Once an existing function is subdivided using Equation (15), we need to check if any of the components is already present in the list of basis functions. If a component is not present, we add it to the list; if it is then we move to the next component and repeat the check. Note that if we want to utilize the weighted basis and a component is present in the list of functions, we would need to update its weight by adding the new coefficient.

The extension to the bivariate case is straightforward: Each bivariate basis function is a tensor product of two univariate ones. We then apply Equation (15) to each of them and then construct all the resulting bivariate components via the usual tensor product.

Given the conditions for the selection of areas to refine stated above, Equation (15) allows for a more intuitive understanding of how the Hierarchical refinement works: When we refine a certain region of the mesh of level l, instead of considering the full tensor product basis of level $l+1$ and selecting the correct functions from there, we can now see it as just substituting the old level l functions with their corresponding components in level $l+1$. This point of view is interesting because it shifts the attention from refining *elements* to refining *functions.*

We are now ready to construct the Hierarchical basis:

Definition 3. The *Hierarchical B-spline basis* $\mathcal{H}$ is recursively constructed as follows:

1. Initialization: $\mathcal{H}^0 = \{N \in \mathcal{N}^0 : \operatorname{supp} N \neq \emptyset\}$
2. Recursive case: $\mathcal{H}^{l+1} = \mathcal{H}_A^{l+1} \cup \mathcal{H}_B^{l+1}$ for $l = 0, \ldots, M-1$, where
$$\begin{aligned} \mathcal{H}_A^{l+1} &= \{N : N \in \mathcal{H}^l,\ \operatorname{supp} N \nsubseteq \Omega^{l+1}\} \\ \mathcal{H}_B^{l+1} &= \{N : N \in \mathcal{N}^{l+1},\ \operatorname{supp} N \subseteq \Omega^{l+1}\} \end{aligned}$$
3. $\mathcal{H} = \mathcal{H}^M$

Note that the above definition does not include the weights. The recursive definition ensures that we always select the correct functions to include in the basis. The first step initializes the Hierarchical basis with all the relevant functions of the underlying tensor product basis $\mathcal{N}^0$. The recursive procedure then updates the basis by removing the coarse functions contained inside the refined region and including the finer ones substituting them.

Figure 4 presents some of the basis functions defined on the same mesh used in Figure 3. In the Classical Hierarchical case, all the functions have rectangular support since they are plain tensor product of univariate functions.

As a result of the definition, the Classical Hierarchical B-spline basis and the associated spaces have the following properties, as proved in [46]:

- The functions in $\mathcal{H}$ are linearly independent.
- The spaces spanned by the basis are nested, i.e. $\operatorname{span}\mathcal{H}^l \subseteq \operatorname{span}\mathcal{H}^{l+1}$.

Borrowing the terminology introduced for T-splines [41, 40], we classify different hierarchical basis by the following definition

Definition 4. We denote a basis $\{N_i\}$

- *standard* if $\sum N_i = 1$
- *semi-standard* if there exists a choice of weights $w_i > 0$, such that $\sum w_i N_i = 1$

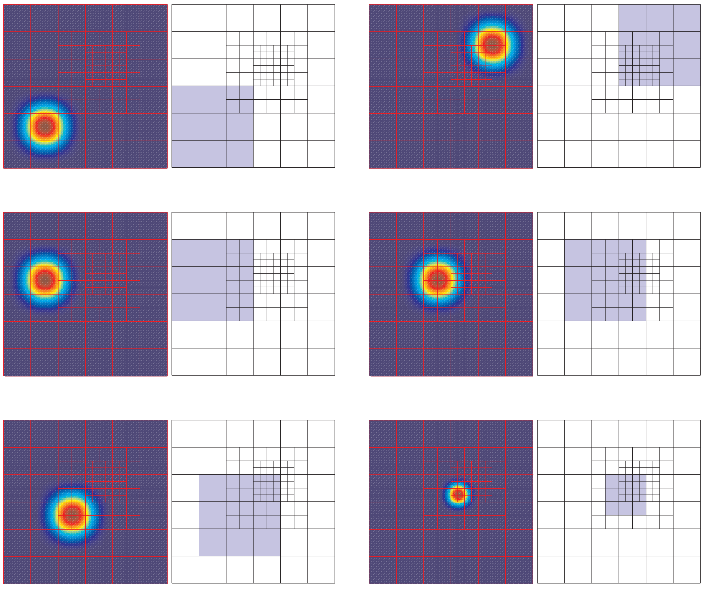


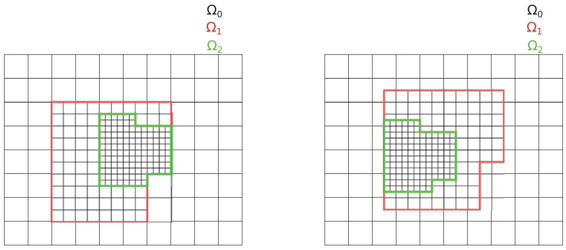
Figure 4: **Classical Hierarchical Basis:** Some of the biquadratic basis functions defined on the same mesh used in Figure 3. For each function, on the left is presented a top view of the evaluation plot and on the right the elements constituting its actual support.

- *non-standard* no choice of $w_i > 0$ exist to ensure partition of unity

The general definition of Hierarchical B-splines allows for all three kinds of basis. A tensor product mesh will yield a standard basis, but this will not be the case for arbitrary meshes. We will define our set of admissible meshes to be given as the following, which will ensure all basis to be semi-standard.

Definition 5. In the Hierarchical setting, we will call a mesh *admissible* if at all levels the area selected for refinement Ω^l is defined as the union of the supports of previous-level basis functions $N \in \mathcal{N}^{l-1}$.

The set of admissible meshes as defined above is a subset of the Hierarchical B-splines as it rules out among other, all non-standard basis. A generalization, which is not considered in this paper is that domain boundaries may not coincide with previous levels. This is known as weak condition on domain boundaries, and is illustrated in Figure 5. Another generalization is small refinement regions in which finer level functions appear, but coarser functions are not removed, see Figure 6. It is important to note here that non-standard basis may be perfectly valid for computations as they are linearly independent and well defined, but they form a different set of



(a) Strong condition on domain boundaries. The boundary of Ω^{l+1} is aligned with the knot lines of $\mathcal{V}^l$.

(b) Weak condition on domain boundaries. The boundary of Ω^{l+1} is aligned with the knot lines of $\mathcal{V}^{l+1}$.

Figure 5: **Hierarchical setting:** Different conditions on the domain boundaries. Weak boundary condition may produce non-standard basis and is not considered in this paper.

admissible meshes, which is not covered by LR B-splines and it is thus not possible to construct a basis on the same mesh and do a comparison study.

We consider Definition 5 to be highly relevant for mesh refinement. This is due to the fact that it is customary for iterative refinement to produce an error measure and refine regions of large errors. For Hierarchical B-splines, this error measure is often defined on an error per basis function level, followed by refinement of the largest error functions [22], [18].

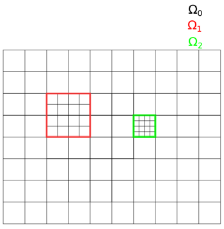


Figure 6: **Hierarchical setting:** Small domain sizes, may not be large enough to remove coarse level functions. This results in a non-standard basis and will not be considered in this paper. For this particular example, neither Ω^1 nor Ω^2 is large enough to remove any level 0 functions (for $p > 1$), but they are both large enough to create level 1 and level 2 functions of degree $p = 2$ or $p = 3$.

The Truncated Hierarchical Basis

While the Hierarchical B-splines presented above provide good flexibility and allow for localized refinement, the number of overlapping basis functions can increase very rapidly with the introduction of new levels. This happens because the large support of the coarse basis functions may overlap with the support of several fine-scale ones. See for example in Figure 4, where the top-right function, defined at level 0, overlaps with all the fine-scale functions in level 2.

This behaviour has a negative impact on the formation and solution of linear system of algebraic equation associated to the solution of the discrete (finite element) variational problem: A higher number of overlaps means we need to perform more functions evaluations and add more elements in the system matrices. This on one side increases the assembly time required to build such matrices, and on the other side it affects the sparsity, spectrum, and conditioning number of the matrices, with consequences on the performance of iterative solvers. In order to address these problems, a new basis for the Hierarchical space was proposed in [18]. The key idea is that we can appropriately *truncate* the coarse basis functions, thus reducing their support and significantly decreasing the number of overlaps.

The truncation of a basis function is defined as follows:

Definition 6. Let T be a basis function defined at level l, and let

$$T = \sum_{j\,:\,N_j \in \mathcal{N}^{l+1}} \alpha_j\, N_j$$

be its representation respect to the fine-scale basis associated to level $l+1$.

The *truncation* of T respect to $\mathcal{N}^{l+1}$ and Ω^{l+1} is defined as

$$\operatorname{trunc}^{l+1} T = \sum_{\substack{j\,:\,N_j \in \mathcal{N}^{l+1},\\ \operatorname{supp} N_j \not\subseteq \Omega^{l+1}}} \alpha_j \, N_j \tag{16}$$

It is clear that the coefficients α_j depend not only on the component N_j they refer to, but also on the function T considered. We omitted the explicit dependance to ease the notation.

The Truncated Hierarchical basis is then defined as follows [18] :

Definition 7. The *Truncated Hierarchical B-spline basis* $\mathcal{T}$ is recursively constructed as follows:

1. Initialization: $\mathcal{T}^0 = \mathcal{H}^0$
2. Recursive case: $\mathcal{T}^{l+1} = \mathcal{T}_A^{l+1} \cup \mathcal{T}_B^{l+1}$ for $l = 0, \ldots, M-1$, where

$$\begin{aligned} \mathcal{T}_A^{l+1} &= \{\operatorname{trunc}^{l+1} T : T \in \mathcal{T}^l \wedge \operatorname{supp} T \not\subseteq \Omega^{l+1}\} \\ \mathcal{T}_B^{l+1} &= \mathcal{H}_B^{l+1} \end{aligned}$$

3. $\mathcal{T} = \mathcal{T}^M$

Note that the representation of T in terms of next-level functions is easily obtained through Equation (15). The truncation mechanism removes all those components of T that are explicitly included in the basis by the recursive step of the definition. This procedure appropriately shrinks the support of all functions that cross over multiple levels in the mesh, effectively reducing the number of overlaps. Also note that the way of expressing the truncation as given in Equation (16) is what we will call an *additive* representation. It is also possible to use a *subtractive* representation, expressing the truncation as

$$\operatorname{trunc}^{l+1} T = T - \sum_{\substack{j\,:\,N_j \in \mathcal{N}^{l+1},\\ \operatorname{supp} N_j \subseteq \Omega^{l+1}}} \alpha_j \, N_j \tag{17}$$

Both these representations have advantages and disadvantages which are discussed in Section 4.

Figure 7 shows the Truncated Hierarchical basis constructed on the same mesh as in Figure 1. Note that no weights are needed to maintain the partition of unity.

Figure 8 presents the same basis functions as in Figure 4 with the truncation procedure applied. As we can see, the support of each Truncated

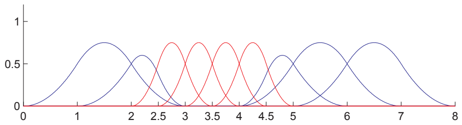


Figure 7: **Truncated Hierarchical Basis:** The quadratic basis on the same mesh as in Figure 1. Partition of unity is automatically achieved by the truncation procedure.

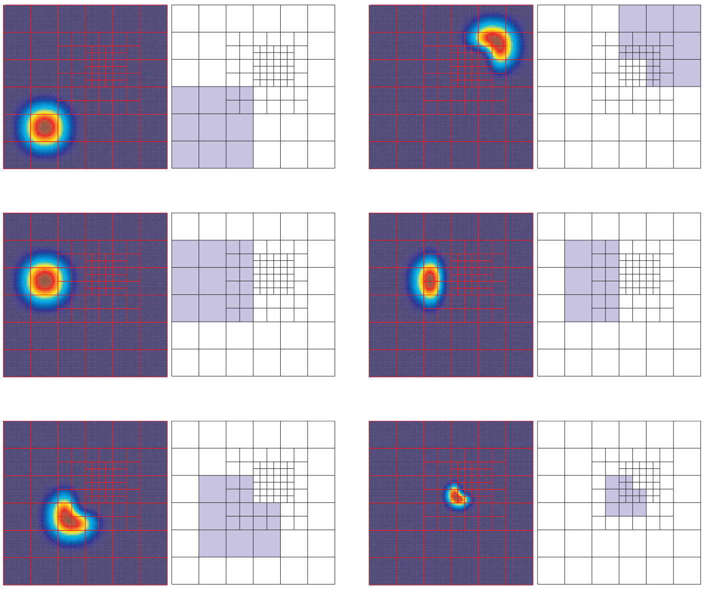


Figure 8: **2D Truncated Hierarchical basis:** The biquadratic basis constructed on the same mesh, and the corresponding functions, as in Figure 4. For each function, on the left is presented a top view of the evaluation plot and on the right the elements constituting its actual support.

function is modified in order to reduce the number of overlaps with finer levels. This, however, makes some functions lose the rectangular shape of their support.

The Truncated Hierarchical basis naturally inherits the properties of the Classical Hierarchical basis, and also adds some more. In particular:

- The functions in $\mathcal{T}$ are linearly independent.
- The spaces are nested, i.e. span$\mathcal{T}^l \subseteq$ span$\mathcal{T}^{l+1}$.
- The basis maintains partition of unity.

In addition, if we consider the Classical Hierarchical basis $\mathcal{H}$ defined on the same mesh as $\mathcal{T}$, then:

- The cardinality of the basis is the same: $|\mathcal{H}| = |\mathcal{T}|$.
- The spaces spanned are the same: span$\mathcal{H} =$ span$\mathcal{T}$.

Proofs for the above can be found in [18].

3.3 LR B-splines

LR B-splines were recently proposed by Dokken et al. in [12] and later applied to Isogeometric Analysis by Johannessen et al. in [22]. We report here some of the theory contained in those papers, while taking a different approach that focuses on clarity and ease of understanding.

LR B-splines differentiate themselves from the Hierarchical cases in the way the refinement is applied: while Hierarchical functions rely on the subdivision rule given in Equation (15) and generate up to $p+2$ new functions from each original B-spline, LR B-splines use the knot insertion procedure, inserting one knot at a time and splitting old B-splines into 2 new ones. The fact the the knots are inserted one at a time is crucial, especially in the bivariate setting: We will show that even when inserting the same knot values as produced by the subdivision rule, the resulting refined B-spline basis may be different.

LR B-splines are locally refined in the same way the standard tensor-product B-splines are. From basic spline theory we know that it is possible to perform knot insertion to enrich the spline space while leaving the geometry description unchanged. In the univariate case, if we want to insert the knot $\hat{\xi}$ between the knots ξ_{i-1} and ξ_i we have

$$B_{\Xi}(\xi) = \alpha_1 \, B_{\Xi_1}(\xi) + \alpha_2 \, B_{\Xi_2}(\xi) \tag{18}$$

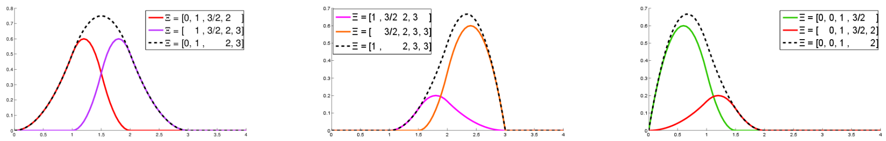


Figure 9: **LR B-splines:** Examples of knot insertion for $\hat{\xi} = \frac{3}{2}$. Dashed lines: The original functions. Colors: The new functions resulting from the splitting.

where

$$\begin{aligned}
\alpha_1 &= \begin{cases} \frac{\hat{\xi}-\xi_1}{\xi_{p+1}-\xi_1} & \xi_1 \leqslant \hat{\xi} \leqslant \xi_{p+1} \\ 1 & \xi_{p+1} \leqslant \hat{\xi} \leqslant \xi_{p+2} \end{cases} \\
\alpha_2 &= \begin{cases} 1 & \xi_1 \leqslant \hat{\xi} \leqslant \xi_2 \\ \frac{\xi_{p+2}-\hat{\xi}}{\xi_{p+2}-\xi_2} & \xi_2 \leqslant \hat{\xi} \leqslant \xi_{p+2} \end{cases}
\end{aligned} \tag{19}$$

and the knot vectors are

$$\begin{aligned}
\Xi \;\, &= [\xi_1, \xi_2 \dots \xi_{i-1}, \quad\; \xi_i \dots \xi_{p+1}, \; \xi_{p+2}] \\
\Xi_1 &= [\xi_1, \xi_2 \dots \xi_{i-1}, \hat{\xi}, \; \xi_i \dots \xi_{p+1} \qquad\quad] \\
\Xi_2 &= [\qquad \xi_2 \dots \xi_{i-1}, \hat{\xi}, \; \xi_i \dots \xi_{p+1}, \; \xi_{p+2}]
\end{aligned}$$

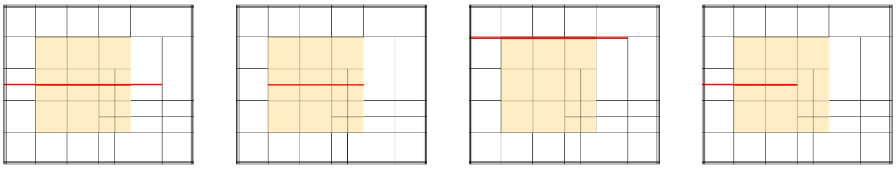
As we can see, inserting one knot splits the original B-spline into two new B-splines described by the local knot vectors Ξ_1 and Ξ_2. The weights α_1 and α_2 are needed to maintain partition of unity. Figure 9 shows some examples of the application of Equation (18).

In the bivariate case, functions are refined one parametric direction at a time. In this case we obtain:

$$\begin{aligned}
B_{\boldsymbol{\Xi}}(\xi, \eta) &= B_{\Xi}(\xi)\, B_{\Psi}(\eta) \\
&= (\alpha_1\, B_{\Xi_1}(\xi) + \alpha_2\, B_{\Xi_2}(\xi))\;\, B_{\Psi}(\eta) \\
&= \alpha_1\, B_{\boldsymbol{\Xi_1}}(\xi, \eta) + \alpha_2\, B_{\boldsymbol{\Xi_2}}(\xi, \eta)
\end{aligned} \tag{20}$$

In the following we will call *meshline extension* all mesh-altering actions like inserting a new meshline, prolonging existing meshlines (possibly connecting two existing ones) or increasing the multiplicity of meshlines. When a new meshline extension is inserted, we need to know which basis functions are affected by it. For this purpose, we give the following definition:

Definition 8. A meshline ϵ is said to *traverse the support* of a function $B_{[\xi_1 \dots \xi_{p_1+2}; \eta_1 \dots \eta_{p_2+2}]}$ if



(a) Line traversing the interior of B

(b) Line traversing the interior of B

(c) Line traversing the edge of B

(d) Line neither traversing the edge nor the interior of B

Figure 10: **LR B-splines:** Examples of lines traversing the support of a basis function.

- ϵ is a horizontal line $\epsilon = [\xi_1^*, \xi_2^*] \times \eta^*$ such that

$$\xi_1^* \leqslant \xi_1, \quad \xi_{p_1+2} \leqslant \xi_2^*, \quad \eta_1 \leqslant \eta^* \leqslant \eta_{p_2+2}$$

- ϵ is a vertical line $\epsilon = \xi^* \times [\eta_1^*, \eta_2^*]$ such that

$$\xi_1 \leqslant \xi^* \leqslant \xi_{p_1+2}, \quad \eta_1^* \leqslant \eta_1, \quad \eta_{p_2+2} \leqslant \eta_2^*$$

In particular, a horizontal line is said to traverse *the interior* of $B_{[\xi_1 \cdots \xi_{p_1+2};\eta_1 \cdots \eta_{p_2+2}]}$ if $\eta_1 < \eta^* < \eta_{p_2+2}$ and traverse *the edge* if $\eta^* = \eta_1$ or $\eta^* = \eta_{p_2+2}$. Similarly, a vertical line is said to traverse the interior if $\xi_1 < \xi^* < \xi_{p_1+2}$ and traverse the edge if $\xi^* = \xi_1$ or $\xi^* = \xi_{p_1+2}$.

Figure 10 shows some examples of lines traversing the support of a basis function.

When a meshline extension is applied, the refinement process is composed of two steps:

1. Split any function which support is traversed by the *new* meshline.

2. For all new functions, check if their support is traversed by any *existing* meshline, and split again if this happens.

In step 1 we test all current functions against one meshline. In step 2 we test all newly created functions against all existing meshlines. Note that when the meshline extension is an actual elongation, possibly connecting two separate existing meshlines, we will use the full length of the resulting line to test the functions for splitting. When a function is flagged for splitting, this is performed through the use of Equations (19) and (20).

In view of the above, we can give the following two definitions:

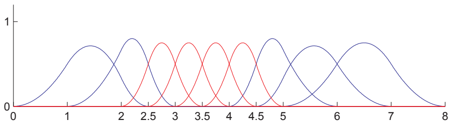


Figure 11: **LR B-splines:** The basis constructed on the same mesh as in figures 1, 2, and 7.

Definition 9. In the LR B-splines setting, an *admissible mesh* is any mesh which can be obtained by a sequence of meshline extensions starting from an initial tensor product mesh. Each extension must cause at least one basis function to be split, and the meshlines must end at existing knot values (they cannot stop at the centre of an element). All tensor product meshes are admissible.

Definition 10. An *LR B-spline* is a function which results from the application of the refinement scheme and Equations (18)-(19). All tensor product B-splines are LR B-splines.

Figure 11 shows the 1D LR B-splines basis defined on the same mesh as in the previous examples at Figures 1, 2, and 7. Note that in the univariate setting the LR B-spline refinement coincide with the normal knot insertion. In this case all the weights sum up to 1 and so the LR B-spline basis is the normal B-splines basis originating from the non-uniform knot vector $\Xi = [0, 1, 2, 2.5, 3, 3.5, 4, 4.5, 5, 6, 7, 8]$, and automatically maintains partition of unity. Also note that in the general LR B-spline setting the notion of levels is not as present as in the Hierarchical setting; we can however define the level of an LR B-spline function using the maximum knotspan contained in its local knot vector.

For a thorough example in the bivariate case we refer the reader to [22, p. 481-483].

Given an initial tensor product mesh $\mathcal{M}_0$, a sequence of meshline extensions $\{\epsilon_i\}_{i=1}^n$ and corresponding admissible meshes $\mathcal{M}_i = \{\mathcal{M}_{i-1} \cup \epsilon_i\}$ and LR B-splines $\mathcal{L}^i$, the following properties hold [12, 22]:

- The spaces are nested: $\text{span}\mathcal{L}^i \subseteq \text{span}\mathcal{L}^{i+1}$.
- The LR B-splines defined on a mesh are not affected by the order in which the meshline extensions have been inserted, i.e. if $\mathcal{M}$ and $\hat{\mathcal{M}}$ are two identical LR B-splines meshes that differ only for the order in which the meshlines extensions have been applied, then the resulting LR B-splines functions are the same.
- The LR B-splines form a partition of unity.

Note that, in general LR B-splines may be linear dependent. This is mostly due to the fact that the single-line insertion mechanism used in this setting allows for several different types of refinement strategies, and the particular choice naturally affects the spline space. However, there are several ways to recover the linear independency as proposed in [12, 22]. The linear independence of LR B-splines depending on the type of refinement strategy used is currently object of research.

In all our examples we will use the *Structured Mesh* refinement presented in [22]. This strategy focuses on refining functions, instead of elements. The idea of refining elements is indeed a legacy from the classical Finite Element methodology. Using the Structured Mesh approach, one instead selects which basis functions to refine. This can be done through the use of custom-built criteria, just as one would do in an adaptive refinement scheme. The idea proposed in [22] is to compute the error pertaining to each basis function as

$$\|e\|^2_{\mathrm{supp}B_i} = \sum_{K \in \mathrm{supp}B_i} \|e\|^2_K \tag{21}$$

i.e. we define the *B-spline error* as the sum of the normal error $\|e\|$ measured in the energy norm over all elements in the support of B_i. Once the functions to be refined are identified, we proceed to insert several knot lines in both directions, halving the knot spans of the largest supported knot interval. Note that the Structured Mesh strategy will yield the same results on the mesh as the subdivision rule used in the Hierarchical setting. This means that all meshes which are admissible in the Hierarchical setting, i.e. they satisfy the conditions of Definition 5, are also admissible in the LR B-splines setting and can be obtained using the Structured Mesh refinement. For this reason we have always used this approach for our examples, as it provides a better ground for comparison.

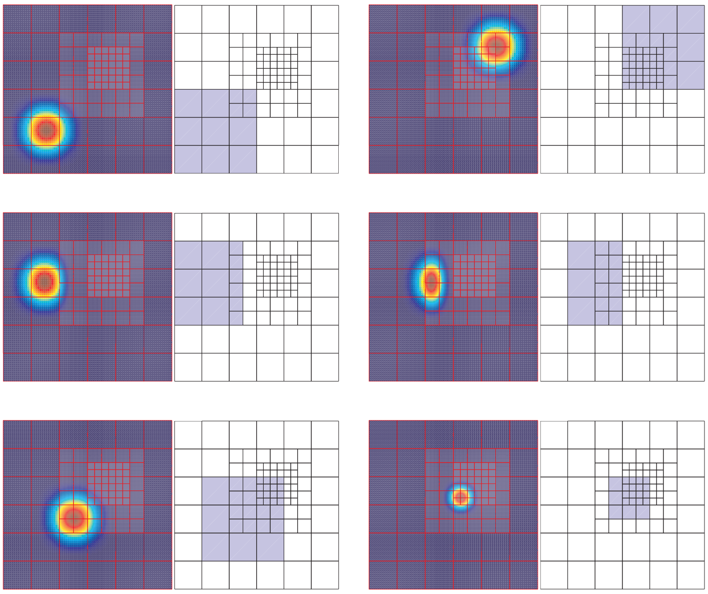


Figure 12: **LR B-splines:** The biquadratic basis constructed on the same mesh, and the corresponding functions, as in Figure 4 and 8. For each function, on the left is presented a top view of the evaluation plot and on the right the elements constituting its actual support.

4 Results

We present here the results of our analysis on the different type of basis functions outlined in Section 3. The *Qualitative analysis* sections collects results regarding the mathematical properties of the various basis, many of which were already briefly listed in the corresponding sections. The *Quantitative analysis* section focuses on implementation and numerical quantities and discusses the properties of the stiffness and mass matrices generated using the different splines functions.

4.1 Qualitative analysis

We would like to start pointing out that, under normal mesh refinement iterations (i.e. excluding special constructed cases), on the qualitative level all the three classes of splines give comparable results. However, there are some interesting distinctive features that are worth to be mentioned.

Different functions

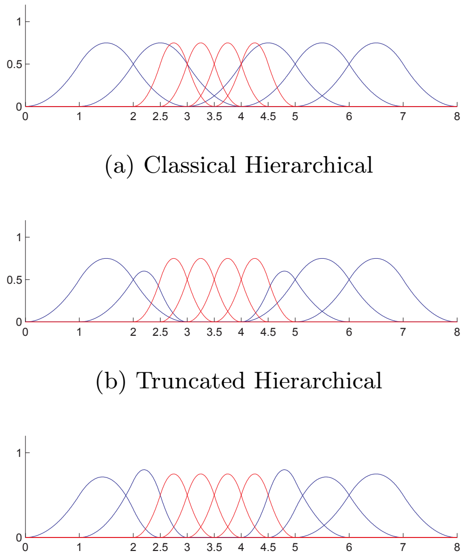
With the notation introduced at page 106 we have that the general functions $H \in \mathcal{H}$, $T \in \mathcal{T}$, and $L \in \mathcal{L}$ can be written as

$$\begin{aligned} H &= N \\ T &= \sum \alpha_i \, N_i \\ L &= \alpha \, B \end{aligned}$$

for appropriate indices i and weights α

This can be seen as follows. Consider a set of nested domains $\Omega^0 \supset \Omega^1 \supset \ldots \supset \Omega^M$, with corresponding tensor mesh $V^0 \subset \ldots \subset V^M$, where $V^l = \Xi^l \otimes \Psi^l$, $\Xi^l \subset \Xi^{l+1}$ and $\Psi^l \subset \Psi^{l+1}$. The knot vectors of $N_i^l \in \mathcal{N}^l$ is picked as *connected subsets* of Ξ^l and Ψ^l. These are uniform both in the index domain of Ξ^l and Ξ^M. This is in contrast to how the LR B-splines are constructed, as these are in general a *unconnected* subset of Ξ^M and Ψ^M and hence non-uniform in the index domain.

For uniform starting meshes under dyadic refinement this becomes slightly more apparent as every N will be comprised of uniform local knot vectors, while any B will potentially be non-uniform. The Classical Hierarchical functions (Definition 3) are then uniform B-splines; Truncated Hierarchical functions (Definitions 6 and 7) are generally a linear combination of these uniform B-splines; LR B-splines functions (Equation (18) and Definition 10) are non-uniform B-splines.



(a) Classical Hierarchical

(b) Truncated Hierarchical

(c) LR B-splines

Figure 13: **Qualitative analysis:** The different quadratic bases constructed using the same knot vector.

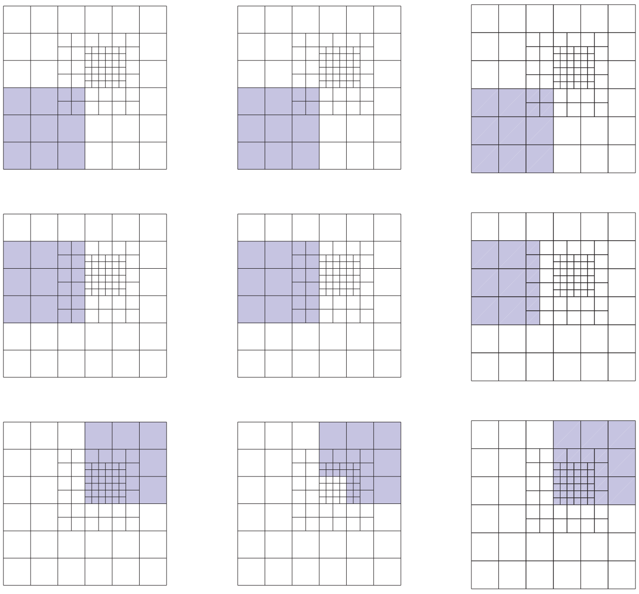
Figure 13 shows the Classical Hierarchical, Truncated Hierarchical and LR B-splines basis for the knot vector $\Xi = [0, 1, 2, 2.5, 3, 3.5, 4, 4.5, 5, 6, 7, 8]$. Note that on the uniform vector $[0 : 8]$ all three families of functions would be exactly the same. In the Classical Hierarchical case, partition of unity is not preserved. In the Truncated Hierarchical case this is achieved automatically by the truncation procedure, which removes some components from the old-level functions. The LR B-splines are instead defined by non-uniform local knot vectors, and also use weights to maintain partition of unity.

Figure 14 shows a comparison of the support for some of the basis functions presented in the Figures 4, 8, and 12.

Different spaces

Perhaps the most important and interesting difference is that for some meshes the Hierarchical basis and the LR B-splines set span different spaces. One such example is given in figure 15.

The central function appearing in the Classical and Truncated Hierarchical setting corresponds to two distinct functions in the LR B-splines set. This is due to the way the refinement works in the LR B-splines setting: As we can see there is one new meshline which completely traverses the support of the central function. When this meshline is inserted, it triggers the LR

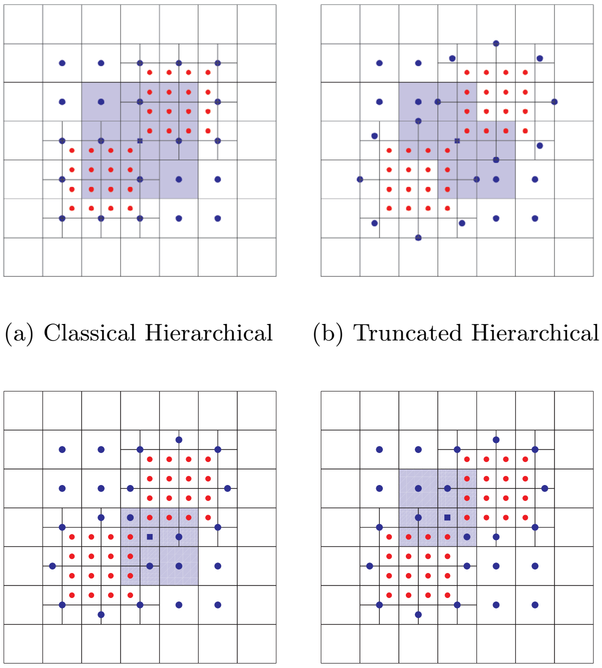


(a) Left Column: Classical Hierarchical

(b) Centre Column: Truncated Hierarchical

(c) Right Column: LR B-splines

Figure 14: **Qualitative analysis - Different Functions:** The support of corresponding biquadratic basis functions in the three spline families presented in the Figures 4, 8, and 12.



(a) Classical Hierarchical

(b) Truncated Hierarchical

(c) LR B-splines, first function

(d) LR B-splines, second function

Figure 15: **Qualitative analysis - Different spaces:** An example of mesh on which the biquadratic Hierarchical bases span different spaces than the LR B-splines basis. The central level 0 function in the Hierarchical cases corresponds to two distinct functions in the LR B-splines basis. Both the Hierarchical bases are constituted of 55 functions; the LR B-splines basis contains 56 functions. The highlighted area is the support of the selected function, represented with a square as anchor symbol. The anchors are placed as described at page 108.

B-splines refinement algorithm which splits the original B-spline into two new ones as expected. This does not happen in the Hierarchical framework, which leaves the function unchanged in the Classical case or appropriately reduces its support in the Truncated case.

Mourrain [32] presented a formula for the maximum dimension of the space of piecewise polynomials with given continuity on a mesh: given a planar mesh with F faces (the elements), ϵ_H horizontal and ϵ_V vertical internal edges and P vertices, the maximum dimension of the space of bivariate piecewise polynomials of degrees (p, q) with continuity (k, l) along element edges is given by

$$\mathcal{S} = (p+1)\,(q+1)\,F - (p+1)\,(l+1)\,\epsilon_H - (q+1)\,(k+1)\,\epsilon_V + (k+1)\,(l+1)\,P + h \tag{22}$$

where h is the homology factor of the mesh, which is equal to zero for all the refinement schemes used here. For Hierarchical B-splines, as well as Truncated Hierarchical, it is possible to add constraints on the mesh topology to ensure spanning the entire space [31], [19]. For LR B-splines one may assure this by the so called "hand in hand" process [12], which again puts restrictions on mesh topologies. For meshes under Definition 5, this however cannot be guaranteed.

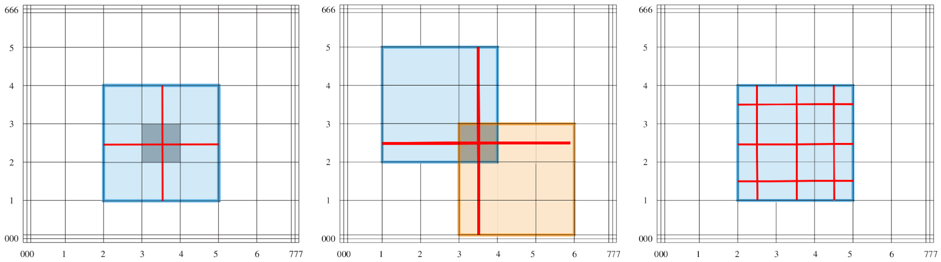
Different refinement strategies

Hierarchical functions rely on Equation (15) to apply the refinement. This procedure halves the local knot vectors of the function and replaces the original B-spline with up to $p + 2$ new functions in the univariate case, or $(p + 2)(q + 2)$ in the bivariate case.

LR B-splines use the knot insertion given in Equation (18), which introduces two new B-splines functions. This procedure allows for more flexibility in the refinement approach as there are no prescriptions on the number or positions of the new knots. In addition to the Structured Mesh strategy, already presented in Section 3, in [22] two other different refinement strategies are proposed: *Minimum Span* and *Full Span*. While all these strategies insert the meshlines so that they halve the knotspans, this is not a requirement as the use of non-uniform knot vectors is already built-in in the definitions of LR B-splines.

The Minimum Span strategy aim is to keep the refinement as localized as possible. Once an element is marked for refinement, a cross is inserted through its centre and the meshlines are made to be as short as possible, while still splitting at least one function.

In the Full Span strategy the idea is to split *all* B-splines with support on a selected element. This is done inserting two meshlines in a cross through



(a) Minimum Span

(b) Full Span (only two functions are shown)

(c) Structured Mesh

Figure 16: **Qualitative analysis:** Different types of refinement strategies using LR B-splines.

the centre of the element. The new meshlines will have to span from the minimum to the maximum knot values of all functions with support on the marked element in both parametric direction. This strategy makes sure that all B-splines with support on the marked element are treated equally, but on the other hand this results in an extension of the refinement away from the selected element, in particular for high polynomial degrees.

A common drawback of both the Full and Minimum Span is that some elements will be traversed by only one meshline and therefore will be split into two rectangular elements, effectively doubling their aspect ratio.

While the possibility of applying other refinement strategies is allowed by the definitions and the theory of LR B-splines, some may lead to linearly dependant sets. The research in these cases is still ongoing and, as of today, the Structured Mesh is the best candidate for a stable refinement algorithm.

Different admissible meshes

A direct consequence of the various refinement approaches available with LR B-splines is that some meshes which are legal in a LR setting cannot be reproduced using Hierarchical splines. On the other hand, LR B-splines are not capable of achieving some configurations of the weak conditions on domain boundaries available in the Hierarchical framework. For an example see Figure 5b. In that case, the meshlines of Ω^1 are stopping in the centre of the elements, a behaviour that is disallowed by the LR definitions.

Another difference is that LR B-splines are currently defined starting from a global tensor-product mesh, i.e. only on rectangular parametric domains. Conversely, Hierarchical B-splines can be defined on non-rectangular parametric domains.

Meshes that are defined on a rectangular domain and also satisfy the conditions of Definition 5 are admissible in both the Hierarchical and LR B-splines framework.

4.2 Quantitative analysis

Here, we first discuss details related to different representation of Truncated Hierarchical basis, and then present the numerical results obtained for different meshes and polynomial degrees,

Representation of Truncated functions

As we briefly mentioned earlier, Truncated Hierarchical B-splines can be represented in an additive or subtractive fashion; we restate Equations (16) and (17) for reading convenience:

$$\operatorname{trunc}^{l+1} T = \sum_{\substack{j\,:\,N_j \in \mathcal{N}^{l+1}, \\ \operatorname{supp} N_j \not\subseteq \Omega^{l+1}}} \alpha_j \, N_j \qquad \text{Additive representation} \tag{23}$$

$$\operatorname{trunc}^{l+1} T = T - \sum_{\substack{j\,:\,N_j \in \mathcal{N}^{l+1}, \\ \operatorname{supp} N_j \subseteq \Omega^{l+1}}} \alpha_j \, N_j \qquad \text{Subtractive representation} \tag{24}$$

where N_j are the components of T with respect to the finer level basis functions and α_j the corresponding weights as given by Equation (15).

When implementing the code we found that choosing one representation over the other yield important consequences. In a typical finite element code one has to deal with two important aspects: determining which basis functions are active over a given element, and then evaluating such functions.

To address the former a convenient way to retrieve or store the support of the functions is essential. In the case of B-splines this is generally easy since the support is identified with the local knot vectors. When the B-spline is a standard tensor product, and the support is therefore rectangular, this becomes even easier since one only needs to check the starting and ending points of the local knot vectors. As we have seen, however, Truncated Hierarchical functions do not always have rectangular support, hence we need a representation that allows for an easy way to retrieve it. The subtractive representation (24) is unfortunately not very helpful in this sense: the fact that a given component is subtracted does not automatically guarantee that the function itself vanishes in that area. Given an element $E \in \operatorname{supp} T$ we should check if *all* possible components on that element are removed in order to know if $\operatorname{trunc} T$ has support on E or not. The additive representation

(23), on the other hand, is much more convenient: We can simply loop over all components and check if *any* of them has support on E.

To address the latter point we need an efficient way to evaluate basis functions. This is even more important in an Isogeometric setting: Since the Cox-de Boor algorithm is a typical bottleneck of the code, we would like to perform as few basis evaluations as possible. In this case the additive representation (23) is not efficient. In a biquadratic case a representation in terms of next-level basis functions comprises of 16 fine-scale functions. This number clearly increases when increasing the polynomial degree: 25 for bicubic functions, 36 for biquartic etc. In addition, an additive representation may require to store the function in terms of the finest-available scale, which would greatly increase the amount of components needed; let's assume, for simplicity, that this is not the case. To give an example, look at the biquadratic basis functions of Figure 8. For each of the Truncated Hierarchical basis functions only 4 components are removed. This means that in an additive representation we would still need to evaluate 12 fine-scale functions in order to compute the value of the B-spline we are interested in. In a subtractive representation we would need to evaluate only 5 functions: The original tensor product B-spline and the 4 components we need to subtract.

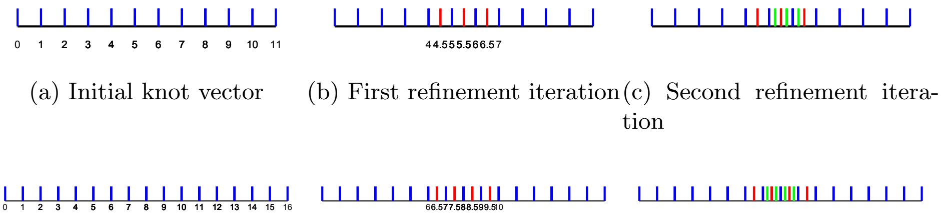
To summarize, we have the following:

- An additive representation (23) is useful when determining the support but not efficient in the function evaluation process;
- A subtractive representation (24) does not allow to easily identify the support of the function but is more efficient in its evaluation.

The above discussions are overall considerations: The disadvantages of the representations might be accounted for by programming the algorithm in a smart efficient way. On the other hand, this is still something that needs to be taken into consideration. For an in-depth discussion on the implementation of Truncated Hierarchical B-splines we refer to [25].

1D examples

We now present the results obtained in various 1D examples. We performed several experiments for polynomial degrees $p = 2, 3, 4$, and 5. In each case we started from a uniform, non-open knot vector $\Xi^0 = [0, 1 \ldots 5\,p + 1]$ and successively applied 6 refinement steps, always refining the central basis function. Note that for odd polynomial degrees a central function always exists, while for even-degree normally there are two functions near the knot vector centre. In this case we chose to always refine the rightmost one. Figure 17 shows as examples the first two refinement iterations for $p = 2$ and $p = 3$.



(a) Initial knot vector (b) First refinement iteration (c) Second refinement iteration

(d) Initial knot vector (e) First refinement iteration (f) Second refinement iteration

Figure 17: **1D Central Refinement:** The first three steps of the refinement process in the cases $p = 2$ (above) and $p = 3$ (below). When two functions are equally close to the centre, the rightmost one is selected for refinement.

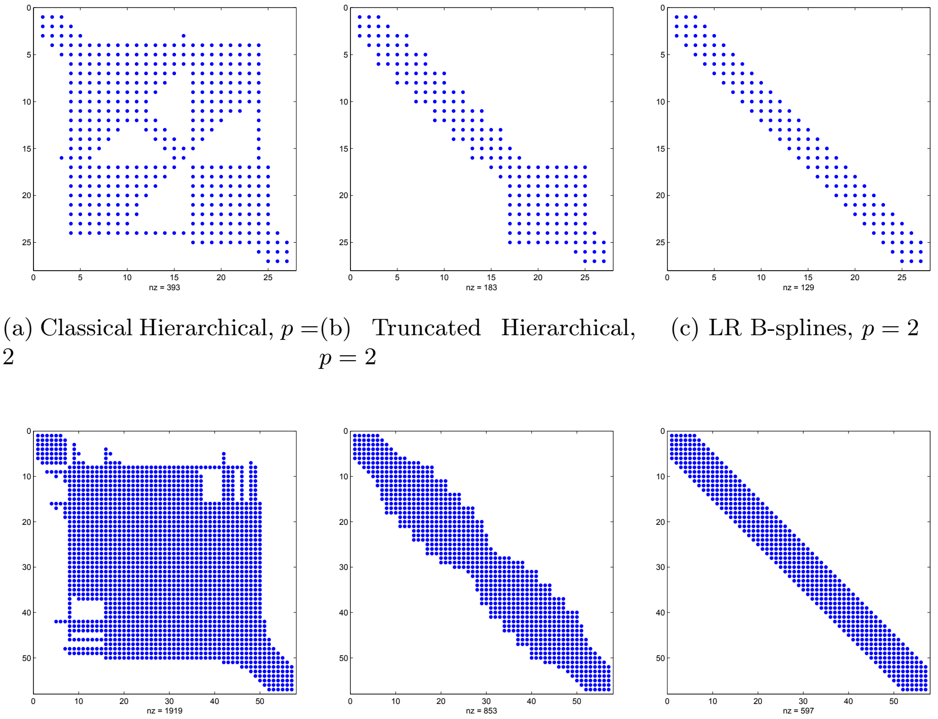
p	HB	THB	LR	H/T	H/LR
2	393	183	129	215%	305%
3	803	315	247	255%	325%
4	1257	629	403	200%	312%
5	1919	853	597	225%	321%

Table 1: **1D Central Refinement:** Number of non-zero elements in the stiffness matrix at the last (6th) refinement iteration. The last two columns present the ratios, rounded to the nearest percentage point.

For each refinement iteration we constructed the stiffness matrix A and the mass matrix M using the Classical Hierarchical, Truncated Hierarchical and LR B-splines functions defined using the same knot vector. We then analysed some important numerical properties of these matrices, namely the sparsity pattern, the conditioning number, and the spectrum. Note that in the univariate case the LR B-splines basis coincides with the standard non-uniform B-splines generated via knot insertion.

Sparsity Figure 18 shows the sparsity patterns of the stiffness matrix at the last refinement iteration after a reordering using the Cuthill-McKee Algorithm [11] has been applied. The top row corresponds to $p = 2$, while to bottom row corresponds to $p = 5$.

As expected the Classical Hierarchical basis functions produce the densest stiffness matrices. This is normal since the support of coarse-level functions remains unaffected by the refinement in neighbouring regions. The values for all polynomial degrees are collected in Table 1.



(a) Classical Hierarchical, $p = 2$

(b) Truncated Hierarchical, $p = 2$

(c) LR B-splines, $p = 2$

(d) Classical Hierarchical, $p = 5$

(e) Truncated Hierarchical, $p = 5$

(f) LR B-splines, $p = 5$

Figure 18: **1D Central Refinement:** Examples of sparsity patterns of the stiffness matrices at the last(6th) refinement iteration. The Cuthill-McKee Algorithm has been applied to optimize the bandwidth. Top: $p = 2$. Bottom: $p = 5$.

Conditioning Numbers The conditioning number of the stiffness and mass matrices can be significantly influenced by the way the boundary conditions are imposed. In order to avoid any such effect we decided to look at the “pure” conditioning numbers, i.e. before any imposition of the boundary conditions. As is well known, with just pure Neumann boundary conditions the stiffness matrix is singular; the conditioning number can then be defined as the ratio between the largest eigenvalue and the smallest non-zero one. This means that for either the stiffness matrix A or the mass matrix M, given the ordered set of their eigenvalues $[\lambda_1, \lambda_2 \dots \lambda_n]$ we define

$$\begin{aligned} \operatorname{cond}(A) &= \frac{\lambda_n}{\lambda_2} \\ \operatorname{cond}(M) &= \frac{\lambda_n}{\lambda_1} \end{aligned} \tag{25}$$

Figure 19 shows the plots for the conditioning numbers of both the stiffness and mass matrices for each polynomial degree considered. While all values are quite close to each other, and always remained in the same order of magnitude for our experiments, it is interesting to note that the Truncated Hierarchical and LR B-splines perform very similarly.

Looking at the plots for the stiffness matrix we can see that, with the exception of the lowest degree, i.e. $p = 2$, the conditioning numbers are ordered as

$$\operatorname{cond}(A_T) < \operatorname{cond}(A_{LR}) < \operatorname{cond}(A_H)$$

while for the mass matrix we always have

$$\operatorname{cond}(M_{LR}) < \operatorname{cond}(M_T) < \operatorname{cond}(M_H)$$

where the subscripts indicates the basis functions used.

We can also see that the conditioning numbers of the stiffness matrices are increasing with each refinement iteration, while the conditioning numbers for the mass matrices for $p = 4$ and 5 are bounded from above and below by a constant. This behaviour was already presented by Gahalaut and Tomar in [15] and Garoni et al. [16], although that result is proven for uniform refinement only.

The Tables 2 and 3 present the numerical data for $p = 2$ and $p = 3$, respectively.

Spectrum Figure 20 shows the spectra of the stiffness and mass matrices for $p = 2$ at the sixth refinement iteration. The eigenvalues of the stiffness matrix are spread over a large interval, while the eigenvalues of the mass matrix are much more clustered. In all cases the eigenvalues tend to be

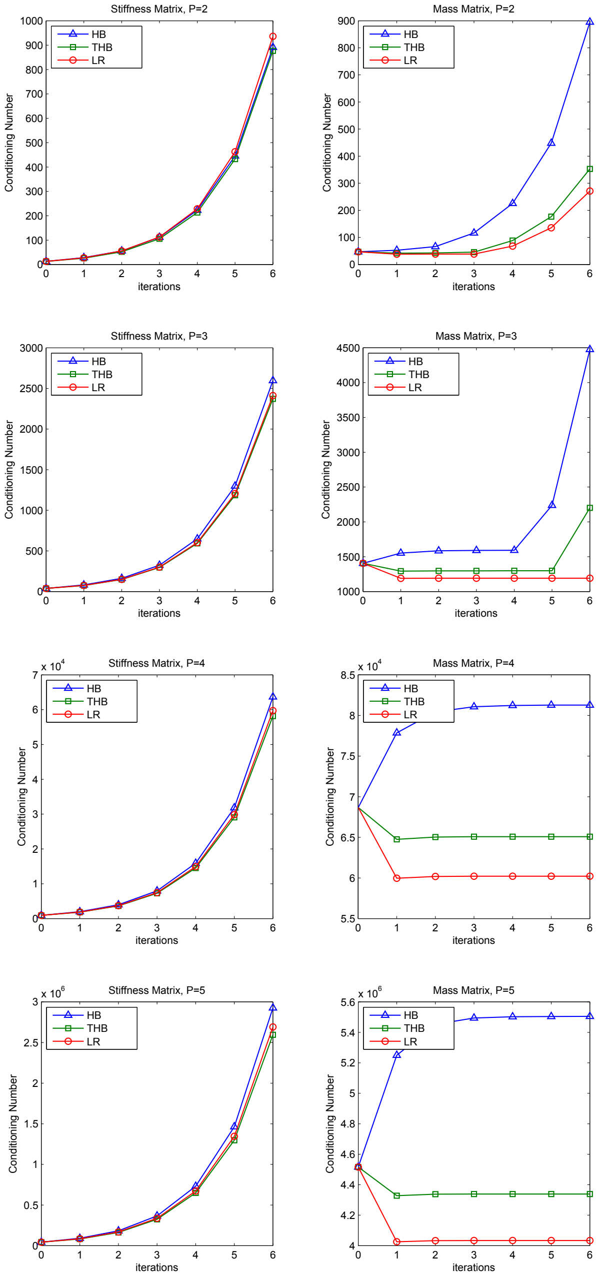


Figure 19: **1D Central Refinement:** Graphs of the conditioning numbers of stiffness matrices (left column) and mass matrices (right column) from $p = 2$ (top) to $p = 5$ (bottom).

Stiffness Matrix

Iter.	0	1	2	3	4	5	6
HB	12.742	28.029	55.751	111.403	222.790	445.579	891.158
THB	12.742	25.825	52.050	105.316	213.368	432.490	876.362
LR	12.742	27.284	55.600	112.638	228.151	462.230	936.191

Mass Matrix

Iter.	0	1	2	3	4	5	6
HB	46.794	52.523	65.893	116.226	225.483	448.117	894.973
THB	46.794	41.516	42.670	45.683	88.248	176.373	352.715
LR	46.794	38.037	38.429	38.594	67.776	135.537	271.070

Table 2: **1D Central Refinement:** The conditioning numbers for $p = 2$ throughout the mesh refinement. Stiffness Matrix above, Mass Matrix below.

Stiffness Matrix

Iter.	0	1	2	3	4	5	6
HB	37.585	81.260	162.294	324.648	649.310	1298.622	2597.244
THB	37.585	74.052	148.15	296.333	592.685	1185.379	2370.764
LR	37.585	75.193	150.678	301.461	602.976	1205.979	2411.972

Mass Matrix

Iter.	0	1	2	3	4	5	6
HB	1405.22	1553.05	1585.28	1590.56	1591.56	2238.16	4476.30
THB	1405.22	1292.26	1296.80	1297.36	1297.47	1297.60	2201.90
LR	1405.22	1190.16	1191.54	1191.79	1191.81	1191.81	1191.81

Table 3: **1D Central Refinement:** The conditioning numbers for $p = 3$ throughout the mesh refinement. Stiffness Matrix above, Mass Matrix below.

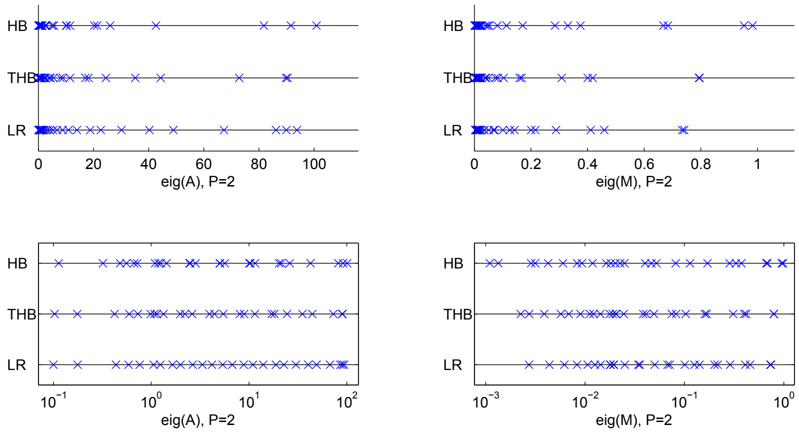


Figure 20: **1D Central Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 2$ at the last refinement iteration. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

denser near the origin, but there is no substantial difference between the various basis functions.

Increasing the polynomial degree has different consequences on the eigenvalues of the stiffness and mass matrices: While the large eigenvalues of the stiffness matrix are reduced, those of the mass matrix are increased. The values of the smallest eigenvalues are instead reduced in all cases, as we would expect by the increase in the conditioning numbers. We noted, however, that only a small number of outliers quickly approaches zero, while the other smallest eigenvalues are still reduced but not as fast. In particular, for the Classical and Truncated Hierarchical basis functions the smallest eigenvalues seems to decrease faster than those associated with LR B-splines. Figure 21 shows the spectra of the stiffness and mass matrices produced with basis functions of degree $p = 5$.

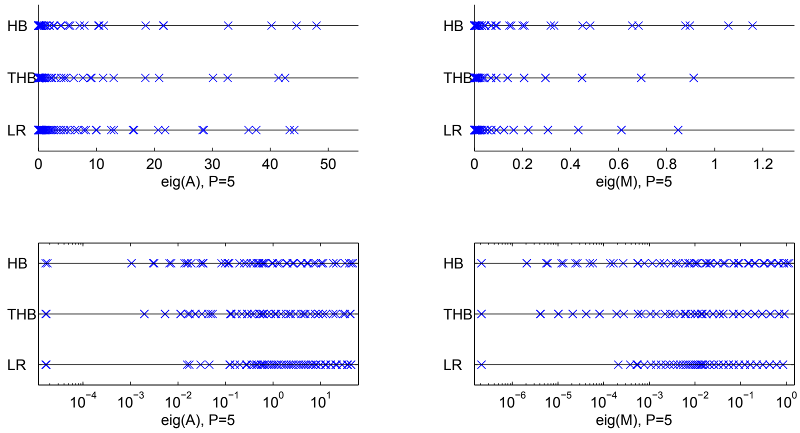


Figure 21: **1D Central Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 5$ at the last refinement iteration. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

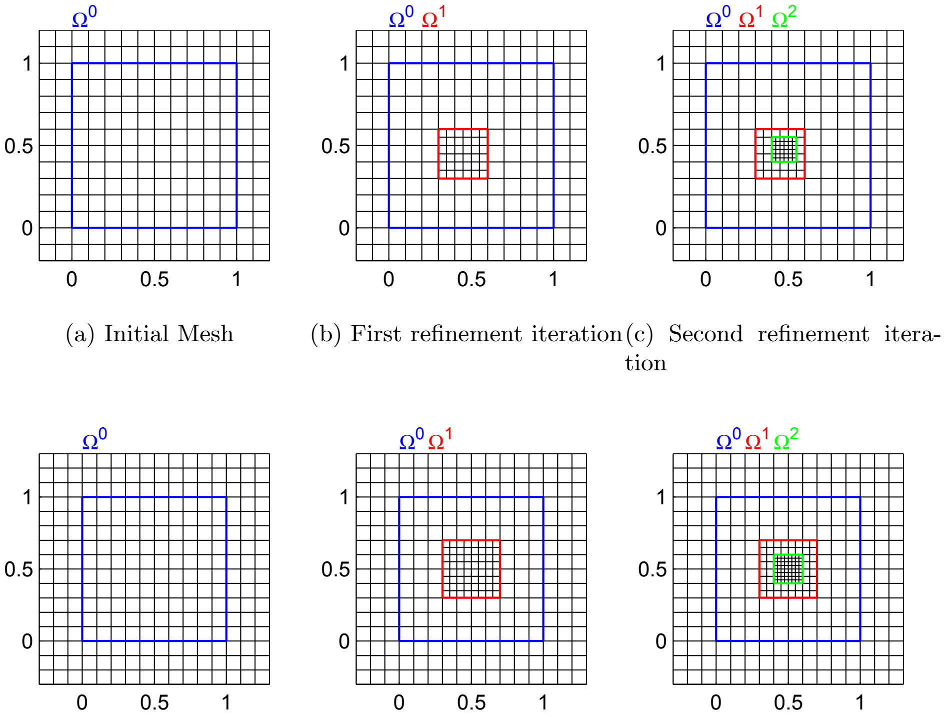
2D Example: Central Refinement

The first of the 2D examples we present is the natural extension of the 1D cases considered above. Starting from a uniform tensor-product mesh, we performed five refinement iterations where at each step the central basis function was selected for refinement. Experiments were conducted for $p = 2, 3, 4$. As in the previous cases, for even polynomial degrees usually none of the basis functions is perfectly in the centre of the mesh; when this happened we chose to refine the lower-left function. Depending on the polynomial degree we also adjusted the knot vectors to have a ghost domain such that the initial tensor product basis constitutes a partition of unity in $\Omega^0 = [0, 1] \times [0, 1]$. Figure 22 shows the first three steps of the refinement process for $p = 2$ and $p = 3$.

As in the previous examples, we constructed the stiffness and mass matrices using Classical Hierarchical, Truncated Hierarchical and LR B-splines basis functions on the same meshes, and compared their numerical properties.

Sparsity Figure 23 shows the sparsity pattern of the stiffness matrices after the fifth refinement iteration for biquadratic and biquartic basis functions. All results are reported in Table 4.

As expected the Classical Hierarchical basis produces significantly denser matrices due to the higher number of overlaps. While it may seem that the



(a) Initial Mesh

(b) First refinement iteration

(c) Second refinement iteration

(d) Initial Mesh

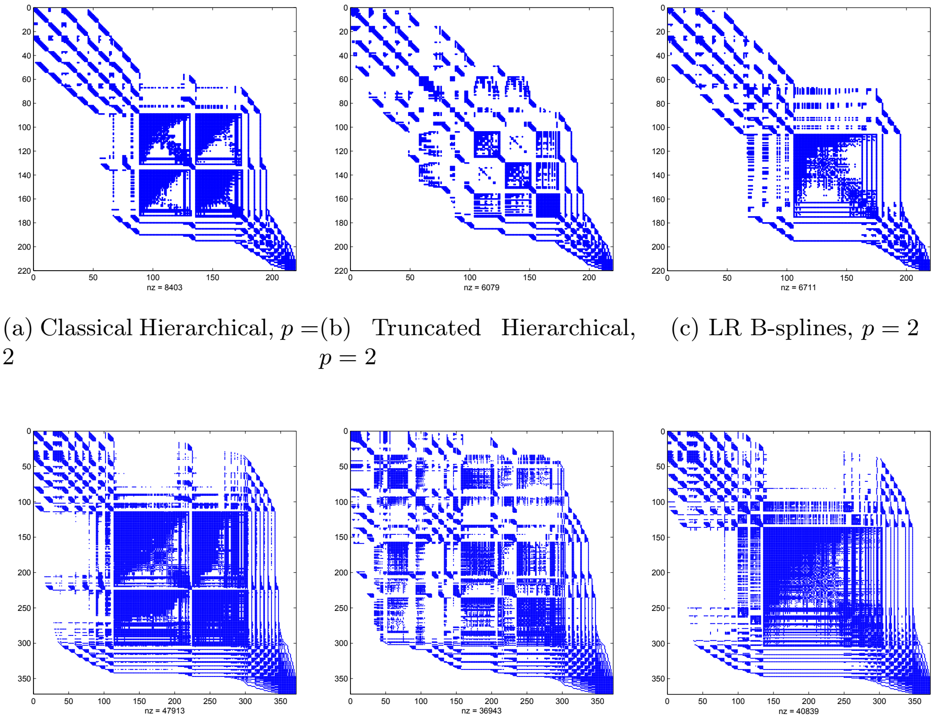
(e) First refinement iteration

(f) Second refinement iteration

Figure 22: **2D Central Refinement:** The first three steps of the refinement process in the cases $p = 2$ (above) and $p = 3$ (below). When four functions are equally close to the centre, the lower-left one is selected for refinement.

p	HB	THB	LR	H/T	H/LR
2	8403	6079	6711	138%	125%
3	23625	14909	18909	158%	125%
4	47913	36943	40839	130%	117%

Table 4: **2D Central Refinement:** Number of non-zero elements in the stiffness matrices at the last (5th) refinement iteration. The last two columns present the ratios, rounded to the nearest percentage point.



