ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/387798167

Comparison of Integration Methods for Cut Elements

Preprint · January 2025

DOI: 10.48550/arXiv.2501.03854

CITATIONS
0

READS
7

3 authors, including:



Guilherme Henrique Teixeira
Graz University of Technology
11 PUBLICATIONS **0** CITATIONS
SEE PROFILE



Michael Loibl
University of the Bundeswehr Munich
4 PUBLICATIONS **8** CITATIONS
SEE PROFILE

All content following this page was uploaded by Guilherme Henrique Teixeira on 08 January 2025.

The user has requested enhancement of the downloaded file.

arXiv:2501.03854v1 [cs.CE] 7 Jan 2025

COMPARISON OF INTEGRATION METHODS FOR CUT ELEMENTS

G. H. TEIXEIRA[1], M. LOIBL[2], B. MARUSSIG[1]

[1] Institute of Applied Mechanics
Graz University of Technology,
Technikerstraße 4/II, 8010, Graz, Austria
e-mail: teixeira@tugraz.at

[2] Institute of Mechanics and Statics,
University of the Bundeswehr Munich.
Werner-Heisenberg-Weg 39, 85577, Neubiberg, Germany
e-mail: michael.loibl@unibw.de

Key words: Integration, Cut Elements, IsoGeometric Analysis

Summary.

Using an interface inserted in a background mesh is an alternative way of constructing a complex geometrical shape with a relative low meshing efforts. However, this process may require special treatment of elements cut by the interface. Our study focuses on comparing the integration of cut elements defined by implicit and parametric curves. We investigate the efficiency and robustness of open-source tools such as Algoim [5](a library for quadrature on implicitly defined geometries) and Ginkgo [2](a library for isogeometric analysis on Boolean operations with a parametric description) with numerical examples computing the area defined by the interface and benchmarks for 2D elasticity problem using the open-source code GeoPDEs [7]. It is concluded that none of the two interface descriptions is preferable with respect to the quality of the integration. Thus, the choice of the interface type depends only on the studied problem and the available curve description, but not on the numerical aspects of the integration.

1 INTRODUCTION

Modelling the geometry and discretising the domain of a problem with a mesh is an important step in the process of solving any boundary value problem. Regarding the construction of a geometric representation, a common way to do it is to define a parametric shape and map this geometry into the physical domain. However, finding a suitable parametrization for complex geometries is in general not an easy task. One way to mitigate this problem is to define the geometry through a background mesh, where an interface specifies a region to be visible. This approach allows us to analyse any geometry by simply selecting a part of the background mesh to define the domain, as can be seen in the scheme presented in Figure 1. This concept is present in methods such as the Immersed Boundary Method [8] or the Finite Cell Method [6].

In the process of selecting the domain, some elements are cut by this interface. In consequence, among several challenges, the numerical integration of cut elements is an important point to be verified [4]. In particular, we would like to investigate, how the quadrature points are distributed over the cut elements, and we would like to observe how this is affected by the type of the

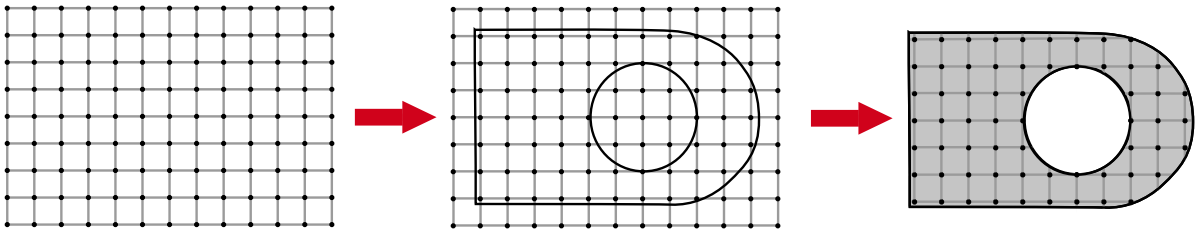

Figure 1: Scheme of domain definition with an interface

interface used. Thereby, we compare parametric and implicit interface descriptions for cut elements. Depending on the type of interface an appropriate quadrature scheme is applied to define the points and weights in each cut element. In particular, in this work, for parametric curve interfaces, a reparametrisation of the element is performed and the quadrature is obtained as in [2]. Whereas for implicit curve interfaces, an approximation of the curve by a Bézier curve is carried out and the corresponding points and weights are defined as in [5].