(a) Classical Hierarchical, $p = 2$

(b) Truncated Hierarchical, $p = 2$

(c) LR B-splines, $p = 2$

(d) Classical Hierarchical, $p = 4$

(e) Truncated Hierarchical, $p = 4$

(f) LR B-splines, $p = 4$

Figure 23: **2D Central Refinement:** The sparsity patterns of the stiffness matrices at the last (5th) refinement iteration. The Cuthill-McKee Algorithm has been applied to optimize the bandwidth. Top: $p = 2$. Bottom: $p = 4$.

Stiffness Matrix

Iter.	0	1	2	3	4	5
HB	83.6793	125.4868	142.641	152.964	159.1009	162.9791
THB	83.6793	94.7517	99.6748	102.4115	103.8121	104.6173
LR	83.6793	91.0205	91.0144	91.0061	91.006	91.0105

Mass Matrix

Iter.	0	1	2	3	4	5
HB	$2.241{\cdot}10^3$	$2.245{\cdot}10^3$	$2.245{\cdot}10^3$	$2.245{\cdot}10^3$	$5.808{\cdot}10^3$	$2.323{\cdot}10^4$
THB	$2.241{\cdot}10^3$	$2.155{\cdot}10^3$	$2.154{\cdot}10^3$	$2.154{\cdot}10^3$	$5.554{\cdot}10^3$	$2.221{\cdot}10^4$
LR	$2.241{\cdot}10^3$	$2.153{\cdot}10^3$	$2.152{\cdot}10^3$	$5.981{\cdot}10^3$	$3.245{\cdot}10^4$	$1.757{\cdot}10^5$

Table 5: **2D Central Refinement:** The conditioning numbers for $p = 2$ in the various iterations of the central refinement. Stiffness Matrix above, Mass Matrix below.

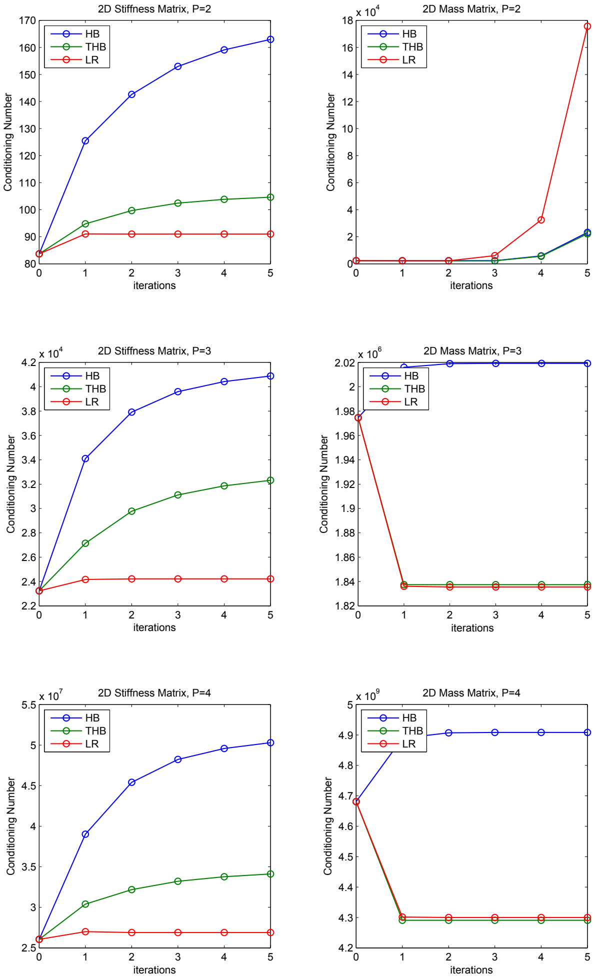
improvement gained by using the Truncated basis is less than in the 1D case, we have to take into consideration that the refined region is very small.

Conditioning Numbers Figure 24 shows the plots of the conditioning numbers of the stiffness and mass matrices produced by the refinement procedure described above. As we can see, the conditioning numbers for the mass matrices are bounded for the higher degrees.

Tables 5 and 6 contain the numerical values of the conditioning numbers for $p = 2$ and 3, respectively.

Spectrum Figure 25 shows the spectra of the matrices obtained from the last refinement iteration using biquadratic basis functions. While there is no substantial difference in the eigenvalue distribution produced by the three refinement methodologies, we can see how the smallest eigenvalue of the mass matrix coming from LR B-splines functions is lower than its Hierarchical counterparts. This explains the higher conditioning number for LR B-splines than the Hierarchical refinement schemes.

Increasing the polynomial degree to $p = 4$ compacts the spectrum, reducing the lower eigenvalues but also the higher ones. As in the univariate case, the smallest eigenvalues are outliers and assume almost the exact same value for all three families of splines; the difference in the magnitude of the conditioning numbers is therefore dictated by the values of the greatest eigenvalues.


2D Stiffness Matrix, P=2

2D Mass Matrix, P=2

2D Stiffness Matrix, P=3

2D Mass Matrix, P=3

2D Stiffness Matrix, P=4

2D Mass Matrix, P=4

Figure 24: **2D Central Refinement:** Graphs of the conditioning numbers of stiffness matrices (left column) and mass matrices (right column) for the bivariate central refinement. $p = 2$ (top), $p = 3$ (middle), and $p = 4$ (bottom).

Stiffness Matrix

Iter.	0	1	2	3	4	5
HB	$2.323{\cdot}10^4$	$3.410{\cdot}10^4$	$3.792{\cdot}10^4$	$3.960{\cdot}10^4$	$4.043{\cdot}10^4$	$4.088{\cdot}10^4$
THB	$2.323{\cdot}10^4$	$2.714{\cdot}10^4$	$2.977{\cdot}10^4$	$3.112{\cdot}10^4$	$3.186{\cdot}10^4$	$3.231{\cdot}10^4$
LR	$2.323{\cdot}10^4$	$2.416{\cdot}10^4$	$2.421{\cdot}10^4$	$2.421{\cdot}10^4$	$2.421{\cdot}10^4$	$2.421{\cdot}10^4$

Mass Matrix

Iter.	0	1	2	3	4	5
HB	$1.975{\cdot}10^6$	$2.016{\cdot}10^6$	$2.019{\cdot}10^6$	$2.019{\cdot}10^6$	$2.019{\cdot}10^6$	$2.019{\cdot}10^6$
THB	$1.975{\cdot}10^6$	$1.837{\cdot}10^6$	$1.837{\cdot}10^6$	$1.837{\cdot}10^6$	$1.837{\cdot}10^6$	$1.837{\cdot}10^6$
LR	$1.975{\cdot}10^6$	$1.836{\cdot}10^6$	$1.836{\cdot}10^6$	$1.836{\cdot}10^6$	$1.836{\cdot}10^6$	$1.836{\cdot}10^6$

Table 6: **2D Central Refinement:** The conditioning numbers for $p = 3$ in the various iterations of the central refinement. Stiffness Matrix above, Mass Matrix below.

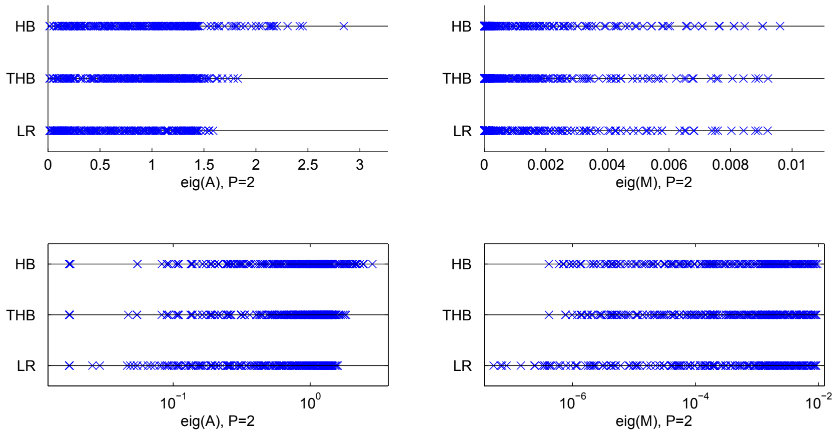


Figure 25: **2D Central Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 2$ at the last (5th) iteration of the central refinement, bivariate case. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

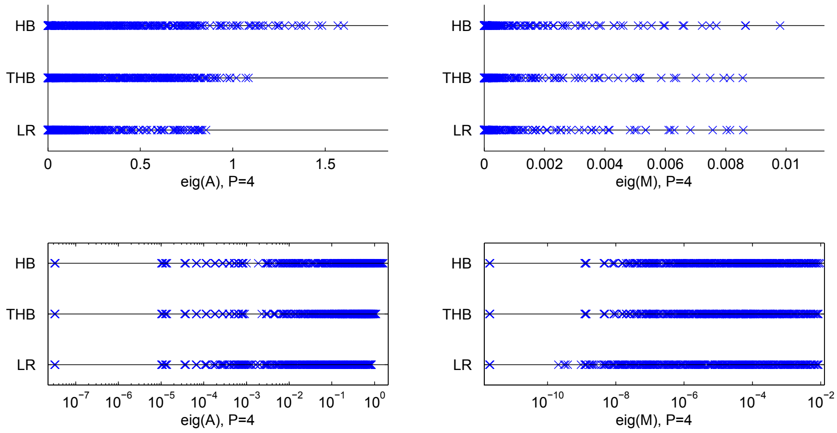


Figure 26: **2D Central Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 4$ at the last iteration of the central refinement, bivariate case. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

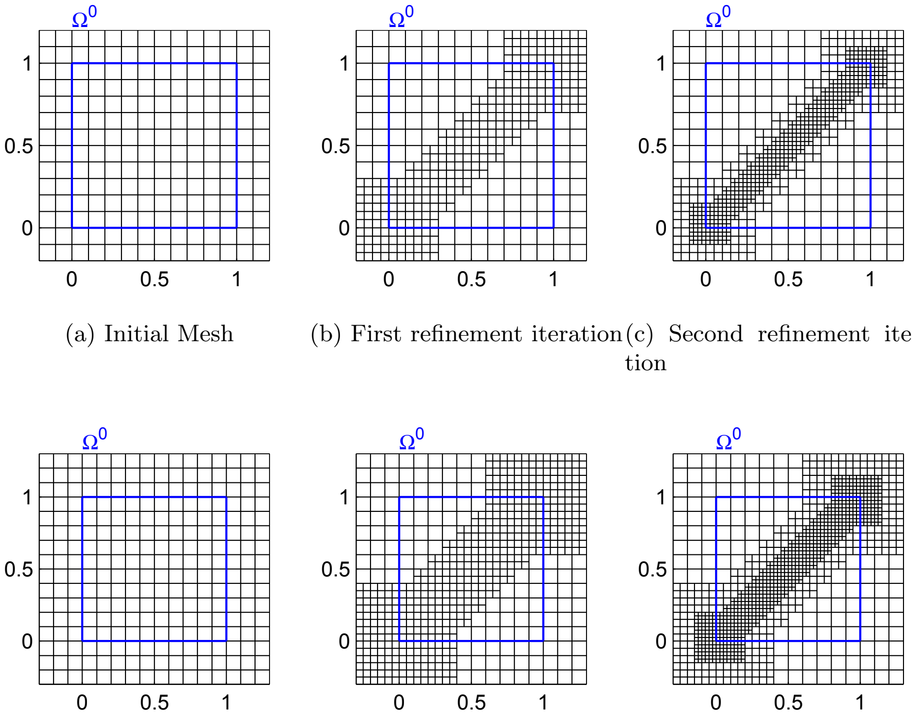
2D Example: Diagonal Refinement

We present here the results obtained applying a diagonal refinement. This configuration is a classical benchmark often used in publications and, since the refinement area is quite large, it provides a different point of view respect to the central refinement illustrated before.

Starting from the usual uniform tensor product mesh, we applied four refinement iterations where we refine at each step all the basis functions along the diagonal. As in the centre refinement case, we considered the polynomial degrees $p = 2, 3$, and 4. Again, the ghost domain was adjusted in order for the first tensor product basis to constitutes a partition of unity in Ω^0. Figure 27 shows the first three meshes in the biquadratic and bicubic cases. Note that we refined also a portion of the ghost domain: This was done in order to avoid having T-joints on the boundary of Ω^0. The integration, however, was carried out only for the elements inside Ω^0, as in the previous cases.

Sparsity Due to the extension of the refinement region, and the number of overlapping zones, we expected to see quite a difference in the sparsity pattern of the matrices produced by the different spline technologies. Figure 28 presents the sparsity patterns for $p = 2$ and $p = 4$. The results are presented in Table 7.

As we can see, due to the larger refined area the use of Truncated Hier-



(a) Initial Mesh

(b) First refinement iteration

(c) Second refinement iteration

(d) Initial Mesh

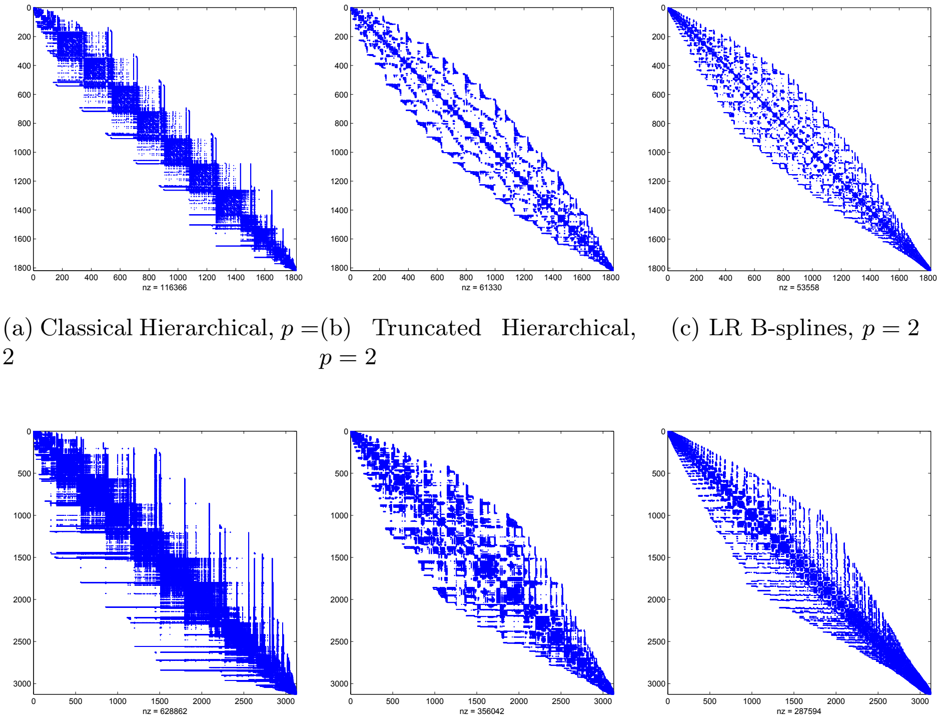
(e) First refinement iteration

(f) Second refinement iteration

Figure 27: **2D Diagonal Refinement:** The first three steps of the refinement process in the cases $p = 2$ (above) and $p = 3$ (below).

p	HB	THB	LR	H/T	H/LR
2	116366	61330	53558	190%	217%
3	304671	164039	140047	186%	218%
4	628862	356042	287594	177%	219%

Table 7: **2D Diagonal Refinement:** Number of non-zero elements in the stiffness matrices at the last refinement iteration of the bivariate diagonal refinement. The last two columns present the ratios, rounded to the nearest percent point.



(a) Classical Hierarchical, $p = 2$

(b) Truncated Hierarchical, $p = 2$

(c) LR B-splines, $p = 2$

(d) Classical Hierarchical, $p = 4$

(e) Truncated Hierarchical, $p = 4$

(f) LR B-splines, $p = 4$

Figure 28: **2D Diagonal Refinement:** The sparsity patterns of the stiffness matrices at the last refinement iteration for the 2D diagonal refinement. The Cuthill-McKee Algorithm has been applied to optimize the bandwidth. Top: $p = 2$. Bottom: $p = 4$.

Stiffness Matrix

Iter.	0	1	2	3	4
HB	83.6793	139.708	169.439	225.641	318.5089
THB	83.6793	97.8692	173.2625	371.711	772.2455
LR	83.6793	95.6921	163.0508	346.4521	716.4894

Mass Matrix

Iter.	0	1	2	3	4
HB	$2.241{\cdot}10^3$	$8.783{\cdot}10^3$	$3.511{\cdot}10^4$	$1.404{\cdot}10^5$	$5.617{\cdot}10^5$
THB	$2.241{\cdot}10^3$	$8.187{\cdot}10^3$	$3.275{\cdot}10^4$	$1.310{\cdot}10^5$	$5.240{\cdot}10^5$
LR	$2.241{\cdot}10^3$	$8.161{\cdot}10^3$	$3.264{\cdot}10^4$	$1.306{\cdot}10^5$	$5.223{\cdot}10^5$

Table 8: **2D Diagonal Refinement:** The conditioning numbers for $p = 2$ in the various iterations of the diagonal refinement. Stiffness Matrix above, Mass Matrix below.

archical or LR B-splines basis functions has a huge impact on the sparsity of the matrices: The Classical Hierarchical basis produces almost twice as many non-zero elements as the Truncated basis, and more than twice those of the LR B-splines basis. It also seems that increasing the polynomial degree somewhat reduces the advantage of the Truncated basis, while the LR B-splines basis maintains the same ratio.

Conditioning Numbers Figure 29 shows the conditioning numbers of the stiffness and mass matrices obtained for this diagonal example. The numerical values of the conditioning numbers for the stiffness and mass matrices are presented in Table 8 for the biquadratic case, and Table 9 for the bicubic case.

Spectrum Figures 30 and 31 present the spectrum of the stiffness and mass matrices in the cases $p = 2$ and 4, respectively. As before, the magnitude of the smallest eigenvalues is the same for all three types of basis functions considered. The value of the conditioning numbers depends therefore from the values of the highest eigenvalues, which is typically greater for the Classical Hierarchical functions.

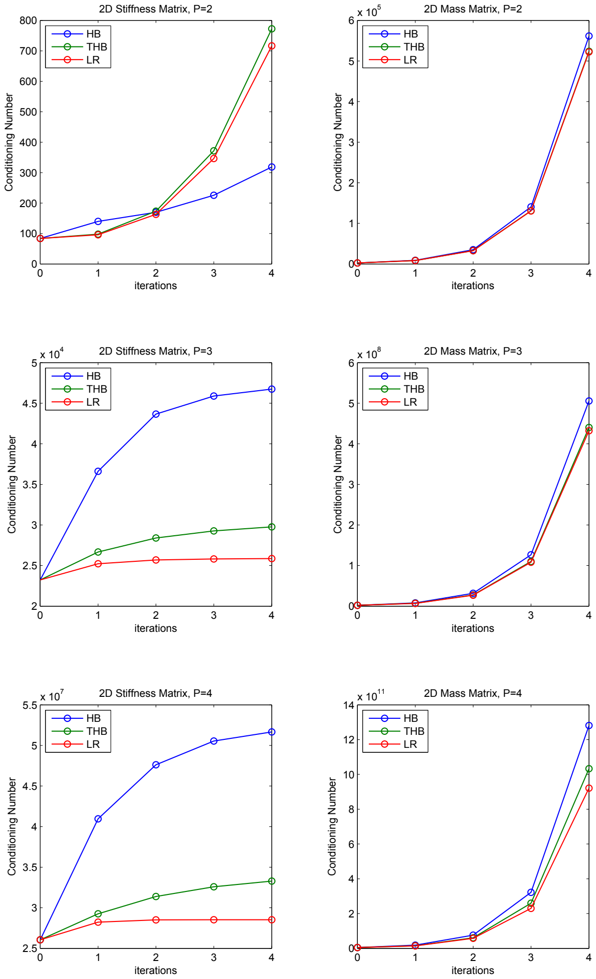


Figure 29: **2D Diagonal Refinement:** Graphs of the conditioning numbers of stiffness matrices (left column) and mass matrices (right column) for the bivariate diagonal refinement. $p = 2$ (top), $p = 3$ (middle), and $p = 4$ (bottom).

Stiffness Matrix

Iter.	0	1	2	3	4
HB	$2.323{\cdot}10^4$	$3.661{\cdot}10^4$	$4.366{\cdot}10^4$	$4.588{\cdot}10^4$	$4.675{\cdot}10^4$
THB	$2.323{\cdot}10^4$	$2.666{\cdot}10^4$	$2.840{\cdot}10^4$	$2.927{\cdot}10^4$	$2.977{\cdot}10^4$
LR	$2.323{\cdot}10^4$	$2.522{\cdot}10^4$	$2.569{\cdot}10^4$	$2.581{\cdot}10^4$	$2.585{\cdot}10^4$

Mass Matrix

Iter.	0	1	2	3	4
HB	$1.975{\cdot}10^6$	$7.885{\cdot}10^6$	$3.160{\cdot}10^7$	$1.264{\cdot}10^8$	$5.057{\cdot}10^8$
THB	$1.975{\cdot}10^6$	$6.876{\cdot}10^6$	$2.750{\cdot}10^7$	$1.100{\cdot}10^8$	$4.400{\cdot}10^8$
LR	$1.975{\cdot}10^6$	$6.753{\cdot}10^6$	$2.701{\cdot}10^7$	$1.080{\cdot}10^8$	$4.321{\cdot}10^8$

Table 9: **2D Diagonal Refinement:** The conditioning numbers for $p = 3$ in the various iterations of the diagonal refinement. Stiffness Matrix above, Mass Matrix below.

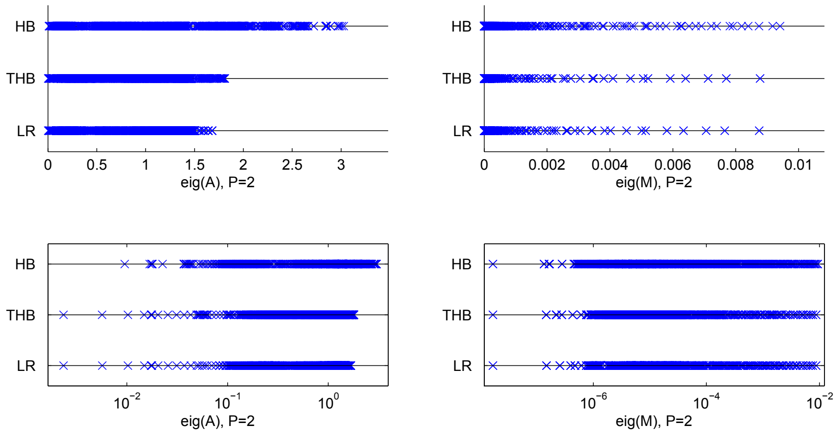


Figure 30: **2D Diagonal Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 2$ at the last iteration of the diagonal refinement. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

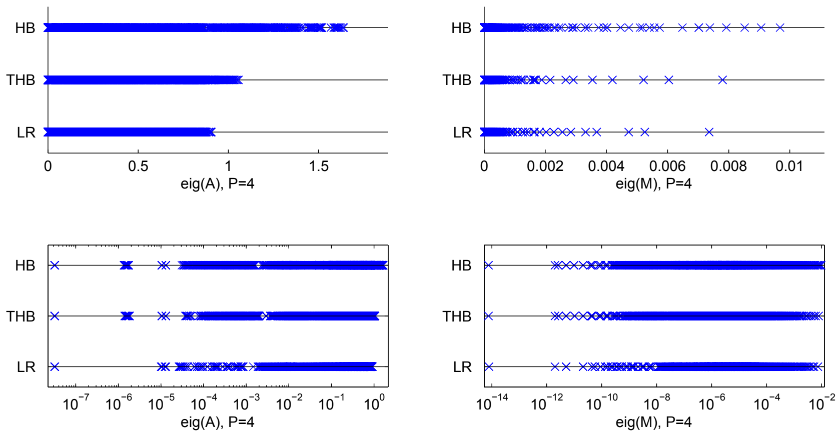


Figure 31: **2D Diagonal Refinement:** The eigenvalues of the Stiffness Matrix (left) and Mass Matrix (right) for $p = 4$ at the last iteration of the diagonal refinement. The plots are shown on a linear scale (top) and logarithmic scale (bottom). The zero eigenvalue of the Stiffness Matrix is omitted.

4.3 Additional Results

In this section we present some additional results. We feel it is unnecessary to give the same level of detail as in the previous examples, hence only the meshes and corresponding graphs of the conditioning numbers are shown. We feel that this gives a solid base of different refinement types that may appear in applications. Ranging from refinement around points, to curves to areas.

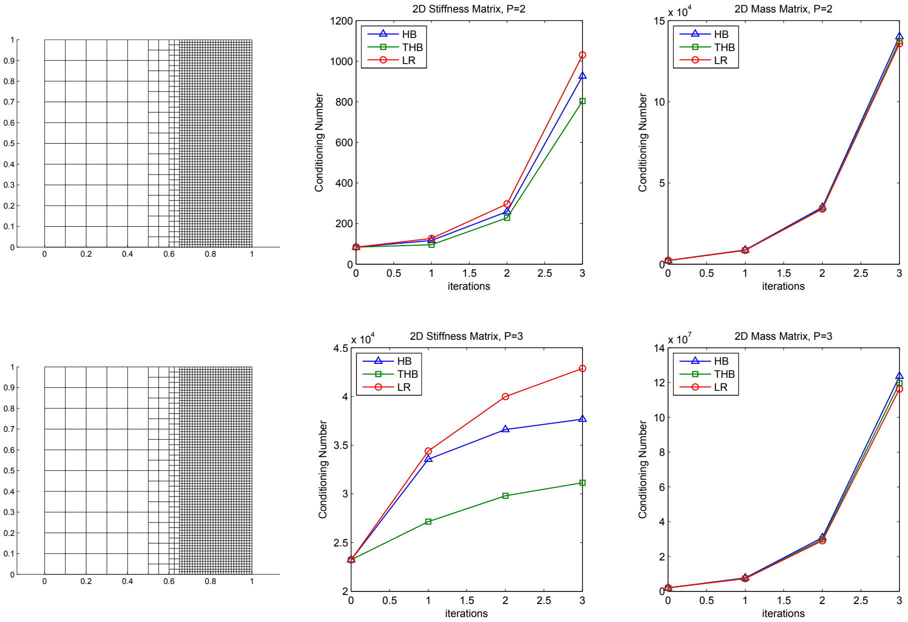


Figure 32: **Half-Side Refinement:** Refinement conducted only in the right half of the mesh. Top: Final mesh and conditioning numbers for $p = 2$. Bottom: Final mesh and conditioning numbers for $p = 3$.

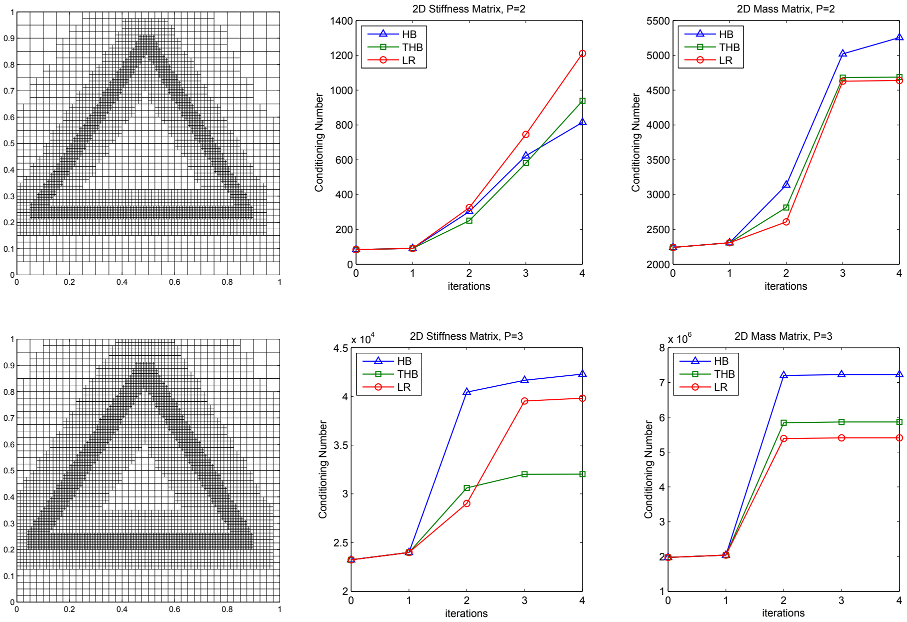


Figure 33: **Triangle Refinement:** Refinement along a triangular path. Top: Final mesh and conditioning numbers for $p = 2$. Bottom: Final mesh and conditioning numbers for $p = 3$.

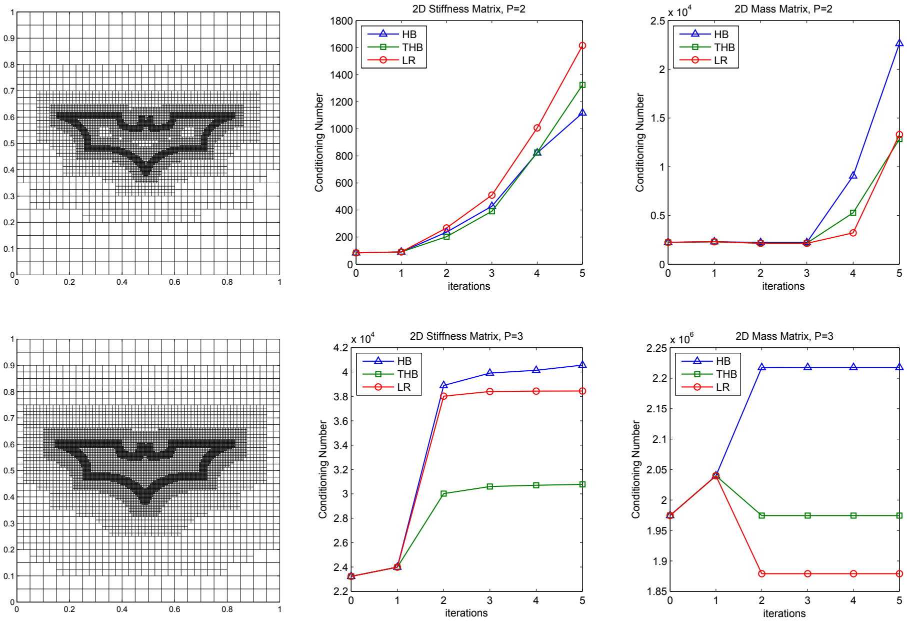


Figure 34: **Logo Refinement:** Refinement along a familiar logo. Top: Final mesh and conditioning numbers for $p = 2$. Bottom: Final mesh and conditioning numbers for $p = 3$.

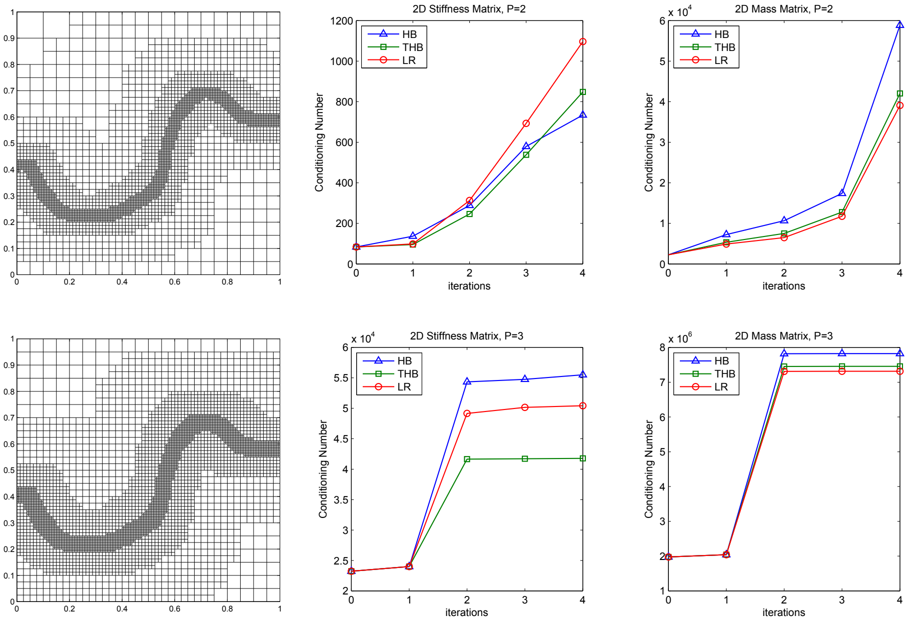


Figure 35: **Curve Refinement:** Refinement along a predefined curve. Top: Final mesh and conditioning numbers for $p = 2$. Bottom: Final mesh and conditioning numbers for $p = 3$.

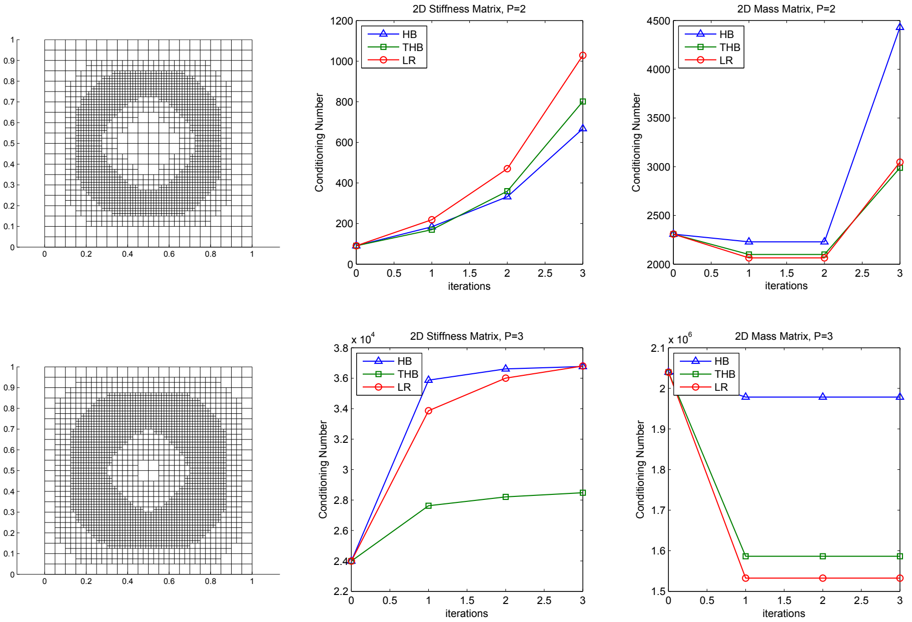


Figure 36: **Circle Refinement:** Refinement along a unit circle. Top: Final mesh and conditioning numbers for $p = 2$. Bottom: Final mesh and conditioning numbers for $p = 3$.

5 Future work

One of the main drawbacks of the Classical Hierarchical basis described in Definition (3) is that it does not preserve partition of unity. This is a very important property of the basis functions used in Isogeometric Analysis as it is linked to the stability of the basis, seen as a relationship between a spline function f and its control points, [17]; moreover, it is also needed for consistency i.e. the need to be able to represent a constant in order to cover the nil space of the corresponding bilinear operator (the same as rigid body translations in elasticity). Due to this, it is normally preferred to use basis functions that maintain partition of unity at all stages of refinement.

As briefly stated in Section 3, one can easily recover the partition of unity of the Hierarchical basis by weighting the functions appropriately through the use of Equation (15). Doing this, however, led to some peculiar results in our experiments. Figure 37 shows the same conditioning numbers as in the last row of Figure 19, the 1D cases with central refinement where we have included also the data of the Weighted Hierarchical basis. Figure 38 shows the corresponding spectrum for this case. Please note that the line corresponding to the Classical Hierarchical basis is now light-blue, while the deep-blue is the Weighted Hierarchical basis.

For a 2D example, Figure 39 shows the same data of the bivariate central refinement presented in Figure 24 for the case $p = 3$. We have included also the data for the weighted Hierarchical basis.

As we can see, the Weighted Hierarchical basis has a drastically different behaviour respect to any of the other spline families considered in this paper, and more research is likely needed.

Another point we would like to investigate more is the strange behaviour we encountered in the numerical examples presented in Section 4 for the polynomial degrees $p = 2$. In almost every case, the pattern which emerged from the experiments is that Truncated Hierarchical and LR B-splines basis are very similar in performance and in particular the general trend for the conditioning number of a matrix X seems to be

$$\mathrm{cond}(X_{LR}) \leqslant \mathrm{cond}(X_T) < \mathrm{cond}(X_H)$$

except for the $p = 2$ cases. We currently do not have a good argument as to why this is happening; a more careful analysis is probably required.

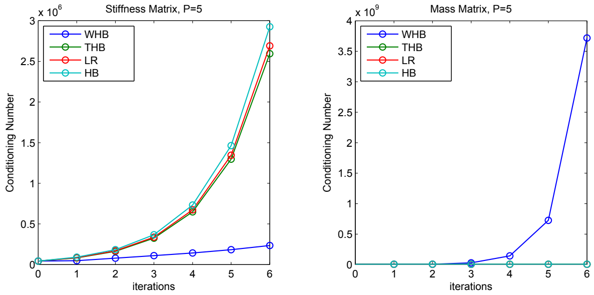


Figure 37: The same plot as in the last row of Figure 19, the 1D example with centre refinement. Here we included also the data for the Weighted Hierarchical basis, which preserves partition of unity.

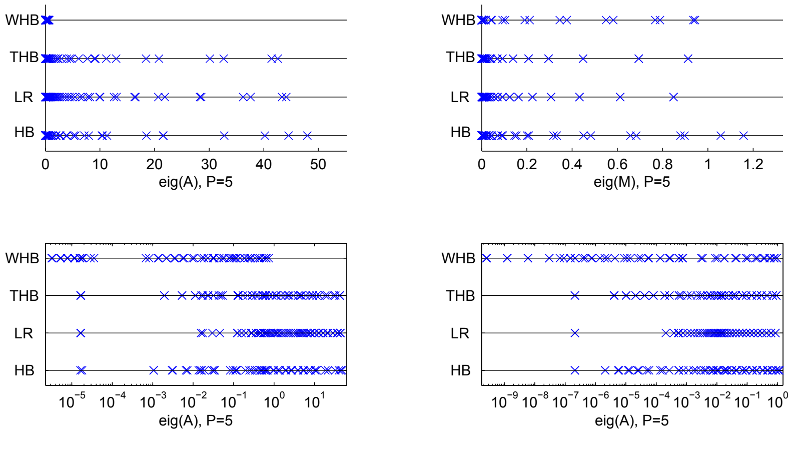


Figure 38: The spectrum corresponding to the 1D centre refinement with $p = 5$ previously presented in Figure 19. We have included also the spectrum for the Weighted Hierarchical basis.

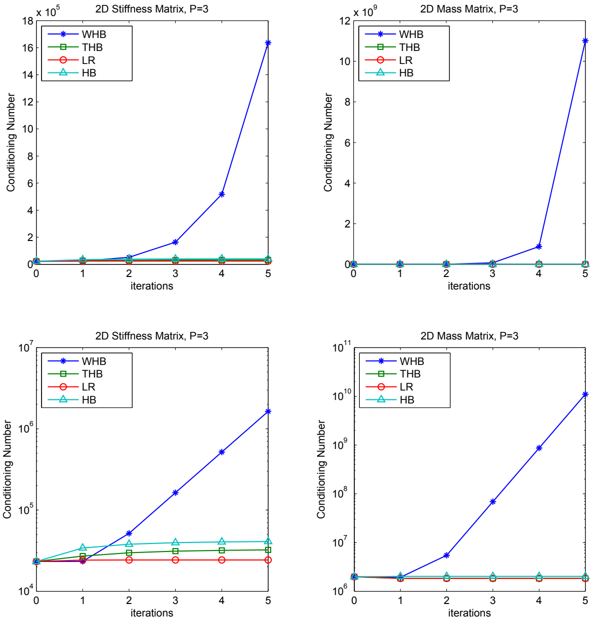


Figure 39: Conditioning numbers for the 2D central refinement example presented in Figure 24, case $p = 3$. We have included also the data for the Weighted Hierarchical Basis.

6 Conclusions

In this paper we have analysed the Classical Hierarchical, Truncated Hierarchical and LR B-splines basis on both a qualitative (more theoretical) and quantitative (more numerical) level. Regarding the qualitative differences we believe that the most important points are:

- The Classical Hierarchical basis does not constitute a partition of unity;
- For some meshes, the basis generated by the Hierarchical B-splines and the structured mesh LR B-spline refinement does indeed produce different function spaces;
- The Hierarchical and LR B-splines frameworks have different admissible meshes;
- While LR B-splines allow for more flexibility regarding the choice of refinement strategies, a formal proof for the linear independence of the resulting set of functions is still lacking.

The difference in the functions spaces is perhaps the most important point. Since both hierarchal and Truncated B-splines span the same space they are both resulting in the same discrete finite element solution, meaning that the differences in the basis functions are going to affect only the number of operations required to get to a certain precision. For LR B-splines versus Hierarchical B-splines, the situation becomes a bit more nuanced as the discrete solution itself can be different.

For the quantitative case we presented several numerical examples which have shown that there is a substantial difference between the three spline families especially for what concerns the sparsity pattern of the matrices. The Classical Hierarchical basis always produced the densest matrices, while those produced by the Truncated Hierarchical and LR B-splines were much more sparse. In particular, it seems that when the refinement region affects only a small portion of the mesh, the Truncated basis yields the best results regarding sparsity; if instead the refinement covers a large portion of the mesh, then the LR B-splines basis produces the most sparse matrices.

When it comes to the conditioning numbers, no clear and defined pattern emerged, and the results seemed very dependant on several factors: the dimension of the problem (1D vs 2D), the matrix considered (Stiffness Matrix vs Mass Matrix) and the polynomial degree. In particular, in the univariate setting we had, for the stiffness matrix,

$$\operatorname{cond}(A_T) < \operatorname{cond}(A_{LR}) < \operatorname{cond}(A_H)$$

while for the mass matrix we had

$$\mathrm{cond}(M_{LR}) < \mathrm{cond}(M_T) < \mathrm{cond}(M_H)$$

In the bivariate setting things are not so definite, but there seem to be a tendency of

$$\mathrm{cond}(M_{LR}) \approx \mathrm{cond}(M_T) < \mathrm{cond}(M_H)$$

for the mass matrix, while the stiffness matrix doesn't seem to show as prominent patterns. An interesting observation is that for low p, the Classical Hierarchical have conditioning numbers of A on par, or below that of Truncated or LR, but for $p > 2$ this is no longer the case.

On a general level we can say that the Classical Hierarchical basis performed worse than the Truncated or the LR one, while these last two frameworks yielded very similar results. Therefore, we conclude that for any application where sparsity or conditioning numbers are important quantities, one of these two refinement schemes are to be preferred.

Acknowledgement

The authors acknowledge the financial support from the Norwegian Research Council and the industrial partners of the ICADA-project (NFR Projects 187993); Ceetron, Det Norske Veritas and Statoil.

Bibliography

[1] M. Ainsworth and J. Oden. *A Posterori Error Estimation in Finite Element Analysis*. 1st. Wiley-Interscience, 2000.

[2] Y. Bazilevs et al. “Isogeometric analysis: approximation, stability and error estimates for h-refined meshes”. In: *Mathematical Models and Methods in Applied Sciences* 16 (2006), pp. 1031–1090.

[3] Y. Bazilevs et al. “Isogeometric analysis using T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 229–263.

[4] M. Borden et al. “A phase-field description of dynamic brittle fracture”. In: *Computer Methods in Applied Mechanics and Engineering* 217–220.0 (2012), pp. 77 –95.

[5] P. Bornemann and F. Cirak. “A subdivision-based implementation of the hierarchical B-spline finite element method”. In: *Computer Methods in Applied Mechanics and Engineering* (2012).

[6] S. Brenner and L. Scott. *The mathematical theory of finite element methods*. Springer, 2005.

[7] A. Bressan. “Some properties of LR-splines”. In: *Comput. Aided Geom. Design* 30.8 (2013), pp. 778–794.

[8] A. Buffa, D. Cho, and M. Kumar. “Characterization of T-splines with reduced continuity order on T-meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 201-204.0 (2012), pp. 112–126.

[9] A. Buffa, D. Cho, and G. Sangalli. “Linear independence of the T-spline blending functions associated with some particular T-meshes”. In: *Computer Methods in Applied Mechanics and Engineering* 199.23-24 (2010), pp. 1437 –1445.

[10] J. Cottrell, T. Hughes, and Y. Bazilevs. *Isogeometric analysis: toward integration of CAD and FEA*. John Wiley & Sons, 2009.

[11] E. Cuthill and J. McKee. “Reducing the Bandwidth of Sparse Symmetric Matrices”. In: *Proceedings of the 1969 24th National Conference.* ACM ’69. New York, NY, USA: ACM, 1969, pp. 157–172.

[12] T. Dokken, T. Lyche, and K. Pettersen. “Polynomial splines over locally refined box-partitions”. In: *Comput. Aided Geom. Des.* 30.3 (Mar. 2013), pp. 331–356.

[13] M. R. Dörfel, B. Jüttler, and B. Simeon. “Adaptive isogeometric analysis by local h-refinement with T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 264 –275.

[14] D. Forsey and R. Bartels. “Hierarchical B-spline refinement”. In: *ACM SIGGRAPH Computer Graphics* 22.4 (1988), pp. 205–212.

[15] K. Gahalaut and S. Tomar. *Condition number estimates for matrices arising in the isogeometric discretizations.* Tech. rep. 23. RICAM, 2012.

[16] C. Garoni et al. “On the spectrum of stiffness matrices arising from isogeometric analysis”. English. In: *Numerische Mathematik* 127.4 (2014), pp. 751–799.

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

[18] C. Giannelli, B. Jüttler, and H. Speleers. “THB-splines: The Truncated basis for hierarchical splines”. In: *Computer Aided Geometric Design* 29.7 (2012), pp. 485 –498.

[19] C. Giannelli and B. Jüttler. “Bases and dimensions of bivariate hierarchical tensor-product splines”. In: *Journal of Computational and Applied Mathematics* 239.0 (2013), pp. 162–178.

[20] T. Hughes. *The finite element method: linear static and dynamic finite element analysis.* Prentice-hall, 1987.

[21] T. Hughes, J. Cottrell, and Y. Bazilevs. “Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement”. In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195.

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

[23] K. Johannessen, T. Kvamsdal, and M. Kumar. “Divergence-conforming discretization for Stokes problem on locally refined meshes using LR B-splines”. In: *Submitted to Computer methods in applied mechanics and engineering* (2014).

[24] C. Johnson. *Numerical solutions of partial differential equations by the finite element method.* Dover Publications, 2009.

[25] G. Kiss, C. Giannelli, and B. Jüttler. *Algorithms and Data Structures for Truncated Hierarchical B-splines.* Tech. rep. 14. Johannes Kepler University, 2012.

[26] R. Kraft. “Adaptive und linear unabhängige Multilevel B-Splines und ihre Anwendungen”. PhD thesis. Stuttgart, 1998.

[27] M. Kumar, T. Kvamsdal, and K. Johannessen. “Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis”. In: *Submitted to Computer methods in applied mechanics and engineering* (2014).

[28] T. Kvamsdal and K. Okstad. “Error estimation based on superconvergent patch recovery using statically admissible stress fields”. In: *International Journal for Numerical Methods in Engineering* 42.3 (1998), pp. 443–472.

[29] X. Li and M. Scott. “Analysis-suitable T-splines: Characterization, refineability, and approximation”. In: *Mathematical Models and Methods in Applied Sciences* 24.06 (2014), pp. 1141–1164. eprint: `http://www.worldscientific.com/doi/pdf/10.1142/S0218202513500796`.

[30] H. Melbø and T. Kvamsdal. “Goal oriented error estimators for Stokes equations based on variationally consistent postprocessing”. In: *Computer methods in applied mechanics and engineering* 192.5 (2003), pp. 613–633.

[31] D. Mokriš, B. Jüttler, and C. Giannelli. “On the completeness of hierarchical tensor-product B-splines”. In: *Journal of Computational and Applied Mathematics* 271.0 (2014), pp. 53–70.

[32] B. Mourrain. “On the dimension of spline spaces on planar T-meshes”. In: *Math. Comp.* 83.286 (2014), pp. 847–871.

[33] N. Nguyen-Thanh et al. “Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids”. In: *Computer Methods in Applied Mechanics and Engineering* 200.21–22 (2011), pp. 1892–1908.

[34] K. Okstad and T. Kvamsdal. “Object-oriented programming in field recovery and error estimation”. In: *Engineering with Computers* 15.1 (1999), pp. 90–104.

[35] K. Okstad, T. Kvamsdal, and K. Mathisen. “Superconvergent patch recovery for plate problems using statically admissible stress resultant fields”. In: *International journal for numerical methods in engineering* 44.5 (1999), pp. 697–727.

[36] A. Quarteroni. *Numerical models for differential problems.* Springer, 2008.

[37] D. Schillinger and E. Rank. “An unfitted hp-adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry”. In: *Computer Methods in Applied Mechanics and Engineering* 200.47 - 48 (2011), pp. 3358 –3380.

[38] D. Schillinger et al. “An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces”. In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012), pp. 116 –150.

[39] M. Scott et al. “Local refinement of analysis-suitable T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 213 (2012), pp. 206–222.

[40] T. Sederberg et al. “T-spline simplification and local refinement”. In: *ACM Transactions on Graphics* 23.3 (2004), pp. 276–283.

[41] T. Sederberg et al. “T-splines and T-NURCCs”. In: *ACM Transactions on Graphics* 22.3 (2003), pp. 477–484.

[42] A. Stahl and T. Kvamsdal. “Post-processing and Visualization Techniques for Isogeometric Analysis Results”. In: *Submitted to Computer methods in applied mechanics and engineering* (2014).

[43] L. B. da Veiga et al. “Analysis-Suitable T-splines are Dual-Compatible”. In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012). Higher Order Finite Element and Isogeometric Methods, pp. 42 –51.

[44] C. Verhoosel et al. “An isogeometric analysis approach to gradient damage models”. In: *International Journal for Numerical Methods in Engineering* 86.1 (2011), pp. 115–134.

[45] C. Verhoosel et al. “An isogeometric approach to cohesive zone modeling”. In: *International Journal for Numerical Methods in Engineering* 87.1-5 (2011), pp. 336–360.

[46] A. Vuong et al. “A hierarchical approach to adaptive local refinement in isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 200.49-52 (2011), pp. 3554–3567.

[47] P. Wang et al. “Adaptive isogeometric analysis using rational PHT-splines”. In: *Computer-Aided Design* 43 (2011), pp. 1438–1448.

[48] Y. Wang, J. Zheng, and H. S. Seah. “Conversion between T-splines and Hierarchical B-splines”. In: *Proceedings of the Eight IASTED International Conference*. Computer Graphics and Imaging. Honolulu, Hawaii, USA, 2005.

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

Paper III

Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis

Mukesh Kumar, Trond Kvamsdal and
Kjetil André Johannessen

Submitted to *Computer Methods in Applied Mechanics and Engineering*, 2014

Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis

Mukesh Kumar, Trond Kvamsdal, and Kjetil André Johannessen

Department of Mathematical Sciences
Norwegian University of Science and Technology, Trondheim, Norway
Department of Applied Mathematics, SINTEF ICT, Norway
e–mail: Mukesh.Kumar@math.ntnu.no, Trond.Kvamsdal@math.ntnu.no,
Kjetil.Johannessen@math.ntnu.no

Abstract

Isogeometric analysis (IGA) based on B-splines or Non-uniform rational B-splines (NURBS) are structured tensor product meshes within each patch [36] and facilitates superconvergence behavior. The recently developed Locally Refined (LR) B-splines [28] and structured adaptive mesh refinement using LR B-splines [39] are consider to be promising candidates to facilitate adaptive superconvergent gradient recovery as they produce local tensor product meshes. The aim of the present article is to develop and verify the efficiency of using superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis.

We start out by addressing the existence of derivative superconvergence points in the computed finite element solution based on B-splines and LR B-splines for our elliptic model problem (1D and 2D Poisson). Then we present a posteriori error estimate inspired by the idea proposed by Zienkiewicz-Zhu [76] where the improved gradient obtained from presented recovery procedures is used, and we also show that our Superconvergent Patch Recovery (SPR) method for the improvement of derivatives fulfills the criteria set out by Ainsworth and Craig in [1] for a *Superconvergent Gradient Recovery Operator*.

The developed a posteriori based adaptive refinement methodology are tested on classical elliptic benchmark problems. The focus is put on optimal convergence rate obtained in the computed solution as well as the effectivity of the proposed error estimators.

1 Introduction

Reliability and efficiency are two major challenges in simulation based engineering. These two challenges may be addressed by error estimation combined with adaptive refinements. A lot of research has been performed on error estimation and adaptive mesh refinement. However, adaptive methods are not yet an industrial tool, partly because the need for a link to traditional Computer Aided Design (CAD)-system makes this difficult in industrial practice. Here, the use of an isogeometric analysis framework introduced by Professor Thomas J. R. Hughes (UT at Austin) and co-workers [36] may facilitate more widespread adoption of this technology in industry, as adaptive mesh refinement does not require any further communication with the CAD system.

Isogeometric analysis (IGA) has been introduced in [36] as an innovative numerical methodology for the discretization of Partial Differential Equations (PDEs), the main idea was to improve the interoperability between CAD and PDE solvers, and to achieve this authors in [36] proposed to use CAD mathematical primitives, i.e. splines and NURBS, also to represent PDE unknowns. The smoothness of splines is useful in improving the accuracy per degree of freedom and solving higher order PDEs via direct approximations. Isogeometric methods have been used and tested on a variety of problems of engineering interests, see [36, 25] and references therein. The development on mathematical front start with h-approximation properties of NURBS in [15], and further studies for hpk-refinements in [66] and for anisotropic approximation in [64]. The recently published review article in Acta Numerica [65] is definitely an advantage in this direction.

Non-uniform rational B-splines (NURBS) are the dominant geometric representation format for CAD. The construction of NURBS are based on a tensor product structure and, as a consequence, knot insertion is a global operation. To remedy this a local refinement can be achieved by breaking the global tensor product structure of multivariate splines and NURBS. In the current literature there are three different ways to achieve local refinements: T-splines, LR splines and hierarchical splines. In this article, we will focus on LR-splines, recently introduced in [28] and further studied in [21, 39]. The reader interested in T-splines and hierarchical are refer to the following references: T-splines have been initially introduced in [59] and their use in isogeometric analysis was first investigated in [17, 29] and later a special class of analysis suitable T-spline is developed in [58]; hierarchical splines have been first introduced in [41] and studied within the isogeometric analysis in the papers [32, 68] and others. Recently, there has been much progress on the topic of the generalization of splines construction which allow for local refinement but an automatic reliable and efficient adaptive refinement routine is still one of the key issues in isogeometric analysis. To achieve a fully automatic refinement routine to solve PDEs problem in adaptive isogeometric analysis the *a posteriori error estimate* is required. This is the subject of current work.

1.1 A posteriori error estimations: an overview

Since 1970s several strategies have been developed to estimate the discretization error of Finite Element (FE) solution. A first posteriori error estimates were introduced by Babuska and Rheinboldt in 1978, see [7, 6]. Since then many different estimation procedures have been introduced. The existing techniques to obtain an energy estimates may be classified into two main categories:

- *Residual based estimates*: The approximate FE solution does not satisfy the governing partial differential equation. This lack of fulfillment is called the residual and the error can be estimated by solving local problems where the load functions are given by the local residuals.
- *Recovery-based estimates*: These estimates employ a projection technique in order to recover a post-processed quantity (usually the stresses) from the FE solution. The error is then estimated by taking the difference between the recovered solution and the FE solution.

The first category of estimates mimics the optimal bounds used to prove the convergence of finite element discretization schemes. For example, an explicit residual based estimates is very easy to implement but it include interpolation constants that are problem dependent and difficult to obtain in general. This makes them less popular among the engineers. While in the implicit residual based approach, a finite element problem with a very fine discretization is solved over each of the local subdomains (either individual element [3, 45], patches of elements [7] or subdomains consists of an element and its neighborhood elements [50]). Depending on how the local problem is linked to the global FE solution different properties of the estimates can be obtained. For instance, the equilibrated element approah, the flux free approach and the constitute relation error, yield estimates that gives an upper bound on the error, while error estimates based on local problem with Dirichlet boundary conditions gives the lower bound on the error [26]. A more detailed discussion about this class of estimates can be found in [2, 3, 67].

The second category of recovery based approach consist of deriving simple smoothing technique that yields a solution field that converges faster than the FE solution. A very popular prototype for such approaches is the Zienkiwics-Zhu estimate (so called ZZ estimate). Initial reference to such estimates can be found in [76], and further development with Superconvergent Patch Recovery (SPR) in [77, 78]. The success of this approach in the engineering community relies on an intuitive mechanical definition and a certain ease of implementation compared to other class of available error estimates, without sacrificing the numerical effectivity. Many contribution also has been devoted to obtain the guaranteed upper bound on the error that some residual based technique offered while retaining the simple implementation of the ZZ-estimates framework. The key idea was that when the recovered stress field is exactly statically admissible, then the ZZ-estimate coincides with the constitutive relation error, and gave the bound on the energy error from above. By following this approach some methodology has been presented in [27, 44, 49] and others, to obtain practical upper bound of the error in energy norm using SPR recovery technique in finite element literature. These smoothing technique were not limited to classical finite element methods but have been extended to enriched approximations in [19, 20] and to smoothed finite elements (SFEM) in [33].

The use of a posteriori error estimator in isogeometric analysis is still in its infancy. To the best of our knowledge only few work has been done in this direction, see [22, 29, 37, 40, 57, 63, 68, 70, 72, 71]. The authors in [29] used the idea of hierarchical bases with bubble functions approach of Bank and Smith [14] to design a posteriori error estimator for T-splines, which was also used in [22, 68]. But their performance was less satisfactory due to the needed saturation assumption as noted on page 41 of [37]. Another simple idea of explicit residual based error estimator has been explored in [37, 63, 70, 72, 71]. They require the computation of constants in Clement-type interpolation operators. Such constant are mesh (element) dependent and often incomputable for general element shape. A global constant can overestimates the local constants, and thus the exact error. Recently, a functional-type a posteriori error estimate for isogeometric discretization is presented in [40]. These type of error estimate, which was introduced in [51, 52] on functional grounds (including integral identity and functional analysis arguments) are applicable for any conforming and non-conforming discretizations and known to provide a guaranteed and computable error bounds. But the hindrance in their popularity is due to high cost of computations which are based on solving a global minimization problem (Majorant minimization problem) in H(div) spaces. In [40], authors made an attempt to to reduce the cost of computations for tensorial spline spaces but the same idea of cost reduction need further study in adaptive isogeometric analysis. In this article we explore another approach to design a posteriori error estimate in setting of Zienkiewicz-Zhu [76] where the improved gradient obtained from recovery procedure is used instead of exact gradient of solution. The recovery based estimators are very popular in engineering community because of their simple implementation and as they provide good effectivity indices. In an extensive study on the quality of different a posteriori error estimates belonging to first two categories above (residual based vs. recovery based), Babuska and co-workers in [10, 9, 13]; conclude that the Superconvergent Patch Recovery (SPR) technique developed by Zienkiewicz and Zhu [77, 78] is the most robust estimator for the class of smooth solutions approximated on patch-wise uniform grids of linear or quadratic elements. In this article, we develop recovery based error estimates for isogeometric discretization and verify their effectiveness for quadratic B-splines and quadratic LR B-splines elements in adaptive isogeometric analysis.

We also address the problem of existence of derivative superconvergence points in the context of B-splines and LR B-splines based Galerkin discretization. The superconvergence in the finite element method (FEM) is a well known phenomenon, where the order of convergence of the finite element error, at certain special points in an element, is higher than the order of convergence of the maximum of the finite element error over that element. These special points are called natural superconvergence points. This phenomena was first address in [48], and the term superconvergence was first used in [30]. Superconvergence has been extensively studied since late 1970s a few references are [4, 12, 11, 31, 42, 43, 46, 53, 54, 61, 62, 74, 73], and several books has been written on superconvergence in the finite element method, e.g., [3, 8, 23, 24, 69, 75]. A systematic *computer based approach* was introduced in [11] for the analysis of superconvergence in the context of the finite element method. It was shown that the existence of natural superconvergence points was equivalent to the existence of roots of a system of polynomial equations. Moreover, the superconvergence points are obtained from these roots, which (the roots) are computed numerically. In special situations, the system of equations can be written explicitly and roots can be computed analytically, as shown in [74, 73]. In this article we follow the main theme of *computer based approach* of [11], but in our approach we involve the computation of local Neumann projection and the computer is used to obtain the location of derivative superconvergence points. We hope that the work presented in this article will initiate more activity on superconvergence in isogeometric discretization and its application in engineering interests.

1.2 Upper error bounds vs. Accurate error estimates

In this section we compare two simple error estimates; an explicit residual based error estimate *vs.* SPR recovery based ZZ-estimate. The main focus will be on the approximation of true error and quality of estimates measure in term of effectivity index θ, which can be defined by the ratio of estimated error by exact FE error.

Let η_{Res} be an explicit residual based error estimate which can be obtained from the Galerkin formulations (20) and (22) of our model problem (17)-(19) after following the standard procedure, [3], and is given by

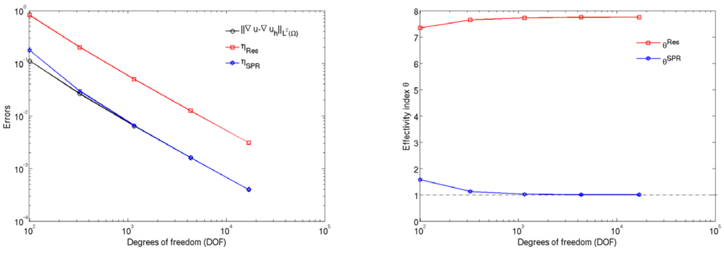
$$\|\nabla u - \nabla u_h\|^2_{L^2(\Omega)} \leq C\eta^2_{Res},$$
$$\eta_{Res} = \sum_{\forall K \in \mathcal{M}} \left(h_K^2 \|r\|^2_{L^2(K)} + \frac{1}{2} h_K \|R\|^2_{L^2(\partial K)} \right), \tag{1}$$

where h_K is the diameter of element $K \in \mathcal{M}$, $r = f + \Delta u_h$ defined the interior residual and R defined the boundary residual $R|_\gamma = g - \frac{\partial u_h}{\partial n}$ for $\gamma \in \partial K \cap \partial \Gamma_N$ and the jump term $R|_\gamma = -\frac{1}{2}\left[\frac{\partial u_h}{\partial n}\right]$ for $\gamma \in \partial K$. The contribution of element jump discontinuity term becomes zeros for smooth spline approximation spaces, which generally have at least C^1-continuity across the element boundaries. The error constant C is generally not known and as a results the bound on the inequality (1) become very conservative. We assume the value of constant C equal to one in the computation of results.

Now we define $\eta_{SPR} := \|\nabla u_h^{SPR} - \nabla u_h\|_{L^2(\Omega)}$ the Superconvergent Patch Recovery (SPR) based error estimate developed in the present article, where ∇u_h^{SPR} is the recovered gradient of the computed FE solution ∇u_h using the SPR recovery procedure of Section 4.2. In Figure 1 we show the comparison between the exact error $\|\nabla u - \nabla u_h\|_{L^2(\Omega)}$, estimated errors η_{Res}, η_{SPR} and the effectivity index θ obtained with the explicit residual based error estimate η_{Res} and the SPR recovery based error estimate η_{SPR} for **Sinus problem** defined by Example 2 of Section 7 using quadratic splines based FE method with uniform h-refinements. The comparison of exact and estimated errors for **L-shaped domain** problem with singularity at the corner $(0, 0)$ defined by Example 8 of Section 7 for both error estimates with adaptive refinements of LR B-splines are shown in Figure 2. Both examples show the accurate estimation of the error in case of SPR recovery procedure in comparison to the upper bound on the error achieved by the explicit residual based error estimator. The SPR recovery based error estimator is known for their h-asymptotic exactness behavior, that means, when the mesh is refined the estimated error converges to the exact error and provide a very accurate approximation of it. This is the main highlight in the present article.

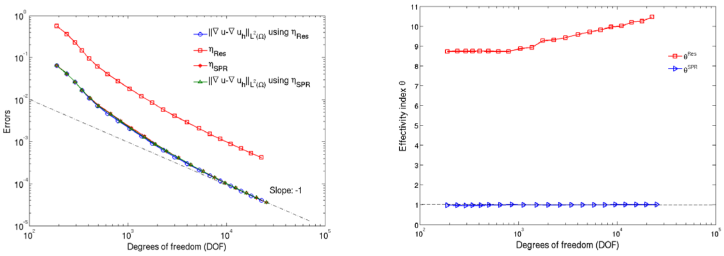
1.3 Aim and outline of the article

The aim of this article is to develop efficient gradient recovery techniques and a *posteriori* error estimation in adaptive isogeometric analysis. We present



(a) Errors (b) Effectivity index θ

Figure 1: **Sinus problem**: Comparison of errors and effectivity index between *residual based error estimate* (η_{Res}) and the present *SPR recovery based error estimate* (η_{SPR}) using quadratic B-splines with uniform h-refinement.



(a) Errors (b) Effectivity index θ

Figure 2: **L-shaped domain problem**: Comparison of errors and effectivity index between *residual based error estimate* (η_{Res}) and the present *SPR recovery based error estimate* (η_{SPR}) using quadratic LR B-splines with adaptive h-refinement.

a posteriori error estimates inspired by the idea introduced by Zienkiewicz-Zhu [76] where the improved gradient obtained through developed recovery procedures is used. We also address the existence of natural derivatives superconvergence points in the approximate solution obtained from spline based finite element method for a model elliptic problem. A computer based proof in the theme of [11, 12] for the computation of these points for the standard case of Poisson and Laplace equations are given. The developed a posteriori based adaptive refinement methodology will be tested on classical elliptic benchmark problems. For the purpose of adaptive isogeometric analysis we consider local h-refinement achieved using *structured mesh* refinement strategy developed in Johannessen et al. [39] based on idea of LR B-splines presented by Dokken et al. [28]. The focus will be on optimal convergence rate obtained in the computed solution as well as the effectivity of the proposed error estimators.

The complete article is organized as follows:

In Section 2, the definitions of B-splines, NURBS and LR B-splines which is necessary to built an approximation spaces in isogeometric analysis is briefly introduced.

In Section 3, a model elliptic problem and its isogeometric FE approximation together with a priori error estimates is introduced. We close the section after developing the idea of a recovery based a posterior error estimation and its asymptotic exactness.

In Section 4, different gradient recovery procedures are developed to improved the derivatives of isogeometric FE solution field. The Superconvergenet patch recovery procedure will be the main focus in this section.

In Section 5, the local behavior of spline based Galerkin discretization is analyzed. The section start with the motivational study of natural superconvergence for one dimensional elliptic problem based on elliptic Ritz projection. Later a more general idea of local Neumann elliptic projection is established, which is suitable for multi-dimensional problems, and based on this we compute the location of true derivative superconvergence points for our model elliptic problems, e.g., Poisson and Laplace equations.

In Section 6, we verify that the SPR recovery procedure of present article satisfy the Abstract Recovery Operator definition (or conditions) of Ainsworth and Craig [1]. These conditions together with superconvergence property of FE approximation is used to show the superconvergence results for the SPR recovery procedure.

Numerical experiments are performed in Section 7.

We end this article with concluding remarks and perspectives based on our findings in Section 8.

2 Approximation spaces in isogeometric analysis

In order to introduce a notation and to provide an overview, we recall the definition and some aspects of isogeometric analysis using B-splines, NURBS and LR B-splines basis functions and their geometry mappings in this section.

2.1 B-splines and NURBS

Given two positive integer p and n, we introduce the (ordered) knot vector

$$\Xi := \{\xi_1, \xi_2, \dots, \xi_{n+p+1}\} \quad \text{with} \quad \xi_i \leq \xi_{i+1} \; \forall i, \tag{2}$$

where p is the degree of the B-spline and n is the number of basis functions (and control points) necessary to describe it. Here we allow repetition of knots, that is, $\xi_i \leq \xi_{i+1}$ $\forall i$. The maximum multiplicity we allow is $p+1$. In the following we will only work with open knot vectors, which means that first and last knots in Ξ have multiplicity $p+1$. Given a knot vector Ξ, univariate B-spline basis functions $B_{i,p}(\xi), i = 1, \dots, n$, are defined recursively by the well known Cox-de Boor recursion formula:

$$B_{i,0}(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1}, \\ 0 & \text{otherwise.} \end{cases} \tag{3}$$

$$B_{i,p}(\xi) = \frac{\xi - \xi_i}{\xi_{i+p} - \xi_i} B_{i,p-1}(\xi) + \frac{\xi_{i+p+1} - \xi}{\xi_{i+p+1} - \xi_{i+1}} B_{i+1,p-1}(\xi) \quad \text{if} \quad \xi_i \leq \xi < \xi_{i+1}, \tag{4}$$

where in (4), we adopt the convention $0/0 = 0$.

Let $B_{i,p}$ and $B_{j,q}$ with $i = 1, \dots, n$ and $j = 1, \dots, m$, are the B-spline basis functions of degree p and q defined by open knot vector $\Xi = \{\xi_1, \xi_2 \dots, \xi_{n+p+1}\}$ and $\Psi = \{\psi_1, \psi_2, \dots, \psi_{m+q+1}\}$, respectively. Then by means of tensor products, a multi-dimensional B-splines can be constructed as $B^{p,q}_{i,j}(\xi, \Psi) = B_{i,p}(\xi) \cdot B_{j,q}(\psi)$. In general, a rational B-spline in $\mathbb{R}^d$ is the projection onto d-dimensional physical space of a polynomial B-spline defined in $(d-1)$-dimensional homogeneous co-ordinate space. Let $C_{ij} \in \mathbb{R}^2$ be the control points and $w_{ij} = (C^w_{ij})_3$ are the positive weights given by projective control points $C^w_{ij} \in \mathbb{R}^3$. Then NURBS basis function on two dimensional parametric space $\hat{\Omega} = [0,1]^2$ are defined as

$$R_{i,j}(\xi,\psi) = \frac{B_{i,p}(\xi)B_{j,q}(\psi)w_{ij}}{\displaystyle\sum_{\hat{i}=1}^{n}\sum_{\hat{j}=1}^{m} B_{\hat{i},p}(\xi)B_{\hat{j},q}(\psi)w_{\hat{i}\hat{j}}} \tag{5}$$

Observe that the continuity and support of NURBS basis function are the same as for B-splines. Furthermore, B-splines can be seen as a special case of NURBS with all weights being equal to one.

2.2 Local h-refinement using LR B-splines

In the following, we present a class of Locally Refined (LR) B-splines space. For a more detailed presentation of LR B-splines, including an overview of corresponding refinement algorithm that results in a proper LR B-spline space to perform structured adaptive refinement in this article, we refer to our previous work in [39].

Local knot vectors

We have seen that a univariate spline basis function is constructed using a recursive formula of (3) and (4) with the global knot vector Ξ. However the support of a B-spline function, $B_{i,p}$, is contained in $[\xi_i, \xi_{i+p+1}]$ and these knots $\{\xi_i, \xi_{i+1} \ldots, \xi_{i+p+1}\}$ only contribute to the definition of $B_{i,p}$. Thus we do not need the global knot vector Ξ to define $B_{i,p}$, instead we consider a *local knot vector*

$$\Xi_i = \{\xi_{i+j}\}_{j=0}^{p+1}, \quad \text{for } i = 1, \ldots, n, \tag{6}$$

and use it in conjunction with (3) and (4) to define $B_{i,p}$, without altering the result in any way. We have illustrated the basis functions given by $\Xi = [0,0,0,1,2,3,3,4,4,4]$ in Figure 3.

The concept of local knot vectors is important for LR B-splines as they are used as the building blocks. Now we recall the concept of knot insertion, that will be the focus of our investigation on LR B-splines, see also [39]. As we are considering the same degree basis in multivariate case so we drop the degree subscript p from the notation $B_{i,p}$.

Knot insertion

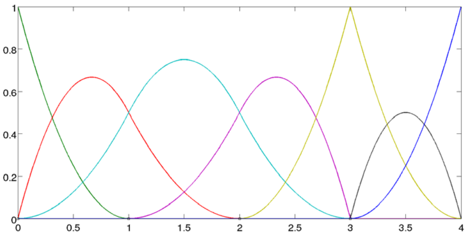


Figure 3: All quadratic basis functions generated by the knot $\Xi = [0,0,0,1,2,3,3,4,4,4]$. Each individual basis function $B_{i,2}$ (represented by different colors) can be described using a local knot vector Ξ_i of length 4 described in (6).

For local h-refinement, we again turn to existing spline theory. Tensor product B-splines form a subset of the LR B-splines and they obey some of the same core refinement ideas. From the tensor product B-spline theory we know that one might insert extra knots to enrich the basis without changing the geometric description. This comes from the fact that we have the available relation between B-splines in the old coarse spline space and in the new enriched spline space. For instance if we want to insert the knot $\hat{\xi}$ into the knot vector Ξ between the knots ξ_{i-1} and ξ_i, then the relation is defined by

$$B_\Xi(\xi) = \alpha_1 B_{\Xi_1}(\xi) + \alpha_2 B_{\Xi_2}(\xi), \tag{7}$$

where

$$\alpha_1 = \begin{cases} 1, & \xi_{p+1} \le \hat{\xi} \le \xi_{p+2} \\ \frac{\hat{\xi}-\xi_1}{\xi_{p+1}-\xi_1}, & \xi_1 \le \hat{\xi} \le \xi_{p+1} \end{cases} \tag{8}$$

$$\alpha_2 = \begin{cases} \frac{\xi_{p+2}-\hat{\xi}}{\xi_{p+2}-\xi_2}, & \xi_2 \le \hat{\xi} \le \xi_{p+2} \\ 1, & \xi_1 \le \hat{\xi} \le \xi_2 \end{cases} \tag{9}$$

and the knot vectors are $\Xi_1 = [\xi_1, \xi_2, ...\xi_{i-1}, \hat{\xi}, \xi_i, ...\xi_{p+1}]$ and
$\Xi_2 = [\xi_2, ...\xi_{i-1}, \hat{\xi}, \xi_i, ...\xi_{p+1}, \xi_{p+2}]$.

Let us look at an example using this technique. Say we want to insert $\hat{\xi} = \frac{3}{2}$ into the B-spline $\Xi = [0,1,2,3]$. This would give us $\alpha_1 = \alpha_2 = \frac{3}{4}$ and the three functions are plotted in Figure 4.

To refine the bivariate B-spline basis function $B_{\Xi,\Psi}(\xi,\psi) = B_\Xi(\xi) \cdot B_\Psi(\psi)$ we consider the refinement of the basis function in one parametric direction at a time. By using the splitting algorithm in (7) when splitting in ξ-

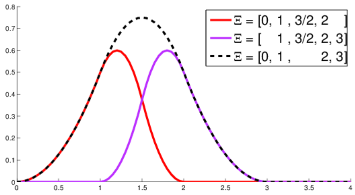


Figure 4: Splitting a B-spline function *via* inserting the knot $\xi = \frac{3}{2}$ in $\Xi = (0, 1, 2, 3)$.

direction, we obtain

$$\begin{aligned} B_{\Xi,\Psi}(\xi,\psi) &= B_{\Xi}(\xi) \cdot B_{\Psi}(\psi) \\ &= (\alpha_1 B_{\Xi_1}(\xi) + \alpha_2 B_{\Xi_2}(\xi)) \cdot B_{\Psi}(\psi) \\ &= \alpha_1 B_{\Xi_1,\Psi}(\xi,\psi) + \alpha_2 B_{\Xi_2,\Psi}(\xi,\psi). \end{aligned}$$

Now we define a weighted B-spline $B^{\gamma}_{\Xi,\Psi}(\xi,\psi) := \gamma B_{\Xi,\Psi}(\xi,\psi)$, where the weight factor $\gamma \in (0, 1]$. This is to ensure that LR B-splines maintain the partition of unity property, and it is noted that the weight factor γ is different from the rational weight w which is common in NURB representation. Refining a bivariate weighted B-splines becomes

$$\begin{aligned} B^{\gamma}_{\Xi,\Psi}(\xi,\psi) &= \gamma B_{\Xi,\Psi}(\xi,\psi) && (10) \\ &= \gamma\alpha_1 B_{\Xi_1,\Psi}(\xi,\psi) + \gamma\alpha_2 B_{\Xi_2,\Psi}(\xi,\psi)) && (11) \\ &= B^{\gamma_1}_{\Xi_1,\Psi}(\xi,\psi) + B^{\gamma_2}_{\Xi_2,\Psi}(\xi,\psi), && (12) \end{aligned}$$

where $B^{\gamma_1}_{\Xi_1,\Psi}$ and $B^{\gamma_2}_{\Xi_2,\Psi}$ are new weighted B-spline basis functions with weights $\gamma_1 = \gamma\alpha_1$ and $\gamma_2 = \gamma\alpha_2$, respectively.

Local refinement algorithm

We now have the main ingredients to formulate the LR B-spline refinement rules. This will be implemented by keeping track of the mesh $\mathcal{M}_\ell$ at level ℓ and the spline space $\mathcal{S}_\ell$. For each B-spline basis $B^{\gamma_k}_{\Xi_k,\Psi_k}$, where k is a single running global index, we store the following:

- Ξ_k, Ψ_k-local knot vectors in the each parametric directions
- γ_k-scaling weights and C_k-control points.

Through the refinement we aim at two points: keeping the partition of unity and leaving the geometric mapping unchanged, i.e. $\sum_{\forall k} B^{\gamma}_{\Xi_k,\Psi_k}(\xi,\psi) =$

1 and $\mathbf{F}(\xi, \psi) = \sum_{\forall k} B^{\gamma}_{\Xi_k,\Psi_k}(\xi, \psi) C_k$ at all levels of refinements.

Assuming a meshline $\mathcal{E}$ is inserted, the refinement process is characterized by two steps.

- **Step 1:** Split any B-spline which support is completely traversed by the *new* meshline - update the weights and control points
- **Step 2:** For all new B-splines, check if their support is completely traversed by any *existing* meshline.

On the basis of that the above characterization is fulfill at each refinement level the following local refinement algorithm (**Algorithm 1**) is proposed in [39] to construct the LR B-spline space. The "Update control points and weight" step is described when a parent basis function B_i split into two newly created B-spline functions B_1 amd B_2 results of splitting by Eq.(10). If B_1 is not present in LR B-spline list then we add it to the list and set its weight and control points equal to its parent function, i.e., $\gamma_1^{new} = \alpha_1 \gamma_i$ and $C_1^{new} = C_i$. While if the newly created function is already exits in our spline space then we just update its control points and weight such as $C_1^{new} := (C_1\gamma_1 + C_i\gamma_i\alpha_1)/(\gamma_1 + \gamma_i\alpha_1)$ and $\gamma_1^{new} := \gamma_1 + \gamma_i\alpha_1$. Finally we remove the old basis functions from the spline space.

We now define an LR spline as an application of the above refinement algorithm.

Definition 1 (LR spline). An LR spline $\mathcal{L}$ consist of $(\mathcal{M}, \mathcal{S})$, where $\mathcal{M}$ is an LR mesh and $\mathcal{S}$ is a set of LR B-splines defined on $\mathcal{M}$, and

- At each refinement level, $\mathcal{M}_{\ell+1} := \mathcal{M}_\ell \cup \mathcal{E}_\ell$, where $\mathcal{E}_\ell$ is a new meshline extension.
- $\mathcal{S}_\ell := \{B_{\Xi_k,\Psi_k}(\xi, \psi)\}_{k=1}^{m}$ is a set of all LR B-splines on $\mathcal{M}_\ell$ as a results of **Algorithm** 1.

In [39] , authors has illustrated two main isotropic h-refinement strategies as shown in Figure 5. A *full span* refinement strategy split an element with a knotline insertion which transverse through the support of every B-splines on the marked elements. The idea of refining elements is a legacy from the finite element method where every inserted vertex would correspond to an additional degree of freedom. With LR B-splines this is not the case as the required length of the inserted meshlines may vary from element to element. Another way of refining LR B-splines is identifying the *B-spline* which needs to be refined as opposed to which elements does, a *structured*

Algorithm 5 Local refinement algorithm

```
**Input parameters**: Spline space ($\mathcal{S}$), LR mesh($\mathcal{M}$), Meshline ($\mathcal{E}$)
**for** every B-spline $B_i \in \mathcal{S}$ **do**
**if** $\mathcal{E}$ traverse support of $B_i$ **then**
    **refine** $B_i$ according to Eq. (10)
    Update control points $C$ and weights $\gamma$
**end if**
**end if**
**end for**
**Update** $\mathcal{S}$ to $\mathcal{S}_{new}$ and $\mathcal{M}$ to $\mathcal{M}_{new}$
**for** every existing $B_i \in \mathcal{S}_{new}$ **do**
**for** every edges $\mathcal{E}_i \in \mathcal{M}_{new}$ **do**
**if** $\mathcal{E}_i$ traverse support of $B_i$ **then**
    **refine** $B_i$ according to Eq. (10)
    Update control points $C$ and weights $\gamma$
    (*These steps may enlarge $\mathcal{S}_{new}$ space further*)
**end if**
**end if**
**end for**
**end for**
```

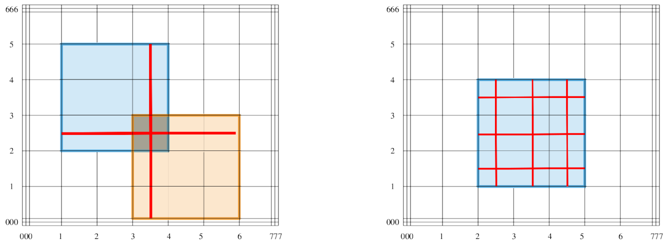
mesh refinement strategy based on this approach is shown in Figure 5(b).

LR spline space properties

Consider an LR spline $(\mathcal{M}, \mathcal{S})$ defined above in Definition 1. Then the following holds true

- $\sum_{\forall k} B^{\gamma}_{\Xi_k,\Psi_k}(\xi, \psi) = 1$, i.e., LR B-splines form a partition of unity.
- $(\mathcal{M}_\ell, \mathcal{S}_\ell) \subset (\mathcal{M}_{\ell+1}, \mathcal{S}_{\ell+1})$, i.e., the LR spline is nested.
- If two meshline insertion sequence $\mathcal{E}$ and $\tilde{\mathcal{E}}$ results in LR spline meshes $\mathcal{M}$ and $\tilde{\mathcal{M}}$ which are equal then the spline spaces $\mathcal{S}$ and $\tilde{\mathcal{S}}$ results on these LR meshes will be equal. This shows LR spline refinement is order independent.
- $\mathcal{S} := \{B_{\Xi_k,\Psi_k}\}_{k=1}^{m}$ does not in general form a linear independent set.

As it has been pointed out that it is not guarantee that an arbitary LR mesh is producing a linear independent set of functions however there is several way to ensure that the system of function is linearly independent,



(a) Full span - split all functions on one element, here only two of all the nine functions with support on this element is depicted

(b) Structured Mesh - split all knot spans on one B-spline, notice that no bad aspect ratio elements are created

Figure 5: The ideas behind the different refinement strategies, here illustrated on a quadratic tensor product mesh. Notice the fundamental difference in that 5a is refining an element, while 5b is refining a B-spline.

see [28, 39]. For the LR spline space obtained as a results of *structured mesh* refinement strategy of Figure 5b, the present authors in [38] has made an attempt to provide a theoretical proof of linear independence on *structured* LR meshes. The work in this article is based on the structured adaptive refinement using LR B-splines presented in this section.

2.3 Geometry mappings

In particular, a single patch domain Ω is a NURBS region associated with the control points C_{ij}, and we introduce the geometrical map $\mathbf{F} : \hat{\Omega} \to \bar{\Omega}$ given by

$$\mathbf{F}(\xi, \psi) = \sum_{i=1}^{n} \sum_{j=1}^{m} C_{ij} R_{i,j}(\xi, \psi). \tag{13}$$

The above equation gives a B-spline region in a special case with all weights being equal to one. For our purpose we assume that the geometry mapping is continuous and bijective which are natural assumption for CAD applications.

Following the isoparametric approach, the space of B-splines and NURBS vector fields on the patch Ω is defined, component by component as the span of the push-forward of their respective basis function, e.g., in case of NURBS

$$V_h = \text{span}\{R_{i,j} \circ \mathbf{F}^{-1}, \quad \text{with} \quad i = 1, \dots, n; \; j = 1, \dots, m\} \tag{14}$$

For LR B-splines, these will instead be defined over a single running global index k using the local knot vectors Ξ_k and Ψ_k (defined by a subsequences of global knot vectors Ξ and Ψ, respectively) by

$$\mathbf{F}(\xi,\psi) = \sum_{k=1}^{N_{dim}} \gamma_k C_k B_{\Xi_k,\Psi_k}(\xi,\psi), \tag{15}$$

where the local knot vectors based spline basis functions are defined by $B_{\Xi_k,\Psi_k}(\xi,\psi) = B_{\Xi_k}(\xi) \cdot B_{\Psi_k}(\psi)$ and γ_k is a weighting factor needed to obtained partition of unity, as discussed in Section 2.2. The isoparametric approach gives the space of LR B-splines vector fields on Ω by

$$V_h = \text{span}\{B_{\Xi_k,\Psi_k}(\xi,\psi) \circ \mathbf{F}^{-1}, \quad \text{with} \quad k = 1,\dots,N_{dim}\}. \tag{16}$$

3 Error estimation

3.1 Model problem

The model problem is Poisson's equation on a open bounded two dimensional domain $\Omega \in \mathbb{R}^2$ with Lipschitz boundary $\Gamma = \Gamma_D \cup \Gamma_N$, where Γ_D and Γ_N are the Dirichlet and Neumann boundaries, respectively. The *strong* form of the boundary value problem: Find the displacement $u : \bar{\Omega} \to \mathbb{R}$ such that

$$\begin{aligned} -\Delta u &= f \text{ on } \Omega; & (17)\\ u &= 0 \text{ on } \Gamma_D; & (18)\\ \mathbf{n}\cdot\nabla u &= g \text{ on } \Gamma_N. & (19) \end{aligned}$$

The data are assumed to be sufficiently smooth, that is, $f \in L^2(\Omega)$, $g \in L^2(\Gamma_N)$ and $\mathbf{n}$ is the unit outward normal vector to Γ. An equivalent formulation of the boundary value problem is the variational formulation seeking $u \in V$ such that

$$a(u,v) = \ell(v) \quad \forall\, v \in V, \tag{20}$$

where the trial and test space V is the usual Sobolev space of functions from $H^1(\Omega)$ whose trace vanishes on the Dirichlet part of the boundary and is define by $V := \{v \in H^1(\Omega) \ : \ v = 0 \text{ on } \Gamma_D\}$.

The form $a(u,v)$ is assumed to be a V-coercive bilinear form on $V \times V$ and the linear functional $l(v)$ is an element of the dual space V', given as

$$a(u,v) = \int_\Omega \nabla u \cdot \nabla v d\Omega \quad \text{and} \quad \ell(v) = \int_\Omega f v d\Omega + \int_{\Gamma_N} g v ds. \tag{21}$$

The existence and uniqueness of the solution to this continuous problem is guaranteed by the Lax-Milgram theorem. The Galerkin finite element approximation to this variational problem may then be given as follow: Given a finite-dimensional subspace $V_h \subset V$ and $\ell \in V'$, find $u_h \in V_h$ such that

$$a(u_h, v_h) = \ell(v_h) \ \ \forall \ v_h \in V_h. \tag{22}$$

In isogeometric setting, the discrete space V_h formed with B-splines/NURBS and LR B-splines are given by (14) and (16), respectively. Let the discrete solution u_h is given by

$$u_h = \sum_{A=1}^{N_{dim}} c_A R_A \tag{23}$$

with the corresponding basis function R_A with respect to B-splines/NURBS or LR B-splines in the physical domain Ω and the control variables c_A. On subsituting this leads to a linear system of equations of the form

$$\mathbf{K}\mathbf{U}_h = \mathbf{f}, \tag{24}$$

where $\mathbf{K}$ is the stiffness matrix induced by the bilinear form $a(\cdot,\cdot)$, $\mathbf{f}$ is the load vector, and $\mathbf{U}_h$ is the coefficient vector of the discrete solution u_h.

Let u and u_h be the exact solution and the isogeometric FE solution of (17)-(19), respectively. The discretization errors are denoted by

$$e(\mathbf{x}) = u(\mathbf{x}) - u_h(\mathbf{x}), \qquad e_\sigma(\mathbf{x}) = \nabla u(\mathbf{x}) - \nabla u_h(\mathbf{x}), \tag{25}$$

where e is the error in the displacement u_h and e_σ is the error in the gradient ∇u_h. We now introduce the following error norms:

$$\|e\|_{L^2(\Omega)} := \|u - u_h\|_{L^2(\Omega)} = \left(\int_\Omega (u - u_h)^2 d\Omega \right)^{1/2} \tag{26}$$

$$\|e_\sigma\|_{L^2(\Omega)} := \|\nabla u - \nabla u_h\|_{L^2(\Omega)} = \left(\int_\Omega (\nabla u - \nabla u_h)^T \cdot (\nabla u - \nabla u_h) d\Omega \right)^{1/2} \tag{27}$$

The associated bilinear form define the the energy norm by

$$\|e\|_E = \sqrt{a(e,e)} = |e|_{H_0^1(\Omega)} = \|e_\sigma\|_{L^2(\Omega)}, \tag{28}$$

which is equivalent to the norm of error e on $H_0^1(\Omega)$ (or the norm of error e_σ on $L^2(\Omega)$).

While we consider here a model problem with a scalar solution, the concepts given in the article are, of course, also applicable to the general linear elasticity problem by using the appropriate vectors of solution variables and the corresponding solution spaces.

3.2 A priori error estimation

FEA

In classical Finite Element Analysis (FEA), the fundamental error estimate for the elliptic boundary value problem, expressed as a bound on the difference between the exact solution, u, and the FEA solution, u_h, takes the form

$$\|u - u_h\|_m \leq Ch^{\beta}\|u\|_r \tag{29}$$

where $\|\cdot\|_k$ is the norm corresponding to the Sobolev space $H^k(\Omega)$, h is a characteristic length scale related to the size of the element in the mesh and $\beta = \min(p+1-m, r-m)$ where p is the polynomial degree of the basis, and C is a constant that does not depend on u and h. The parameter r describe the regularity of the exact solution u and $2m$ is the order of the differential operator of the corresponding PDE.

B-splines and NURBS

In this section we review the basic idea to obtain a priori error estimate results analogous to (29) for NURBS based isogeometric method. For the technical details we encourage the reader to consult the original article [16].

Define a support extension $\bar{Q}$ of an element Q of the mesh Q_h in the parametric domain $\hat{\Omega}$, as a union of the supports of basis functions whose support intersect the element Q. Similarly, we define the physical support extension $\bar{K}$ of an element $K = \mathbf{F}(Q)$ of the physical mesh $\mathcal{M}_h$, as the image of $\bar{Q}$ through the geometric mapping, i.e., $\bar{K} = \mathbf{F}(\bar{Q})$. Given a function $\hat{v} \in L^2(\hat{\Omega})$, let $\Pi_{S_h} : L^2(\hat{\Omega}) \to S_h$ be the projection operator (Quasi-interpolants) on the B-spline space S_h and defined as (see [16]).

$$\Pi_{S_h}\hat{v} := \sum_{k=1}^{N_{dim}} \lambda_k(\hat{v})B_k, \tag{30}$$

where the linear functional $\lambda_k \in L^2(\hat{\Omega})'$ determine the dual basis of B-splines, i.e., $\lambda_k(B_j) := \delta_{k,j},\ \forall\ k, j$. The corresponding projector operator over the NURBS space N_h in the the parametric domain $\hat{\Omega}$, say Π_{N_h}, is defined by means of Π_{S_h} and the definition of the NURBS basis functions through the weighting function w. In particular, $\Pi_{N_h} : L^2(\hat{\Omega}) \to N_h$ reads:

$$\Pi_{N_h}\hat{v} := \frac{\Pi_{S_h}(w\hat{v})}{w}, \quad \forall\ \hat{v} \in L^2(\hat{\Omega}). \tag{31}$$

In this manner, the projection operator on V_h, the NURBS space in physical domain Ω, is given by

$$\Pi_{V_h} : L^2(\Omega) \to V_h, \qquad \Pi_{V_h} v := (\Pi_{N_h}(\hat{v})) \circ \mathbf{F}^{-1}, \;\; \forall \; v \in L^2(\Omega). \tag{32}$$

By following the theoretical results from [16], we have the following interpolation result on the physical domain Ω:

Lemma 1 (Bazilevs et al. [16], Theorem 3.2)**.** *For the projection operator* Π_{V_h} *the following estimate holds for all* $v \in H^\ell(\Omega)$ *with* m *and* ℓ *integers such that* $0 \le m \le \ell \le p+1$

$$\sum_{K \in \mathcal{K}_h} |v - \Pi_{V_h} v|^2_{H^m(K)} \le C_{shape} \sum_{K \in \mathcal{K}_h} h_k^{2(\ell-m)} \sum_{i=0}^{\ell} \|\nabla \boldsymbol{F}\|^{2(\ell-m)}_{L^\infty(\bar{Q})} |v|^2_{H^i(\bar{K})}. \tag{33}$$

The constant C_{shape} depends on p and the shape (but not the size) of the domain Ω, as well as the shape regularity of the mesh.

Now assuming sufficient regularity (for the dual problem), a classical convergence analysis and the duality argument (Aubin-Nitsche's trick) easily give the following result.

Theorem 1. *Let* $u \in H^r(\Omega)$ *be the exact solution of the elliptic boundary value problem and* $u_h \in V_h$ *be the approximate solution obtained with the NURBS based isogeometric discretization (22). Then, the following a priori error estimate holds for* $0 \le m \le r \le p+1$*:*

$$\|u - u_h\|_m \le C_{shape} h^\beta \|u\|_r, \quad \textit{where} \;\; \beta = \min(p+1-m, r-m). \tag{34}$$

For the uniform h-refinement, it is interesting to see from (29) and (34) that the isogeometric solution obtained using C^{p-1} NURBS of degree p can converge at same rate as FEA polynomial of degree p while remaining more efficient in term of degrees of freedom (DOF).

3.3 A posteriori error estimation

The standard a priori error estimate for the exact error given in previous section tells us about the rate of convergence which we can anticipate but is of limited use if we wish to find a numerical estimate of the accuracy. One way in which we might get a realistic estimate or bound upon the discretization error is to use the approximation solution u_h itself in estimating $\|e\|_E$. The idea of using u_h to estimate the error is called *a posteriori error estimation* and some variety of methods to use it have been seen in literature, see [3] and [67] for detailed survey on this topic.

The criterion of what constitutes a good method of using u_h is quantified by the condition of asymptotic exactness of the resulting a posteriori error estimator, introduced by Babuska and Rheinboldt [5].

Definition 2. **[Asymptotic Exactness]** Let η be an a posteriori error estimator, then if under reasonable regularity assumptions on u and the data of the problems, and the family of meshes $\mathcal{M}_h$, we have that

$$\|e\|_E \approx \{1 + O(h^\gamma)\}\eta \quad \text{as} \quad h \to 0, \tag{35}$$

where $\gamma > 0$ is independent of h and the constant in the $O(h^\gamma)$ term depends upon the data of problem only, then we say that η is an asymptotically exact a posteriori error estimator.

This article is motivated from an error estimate procedure developed by Zienkiewicz and Zhu [76] in clasical FE methods, where the Superconvergent Patch Recovery [77, 78] has proved to be effective and economical both in evaluating errors and driving adaptive mesh refinement. We first design and analyze the Superconvergent Patch Recovery procedure to improve the gradient field $\sigma_h^* := \nabla u_h^*$ for B-splines/NURBS based isogeometric FE methods. Then the improved gradient field ∇u^* is used instead of exact solution ∇u in (27), in theme of Zienkiewicz and Zhu [76], to compute the estimated error through

$$\eta = \|\nabla u^* - \nabla u_h\|_{L^2(\Omega)} = \left(\int_\Omega (\nabla u^* - \nabla u_h)^T \cdot (\nabla u^* - \nabla u_h) d\Omega\right)^{1/2}. \tag{36}$$

Effectivity index (θ):

The quality of the error estimate $\eta = \|\nabla u^* - \nabla u_h\|_{L^2(\Omega)}$ is measured by its effectivity index which is given by the ratio of the estimated error to actual error, i.e.,

$$\theta = \frac{\eta}{\|e\|_E} = \frac{\|\nabla u^* - \nabla u_h\|_{L^2(\Omega)}}{\|\nabla u - \nabla u_h\|_{L^2(\Omega)}}. \tag{37}$$

In context to the Definition 2, the error estimator is said to be asymptitically exact if θ approaches unity as the exact error $\|e\|_E$ tends to zero (or as $h \to 0$). Notice that the reliability of the estimator is dependent on the quality of the recovered quantity ∇u_h^* obtained through the recovery procedure. The following result from [78] demonstrate how an asymptotically exact error estimator can be achieved.

Theorem 2. *Suppose* $\|e^*\|_E = \|\nabla u - \nabla u_h^*\|_{L^2(\Omega)}$ *is the error in the recovered solution, then the error estimator* η *is asymptotically exact if*

$$\frac{\|e^*\|_E}{\|e\|_E} \to 0 \qquad as \qquad \|e\|_E \to 0. \tag{38}$$

Thus the condition of asymptotic exactness of the error estimator can be acheived if $\|e^*\|_E$ converges at higher rate than $\|e\|_E$. It follow that if $\|e^*\|_E$ is superconvergent, i.e., $\|e^*\|_E = O(h^{p+\alpha})$ with $\alpha > 0$, in comparison to the discretization error $\|e\|_E = O(h^p)$, then asymptotic exactness is assured and we also get

$$1 - O(h^\alpha) \leq \theta \leq 1 + O(h^\alpha). \tag{39}$$

The recovery procedures developed in this article is claimed to be superconvergent of order 1 in case of uniform refinements and of some order $\alpha \in (0, 1]$ for *structured* LR meshes obtained *via* adaptive h-refinement algorithm of LR B-splines as described in Section 2. We will show some numerical examples to illustrate their superconvergence behavior in Section 7.

It should be noted that while the higher rate of convergence $\|e^*\|_E = O(h^{p+\alpha})$ with $\alpha > 0$ is needed to show asymptotic exactness, the error estimator will always be practically applicable providing the recovered value are more accurate (through not necessary superconvergent) than those obtained in the direct FE computation. If for instance **consistently** we have

$$\frac{\|e^*\|_E}{\|e\|_E} \equiv \delta \leq 0.2 \tag{40}$$

then the effectivity index θ will be within its practical limits of $[0.8, 1.2]$.

4 Gradient recovery techniques : Postprocessing

In this section we present different recovery procedures for the improvement of computed gradient $\boldsymbol{\sigma}_h := \nabla u_h$ in NURBS (or B-splines, LR B-splines) based isogeometric analysis. The gradient $\boldsymbol{\sigma}_h$ computed from direct FE computation u_h in (23) is used to improved the gradient value $\boldsymbol{\sigma}^* = \nabla u^*$ in two different ways, either through global projections over the whole domain Ω or by local smoothing of each gradient components over small patches of elements. We first explain two global recovery procedures termed as Continuous L^2 projection (CL2P) and Discrete least square fitting (DLSF), the computed gradient components of the solution is projected onto the same NURBS (or B-splines, LR B-splines) space that was used for the computation of displacement u_h in FE approximation (22).

4.1 Global recovery procedures

It is possible to obtain more accurate gradient of FE solution by a *projection* or *variational recovery process*. These approaches is originally due to Oden and Brauchli [47] and Hinton and Campbell [35] and has been used used to construct the error estimate in FE stresses [76]. We seek the improved gradient field

$$\boldsymbol{\sigma}^* = \mathbf{R}\hat{\boldsymbol{c}}_\sigma \tag{41}$$

where $\mathbf{R}$ is the matrix corresponding to the functions used in representation of displacement field and $\hat{\mathbf{c}}_\sigma$ is the unknown global vector of required *new* control variables.

Continuous L^2-projection (CL2P)

The improved gradient field $\boldsymbol{\sigma}^*$ defined by (41) is obtained by global L^2-projection, where the unknown control variables $\hat{\boldsymbol{c}}_\sigma$ are now determined by forcing a least square fit of $\boldsymbol{\sigma}^*$ to the computed gradient $\boldsymbol{\sigma}_h$. That is, the functional

$$\mathcal{J}(\hat{\boldsymbol{c}}_\sigma) = \int_\Omega (\boldsymbol{\sigma}^* - \boldsymbol{\sigma}_h)^T \cdot (\boldsymbol{\sigma}^* - \boldsymbol{\sigma}_h) d\Omega \tag{42}$$

is minimized with respect to $\hat{\mathbf{c}}_\sigma$. The minimization of (42) is carried out by letting

$$\frac{\partial \mathcal{J}}{\partial \hat{\boldsymbol{c}}_\sigma} = 2\int_\Omega (\frac{\partial \boldsymbol{\sigma}^*}{\partial \hat{\boldsymbol{c}}_\sigma})^T \cdot (\boldsymbol{\sigma}^* - \boldsymbol{\sigma}_h) d\Omega = \mathbf{0}$$

which yield a linear system

$$\int_\Omega \mathbf{R}^T \mathbf{R} d\Omega \hat{\boldsymbol{c}}_\sigma = \int_\Omega \mathbf{R}^T \boldsymbol{\sigma}_h d\Omega \quad \text{or} \quad \mathbf{A}\hat{\boldsymbol{c}}_\sigma = \mathbf{B}_\sigma, \tag{43}$$

where

$$\mathbf{A} = \int_\Omega \mathbf{R}^T \mathbf{R} d\Omega \qquad \text{and} \qquad \mathbf{B}_\sigma = \int_\Omega \mathbf{R}^T \boldsymbol{\sigma}_h d\Omega.$$

The above process is called global L^2 projection because $\boldsymbol{\sigma}^*$ is a field that is obtained by projecting the computed gradient components $\boldsymbol{\sigma}_h$ onto the same function space as used for the displacement u_h.

The size of global smoothing matrix $\mathbf{A}$ depends on the number of control variables and it has the sparsity pattern as defined by the support of basis functions. In fact, it is much same as the mass matrix used in the problems of dynamics. We use here the full Gauss-quadrature points to solve the system (43) and the cost involving in it has therefore the same growth rate as the original stiffness equation system. However the global projection here still only a fraction of the total cost of the recomputed solution on some refined mesh.

Discrete least square fitting (DLSF)

The improved gradient field $\boldsymbol{\sigma}^*$ defined by (41) is obtained by global discrete least square fitting, where the unknown control variable $\hat{\boldsymbol{c}}_\sigma$ are now determined by ensuring a least square fit of (41) to the set of superconvergent or at least high accuracy sampling points existing in each knot element of a single patch domain considered. We define this procedure for a single patch domain. The procedure for computational domain constructed of several multi-patch domains can be defined for each patch separately. We minimize

$$\mathcal{H}(\hat{\boldsymbol{c}}_\sigma) = \sum_{k=1}^{N_{total}} \left(\boldsymbol{\sigma}^*(\mathbf{x}_k) - \boldsymbol{\sigma}_h(\mathbf{x}_k)\right)^2, \tag{44}$$

where $\boldsymbol{\sigma}_h$ is the gradient component obtained from isogeometric discretization and N_{total} is the total number of the optimal sampling points in each single patch of computational domain. By substituting from (41) into (44) it follows

$$\mathcal{H}(\hat{\boldsymbol{c}}_\sigma) = \sum_{k=1}^{N_{total}} \left(\mathbf{R}_k^T \hat{\boldsymbol{c}}_\sigma - \boldsymbol{\sigma}_h(\mathbf{x}_k)\right)^2. \tag{45}$$

The minimization of (45) is carried out by letting

$$\frac{\partial \mathcal{H}}{\partial \hat{\boldsymbol{c}}_\sigma} = 0 \quad \Rightarrow \quad \mathbf{A}\hat{\boldsymbol{c}}_\sigma = \mathbf{B}_\sigma, \tag{46}$$

with

$$\mathbf{A} = \sum_{k=1}^{N_{total}} \mathbf{R}_k^T \mathbf{R}_k, \qquad \text{and} \qquad \mathbf{B}_\sigma = \sum_{i=1}^{N_{total}} \mathbf{R}_k^T \boldsymbol{\sigma}_h(\mathbf{x}_k).$$

The above process is called discrete least square fitting of the computed gradient components $\boldsymbol{\sigma}_h$ onto the same NURBS (or B-spline) function space as used for the displacement $\mathbf{u}_h$.

Now having the new control variables of the improved gradient $\boldsymbol{\sigma}^*$, the related surface can be constructed and the same FE implementation routine can be used for the computation of smooth gradient field and the error quantities.

It should be noted that the DLSF procedure in this article will be valid only for spline or NURBS elements in isogeometric analysis, while in classical C^0-Lagrange finite elements the present DLSF procedure can not be defined because the total number of optimal sampling points N_{total} needed to perform least square fitting (where either the reduced integration points or Barlow points are chosen) will be less than the total degrees of freedom

N_{dim}. In contrary, one can always define a local/global discrete least fitting procedure which will be valid provided it has enough sampling points, in that case the full $(p+1) \times (p+1)$ Gauss quadrature points has to be consider in each elements, and such types of smoothing procedures are presented in [35].

4.2 Superconvergent patch recovery (SPR)

This section is inspired by the original Superconvergent Patch Recovery(SPR) procedure of [77] and its main idea of existence of some points with high accuracy, i.e., derivatives superconvergent points. As the existence and location of such superconvergent points in isogeometric analysis is not known in literature so far. Thus we decide to use the term sampling points of high accuracy instead of *true* derivatives superconvergence points for the SPR recovery procedure in this section. In Section 5 we will discuss the existence and location of true derivatives superconvergence points for one and two dimensional elliptic model problem and finally the computation based on these points are shown in Section 7.

The idea of original SPR procedure of [77] is to improve the gradient value of the computed FE solution at nodal points. To improve the component of the gradient at a node an *element patch* is defined, usually consisting of all elements to which the nodes belongs. Now, a polynomial function is defined globally consisting of the monomials used for the shape function of the elements at stake. The coefficients of the polynomial are defined such that the polynomial matches the component of the gradient as much as possible at the reduced integration (or superconvergence) points of the patch (in a least squares sense). Finally, an improved gradient in the node is obtained by evaluating this polynomial. This is done for all gradient components separately.

The SPR procedure in this article is explained in three main steps; (*i*) Path recovery procedure, (*ii*) Element path configuration, and (*iii*) Global recovered field representation. In the first step we consider a local least square fitting procedure similar to the original SPR procedure of [77]. The patch element configuration here will differ from the element patch in classical FEM, here we formed an element patch with respect to the support of each basis functions of the B-spline/NURBS space, as the displacement basis function in B-spline/NURBS based isogeometric analysis is not interpolatry in nature. The conjoining of polynomial expansion is consider to get the global representation of recovered field in displacement spaces where a weighting argument based on partition of unity of displacement basis functions is used.

Patch recovery procedure

We explain a local smoothing procedure for the improved gradient component

$$\sigma_\alpha^* = \mathbf{P}(\mathbf{x})\mathbf{a}_\alpha \tag{47}$$

where $\mathbf{P}$ is a matrix of monomials, at least of same degree as displacement space, in the Cartesian co-ordinates $\mathbf{x}$ on the patch of elements and $\mathbf{a}_\alpha$ is the vector of unknown coefficients with the component $\alpha = x, y$. The coefficients a_α are then determined from a least square fit of the field σ_α^* to the values of computed σ_α^h at given sampling points $\{\mathbf{x}_i\}_i^{n_{sp}^{elp}}$ within each element patch, i.e., we minimize the following

$$\mathcal{F}(a_\alpha) = \sum_{i=1}^{n_{sp}^{el}} (\sigma_{\alpha,i}^* - \sigma_{\alpha,i}^h)^T (\sigma_{\alpha,i}^* - \sigma_{\alpha,i}^h) \tag{48}$$

The stationary condition for $\mathcal{F}(a_\alpha)$, gives

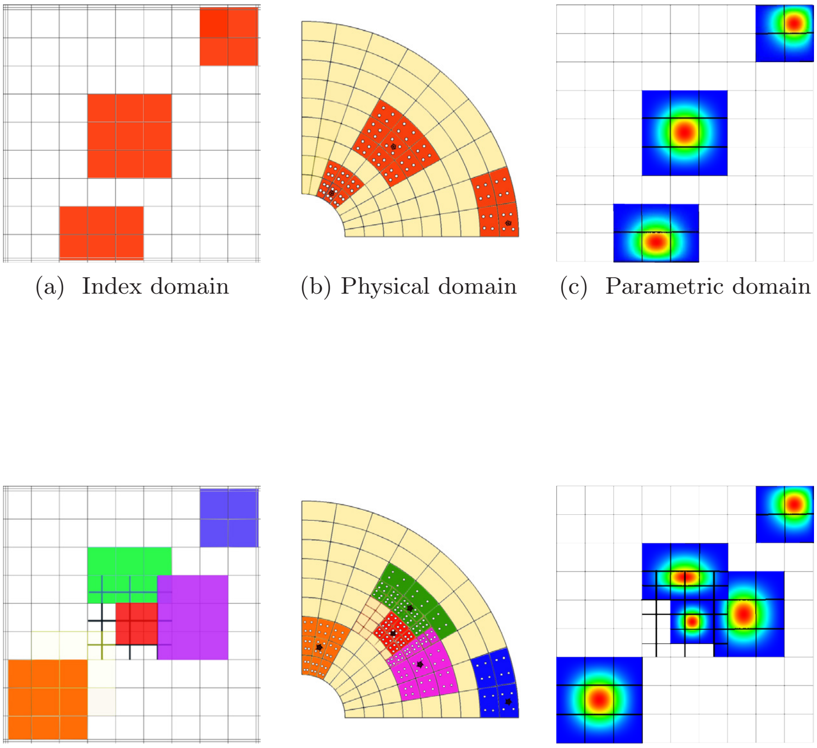
$$\frac{\partial \mathcal{F}}{\partial a_\alpha} = 0 \quad \Rightarrow \quad \mathbf{D} a_\alpha = \mathbf{G} \quad \Rightarrow \quad a_\alpha = \mathbf{D}^{-1}\mathbf{G}, \tag{49}$$

where

$$\mathbf{D} = \sum_{i=1}^{n_{sp}^{elp}} \mathbf{P}_i^T(\mathbf{x}_i)\mathbf{P}_i(\mathbf{x}_i) \qquad \text{and} \qquad \mathbf{G} = \sum_{i=1}^{n_{sp}^{elp}} \mathbf{P}_i^T(\mathbf{x}_i)\sigma_{\alpha,i}^h$$

Patch configurations

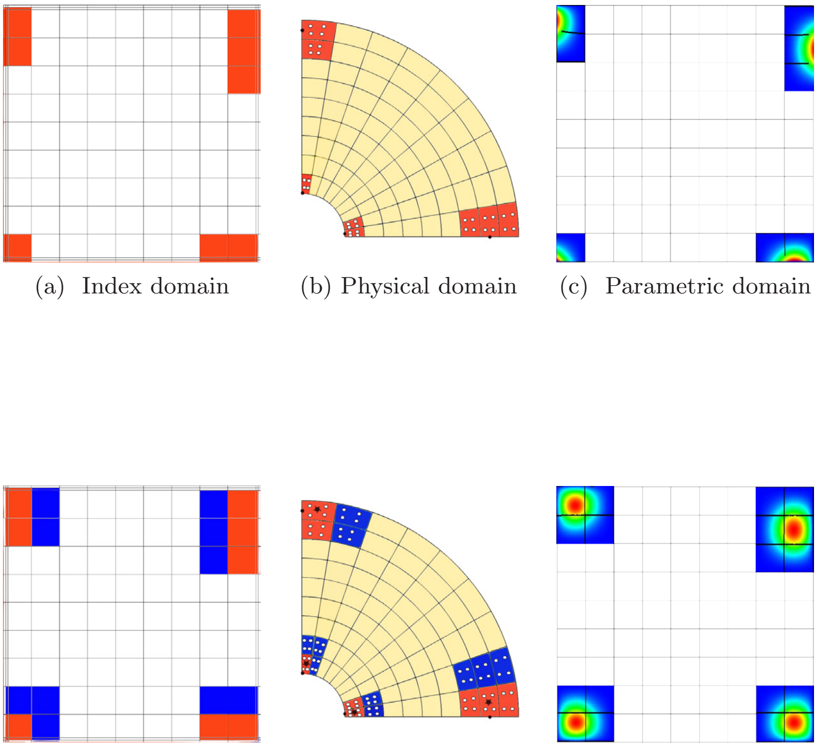
The patch configuration in isogeometric analysis is motivated from its definition in classical FEM. In FEM, the patch is a collection of elements surrounding at nodal points [77]. In IGA, we consider a patch with respect to each basis functions and it is defined by the support of that basis function. The general element patches formation with the support of quadratic B-splines/NURBS are shown in Figure 6. Similar to FEM, here we also have the concept of boundary element patches as shown in Figure 7 (first row), which does not contain sufficient number of elements for the local discrete least square fitting procedure. These special cases can be handle with the concept of extended the domain of element patches or by considered the inner patch to do the recovery procedure for that boundary basis functions. We consider the approach of using the inner element patch to recovery the value for the boundary basis function. The different cases along the boundary are shown in Figure 7.



(a) Index domain (b) Physical domain (c) Parametric domain

(d) Index domain (e) Physical domain (f) Parametric domain

Figure 6: **Inner element patch**: The element patch formation with respect to the support of quadratic B-Spline/NURBS basis function *first* row represents inner patch for tensor product case, *second* row represents inner patches on general LR mesh (or unstructured mesh), in index domain, physical domain and parametric domain, respectively (from left to right).



(a) Index domain (b) Physical domain (c) Parametric domain

(d) Index domain (e) Physical domain (f) Parametric domain

Figure 7: **Special cases**: The element patch formation with respect to the support of quadratic B-Spline/NURBS basis function *first* row represents general boundary patch and *second* row represents extended patch along the boundary of the domain, in index domain, physical domain and parametric domain, respectively (from left to right).

Global recovered gradient field

In SPR-procedure, the system (49) is established and solved for the unknown $\mathbf{a}_\alpha$ on a local patch of the elements as illustrated in Figures 6-7. The recovered gradient may be computed by evaluating (47) at the desired points within the patch. When the gradient recovery is used for error estimation, we are normally interested in recovered values at the element-interior points (full integration points). Since the specific element belongs to more than one patch, the patch recovery does not provide a unique gradient value at such points. In order to construct a global recovered gradient field, Blacker and Belytschko [18] proposed to conjoin the polynomial expansions, $\sigma^* = \mathbf{Pa}$ for all the patches containing the actual element using displacement basis as a weighting functions. Adopting this approach here, we propose to recovered the gradient field at any point $\mathbf{x}$ through

$$\sigma^*(\mathbf{x}) = \sum_{\forall A} \sigma^*_A R_A(\mathbf{x}) \tag{50}$$

where R_A is the B-splines/NURBS basis function and $\sigma^*_A(\mathbf{x})$ is a local recovered gradient field in the form (47) corresponding to the element patch formed with respect to the support of basis function R_A. Here the partition of unity property of the shape functions is used to provide the proper weighting functions in (50). Once we have more accurate gradient field value at element patch level from (47) then one can also use some global method like interpolation or L^2 projection to get the global representation of the recovered gradient field in the displacement space *via* finding the new control variable of the recovered field in displacement space. In the present work, we are interested in error norm evaluation and thus the procedure of conjoining described by (50) is considered as it is local and efficient.

5 Local behavior of spline based Galerkin discretization

In this section we first present a motivational study for the existence of natural superconvergence points for spline based Galerkin discretization of Poisson problem in 1D. In this context we will follow the arguments given in Chapter 1 from Wahlbin [69]. Later we present a general approach based on local Neumann projection to compute these superconvergence points in the computed FE solution based on B-splines and LR B-splines for one and two-dimensional elliptic model problem.

5.1 Motivational study for the existence of superconvergence points

We consider the one-dimensional Poisson problem with $\Omega = (0,1)$,

$$-\frac{d^2u}{dx^2} = f(x), \qquad \text{with bc's} \quad u(0) = u(1) = 0. \tag{51}$$

The weak formulation of (51) is to find: $u \in H_0^1(\Omega)$ such that

$$\mathcal{A}(u,\chi) \equiv \int_\Omega \left(\frac{du}{dx}\right)\left(\frac{d\chi}{dx}\right) dx = \int_\Omega f\chi dx, \ \ \forall\, \chi \in H_0^1(\Omega). \tag{52}$$

We choose a splines space of degree of p with smoothness $0 \le \mu \le p-1$ on the discretized mesh Δ_h defined by

$$\mathcal{S}^{p,\mu}_{\Delta_h,0} \equiv \{\chi(x) : \chi \in C^\mu(\Omega) \cap C^0(\overline{\Omega}),\ \chi|_{\Omega_e} \in \mathbb{P}_p(\Omega_e), \chi(0) = \chi(1) = 0\} \subseteq H_0^1(\Omega).$$

The FE spline based approximation u_h of u in $\mathcal{S}^{p,\mu}_{\Delta_h,0}$ is given by: find $u_h \in \mathcal{S}^{p,\mu}_{\Delta_h,0}$ such that

$$\mathcal{A}(u_h,\chi) = (f,\chi), \ \ \forall\, \chi \in \mathcal{S}^{p,\mu}_{\Delta_h,0}. \tag{53}$$

Ritz Projection: We define another approximation $\tilde{u}_h$ to u which is the Ritz projection (or elliptic projection for present case) given by: find $\tilde{u}_h \in \mathcal{S}^{p,\mu}_{\Delta_h,0}$ such that

$$\left(\frac{d}{dx}(u - \tilde{u}_h), \frac{d\chi}{dx}\right) = 0, \ \ \forall\, \chi \in \mathcal{S}^{p,\mu}_{\Delta_h,0}. \tag{54}$$

Let $\mathcal{S}^{p-1,\mu-1}_{\Delta_h} = \{\chi(x) : \chi \in C^{\mu-1}(\Omega) \cap C^0(\overline{\Omega}),\ \chi|_{\Omega_e} \in \mathbb{P}_{p-1}(\Omega_e)\}$ be the spline space of degree $p-1$ with smoothness $\mu - 1$ and $B_i \in \mathcal{S}^{p-1,\mu-1}_{\Delta_h}$ be a B-spline basis function $B_i(x) > 0$ with support in $\mathcal{J}_i = (x_i, x_{i+k_d})$, where $k_d = [\frac{p}{p-\mu}]^+$ denote the smallest integer $\ge \frac{p}{p-\mu}$.

Now define $\psi_i(x)$ such that

$$\psi_i(x) = \int_0^x B_i(y)ds - x\int_0^1 B_i(y)ds$$

which belongs to $\mathcal{S}^{p,\mu}_{\Delta_h,0}$. Then

$$\int_{\mathcal{J}_i} \frac{d}{dx}(u - \tilde{u}_h)B_i = \left(\frac{d}{dx}(u - \tilde{u}_h), (\frac{d\psi_i}{dx} + \int_0^1 B_i)\right) = \left(\frac{d}{dx}(u - \tilde{u}_h), \frac{d\psi_i}{dx}\right) = 0. \tag{55}$$

Since $B_i > 0$ on $\mathcal{J}_i$. There exists a point $\eta_i \in \mathcal{J}_i$ such that

$$\frac{d(u - \tilde{u}_h)}{dx}(\eta_i) = 0. \tag{56}$$

We conclude this result in the form of following theorem.

Theorem 3. *Let $k_d = [\frac{p}{p-\mu}]^+$, and let $\mathcal{J}_i = (x_i, x_{i+k_d})$, for any $i = 0, 1, \dots, N - k_d$. There exists a point $\eta_i \in \mathcal{J}_i$ such that $\frac{d(u-\tilde{u}_h)}{dx}(\eta_i) = 0$.*

Similar to the above derivative result, a corresponding result for displacement value is as follows:

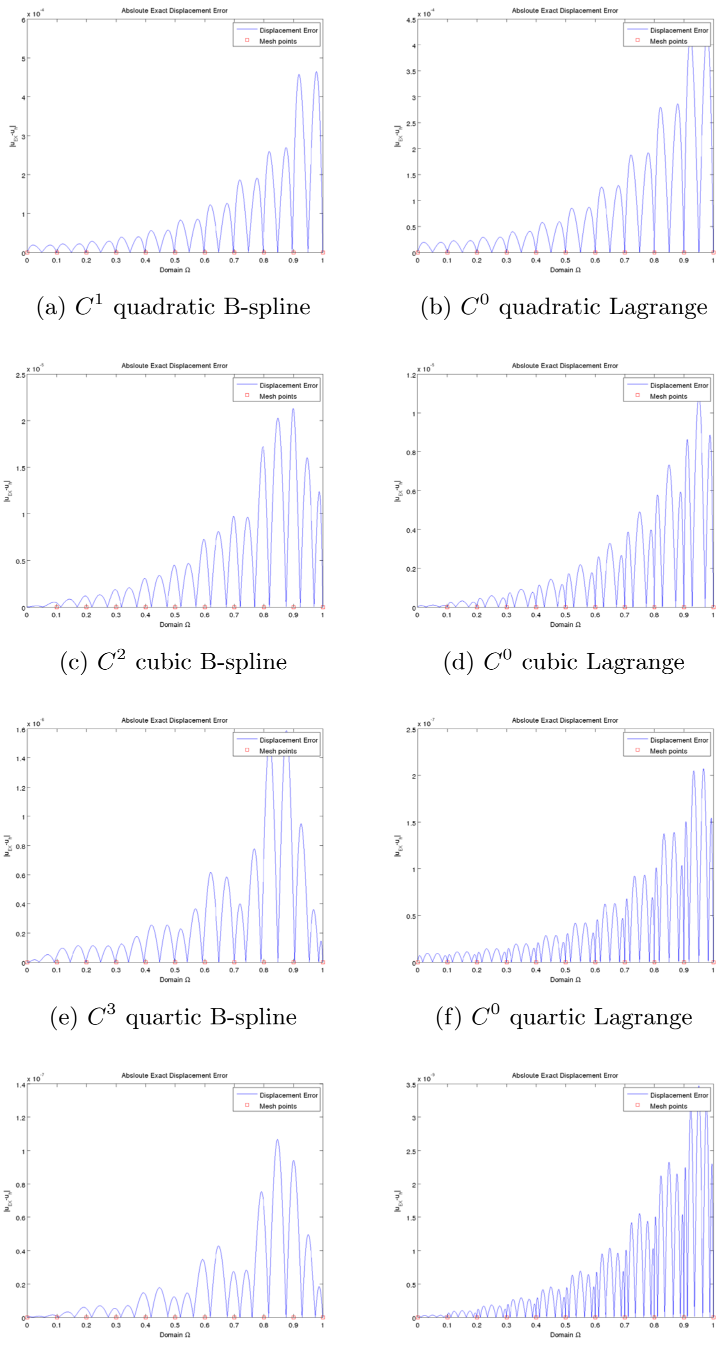
Theorem 4. *Let $k_u = [\frac{p-1}{p-2-\max(-1,\mu-2)}]^+$ for $p \geq 2$, and let $\mathcal{J}_i = (x_i, x_{i+k_u})$, for any $i = 0, 1, \dots, N - k_u$. There exists a point $\eta_i \in \mathcal{J}_i$ such that $(u - \tilde{u}_h)(\eta_i) = 0$.*

Proof. See Theorem 1.4.2 of Wahlbin [69]. □

From (52),(53) and (54), the uniqueness of Ritz projection gives that $u_h = \tilde{u}_h$. Thus the above two results hold for FE approximation u_h itself. These results does not give any information about the superconvergence points but they tell us about the existence of such points in spline based Galerkin discretization.

Now we consider a numerical example for problem (51) with given exact solution $u = x^2 - \frac{\sinh 4x}{\sinh 4}$ and the spline based FE approximation u_h obtained by (53). In Figures 8-9, we present the graph of absolute value of exact displacement error $(u - u_h)(x)$ for $x \in \Omega$ and the derivative error $\frac{d}{dx}(u - u_h)(x)$ for $x \in \Omega$. It is interesting to note that the absolute displacement error and the derivative error has zeros at several points in the domain $\Omega = (0, 1)$. For the sake of observation we also present the case of classical C^0 Lagrange elements as shown in right column of Figures 8-9 and there also we notice that the absolute displacement error and the derivative error has zeros at several points in the domain $\Omega = (0, 1)$.

In Chapter 1 of Wahlbin [69], the study based on Element Orthogonality Analysis (EOA), with certain restriction on the mesh distribution (e.g. locally symmetry), is presented to compute the location of natural superconvergence points. Table 1 summarize the superconvergence results for aysmptotic h-Galerkin formulation (as $h \to 0$ the superconvergent points for the displacement and derivative error converges to the given values in Table 1), see also page 21 of Wahlbin [69]. Here $L_p(x)$ denotes the Legendre polynomial of degree p and $L'_p(x)$ be its first derivative. For the uniform



(a) C^1 quadratic B-spline

(b) C^0 quadratic Lagrange

(c) C^2 cubic B-spline

(d) C^0 cubic Lagrange

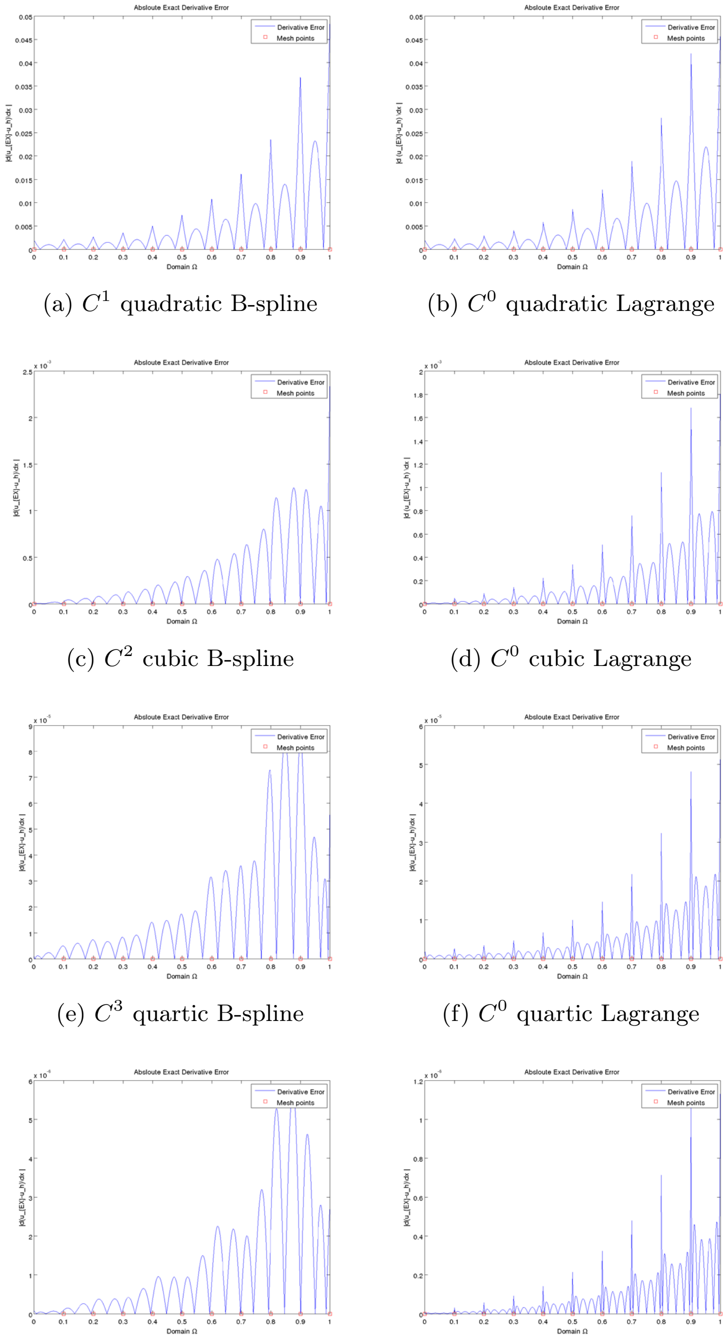
(e) C^3 quartic B-spline

(f) C^0 quartic Lagrange

(g) C^4 quintic B-spline

(h) C^0 quintic Lagrange

Figure 8: Absolute displacement error in Galerkin FE spline discretization using C^{r-1} smooth splines and C^0 Lagrange spaces for degree $p = 2, 3, 4, 5$ with uniform mesh width $h = 1/10$.



(a) C^1 quadratic B-spline

(b) C^0 quadratic Lagrange

(c) C^2 cubic B-spline

(d) C^0 cubic Lagrange

(e) C^3 quartic B-spline

(f) C^0 quartic Lagrange

(g) C^4 quintic B-spline

(h) C^0 quintic Lagrange

Figure 9: Absolute derivative error in Galerkin FE spline discretization using C^{r-1} smooth splines and C^0 Lagrange spaces for degree $p = 2, 3, 4, 5$ with uniform mesh width $h = 1/10$.

mesh distribution the location of these points can be confirmed from the numerical results shown in Figures 8-9.

$\mathcal{S}^{\mu,p}_{\Delta_h}$- Spline space	Mesh restriction	Function values	First derivative
$\mu = 0$, $p \geq 1$	Complete general meshes	(i) $O(h^{2p})$ at meshpoints (ii) $O(h^{p+2})$ at zeros of $L'_p(x)$	$O(h^{p+1})$ at zeros of $L_p(x)$
$\mu = 1$, p: Even	Meshes uniform in $C_1 h \ln 1/h$ neighborhood of point (similarly away from $\partial\Omega$)	$O(h^{p+2})$ at mesh- and midpoints and at zeros of $L'_p(x)$	$O(h^{p+1})$ at zeros of $L_p(x)$
$\mu = 1$, p: Odd	as above	$O(h^{p+2})$ at $p-1$ zeros of $Q(x) = L_{p-1}(x) - \frac{L'_{p-1}(x)}{L'_{p+1}(x)} L_{p+1}(x)$	$O(h^{p+1})$ at mesh- and midpoints, also at $p-3$ zeros of $Q'(x)$
$\mu = 2$, $p = 3$	as above	$O(h^{p+2})$ at two points, zeros of $Q(x) = L_{p-1}(x) - \frac{L'_{p-1}(x)}{L'_{p+1}(x)} L_{p+1}(x)$	$O(h^{p+1})$ mesh and midpoints
$\mu \geq 1$, p : Odd	General meshes, symmetry about the point in $C_1 h \ln 1/h$ neighborhood of point (similarly away from $\partial\Omega$)		$O(h^{p+1})$ at mesh points
$\mu \geq 1$, p : Even	as above	$O(h^{p+2})$ at mesh points	

Table 1: Summary of the superconvergence results for the aysmptotic h-Galerkin formulation, see also page 21 of Wahlbin [69]

The results summarized in the Table 1 is valid for uniform mesh distribution (and away from the boundary for certain cases) and also for the physical spline space results after linear transformation mappings. Using the tensor product argument the results also follow for 2D spline spaces with the same restriction of uniform mesh distribution (and away from the boundary) and the results can be confirmed for physical spline spaces results after bilinear transformation mappings.

Now we will present a general approach for analyzing the local behavior of spline based Galerkin discretization. This approach is motivated from setting of *computer based proof of existence of superconvergence points* of Babuška et al. [12], [11], [8], and can be used to analyze the superconvergence in spline based finite element methods in any dimension. To be consistent with the earlier work on superconvergence of Babuška and his co-workers we consider the similar notations as described in [8]. We first explain the main idea of superconvergence in one dimensional setting for the function space satisfies some assumptions are described below. We also present some numerical results to illustrate the role of this methodology to compute these points in one and two dimensional cases.

5.2 Spline spaces in 1D

To make this approach more understandable we first present this in simple one dimensional spline space setting. Below, we will give the assumptions on which the analysis is based. Denote an interval (or subdomain) of size H centered at the point $\bar{x}$ by

$$K(\bar{x}, H) := \left(\bar{x} - \frac{1}{2}H, \bar{x} + \frac{1}{2}H\right). \tag{57}$$

We will first consider the case of interior mesh elements and assume that:

Assumption 1. Define $K(\bar{x}, H_0)$ and $K(\bar{x}, H_1)$ be two mesh intervals with $H_1 < H_0 \leq H$ coincide exactly with a patch of elements, namely

$$K(\bar{x}, H_i) := \cup_{j=n_i}^{m_i} \Delta_{h,j}, \quad \forall \;\; i = 0, 1, \tag{58}$$

where n_i and m_i denotes the first and the last element from the mesh discretization Δ_h which belong to $K(\bar{x}, H_i)$.

Assumption 2. Let the exact solution u satisfy

$$\left\| \frac{d^{p+2}u}{dx^{p+2}} \right\|_{L^\infty(K(\bar{x},H))} \leq \mathcal{C}_1 < \infty \tag{59}$$

and

$$0 < \mathcal{C}_2 \leq \left| \frac{d^{p+1}u(\bar{x})}{dx^{p+1}} \right| \tag{60}$$

Assumption 3. (Pollution under control) The meshes Δ_h are such that the error $E = u - u_h$ satisfies

$$\|E\|_{L^2(K(\bar{x},H))} \leq \mathcal{C}h^\beta \sqrt{H} \qquad \text{or} \qquad \|E\|_{L^\infty(K(\bar{x},H))} \leq \mathcal{C}h^\beta \tag{61}$$

with

$$\beta \geq (p+1) - \ell, \;\; 0 < \ell < 1 \tag{62}$$

and $\mathcal{C}$ depending only on $\mathcal{C}_1$.

Assumption 3 is a more general characterization of the case of negligible pollution which has meaning for general meshes in higher dimensions, as well as in simplest one dimensional setting.

Now we are going to prove series of some lemma which lead us to the final result of this section. For this, let $U_{EX}^{\bar{x},(p+1)}$ be the $(p+1)^{th}$ degree Taylor series expansion of u centered at $\bar{x}$, defined by,

$$U_{EX}^{\bar{x},(p+1)} := \sum_{k=0}^{p+1} \frac{1}{k!} \frac{d^k}{dx^k}(u)(\bar{x})(x - \bar{x})^k. \tag{63}$$

Lemma 2. *Let u satisfies the Assumption 2, and let* $U_{EX}^{\bar{x},(p+1)}$ *be the* $(p+1)^{th}$ *degree Taylor series expansion of u centered at* $\bar{x}$*, defined by (63). Then we have*

$$\left\| \frac{d^r}{dx^r}(u - U_{EX}^{\bar{x},(p+1)}) \right\|_{L^\infty(K(\bar{x},H_i))} \leq \mathcal{C} H_i^{p+2-r} \tag{64}$$

for $r = 0, 1, 2$ *with the constant depending on* $\mathcal{C}_1$ *and* p*.*

Proof. The proof of this lemma can be easily obtained after using integral form of reminder of Taylor expansion with the Assumption 2. See also proof of Lemma 4.7.2 from Babuška and Strouboulis [8].

□

Neumann Projection in 1D: Define $\mathcal{S}_{\Delta_h}^p(K(\bar{x}, H))$ the restriction of the spline space $\mathcal{S}_{\Delta_h}^p$ in the patch of elements which belong to $K(\bar{x}, H)$ as

$$\mathcal{S}_{\Delta_h}^p(K(\bar{x}, H)) := \left\{ \chi \in C^{p-1}(K(\bar{x}, H)) \cap \mathbb{P}_p \mid \exists\ w \in \mathcal{S}_{\Delta_h}^p \ : \ \chi \equiv w|_{K(\bar{x},H)} \right\}. \tag{65}$$

Let $U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}$ be the Neumann $\mathcal{A}$-projection of $U_{EX}^{\bar{x},(p+1)}$ into the spline space $\mathcal{S}_{\Delta_h}^p(K(\bar{x}, H))$ as the solution of following discrete problem: find $U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H} \in \mathcal{S}_{\Delta_h}^p(K(\bar{x}, H))$ such that

$$\mathcal{A}_{S(\bar{x},H)}(U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}, \chi) = 0 \quad \forall \ \ \chi \in \mathcal{S}_{\Delta_h}^p(K(\bar{x}, H)), \tag{66}$$

with

$$\int_{S(\bar{x},H)} (U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}) = 0, \tag{67}$$

where the bilinear form is defined by $\mathcal{A}_{K(\bar{x},H)}(u, v) = \int_{K(\bar{x},H)} \left(\frac{du}{dx}\right)\left(\frac{dv}{dx}\right)$. Note that $U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}$ exists, and is uniquely determined from (66)-(67). This local Neumann projection is very important in analyse of error distribution such as contribution of local and global error and also in obtaining the superconvergence points. It also has a general significance as it can be simply extended to higher dimension cases.

Let us denote the last term of Taylor expansion of (63) by $Q_{EX}^{\bar{x},p+1}$, given as

$$Q_{EX}^{\bar{x},p+1} = \frac{1}{(p+1)!} \frac{d^{(p+1)}}{dx^{(p+1)}}(u)(\bar{x})(x - \bar{x})^{(p+1)}. \tag{68}$$

and let $Q^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}} \in \mathcal{S}^p_{\Delta_h}(K(\bar{x},H))$ be its Neumann $\mathcal{A}$-projection defined by (66)-(67).

Lemma 3. *Under the Assumption 2, we have*

$$\left\|\frac{d}{dx}(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(K(\bar{x},H))} = \left\|\frac{d}{dx}(Q^{\bar{x},(p+1)}_{EX} - Q^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(K(\bar{x},H))} \leq \mathcal{C}h^p\sqrt{H}. \tag{69}$$

and

$$\left\|(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(K(\bar{x},H))} = \left\|(Q^{\bar{x},(p+1)}_{EX} - Q^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(K(\bar{x},H))} \leq \mathcal{C}h^{p+1}\sqrt{H}. \tag{70}$$

Proof. Note that $U^{\bar{x},(p)}_{EX} \in \mathbb{P}_p \subseteq \mathcal{S}^p_{\Delta_h}$, we get

$$U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}} \equiv Q^{\bar{x},(p+1)}_{EX} - Q^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}}. \tag{71}$$

By the construction of Neumann projection in (66)-(67), we obtain that $U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}}$ satisfies the orthogonality condition

$$\int_{K(\bar{x},H)} \left(\frac{d}{dx}(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right)\left(\frac{d\chi}{dx}\right) = 0 \quad \forall\, \chi \in \mathcal{S}^p_{\Delta_h}(K(\bar{x},H)). \tag{72}$$

It follows that $\frac{d}{dx}(U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})$ is the best approximation of $\frac{d}{dx}(U^{\bar{x},(p+1)}_{EX})$ from $\mathcal{S}^{(p-1)}_{\Delta}(S(\bar{x},H))$ in the L^2-norm, and hence

$$\begin{aligned}\left\|\frac{d}{dx}(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(K(\bar{x},H))} &\leq \mathcal{C}h^p\left\|\frac{d^{p+1}}{dx^{p+1}}(U^{\bar{x},(p+1)}_{EX})\right\|_{L^2(K(\bar{x},H))} \quad (73)\\ &\leq \mathcal{C}h^p\sqrt{H}\left\|\frac{d^{p+1}}{dx^{p+1}}(U^{\bar{x},(p+1)}_{EX})\right\|_{L^\infty(K(\bar{x},H))}.\end{aligned}$$

After using Assumption 2 we obtained the required result (69).

Now after employing the standard Aubin and Nitsche trick we have

$$\left\|(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(S(\bar{x},H))} \leq \mathcal{C}h\left\|\frac{d}{dx}(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})\right\|_{L^2(S(\bar{x},H))} \tag{74}$$

On combining the results from (69) and (74) we obtain the required result (70).

□

Next we aim to establish a relationship between the exact FE error $u - u_h$ and the error in Neumann projection of its asymptotic expansion, i.e., $U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H}$ on some interior elements patch $K(\bar{x}, \gamma H)$, for $0 < \gamma < 1$.

Assume that we have a basic FE approximation $u_h \in \mathcal{S}^p_{\Delta_h}(K(\bar{x}, H))$ to the function u which is sufficiently smooth in $K(\bar{x}, H)$, cf. Assumption 2, such that

$$\left(\frac{d}{dx}(u - u_h), \frac{d\chi}{dx}\right) = 0, \quad \forall \chi \in \mathcal{S}^{p,comp}_{\Delta}(K(\bar{x}, H)), \tag{75}$$

where $\mathcal{S}^{p,comp}_{\Delta}(K(\bar{x}, H))$ denotes the restrictions of the functions in $\mathcal{S}^p_{\Delta}$ with compact support in the interior of $K(\bar{x}, H)$Â.

Further, we can write

$$u - u_h = U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H} + \left((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\right). \tag{76}$$

On differentiating (76), we get

$$\frac{d}{dx}(u - u_h) = \frac{d}{dx}(U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H}) + \left(\frac{d}{dx}((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H}))\right). \tag{77}$$

Now on the interval $K(\bar{x}, H_1)$, where $H_1 < H_0 \leq H$, we write

$$\begin{aligned} \frac{d}{dx}(u - u_h)(x) \quad = \quad & \underbrace{\frac{d}{dx}(U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})(x)}_{(I)} \\ & + \underbrace{\frac{d}{dx}((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H}))(x)}_{(II)}, \end{aligned} \tag{78}$$

for $x \in K(\bar{x}, H_1)$.

To obtain the bound on the (II)-term of (78), from (66) and (75) we have

$$\left(\frac{d}{dx}((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})), \frac{d\chi}{dx}\right) = 0, \quad \forall \chi \in \mathcal{S}^{p,comp}_{\Delta}(K(\bar{x}, H)). \tag{79}$$

For the problem (79), we apply the interior error estimate results from Theorem 1.2 of Schatz and Wahlbin [55], here we consider this results as a proposition by assuming that all the assumptions of Theorem 1.2 of [55] will be satisfied and the results became true for the spline element case, to obtain

$$\begin{aligned}
&\|(u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\|_{W^1_\infty(K(\bar{x},H_1))} \leq \\
&\qquad C \sum_{\chi \in K(\bar{x},H)} (\|(u - U_{EX}^{\bar{x},(p+1)}) - \chi\|_{W^1_\infty(K(\bar{x},H))} \\
&\qquad + H^{-1}\|(u - U_{EX}^{\bar{x},(p+1)}) - \chi\|_{L_\infty(S(\bar{x},H))}) \\
&\qquad + CH^{-3/2}\|(u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\|_{L^2(K(\bar{x},H))}
\end{aligned} \tag{80}$$

Using for χ the spline quasi-interpolant of $u - U_{EX}^{\bar{x},(p+1)}$ into $\mathcal{S}^p_{\Delta_h}(K(\bar{x},H))$ (Theorem 6.18, [56]), and Lemma 2, we get

$$\begin{aligned}
\|(u - U_{EX}^{\bar{x},(p+1)}) - \chi\|_{W^1_\infty(K(\bar{x},H))} &\leq Ch^p\|u - U_{EX}^{\bar{x},(p+1)}\|_{W^{p+1}_\infty(\widetilde{K}(\bar{x},H))} \\
&\leq Ch^pH.
\end{aligned} \tag{81}$$

Similarly, we get

$$H^{-1}\|(u - U_{EX}^{\bar{x},(p+1)}) - \chi\|_{L_\infty(K(\bar{x},H))} \leq Ch^{p+1} \leq Ch^pH. \tag{82}$$

Using (81) and (82) in (80), we get

$$\begin{aligned}
&\|(u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\|_{W^1_\infty(K(\bar{x},H_1))} \leq \\
&\qquad Ch^pH + CH^{-3/2}\|(u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\|_{L^2(K(\bar{x},H))}.
\end{aligned} \tag{83}$$

After using the Assumption 3 with the results of Lemma 3, we obtain

$$\begin{aligned}
&H^{-3/2}\|(u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\|_{L^2(K(\bar{x},H))} \leq \\
&\qquad H^{-3/2}\left(\|(u - u_h)\|_{L^2(K(\bar{x},H))} + \|U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H}\|_{L^2(K(\bar{x},H))}\right) \\
&\qquad \leq CH^{-1}h^{p+1-\ell}.
\end{aligned} \tag{84}$$

Now, It follows from (78), (80), (83) and (84) that

$$\frac{d}{dx}(u - u_h)(x) = \frac{d}{dx}(U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})(x) + R(x), \quad \text{for } x \in K(\bar{x},H_1). \tag{85}$$

where $R(x) = \frac{d}{dx}\left((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\right)(x)$ and satisfies

$$\left\|\frac{d}{dx}\left((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}^p_{\Delta_h}}^{\bar{x},H})\right)\right\|_{L^\infty(K(\bar{x},H_1))} \leq C(h^pH + h^{p+1-\ell}H^{-1}), \tag{86}$$

with $H = Ch^{\delta}$ we get

$$\begin{aligned}\left\|\frac{d}{dx}\left((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H})\right)\right\|_{L^\infty(K(\bar{x},H_1))} &\leq C(h^{p+\delta} + h^{p+1-\ell-\delta}) \\ &\leq Ch^{p+\min\{\delta,1-\ell-\delta\}}(87)\end{aligned}$$

Letting $\vartheta = \min\{\delta, 1-\ell-\delta\}$, where $\vartheta > 0$ provided $\ell + \delta < 1$, this gives

$$\left\|\frac{d}{dx}\left((u - U_{EX}^{\bar{x},(p+1)}) - (u_h - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H})\right)\right\|_{L^\infty(K(\bar{x},\gamma H))} \leq Ch^{p+\vartheta}, \tag{88}$$

i.e., the term $R(x)$ in (85) is thus "superconvergent" for the exact first derivative error.

Remark 5.1. *Note that* $U_{EX}^{\bar{x},(p)} \in \mathbb{P}_p \subseteq \mathcal{S}_{\Delta_h}^p$ *gives*

$$U_{EX}^{\bar{x},(p+1)} - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H} \equiv Q_{EX}^{\bar{x},(p+1)} - Q_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}. \tag{89}$$

Thus to get the derivative superconvergence points from the results (85), we need to find the zeros of

$$\frac{d}{dx}(Q_{EX}^{\bar{x},(p+1)} - Q_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H})(\eta) = 0, \quad \forall\, \eta \in K(\bar{x}, H_1), \text{ where } H_1 < H_0 \leq H. \tag{90}$$

Further, the definition of $Q_{EX}^{\bar{x},(p+1)}$ *in (68) reduce the problem (90) to find the zero for a single monomial* $W(x) = (x - \bar{x})^{(p+1)}$ *for the spline approximation space* $\mathcal{S}_{\Delta_h}^p$ *in 1D.*

Now we present two cases to show how to compute the derivative superconvergence points using the results of this section. We consider the one-dimensional version of the model Poisson problem with $\Omega = (0,1)$, where the exact solution is $u(x) = \sin(\pi x/2)$ with Dirichlet boundary condition at both the end.

Example with uniform mesh distribution

We denote u_h the FE spline based approximation of u in $\mathcal{S}_{\Delta_h}^p$, i.e. B-splines of degree p on uniform mesh with $h = 1/8$. In Figure 10a, we present the graph of $\frac{d}{dx}(u - u_h)(x)$ for $x \in \bar{\Omega}$. It is interesting to note that $\frac{d}{dx}(u - u_h)(x)$ is zero at several points in the domain Ω. It is clear from the results (85) with Remark 5.1 that, the superconvergence points for $\frac{d}{dx}(u - u_h)$ in the interior domain $K(\bar{x}, H_1) = (x_j, x_{j+1}) \subset\subset \Omega$ are the roots of $\frac{d}{dx}(W - W_h)$ in $K(\bar{x}, H_1)$, where W_h is the Neummann projection of monomial $W = (x - \bar{x})^3$, with $\bar{x} = (x_j + x_{j+1})/2$ on B-spline subspace $\mathcal{S}^p_{\Delta_h}(K(\bar{x}, H))$ of $\mathcal{S}^p_{\Delta_h}$ as defined by (66)-(67). In Figure 10b, we present the graph of $|\frac{d}{dx}(u - u_h)(x)|$ and $|\frac{d}{dx}(W - W_h)(x)|$ for $x \in (3/8, 4/8)$ using quadratic B-spline space on uniform mesh $h = 1/8$. We also show the derivative superconvergence points x_i^*s (red circle) where the derivative error $|\frac{d}{dx}(u - u_h)(x_i^*)|$, $i = 1, 2$ is much smaller than the $\max_{\forall x} |\frac{d}{dx}(u - u_h)|$ for $x \in (3/8, 4/8)$ (one interval view) in Figure 10c. From this case the computation shows that the two Gauss Legendre points will be the true derivative superconvergence points on that element.

Example with non-uniform mesh distribution

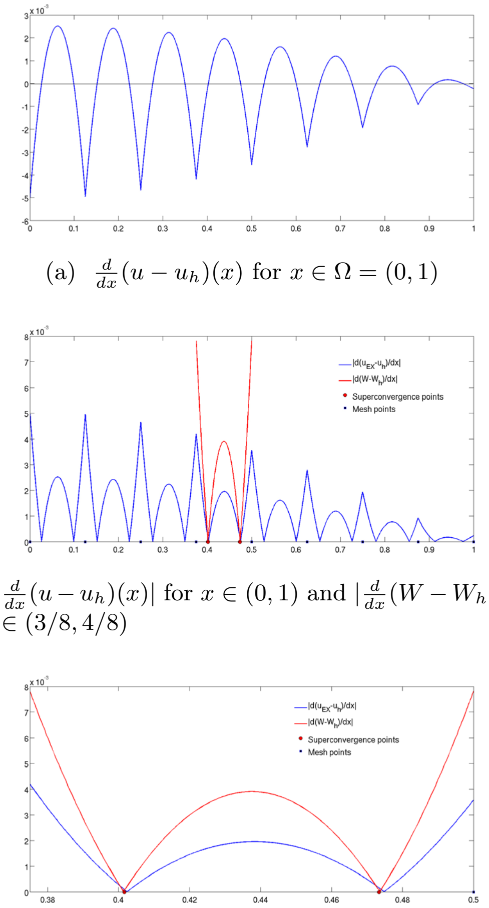
Now to distinguish with the earlier case we consider a non-uniform mesh Δ_h and compute the superconvergence points on a larger interior domain than a single interval. We consider to compute the derivative superconvergence points for $\frac{d}{dx}(u - u_h)$ on a larger interior domain $K(\bar{x}, H_1) = (x_{j-2}, x_{j+1}) \subset\subset \Omega$ via finding the zeros of $\frac{d}{dx}(W - W_h)$ in (x_{j-2}, x_{j+1}), where W_h is the Neummann projection of monomial $W = (x - \bar{x})^3$, with $\bar{x} = (x_{j-2} + x_{j+1})/2$ on spline subspace $\mathcal{S}^p_{\Delta_h}(K(\bar{x}, H))$ of $\mathcal{S}^p_{\Delta_h}$ as defined by (66)-(67). Here the domain Ω is discretized with non-uniform mesh with width $h_1 = 1/16$ and $h_2 = 1/8$. In Figure 11, we present the graph of $\frac{d}{dx}(u - u_h)(x)$ for $x \in \Omega$ and $|\frac{d}{dx}(W - W_h)(x)|$ for $x \in (6/16, 5/8)$ using quadratic B-spline space on given non-uniform mesh. We also show the derivative superconvergence points x_i^*s (red circle) where the exact derivative error $|\frac{d}{dx}(u - u_h)(x_i^*)|$, $i = 1, 2$ is much smaller than the $\max_{\forall x} |\frac{d}{dx}(u - u_h)|$ for $x \in (7/16, 1/2)$ (one interval view) in Figure 11c. This case shows that the local mesh topology will play a role in exact location of these derivative superconvergent points for spline spaces in 1D.

Extension up to the boundary

The results of this section was based on the assumption that $K(\bar{x}, H)$ is an interior patch of elements; we will now generalize them for patches $K(\bar{x}, H)$ which extend up to the boundary, see also [9, 8]. We consider the case of left boundary of the domain and assume that all the Assumptions 1-3 hold for $\bar{x} = x_L$ the left boundary point of the domain Ω. We then have

$$\frac{d}{dx}(u - u_h)(x) = \frac{d}{dx}(U^{\bar{x},(p+1)}_{EX} - U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}})(x) + \phi^{x_L,h} + Ch^{p+\vartheta}, \tag{91}$$

where $\phi^{x_L,h} \in \mathcal{S}^p_{\Delta_h}(K(x_L, H))$ such that $u_h + \phi^{x_L,h}$ satisfies the boundary

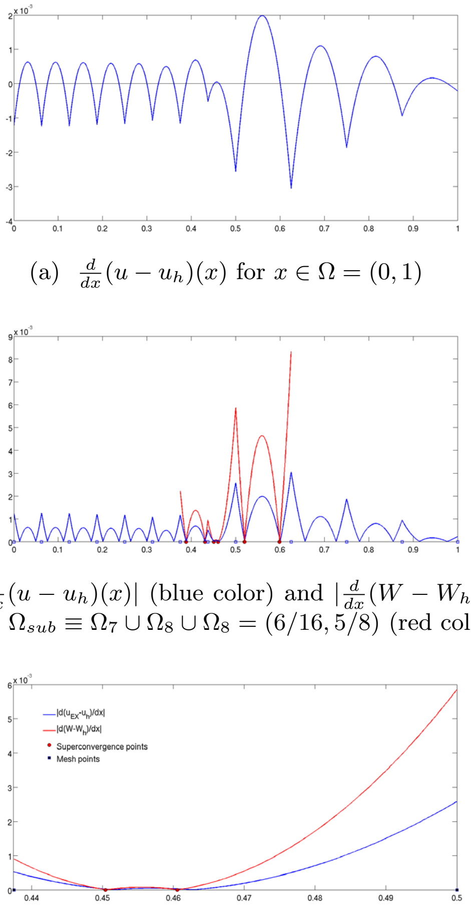


(a) $\frac{d}{dx}(u-u_h)(x)$ for $x \in \Omega = (0,1)$

(b) $|\frac{d}{dx}(u-u_h)(x)|$ for $x \in (0,1)$ and $|\frac{d}{dx}(W-W_h)(x)|$ on $x \in (3/8, 4/8)$

(c) On $\Omega_4 = (3/8, 4/8)$

Figure 10: (a) Graph of $\frac{d}{dx}(u-u_h)(x)$ for $x \in (0,1)$ using quadratic B-spline space on uniform mesh $h = 1/8$ (b) Graph of $|\frac{d}{dx}(u-u_h)(x)|$ for $x \in (0,1)$ and $|\frac{d}{dx}(W-W_h)(x)|$ for $x \in (3/8, 4/8)$ using quadratic B-spline space on uniform mesh $h = 1/8$ (c) the element view $(3/8, 4/8)$.



(a) $\frac{d}{dx}(u - u_h)(x)$ for $x \in \Omega = (0, 1)$

(b) $|\frac{d}{dx}(u - u_h)(x)|$ (blue color) and $|\frac{d}{dx}(W - W_h)(x)|$ on $x \in \Omega_{sub} \equiv \Omega_7 \cup \Omega_8 \cup \Omega_8 = (6/16, 5/8)$ (red color)

(c) On $\Omega_8 = (7/16, 1/2)$

Figure 11: (a) Graph of $\frac{d}{dx}(u - u_h)(x)$ for $x \in (0, 1)$ using quadratic B-spline space on non-uniform mesh (b) Graph of $|\frac{d}{dx}(u - u_h)(x)|$ for $x \in (0, 1)$ and $|\frac{d}{dx}(W - W_h)(x)|$ for $x \in (6/16, 5/8)$ using quadratic B-spline space on non-uniform mesh (c) the element view $x \in (7/16, 1/2)$.

conditions at $x = x_L$, and

$$\mathcal{A}_{K(x_L,\infty)}(\phi^{x_L,h}, v_h) = 0, \quad \forall\, v_h \in \mathcal{S}^p_{\Delta_h}(K(x_L,\infty)) \tag{92}$$

and

$$\lim_{x\to\infty} \frac{d}{dx}(\phi^{x_L,h}(x)) = 0. \tag{93}$$

It is clear that $\phi^{x_L,h}$ is the boundary layer correction and (93) as the decay condition. In some special cases: (i) homogenous Dirichlet boundary condition, $u(x_L) = 0$, $\phi^{x_L,h}$ must satisfy the Dirichlet boundary condition

$$\phi^{L,h}(x_L) = -(U^{x_L,(p+1)}_{EX} - U^{x_L,H}_{\mathcal{S}^p_{\Delta_h}})(x_L). \tag{94}$$

(ii) homogenous Neumann boundary condition, $\frac{du}{dx}(x_L) = 0$, $\phi^{x_L,h}$ must satisfy

$$\mathcal{A}_{K(x_L,\infty)}(\phi^{x_L,h}, \phi^h_0) = -\mathcal{A}_{K(x_L,\infty)}(U^{x_L,(p+1)}_{EX} - U^{x_L,H}_{\mathcal{S}^p_{\Delta_h}}, \phi^h_0) = 0, \tag{95}$$

where ϕ^h_0 is the basis function for the node x^h_0. The right hand side vanish identically because of the definition of Neumann projection.

In both the above cases, we have

$$\phi^{x_L,h} \equiv constant. \tag{96}$$

Hence there is no boundary layer correction term in case of the one-dimensional model problem.

Remark 5.2. *As we have seen above, in the considered model problem $\phi^{x_L,h}$ is constant, and the superconvergence points for the interior elements are valid up to the boundary. However, this is not the case, in general, for higher dimensions.*

5.3 Spline spaces in 2D

Now we describe the methodology for two dimensional spline spaces. We will make the following assumptions. Denote the subdomain of size H centered at the point $\bar{\mathbf{x}}$ by

$$K(\bar{\mathbf{x}}, H) := \left(\bar{x}_1 - \frac{1}{2}H, \bar{x}_1 + \frac{1}{2}H\right) \times \left(\bar{x}_2 - \frac{1}{2}H, \bar{x}_2 + \frac{1}{2}H\right). \tag{97}$$

We will first consider the case of interior mesh elements and assume that:

Assumption I. Define $K(\bar{\mathbf{x}}, H_0)$ and $K(\bar{\mathbf{x}}, H_1)$ be two subdomains with $H_1 < H_0 \le H$ coincide exactly with a patch of elements, namely

$$K(\bar{\mathbf{x}}, H_i) := \cup_{(k_i,l_i)\in I} \Delta_{h,(k_i,l_i)}, \quad \forall \ \ i = 0, 1, \tag{98}$$

where I is a set of indices which enumerates the cells which belongs in $K(\bar{\mathbf{x}}, H_i)$. Here we assume that H converges to zero with a slower rate than h, namely

$$\mathcal{C}_1 h^\delta \le H \le \mathcal{C}_2 h^\delta, \quad 0 < \delta < 1. \tag{99}$$

Assumption II. Let the exact solution u satisfy

$$\max_{0\le i,j\le p+2, i+j=p+2} \left\| \frac{\partial^{p+2} u}{\partial x_1^i \partial x_2^j} \right\|_{L^\infty(K(\bar{\mathbf{x}},H))} \le \mathcal{C} < \infty \tag{100}$$

and

$$0 < \mathcal{C}_0 \le \sum_{0\le i,j\le p+1, i+j=p+1} \left| \frac{\partial^{p+2} u}{\partial x_1^i \partial x_2^j}(\bar{\mathbf{x}}) \right|. \tag{101}$$

Assumption III. (Pollution under control) The meshes Δ_h are such that the error $E = u - u_h$ satisfies

$$\|E\|_{L^2(K(\bar{\mathbf{x}},H))} \le \mathcal{C} h^\beta \sqrt{H} \qquad \text{or} \qquad \|E\|_{L^\infty(K(\bar{\mathbf{x}},H))} \le \mathcal{C} h^\beta \tag{102}$$

with

$$\beta \ge (p+1) - \ell, \quad 0 < \ell < 1. \tag{103}$$

Let us now described the two dimensional analogue of the approach presented in 1D case. We will employ the $(p+1)^{th}$ degree Taylor series expansion of u at $\bar{\mathbf{x}}$, namely,

$$u = U_{EX}^{\bar{\mathbf{x}},p} + Q_{EX}^{\bar{\mathbf{x}},(p+1)} + R_{EX}^{\bar{\mathbf{x}},(p+1)} \tag{104}$$

where

$$Q_{EX}^{\bar{\mathbf{x}},k} := \sum_{0\le i,j\le k, i+j=k} \frac{1}{i!j!} \frac{\partial^k u}{\partial x_1^i \partial x_2^j}(\bar{\mathbf{x}})(x_1 - \bar{x}_1)^i (x_2 - \bar{x}_2)^j \tag{105}$$

and

$$U_{EX}^{\bar{\mathbf{x}},p} := \sum_{k=0}^{p} Q_{EX}^{\bar{\mathbf{x}},k} \tag{106}$$

and $R_{EX}^{\bar{\mathbf{x}},(p+1)}$ is the remainder. We will also let

$$Q_{(i,j)}^{\bar{\mathbf{x}},(p+1)} := (x_1 - \bar{x}_1)^i (x_2 - \bar{x}_2)^j, \quad \text{where } 0 \le i, j \le p+1, i+j = p+1. \tag{107}$$

Neumann Projection in 2D: Define $\mathcal{S}^p_{\Delta_h}(K(\bar{\mathbf{x}},H))$ the restriction of the spline space $\mathcal{S}^p_{\Delta_h}$ in the patch of elements which belong to $K(\bar{\mathbf{x}},H)$ as

$$\mathcal{S}^p_{\Delta}(K(\bar{x},H)) := \left\{\chi \in C^{p-1,p-1}(K(\bar{\mathbf{x}},H)) \cap \mathbb{P}_p \mid \exists\ w \in \mathcal{S}^p_{\Delta}\ :\ \chi \equiv w|_{K(\bar{\mathbf{x}},H)}\right\}. \tag{108}$$

Let $U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_\Delta}$ be the Neumann $\mathcal{A}$-projection of $U^{\bar{\mathbf{x}},(p+1)}_{EX}$ into the spline space $\mathcal{S}^p_{\Delta_h}(K(\bar{\mathbf{x}},H))$ as the solution of following discrete problem: find $U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}} \in \mathcal{S}^p_{\Delta_h}(K(\bar{\mathbf{x}},H))$ such that

$$\mathcal{A}_{S(\bar{\mathbf{x}},H)}((U^{\bar{\mathbf{x}},(p+1)}_{EX} - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}}),\chi) = 0 \quad \forall\ \ \chi \in \mathcal{S}^p_{\Delta_h}(K(\bar{\mathbf{x}},H)), \tag{109}$$

with

$$\int_{K(\bar{\mathbf{x}},H)} (U^{\bar{\mathbf{x}},(p+1)}_{EX} - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}}) = 0, \tag{110}$$

where the bilinear form is defined by $\mathcal{A}_{K(\bar{\mathbf{x}},H)}(u,v) = \int_{K(\bar{\mathbf{x}},H)} \nabla u \cdot \nabla v \ d\Omega$. Note that $U^{\bar{x},H}_{\mathcal{S}^p_{\Delta_h}}$ exists, and is uniquely determined from (109)-(110).

Now assume that we have a basic FE approximation $u_h \in \mathcal{S}^p_{\Delta_h}(K(\bar{\mathbf{x}},H))$ to the function u which is sufficiently smooth in $K(\bar{\mathbf{x}},H)$, cf. Assumption II, such that

$$(\nabla(u-u_h),\nabla\chi) = 0, \quad \forall\chi \in \mathcal{S}^{p,comp}_{\Delta}(K(\bar{\mathbf{x}},H)), \tag{111}$$

where $\mathcal{S}^{p,comp}_{\Delta}(K(\bar{\mathbf{x}},H))$ denotes the restrictions of the functions in $\mathcal{S}^p_\Delta$ with compact support in the interior of $K(\bar{\mathbf{x}},H)$Â.

Analogous to the 1D case, we can write

$$u - u_h = U^{\bar{\mathbf{x}},(p+1)}_{EX} - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}} + \left((u - U^{\bar{\mathbf{x}},(p+1)}_{EX}) - (u_h - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}})\right). \tag{112}$$

On differentiating (112), we get

$$\frac{\partial}{\partial x_i}(u-u_h) = \frac{\partial}{\partial x_i}(U^{\bar{\mathbf{x}},(p+1)}_{EX} - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}}) + \frac{\partial}{\partial x_i}\left((u - U^{\bar{\mathbf{x}},(p+1)}_{EX}) - (u_h - U^{\bar{\mathbf{x}},H}_{\mathcal{S}^p_{\Delta_h}})\right), \tag{113}$$

for $i = 1, 2$. Now using (109),(110) and (111) with the similar arguments of 1D analysis, we can obtained an analogue results in 2D which is stated as follows:

Theorem 5. *Under the Assumptions I-III, there exists an $\delta \in (0,1)$ such that $\mathcal{C}_1 h^\delta \leq H \leq \mathcal{C}_2 h^\delta$, and*

$$\max_{i=1,2} \left\| \frac{\partial}{\partial x_i}(u - u_h) - \frac{\partial}{\partial x_i}(U^{\bar{\boldsymbol{x}},(p+1)}_{EX} - U^{\bar{\boldsymbol{x}},H}_{\mathcal{S}^p_{\Delta_h}}) \right\|_{L^\infty(K(\bar{\boldsymbol{x}},\gamma H))} \leq \mathcal{C} h^{p+\mu} \tag{114}$$

where $K(\bar{\boldsymbol{x}}, \gamma H))$ *is an interior subdomain with* $0 < \gamma < 1$

Proof. The proof follows analogous steps as in 1D case. □

The above result can also be written as, for each components $i = 1, 2$;

$$\frac{\partial}{\partial x_i}(u - u_h)(\eta) = \frac{\partial}{\partial x_i}(U_{EX}^{\bar{\mathbf{x}},(p+1)} - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{\mathbf{x}},H})(\eta) + R_i(\eta), \quad \text{for } \eta \in K(\bar{\mathbf{x}}, \gamma H). \tag{115}$$

where $\|R_i(x)\|_{L^\infty(K(\bar{\mathbf{x}},\gamma H))} \leq Ch^{p+\mu}$.

In nonhomogenous case e.g. Poisson equation

Note that $U_{EX}^{\bar{\mathbf{x}},(p)} \in \mathbb{P}_p \subseteq \mathcal{S}_{\Delta_h}^p$, i.e., the FE approximation is able to reproduce exactly all polynomial of degree p, which is true in case of (bi-p) tensor product space with quadrilaterals, gives

$$U_{EX}^{\bar{\mathbf{x}},(p+1)} - U_{\mathcal{S}_{\Delta_h}^p}^{\bar{\mathbf{x}},H} \equiv Q_{EX}^{\bar{\mathbf{x}},(p+1)} - Q_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H}. \tag{116}$$

Thus to get the derivative superconvergence points from the results (115), we need to find the **common** zeros of

$$\frac{\partial}{\partial x_i}(Q_{EX}^{\bar{x},(p+1)} - Q_{\mathcal{S}_{\Delta_h}^p}^{\bar{x},H})(\eta) = 0, \quad \forall\, \eta \in K(\bar{x}, \gamma H), 0 < \gamma < 1, \text{ for } i = 1, 2. \tag{117}$$

Further, for the tensor product spline approximation space $\mathcal{S}_{\Delta_h}^p$ in 2D, the definition of $Q_{EX}^{\bar{x},(p+1)}$ in (105) reduce the problem (117) to find the zeros only for the case of two monomials

$$Q_1(\mathbf{x}) = (x_1 - \bar{x}_1)^{(p+1)} \qquad \text{and} \qquad Q_2(\mathbf{x}) = (x_2 - \bar{x}_2)^{(p+1)}.$$

For homogenous case e.g. the Laplace equation

The number of monomial to find the zeros in (115) is further reduce for the case of Laplace equation, in this case $f = 0$, where it is known a priori that u satisfies the isotropic Laplacian $\Delta u = 0$. In this case we have,

$$Q_{EX}^{\bar{\mathbf{x}},(p+1)} = \frac{1}{(p+1)!}\frac{\partial^{p+1}}{\partial x_1^{p+1}}(u)(\bar{\mathbf{x}})Q_{1,hom}^{\bar{\mathbf{x}},(p+1)} + B(u)(\bar{\mathbf{x}})Q_{2,hom}^{\bar{\mathbf{x}},(p+1)} \tag{118}$$

where

$$B(u)(\bar{\mathbf{x}}) = \begin{cases} \frac{1}{(p+1)!}\frac{\partial^{p+1}u}{\partial x_1^p \partial x_2}(\bar{\mathbf{x}}), & \text{if } p \text{ is even} \\ (-1)^{(p-1)/2}\frac{1}{(p+1)!}\frac{\partial^{p+1}u}{\partial x_2^{p+1}}(\bar{\mathbf{x}}), & \text{if } p \text{ is odd} \end{cases} \tag{119}$$

and

$$Q^{\bar{\mathbf{x}},(p+1)}_{1,hom}(x_1,x_2) = Re((z-\bar{z})^k), \qquad Q^{\bar{\mathbf{x}},(p+1)}_{2,hom}(x_1,x_2) = Im((z-\bar{z})^k)$$

are the harmonic monomials of degree $(p+1)$ centered at $\bar{\mathbf{x}}$, where $z = x_1 + ix_2$ and $\bar{z} = \bar{x}_1 + i\bar{x}_2$.

For $p = 1$, we obtain

$$\begin{aligned} Q^{\bar{\mathbf{x}},(2)}_{1,hom}(x_1,x_2) &= (x_1-\bar{x}_1)^2 - (x_2-\bar{x}_2)^2, \\ Q^{\bar{\mathbf{x}},(2)}_{2,hom}(x_1,x_2) &= (x_1-\bar{x}_1)(x_2-\bar{x}_2), \end{aligned}$$

while $p = 2$,

$$\begin{aligned} Q^{\bar{\mathbf{x}},(3)}_{1,hom}(x_1,x_2) &= (x_1-\bar{x}_1)^3 - 3(x_1-\bar{x}_1)(x_2-\bar{x}_2)^2 \\ Q^{\bar{\mathbf{x}},(3)}_{2,hom}(x_1,x_2) &= -(x_2-\bar{x}_2)^3 + 3(x_1-\bar{x}_1)^2(x_2-\bar{x}_2) \end{aligned}$$

and $p = 3$ we obtain

$$\begin{aligned} Q^{\bar{\mathbf{x}},(4)}_{1,hom}(x_1,x_2) &= (x_1-\bar{x}_1)^4 + (x_2-\bar{x}_2)^4 - 6(x_1-\bar{x}_1)^2(x_2-\bar{x}_2)^2 \\ Q^{\bar{\mathbf{x}},(4)}_{2,hom}(x_1,x_2) &= (x_1-\bar{x}_1)^3(x_2-\bar{x}_2) - (x_1-\bar{x}_1)(x_2-\bar{x}_2)^3. \end{aligned}$$

Thus for the tensor product (bi-p) spline spaces in 2D the problem of finding the zeros (117) will reduce only to these two polynomial $Q^{\bar{\mathbf{x}},(p+1)}_{i,hom}$, $i = 1,2$.

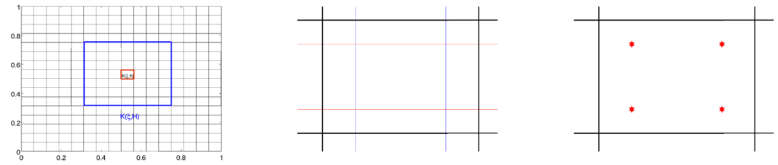
Computation of derivative superconvergence points for tensor product spline spaces in 2D

Uniform mesh partitions:

To find the derivative superconvergence points for Poisson problem in 2D, we first consider the case of tensor product spline approximation space $\mathcal{S}^p_{\Delta_h}$ with uniform mesh distribution. The computed derivatives superconvergence points are obtained by finding the common zeros of the derivatives of the difference between the monomials $\{Q_1, Q_2\}$ and its Neumann elliptic projections, where the monomials are defined as

$$Q_1(\mathbf{x}) = (x_1-\bar{x}_1)^{(p+1)} \qquad \text{and} \qquad Q_2(\mathbf{x}) = (x_2-\bar{x}_2)^{(p+1)}.$$

In Figure 12, we consider the case with respect to quadratic splines with $p = 2$. The local subdomain $K(\xi, H)$ to compute the Neumann projection for each monomials and the element of interest $K(\xi, h)$ to compute derivative superconvergence points are shown in Figure 12(a). The blue lines and



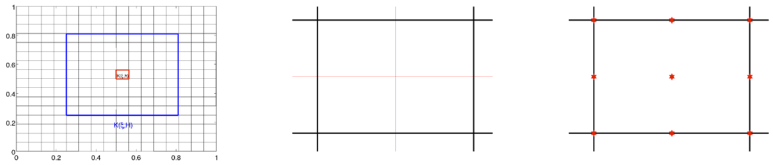
(a) Computational domain (b) Zeros of derivatives (c) Derivative Sup. pts.

Figure 12: Tensor product case with uniform mesh partition: (a) Computational domain for an element $(K(\xi, h))$ and subdomain $K(\xi, H)$ for Neumann projection with quadratic B-spline tensor product mesh with uniform spacing $h = 1/16$; (b) Zeros of the derivatives for $Q_1(x)$ and $Q_2(x)$; (c) Derivatives superconvergence points at element level: (2×2)-Gauss Legendre points.

red lines within the element of interest shown in Figure 12(b) are the Gauss-lines and they represents the location of derivative zeros with respect to $Q_1(x)$ and $Q_2(x)$ monomials, respectively. The common zeros of these lines, as (2×2)-Gauss Legendre points, are shown in Figure 12(c) which will be the derivative superconvergence points for tensor product quadratic spline spaces. Similar to the quadratic case, in Figure 13 we compute the location of computed derivative superconvergence points for tensor product cubic spline spaces, which will be the (3×3)-Gauss Lobatto points. Using the same methodology the derivative superconvergence points at the element level for the case of C^0-quadratic splines, C^0-cubic splines and C^1-cubic splines are shown in Figure 14. The C^0-quadratic splines represent the case of classical Lagrange elements and (2×2)-Gauss points will be the derivatives superconvergence points within each elements, while C^1-cubic splines share the same location of derivative superconvergence points within each elements as does C^2-cubic splines. For C^0-cubic splines we obtain the (3×3)-Gauss Legendre points as derivative superconvergence points within each elements.

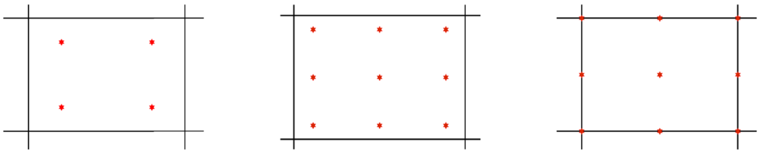
Non-uniform mesh partitions:

For the tensor product case in 2D, we can also compute the location of derivative superconvergence points results after computing the location of points from 1D analysis in each direction. In Figure 15(a) we consider a case with respect to tensor product spline approximation space $\mathcal{S}^2_{\Delta_h}$ in 2D with non-uniform mesh distribution, here the mesh interface lines are shown in dark black lines. We discuss different cases arises by enforcing the C^0 or C^1 continuity along mesh interface lines for C^0 and C^1 quadratic spline



(a) Computational domain (b) Zeros of derivatives (c) Derivative Sup. pts.

Figure 13: Tensor product case with uniform mesh partition: (a) Computational domain for an element $(K(\xi, h))$ and subdomain $K(\xi, H)$ for Neumann projection with cubic B-spline tensor product mesh with uniform spacing $h = 1/16$; (b) Zeros of the derivatives for $Q_1(x)$ and $Q_2(x)$; (c) Derivatives superconvergence points at element level: (3×3)-Gauss Lobatto points.



(a) C^0-quadratic spline (b) C^0-cubic spline (c) C^1-cubic spline

Figure 14: Derivative superconvergence points for tensor product case with uniform mesh partition: (a) C^0-quadratic spline: (2×2)-Gauss Legendre points; (b) C^0-cubic spline: (3×3)-Gauss Legendre points; (c) C^1-cubic spline: (3×3)-Gauss Lobatto points.

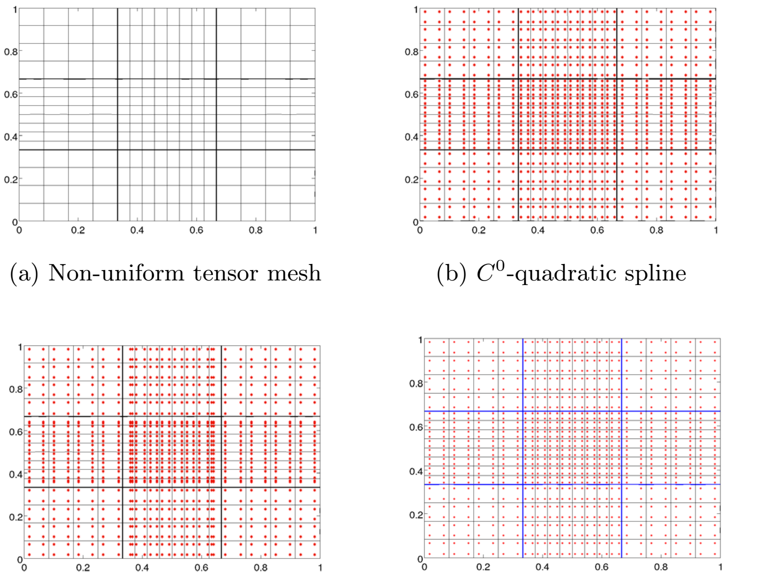
spaces. In Figure 15(b) we show the location of computed derivative superconvergence points for C^0-quadratic splines using the Neumman elliptic projection in 2D. We obtain the same results by using the Neumman elliptic projection for 1D case in each directions and then taking the tensor product of those points. The results for C^1-quadratic splines with C^1 and C^0-continuity lines along mesh interface lineas are shown in Figures 15(c) and 15(d), respectively. When there is C^1 continuity along the interface lines then there will be a shift towards the fine meshes while for the case of C^0 continuity along the interface the derivative superconvergence points will remains at (2×2)-Gauss Legendre points as the case with C^0-quadratic splines. The results for tensor product spline approximation space $\mathcal{S}^p_{\Delta_h}$ of degree three with non-uniform mesh distribution, with different cases arises by enforcing the C^0, C^1 and C^2 continuity along mesh interface lines for C^0, C^1 and C^2 cubic splines are shown in Figure 16. Due to the presence of C^0 continuity lines along the mesh interfaces and C^{-1} lines along the boundary for C^2-cubic spline case as shown in Figure 16 (d), the derivative points in immediate neighborhood of the reduced continuity interface line will shift at the derivative superconvergent lines of their reduced continuity counterparts while in other part of the domain they will be at $(3{\times}3)$-Gauss Lobatto points.

Computation of derivative superconvergence points for LR B-spline spaces in 2D

To find the derivative superconvergence points for Poisson problem on adaptive structured mesh of LR B-splines of degree two, $\mathcal{S}^2_{\Delta_h}$, we consider an example of LR mesh shown in Figure 17(a) with three domain of interests $K_i(\xi, H_i)$, for $i = 1, 2, 3$. The domain of interest $K_1(\xi, H_1)$ is considered within the fine mesh of two level of refinements, while the cases of $K_2(\xi, H_2)$ and $K_3(\xi, H_3)$ represents different interface regions affected by one side and both sides of refinements, respectively. The B-splines basis functions representation on the LR mesh of 17(a) is shown 17(b). The green circle represents the coarse basis functions and red circle represents the fine basis functions, while the blue circle represents the coarse basis function whose supports was affected via the local refinement. The computed derivatives are obtained by finding the zeros of the derivatives of the difference between the monomials $\{Q_1, Q_2\}$ and its Neumann elliptic projections, where the monomials are defined as

$$Q_1(\mathbf{x}) = (x_1 - \bar{x}_1)^{(p+1)} \qquad \text{and} \qquad Q_2(\mathbf{x}) = (x_2 - \bar{x}_2)^{(p+1)}.$$

The zeros of derivatives components with respect to $Q_1(x)$ and $Q_2(x)$ is represented by the blue and red lines in Figures 17(c),(e) and (g) for the domain of interests $K_1(\xi, H_1)$,$K_2(\xi, H_2)$ and $K_3(\xi, H_3)$, respectively. While



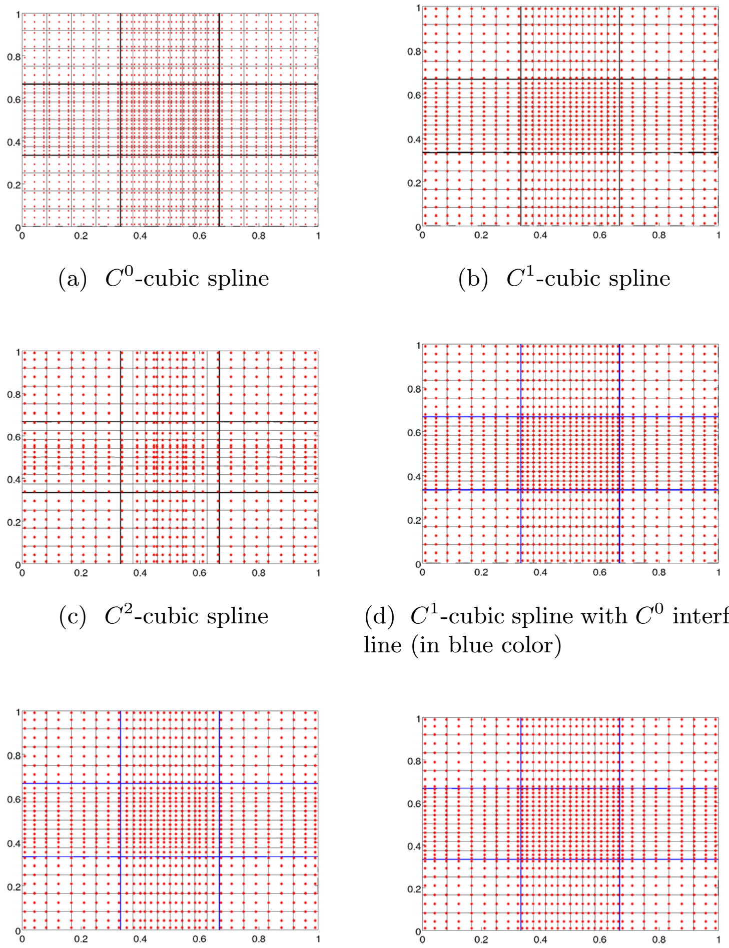
(a) Non-uniform tensor mesh

(b) C^0-quadratic spline

(c) C^1-quadratic spline

(d) C^1-quadratic spline with C^0 interface line

Figure 15: Derivative superconvergence points for quadratic tensor product case with non-uniform mesh partition: (a) non-uniform mesh tensor mesh; (b) C^0-quadratic spline; (b) C^1-quadratic spline; (d) C^1-quadratic spline with C^0 interface line (in blue color).



(a) C^0-cubic spline

(b) C^1-cubic spline

(c) C^2-cubic spline

(d) C^1-cubic spline with C^0 interface line (in blue color)

(e) C^2-cubic spline with C^1 interface line (in blue color)

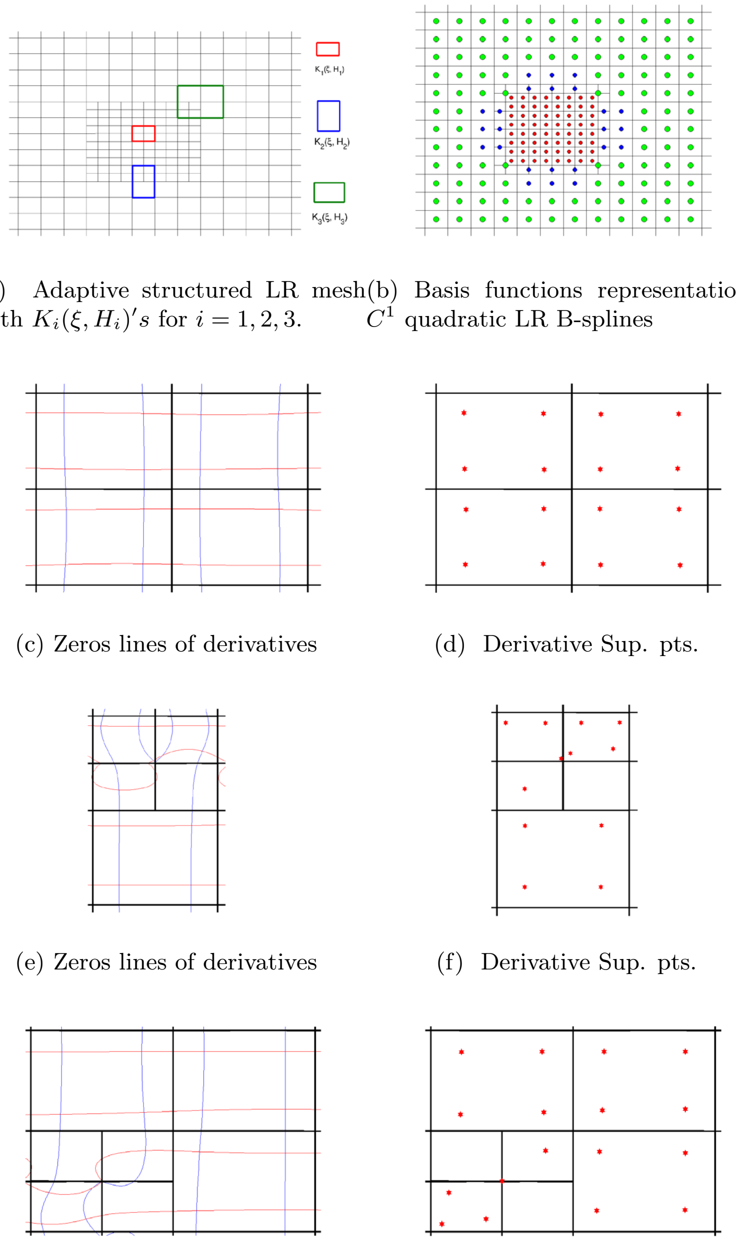
(f) C^2-cubic spline with C^0 interface line (in blue color)

Figure 16: Derivative superconvergence points for cubic tensor product case with non-uniform mesh partition: (a) C^0-cubic spline; (b) C^1-cubic spline; (c) C^1-cubic spline; (d) C^1-cubic spline with C^0 interface line (in blue color); (e) C^2-cubic spline with C^1 interface line (in blue color); (f) C^2-cubic spline with C^0 interface line (in blue color).

the location of computed derivative superconvergence points for these cases are given by the common zeros of the derivatives with respect to $Q_1(x)$ and $Q_2(x)$ are shown in Figures 17(d),(f) and (h).

The computed derivative superconvergence points for the case of LR B-spline mesh of C^0-quadratic B-splines for the domain of interests $K_i(\xi, H_i)$, for $i = 1, 2, 3$ of Figure 18(a) are shown in Figures 18.

We also found that the location of derivative superconvergence points for tensor product cases as discussed here will remain same for Poisson and Laplace equations. So all the above results also hold for the case of Laplace equation.



(a) Adaptive structured LR mesh with $K_i(\xi, H_i)'s$ for $i = 1, 2, 3$.

(b) Basis functions representation: C^1 quadratic LR B-splines

(c) Zeros lines of derivatives

(d) Derivative Sup. pts.

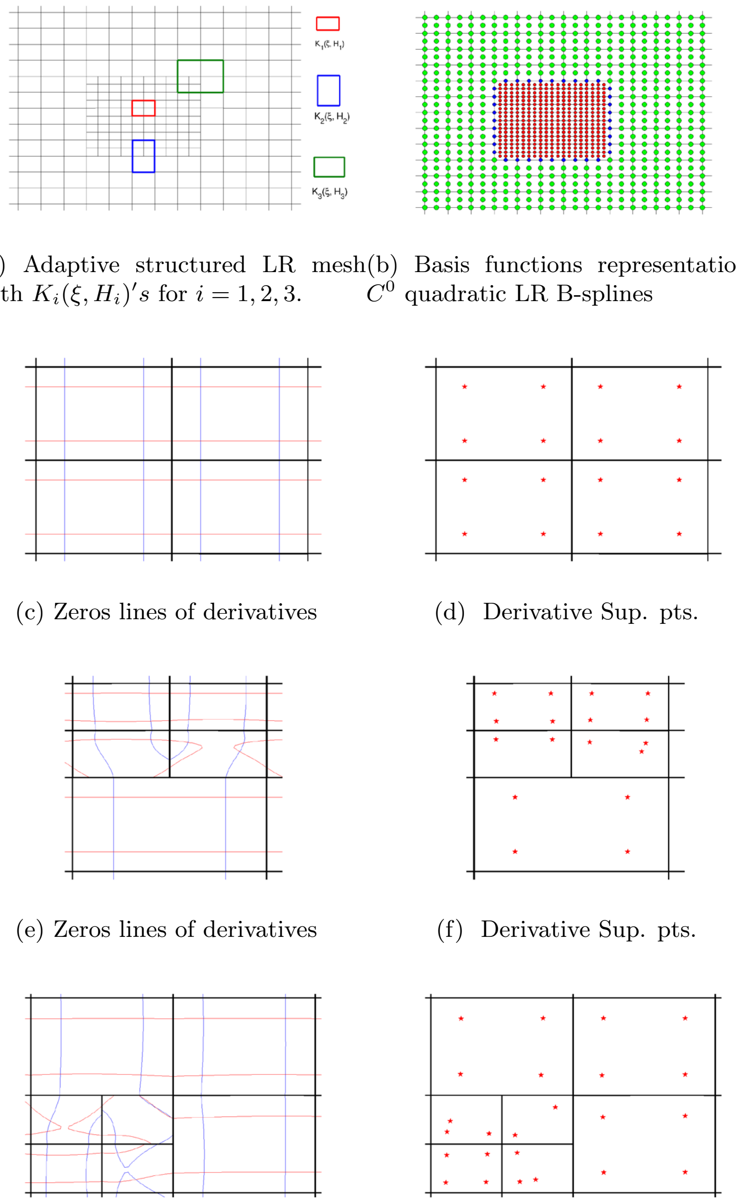
(e) Zeros lines of derivatives

(f) Derivative Sup. pts.

(g) Zeros lines of derivatives

(h) Derivative Sup. pts.

Figure 17: C^1 **quadratic LR B-splines case**: (a) Structured LR mesh with computational domain of interests $(K_i(\xi, H_i))$, i=1,2,3; (b) C^1-quadratic LR B-splines basis function representation; (c) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_1(\xi, H_1)$; (d) Derivatives superconvergence points at $K_1(\xi, H_1)$; (e) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_2(\xi, H_2)$; (f) Derivatives superconvergence points at $K_2(\xi, H_2)$; (g) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_3(\xi, H_3)$; (h) Derivatives superconvergence points at $K_3(\xi, H_3)$.



(a) Adaptive structured LR mesh with $K_i(\xi, H_i)'s$ for $i = 1, 2, 3$.

(b) Basis functions representation: C^0 quadratic LR B-splines

(c) Zeros lines of derivatives

(d) Derivative Sup. pts.

(e) Zeros lines of derivatives

(f) Derivative Sup. pts.

(g) Zeros lines of derivatives

(h) Derivative Sup. pts.

Figure 18: C^0 **quadratic LR B-splines case**: (a) Structured LR mesh with computational domain of interests ($K_i(\xi, H_i)$), i=1,2,3; (b) C^0-quadratic LR B-splines representation; (c) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_1(\xi, H_1)$; (d) Derivatives superconvergence points at $K_1(\xi, H_1)$; (e) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_2(\xi, H_2)$; (f) Derivatives superconvergence points at $K_2(\xi, H_2)$; (g) Zeros lines of the derivatives for $Q_1(x)$ and $Q_2(x)$ on $K_3(\xi, H_3)$; (h) Derivatives superconvergence points at $K_3(\xi, H_3)$.

6 Abstract recovery operator G_h

In this section we define the *abstract* recovery operator G_h which act on the FE approximation u_h to give an approximation to the gradient ∇u. In particular we will focus on the set of conditions proposed by Ainsworth and Craig in [1] which constitutes a good approximation to the gradient in order that the resulting estimator will be asymptotically exact. A gradient recovery operator on X_h is a linear operator $G_h : X_h \to [X_h]^n$ that enjoy the following properties:

(i) **Consistency condition**: Whenever $u \in \mathbb{P}_{p+1}(\Omega)$

$$G_h(\mathcal{I}_h u) \equiv \nabla u \tag{120}$$

where $\mathcal{I}_h$ is the interpolation or projection operator.

(ii) **Localizing condition**: In order to ensure that the scheme is truly local and the sub-domain $\hat{\Omega}_i^h := \cup_{j \in adj(i)} \Omega_j^h$ (or element patch) are small, we shall make a restriction upon the cardinality of the indexing sets. Then the localizing condition becomes: For any $x^* \in \Omega_i^h$, $G_h[v](x^*)$ depends only upon the values of ∇v on the domain $\hat{\Omega}_i^h$. Further, $i \in \text{adj}(i)$ and there should exist a constant M, which is independent of h such that,

$$\text{card}[\text{adj}(i)] \leq M, \qquad \forall i. \tag{121}$$

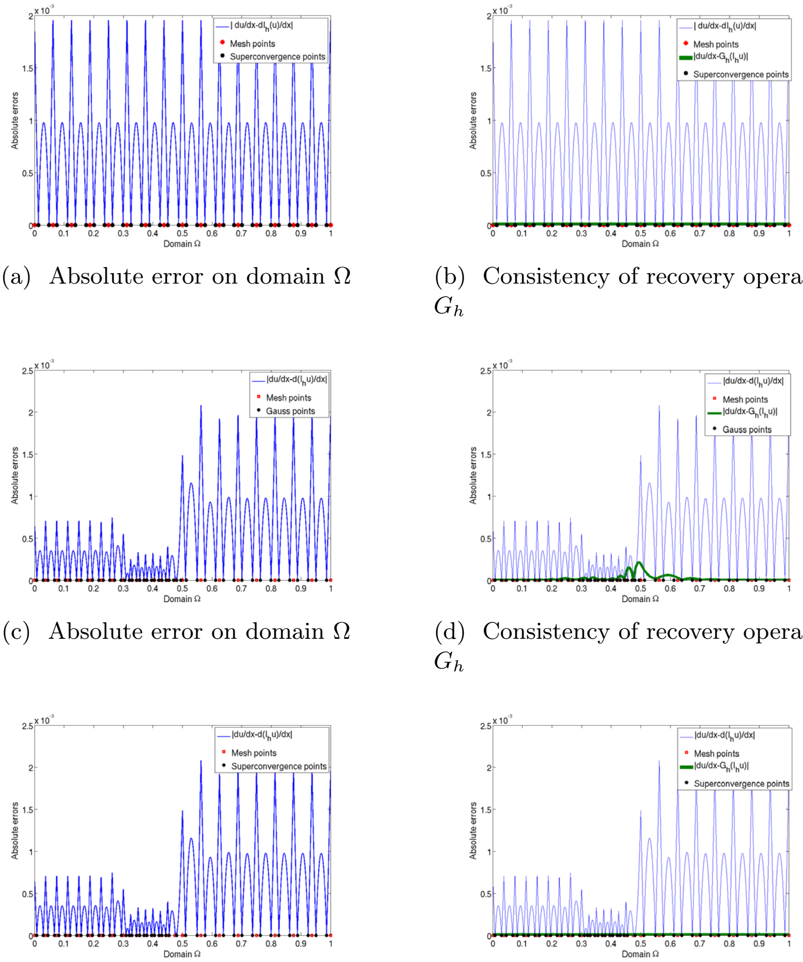
(iii) **Boundedness and linearity condition**: There exists a constant C, which is independent of h, such that

$$|G_h[v]|_{0,\infty,\Omega_i^h} \leq C|v|_{1,\infty,\hat{\Omega}_i^h}. \tag{122}$$

Now we will show that the SPR recovery operator presented in Section 4.2 will satisfy the above stated set of conditions. It can be seen easily that the present SPR recovery operator will satisfy the localization condition (ii) and boundedness and linearity conditions (iii) by following the standard techniqe as shown in an example from [1]. Here we mainly focus to satisfy the consistency condition (i) which will be in general expect to obtain an approximation consistent with the true gradient under favorable circumstances.

One dimensional case

In Figures 19(a)-19(b), we show that the consistency condition (i) is satisfied by the SPR recovery operator, considered as G_h here, for quadratic spline approximating space on uniform mesh using two Gauss Legendre points as the true superconvergence points. Here we consider the exact solution $u(x) = x^3 \in \mathbb{P}_3(\Omega)$ and $\mathcal{I}_h$ as the Neumann projection operator defined by (66)-(67). In general one can choose $\mathcal{I}_h$ as a interpolation operator instead of local Neumann projection as considered here, but the local Neumann projection has general significance and can be extend in multi-dimensional cases. In Figure 19(a) we show the absolute error in the derivative of u and the location of superconvergence points with dark black circle which coincide with the two Gauss Legendre points on each mesh intervals. While in Figure 19(b) we showed that the recovered gradient obtained using the SPR recovery operator of Section 4.2 results in exact derivative du/dx and the absolute error becomes zeros which is shown with the green curve. In Figure 19(d) we show that the consistency condition is not satisfied when we use the two Gauss Legendre points instead of true superconvergence points on a non-uniform mesh with quadratic spline space approximation. While in Figure 19(f) the consistency conditions is again satisfied when we use the true superconvergence points instead of two Gauss Legendre points in the SPR recovery operator G_h on non-uniform mesh. It can also be noted from Figure 19(d) that only near the interface mesh interval of the non-uniform mesh the absolute consistency error will not be zero, although it has smaller value than the error in Neumann projection, but at other part of the domain the error in recovered derivative become zeros and the consistency condition is satisfied.



(a) Absolute error on domain Ω

(b) Consistency of recovery operator G_h

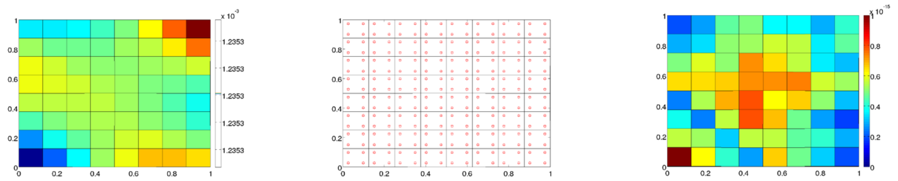
(c) Absolute error on domain Ω

(d) Consistency of recovery operator G_h

(e) Absolute error on domain Ω

(f) Consistency of recovery operator G_h

Figure 19: One dimensional case: The consistency of SPR recovery operator considered as G_h here; *first* row satisfies the consistency condition on uniform mesh with two Gauss Legendre points as sampling point in the SPR procedure, *second* row does not satisfies the consistency condition on a non-uniform mesh with two Gauss Legendre points as sampling point in SPR procedure, *third* row satisfies the consistency condition on general non-uniform mesh with computed true derivative superconvergence points as sampling point in SPR procedure, where we have considered $u = x^3 \in \mathbb{P}_3(\Omega)$ and $\mathcal{I}_h$ a local Neumann projection of u in quadratic B-spline space.



(a) $\|\nabla u - \nabla(\mathcal{I}_h u)\|_{L^2(\Omega)} := 9.88e-03$

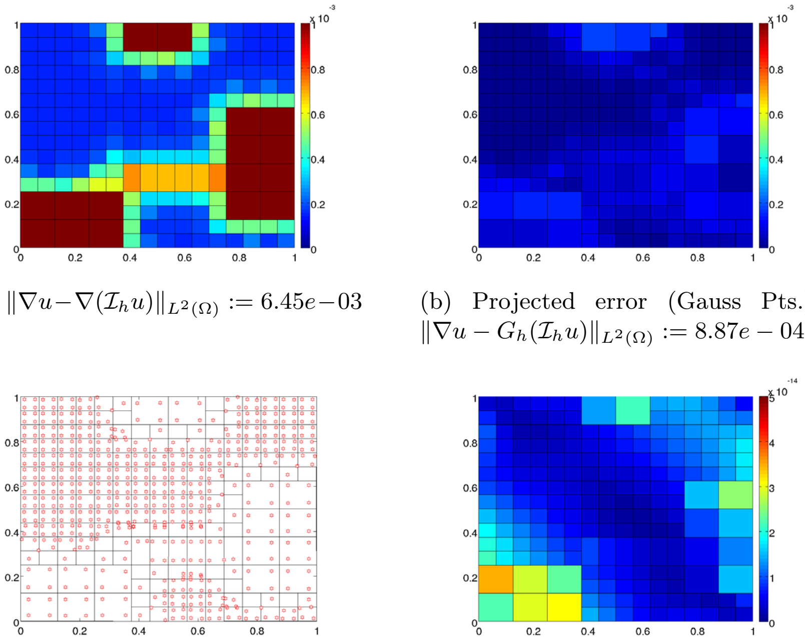
(b) LR mesh with superconvergence points

(c) $\|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)} := 4.17e-15$

Figure 20: Two dimensional case: The consistency of SPR recovery operator considered as G_h here; The SPR recovery operator G_h satisfies the consistency condition on uniform mesh with (2×2)-Gauss Legendre points as sampling point in SPR procedure, where we have considered $u = (2x^3 - 3x^2) + (2y^3 - 3y^2) \in \mathbb{P}_3(\Omega)$ and $\mathcal{I}_h$ a local Neumann projection of u in quadratic B-spline space.

Two dimensional case

In Figure 20, we show that the consistency condition (i) is satisfied by the SPR recovery operator, considered as G_h here, for quadratic spline approximating space on uniform mesh using (2×2)-Gauss Legendre points as the true superconvergence points. Here we consider the exact solution $u = (2x^3 - 3x^2) + (2y^3 - 3y^2) \in \mathbb{P}_3(\Omega)$ and $\mathcal{I}_h$ as the Neumann projection operator defined by (109)-(110). Figure 20(a) shows the L^2-norm error in the gradient of u and the location of superconvergence points which coincide with the (2×2)-Gauss Legendre points on each mesh elements is shown in Figure 20(b). While in Figure 20(c) we show that the recovered gradient obtained using the SPR recovery operator of Section 4.2 results in exact gradient and the absolute error in the projected gradient becomes numerically zero as shown in Figure 20(c). In Figure 21 we consider the case with a general non-uniform mesh of quadratic LR B-spline. It can be seen from Figure 21(b) that the consistency condition is not satisfied when we use the (2×2)-Gauss Legendre points on a general non-uniform mesh with quadratic LR B-spline space approximation in our SPR procedure. While in Figure 21(d) the consistency conditions is satisfied when we use the true superconvergence points instead of (2×2)-Gauss Legendre points in our SPR recovery operator G_h on a general non-uniform mesh. It can also be noticed from Figure 21(b) that although the consistency condition is not satisfied on a general non-uniform mesh using (2×2)-Gauss Legendre points still the L^2-norm of the projected error $\|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)}$ has a smaller value than the error in Neumann projection $\|\nabla u - \nabla(\mathcal{I}_h u)\|_{L^2(\Omega)}$.



(a) $\|\nabla u - \nabla(\mathcal{I}_h u)\|_{L^2(\Omega)} := 6.45e-03$

(b) Projected error (Gauss Pts.), $\|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)} := 8.87e-04$

(c) Computed derivative superconvergence points

(d) Projected error (Sup. Pts.), $\|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)} := 1.42e-13$

Figure 21: Two dimensional case: The consistency of SPR recovery operator considered as G_h here; *a* L^2-norm of the error between the gradient of u and its projection $\mathcal{I}_h u$, (b) SPR operator G_h with (2×2)-Gauss Legendre points as sampling point on a general non-uniform LR mesh does not satisfies the consistency condition, (c) computed derivative superconvergence points using computer based algorithm (d) SPR operator G_h with computed derivative superconvergence points as sampling point on a general non-uniform LR mesh does satisfies the consistency condition. Here we have considered $u = (2x^3 - 3x^2) + (2y^3 - 3y^2) \in \mathbb{P}_3(\Omega)$ and $\mathcal{I}_h$ a local Neumann projection of u in quadratic LR B-spline space.

The results of this section is useful in establishing the superconvergence behavior of the SPR recovery procedure. Using the above consistency condition with Bramble-Hilbert lemma we can obtain the following results.

Lemma 4. *Let $u \in H^{p+2}(\Omega) \cap H_0^1(\Omega)$ be the given exact solution and $\mathcal{I}_h u$ be its elliptic Neumann projection in $\mathcal{S}^p_{\Delta_h,0}$. Suppose that G_h operator satisfies (i)-(iii), then*

$$\|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)} \leq Ch^{p+1}\|u\|_{H^{p+2}(\Omega)}. \tag{123}$$

Proposition 1. *Let $u \in H^{p+2}(\Omega) \cap H_0^1(\Omega)$ be the exact solution of the elliptic problem (17)-(19) and $u_h \in \mathcal{S}^p_{\Delta_h,0}$ be the spline based FE solution (22). Then*

$$\|\nabla u - G_h(u_h)\|_{L^2(\Omega)} \leq Ch^{p+1}\|u\|_{H^{p+2}(\Omega)} \tag{124}$$

Proof. The proof is based on triangle inequality. It can be written as

$$\|\nabla u - G_h(u_h)\|_{L^2(\Omega)} \leq \|\nabla u - G_h(\mathcal{I}_h u)\|_{L^2(\Omega)} + \|G_h(\mathcal{I}_h u) - G_h(u_h)\|_{L^2(\Omega)}.$$

Using the results of Lemma 4 and, the boundedness and linearity condition (iii) with quasi-uniform mesh having an upper bound on G_h independent of h, we obtain

$$\|\nabla u - G_h(u_h)\|_{L^2(\Omega)} \leq Ch^{p+1}\|u\|_{H^{p+2}(\Omega)} + C\|\nabla \mathcal{I}_h u - \nabla u_h\|_{L^2(\Omega)}, \tag{125}$$

where C is a constant independent of h.

Now assuming the superconvergence property in FE solution u_h, which is true under certain regularity conditions regarding the partition Δ_h, the regularity of the true solution and the topology of the mesh, an estimate of the following form holds:

$$\|\nabla u_h - \nabla \mathcal{I}_h u\|_{L^2(\Omega)} \leq C(u)h^{p+1}, \tag{126}$$

while a priori error estimates on the other hand gives $\|\nabla u - \nabla u_h\|_{L^2(\Omega)} \leq C(u)h^p$.

We can obtain the required result (124) by inserting (126) into (125). □

Remark 6.1. *The results of Proposition 1 shows the superconvergence in the SPR recovery procedure of order 1 which can be obtained in very special cases, e.g. with uniform mesh topology and with enough regularity of the solution. In general, on practical quasi-uniform meshes, the results may reduce to*

$$\|\nabla u - G_h(u_h)\|_{L^2(\Omega)} \leq C(u)h^{p+\alpha}, \tag{127}$$

where $\alpha \in (0, 1]$. Then the SPR recovery procedure is superconvergent of order α instead of order 1.

The proof of Proposition 1 was given under the assumption of the superconvergence property of FE solution (126), which is definitely a subject of further research for isogeometric discretization. But in the numerical results of Section 7 we verify the result of Proposition 1 for problems with smooth solution.

7 Numerical results

In this section we report some numerical studies to demonstrate the accuracy of the recovered derivatives achieved by the proposed recovery procedures and their rates of convergence. The main focus will be to study the superconvergence behavior of recovery procedures and the performance of recovery based error estimators developed in this article. We have divided numerical results section into three main parts where we will study the followings:

- Superconvergence behavior of gradient recovery procedures under uniform h-refinement.
- Superconvergence behavior of gradient recovery procedures under adaptive meshes obtained through *Structured mesh* refinement strategy of LR B-splines.
- Adaptive isogeometric analysis using a posteriori error estimators proposed in this article.

We consider several numerical examples for model elliptic problems with available exact solution. The main study involve the computation of the exact energy error

$$\|e\|_E = \|\nabla u - \nabla u_h\|_{L^2(\Omega)}, \tag{128}$$

the error in recovered gradient field

$$\|e^*\|_E = \|\nabla u - \nabla u_h^*\|_{L^2(\Omega)}, \tag{129}$$

with the error estimator and element indicator defined by

$$\eta^* = \left(\sum_{el=1}^{Nel} (\eta^*_{el})^2 \right)^{1/2} \quad \text{where} \quad \eta^*_{el} = \|\nabla u_h^* - \nabla u_h\|_{L^2(\Omega_{el})}. \tag{130}$$

We also consider the global and local effecitivity indices to measure the quality of recovered solution field, which are defined by

$$\theta^* = \frac{\eta^*}{\|\nabla u - \nabla u_h\|_{L^2(\Omega)}} \quad \text{and} \quad \theta^*_{local} = \frac{\|\nabla u_h^* - \nabla u_h\|_{L^2(\Omega_{el})}}{\|\nabla u - \nabla u_h\|_{L^2(\Omega_{el})}}. \tag{131}$$

Here the superscript $*$ represents the recovery procedures and will be replaced appropriately by $* =$ CL2P, DLSF, SPR in the article.

Marking strategy

The marking strategy, that is, the method of how to choose the basis functions for refinement in structured mesh refinement is taken from [39], where once we have the value of estimated error at element level given by Eq. (130), then we sum the element error on all elements within the support of each basis function. In the refinement strategies we always choose to refine some percentages of basis functions which contributed the most error in FEM computation. In [39], it was demonstrated that for a fixed percentage say $\beta = 5, 10, 20$ one always achieved a proper adaptive refinement process resulting in optimal convergence rates. For the implementation in this article we always consider $\beta = 5$ of the basis function to refine in each adaptive refinement steps as this choice doesn't give over-refinements.

7.1 Superconvergent gradient recovery under uniform h-refinement

We consider three test examples with given smooth solution. As we have mentioned above the aim here is to show the superconvergent behavior of gradient recovery procedures under uniform h-refinement. For this we performed the FE computation using quadratic B-spline spaces under uniform h-refinement. The location of optimal sampling points, i.e., the derivative superconvergence points at (2×2) Gauss Legendre points for this case has been explored in Section 5. Thus we considered (2×2)-Gauss Legendre points as the sampling points in our recovery procedure of DLSF and SPR defined in Section 4.

Example 1. (One dimensional problem)

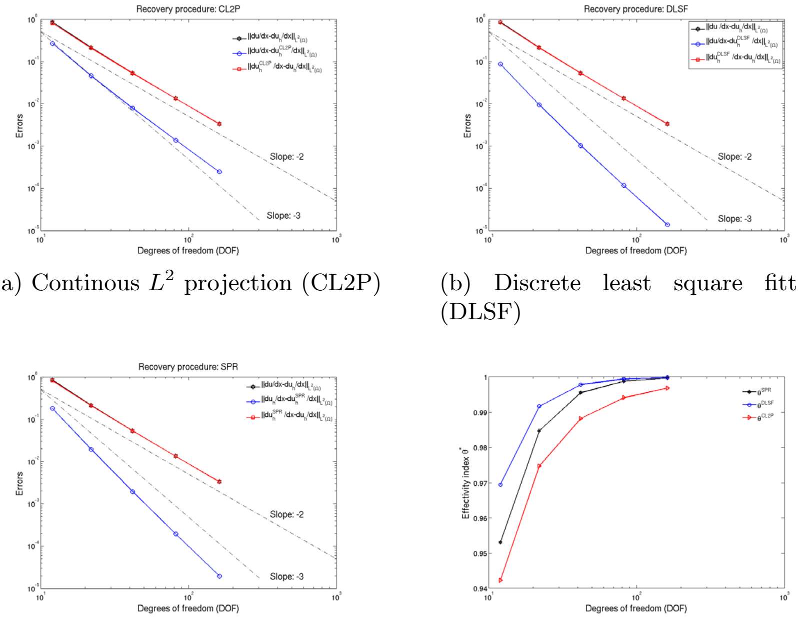
We consider the following one dimensional problem

$$-\frac{d^2u}{dx^2} = f \quad \text{on } I = (0,1), \quad \text{with} \quad u(0) = 0, \quad \text{and} \quad u(1) = 0. \tag{132}$$

The function f is chosen so that the exact solution is of the form

$$u = x^2 - \frac{\sinh 4x}{\sinh 4}. \tag{133}$$

The error plots of exact error, projected error and estimated error for the recovery procedures $* =$CL2P, DLSF, SPR are shown in Figures 22(a)-(c), respectively. The performance of the effectivity index θ^* for $* =$CL2P, DLSF,



(a) Continous L^2 projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent path recovery (SPR)

(d) Effectivity index θ^*

Figure 22: **One dimensional problem**: Errors and effectivity index results obtained with different recovery procedures (CL2P, DLSF and SPR) using quadratic B-spline space to approximate the solution u_h with uniform h-refinements.

SPR are shown in Figure 22(d). From the error plots it can be notice that the projected error for CL2P recovery procedure is not superconvergent of order one; although a more accurate result than exact error is achieved. While the error plots shown in Figure 22(b)-(c) clearly shows the superconvergent behavior of DLSF and SPR recovery procedures. This results in asymptotic exactness of estimated error with respect to uniform h-refinement, where DLSF and SPR recovery procedure are fastly approaching to the value 1. Although the CL2P recovery procedure does not produce a superconvergent recovered gradient of one order higher convergence rate, it still gives a fairly accurate error estimate. The reason being that the recovered gradient is accurate enough to give an estimated error in energy quite close to the exact one. The numerical results also demonstrate that to achieve an asymptotically exact a posteriori error estimator in the energy norm, the recovered derivatives need not necessarily be superconvergent of order 1 at every points. An example of global continuous L^2 projection (CL2P) fits in this category.

Example 2. (Sinus problem)

Next we consider the following two dimensional problem

$$-\Delta u = f \quad \text{in } \Omega, \tag{134}$$

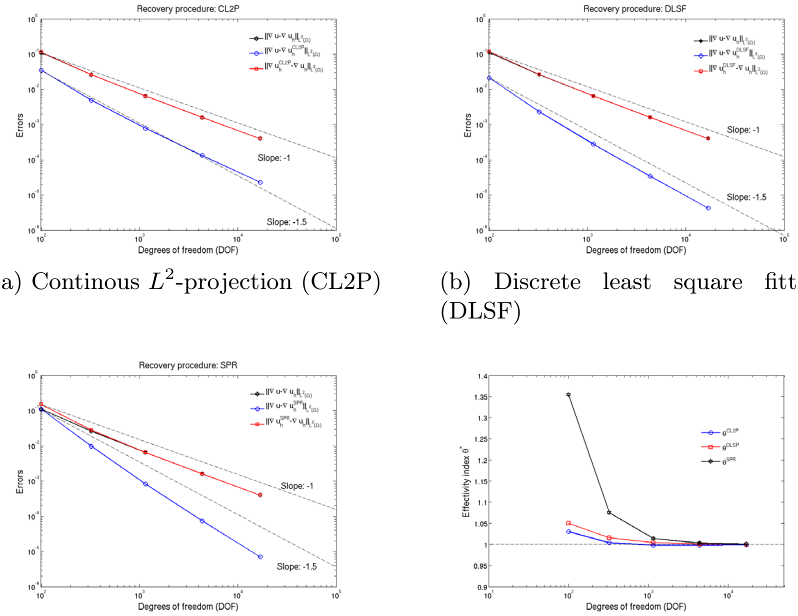
with homogenous boundary conditions

$$u = 0 \quad \text{on} \;\; \partial\Omega. \tag{135}$$

Here $\Omega = (0,1) \times (0,1)$ is considered as a square domain and f is constructed to correspond to the exact solution

$$u(x,y) = \sin(2\pi x)\sin(2\pi y). \tag{136}$$

The error plots of exact error, projected error and estimated error for the recovery procedures $* =$ CL2P, DLSF, SPR are shown in Figures 23(a)-(c), respectively. While the performance of the effectivity index θ^* for $* =$ CL2P, DLSF, SPR are shown in Figure 23(d). From the error plots, similar to 1D case, it can be noticed that the projected error for CL2P recovery procedure is not superconvergent of order one; although a more accurate result than exact error is achieved. While the error plots shown in Figures 23(b)-(c) clearly show the superconvergent behavior of DLSF and SPR recovery procedures. This results in asymptotic exactness of estimated error with respect to uniform h-refinement, where DLSF and SPR recovery procedure are rapidly approaching to the value 1. The CL2P recovery procedure show similar behavior as for the One dimensional example, see the comment given above regarding the effectivity index. The comparison of distribution of exact error vs. estimated error (at element level) obtained by different recovery procedures for quadratic B-splines approximate solution u_h at step 2 of uniform refinements is shown in Figure 24, whereas the comparison of deviation of local effectivity index $|\theta^*_{local} - 1|$ obtained by different recovery procedures at step 2 of uniform refinement is shown in Figure 25. The results displayed in Figures 24-25 indicates that for smooth problems with uniform h-refinement the DLSF recovery procedure gives better results in comparison to the two other recovery procedures. The accuracy of the SPR procedure is very good inside the domain. However, some disturbances may be observed along the boundary. This is well known behavior of the SPR-procedure and might be handled by introducing special recovery schemes for the patches along the boundary (see e.g. [44] and [49]). Notice that the global recovered gradient field using SPR (without special treatments of the boundaries) is still superconvergent of order one.



(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 23: **Sinus problem**: Errors and effectivity index results obtained with different recovery procedures (CL2P, DLSF and SPR) using quadratic B-spline space to approximate the solution u_h with uniform h-refinements.

In this example we also notice that the estimated error for the recovery procedures is conservative and giving a bound on the exact error from above. Herein we do not guarantee that our error estimates are bounding the exact error (either from above or below). We focus instead on h-asymptotic exactness. Guaranteed upper and lower bounds is a topic for future investigation - see the discussion in the end of this article.

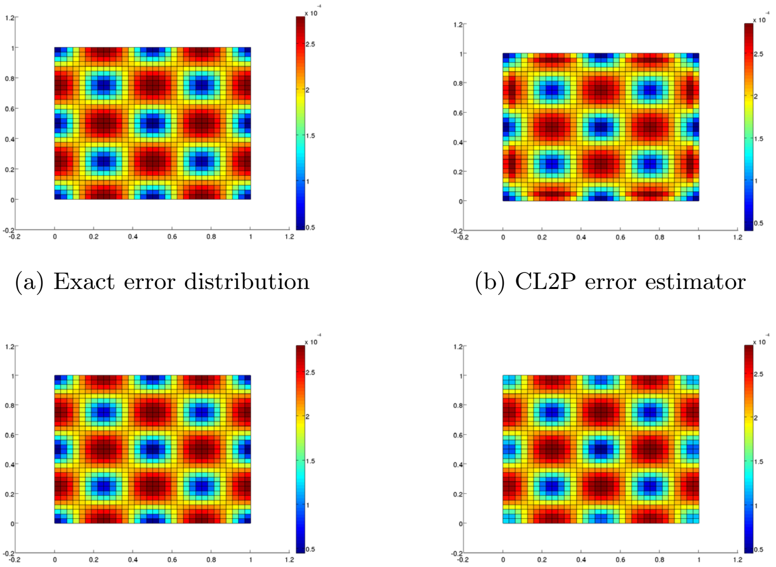
Example 3. (Smooth solution with non-homogeneous Dirichlet Bc's)

Now we consider the elliptic problem (134) on the square domain $\Omega = (0,1)\times(0,1)$ with non-homogenous boundary conditions

$$u = g \quad \text{on} \quad \partial\Omega, \tag{137}$$

f and g are constructed to correspond to the exact solution

$$u(x,y) = (x^3 + y^2)\sin(xy). \tag{138}$$



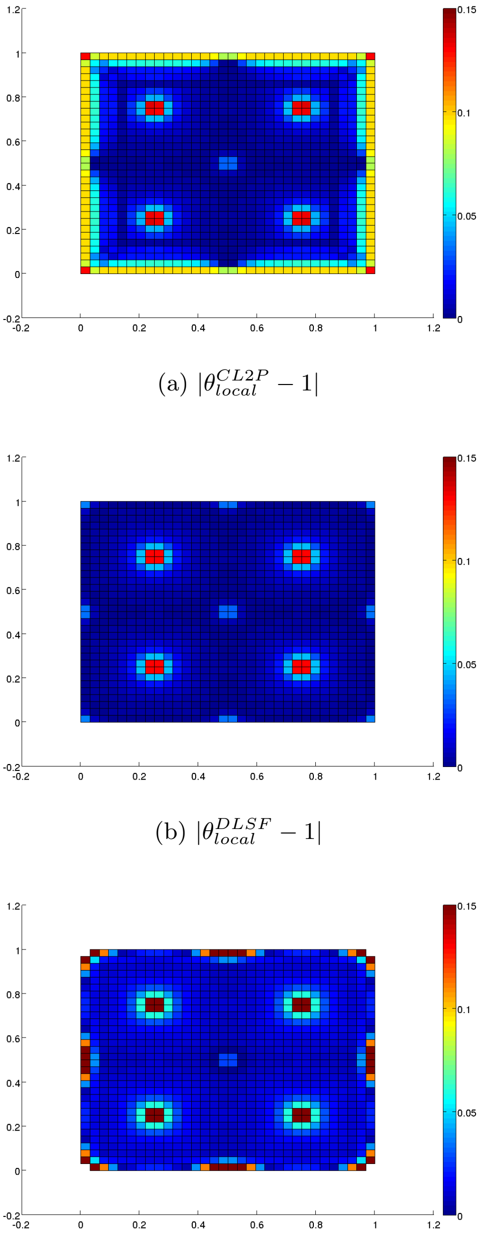
(a) Exact error distribution

(b) CL2P error estimator

(c) DLSF error estimator

(d) SPR error estimator

Figure 24: **Sinus problem**: Comparison of distribution of exact error vs. error estimator (at element level) obtained by Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) for quadratic B-splines approximate solution u_h at step 2 of uniform refinements.



(a) $|\theta_{local}^{CL2P} - 1|$

(b) $|\theta_{local}^{DLSF} - 1|$

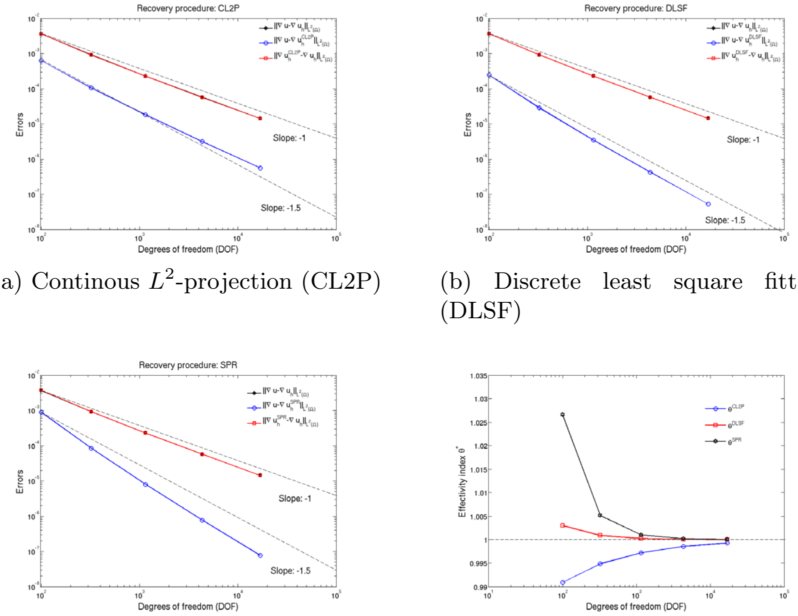
(c) $|\theta_{local}^{SPR} - 1|$

Figure 25: **Sinus problem**: Comparison of deviation of local effectivity index $|\theta_{local}^{*} - 1|$ obtained by Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) for quadratic B-splines approximate solution u_h at step 2 of uniform refinements.

The comparison of error plots, global effectivity index θ^* and deviation of local effectivity index $|\theta^*_{local} - 1|$, for $*$=SPR, DLSF, CL2P, recovery procedures are plotted in Figures 26-28. It can be observed that the global effectivity index θ^* converges to one rapidly for both the DSLF and SPR procedures after the mesh has been refined two times. The effectivity index for CL2P is also approaching one after some refinements, but with slightly less rate.

Discussion of results obtained for smooth problems

The numerical results obtained in Example 1-3 all confirm that the recovered gradient using DLSF and SPR are superconvergent of one order for smooth problems, whereas CL2P recover gradients are definitely more accurate than exact errors in computed FE solutions, and sometimes superconvergence of order between zero and one (here close to one) is also observed. All the recovery schemes are shown to be h-asymptotic exact for such smooth problems. However, the deviation from unity of the local effectivity index is pronounced along the boundary for SPR in all cases, whereas the DLSF handles homogenous Dirichlet boundary conditions quite well but not non-homogenous Dirichlet. We just remark that there exist remedies for handling the issues of boundary conditions for SPR ([44, 49]).



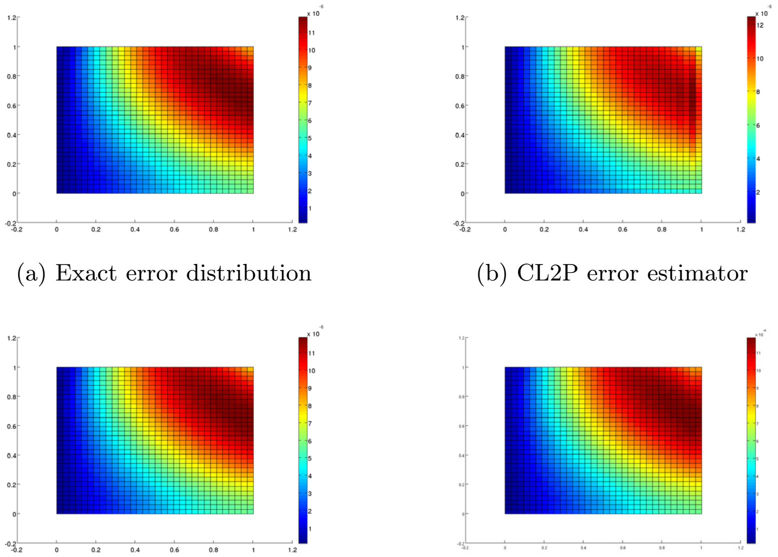
(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent path recovery (SPR)

(d) Effectivity index θ^*

Figure 26: **Smooth problem with non-homogenous Bc's**: Errors and effectivity index results obtained with different recovery procedures (CL2P, DLSF and SPR) using quadratic B-spline space to approximate the solution u_h with uniform refinements.



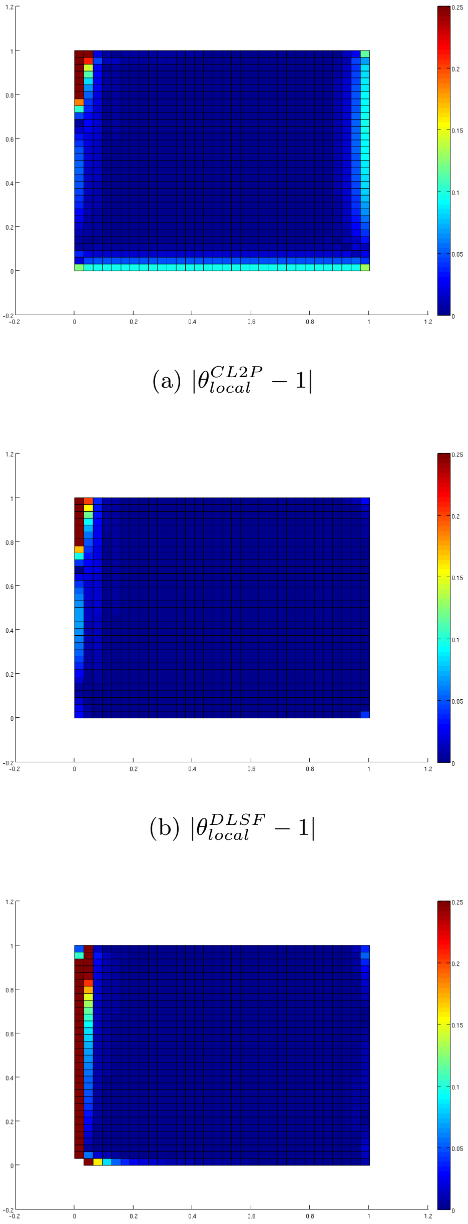
(a) Exact error distribution

(b) CL2P error estimator

(c) DLSF error estimator

(d) SPR error estimator

Figure 27: **Smooth problem with non-homogenous Bc's**: Comparison of distribution of exact error vs. error estimator (at element level) obtained by Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) for quadratic B-splines approximate solution u_h at step 3 of uniform refinements.

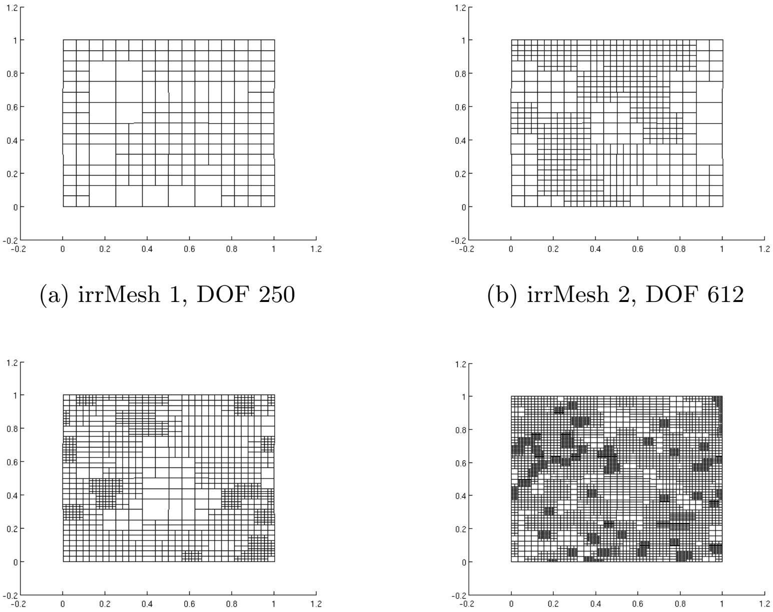


(a) $|\theta_{local}^{CL2P} - 1|$

(b) $|\theta_{local}^{DLSF} - 1|$

(c) $|\theta_{local}^{SPR} - 1|$

Figure 28: **Smooth problem with non-homogenous Bc's**: Comparison of deviation of local effectivity index $|\theta_{local}^{*} - 1|$ obtained by Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) for quadratic B-splines approximate solution u_h at step 3 of uniform refinements.



(a) irrMesh 1, DOF 250

(b) irrMesh 2, DOF 612

(c) irrMesh 3, DOF 1173

(d) irrMesh 4, DOF 1785

Figure 29: A set of randomly generated irregular meshes for quadratic LR B-spline elements.

7.2 Superconvergent gradient recovery under adaptive meshes

Example 4. (Smooth problem on irregular meshes)

To investigate the significance of using true superconvergent points compared to classical Gauss integration points (i.e. (2×2) for p=2) we redo the *Smooth solution with non-homogenous BC's* using quadratic LR B-splines defined on a set of arbitrary given irregular meshes as displayed in Figure 29. In Tables 2-4, we report the obtained error norms as well as the effectivity indices θ^*. When using the (2×2)-Gauss Legendre points the DLSF and SPR gives results in the range $0.75 - 0.96$ and $0.80 - 1.02$, respectively. Furthermore, we see that for all except second irregular mesh the effectivity index for SPR using (2×2)-Gauss Legendre points is in the range $0.96 - 1.02$. The corresponding results when using true superconvergent points are $0.96 - 1.04$ and $0.98 - 1.02$ for DLSF and SPR, respectively. Thus, using true superconvergent points clearly improve the effectivity indices for both DLSF and SPR for the given problem. However, in practice when the meshes are refined in order to smooth the error distribution we might expect less differences, and this issue will be investigated below. Finally we remark that

CL2P in general gives the less reliable effectivity indices.

Meshes	Exact error $\|\nabla u - \nabla u_h\|_{L^2(\Omega)}$	Projected error $\|\nabla u - \nabla u_h^{CL2P}\|_{L^2(\Omega)}$	Error estimate $\|\nabla u_h^{CL2P} - \nabla u_h\|_{L^2(\Omega)}$	Effectivity index θ^{CL2P}
irrMesh 1	1.56e-03	7.05e-04	1.41e-03	0.90
irrMesh 2	1.78e-03	1.27e-03	1.25e-03	0.70
irrMesh 3	9.17e-04	5.87e-04	7.09e-04	0.77
irrMesh 4	1.47e-03	4.66e-04	1.40e-03	0.95

Table 2: **Smooth problem with non-homogenous Bc's**: Comparison of errors and effectivity index θ^{CL2P} for Continous L^2 projection (CL2P) procedure on randomly generated irregular meshes of LR B-splines spaces.

Meshes	Sampling points	Exact error $\|\nabla u - \nabla u_h\|_{L^2(\Omega)}$	Projected error $\|\nabla u - \nabla u_h^{DLSF}\|_{L^2(\Omega)}$	Error estimate $\|\nabla u_h^{DLSF} - \nabla u_h\|_{L^2(\Omega)}$	Effectivity index θ^{DLSF}
irrMesh	Gauss pts.	1.56e-03	5.59e-04	1.43e-03	0.92
1	Sup. pts.		3.02e-04	1.57e-03	1.01
irrMesh	Gauss pts.	1.78e-03	8.74e-04	1.35e-03	0.75
2	Sup. pts.		2.44e-04	1.71e-03	0.96
irrMesh	Gauss pts.	9.17e-04	4.89e-04	7.25e-04	0.79
3	Sup. pts.		2.88e-04	9.55e-04	1.04
irrMesh	Gauss pts.	1.47e-03	4.12e-04	1.41e-03	0.95
4	Sup. pts.		3.40e-04	1.42e-03	0.96

Table 3: **Smooth problem with non-homogenous Bc's**: Comparison of errors and effectivity index θ^{DLSF} for Discrete least square fitting (DLSF) procedure on randomly generated irregular meshes of LR B-splines spaces.

Meshes	Sampling points	Exact error $\|\nabla u - \nabla u_h\|_{L^2(\Omega)}$	Projected error $\|\nabla u - \nabla u_h^{SPR}\|_{L^2(\Omega)}$	Error estimate $\|\nabla u_h^{SPR} - \nabla u_h\|_{L^2(\Omega)}$	Effectivity index θ^{SPR}
irrMesh	Gauss pts.	1.56e-03	5.48e-04	1.52e-03	0.97
1	Sup. pts.		4.21e-04	1.59e-03	1.01
irrMesh	Gauss pts.	1.78e-03	9.45e-04	1.44e-03	0.80
2	Sup. pts.		2.77e-04	1.75e-03	0.98
irrMesh	Gauss pts.	9.17e-04	2.92e-04	8.85e-04	0.96
3	Sup. pts.		9.61e-05	9.17e-04	1.00
irrMesh	Gauss pts.	1.47e-03	3.84e-04	1.50e-03	1.02
4	Sup. pts.		3.07e-04	1.51e-03	1.02

Table 4: **Smooth problem with non-homogenous Bc's**: Comparison of errors and effectivity index θ^{SPR} for Superconvergent patch recovery (SPR) procedure on randomly generated irregular meshes of LR B-splines spaces.

Example 5. (Smooth problem on adaptive meshes)

To investigate the significance of using true superconvergent points compared to classical Gauss integration points (i.e. (2×2) for p=2) on realistic adapted meshes based on exact error distribution (see Figure 30) we redo the *Smooth solution with non-homogenous BC's* using quadratic LR B-splines. In Table 5 we report the obtained effectivity indices θ^* for different recovery procedures. When using the (2×2)-Gauss Legendre points the DLSF and SPR gives results in the range $0.97 - 1.00$ and $0.99 - 1.02$, respectively. The corresponding results when using true superconvergent points are $1.00 - 1.00$ and $1.00 - 1.02$ for DLSF and SPR, respectively. Thus, using true superconvergent points again improve the effectivity indices for both DLSF and SPR, but only marginally. However, as seen from Figure 31 the use of true superconvergent points within DLSF and SPR slightly lower the error in recovered gradients as well as make the error estimates slightly more conservative. Finally, we remark that CL2P in general again gives the less reliable effectivity indices $(0.96 - 0.99)$, but this time fairly close to one.

On comparing the results obtained on these practical meshes with the results obtained on irregular set of meshes shown in Figure 29, we see that the results especially with the choice of (2×2)-Gauss Legendre points in our recovery procedures of DLSF and SPR improved a lot and the effectivity indices obtained on these practical meshes will always be close to unity or approaching to unity in h-asymptotic sense. This results remind us the smooth solution definition from [8] with respect to a given discrete mesh. A given smooth solution on a particular irregular mesh can also behave like a non-smooth solution; in contrary a given non-smooth solution on a proper generated adapted meshes can behave like a smooth solution. Our experience is (see e.g. [44, 49]) that the reliability of the DLSF and SPR procedures improves throughout an adaptive refined mesh-sequence based on proper a posteriori error estimates. We believe that is the reason for why DLSF and SPR performs better in all the adapted meshes in Figure 30 compared to the irregular meshes displayed in Figure 29.

The above case was studied for the problem with smooth solution. Now we consider the problem with *rough right hand side* which exhibits internal layer behavior given by Example 6 below.

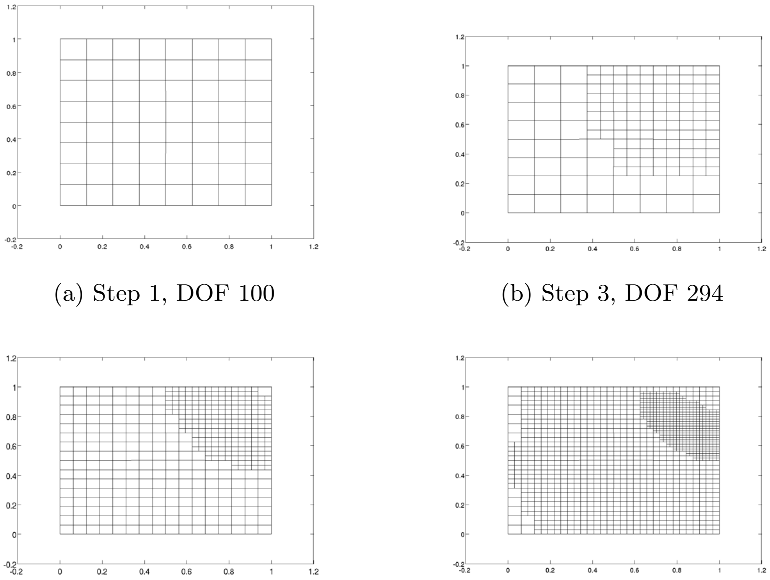
Example 6. (Circular wave front problem on adaptive meshes)

The governing equation of the problems is

$$-\Delta u = f \quad \text{in } \Omega, \tag{139}$$

Steps	θ^{CL2P}	θ^{DLSF}	θ^{DLSF}	θ^{SPR}	θ^{SPR}
		Gauss pts.	Sup. pts.	Gauss pts.	Sup. pts.
1	0.99	1.00	1.00	1.02	1.02
3	0.99	0.99	1.00	1.01	1.02
5	0.96	0.97	1.00	0.99	1.00
7	0.99	0.99	1.00	1.00	1.01

Table 5: **Smooth problem with non-homogenous Bc's**: Effectivity index θ^* for Continous L^2-projection (CL2P) and, Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) procedures using (2×2)-Gauss Legendre points and computed derivative superconvergence points (Sup. pts.) on adaptive meshes of Figure 30.



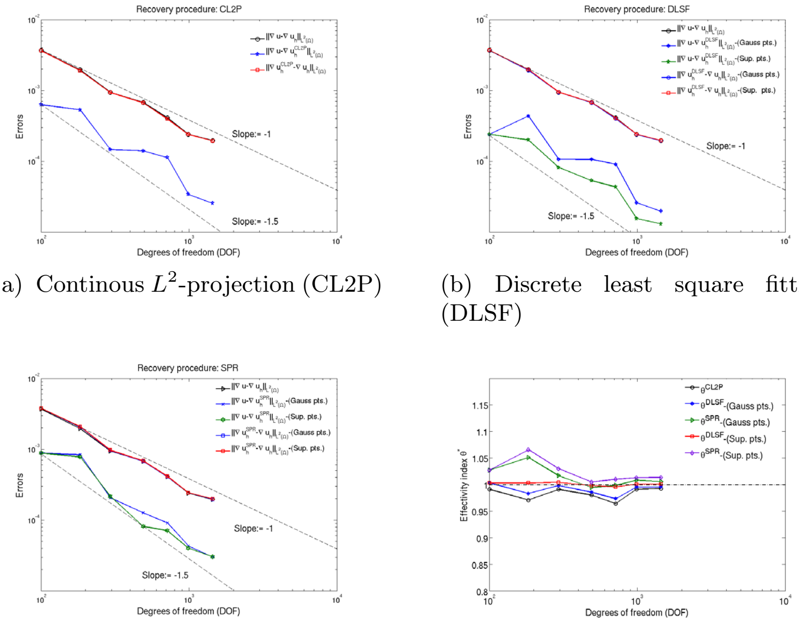
(a) Step 1, DOF 100

(b) Step 3, DOF 294

(c) Step 5, DOF 716

(d) Step 7, DOF 1447

Figure 30: **Smooth problem with non-homogenous Bc's**: Meshes obtained by means of adaptive refinements based on exact error estimate and the use of LR B-splines at different steps.



(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 31: **Smooth problem with non-homogenous Bc's**: Comparison of error plots and effectivity index by Continous L^2-projection (CL2P) and, Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) using (2×2)-Gauss Legendre points and computed derivative superconvergence points in adaptive isogeometric analysis using quadratic LR B-splines based of exact error estimator.

Steps	θ^{CL2P}	θ^{DLSF}	θ^{DLSF}	θ^{SPR}	θ^{SPR}
		Gauss pts.	Sup. pts.	Gauss pts.	Sup. pts.
1	1.63	1.67	1.67	2.24	2.24
3	0.98	1.00	1.06	1.28	1.35
7	0.96	0.99	1.00	1.04	1.05
10	0.98	0.99	1.00	1.01	1.01

Table 6: **Circular wave front problem**: Effectivity index θ^* for Continous L^2-projection (CL2P) and, Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) procedures using (2×2)-Gauss Legendre points and computed derivative superconvergence points (Sup. pts.) on adaptive meshes of Figure 32.

with homogenous boundary conditions

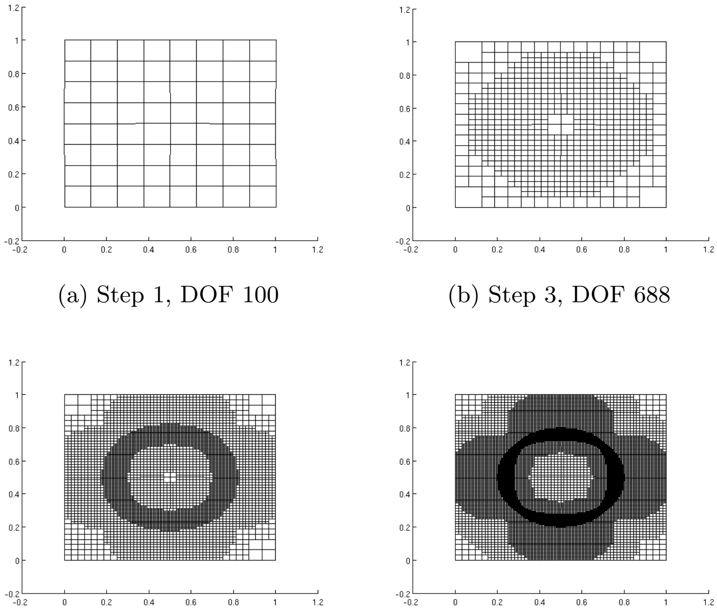
$$u = g \quad \text{on} \quad \partial\Omega. \tag{140}$$

Here $\Omega = (0,1)^2$ is a square domain and f, g are constructed to correspond to the exact solution

$$u(x,y) = \arctan(S(r - r_0)), \text{ where } r = \sqrt{(x-1/2)^2 + (y-1/2)^2}. \tag{141}$$

with $r_0 = 1/16$ is the distance from the wave front to the center of the circle, and $S = 20$ gives a mild steepness of the wave front along the circular region in the interior domain Ω.

The comparison of the performance of the error estimators on a set of quasi-uniform meshes obtained via the adaptive refinement algorithm using exact error estimate with structured mesh refinement strategy of LR B-splines of degree two are presented in Figures 32-33. The results with respect to two choices of sampling points, i.e., (2×2)-Gauss Legendre points and computed derivative superconvergence points (Sup. pts.), for DLSF and SPR recovery procedures are also given in Table 6. It can be seen from the figures that both choices of sampling points in our recovery procedures gives good effectivity index and provides h-asymptotic results as expected. However, the use of true (computed) derivative superconvergence points again provides slightly better accuracy and convergence in the recovered gradient field and slightly faster asymptotic exactness results.



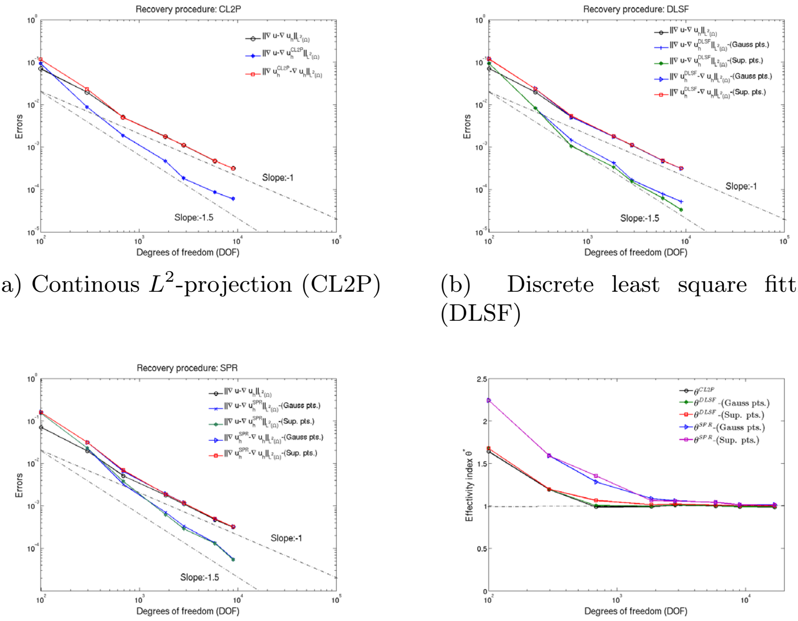
(a) Step 1, DOF 100

(b) Step 3, DOF 688

(c) Step 7, DOF 5880

(d) Step 10, DOF 16809

Figure 32: **Circular wave front problem**: Meshes obtained by means of adaptive refinements based on exact error estimate and the use of LR B-splines at different steps.



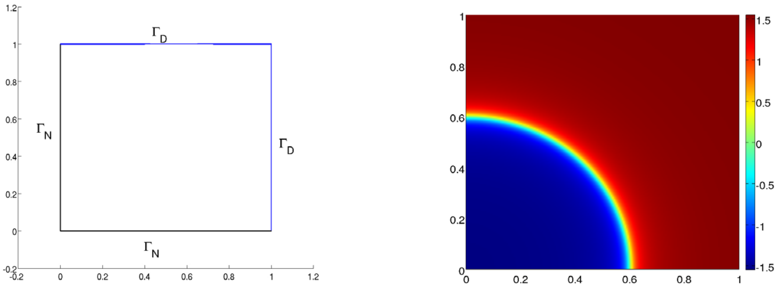
(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 33: **Circular wave front problem**: Comparison of error plots and effectivity index by Continous L^2-projection (CL2P) and, Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR) using Gauss points and computed derivative superconvergence points in adaptive isogeometric analysis using quadratic LR B-splines based of exact error estimator.



(a) Internal layer problem description

(b) Exact solution u

Figure 34: **Internal layer problem**: Problem description and Exact solution u.

7.3 Adaptive isogeometric analysis using a posteriori error estimators

The purpose of this section is to demonstrate the performance of developed recovery based error estimators in an adaptive isogeometric analysis, in particular, we show that the developed SPR procedure does produce adaptive meshes on which the recovered gradient is indeed superconvergent. We present the numerical results with respect to each developed recovery based error estimator $\eta^* = \|\nabla u_h^* - \nabla u_h\|_{L^2(\Omega)}$, with $* =$ CL2P, DLSF, SPR, using quadratic LR B-spline based structured mesh refinement algorithm. As we have seen from the numerical results of presented above that the performance of DLSF and SPR recovery procedure with the choice of (2×2)-Gauss Legendre points as sampling points is very comparable to the case when true computed derivative superconvergence points are used. The computed derivative superconvergence points is is a bit cumbersome to use in practice (complicates in some cases the choice of patches). For these reasons we recommend to use the (2×2)-Gauss Legendre points as sampling points in DLSF and SPR recovery procedure for quadratic LR B-splines.

Example 7. (Internal layer problem)

The governing equation of internal layer problem is

$$-\Delta u = f \quad \text{in } \Omega := (0,1)^2, \tag{142}$$

with the boundary conditions

$$u = u_d \quad \text{on} \quad \Gamma_D \quad \text{and} \quad \frac{\partial u}{\partial n} = g \quad \text{on} \quad \Gamma_N, \tag{143}$$

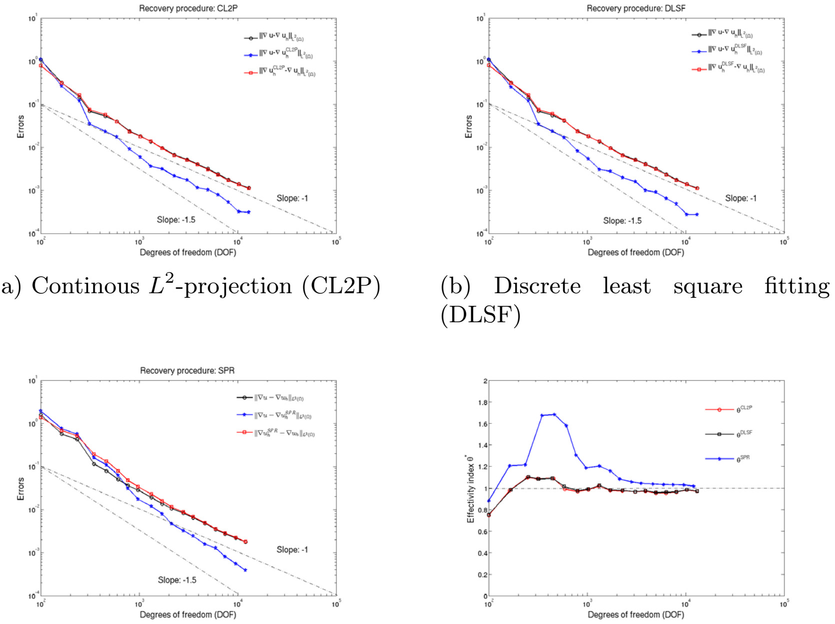
Here the boundary data u_d, g and f are constructed to correspond to the exact solution

$$u(x, y) = \arctan\left(S\left(\sqrt{x^2 + y^2} - 0.60\right)\right). \tag{144}$$

We consider the problem with $S = 60$ in the exact solution u, which exhibits the curved internal layer of width $O(1/60)$ in interior of the domain. The set up of the problem with given boundary conditions and the exact solution u are shown in Figure 34.

The comparison of error plots and effectivity index of different recovery procedures using respective recovery based error estimators are shown Figure 35. The results obtained for the error in projected field is more comparable between the recovery procedures. With SPR recovery we obtained higher rate of convergence in projected error in comparison to other recovery procedures, while the results for CL2P and DLSF is initially more accurate than the SPR recovery. This problem exhibits an internal layer of width $O(1/60)$ due to the rough right hand side function f. In order to obtain optimal convergence rate, i.e. $O(h^{p+1})$, the internal layer has to be resolved. For uniform refined meshes this means when $h < 1/60$. Starting with $h = 1/8$ it means 3 uniform mesh refinement steps resulting in $h = 1/64$ which is about $DOF = 66^2 = 4356$ degrees of freedom. Using the developed a posteriori error estimates we starts to resolve the internal layer after 6 adaptive refinement steps where $h_{max} = 1/8$ and $h_{min} = 1/64$ with approximately $DOF = 800$ degrees of freedom. That we start resolving the internal layer after 6 refinement steps is seen from the fact that the effectivity indices are starting approaching to 1 at this stage in the adaptive process, see Figure 36. The observed behavior of the SPR-procedure for coarse meshes i.e. before the internal layer is properly resolved complies well with other investigations, see e.g. [78] and [44]. The SPR-procedure works well when we have smooth error distribution throughout the mesh. Thus, for non-smooth problems the SPR-procedure can only give good effectivity indices after some initial refinement steps that get rid of any present pollution. All known experience from more than two decades of use of SPR-procedures has shown that adaptive finite element methods based on SPR error estimates are able to achieve smooth error distribution for non-smooth problems having singularity points/lines or rough right hand sides.

The comparison of the deviation in local effectivity index for these recovery procedures at various steps are presented in Figure 37. We observe that not only gives the three different recovery procedures good global effectivity indices but also the local effectivity index at element level is fairly close to



(a) Continous L^2-projection (CL2P)

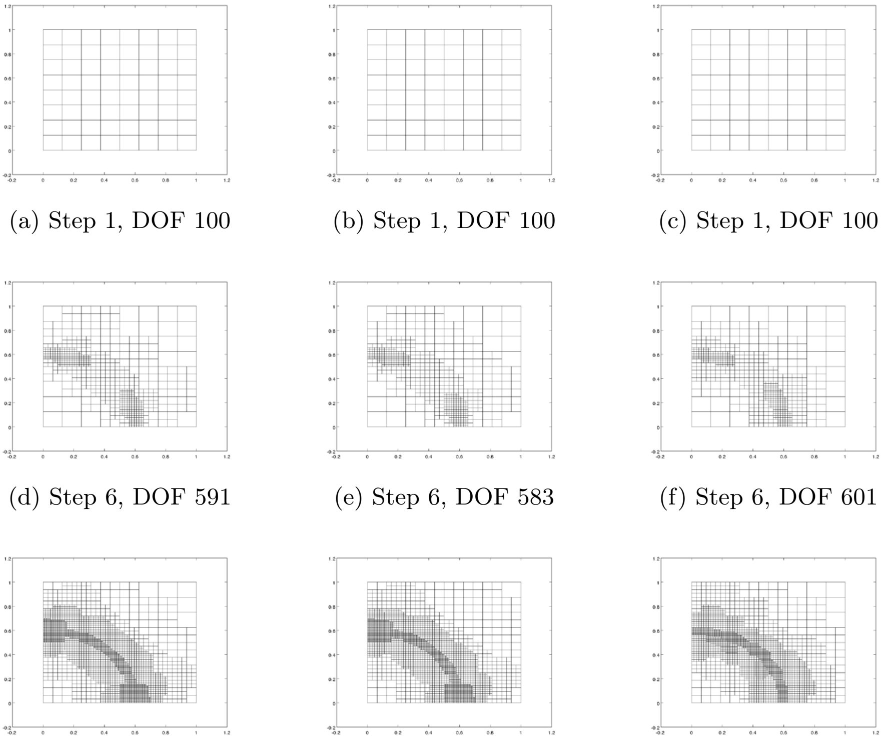
(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 35: **Internal layer problem**: Error plots and effectivity index for different recovery based a posteriori error estimators in adaptive isogeometric analysis using quadratic LR B-splines.

one. However, along the boundary the results for SPR are less accurate then the other two methods -this complies with the observation done in earlier examples described above. In any case all the recovery procedure capture very well the location of the internal layer and the numerical solution based on adaptive refinement using these error estimates all attain optimal rate of convergence.



(a) Step 1, DOF 100

(b) Step 1, DOF 100

(c) Step 1, DOF 100

(d) Step 6, DOF 591

(e) Step 6, DOF 583

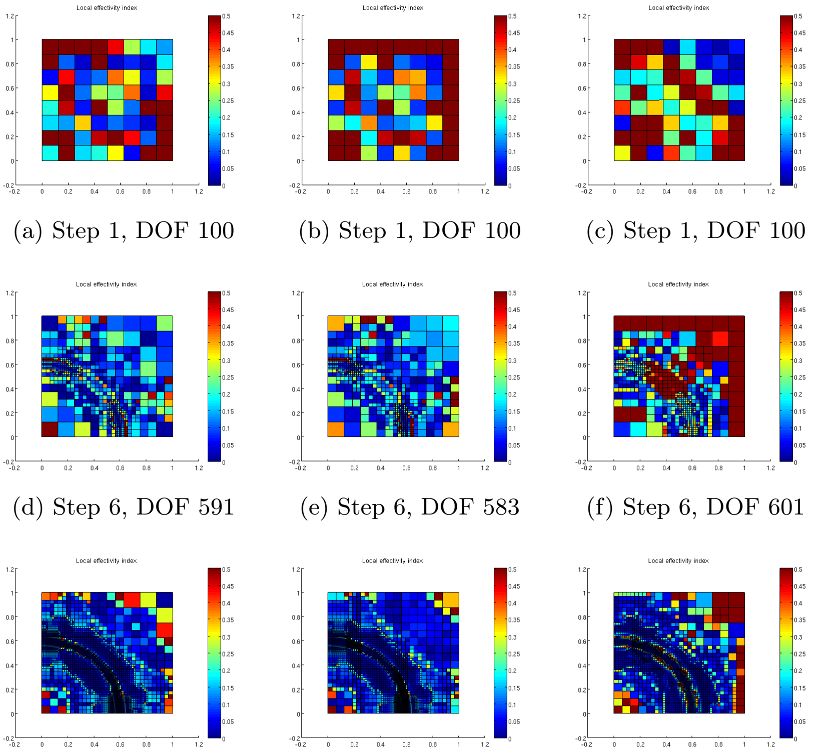
(f) Step 6, DOF 601

(g) Step 12, DOF 3072

(h) Step 12, DOF 3094

(i) Step 12, DOF 2660

Figure 36: **Internal layer problem**: Adapted LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for Internal layer problem. The columns from left to right represents the cases with respect to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.



(a) Step 1, DOF 100

(b) Step 1, DOF 100

(c) Step 1, DOF 100

(d) Step 6, DOF 591

(e) Step 6, DOF 583

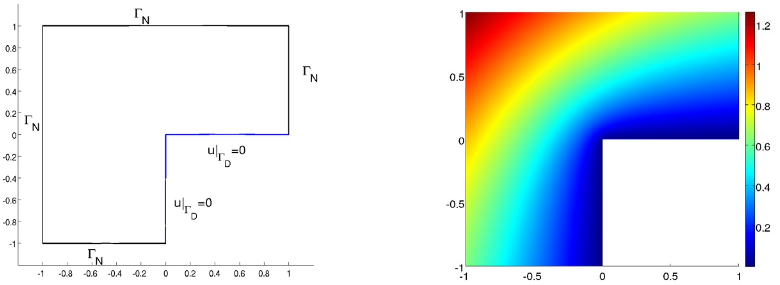
(f) Step 6, DOF 601

(g) Step 12, DOF 3072

(h) Step 12, DOF 3094

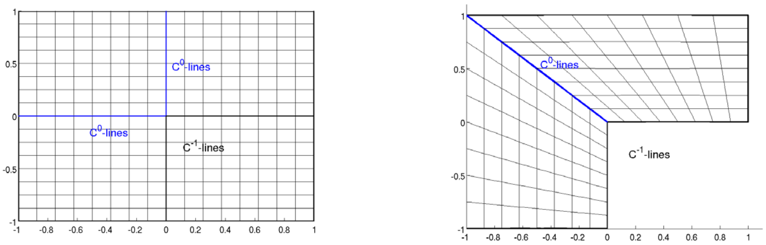
(i) Step 12, DOF 2660

Figure 37: **Internal layer problem**: Comparison of absolute value of the deviation effectivity index at element level $|1-\theta^*_{el}|$ at LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for Internal layer problem. The columns from left to right represents the cases with respect to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.



(a) L-shaped problem description (b) Exact solution u

Figure 38: **L-shaped problem**: Problem description and Exact solution u.



(a) First parametrization (b) Second parametrization

Figure 39: Parametrizations for L-shaped domain problem with coarse initial meshes.

Example 8. (L-shaped domain problem)

The governing equation of L-shaped domain problem is

$$\Delta u = 0 \quad \text{in } \Omega, \tag{145}$$

with the boundary conditions

$$u = 0 \quad \text{on} \quad \Gamma_D \quad \text{and} \quad \frac{\partial u}{\partial n} = g \quad \text{on} \quad \Gamma_N, \tag{146}$$

Here $\Omega = (-1,1)^2 \setminus (0,1) \times (-1,0)$ is a L-shape domain and g is constructed to correspond to the exact solution

$$u(x,y) = r^{\frac{2}{3}} \sin\left(\frac{2\theta}{3}\right), \quad \text{with} \quad r = (x^2+y^2)^{\frac{1}{2}}, \quad \theta = \tan^{-1}\left(\frac{y}{x}\right). \tag{147}$$

The set up of the problem with given boundary conditions and the exact solution u are shown in Figure 38.

For the given elliptic problem, re-entrant corner at $(0, 0)$ in the domain cause a singularity in the solution. It is known that the convergence for uniform mesh refinement is limited by the strength of the singularity, i.e., the convergence rate (versus degrees of freedoms) is equal to $-1/3$. For problems where the solution is not sufficiently smooth, $u \notin H^{p+1}(\Omega)$, as is the case for the L-shaped domain problem that has a singularity point at its boundary, we do not obtain optimal convergence rate when we do uniform mesh refinement. In particular, the use of high order polynomials is then inefficient.

We consider two parametrizations to solve the L-shaped domain problem as shown in Figure 39. The first parametrization represent an identity mapping $\mathbf{F} = Id$ and the geometry is constructed by inserting the additional C^0 and C^{-1}-continuity lines along x-axis and y-axis of the domain and by chopping off the lower region. This is a unique feature with LR B-splines based on multiple mesh lines insertion shown by dark black and blue lines of C^{-1} and C^0 continuity, respectively, in Figure 39(a). Our second parametrization is based on bilinear mapping $\mathbf{F}$ using quadratic LR B-splines and the coarse geometry is shown in Figure 39(b). The lighter black lines in both parametrizations represent the C^1-continuity lines for quadratic LR B-splines space.

Using the first parametrization, shown in Figure 39(a), the error plots and effectivity index of different recovery procedures using respective recovery based error estimators are shown Figure 40. The results with respect to all recovery procedures shows the superconvergence in projected error. This problem is known for the singularity at re-entrant corner $(0, 0)$ and all the recovery procedure takes few step of initial refinement alogirthm to properly handle the problem caused by the pollution effect. It can be noticed from the effectivity index plots given in Figure 40 that after few steps of the refinement algorithm the projected error becomes more accurate than the corresponding energy error of the finite element solution, and the effectivity index starts converging to 1. We also notice that the effectivity index of the SPR recovery is very close to 1 and converging to 1 in h-asymptotic exactness sense better than global recovery methods. We observe from Figure42 that not only gives the three different recovery procedures good global effectivity indices but also the local effectivity index at element level is fairly close to one. However, along the boundary the results for SPR are again less accurate then the other two methods. In any case all the recovery procedure capture very well the singularity in the solution at the re-entrant corner and the numerical solution based on adaptive refinement using these error estimates all attain optimal rate of convergence. The corresponding adaptive meshes obtained at different refinement steps are shown in Figure 41.

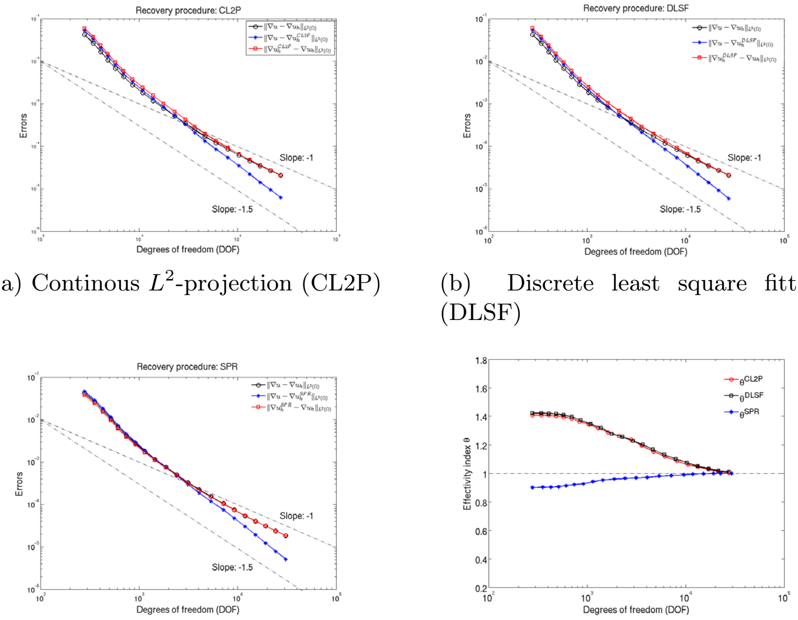
Now we consider the second parametrization, shown in Figure 39(b), to solve the L-shaped domain problem. The error plots and effectivity index of different recovery procedures using respective recovery based error estimators are shown Figure 43. The results with respect to all the recovery procedures shows the superconvergence behavior in projected error on adapted generated meshes. This parametrization seems to be more suited for the global recovery methods than the previous one. The global recovery procedures produce a more accurate recovered field than the SPR recovery at few initial refinement steps. After few step of initial refinement the projected error shows superconvergence behavior and at the end of the refinement process the SPR recovery procedure have a slightly higher convergence rate than the two other ones. The effectivity index plots given in Figure 43(d) shows that the effectivity index value is converging to 1 for the SPR recovery procedure. For the global recovery procedures the effectivity index value is first converging to 1 from above and then attain a value close to but below 1.

Regarding the ability to capture the error distribution, i.e. the behavior of the local element effectivity indices the results (see Figure 45) are the same as for the other parametrization described above. Similarly, we again achieve optimal convergence rates in the computed finite element solution, see Figure 44.

Remark 7.1. *We would like to underline the very high quality of the obtained effectivity indices obtained herein for SPR compared to earlier experience with SPR and unstructured adaptive mesh refinement of linear finite elements, see [44]. Figure 17b in [44] shows an effectivity index for the SPR-procedure in the range of* $(0.65 - 0.8)$ *for* $DOF < 1000$, *whereas we herein achieve effectivity indices in the range* $(0.90 - 1.00)$ *and* $(0.95 - 1.00)$ *for the first and second parametrization, correspondingly.*

8 Conclusion and perspectives

Isogeometric analysis (IGA) based on B-splines or NURBS are structured tensor product meshes within each patch [36] and facilitates superconvergence behavior. The recently developed LR B-splines [28] and structured adaptive mesh refinement using LR B-splines [39] are consider to be promising candidates to facilitate adaptive superconvergent gradient recovery as they produce local tensor product meshes.



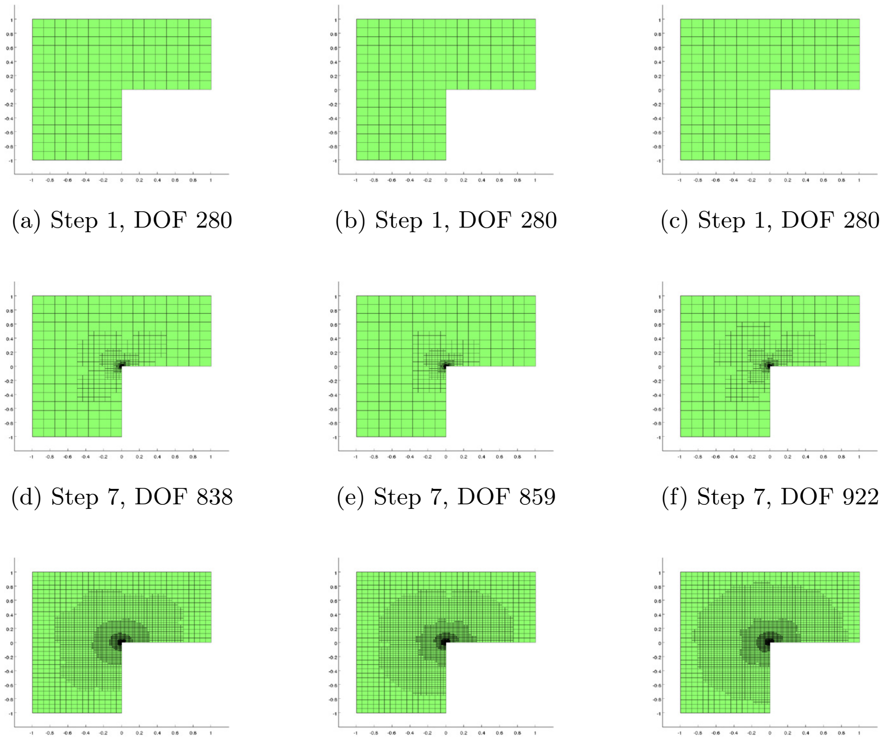
(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 40: **L-shaped domain problem on first parametrization**: Error plots and effectivity index for different recovery based a posteriori error estimators in adaptive isogeometric analysis using quadratic LR B-splines.



(a) Step 1, DOF 280

(b) Step 1, DOF 280

(c) Step 1, DOF 280

(d) Step 7, DOF 838

(e) Step 7, DOF 859

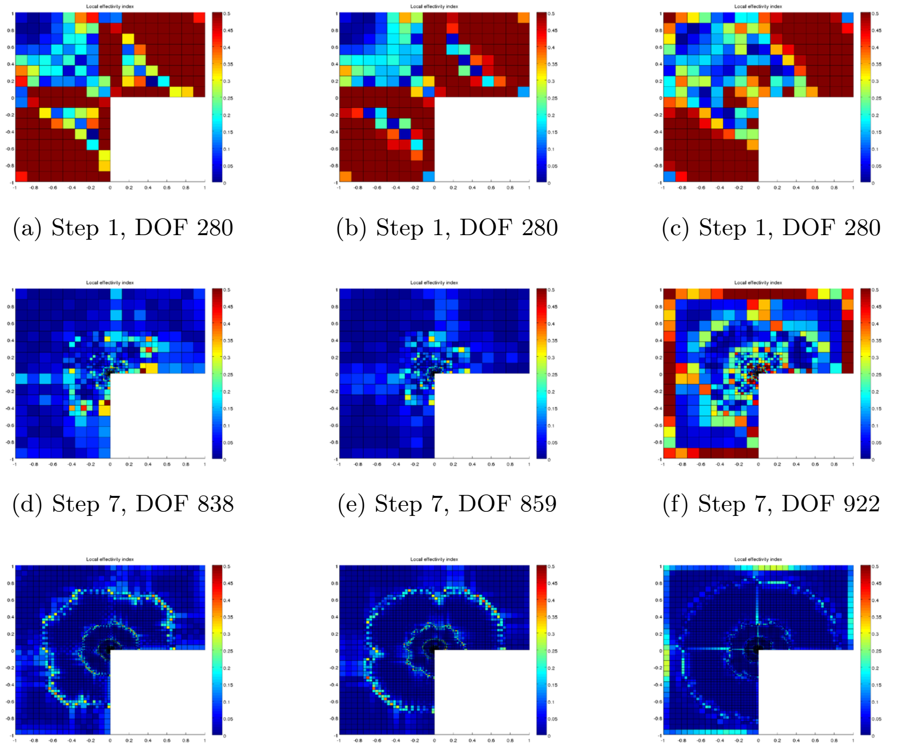
(f) Step 7, DOF 922

(g) Step 14, DOF 4743

(h) Step 14, DOF 4653

(i) Step 14, DOF 5364

Figure 41: **L-shaped domain problem first parametrization**: LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for L-shaped domain problem. The columns from left to right represents the cases with respect to to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.



(a) Step 1, DOF 280

(b) Step 1, DOF 280

(c) Step 1, DOF 280

(d) Step 7, DOF 838

(e) Step 7, DOF 859

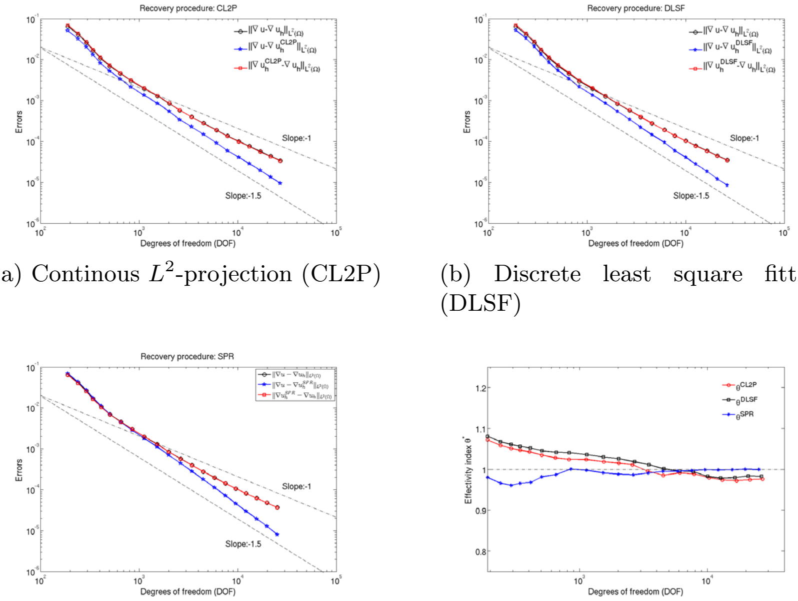
(f) Step 7, DOF 922

(g) Step 14, DOF 4743

(h) Step 14, DOF 4653

(i) Step 14, DOF 5364

Figure 42: **L-shaped domain problem on first parametrization**: Comparison of absolute value of the deviation effectivity index at element level $|1 - \theta^*_{el}|$ at LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for L-shaped domain problem. The columns from left to right represents the cases with respect to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.



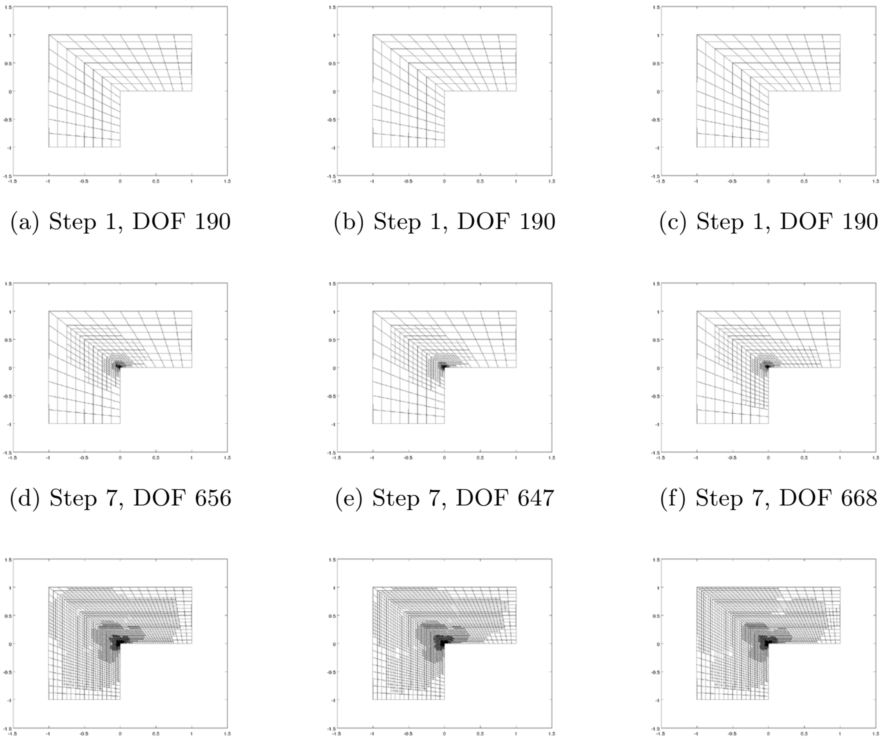
(a) Continous L^2-projection (CL2P)

(b) Discrete least square fitting (DLSF)

(c) Superconvergent patch recovery (SPR)

(d) Effectivity index θ^*

Figure 43: **L-shaped domain problem on second parametrization**: Error plots and effectivity index for different recovery based a posteriori error estimators in adaptive isogeometric analysis using quadratic LR B-splines.



(a) Step 1, DOF 190

(b) Step 1, DOF 190

(c) Step 1, DOF 190

(d) Step 7, DOF 656

(e) Step 7, DOF 647

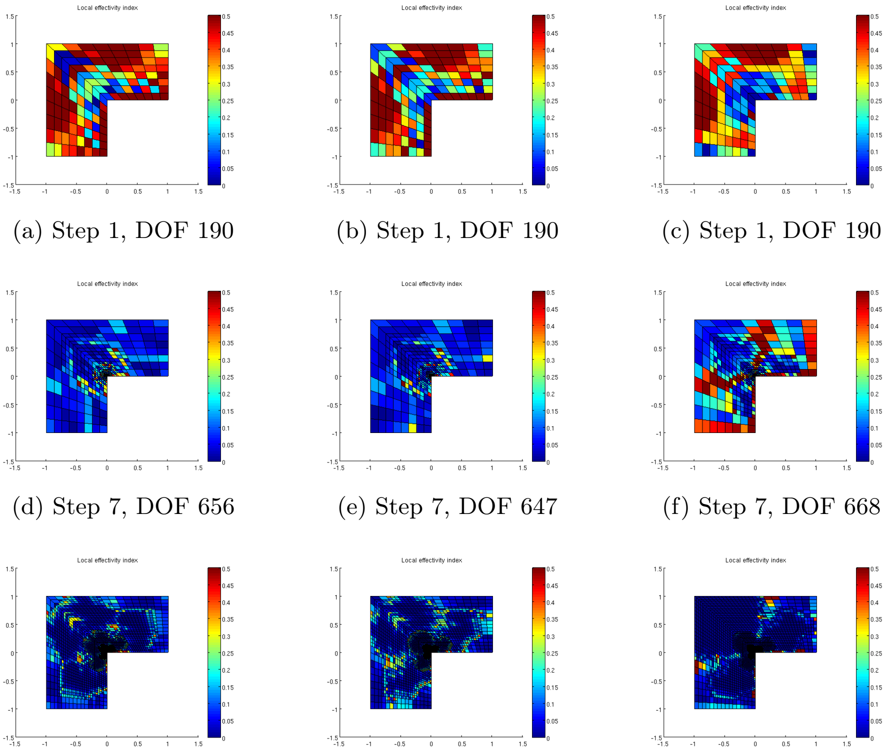
(f) Step 7, DOF 668

(g) Step 14, DOF 4563

(h) Step 14, DOF 4509

(i) Step 14, DOF 4357

Figure 44: **L-shaped domain problem on second parametrization**: LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for L-shaped domain problem. The columns from left to right represents the cases with respect to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.



(a) Step 1, DOF 190

(b) Step 1, DOF 190

(c) Step 1, DOF 190

(d) Step 7, DOF 656

(e) Step 7, DOF 647

(f) Step 7, DOF 668

(g) Step 14, DOF 4563

(h) Step 14, DOF 4509

(i) Step 14, DOF 4357

Figure 45: **L-shaped domain problem on second parametrization**: Comparison of absolute value of the deviation effectivity index at element level $|1 - \theta^*_{el}|$ at LR meshes obtained via adaptive LR B-splines refinement algorithm using different recovery based error estimators at different refinement steps for L-shaped domain problem. The columns from left to right represents the cases with respect to Continous L^2-projection (CL2P), Discrete least square fitting (DLSF) and Superconvergent patch recovery (SPR), respectively.

We start out by addressing the existence of derivative superconvergence points in the computed finite element solution based on B-splines and LR B-splines for our elliptic model problem (1D and 2D Poisson). Inspired by earlier theoretical work presented in Wahlbin [69] and computer based proof of Babuska [8, 12, 11], we demonstrate that we are able to compute true derivative superconvergence points by means of local Neumann projection of a set of proper monomials for both uniform B-splines as well as for general (non-uniform) adapted LR B-splines. For uniform B-splines the true derivative superconvergent points are located at different location than the case of classical C^0 Lagrange elements. Thus, the continuity of the underlying finite element basis plays an important role for the location of true derivative superconvergent points. For the case of quadratic C^1 B-splines, on uniform mesh partition, they share the same location given by the (2×2)-Gauss Legendre points (or Barlow points) for classical quadratic C^0 Lagrange elements. While in case of cubic C^2 and C^1 B-spline spaces, on uniform mesh partition, the derivative superconvergence points will be at (3×3)-Gauss Lobatto points within each elements in contrary to the (3×3)-Gauss Legendre points (or Barlow points) for classical cubic C^0 Lagrange elements.

The main part of the article focus on a study of three different gradient recovery techniques for the purpose of enabling effective adaptive refinement in isogeometric analysis: Continuous L^2-projection (CL2P), Discrete least square fitting (DLSF), and Superconvergent Patch Recovery (SPR).

The main findings are:

- The difference between using true superconvergent derivative points and (2×2)-Gauss Legendre points for p=2 is noticeable but not pronounced for the accuracy of the recovered gradient field and the corresponding global effectivity indices.

- Adaptive refinement using all the three recovery based a posteriori error estimates provides optimal convergence rate

- The obtained global effectivity indices are for all the three recovery techniques remarkable close to one - This is in contrast with residual based error estimates.

- The local elementwise effectivity indices for all the three recovery techniques are close to one after some initial refinement steps to take care of any possible pollution effect.

- The main difference between the three recovery methods is that for both CL2P and DLSF one have to solve a global (mass matrix) problem, whereas SPR only involve solution of a local problem.

The performance of global CL2P and DLSF recovery procedures presented in this article are very comparable in comparison to the local SPR recovery procedure. Thus it can be worthy to developed a local version of these recovery procedures and some work in this direction to design a local projection has been recently presented in [34], [60]. It is worth to explore this possibility based on Bezier extraction technique in adaptive isogeometric analysis.

The aim of the present study has been to develop adaptive recovery procedures that produce global (and local) effectivity indices close to one and being h-asymptotic exact. A natural extension will be to provide guaranteed upper and lower bounds and that will be pursued in an upcoming article.

Acknowledgements

The authors gratefully acknowledge the the financial support from the Norwegian Research Council and the industrial partners of the ICADA project (Ceetron, DNV GL and Statoil) and the FME NOWITECH. They also acknowledge the support from the other co-workers in the IFEM software team.

Bibliography

[1] M. Ainsworth and A. Craig. “A posteriori error estimators in the finite element method”. In: *Numer. Math.* 60.4 (1992), pp. 429–463.

[2] M. Ainsworth and J. T. Oden. “A posteriori error estimation in finite element analysis”. In: *Comput. Methods Appl. Mech. Engrg.* 142.1-2 (1997), pp. 1–88.

[3] M. Ainsworth and J. T. Oden. *A posteriori error estimation in finite element analysis*. Pure and Applied Mathematics (New York). Wiley-Interscience [John Wiley & Sons], New York, 2000, pp. xx+240.

[4] A. B. Andreev and R. D. Lazarov. “Superconvergence of the gradient for quadratic triangular finite elements”. In: *Numer. Methods Partial Differential Equations* 4.1 (1988), pp. 15–32.

[5] I. Babuška and W. C. Rheinboldt. “A posteriori error analysis of finite element solutions for one-dimensional problems”. In: *SIAM J. Numer. Anal.* 18.3 (1981), pp. 565–589.

[6] I. Babuška and W. C. Rheinboldt. “A posteriori error estimates for the finite element method”. In: *Internat. J. Numer. Methods Eng.* 12 (1978), pp. 1597–1615.

[7] I. Babuška and W. C. Rheinboldt. “Error estimates for adaptive finite element computations”. In: *SIAM J. Numer. Anal.* 15.4 (1978), pp. 736–754.

[8] I. Babuška and T. Strouboulis. *The finite element method and its reliability*. Numerical Mathematics and Scientific Computation. The Clarendon Press, Oxford University Press, New York, 2001, pp. xii+802.

[9] I. Babuška, T. Strouboulis, and C. S. Upadhyay. “A model study of the quality of a posteriori error estimators for finite element solutions of linear elliptic problems, with particular reference to the behavior near the boundary”. In: *Internat. J. Numer. Methods Engrg.* 40.14 (1997), pp. 2521–2577.

[10] I. Babuška, T. Strouboulis, and C. S. Upadhyay. “A model study of the quality of a posteriori error estimators for linear elliptic problems. Error estimation in the interior of patchwise uniform grids of triangles”. In: *Comput. Methods Appl. Mech. Engrg.* 114.3-4 (1994), pp. 307–378.

[11] I. Babuška et al. “Computer-based proof of the existence of superconvergence points in the finite element method; superconvergence of the derivatives in finite element solutions of Laplace’s, Poisson’s, and the elasticity equations”. In: *Numer. Methods Partial Differential Equations* 12.3 (1996), pp. 347–392.

[12] I. Babuška et al. “η%-superconvergence in the interior of locally refined meshes of quadrilaterals: superconvergence of the gradient in finite element solutions of Laplace’s and Poisson’s equations”. In: *Appl. Numer. Math.* 16.1-2 (1994). A Festschrift to honor Professor Robert Vichnevetsky on his 65th birthday, pp. 3–49.

[13] I. Babuška et al. “Validation of a posteriori error estimators by numerical approach”. In: *Internat. J. Numer. Methods Engrg.* 37.7 (1994), pp. 1073–1123.

[14] R. E. Bank and R. K. Smith. “A posteriori error estimates based on hierarchical bases”. In: *SIAM J. Numer. Anal.* 30.4 (1993), pp. 921–935.

[15] Y. Bazilevs et al. “Isogeometric analysis: approximation, stability and error estimates for h-refined meshes”. In: *Math. Models Methods Appl. Sci.* 16.7 (2006), pp. 1031–1090.

[16] Y. Bazilevs et al. “Isogeometric analysis: approximation, stability and error estimates for h-refined meshes”. In: *Math. Models Methods Appl. Sci.* 16.7 (2006), pp. 1031–1090.

[17] Y. Bazilevs et al. “Isogeometric analysis using T-splines”. In: *Comput. Methods Appl. Mech. Engrg.* 199.5-8 (2010), pp. 229–263.

[18] T. Blacker and T. Belytschko. “Superconvergent patch recovery with equilibrium and conjoint interpolant enhancements”. In: *Internat. J. Numer. Methods Eng.* 37 (1994), pp. 517–536.

[19] S. Bordas and M. Duflot. “Derivative recovery and a posteriori error estimate for extended finite elements”. In: *Comput. Methods Appl. Mech. Engrg.* 196.35-36 (2007), pp. 3381–3399.

[20] S. Bordas, M. Duflot, and P. Le. “A simple error estimator for extended finite elements”. In: *Comm. Numer. Methods Engrg.* 24.11 (2008), pp. 961–971.

[21] A. Bressan. “Some properties of LR-splines”. In: *Comput. Aided Geom. Design* 30.8 (2013), pp. 778–794.

[22] A. Buffa, D. Cho, and M. Kumar. “Characterization of T-splines with reduced continuity order on T-meshes”. In: *Comput. Methods Appl. Mech. Engrg.* 201/204 (2012), pp. 112–126.

[23] C. M. Chen. *Structure Theory of Superconvergence of Finite Elements (in Chinese).* Hunan Science and Technology Publishing House, Changsha, 2001.

[24] C. M. Chen and Y. Q. Huang. *High Accuracy Theory of Finite Element Methods (in Chinese).* Hunan Science and Technology Publishing House, Changsha, 1995.

[25] J. A. Cottrell, T. J. R. Hughes, and Y. Bazilevs. *Isogeometric Analysis. Toward Integration of CAD and FEA.* Pure and Applied Mathematics (New York). John Wiley & Sons, 2009.

[26] P. Díez, N. Parés, and A. Huerta. “Recovering lower bounds of the error by postprocessing implicit residual a posteriori error estimates”. In: *Internat. J. Numer. Methods Engrg.* 56.10 (2003), pp. 1465–1488.

[27] P. Díez, J. J. Ródenas, and O. C. Zienkiewicz. “Equilibrated patch recovery error estimates: simple and accurate upper bounds of the error”. In: *Internat. J. Numer. Methods Engrg.* 69.10 (2007), pp. 2075–2098.

[28] T. Dokken, T. Lyche, and K. F. Pettersen. “Polynomial splines over locally refined box-partitions”. In: *Comput. Aided Geom. Design* 30.3 (2013), pp. 331–356.

[29] M. R. Dörfel, B. Jüttler, and B. Simeon. “Adaptive isogeometric analysis by local h-refinement with T-splines”. In: *Comput. Methods Appl. Mech. Engrg.* 199.5-8 (2010), pp. 264–275.

[30] J. J. Douglas and T. Dupont. “Some superconvergence results for Galerkin methods for the approximate solution of two-point boundary problems”. In: *Topics in numerical analysis (Proc. Roy. Irish Acad. Conf., University Coll., Dublin, 1972).* Academic Press, London, 1973, pp. 89–92.

[31] J. J. Douglas, T. Dupont, and M. F. Wheeler. “An L^∞ estimate and a superconvergence result for a Galerkin method for elliptic equations based on tensor products of piecewise polynomials”. In: *Rev. Française Automat. Informat. Recherche Opérationnelle Sér Rouge* 8.R-2 (1974), pp. 61–66.

[32] C. Giannelli, B. Jüttler, and H. Speleers. “THB-splines: the truncated basis for hierarchical splines”. In: *Comput. Aided Geom. Design* 29.7 (2012), pp. 485–498.

[33] O. A. González-Estrada et al. "Efficient recovery-based error estimation for the smoothed finite element method for smooth and singular linear elasticity". In: *Comput. Mech.* 52.1 (2013), pp. 37–52.

[34] S. Govindjee et al. "Convergence of an efficient local least-squares fitting method for bases with compact support". In: *Comput. Methods Appl. Mech. Engrg.* 213/216 (2012), pp. 84–92.

[35] E. Hinton and J. S. Campbell. "Local and global smoothing of discontinuous finite element functions using a least squares method". In: *Internat. J. Numer. Methods Eng.* 8 (1974), pp. 461–480.

[36] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs. "Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement". In: *Comput. Methods Appl. Mech. Engrg.* 194.39-41 (2005), pp. 4135–4195.

[37] K. A. Johannessen. *An adaptive isogeometric finite element analysis.* Master thesis, Norwegian University of Science and Technology, Norway. 2009.

[38] K. A. Johannessen, M. Kumar, and T. Kvamsdal. "Isogeometric divergence-conforming discretization for Stokes problem on locally refined meshes using LR B-splines". In: *Comput. Methods Appl. Mech. Engrg.* To be submitted (2014).

[39] K. A. Johannessen, T. Kvamsdal, and T. Dokken. "Isogeometric analysis using LR B-splines". In: *Comput. Methods Appl. Mech. Engrg.* 269 (2014), pp. 471–514.

[40] S. K. Kleiss and S. Tomar. "Guaranteed and sharp a posteriori error estimates in isogeometric analysis". In: *Preprint, arXiv:1304.7712* (2013).

[41] R. Kraft. *Adaptive and Linearly Independent Multilevel B-splines.* Bericht. SFB 404, Geschäftsstelle, 1997.

[42] M. Křížek. "Superconvergence phenomena on three-dimensional meshes". In: *Int. J. Numer. Anal. Model.* 2.1 (2005), pp. 43–56.

[43] M. Křížek and P. Neittaanmäki. "Bibliography on superconvergence". In: *Finite element methods (Jyväskylä, 1997).* Vol. 196. Lecture Notes in Pure and Appl. Math. Dekker, New York, 1998, pp. 315–348.

[44] T. Kvamsdal and K. M. Okstad. "Error estimation based on superconvergent patch recovery using statically admissible stress fields". In: *Internat. J. Numer. Methods Engrg.* 42.3 (1998), pp. 443–472.

[45] P. Ladevèze and D. Leguillon. “Error estimate procedure in the finite element method and applications”. In: *SIAM J. Numer. Anal.* 20.3 (1983), pp. 485–509.

[46] P. Lesaint and M. Zlámal. “Superconvergence of the gradient of finite element solutions”. In: *RAIRO Anal. Numér.* 13.2 (1979), pp. 139–166.

[47] J. T. Oden and H. J. Brauchli. “On the calculation of consistent stress distributions in finite element approximations”. In: *Internat. J. Numer. Methods Eng.* 3 (1971), pp. 317–325.

[48] L. A. Oganesjan and L. A. Ruhovec. “An investigation of the rate of convergence of variation-difference schemes for second order elliptic equations in a two-dimensional region with smooth boundary”. In: *Ž. Vyčisl. Mat. i Mat. Fiz.* 9 (1969), pp. 1102–1120.

[49] K. Okstad, T. Kvamsdal, and K. Mathisen. “Superconvergent patch recovery for plate problems using statically admissible stress resultant fields”. In: *International journal for numerical methods in engineering* 44.5 (1999), pp. 697–727.

[50] N. Parés, H. Santos, and P. Díez. “Guaranteed energy error bounds for the Poisson equation using a flux-free approach: solving the local problems in subdomains”. In: *Internat. J. Numer. Methods Engrg.* 79.10 (2009), pp. 1203–1244.

[51] S. I. Repin. “A posteriori error estimates for approximate solutions to variational problems with strongly convex functionals”. In: *J. Math. Sci. (New York)* 97.4 (1999). Problems of mathematical physics and function theory, pp. 4311–4328.

[52] S. I. Repin. “A posteriori error estimation for variational problems with uniformly convex functionals”. In: *Math. Comp.* 69.230 (2000), pp. 481–500.

[53] A. H. Schatz. “Pointwise error estimates, superconvergence and extrapolation”. In: *Finite element methods (Jyväskylä, 1997)*. Vol. 196. Lecture Notes in Pure and Appl. Math. Dekker, New York, 1998, pp. 237–247.

[54] A. H. Schatz, I. H. Sloan, and L. B. Wahlbin. “Superconvergence in finite element methods and meshes that are locally symmetric with respect to a point”. In: *SIAM J. Numer. Anal.* 33.2 (1996), pp. 505–521.

[55] A. H. Schatz and L. B. Wahlbin. “Interior maximum-norm estimates for finite element methods. II”. In: *Math. Comp.* 64.211 (1995), pp. 907–928.

[56] L. L. Schumaker. *Spline functions: basic theory.* Third. Cambridge Mathematical Library. Cambridge University Press, Cambridge, 2007, pp. xvi+582.

[57] M. A. Scott, D. C. Thomas, and E. J. Evans. "Isogeometric spline forests". In: *Comput. Methods Appl. Mech. Engrg.* 269 (2014), pp. 222–264.

[58] M. A. Scott et al. "Local refinement of analysis-suitable T-splines". In: *Comput. Methods Appl. Mech. Engrg.* 213/216 (2012), pp. 206–222.

[59] T. W. Sederberg et al. "T-splines and T-NURCCs". In: *ACM Trans. Graph.* 22.3 (July 2003), pp. 477–484.

[60] D. C. Thomas et al. "Bé zier projection: a unified approach for local projection and quadrature-free refinement and coarsening of NURBS and T-splines with particular application to isogeometric design and analysis". In: *Preprint arXiv:1404.7155* (2014).

[61] V. Thomée. "High order local approximations to derivatives in the finite element method". In: *Math. Comp.* 31.139 (1977), pp. 652–660.

[62] V. Thomée. "Negative norm estimates and superconvergence in Galerkin methods for parabolic problems". In: *Math. Comp.* 34.149 (1980), pp. 93–113.

[63] L. Tian, F. Chen, and Q. Du. "Adaptive finite element methods for elliptic equations over hierarchical T-meshes". In: *J. Comput. Appl. Math.* 236.5 (2011), pp. 878–891.

[64] L. Beirão da Veiga, D. Cho, and G. Sangalli. "Anisotropic NURBS approximation in isogeometric analysis". In: *Comput. Methods Appl. Mech. Engrg.* 209/212 (2012), pp. 1–11.

[65] L. Beirão da Veiga et al. "Mathematical analysis of variational isogeometric methods". In: *Acta Numerica* 23 (May 2014), pp. 157–287.

[66] L. Beirão da Veiga et al. "Some estimates for *h*-*p*-*k*-refinement in isogeometric analysis". In: *Numer. Math.* 118.2 (2011), pp. 271–305.

[67] R. Verfürth. *A posteriori error estimation techniques for finite element methods.* Numerical Mathematics and Scientific Computation. Oxford University Press, Oxford, 2013, pp. xx+393.

[68] A.-V. Vuong et al. "A hierarchical approach to adaptive local refinement in isogeometric analysis". In: *Comput. Methods Appl. Mech. Engrg.* 200.49-52 (2011), pp. 3554–3567.

[69] L. B. Wahlbin. *Superconvergence in Galerkin finite element methods.* Vol. 1605. Lecture Notes in Mathematics. Springer-Verlag, Berlin, 1995, pp. xii+166.

[70] P. Wang et al. “Adaptive isogeometric analysis using rational PHT-splines”. In: *Comput.-Aided Des.* 43.11 (2011), pp. 1438–1448.

[71] G. Xu et al. “Optimal analysis-aware parameterization of computational domain in 3D isogeometric analysis”. In: *Comput.-Aided Des.* 45.4 (2013), pp. 812–821.

[72] G. Xu et al. “Parameterization of computational domain in isogeometric analysis: methods and comparison”. In: *Comput. Methods Appl. Mech. Engrg.* 200.23-24 (2011), pp. 2021–2031.

[73] Z. Zhang. “Derivative superconvergent points in finite element solutions of harmonic functions—a theoretical justification”. In: *Math. Comp.* 71.240 (2002), 1421–1430 (electronic).

[74] Z. Zhang. “Derivative superconvergent points in finite element solutions of Poisson’s equation for the serendipity and intermediate families—a theoretical justification”. In: *Math. Comp.* 67.222 (1998), pp. 541–552.

[75] C. D. Zhu and Q. Lin. *The Superconvergence Theory of the Finite Element Method (in Chinese).* Hunan Science and Technology Publishing House, Changsha, 1989, pp. iv+314.

[76] O. C. Zienkiewicz and J. Z. Zhu. “A simple error estimator and adaptive procedure for practical engineering analysis”. In: *Internat. J. Numer. Methods Engrg.* 24.2 (1987), pp. 337–357.

[77] O. C. Zienkiewicz and J. Z. Zhu. “The superconvergent patch recovery and a posteriori error estimates. I. The recovery technique”. In: *Internat. J. Numer. Methods Engrg.* 33.7 (1992), pp. 1331–1364.

[78] O. C. Zienkiewicz and J. Z. Zhu. “The superconvergent patch recovery and a posteriori error estimates. II. Error estimates and adaptivity”. In: *Internat. J. Numer. Methods Engrg.* 33.7 (1992), pp. 1365–1382.

Divergence-conforming discretization for Stokes problem on locally refined meshes using LR B-splines

Kjetil André Johannessen, Mukesh Kumar and Trond Kvamsdal

Paper IV

Submitted to *Computer Methods in Applied Mechanics and Engineering*, 2014

Divergence-conforming discretization for Stokes problem on locally refined meshes using LR B-splines

Kjetil André Johannessen, Mukesh Kumar and Trond Kvamsdal
Department of Mathematical Sciences
Norwegian University of Science and Technology, Trondheim, Norway.
E-mail: Kjetil.Johannessen@math.ntnu.no, Mukesh.Kumar@math.ntnu.no and
Trond.Kvamsdal@math.ntnu.no

Abstract

To solve the incompressible flow problems using isogeometric analysis, the div-compatible spline spaces was originally introduced by Buffa [8, 11], and later developed by Evans [18]. In this paper, we construct div-compatible spline spaces with local refinement capability using Locally Refined (LR) B-splines. We argue that the splines spaces generated on locally refined meshes will satisfy compatibility provided they span the entire function spaces as governed by Mourrain [27] dimension formula. We show that the locally *structured* refined LR B-splines, introduced by Johannessen et.al [23], fulfills the necessary requirements for being div-compatible. Further, we consider these div-compatible LR B-spline spaces to approximate the velocity and pressure fields in mixed discretization for Stokes problem and a set of standard benchmark tests are performed to show the stability, efficiency and the conservation properties of the discrete velocity fields in adaptive isogeometric analysis.

1 Introduction

Isogeometric analysis (IGA) was introduced in [21] as an innovative numerical methodology for the discretization of Partial Differential Equations (PDEs), the main idea was to improve the interoperability between Computer Aided Design (CAD) and PDE solvers, and to achieve this, the authors in [21] proposed to use CAD mathematical primitives, i.e. splines and NURBS, to also represent PDE unknowns. The smoothness of splines is a new ingredient that yields several advantages: for example, it improves the accuracy per degree of freedom and allows for the direct approximation of higher order PDEs. Isogeometric methods have been used and tested on a variety of problems of engineering interests, for flow simulations [1, 4, 5, 8, 12, 18, 15, 16], and for electromagnetic problems [9, 10, 30].

In electromagnetic and incompressible fluids flow simulations, numerical discretization have to preserve the geometric structure of underlying PDEs in order to avoid spurious behaviors, instability or non-physical solution. Thus the numerical discretizations have to be related with a discrete De Rham complex. Compatible spaces for finite element approximations were in general introduced by Arnold *et al.* [2], and more recently in isogeometric analysis context, by Buffa *et al.* [8, 11]. High regularity of splines is advantageous for constructing compatible spaces. Some initial work to show the potential impact of compatible spline based methodology for electromagnetic wave computations was presented in [9, 30] and recently using T-spline complexes in [10]. For incompresible fluid flows, Buffa et al. [8] proposed the first isogeometric Stokes solver based on three choices of discrete spline spaces to approximate the mixed discretization for Stokes problem, which were seen as a smooth extension of Taylor-Hood (TH), Nedelec (N) and Raviart-Thomas (RT) pair of Finite Element (FE) spaces. One of their main finding was the smooth Raviart-Thomas (RT) pair of spline based FE spaces provides divergence-free discrete solutions. Later using the idea of div-compatible spline spaces presented in the setting of discrete differential forms [11], a series of isogeometric divergence conforming spline discretizations were derived to solve Stokes and Brinkman equations, steady and unsteady Navier-Stokes equations in by Evans in [15], [16], and [17]. These initial developments show that isogeometric analysis is a highly accurate and efficient methodology to solve incompressible flow problems. In this paper our aim is to develop div-comptaible spaces on locally refined meshes and explore the benefit of adaptive refinement in solving incompressible flow problem in term of efficiency. The adaptivity (or local refinement) is required, if one wants to capture the strong singularities, information about recirculation eddy in fluids and achieve an optimal rate of convergence.

Non-uniform rational B-splines (NURBS) are the dominant geometric representation format for CAD. The construction of NURBS are based on a tensor product structure and, as a consequence, knot insertion is a global operation. To remedy this a local refinement can be achieved by breaking the global tensor product structure of multivariate splines and NURBS. Several techniques have been proposed to address this, among others are T-splines [34],[33], Hierarchical B-splines [19],[24], Truncated Hierarchical B-splines [20] and Locally Refined (LR) B-splines [13]. While initially, most of the references address the problem from a CAD point of view, later years have seen them applied to isogeometric analysis. For T-splines consider [3, 14, 32, 36, 35], for Hierarchical B-splines consider [28, 37, 6, 31], and for LR-splines see [23, 25].

1.1 Aim and outline of the paper

The aim of this article is to present a class of compatible spline spaces with local refinement capability which form a De Rham complex and provide a stable, divergence-free discretization of the 2D Stokes problem.

The paper is organized as follows:

We start the Section 2 with our model Stokes problem which can be seen as a prototype of viscous incompressible flows. The necessary conditions to derive a divergence conforming spline discretization and the main results of the paper are presented.

In Section 3, we introduce the basic concepts of splines over locally refined Box-meshes (or T-meshes). We present the dimension formula as given by Mourrain [27] which will be useful in proving the compatibility among the splines spaces on locally refined meshes. Further we discuss the construction of derivatives spaces on locally refined meshes.

In Section 4, we present three different complete De Rham complexes on locally refined meshes. The complexes are characterized by their boundary condition for the velocity space: (*i*) without boundary condition, (*ii*) with no penetration boundary condition, and (*iii*) with no slip boundary condition.

To build a basis on the Box meshes we introduce the locally refined (LR) B-splines in Section 5. We give a brief overview of their generality before defining a subclass which we will use for the local refinement. This is the structured mesh refinement as introduced in [23].

In Section 6, we present some numerical results for Stokes problems. The numerical stability, convergence rates, efficiency and conservation properties of the proposed LR B-spline discretization in adaptive isogeometric analysis will be main focus.

Finally, some conclusions and perspectives are included in Section 7.

2 Stokes problem and Divergence-conforming spline discretization

To model a viscous incompressible flow, we consider the following Stokes problem: find $(\mathbf{u}, p)$ such that

$$\left\{ \begin{array}{rcl} -\mu\nabla^2\mathbf{u} + \nabla p & = & \mathbf{f} \;\text{ in }\; \Omega \\ \nabla\cdot\mathbf{u} & = & 0 \;\text{ in }\; \Omega \end{array} \right. \tag{1}$$

with suitable boundary conditions and μ is the constant viscosity term. We consider $\mathbf{f} \in (L^2(\Omega))^2$ and $\Omega \subset \mathbb{R}^2$ is a physical domain under a geometrical mapping $\mathbf{F}$ on parametric domain $\hat{\Omega} = (0,1)^2$, where the mapping is assumed to be piecewise smooth and has piecewise smooth inverse.

We considered the mixed variational form of (1); find $\mathbf{u} \in \mathbf{H}^1(\Omega)$ and $p \in L^2(\Omega)$ such that

$$\left\{ \begin{array}{rcl} a(\mathbf{u},\mathbf{v}) + b(\mathbf{v},p) & = & f(\mathbf{v}) \;\; \forall \mathbf{v} \in \mathbf{H}^1(\Omega) \\ b(q,\mathbf{u}) & = & 0 \;\; \forall q \in L^2(\Omega) \end{array} \right. \tag{2}$$

where

$$a(\mathbf{u},\mathbf{v}) = \int_\Omega \mu\nabla\mathbf{u} : \nabla\mathbf{v}$$

$$b(\mathbf{v},q) = -\int_\Omega q \;\text{div}\;\mathbf{v} \quad \text{and} \quad f(\mathbf{v}) = \int_\Omega \mathbf{f}\cdot\mathbf{v}.$$

The discrete mixed form of (2), in which an approximation of $(\mathbf{u}_h, p_h)$ to the exact solution $(\mathbf{u}, p)$ of (1) is obtain by solving the problem: find $\mathbf{u}_h \in V_h \subset \mathbf{H}^1(\Omega)$ and $p_h \in Q_h \subset L^2(\Omega)$ such that

$$\left\{ \begin{array}{rcl} a(\mathbf{u}_h,\mathbf{v}_h) + b(\mathbf{v}_h,p_h) & = & f(\mathbf{v}_h) \;\; \forall \mathbf{v}_h \in V_h \\ b(q_h,\mathbf{u}_h) & = & 0 \;\; \forall q_h \in Q_h. \end{array} \right. \tag{3}$$

In order to guarantee the stability, we consider the choices of discrete pair of approximation spaces $\{V_h, Q_h\}$ which satisfy the *inf-sup stability* conditions, i.e.,

$$\inf_{\substack{q_h\in Q_h\\ q_h\neq 0}} \sup_{\substack{\mathbf{v}_h\in V_h\\ \mathbf{v}_h\neq 0}} \frac{\int_\Omega q_h \text{div}\;\mathbf{v}_h}{\|q_h\|^2_{L^2(\Omega)}\|\mathbf{v}_h\|_{H^1(\Omega)}} \geq c_{is} > 0 \tag{4}$$

where c_{is} is the inf-sup constant independent of h.

The discrete velocity approximation $\mathbf{u}_h$ of problem (1) is *in general* not exactly divergence-free, i.e., div $\mathbf{u}_h \neq 0$. A sufficient conditions that guarantees divergence-free velocities is

$$\{\text{div } \mathbf{v} : \mathbf{v} \in V_h\} \subseteq Q_h, \tag{5}$$

which will conflict with (4), unless the equality holds in (5).

The discretization techniques that produce an exactly divergence-free velocity fields are of great practical interests and are not easy to devise in the framework of classical finite elements. In the context of Isogeometric Analysis, a few choices of spline based discrete pair of spaces (V_h, Q_h) which satisfy (4) have been presented in Buffa et al. [9]. These choices are seen as spline generalization of well known FE spaces, namely Taylor-Hood (TH) elements, Nédélec(N) elements of the second family and Raviart-Thomas (RT) elements. Moreover, the spline generalization of Raviart-Thomas elements introduced in [8] enjoys also the property (5) and thus provides divergence-free discrete solutions.

On a parametric domain $\hat{\Omega}$ with the associated mesh $\mathcal{M}$ the choice of spline spaces (V_h, Q_h) for the Raviart-Thomas (RT) elements can be defined as

$$\hat{V}_h^{RT} = \mathcal{S}_{k+1,\ell}^{p+1,q}(\mathcal{M}) \times \mathcal{S}_{k,\ell+1}^{p,q+1}(\mathcal{M}); \qquad \hat{Q}_h^{RT} = \mathcal{S}_{k,\ell}^{p,q}(\mathcal{M}); \tag{6}$$

$$\tag{7}$$

where $\mathcal{S}_{k,\ell}^{p,q}(\mathcal{M})$ denotes the two dimensional spline space of degree (p, q) and continuity (k, ℓ) in both directions, respectively.

In the results of Theorem 3.1 of Buffa et al. [9], a characterization for the range of the div operator in the situations of interest for the Raviart-Thomas elements $\hat{V}_h \times \hat{Q}_h$ which ensure the stability and divergence free conforming solutions were presented. In this paper, we extend their characterization (results of Theorem 3.1) for the case of splines spaces with locally refinement capability. The main result is given as follows:

Theorem 1. *Let $\mathcal{S}^{p+1,q+1}(\mathcal{M})$ be a spline space over a general homology*[2] *free box mesh $\mathcal{M}$ in parametric domain $\hat{\Omega}$, then the following pairs of spaces*

[2] See discussion in Section 3

are equal

$$
\begin{aligned}
\{div(\boldsymbol{v}) : \boldsymbol{v} \in \hat{V}_h\} &= \{q \in \hat{Q}_h\}; && (8)\\
\{div(\boldsymbol{v}) : \boldsymbol{v} \in \hat{V}_h,\ \boldsymbol{v} \cdot \boldsymbol{n}|_{\partial\Omega} = 0\} &= \{q \in \hat{Q}_h : \int q = 0\}; && (9)\\
\{div(\boldsymbol{v}) : \boldsymbol{v} \in \hat{V}_h,\ \boldsymbol{v}|_{\partial\Omega} = \boldsymbol{0}\} &= \{q \in \hat{Q}_h : \int q = 0 \ \ \textit{with,} && (10)\\
&\qquad q(\hat{\boldsymbol{x}}_i) = 0, i = 1...4\};
\end{aligned}
$$

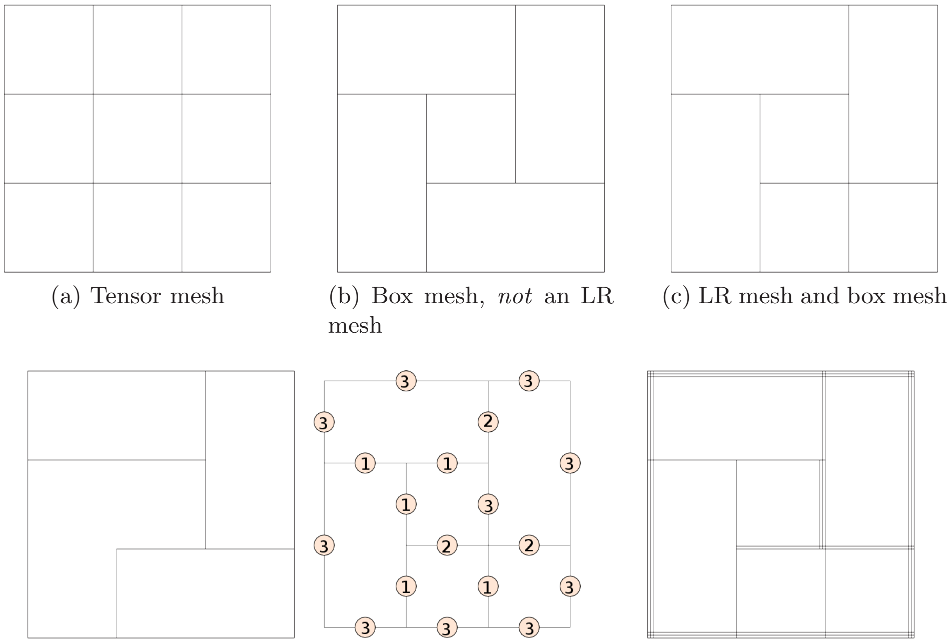
with

$$\hat{V}_h = \mathcal{S}^{p+1,q} \times \mathcal{S}^{p,q+1} \quad \textit{and} \quad \hat{Q}_h = \mathcal{S}^{p,q};$$

where $\mathcal{S}^{p,q}(\mathcal{M})$ *denotes the two dimensional spline space of degree* (p, q) *in both directions, respectively, and* $\boldsymbol{n}$ *denotes the outward unit normal to the boundary of* $\hat{\Omega}$ *and* $\hat{\boldsymbol{x}}_i, i = 1, ..., 4$ *denote its four corners.*

Proof. The proof of this theorem is given in three steps, as the result of Theorems 2-4 for the case of (8)-(10), respectively, see Section 4. □

Although the results of Theorem 1 has a general significance for spline spaces over a general box meshes, here we consider the structured LR B-spline spaces, which is a subset of LR B-splines and constructed through the structured local refinement algorithm of Johannessen et al. [23], as a suitable candidate for div-compatible splines spaces with local refinement capability.



(a) Tensor mesh

(b) Box mesh, *not* an LR mesh

(c) LR mesh and box mesh

(d) Not an LR-mesh, nor a box mesh

(e) LR mesh with multiplicities

(f) Alternative way of drawing (e)

Figure 1: Note that there is no way to create the box mesh (b) from single line insertions (starting at tensor mesh) where every intermediate state is also a box mesh. This is a prerequisite for all LR meshes.

3 Spline spaces over planar box meshes

The aim of this section is to set the notations and briefly state the definitions of several types of unstructured-meshes and present the dimension of spline spaces over them. The dimension argument was first presented by Mourrain [27] and later extended to multivariate case by Pettersen [29].

In the literature, one can classify several types of unstructured-meshes as defined below.

Definition 1. A **Box Mesh** or T-mesh is a partitioning of a two-dimensional rectangular domain $[x_0, x_n] \times [y_0, y_n]$ into smaller rectangles by horizontal and vertical lines.

Definition 2. A **Tensor Mesh** is a box mesh where there are no T-joints, i.e., all horizontal and vertical lines span the entire length $[x_0, x_n]$ or $[y_0, y_n]$.

Definition 3. An **LR-Mesh** $\mathcal{M}_n$ is a box mesh which is the result from a series of single line insertions $\{E_i\}_{i=1}^n$ starting from a tensor mesh $\mathcal{M}_0$, i.e. $\mathcal{M}_n \supset \mathcal{M}_{n-1} \supset ... \supset \mathcal{M}_1 \supset \mathcal{M}_0$ and each intermediate state $\mathcal{M}_{i+1} = \{\mathcal{M}_i \cup E_i\}$ is a also a box mesh.

In other words, it is be possible to create these meshes by inserting one line at a time, where these lines never stop in the center of an element (knot span). Figure 1 shows a different types of unstructured meshes.

Definition 4. A box mesh, Tensor Mesh or LR-Mesh **with multiplicities** is a mesh $\mathcal{M}$ where each line segment has a corresponding integer value μ, called the line multiplicity. Each multiplicity must satisfy $0 < \mu \leq s$, where s is the polynomial degree (in x-direction for vertical lines and in y-direction for horizontal lines).

Now we define the spline space $\mathbb{S}^{p,q}(\mathcal{M})$ in term of piecewise polynomials of a given box mesh $\mathcal{M}$ with multiplicities:

$$\mathbb{S}^{p,q}(\mathcal{M}) = \left\{ \varphi|_F \in \mathbb{P}_{p,q} \wedge \varphi|_E \in C^{k(E)} \right\} \tag{11}$$

where $\mathbb{P}_{p,q}$ is polynomials of bi-degree (p,q) and $\mathcal{M} = \{\boldsymbol{F}, \boldsymbol{E}_H, \boldsymbol{E}_V, \boldsymbol{V}\}$ is the mesh defined by the collection of faces F, horizontal edges E_H, vertical edges E_V and vertices V. A continuity k is assigned to each edge and is given by the multiplicity $\mu(E)$ as

$$k(E) = \begin{cases} p - \mu(E), & \text{for vertical edges } E_V \\ q - \mu(E), & \text{for horizontal edges } E_H. \end{cases} \tag{12}$$

Note that it is also associated a horizontal and vertical continuity with each vertex

$$\boldsymbol{k}(V) = \begin{bmatrix} k_1(V) \\ k_2(V) \end{bmatrix} = \begin{bmatrix} \min\{k(E_V)\} \\ \min\{k(E_H)\} \end{bmatrix} \tag{13}$$

where E_V is all vertical edges connected to this particular vertex, and likewise for horizontal edges.

Note that in the case of uniform mesh continuities (k,l), we write $\mathbb{S}^{p,q}_{k,l}(\mathcal{M})$. Even if the continuities are implicitly defined in the mesh $\mathcal{M}$ and hence it is possible to drop the continuity subscripts, we feel that they emphasize some key facts and it is illustrative to keep them whenever possible. However, all results presented in this paper will hold true for mixed continuities $\mathbb{S}^{p,q}(\mathcal{M})$.

Now using the result from Mourrain [27], it can be shown that the dimension of the spline space $\mathbb{S}^{p,q}(\mathcal{M})$ will be

$$\begin{aligned}\dim\left(\mathbb{S}^{p,q}(\mathcal{M})\right) &= \sum_{F\in\boldsymbol{F}}(p+1)(q+1) \\ &- \sum_{E_V\in\boldsymbol{E}_V}(p+1)(k(E_V)+1) \\ &- \sum_{E_H\in\boldsymbol{E}_H}(q+1)(k(E_H)+1) \\ &+ \sum_{V\in\boldsymbol{V}}(k_1(V)+1)(k_2(V)+1) \\ &+ \mathcal{H}^{p,q}(\mathcal{M})\end{aligned} \tag{14}$$

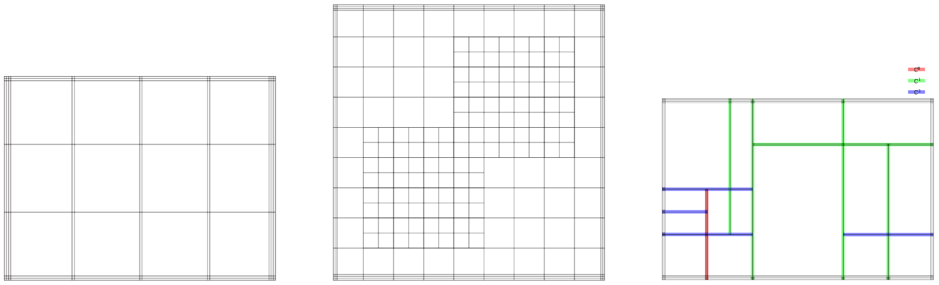
where $\mathcal{H}^{p,q}(\mathcal{M})$ denotes the homology term that depends on a given mesh $\mathcal{M}$ and polynomial bi-degree (p,q). The homology is rather cumbersome to handle, but we will see later that for all practical meshes considered in this paper they have zero homology term. For uniform continuity (k,l) across the entire mesh, and with $\mathcal{H}^{p,q}(\mathcal{M})=0$, the above (14) is simplified to

$$\begin{aligned}\dim\left(\mathbb{S}^{p,q}_{k,l}(\mathcal{M})\right) &= (p+1)(q+1)\#F \\ &- (p+1)(l+1)\#E_V \\ &- (q+1)(k+1)\#E_H \\ &+ (k+1)(l+1)\#V\end{aligned} \tag{15}$$

where $\#F$ is the number of faces in the mesh, $\#E_V$ and $\#E_H$ is the number of interior vertical and horizontal edges, respectively, and $\#V$ is the number of interior vertices. In the later sections of the paper we shorten the notation and simply write F for the number of faces $\#F$ and likewise for edges and vertices.

Proposition 1. *For an LR mesh $\mathcal{M}$, a sufficient condition for the homology term $\mathcal{H}^{p,q}(\mathcal{M})$ to be zero is that it is constructed of horizontal lines spanning $p+1$ elements and vertical lines spanning $q+1$ elements.*

Proof. For this consult Pettersen [29]. □



(a) Mesh description:
$p=3,\ q=2$
$k=1,\ l=1$
$\#F=12,\ \#V=6$
$\#E_H=8,\ \#E_V=12$
$\dim(\mathbb{S}^{p,q}_{k,l}) = 50$ according to (15)

(b) Mesh description:
$p=2,\ q=3$
$k=1,\ l=2$
$\#F=174,\ \#V=171$
$\#E_H=172,\ \#E_V=172$
$\dim(\mathbb{S}^{p,q}_{k,l}) = 190$ according to (15)

(c) Mesh description:
$p=2,\ q=3$,
mixed continuity
$\#F=15,\ \#V=12$
$\#E_H=12,\ \#E_V=14$
$\mathcal{H}^{p,q}(\mathcal{M}) = 0$ and $\dim(\mathbb{S}^{p,q}) = 38$ according to (14)

Figure 2: The dimension of different spline spaces $\mathbb{S}^{p,q}$ over box-meshes of bi-degree (p,q) and varying smoothness.

3.1 Derivative spaces

The derivative spaces of $\mathbb{S}^{p,q}(\mathcal{M})$ defined as a piecewise polynomial space in (11) over an arbitrary box mesh $\mathcal{M}$ can be defined as follows:

Definition 5. Let $\{\varphi_i\}_{i=1}^n$ be a basis for the space $\mathbb{S}^{p,q}(\mathcal{M})$ as defined in (11). Then both components of the **derivative spaces** can be defined as

$$\partial_x\mathbb{S}^{p,q}(\mathcal{M}) = \operatorname{span}\left\{\frac{\partial}{\partial x}\varphi_i(x,y)\right\}_{i=1}^n \tag{16}$$

$$\partial_y\mathbb{S}^{p,q}(\mathcal{M}) = \operatorname{span}\left\{\frac{\partial}{\partial y}\varphi_i(x,y)\right\}_{i=1}^n . \tag{17}$$

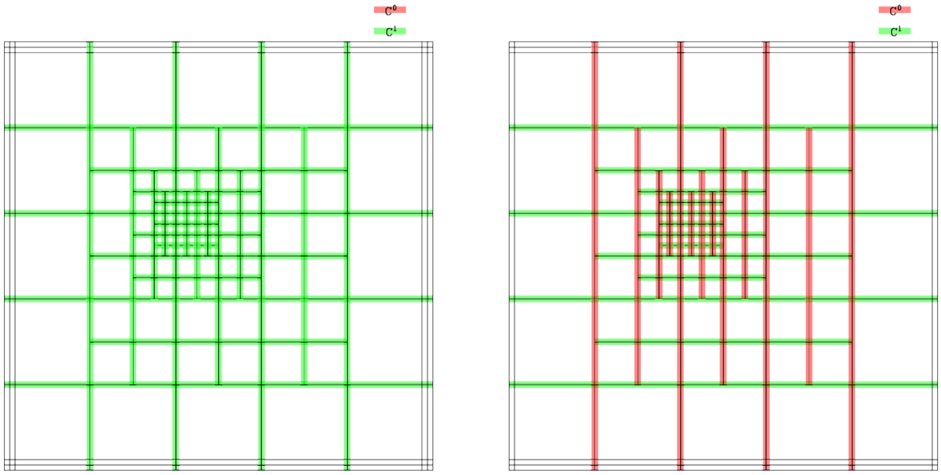
We make the following observation about the derivative spaces of Definition 5.

Proposition 2. *Let $\mathcal{M}$ be an arbitrary box mesh with multiplicities and $\mathbb{S}^{p,q}(\mathcal{M})$ be a spline space over $\mathcal{M}$. Then we obtain*

$$\partial_x\mathbb{S}^{p,q}(\mathcal{M}) \subseteq \mathbb{S}^{p-1,q}(\mathcal{M}) \tag{18}$$

$$\partial_y\mathbb{S}^{p,q}(\mathcal{M}) \subseteq \mathbb{S}^{p,q-1}(\mathcal{M}). \tag{19}$$

Proof. For a given $\varphi \in \mathbb{S}^{p,q}(\mathcal{M})$, under the x-derivative operation, the polynomial degree is reduced by one, i.e. $\frac{\partial\varphi}{\partial x}|_F \in \mathbb{P}^{p-1,q}$, and the continuity across vertical directions are also reduce by one, i.e. $\frac{\partial\varphi}{\partial x}|_{E_V} \in C^{k(E_V)}$, where



(a) Mesh description of $\mathbb{S}^{p,q}_{k,l}(\mathcal{M})$ with
$p = 2,\ q = 2,\ k = 1,\ l = 1$
$\#F = 106,\ \#V = 115$
$\#E_H = 110,\ \#E_V = 110$
$\dim(\mathbb{S}^{p,q}_{k,l}) = 94$ according to (15)

(b) Mesh description of $\mathbb{S}^{p,q}_{k,l}(\mathcal{M})$ with
$p = 1,\ q = 2,\ k = 0,\ l = 1$
$\#F = 106,\ \#V = 115$
$\#E_H = 110,\ \#E_V = 110$
$\dim(\mathbb{S}^{p,q}_{k,l}) = 96$ according to (15)

Figure 3: **Derivative spaces:** *A counter example*: On this box mesh, It is shown that the derivative space is *not* the space of all polynomials of one less degree and continuity: $\partial_x \mathbb{S}^{p,q}_{k,l}(\mathcal{M}) \neq \mathbb{S}^{p-1,q}_{k-1,l}(\mathcal{M})$ since $\dim(\mathbb{S}^{p-1,q}_{k-1,l}(\mathcal{M})) > \dim(\mathbb{S}^{p,q}_{k,l}(\mathcal{M}))$. The continuity colours are derived from the edge multiplicities which are the same for both figures.

$k(E_V) = p - 1 - \mu(E_V)$ with the edge multiplicity $\mu(E_V)$. While the continuity across horizontal edges remain unchanged. Hence $\frac{\partial \varphi}{\partial x} \in \mathbb{S}^{p-1,q}(\mathcal{M})$. The proof of (19) is analoge. □

Proposition 3. *Let $\mathcal{M}$ be a tensor mesh with uniform multiplicities, i.e. $\mu(E_V) = p - k,\ \forall E_V$ and $\mu(E_H) = q - l,\ \forall E_H$ with (k, l) being global continuities in each direction. Then*

$$\partial_x \mathbb{S}^{p,q}_{k,l}(\mathcal{M}) = \mathbb{S}^{p-1,q}_{k-1,l}(\mathcal{M}) \tag{20}$$

Proof. See [8] for the proof. □

The results of Proposition 3 does not hold for general box meshes. For a counter example see Figure 3, where we present an example to show this case on a given box mesh.

4 The spline complex over box meshes

We note that the results of this section will hold true for any properly defined spline spaces over box meshes. We consider the dimensionality argument approach to prove the compatibility in the spline spaces, where the dimensional formula's of Mourrain [27] for box meshes is used as a main tool. Thus its become a requirement that the spline spaces should span the full space of piecewise smooth polynomials as given by (14).

Theorem 2. *Let $\mathcal{M}$ be a given box mesh with multiplicities and $\mathbb{S}^{p+1,q+1}(\mathcal{M})$ be a spline space as defined in* (11). *If the homology term $\mathcal{H}^{p+1,q+1}(\mathcal{M}) = 0$, then the spline spaces X_h^0, X_h^1 and X_h^2 form a De Rham complex and the following sequence is exact*

$$\mathbb{R} \rightarrow X_h^0 \xrightarrow{\boldsymbol{rot}} X_h^1 \xrightarrow{div} X_h^2 \rightarrow 0 \tag{21}$$

where

$$\begin{aligned} X_h^0 &= \mathbb{S}^{p+1,q+1}(\mathcal{M}) \\ X_h^1 &= \mathbb{S}^{p+1,q}(\mathcal{M}) \times \mathbb{S}^{p,q+1}(\mathcal{M}) \\ X_h^2 &= \mathbb{S}^{p,q}(\mathcal{M}). \end{aligned}$$

Proof. The proof follows the same structure as outlined by Buffa el al. [10]. To prove (21), we need to show the following:

$$\mathbb{R} = \ker(\mathbf{rot}) \tag{22}$$

$$\mathrm{im}(\mathbf{rot}) = \ker(\mathrm{div}) \tag{23}$$

$$\mathrm{im}(\mathrm{div}) = X_h^2, \tag{24}$$

where $\mathbf{rot}(\varphi) = [\partial_y \varphi, -\partial_x \varphi]^T$ and $\mathrm{div}(\boldsymbol{u}) = \partial_x u_1 + \partial_y u_2$.

The proof of (22) is straightforward. We observe that

$$\varphi \in X_h^0 : \mathbf{rot}(\varphi) = [0,0]^T \Leftrightarrow \varphi = c \in \mathbb{R}.$$

Hence $\mathbb{R} = \ker(\mathbf{rot})$.

To prove (23), we first note that

$$\mathrm{im}(\mathbf{rot}) \subseteq \ker(\mathrm{div}) \quad \text{since} \quad \forall \varphi \in X_h^0 \Rightarrow \mathrm{div}(\mathbf{rot}(\varphi)) = 0.$$

While to show $\mathrm{im}(\mathbf{rot}) \supseteq \ker(\mathrm{div})$, assume $\mathrm{div}(\boldsymbol{u}) = 0$. Then there exists an $\varphi \in X_h^0$ such that $\boldsymbol{u} = \mathbf{rot}(\varphi)$ and φ is given as

$$\varphi(x,y) = -\int_0^x u_2(t,0)\, dt + \int_0^y u_1(x,t)\, dt.$$

The proof of (24) is based on dimensionality argument of the spline spaces. First using Proposition 2 we obtain $\mathrm{im}(\mathrm{div}) \subseteq X_h^2$. Now to establish equality we need to show that the dimensions of both spaces are equal, i.e.,

$$\begin{aligned}
\dim(\mathrm{im}(\mathrm{div})) &= \dim(X_h^1) - \dim(\ker(\mathrm{div})) \\
&= \dim(X_h^1) - \dim(\mathrm{im}(\mathbf{rot})) \\
&= \dim(X_h^1) - \dim(X_h^0) + \dim(\ker(\mathbf{rot})) \\
&= \dim(X_h^1) - \dim(X_h^0) + 1. \qquad (25)
\end{aligned}$$

After assuming the uniform continuity over the mesh $\mathcal{M}$ we obtain from (15):

$$\begin{aligned}
\dim(\mathbb{S}^{p+1,q+1}_{k+1,l+1}) &= (p+2)(q+2)F - (p+2)(l+2)E_H - (q+2)(k+2)E_V + (k+2)(l+2)V \\
\dim(\mathbb{S}^{p+1,q}_{k+1,l}) &= (p+2)(q+1)F - (p+2)(l+1)E_H - (q+1)(k+2)E_V + (k+2)(l+1)V \\
\dim(\mathbb{S}^{p,q+1}_{k,l+1}) &= (p+1)(q+2)F - (p+1)(l+2)E_H - (q+2)(k+1)E_V + (k+1)(l+2)V \\
\dim(\mathbb{S}^{p,q}_{k,l}) &= (p+1)(q+1)F - (p+1)(l+1)E_H - (q+1)(k+1)E_V + (k+1)(l+1)V.
\end{aligned}$$

Using (25), the problem reduce to show only

$$\begin{aligned}
\dim(X_h^2) &= \dim(X_h^1) - \dim(X_h^0) + 1 \\
\dim(\mathbb{S}^{p,q}_{k,l}) &= \dim(\mathbb{S}^{p+1,q}_{k+1,l}) + \dim(\mathbb{S}^{p,q+1}_{k,l+1}) - \dim(\mathbb{S}^{p+1,q+1}_{k+1,l+1}) + 1
\end{aligned}$$

which can be done once we realize that the Euler characteristic of a planar graph is 1, i.e.

$$F - E_V - E_H + V = 1.$$

Note that the assumption on the uniform continuity used here is not required as it is possible to obtain the same conclusion by using (14) in (25). □

The main aim of the paper is to show the use of locally refined div-compatible spline spaces in mixed FE discretization to solve the incompressible flow problems. Thus we now present the extension of a De Rhams complex result of Theorem 2 after imposing the boundary conditions on the velocity field. We consider two main cases of imposing the boundary conditions.

4.1 No penetration boundary conditions

The No penetration boundary condition on the velocity is defined by $\boldsymbol{u}\cdot\boldsymbol{n} = 0$ on the domain boundary. In order to produce an exact De Rham complex, we need to impose corresponding boundary conditions for the other spaces.

Theorem 3. *Let $\mathcal{M}$ be a given box mesh with multiplicities and $\mathbb{S}^{p+1,q+1}(\mathcal{M})$ be a spline space as defined in* (11)*. If the homology term $\mathcal{H}^{p+1,q+1}(\mathcal{M}) = 0$,* *then the spline spaces* Y_h^0, Y_h^1 *and* Y_h^2 *form a De Rham complex and the following sequence is exact*

$$0 \to Y_h^0 \xrightarrow{\boldsymbol{rot}} Y_h^1 \xrightarrow{div} Y_h^2 \xrightarrow{\int} 0 \tag{26}$$

where

$$\begin{array}{lcll} Y_h^0 & = & \{\varphi \in \mathbb{S}^{p+1,q+1}(\mathcal{M}) & : \ \varphi = 0 \ on \, \Gamma\} \\ Y_h^1 & = & \{\boldsymbol{u} \in \mathbb{S}^{p+1,q}(\mathcal{M}) \times \mathbb{S}^{p,q+1}(\mathcal{M}) & : \ \boldsymbol{u}\cdot\boldsymbol{n} = 0 \ on \, \Gamma\} \\ Y_h^2 & = & \{p \in \mathbb{S}^{p,q}(\mathcal{M}) & : \ \int_\Omega p = 0\}. \end{array}$$

Here Γ *is the boundary of our domain and* $\boldsymbol{n}$ *is the outward pointing unit normal.*

Proof. The proof follows the main structure of Theorem 2. The main difference in the proof is to show that im(div) $= Y_h^2$. For this, we need to introduce the exterior edges and vertices along the boundary of our domain to account for the lost degrees of freedom when imposing the constraints. Let E_H^E, E_V^E, V_H^E and V_V^E denote the number of horizontal edges (at the top/bottom of our domain), vertical edges (left/right), horizontal vertices (top/bottom) and vertical vertices (left/right), respectively. Here, we do not count the four corner vertices among V_V^E and V_H^E as these do not contribute to inter-element regularity. Then we obtain

$$\begin{array}{rcllllll} \dim(Y_h^0) & = & \dim(\mathbb{S}^{p+1,q+1}_{k+1,l+1}) & -\ (p+2)E_H^E & -\ (q+2)E_V^E & +\ (k+2)V_H^E & +\ (l+2)V_V^E & +4 \\ \dim(Y_h^{1,1}) & = & \dim(\mathbb{S}^{p+1,q}_{k+1,l}) & & -\ (q+1)E_V^E & & +\ (l+1)V_V^E & \\ \dim(Y_h^{1,2}) & = & \dim(\mathbb{S}^{p,q+1}_{k,l+1}) & -\ (p+1)E_H^E & & +\ (k+1)V_H^E & & \\ \dim(Y_h^2) & = & \dim(\mathbb{S}^{p,q}_{k,l}) & & & & & -1. \end{array}$$

□

The term +4 in the first line is due to the four corners being constrained twice in the four terms prior to this. Realizing that dim(ker(**rot**)) = 0, due to the boundary conditions, we have to show the dimension equality

$$\dim(Y_h^2) = \dim(Y_h^1) - \dim(Y_h^0).$$

Here we use the fact that splitting a boundary curve into edges and vertices, we have the formula: $E_H^E - V_H^E = 1$, which is enough to prove the dimensions match.

4.2 No slip boundary conditions

The No slip boundary condition on the velocity field is defined by $\boldsymbol{u} = \boldsymbol{0}$ on the boundary. Again, we will need to provide corresponding restraints on the accompanying spaces (pressure and potential) to make the spline complex exact.

Theorem 4. *Let $\mathcal{M}$ be a given box mesh with multiplicities and $\mathbb{S}^{p+1,q+1}(\mathcal{M})$ be a spline space as defined in* (11)*. If the homology term $\mathcal{H}^{p+1,q+1}(\mathcal{M}) = 0$, then the spline spaces Z_h^0, Z_h^1 and Z_h^2 form a De Rham complex and the following sequence is exact*

$$0 \to Z_h^0 \xrightarrow{\boldsymbol{rot}} Z_h^1 \xrightarrow{div} Z_h^2 \xrightarrow{\int} 0 \tag{27}$$

where

$$\begin{array}{rcll} Z_h^0 & = & \{\varphi \in \mathbb{S}^{p+1,q+1}(\mathcal{M}) & : \ \varphi = 0 \ \wedge \ \frac{\partial \varphi}{\partial \boldsymbol{n}} = 0 \ on \, \Gamma\} \\ Z_h^1 & = & \{\boldsymbol{u} \in \mathbb{S}^{p+1,q}(\mathcal{M}) \times \mathbb{S}^{p,q+1}(\mathcal{M}) & : \ \boldsymbol{u} = \boldsymbol{0} \ on \, \Gamma\} \\ Z_h^2 & = & \{p \in \mathbb{S}^{p,q}(\mathcal{M}) & : \ \int_\Omega p = 0 \ \wedge \ p(x_i) = 0 \ , \ i = \{1...4\}\} \end{array}$$

and $\mathcal{M}$ is a box mesh with multiplicities. Here Γ is the boundary of our domain and $\boldsymbol{n}$ is the outward pointing unit normal while x_i are the four corner points.

Proof. The proof here follows a similar structure of the dimensionality argument as above and we obtain

$$\begin{array}{rcllllll} \dim(Z_h^0) & = & \dim(\mathbb{S}_{k+1,l+1}^{p+1,q+1}) & - \ 2(p+2)E_H^E & - \ 2(q+2)E_V^E & + \ 2(k+2)V_H^E & + \ 2(l+2)V_V^E & +16 \\ \dim(Z_h^{1,1}) & = & \dim(\mathbb{S}_{k+1,l}^{p+1,q}) & - \ (p+2)E_H^E & - \ qE_V^E & + \ (k+2)V_H^E & + \ lV_V^E & +4 \\ \dim(Z_h^{1,2}) & = & \dim(\mathbb{S}_{k,l+1}^{p,q+1}) & - \ (p+1)E_H^E & - \ (q+2)E_V^E & + \ (k+1)V_H^E & + \ (l+2)V_V^E & +4 \\ \dim(Z_h^2) & = & \dim(\mathbb{S}_{k,l}^{p,q}) & & & & & -5 \end{array}$$

To impose both the function value, and its normal derivative, we count the 4 corners and their associated normals twice and hence we need to add 16 to compensate. Again, we have that $\dim(\ker(\mathbf{rot})) = 0$, and the dimension equality

$$\dim(Z_h^2) = \dim(Z_h^1) - \dim(Z_h^0),$$

which can be verify by standard arithmetics. □

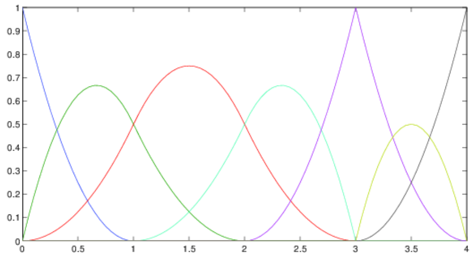


Figure 4: All quadratic basis functions generated by the knot $\Xi = [0, 0, 0, 1, 2, 3, 3, 4, 4, 4]$. Each individual basis function can be described using a local knot vector of lenght 4.

5 LR-splines

In this section we will show how to construct a spline basis. We first introduce the traditional tensor product B-splines as defined by the Cox-de Boor recursion formula, and then continue by presenting locally refined (LR) B-splines. While these allow for quite general meshes, our focus will be the subclass arising from the refinement scheme and our adaptive solvers. These are denoted “structured mesh refinement” and is discussed in the last part of this section.

5.1 Univariate B-splines

Consider a knot vector of non-decreasing knots $\{x_i\}_{i=1}^{n+p+1}$. By elementary spline theory, we can construct a basis on the domain $[x_{p+1}, x_{n+1}]$ by piecewise smooth polynomials using the Cox-de Boor recursion formula

$$\begin{aligned} N_{i,p}(x) &= \frac{x - x_i}{x_{i+p} - x_i} N_{i,p-1}(x) + \frac{x_{i+p+1} - x}{x_{i+p+1} - x_{i+1}} N_{i-1,p-1}(x) \qquad (28) \\ N_{i,0}(x) &= \begin{cases} 1 & \text{if } x_i \le x < x_{i+1} \\ 0 & \text{else} \end{cases} \end{aligned}$$

where, by slight abuse of notation, we define that $\frac{0}{0} := 0$. It is customary (but not required) that the knot vector is *open*, that is the first $p+1$ entires are equal as well as the last $p+1$ entires are equal. In Figure 4 we show an example of a basis constructed on a uniform open knot vector. We will in the following refer to the basis functions $N_{i,p}(x)$ as B-splines. The knot vector holds all the information of the basis constructed. In particular, the following is true

- the B-splines N_i are polynomial and C^∞ in between the knots

- the B-splines are C^{p-m} at the knots, where m is the knot multiplicity
- each B-spline is dependent on exactly $p+2$ knots.

It is the last point, which will allow us to define a local knot vector corresponding to each B-spline, and this observation will be utilized below to introduce LR B-splines.

Definition 6. A bivariate **B-spline** $B(x,y)$ of bi-degree (p,q) is a separable function $B:\mathbb{R}^2\to\mathbb{R}$

$$B_{\Xi^H,\Xi^V}(x,y)=N_{\Xi^H}(x)N_{\Xi^V}(y) \tag{29}$$

defined by the nondecreasing local knot vectors $\Xi^H\in\mathbb{R}^{p+2}$ and $\Xi^V\in\mathbb{R}^{q+2}$, where $N_{\Xi^H}(x)$ and $N_{\Xi^V}(y)$ are univariate B-spline functions defined by the Cox-de Boor recursion formula (28).

We will often just denote a single B-spline by B_i where it is understood that the local knot vectors Ξ^H and Ξ^V are constructed using the refinement algorithm below.

5.2 Refinement of B-splines

At the core of the local refinement, i.e. knot insertion, rests the fact that a single coarse B-spline may be described using a linear combination of two finer B-splines, their relation given by

$$N_\Xi(x)=\alpha_1 N_{\Xi_1}(x)+\alpha_2 N_{\Xi_2}(x), \tag{30}$$

where

$$\begin{aligned}\alpha_1 &= \begin{cases}1, & x_{p+1}\le\hat{x}\le x_{p+2}\\ \frac{\hat{x}-x_1}{x_{p+1}-x_1}, & x_1\le\hat{x}\le x_{p+1}\end{cases}\\ \alpha_2 &= \begin{cases}\frac{x_{p+2}-\hat{x}}{x_{p+2}-x_2}, & x_2\le\hat{x}\le x_{p+2}\\ 1, & x_1\le\hat{x}\le x_2\end{cases}\end{aligned} \tag{31}$$

and the knot vectors are

$$\begin{aligned}\Xi &= [x_1,x_2,...x_{i-1},\quad x_i,...x_{p+1},x_{p+2}]\\ \Xi_1 &= [x_1,x_2,...x_{i-1},\boldsymbol{\hat{x}},\ x_i,...x_{p+1}\quad]\\ \Xi_2 &= [\quad x_2,...x_{i-1},\boldsymbol{\hat{x}},\ x_i,...x_{p+1},x_{p+2}].\end{aligned}$$

Note that the insertion of the knot $\hat{x}$ into Ξ yields a knot vector of size $p+3$, meaning that it is generating two B-splines. These two B-splines are

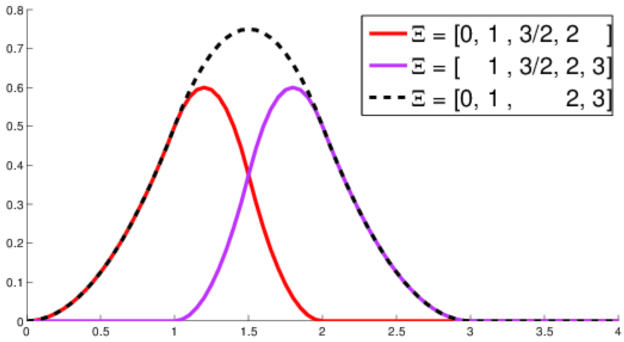
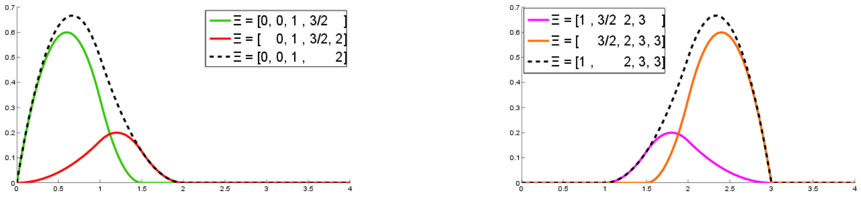


Figure 5: Splitting the B-spline $\Xi = [0, 1, 2, 3]$ into two separate B-splines by inserting the knot $\frac{3}{2}$.

the one being described by the local knot vectors Ξ_1 and Ξ_2, both of size $p + 2$.

Let us look at an example using this technique. Say we want to insert $\hat{x} = \frac{3}{2}$ into the B-spline $\Xi_3 = [0, 1, 2, 3]$. This would give us $\alpha_1 = \alpha_2 = \frac{3}{4}$ and the three functions are plotted in Figure 5. If one were to insert the knot $\hat{x} = \frac{3}{2}$ into the *set* of B-splines in Figure 4, then this will require two more functions to be split, namely the function $\Xi_2 = [0, 0, 1, 2]$ and $\Xi_4 = [1, 2, 3, 3]$. All the three splitting shown in Figure 5–6 will then take place. This insertion will replace three old B-splines with four new linearly independent B-splines (see the knot vectors in the figure legend to identify the four distinctive new B-splines).

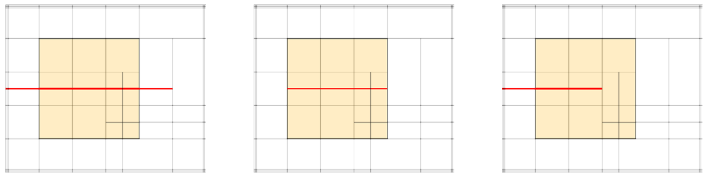


(a) Inserting $x = \frac{3}{2}$ in $\Xi = (0, 0, 1, 2)$.

(b) Inserting $x = \frac{3}{2}$ in $\Xi = (1, 2, 3, 3)$.

Figure 6: Displaying function splitting in the case that $\hat{x}$ is not at the knotvector center.

Bivariate functions are refined in one parametric direction at a time. Using the fact that they are separable we are able to reuse (30) to split one direction and reassemble the bivariate functions after. This can be done as follows



(a) Line traversing B

(b) Line traversing B

(c) Line not traversing B

Figure 7: Traversing the support of a basis function.

$$\begin{aligned} B_{\boldsymbol{\Xi}}(x,y) &= B_{\Xi^H}(x)B_{\Xi^V}(y) \\ &= \Big(\alpha_1 B_{\Xi^H_1}(x) + \alpha_2 B_{\Xi^H_2}(x)\Big) B_{\Xi^V}(y) \\ &= \alpha_1 B_{\boldsymbol{\Xi}_1}(x,y) + \alpha_2 B_{\boldsymbol{\Xi}_2}(x,y). \end{aligned} \tag{32}$$

5.3 Local refinement algorithm

When talking about LR B-splines, we usually distinguish between the mesh $\mathcal{M}$ and the set of B-splines $\mathcal{S}$. The mesh is limited to an LR mesh (see Definition 3) and is represented by the set of all lines; vertical and horizontal. The function space $\mathcal{S}$ is represented by the B-splines themselves, which are uniquely determined by their local knot vectors. The refinement algorithm is the interplay in between these two entities and is categorized by two operations: traversing and splitting.

Definition 7. A line in the mesh $\mathcal{M}$ is said to **traverse** a B-spline B_i if it passes through its support, and all of its support.

See Figure 7 for examples on traversing meshlines.

Definition 8. A knot-line is said to **exist** in a B-spline B_{Ξ^H,Ξ^V} if its (constant) knot value is represented in Ξ^H for vertical lines or Ξ^V for horizontal lines.

Definition 9. A B-spline B_i can be **split** at the knot x (or y) by the application of (32) producing two new B-splines B_1 and B_2. When inserting the two new B-splines into the existing space $\mathcal{S}$, we either update their control points and weights (if any) if they exist already, or create a new entry if they do not exist.

Note that the B-splines are uniquely determined by their local knot vector, and this is used to identify equal (existing) B-splines. Moreover we note the earlier remark on the weights and the control points. The first is a simple multiplication of the B-spline by some scalar γ to maintain the partition of unity, while the second are the control points, often used for the representation of geometric mappings. The weights are optional, in the sense that they offer nothing in terms of the span of the functions (they do however affect numerical stability). The control points are optional if we are not considering a geometric mapping, but rather is only working in the parametric space.

Algorithm 6 Refinement algorithm

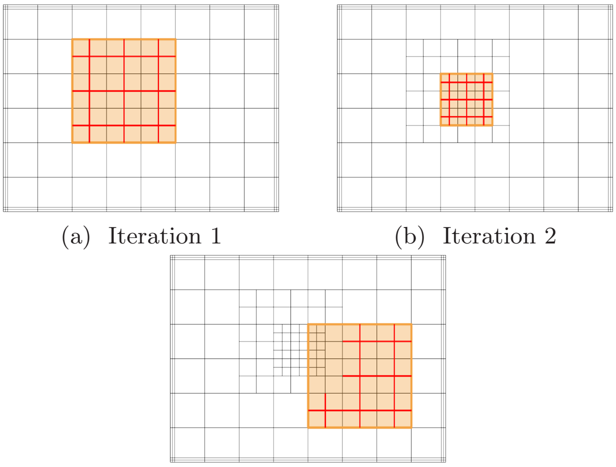
```
 1: Insert new line E
 2: for every B-spline $B_i \in \mathcal{S}$ do
 3:   if E traverse $B_i$ and E does not exist in $B_i$ then
 4:     split $B_i$
 5:   end if
 6: end for
 7: for every newly created B-spline $B_j$ from line 4 or 10 do
 8:   for every existing line $E \in \mathcal{M}$ do
 9:     if E traverse $B_j$ and E does not exist in $B_j$ then
10:       split $B_j$
11:     end if
12:   end for
13: end for
```

We have in this section deliberately simplified several points in the presentation. For a more technical introduction (including details on the weights and control points) we refer the reader to [13] or [23]. For our discussion in this paper however, it is enough to consider the functions as defined in the parametric domain and without weights.

A motivational factor for the use of LR B-splines with spline complexes in Section 4 is their direct construction on the mesh. The integration mesh is the same as the LR mesh where the edge multiplicities are used to construct the reduced continuity lines. For implementation purposes this allows the user to work on a common mesh $\mathcal{M}$, and construct several sets of basis functions $\mathcal{S}^{p+1,q+1}(\mathcal{M})$, $\mathcal{S}^{p+1,q}(\mathcal{M})$, $\mathcal{S}^{p,q+1}(\mathcal{M})$ and $\mathcal{S}^{p,q}(\mathcal{M})$. This not only speeds up computation, but also reduce implementation complexity.

5.4 The LR B-spline complex

We will in this section present the structured mesh refinement as introduced in [23]. It has been shown to provide optimal convergence rates under adaptive refinement for a number of problems containing singularities or rough



(a) Iteration 1

(b) Iteration 2

(c) Iteration 3

Figure 8: Three iterations of an example B-spline refinement given in definition 10. Notice that we at each iteration halve the largest supported elements. A selection of LR B-splines over the mesh from iteration 3 is depicted in Figure 9

right-hand sides and we consider it a good choice for our local refinement strategy.

Definition 10. A **B-spline refinement** on an LR spline $\mathcal{L} = \{\mathcal{M}, \mathcal{S}\}$ is a refinement scheme where one B-spline $B \in \mathcal{S}$ dictates a set of meshline insertions such that the largest knotspan of the local knot vector in B is halved.

See Figure 8 for an example B-spline refinement.

Definition 11. A **Structured LR Mesh** of degree (p, q) is a box mesh resulting from a series of B-spline refinements on an LR spline.

We note that the structured LR B-splines and Hierarchical refined B-splines may produce similar meshes. However, as shown in [22] they are in general not identical, and they produce finite element matrices with different sparsity patterns and conditioning numbers.

Proposition 4. *Any structured LR mesh have homology term* $\mathcal{H}^{p,q}(\mathcal{M})$ *equal to zero.*

Proof. Since every B-spline knot in the local knot vector is appearing in the mesh, and the knot vectors are composed of $p+2$ and $q+2$ knots respectively, we know that each B-spline will span at least $(p+1) \times (q+1)$ elements.

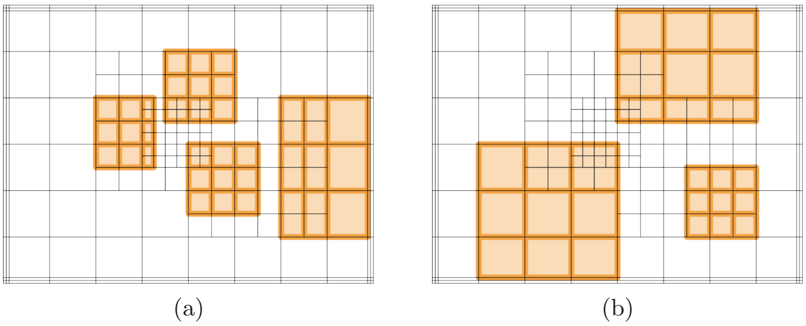


Figure 9: Some example quadratic LR B-splines over the LR mesh from Figure 8c

Every new line inserted into the mesh will span this length and hence the homology term never increases. Since our initial mesh: a tensorial mesh, have $\mathcal{H} = 0$ the proof is complete. □

Proposition 5. *A structured LR mesh of degree (p, q) is also a structured mesh of all degrees $(\hat{p}, \hat{q})$, where $\hat{p} \leq p$ and $\hat{q} \leq q$.*

Proof. We here note that the definition of structured LR mesh is linked to the polynomial degree of the basis constructed on it. For tensor products, we have that every lower order function is completely contained in the support of a function of larger polynomial degree; in both directions. Due to Algorithm 1, when a larger B-spline split, we note that the lower order functions will also be split. Any B-spline of bi-degree (p, q) is thus guaranteed to contain enough functions of lower degree to span it's own support. □

The contrary is not the case. For a structured mesh of bi-degree (p, q), it is not guaranteed that it will be for degree $(p + 1, q)$ or $(p, q + 1)$.

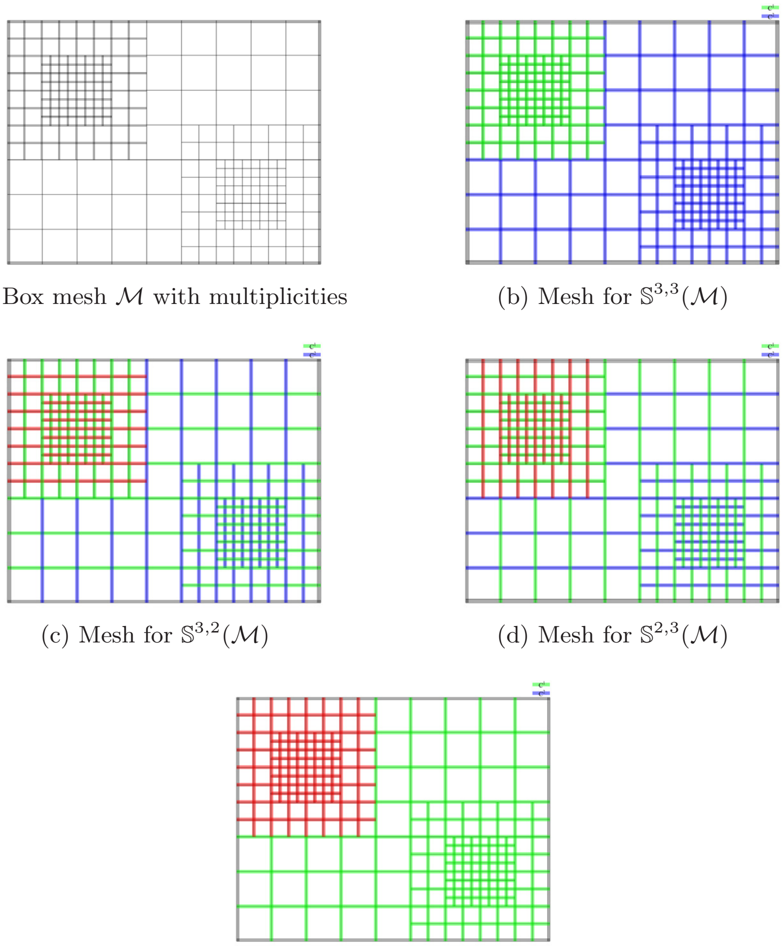
We now are able to construct our spline complex which we will use to discretize the Stokes equations. Consider the four LR splines given on the same structured mesh $\mathcal{M}$ of degree $(p + 1, q + 1)$

$$\begin{aligned}
\mathcal{L}^0 &= \{\mathcal{M}, \mathcal{S}^{p+1,q+1}\} \\
\mathcal{L}^{1,1} &= \{\mathcal{M}, \mathcal{S}^{p+1,q}\} \\
\mathcal{L}^{1,2} &= \{\mathcal{M}, \mathcal{S}^{p,q+1}\} \\
\mathcal{L}^2 &= \{\mathcal{M}, \mathcal{S}^{p,q}\}.
\end{aligned}$$

In order to remain a structured mesh and satisfy a complete De Rham complex, we let the highest degree dictate the B-spline refinements which will drive our adaptive solvers. Where the velocity be given on $\mathcal{L}^{1,1} \times \mathcal{L}^{1,2}$ and the pressure be given on $\mathcal{L}^2$. Setting the LR B-splines $\mathcal{S}$ as the compatible spaces, we have

$$\begin{aligned} X_h^0 &= \mathcal{S}^{p+1,q+1} \\ X_h^1 &= \mathcal{S}^{p+1,q} \times \mathcal{S}^{p,q+1} \\ X_h^2 &= \mathcal{S}^{p,q} \end{aligned}$$

without boundary conditions. Replace X with Y or Z for no-penetration or no-slip boundary conditions, respectively. In Figure 10 we show an example structured LR mesh with varying continuities. Figure 11 shows the corresponding LR B-splines basis representation constructed on the same mesh. By constructing them of different polynomial degrees, we ensure they form a complete De Rham complex.



(a) Box mesh $\mathcal{M}$ with multiplicities

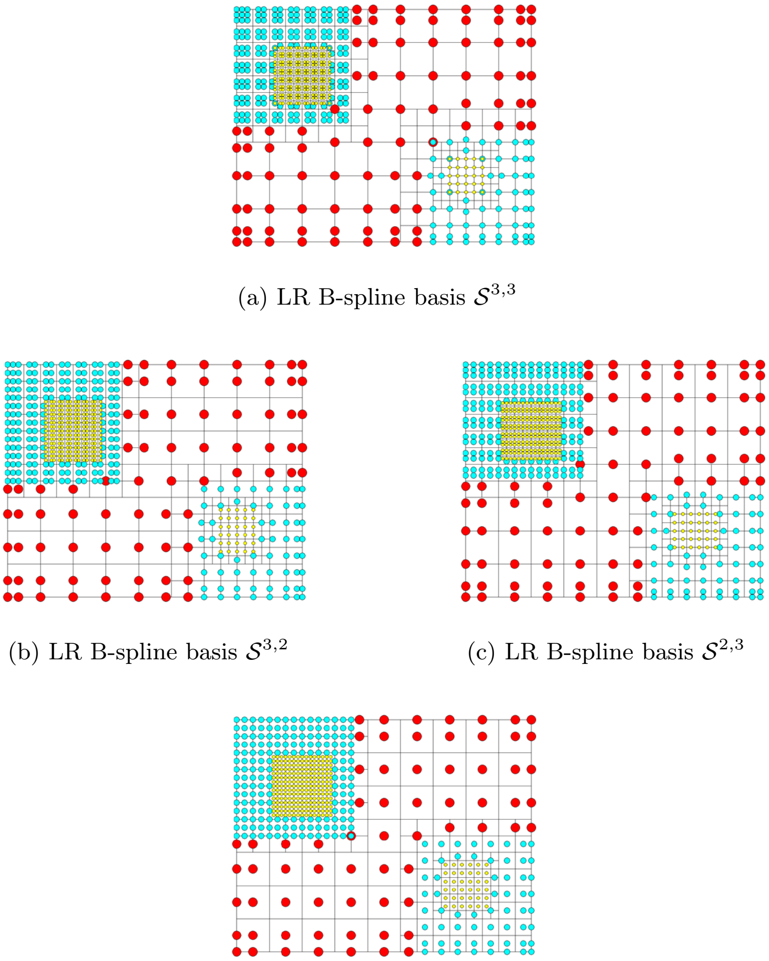
(b) Mesh for $\mathbb{S}^{3,3}(\mathcal{M})$

(c) Mesh for $\mathbb{S}^{3,2}(\mathcal{M})$

(d) Mesh for $\mathbb{S}^{2,3}(\mathcal{M})$

(e) Mesh for $\mathbb{S}^{2,2}(\mathcal{M})$

Figure 10: Example spline spaces over a box mesh $\mathcal{M}$ with multiplicities. Note that it is the same mesh which is used for all figures. The continuity is derived from the polynomial degree of the basis as well as the knotline multiplicity. To construct the spline complex we let X_h^0 be given over (b), X_h^1 be given over (c) and (d), while X_h^2 is defined over (e). When solving the Stokes problem, we let the velocity $\boldsymbol{u}_h \in X_h^1$ and the pressure $p_h \in X_h^2$. The basis functions of X_h^0 is used for refinement purposes to ensure that the De Rham diagram is exact and all meshes are legal.



(a) LR B-spline basis $\mathcal{S}^{3,3}$

(b) LR B-spline basis $\mathcal{S}^{3,2}$

(c) LR B-spline basis $\mathcal{S}^{2,3}$

(d) LR B-spline basis $\mathcal{S}^{2,2}$

Figure 11: Example LR B-spline basis functions over a structured LR mesh $\mathcal{M}$. The functions are plotted at their Greville abscissa and colored according to the following rules: No yellow functions have support outside the finest elements, no teal functions have support on the largest elements and the functions represented in red color are the only ones having support on the largest elements.

6 Numerical results

In this section we present some numerical results to illustrate the performance of compatible LR B-splines discretization for solving the incompressible Stokes problem. The main focus is to show the following:

- Numerical stability for compatible LR B-splines discretization
- Divergence-free computed FE solution
- Efficiency and optimal convergence rate achieved by adaptive analysis.

In the numerical computation we consider three different choices of discrete spaces for the approximation of velocity and pressure fields in the mixed discretization (3) for Stokes problem. These choices of spline spaces, i.e, $\left(\tilde{V}_h, \tilde{Q}_h\right)$, over a general box mesh $\mathcal{M}$ in parametric domain $\hat{\Omega}$ are defined as:

$$\text{Type I} \;:\; \tilde{V}_h := \hat{V}_h; \quad \tilde{Q}_h := \hat{Q}_h; \tag{33}$$

$$\text{Type II} \;:\; \tilde{V}_h := \{\text{div}(\mathbf{v}) : \mathbf{v} \in \hat{V}_h,\ \mathbf{v}\cdot\mathbf{n}|_{\partial\Omega} = 0\}; \tag{34}$$

$$\tilde{Q}_h := \{q \in \hat{Q}_h : \int q = 0\};$$

$$\text{Type III} \;:\; \tilde{V}_h := \{\text{div}(\mathbf{v}) : \mathbf{v} \in \hat{V}_h,\ \mathbf{v}|_{\partial\Omega} = \mathbf{0}\}; \quad \text{and} \tag{35}$$

$$\tilde{Q}_h := \{q \in \hat{Q}_h : \int q = 0 \quad \text{with} \quad q(\hat{\mathbf{x}}_i) = 0, \quad i = 1, \ldots, 4\}.$$

with

$$\hat{V}_h = \mathcal{S}^{p+1,q}_{k+1,\ell} \times \mathcal{S}^{p,q+1}_{k,\ell+1} \quad \text{and} \quad \hat{Q}_h = \mathcal{S}^{p,q}_{k,\ell};$$

where $\mathcal{S}^{p,q}_{k,\ell}(\mathcal{M})$ denotes the two dimensional spline space of degree (p, q) and continuity (k, ℓ) in both directions, respectively, and $\mathbf{n}$ denotes the outward unit normal to the boundary of $\hat{\Omega}$ and $\hat{\mathbf{x}}_i, i = 1, ..., 4$ denote its four corners. In the numerical results presented in this section we always consider the case of equal degree approximation in both direction, i.e., $p = q$.

Error evaluation:

For our model Stokes problem, we distinguish between the velocity and pressure errors. We compute the error in velocity using the H^1 semi-norm defined by

$$|\boldsymbol{u} - \boldsymbol{u}_h|^2_{H^1(\Omega)} = \int_\Omega \nabla(\boldsymbol{u} - \boldsymbol{u}_h) : \nabla(\boldsymbol{u} - \boldsymbol{u}_h)\, d\Omega, \tag{36}$$

and the pressure error in L^2 norm;

$$\|p - p_h\|^2_{L^2(\Omega)} = \int_\Omega (p - p_h)^T \cdot (p - p_h) d\Omega. \tag{37}$$

For smooth problems, a div-compatible B-spline discretization is expected to satisfy:

$$\begin{aligned} |\boldsymbol{u} - \boldsymbol{u}_h|_{H^1} &= \mathcal{O}(h^s) \\ \|p - p_h\|_{L^2} &= \mathcal{O}(h^{s+1}) \end{aligned} \tag{38}$$

where s is the lowest polynomial degree in the approximation spaces pair of (V_h, Q_h), i.e., $s = \min(p, q)$, for the div-compatible LR discretization of $\boldsymbol{u}_h$ and p_h, and h is the radius of smallest circle encompassing the largest element in our discretization.

For locally refined adaptive meshes, we have a wide range of element sizes and it becomes misleading to measure the errors in terms of element size. We then reformulate the relations in (38) in terms of degrees-of-freedom n_{dof}. By observing that a uniform mesh in two dimensions has $n_{\text{dof}} = \mathcal{O}(h^{-2})$, we state that the optimal rate of convergence, as measured against degrees of freedom is

$$\begin{aligned} |\boldsymbol{u} - \boldsymbol{u}_h|_{H^1} &= \mathcal{O}(n_{\text{dof}}^{-s/2}) \\ \|p - p_h\|_{L^2} &= \mathcal{O}(n_{\text{dof}}^{-(s+1)/2}). \end{aligned} \tag{39}$$

Whenever the exact solution is available, we define the error estimate η_F using energy norm at per element (or face) F as

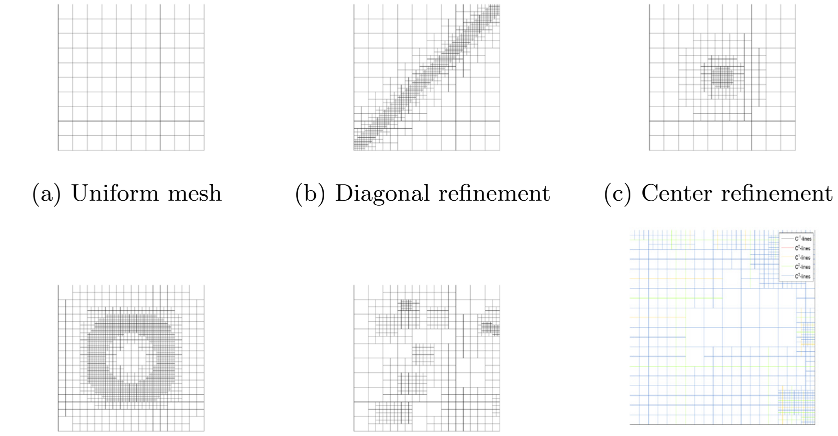
$$\eta_F^2 = \mu|\boldsymbol{u} - \boldsymbol{u}_h|^2_{H^1} + \|p - p_h\|^2_{L^2} \tag{40}$$

and an error contribution to each B-spline basis function η_B as

$$\eta_B^2 = \sum_{F \in \text{supp}(B)} \eta_F^2. \tag{41}$$

Marking strategy

The marking strategy, that is, the method of how to choose the basis functions for refinement in structured mesh refinement is taken from [23], where once we have the value of estimated error at element level given by (41) (here elements is the same as face that we denote F), then we sum the element error on all elements within the support of each basis function. In the refinement strategies we always choose to refine some percentages of



(a) Uniform mesh (b) Diagonal refinement (c) Center refinement

(d) Circular refinement (e) Random refinement (f) Mixed continuities

Figure 12: **Stability tests on Structured LR meshes:** LR meshes used for evaluation of the inf-sup constant. Note that the refinements (e)-(f) were computed randomly and as such differ between each simulation and discretization. The other meshes were computed algorithmically, and only depend on p.

basis functions which contributed the most error in FEM computation. In [23], it was demonstrated that for a fixed percentage say $\beta = 5, 10, 20$ one always achieved a proper adaptive refinement process resulting in optimal convergence rates. For the implementation in this article we always consider $\beta = 10$ of the basis function to refine in each adaptive refinement steps as this choice doesn't give over-refinements. In our refinement strategy, we always refine the LR B-spline basis functions of the potential space $\mathcal{S}^{p+1,q+1}(\mathcal{M})$ and then the construction of div-compatible LR B-spline spaces follows as we discussed in Section 5.

6.1 Stability tests of Structured LR meshes

The performance of our methodology is based on the notion of Ladyženskaja-Babuška-Brezzi(LBB) condition, or the discrete inf-sup condition, cf. (4). The different choices of discrete spaces (Q_h, V_h) as Type I, II, and III as defined above is considered on a set of *structured* LR meshes as shown in Figure 12. These meshes are constructed via some particular refinements (see Figures 12(a)-(d)) or randomly generated meshes (see Figures 12(e)-(f)), and represent the case of different compatible spline spaces of degrees p. Tables 1-3 show the computed values of inf-sup constant c_{is} with different

p	Uniform	Diagonal	Center	Circle	Random	Mixed
1	0.9370	0.6535	0.6837	0.6802	0.5876	0.6606
2	0.9375	0.6098	0.6204	0.6459	0.6278	0.5873
3	0.8818	0.5378	0.8818	0.5295	0.5171	0.5401

Table 1: **Stability tests on Structured LR meshes:** : Computed inf-sup constant c_{is} for Type I discretization without boundary conditions.

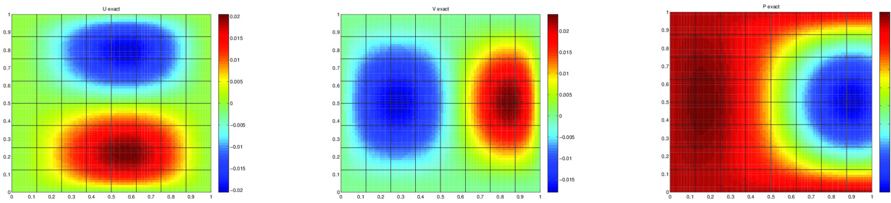
p	Uniform	Diagonal	Center	Circle	Random	Mixed
1	0.9306	0.6534	0.6837	0.6802	0.5842	0.6222
2	0.8912	0.6097	0.6204	0.6458	0.6117	0.5680
3	0.8240	0.5378	0.8240	0.5293	0.5810	0.5524

Table 2: **Stability tests on Structured LR meshes:** : Computed inf-sup constant c_{is} for Type II discretization with no penetration boundary conditions $\boldsymbol{u} \cdot \boldsymbol{n} = 0$.

choices of discrete spaces of Type I-III, respectively. It is confirmed from our computation in Tables 1- 3 that the inf-sup constant $c_{is} > 0$ and has large values.

p	Uniform	Diagonal	Center	Circle	Random	Mixed
1	0.4591	0.4243	0.4592	0.4108	0.3540	0.3481
2	0.4908	0.4534	0.4908	0.4761	0.4447	0.2338
3	0.4833	0.4561	0.4833	0.4708	0.1387	0.3095

Table 3: **Stability tests on Structured LR meshes:** : Computed inf-sup constant c_{is} for Type III discretization with no slip boundary conditions $\boldsymbol{u} = \boldsymbol{0}$.

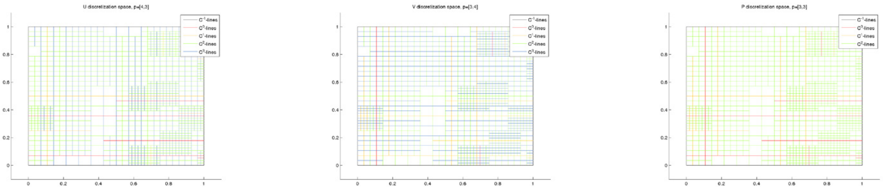


(a) Exact velocity x-component

(b) Exact velocity y-component

(c) Exact pressure

Figure 13: **Stokes problem with smooth solution**: The exact solution of the Stokes problem given in (42).

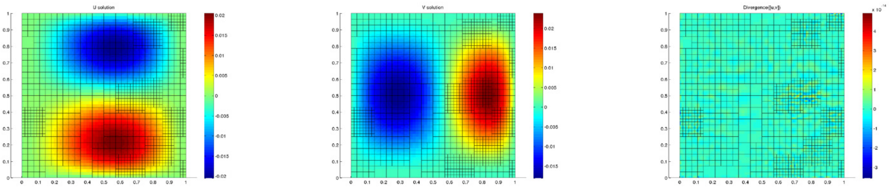

U discretization space, p=[4,3]
V discretization space, p=[3,4]
P discretization space, p=[3,3]
C⁻¹-lines
C⁰-lines
C¹-lines
C²-lines
C³-lines

(a) $\mathcal{S}^{4,3}(\mathcal{M})$ for velocity x-component

(b) $\mathcal{S}^{3,4}(\mathcal{M})$ for velocity y-component

(c) $\mathcal{S}^{3,3}(\mathcal{M})$ for pressure

Figure 14: **Stokes problem with smooth solution**: Compatible LR B-splines approximation spaces for the velocity and pressure fields on the irregular LR mesh $\mathcal{M}$ with mixed continuities using LR B-splines.



(a) Computed velocity x-component

(b) Computed velocity y-component

(c) Divergence of computed velocity

Figure 15: **Stokes problem with smooth solution**: Finite element solution of Stokes problem with smooth solution using LR B-spline compatible spaces on a (randomly generated) irregular LR mesh, with no-slip boundary conditions $\boldsymbol{u} = \mathbf{0}$. The LR B-spline compatible discretization shows pointwise divergence free solution up to machine precision of $\mathcal{O}(10^{-14})$.

6.2 Divergence-free computed FE solution

Example 1: Stokes problem with smooth solution

We consider an example of Stokes problem with smooth solution on a square domain $\Omega = (0,1)^2$ with no-slip boundary conditions as presented in Buffa et al. [8]. The viscosity term is taken as $\mu = 1$ and $\mathbf{f}$ is constructed based on the exact solution given as:

$$\begin{aligned} \boldsymbol{u} &= \begin{bmatrix} 2e^x(x-1)^2x^2(y^2-y)(2y-1) \\ -e^x(x-1)x(-2+x(x+3))(y-1)^2y^2 \end{bmatrix} \\ p &= (-424+156e+(y^2-y)(-456+e^x(456+x^2(228-5(y^2-y)) \qquad (42) \\ &\quad +2x(-228+(y^2-y))+2x^3(-36+(y^2-y))+x^4(12+(y^2-y)))). \end{aligned}$$

These exact solutions are depicted in Figure 13. We consider the Type III discretization to solve this problem with the choice of compatible LR B-spline spaces defined on irregular randomly generated LR mesh in Figure 14. The computed FE solution using these compatible LR B-spline spaces are shown in Figure 15. It can be observe from the divergence of computed velocity field shown in Figure 15(c) that the LR B-spline compatible discretization gives the pointwise divergence free solution up to machine precision of $\mathcal{O}(10^{-14})$.

6.3 Optimal convergence rates

For the Type III choice of discrete spaces with tensor product B-splines, it has been pointed out in [8] that the error in H^1-seminorm of the velocity will be of optimal order, i.e., $\mathcal{O}(h^p)$, whereas the error in L^2-norm of pressure is limited to linear convergence, regardless the polynomial degree of approximation spaces used. The results in Figure 16 shows the same behavior in our FE computations for compatible B-splines spaces with uniform h-refinement for Stokes problem with smooth solution. The authors in [8] also proposed two solutions based on either removal of corner degrees of freedom or using a particular case of T-splines, and their results show that both choices gave an optimal rate of convergence, i.e. $O(h^{p+1})$, for the pressure.

In our computation we found that the choice of four constraints on discrete pressure space Q_h in Type III discrete space setting results in singularities in the computed pressure solution and that is the main reason for linear rate of convergence in pressure error, regardless the degree of approximations. In literature, we have seen use of adaptive refinement algorithms is very promising to resolve these kind of singularities in the computed FE solution and providing an optimal rate of convergence. Here we consider

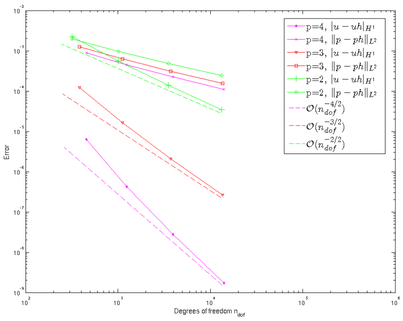
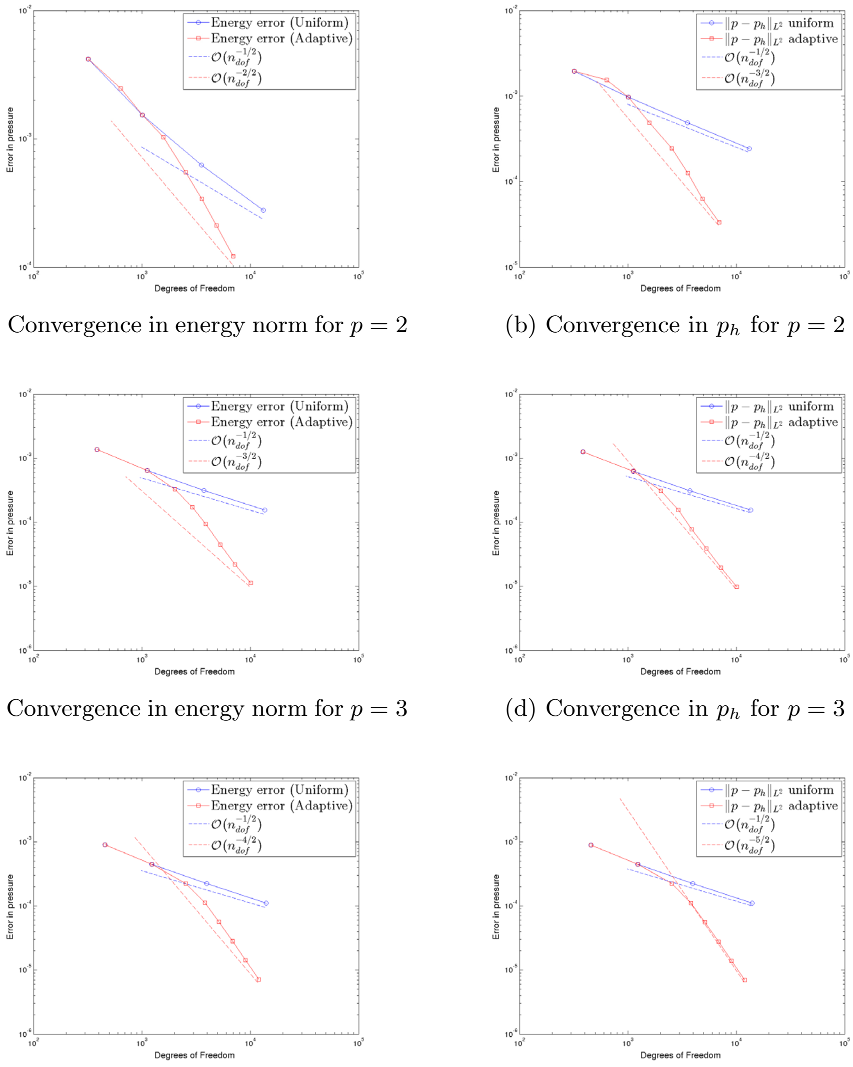


Figure 16: **Stokes problem with smooth solution:** Convergence rates for compatible spline discretization with the choice of Type III discrete spaces. The error in L^2-norm of pressure shows only linear convergence rates, regardless of polynomial degree of discretization, while the error in velocity achieve optimal rates.

to perform adaptive refinement using compatible LR B-spline discretization with Type III space setting and the exact energy error estimate is used as an refinement indicator. The error plots results presented in Figure 17 show that an optimal rate of convergence is achieved via adaptive refinements, i.e., $O(h^{p+1})$ for L^2-norm of the pressure and $O(h^p)$ for the energy norm of the solution. The LR meshes obtained at different steps of adaptive refinements are given in Figure 18.

The singularities introduced here are however artificial. Observe that the exact solution (42) does not have a pressure which vanish at the corners. Strongly enforcing the pressure to the correct value will restore the optimal convergence rates, but this is not possible to do in general when the pressure solution is not known. The typical setup for no-slip (or prescribed slip) problems from a physical point of view is to specify the velocity field at the boundary and not know anything of the pressure. See Section 6.4 for an example of this type of problem setup. When the pressure in the corners is not known we will in this paper set it (wrongly) to zero.



(a) Convergence in energy norm for $p = 2$

(b) Convergence in p_h for $p = 2$

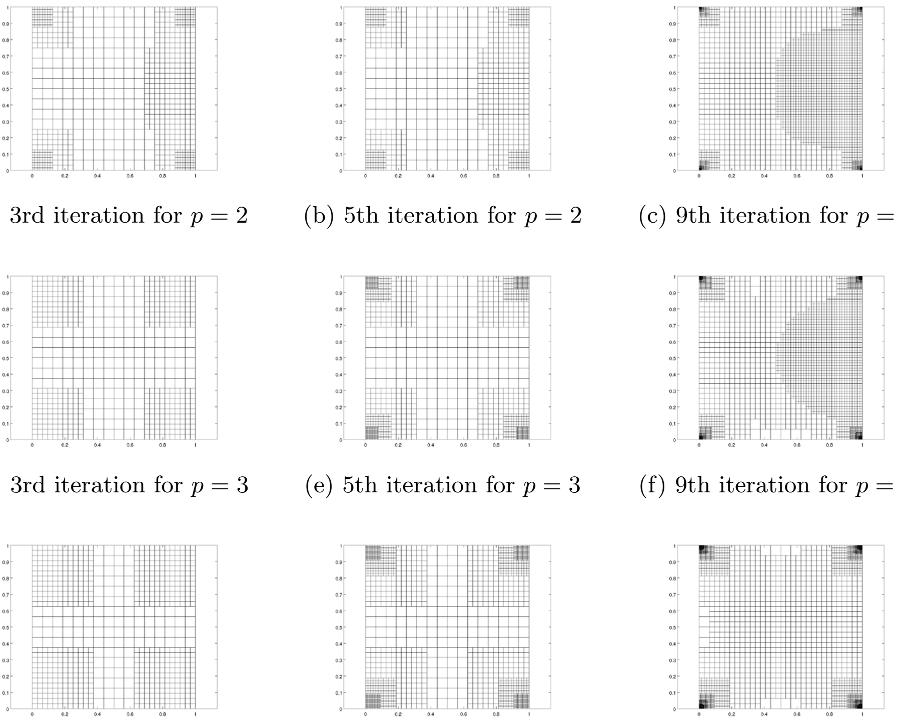
(c) Convergence in energy norm for $p = 3$

(d) Convergence in p_h for $p = 3$

(e) Convergence in energy norm for $p = 4$

(f) Convergence in p_h for $p = 4$

Figure 17: **Stokes problem with smooth solution:** Convergence rates for adaptive compatible LR spline discretization with the choice of Type III discrete spaces based on exact energy error. The error in energy norm and error in L^2-norm of pressure shows optimal rate of convergence.



(a) 3rd iteration for $p = 2$ (b) 5th iteration for $p = 2$ (c) 9th iteration for $p = 2$

(d) 3rd iteration for $p = 3$ (e) 5th iteration for $p = 3$ (f) 9th iteration for $p = 3$

(g) 3rd iteration for $p = 4$ (h) 5th iteration for $p = 4$ (i) 9th iteration for $p = 4$

Figure 18: **Stokes problem with smooth solution:** The LR meshes obtained via adaptive refinements using compatible LR B-splines with Type III discrete setting. As the polynomial degree increases, the discrepancy between pressure error and velocity error for uniform meshes increases. Thus, for higher polynomial degrees the error in the pressure is dominant, resulting in more refinement at the corners.

6.4 The benchmark problem: Lid-driven cavity flow

In this section, we investigate the effectiveness of our methodology for a benchmark case of incompressible flows: two dimensional lid driven cavity problem. The problem setup with a square domain $\Omega = (0,1)^2$ with fixed no-slip boundary conditions on the left, right and bottom side of the domain, with a prescribed velocity $\boldsymbol{u} = [1,0]$ in positive horizontal direction (i.e. to the right) on the top edge is illustrated in Figure 19. The viscosity term is taken as $\mu = 1$ and the forcing $\mathbf{f}$ is defined as zero. The problem setup is known to induce failures in unstable formulations due to the pressure singularities at both top corners of the domain, while in some particular region of interests around both lower corners a infinite series of recirculation regions appears.

The exact solution for the lid driven cavity problem is not known, so we decide to locally refined the LR mesh at all the four corners by hand. The local refinement at the two corners at the top is introduced to suppress the pollution effect of the singularities in the pressure field at those points. The refinement at the two lower corners are introduced to resolve the recirculation zone with high accuracy. The first four hand made locally refined LR meshes are given in Figure 21. To solve the lid driven cavity problem, we consider the Type III pair of discrete approximation spaces on these LR meshes where a no-slip condition is imposed on all sides of the domain for velocity field, with the exception of prescribed velocity at the top which is enforced strongly; and zero average pressure with additional four constraints to force the pressure value to zero at the corner. The computed FE solutions, i.e., the component of velocity and pressure with the divergence of computed velocity field, i.e., $\text{div}(\mathbf{u}_h)$ are shown in Figure 22. The computed solution display pointwise divergence-free solution (up to machine precision) with no spurious oscillations in Figure 22, whereas the velocity profile across the center (horizontal and vertical direction) are shown in Figure 22.

The streamlines plots of the computed velocity field $\boldsymbol{u}_h$ are shown in Figure 23a. Due to the presence of local refinement at the bottom corners we observe three *moffat eddies* in our computed FE solution, even for relatively low degrees of freedom. In our FE-discretization used to compute the streamline plots displayed in Figure 23, we consider $(p,q) = (1,1)$ and $n_{dof} = 3649$ for the pressure field, and $(p,q) = (1,2) \times (2,1), n_{dof} = 7332$ for the velocity fields.

To illustrate the performance our methodology for the lid cavity driven problem, we consider to compare the *curl* of our computed FE solution at

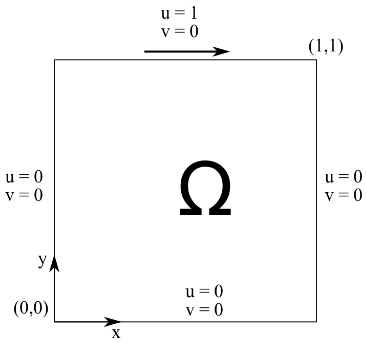


Figure 19: **Lid-driven cavity:** Problem setup for the lid-driven cavity flow.

the point $\boldsymbol{x} = (1, 0.95)$ to available results in literature. The value of the curl of our computed solution at $(1, 0.95)$ using the compatible LR B-spline discretization with uniform and adaptive refinement, along with two available reference solutions in literature, see [18],[7], are shown in Tables 4–7 for Type III pair of discrete spaces of degrees, $p = 1, 2, 3, 4$, respectively. Since our formulation in this paper are based on strong enforcement of the boundary conditions and in Type III pairing of spaces we enforced the pressure at the top corner points to zero that causes the singularities in the solution. This degrades the convergence in the pressure error and hence produce worse result for uniform refinement than those tabulated in [18]. However, our adaptive methodology using local refinement is able to compensate after some iterations. From the results presented in Tables 4–7 the convergence in adaptive refinement can be noticed and as the value of p increases the *curl* value of the computed solution quickly approach to the reference value as given by using Pseduspectral method of [7]. While if we compare our adaptive results with the uniform refinement results of Evans [18] then the efficiency achieved by our adaptive methodology in term of degrees of freedom is clearly noticeable. The difference in number of degrees of freedom for the case of uniform refinement between the present study and the one reported in [18] is due to strong and weak enforcement of of the Dirichlet boundary conditions, respectively.

Present method	h_{min}	h_{max}	n_{dof}	ω
Uniform B-spline	1/16	1/16	764	-14.6690
Uniform B-spline	1/32	1/32	3 068	7.6529
Uniform B-spline	1/64	1/64	12 284	15.2317
Adaptive step #1	1/32	1/16	1 676	7.6529
Adaptive step #2	1/64	1/16	2 588	15.2310
Adaptive step #3	1/128	1/16	3 500	20.7713
Adaptive step #4	1/256	1/16	4 412	22.5232
Adaptive step #5	1/512	1/16	5 324	23.2820
Adaptive step #6	1/1024	1/16	6 236	23.6602
Adaptive step #7	1/2048	1/16	7 148	23.8492
Adaptive step #8	1/4096	1/16	8 060	23.9436
Spline disct.(Ref. [18])	1/64	1/64	12 804	19.0446
Spline disct.(Ref. [18])	1/256	1/256	198 660	25.3224
Pseudospectral (Ref. [7])	-	-	-	27.2790

Table 4: **Lid-driven cavity flow:** Computed values for $\omega = \text{curl}(\boldsymbol{u}_h)$ at the point $\boldsymbol{x} = (1, 0.95)$ for $p = 1$.

Present method	h_{min}	h_{max}	n_{dof}	ω
Uniform B-spline	1/16	1/16	863	-0.2911
Uniform B-spline	1/32	1/32	3 263	17.8810
Uniform B-spline	1/64	1/64	12 671	23.5541
Adaptive step #1	1/32	1/16	1 775	17.8810
Adaptive step #2	1/64	1/16	2 687	23.5540
Adaptive step #3	1/128	1/16	3 599	25.3342
Adaptive step #4	1/256	1/16	4 511	26.3943
Adaptive step #5	1/512	1/16	5 423	26.9274
Adaptive step #6	1/1024	1/16	6 335	27.1947
Adaptive step #7	1/2048	1/16	7 247	27.3283
Adaptive step #8	1/4096	1/16	8 159	27.3951
Spline disct.(Ref. [18])	1/64	1/64	13 199	32.8197
Spline disct.(Ref. [18])	1/256	1/256	200 207	27.3440
Pseudospectral (Ref. [7])	-	-	-	27.2790

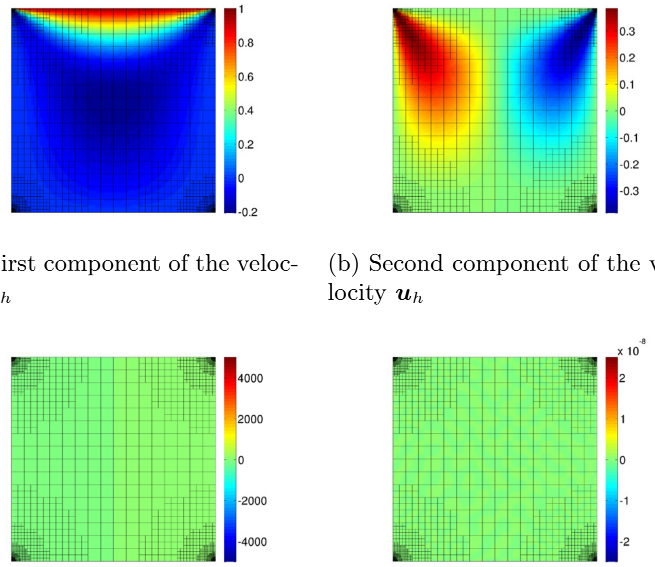
Table 5: **Lid-driven cavity flow:** Computed values for $\omega = \text{curl}(\boldsymbol{u}_h)$ at the point $\boldsymbol{x} = (1, 0.95)$ for $p = 2$.

Present method	$h_{\min}$	$h_{\max}$	n_{dof}	ω
Uniform B-spline	1/16	1/16	968	10.9593
Uniform B-spline	1/32	1/32	3 464	22.1396
Uniform B-spline	1/64	1/64	13 064	24.6936
Adaptive step #1	1/32	1/16	1 880	22.1396
Adaptive step #2	1/64	1/16	2 792	24.6937
Adaptive step #3	1/128	1/16	3 704	25.7774
Adaptive step #4	1/256	1/16	4 616	26.5183
Adaptive step #5	1/512	1/16	5 528	26.9092
Adaptive step #6	1/1024	1/16	6 440	27.1048
Adaptive step #7	1/2048	1/16	7 352	27.2025
Adaptive step #8	1/4096	1/16	8 264	27.2514
Spline disct.(Ref. [18])	1/64	1/64	13 600	29.9294
Spline disct.(Ref. [18])	1/256	1/256	201 760	27.5264
Pseudospectral (Ref. [7])	-	-	-	27.2790

Table 6: **Lid-driven cavity flow:** Computed values for $\omega = \text{curl}(\boldsymbol{u}_h)$ at the point $\boldsymbol{x} = (1, 0.95)$ for $p = 3$.

Present method	$h_{\min}$	$h_{\max}$	n_{dof}	ω
Uniform B-spline	1/16	1/16	1 079	17.9896
Uniform B-spline	1/32	1/32	3 671	22.1748
Uniform B-spline	1/64	1/64	13 463	23.7481
Adaptive step #1	1/32	1/16	1 991	22.1745
Adaptive step #2	1/64	1/16	2 903	23.7487
Adaptive step #3	1/128	1/16	3 815	26.5231
Adaptive step #4	1/256	1/16	4 727	26.6432
Adaptive step #5	1/512	1/16	5 639	26.9621
Adaptive step #6	1/1024	1/16	6 551	27.1183
Adaptive step #7	1/2048	1/16	7 463	27.1964
Adaptive step #8	1/4096	1/16	8 375	27.2354
Pseudospectral (Ref. [7])	-	-	-	27.2790

Table 7: **Lid-driven cavity flow:** Computed values for $\omega = \text{curl}(\boldsymbol{u}_h)$ at the point $\boldsymbol{x} = (1, 0.95)$ for $p = 4$.



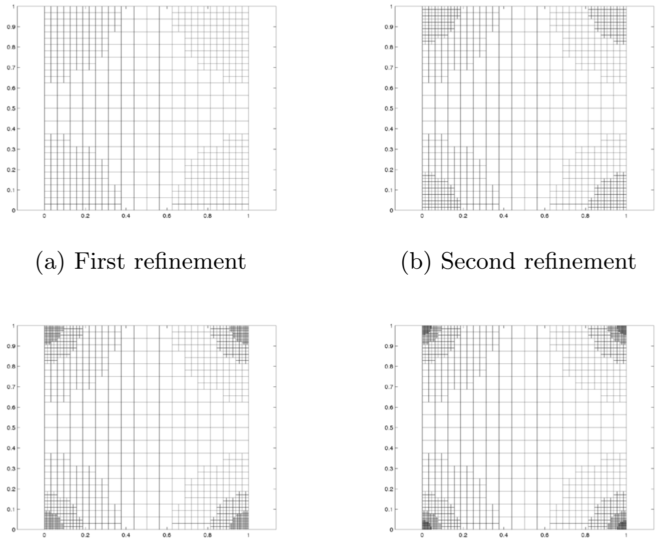
(a) First component of the velocity $\boldsymbol{u}_h$

(b) Second component of the velocity $\boldsymbol{u}_h$

(c) Pressure solution p_h

(d) $\operatorname{div}(\boldsymbol{u}_h)$

Figure 20: **Lid-driven cavity flow:** FE solution plot of the compute velocity, pressure as well as the divergence of the computed solution. Integrating the divergence over the entire domain gives $\|\operatorname{div}(\boldsymbol{u}_h)\|_{L^2} = 2.0 \cdot 10^{-11}$.



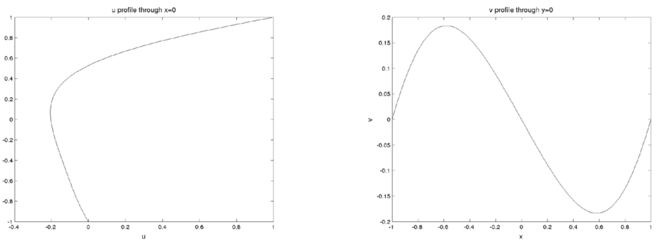
(a) First refinement

(b) Second refinement

(c) Third refinement

(d) Fourth refinement

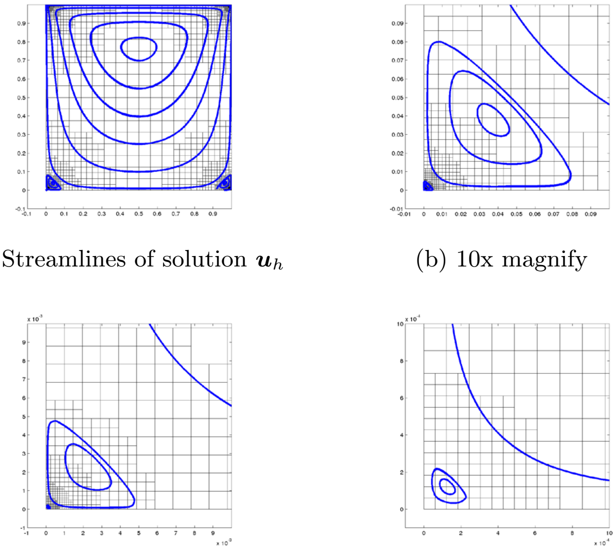
Figure 21: **Lid-driven cavity flow:** The hand-made adaptive mesh refinement used to solve the Lid-driven cavity problem. For each iteration, we refine every B-spline completely contained within a $6.5 \cdot 2^{-4-i}$ radius of each corner.



(a) Values of the first velocity component at a vertical line through the center

(b) Values of the second velocity component at a horizontal line through the center

Figure 22: **Lid-driven cavity flow:** Velocity profiles across the center of the domain.



(a) Streamlines of solution $\boldsymbol{u}_h$

(b) 10x magnify

(c) 100x magnify

(d) 1000x magnify

Figure 23: **Lid-driven cavity flow:** Streamlines of the solution $\boldsymbol{u}_h$. We achieve pointwise divergent-free solution (up to machine precision) using Type III discrete spaces. By refining around the corners we observe three moffatt eddies around the corners, even for relatively low degrees of freedom. The discretization used shown here is $(p, q) = (1, 1)$ and $n = 3649$ for the pressure, and $(p, q) = (1, 2) \times (2, 1), n = 7332$ for the velocity.

7 Conclusions

The aim of this paper has been to take the first step in this direction by extending the previous work on div-compatible spaces by Buffa *et al.* [8, 11] and Evans [18] to be applicable for finite element analysis using LR B-splines. Herein, we have developed the methodology for making div-compatible LR B-spline spaces, i.e. which form complete De Rham complexes. These are stable, pointwise divergent-free and facilitate local refinement capabilities. The numerical tests demonstrate significant improvements in accuracy per degrees of freedom when solving the Stokes problems.

No-slip discretizations are challanging as they require the pressure to be specified in the corners. These will typically require special treatment or we will loose convergence properties. We show that it is possible to set the corner values of the pressure to zero and "refine away" any problems arising from this making it possible to use the strong formulation for no-slip problems. This methodology is conceptually simple and still produce a compatible pointwise divergent-free solution which show optimal convergence in both pressure and velocity.

We have shown that the properties of compatible space discretization carry over from tensor product analysis to locally refined meshes.
The authors consider the following topics suited for future work in this field

- Develop a closed expression for the pressure at the four corner points, only dependent on a prescribed slip $\boldsymbol{u} = \boldsymbol{g}$ on the boundary and the source term $\boldsymbol{f}$ in the interior.
- Include divergence-conforming mappings to handle realistic physical domains [8].
- Consider multiple patches or non-rectangular parametric domains for more complex geometries.
- Enable the use of weakly enforced boundary conditions [18].
- Extend to 3D by the use of the dimensional formula of Pettersen [29].
- Develop suitable error estimates for Stokes and Navier-Stokes flows.
- Investigate the applications of this in electromagnetic differential equations [10].
- Show that the Hierarchical B-splines also satisfy the dimensional formula [26] and can be applied in the same framework.

Acknowledgement

The authors gratefully acknowledge the the financial support from the Norwegian Research Council and the industrial partners of the ICADA project (Ceetron, DNV GL and Statoil) and the FME NOWITECH (www.nowitech.no). They also acknowledge the support from the other co-workers in the IFEM software team.

Bibliography

[1] I. Akkerman et al. “The role of continuity in residual-based variational multiscale modeling of turbulence”. In: *Comput. Mech.* 41.3 (2008), pp. 371–378.

[2] D. N. Arnold, R. S. Falk, and R. Winther. “Finite element exterior calculus, homological techniques, and applications”. In: *Acta numerica* 15 (2006), pp. 1–155.

[3] Y. Bazilevs et al. “Isogeometric analysis using T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 229–263.

[4] Y. Bazilevs et al. “Isogeometric fluid-structure interaction: theory, algorithms, and computations”. In: *Comput. Mech.* 43.1 (2008), pp. 3–37.

[5] Y. Bazilevs et al. “Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes”. In: *Comput. Methods Appl. Mech. Engrg.* 199.13-16 (2010), pp. 780–790.

[6] P. Bornemann and F. Cirak. “A subdivision-based implementation of the hierarchical B-spline finite element method”. In: *Computer Methods in Applied Mechanics and Engineering* (2012).

[7] O. Botella and R. Peyret. “Benchmark spectral results on the lid-driven cavity flow”. In: *Computers & Fluids* 27.4 (1998), pp. 421–433.

[8] A. Buffa, C. de Falco, and G. Sangalli. “IsoGeometric Analysis: Stable elements for the 2D Stokes equation”. In: *International Journal for Numerical Methods in Fluids* 65.11-12 (2011), pp. 1407–1422.

[9] A. Buffa, G. Sangalli, and R. Vázquez. “Isogeometric analysis in electromagnetics: B-splines approximation”. In: *Comput. Methods Appl. Mech. Engrg.* 199.17-20 (2010), pp. 1143–1152.

[10] A. Buffa, G. Sangalli, and R. Vázquez. “Isogeometric methods for computational electromagnetics: B-spline and T-spline discretizations”. In: *J. Comput. Phys.* 257.part B (2014), pp. 1291–1320.

[11] A. Buffa et al. “Isogeometric discrete differential forms in three dimensions”. In: *SIAM J. Numer. Anal.* 49.2 (2011), pp. 818–844.

[12] K. Chang, T. J. R. Hughes, and V. M. Calo. “Isogeometric variational multiscale large-eddy simulation of fully-developed turbulent flow over a wavy wall”. In: *Comput. & Fluids* 68 (2012), pp. 94–104.

[13] T. Dokken, T. Lyche, and K. Pettersen. “Polynomial splines over locally refined box-partitions”. In: *Comput. Aided Geom. Des.* 30.3 (Mar. 2013), pp. 331–356.

[14] M. R. Dörfel, B. Jüttler, and B. Simeon. “Adaptive isogeometric analysis by local h-refinement with T-splines”. In: *Computer Methods in Applied Mechanics and Engineering* 199.5-8 (2010), pp. 264 –275.

[15] J. A. Evans and T. J. R. Hughes. “Isogeometric divergence-conforming B-splines for the Darcy-Stokes-Brinkman equations”. In: *Math. Models Methods Appl. Sci.* 23.4 (2013), pp. 671–741.

[16] J. A. Evans and T. J. R. Hughes. “Isogeometric divergence-conforming B-splines for the steady Navier-Stokes equations”. In: *Math. Models Methods Appl. Sci.* 23.8 (2013), pp. 1421–1478.

[17] J. A. Evans and T. J. R. Hughes. “Isogeometric divergence-conforming B-splines for the unsteady Navier-Stokes equations”. In: *J. Comput. Phys.* 241 (2013), pp. 141–161.

[18] J. Evans. “Divergence-free B-spline Discretizations for Viscous Incompressible Flows”. PhD thesis. The University of Texas at Austin, 2011.

[19] D. Forsey and R. Bartels. “Hierarchical B-spline refinement”. In: *ACM SIGGRAPH Computer Graphics* 22.4 (1988), pp. 205–212.

[20] C. Giannelli, B. Jüttler, and H. Speleers. “THB-splines: The Truncated basis for hierarchical splines”. In: *Computer Aided Geometric Design* 29.7 (2012), pp. 485 –498.

[21] T. Hughes, J. Cottrell, and Y. Bazilevs. “Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement”. In: *Computer Methods in Applied Mechanics and Engineering* 194.39-41 (2005), pp. 4135–4195.

[22] K. A. Johannessen, F. Remonato, and T. Kvamsdal. “On the similarities and differences between Classical Hierarchical, Truncated Hierarchical and LR B-splines”. In: *Submitted to Isogeometric Special Issue: Computer Methods in Applied Mechanics and Engineering* (2014).

[23] K. Johannessen, T. Kvamsdal, and T. Dokken. "Isogeometric analysis using LR B-splines". In: *Computer Methods in Applied Mechanics and Engineering* 269 (2014), pp. 471–514.

[24] R. Kraft. "Adaptive und linear unabhängige Multilevel B-Splines und ihre Anwendungen". PhD thesis. Stuttgart, 1998.

[25] M. Kumar, T. Kvamsdal, and K. Johannessen. "Superconvergent patch recovery and a posteriori error estimation technique in adaptive isogeometric analysis". In: *Submitted to Computer methods in applied mechanics and engineering* (2014).

[26] D. Mokriš, B. Jüttler, and C. Giannelli. "On the completeness of hierarchical tensor-product B-splines". In: *Journal of Computational and Applied Mathematics* 271.0 (2014), pp. 53–70.

[27] B. Mourrain. "On the dimension of spline spaces on planar T-meshes". In: *Math. Comp.* 83.286 (2014), pp. 847–871.

[28] N. Nguyen-Thanh et al. "Isogeometric analysis using polynomial splines over hierarchical T-meshes for two-dimensional elastic solids". In: *Computer Methods in Applied Mechanics and Engineering* 200.21–22 (2011), pp. 1892–1908.

[29] K. F. Pettersen. *On the dimension of multivariate spline spaces.* Tech. rep. Sintef ICT, 2013.

[30] A. Ratnani and E. Sonnendrücker. "An arbitrary high-order spline finite element solver for the time domain Maxwell equations". In: *J. Sci. Comput.* 51.1 (2012), pp. 87–106.

[31] D. Schillinger and E. Rank. "An unfitted hp-adaptive finite element method based on hierarchical B-splines for interface problems of complex geometry". In: *Computer Methods in Applied Mechanics and Engineering* 200.47 - 48 (2011), pp. 3358 –3380.

[32] D. Schillinger et al. "An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces". In: *Computer Methods in Applied Mechanics and Engineering* 249–252.0 (2012), pp. 116 –150.

[33] T. Sederberg et al. "T-spline simplification and local refinement". In: *ACM Transactions on Graphics* 23.3 (2004), pp. 276–283.

[34] T. Sederberg et al. "T-splines and T-NURCCs". In: *ACM Transactions on Graphics* 22.3 (2003), pp. 477–484.

[35] C. Verhoosel et al. “An isogeometric analysis approach to gradient damage models”. In: *International Journal for Numerical Methods in Engineering* 86.1 (2011), pp. 115–134.

[36] C. Verhoosel et al. “An isogeometric approach to cohesive zone modeling”. In: *International Journal for Numerical Methods in Engineering* 87.1-5 (2011), pp. 336–360.

[37] A. Vuong et al. “A hierarchical approach to adaptive local refinement in isogeometric analysis”. In: *Computer Methods in Applied Mechanics and Engineering* 200.49-52 (2011), pp. 3554–3567.

Appendix A

Software

1 Open source

The software used in the preparation of this work has been made publicly available as open source under GNU public licence v2. It can be downloaded for free at

https://github.com/VikingScientist/LRsplines.

The primary development language is c++, but a matlab wrapper has also been developed for convenience and is available at

https://github.com/VikingScientist/MatlabLR.

These two packages contain all software for LR B-spline refinement, evaluation and manipulation. We will here give a very brief introduction on how the code is structured. For complete technical documentation, we refer to the website

http://www.lrbsplines.com/

where all details are available.

At the core of the package is the refinement algorithm. We reiterate it here for reference.

2 Data structures

It is interesting to note that the refinement algorithm above only requires two primitives: B-splines and Meshlines. The former is characterized by its local knot vector Ξ and Ψ while the latter by its start- and end position in the parametric space.

Algorithm 7 Refinement algorithm

```
 1: Insert new line E
 2: for every B-spline $B_i \in \mathcal{S}$ do
 3:   if E traverse $B_i$ and E does not exist in $B_i$ then
 4:     split $B_i$
 5:   end if
 6: end for
 7: for every newly created B-spline $B_j$ from line 4 or 10 do
 8:   for every existing line $E \in \mathcal{M}$ do
 9:     if E traverse $B_j$ and E does not exist in $B_j$ then
10:       split $B_j$
11:     end if
12:   end for
13: end for
```

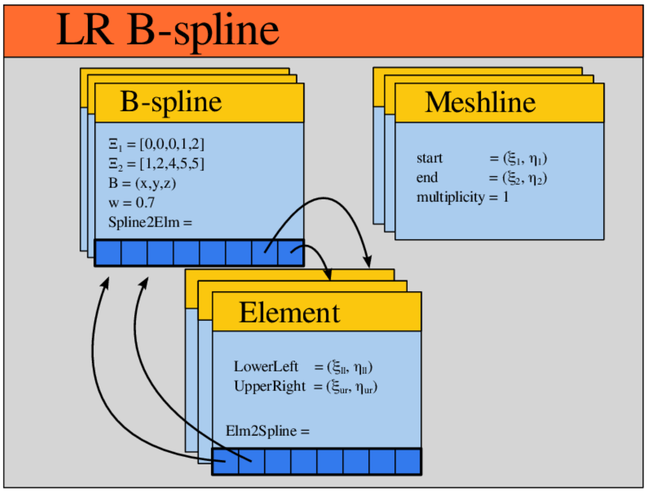


Figure 1: Data structures used to represent LR B-splines

This is the absolute minimal implementation possible for LR B-splines. Note that it is possible to derive the polynomial degree from the length of the local knot vectors, and that it is implicitly defined that the meshlines have multiplicity 1.

However it is customary to include additional information. We usually include a scalar weight corresponding to each B-spline to maintain the partition of unity. Moreover, in the case of a mapped geometry, we include the control points. To allow for reduced continuity, we add the meshline multiplicity.

This will generate a structure much like this

```
class BSpline {
  vector<double> knot_xi;     // local knot vectors
  vector<double> knot_psi;
  vector<double> controlpoint;// geometric mapping coefficients
  double         weight; // scalar weights for partition of unity
}

class Meshline {
  double start[2];       // starting point in parametric space
  double stop[2];        // ending point
  int    multiplicity;   // higher mult. reduces continuity
}

class LRSplineSurface {
  vector<Bspline>  functions; // all functions in our space
  vector<Meshline> lines;     // all lines in our mesh
}
```

Listing : **C++ class definitions:** a general idea into how the code is built up. Note that this is not the complete class definition as several containers are not `vector` and additional convenience data is often stored, but irrelevant to the global structure.

which is not far from the actual implementation. The containers used above are here listed as `std::vector`, but this is not ideal due to the fact that the refinement algorithm will quite frequently add or remove B-spline objects. What is proposed is to create a unique hash code from the B-spline knot vectors. This will serve as the B-spline signature and make it easy to check if it exists already, and allows for both fast insertion and removal by using a Hashset container instead of the vector. The exact nature of the hash function is based on heuristics.

When we are using this for finite element analysis, we quickly realize that it is convenient, although not necessary, to include the class "element" or knot span. We keep track of which B-splines have support on which element and will later use this for assembly of system matrices. The element is defined by its bounding box in the parametric space given by the lower left and upper right corner.

```
class Element {
  double lower_left[2];  // parametric bounding
  double upper_right[2]; // box of this element

  vector<Bspline> support; // we keep track of which functions
                           // have support on all elements
}
```

Listing : **C++ class definitions:** a general idea into how the code is built up. Note that this is not the complete class definition as several containers are not `vector` and additional convenience data is often stored, but irrelevant to the global structure.

The element class may be organized in different ways and it is believed that an oct-tree type container is advantageous, although at the time of publishing this is not yet implemented. Oct-tree allows for quick searching and also allows for fast updating when elements are split in two by new meshline insertions.

The complete overview of the data structures used is given in Figure 1.

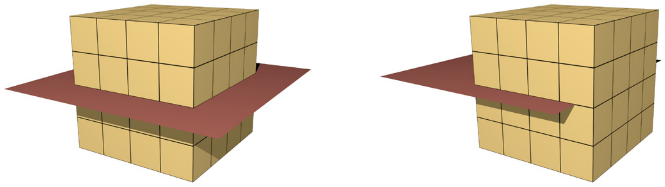
Appendix B

Moving to 3D: Trivariate LR B-splines

There is a substantially smaller set of literature on locally refined trivariate splines than there is on bivariate and several questions remain unanswered. For instance, the volumetric version of Mourrains dimension formula [27] has not been generalized to higher dimensions in a peer-reviewed journal. Nevertheless, the initial formulation of the LR B-splines by Dokken [13] does include the trivariate extension. Moreover, the data structures above is created in a "meshless" manner in the sense that topology information is not stored. This allows for far easier extensions to higher dimension. The trivariate LR B-splines are implemented in addition to the bivariate and published as open source in the core library.

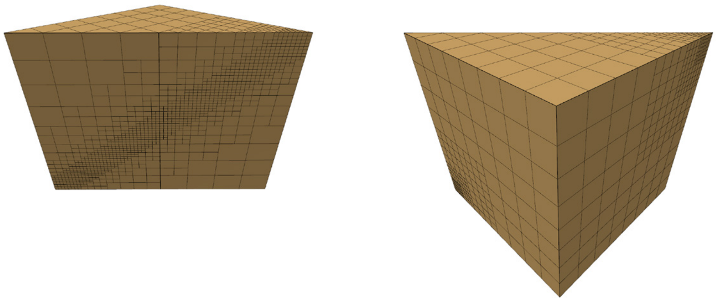
We note that the core of the refinement algorithm is rather simple. It consists of multiple times checking if a line traverse a rectangle. The extension of this into higher dimensions is straightforward. We let our functions, i.e. B-splines be defined by three local knot vectors Ξ, Ψ and Z and change our meshlines to be planar mesh rectangles, now defined by the two corner points. The beauty of this is that we only need to redefine "traverse" in algorithm 7 to work with mesh rectangles instead of meshlines. This is similar to the bivariate case in the sense that a mesh rectangle needs to cut the support, and all of the support, in order to provoke a splitting of the B-splines.

What is even more convenient is the fact that most algorithms carry over little to no changes. In particular the peeling algorithm which continuously "peels" away B-splines which cannot participate in a linear dependence relation, is only dependent on knowing which functions have support on any given element. In short, the most fundamental changes from an algorithmic point of view is the pre- and post-processing. That is retrieval of boundaries



(a) Mesh rectangle traversing B (b) Mesh rectangle not traversing B

Figure 1: Traversing the support of a basis function in 3D. The refinement algorithm simply checks the boxed support of the basis functions, and if these are traversed by planar mesh rectangles.



(a) Viewed from inside the cube (b) Viewed from outside the cube

Figure 2: 3D Diagonal refinement of a cube using bicubic basis functions. Here sliced at the plane $x = y$ to show the interior refinement. The actual geometry is the full cube.

for boundary conditions or visualizations.

Appendix C

On the potential function in compatible space discretization

In Paper IV, we construct compatible spaces and show that they form a complete de Rham complex. We collect these results in Theorem 2, 3 and 4. Seeing as these theorems are considered the main results of the paper, we would like to elaborate on a specific part. During the proof, we say that

$$\operatorname{im}(\mathbf{rot}) = \ker(\operatorname{div}) \tag{1}$$

for the sequence $\mathbb{R} \to X_h^0 \to X_h^1 \to X_h^2 \to 0$, $0 \to Y_h^0 \to Y_h^1 \to Y_h^2 \to 0$ and $0 \to Z_h^0 \to Z_h^1 \to Z_h^2 \to 0$.

This can be shown by proving that any potential field has a divergence-free velocity, and any divergence-free velocity has a potential field. Written more formally, we say that

$$\begin{aligned} \forall \varphi \in X_h^0 \quad \exists \quad \boldsymbol{u} \in X_h^1 : \mathbf{rot}(\varphi) = \boldsymbol{u} \wedge \operatorname{div}(\boldsymbol{u}) = 0 \qquad (2) \\ \forall \boldsymbol{u} \in X_h^1 : \operatorname{div}(\boldsymbol{u}) = 0 \quad \exists \quad \varphi \in X_h^0 : \mathbf{rot}(\varphi) = \boldsymbol{u} \qquad (3) \end{aligned}$$

and likewise for the spaces with boundary conditions, i.e. Y_h and Z_h.

1 Proving im(rot)⊆ker(div)

First note that any potential field maps to a divergence-free space under the **rot** operator

$$\operatorname{div}(\mathbf{rot}(\varphi)) = \operatorname{div}\left(\begin{bmatrix} \frac{\partial \varphi}{\partial y} \\ -\frac{\partial \varphi}{\partial x} \end{bmatrix}\right) = \frac{\partial^2 \varphi}{\partial x \partial y} - \frac{\partial^2 \varphi}{\partial x \partial y} = 0. \tag{4}$$

$$\mathbf{rot}(\varphi) = \begin{bmatrix} \frac{\partial \varphi}{\partial y} \\ -\frac{\partial \varphi}{\partial x} \end{bmatrix} = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \tag{5}$$

By considering spaces of boundary conditions, we will also need to check if these are satisfied under mapping. First consider a $\varphi \in Y_h^0$. As $\varphi = 0$ along the edges, we have that the tangential derivative along the edge is zero. That is $\frac{\partial \varphi}{\partial x} = -u_2 = 0$ on the top and bottom edge, while $\frac{\partial \varphi}{\partial y} = u_1 = 0$ on the left and right edge. Hence $\mathbf{rot}(\varphi) \in Y_h^1$.

For any $\varphi \in Z_h^0$ we have that both the normal and tangential derivative along the edge is zero, i.e. $\frac{\partial \varphi}{\partial x} = \frac{\partial \varphi}{\partial y} = 0$ on all edges. Hence $\boldsymbol{u} = \mathbf{0}$ on the boundary and $\mathbf{rot}(\varphi) \in Z_h^1$.

2 Proving im(rot)⊇ker(div)

Assume we have $\boldsymbol{u} \in X_h^0$, such that $\text{div}(\boldsymbol{u}) = 0$. We then want to construct a potential field φ such that $\mathbf{rot}(\varphi) = \boldsymbol{u}$. This can be formalized as

$$\begin{aligned} \frac{\partial \varphi}{\partial y} &= u_1(x,y) && (6) \\ \frac{\partial \varphi}{\partial x} &= -u_2(x,y). && (7) \end{aligned}$$

We integrate the first line to get

$$\varphi(x,y) = \int_a^y u_1(x,t)\ dt + C(x) \tag{8}$$

which differentiated with x becomes

$$\frac{\partial \varphi}{\partial x} = \int_a^y \partial_x u_1(x,t)\ dt + \partial_x C(x). \tag{9}$$

Setting in (7) yields

$$\begin{aligned} -u_2(x,y) &= \int_a^y \partial_x u_1(x,t)\ dt + \partial_x C(x) \\ \partial_x C(x) &= -u_2(x,y) - \int_a^y \partial_x u_1(x,t)\ dt \\ \partial_x C(x) &= -u_2(x,y) - \int_a^y \partial_x u_1(x,t) + \partial_y u_2(x,t) - \partial_y u_2(x,t)\ dt \\ \partial_x C(x) &= -u_2(x,y) + \int_a^y \partial_y u_2(x,t)\ dt \\ \partial_x C(x) &= -u_2(x,y) + u_2(x,y) - u_2(x,a) \\ \partial_x C(x) &= -u_2(x,a) \\ C(x) &= -\int_b^x u_2(y,a)\ dt + c && (10) \end{aligned}$$

where we have used that $\operatorname{div}(\boldsymbol{u}(x,t)) = \partial_x u_1(x,t) + \partial_y u_2(x,t) = 0$. By setting the constants all to zero, i.e. $a = b = c = 0$ and combining (8) and (10) we arrive at the potential field presented in the paper

$$\varphi(x,y) = \int_0^y u_1(x,t)\,dt - \int_0^x u_2(t,0)\,dt \tag{11}$$

and it can be checked that this satisfies $\mathbf{rot}(\varphi) = \boldsymbol{u}$.

2.1 The no penetration spaces Y_h

The very same function will work under spaces with boundary conditions as well, but may be simplified. Note that both Y_h^1 and Z_h^1 consists of functions which second component vanishes at the bottom edge, i.e. $u_2(t,0) = 0$. For these spaces it is enough to consider the potential function

$$\varphi(x,y) = \int_0^y u_1(x,t)\,dt. \tag{12}$$

We show that this potential satisfies all boundary conditions, since we have from the derivations above that $\mathbf{rot}(\varphi) = \boldsymbol{u}$. Assuming the unit domain $\Omega = [0,1]^2$ we have for any $\boldsymbol{u} \in Y_h^1$

$$\begin{aligned} u_1(0,y) &= 0 \\ u_1(1,y) &= 0 \\ u_2(x,0) &= 0 \\ u_2(x,1) &= 0 \end{aligned}$$

which in turn gives

$$\begin{aligned} \varphi(0,y) &= \int_0^y u_1(0,t)\,dt = 0 \\ \varphi(1,y) &= \int_0^y u_1(1,t)\,dt = 0 \\ \varphi(x,0) &= \int_0^0 u_1(x,t)\,dt = 0 \\ \varphi(x,1) &= \int_0^1 u_1(x,t)\,dt. \end{aligned}$$

The first two integrals equals to zero due to the boundary conditions on $\boldsymbol{u}$, while the third term is zero due to the integration range is zero. The final term is also zero, but the reason is a little more subtle. For any given x, consider the closed line integral containing the subdomain $D = [0, x] \times [0, 1]$, i.e.

$$\begin{aligned}\oint \boldsymbol{u} \cdot \hat{\boldsymbol{n}}\, dS \quad = \quad & \int_0^1 u_1(x,t)\, dt + \\ & \int_x^0 u_2(t,1)\, dt + \\ & \int_1^0 -u_1(0,t)\, dt + \\ & \int_0^x -u_2(t,0)\, dt.\end{aligned}$$

Of the four right hand side integrals, only the first one does not immediately vanish as the three other are along boundary curves and have zero contribution due to $\boldsymbol{u} \in Y_h^0$. By using the divergence theorem we have that the left hand side integral is zero

$$\oint \boldsymbol{u} \cdot \hat{\boldsymbol{n}}\, dS = \iint_D \operatorname{div}(\boldsymbol{u})\, dA = 0.$$

We are left with $\int_0^1 u_1(x,t)\, dt = 0$ which proves that $\varphi(x,y) = 0$ on all edges and hence $\varphi \in Y_h^0$.

2.2 The no slip spaces Z_h

The no slip spaces are conceptually no different than the no penetration spaces, but we include them here for completeness. The boundary conditions on Z_h^1 state

$$\begin{aligned} u_1(0,y) = \quad & u_2(0,y) & = 0 \\ u_1(1,y) = \quad & u_2(1,y) & = 0 \\ u_1(x,0) = \quad & u_2(x,0) & = 0 \\ u_1(x,1) = \quad & u_2(x,1) & = 0 \end{aligned}$$

and we need to show that the generated φ satisfies the boundary conditions on Z_h^0 which states

$$\begin{aligned} \varphi(0,y) = \quad & \tfrac{\partial \varphi}{\partial x}(0,y) & = 0 \\ \varphi(1,y) = \quad & \tfrac{\partial \varphi}{\partial x}(1,y) & = 0 \\ \varphi(x,0) = \quad & \tfrac{\partial \varphi}{\partial y}(x,0) & = 0 \\ \varphi(x,1) = \quad & \tfrac{\partial \varphi}{\partial y}(x,1) & = 0. \end{aligned}$$

The conditions that $\varphi = 0$ on the boundary is analog to the previous section, and we now show that the normal derivative is also zero. We first remember that $\frac{\partial\varphi}{\partial x} = -u_2(x, y)$ and $\frac{\partial\varphi}{\partial y} = u_1(x, y)$, which means

$$\begin{aligned}
\frac{\partial\varphi}{\partial x}(0, y) = & \quad -u_2(0, y) & = 0 \\
\frac{\partial\varphi}{\partial x}(1, y) = & \quad -u_2(1, y) & = 0 \\
\frac{\partial\varphi}{\partial y}(x, 0) = & \quad u_1(x, 0) & = 0 \\
\frac{\partial\varphi}{\partial y}(x, 1) = & \quad u_1(x, 1) & = 0.
\end{aligned}$$

We conclude that if we have a $\boldsymbol{u} \in Z_h^1$, we may create a $\varphi \in Z_h^0$ which will satisfy all the boundary conditions in Z_h^0, and also $\boldsymbol{u} = \mathbf{rot}(\varphi)$.