2 INTERFACE REPRESENTATIONS

2.1 Parametric curves

A parametric curve is defined by the map from the parameter domain to the physical domain, where each point in the parameter domain has its own image in the physical domain. Bézier curves, B-splines, and NURBS are examples of parametric curves. In the comparisons presented here, NURBS curves $C(\xi)$ are used to create the interface, which can be defined by the linear combination of NURBS basis functions $R_{j,p}$ of degree p scaled by control points $\mathbf{c}_i$

$$\mathbf{C}(\xi) = \sum_{j=1}^{n} R_{j,p}(\xi)\mathbf{c}_i. \tag{1}$$

At least one edge of a cut element is defined by this interface. And for such elements where the interface is defined by a NURBS curve or portion of it, we employ the open-source code Ginkgo [2] to compute the quadrature points and weights. In this approach, the first step to obtain the quadrature points is defined by the reparametrisation of the element cut by the interface. However, sometimes the reparametrisation of an element with only one B-spline surface is not possible. So, the cut element is divided into several tiles, where each of them can be reconstructed using B-spline surfaces. Once this is done, the quadrature points from a reference element are mapped to each one of these small parts of the reparametrised element.

2.2 Implicit curves

An implicit curve is defined by a locus of points for which an equation takes a specific value. The points with the same value form isolines and, for convenience, the zero set is usually chosen as the isoline that implicitly describes the curve. These points are defined to fulfil the condition

$$f(x, y) = 0. \tag{2}$$

The use of implicit curves to define the domain can be applied to solve topology optimisation problems via the Level-set Method [1]. In order to compute the integral in implicitly defined

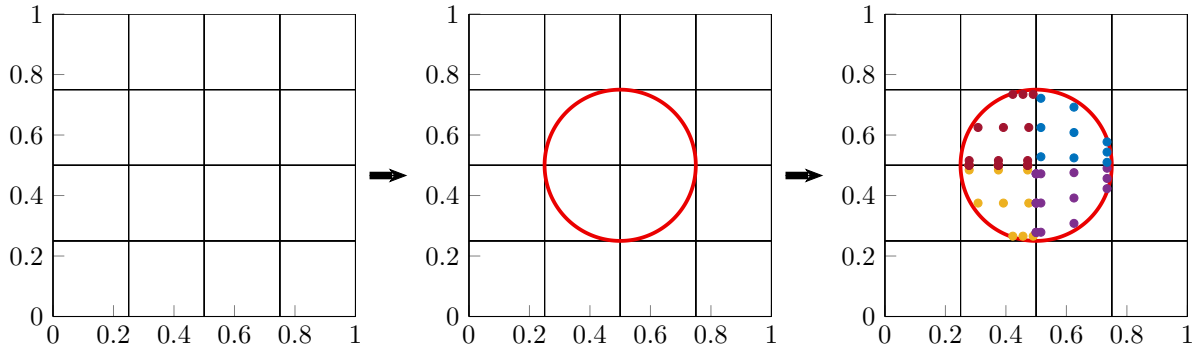

Figure 2: Procedure for obtaining the quadrature points

domains, we apply the open-source code Algoim [5] to determine quadrature points and weights. In this approach, the implicit curve is approximated by a Bézier curve and then the algorithm is used to obtain the quadrature rule for domains implicit defined by multivariate polynomials. The quadrature points and weights are obtained by dividing the domain in pieces where in each one of them the appropriate quadrature scheme is applied for one-dimensional integrals [5].

3 METHODOLOGY

The process of defining the quadrature rule on the cut elements begins with the definition of a background mesh and is followed by the insertion of an interface that creates the domain of interest. The difference between the approaches considered here is related to how these interfaces are defined for an identical geometry. Once the interface has been defined, the elements are identified as active and non-active. The non-active elements are neglected in the computation and the active elements are divided into two subgroups cut and uncut elements. For the uncut elements, a Gaussian quadrature is defined in a reference element and then mapped to the physical element, while for the cut elements the appropriate approach is applied depending on the type of interface. Figure 2 describes the steps to obtain the quadrature rule. In a pre-existing background mesh, an interface defines the domain where, depending on the type of interface, the quadrature points are provided using the appropriate open-source code (Algoim [5] or Ginkgo [2]), which in sequence allows the numerical integration to be performed by simply evaluating the function at the quadrature points and scaling by the corresponding quadrature weights. In order to compare the different tools, first we consider the following verification settings:

- Computation of the area enclosed by an interface for different refinement levels; and
- computation of the area enclosed by an interface for a fixed refinement level while the geometry moves.

With these two settings we can verify not only the accuracy but also the robustness of the two approaches comparing the relative error with respect to the size of mesh and the relative error with respect to a given step. Once the behaviour of the tools is observed, we investigate how the definition of the interface could affect the solution of 2D elasticity problems, where a background mesh is defined using B-splines. For this, the open-source code for Isogeometric Analysis GeoPDEs [7] is used. In this situation, two different scenarios of cut curves are set

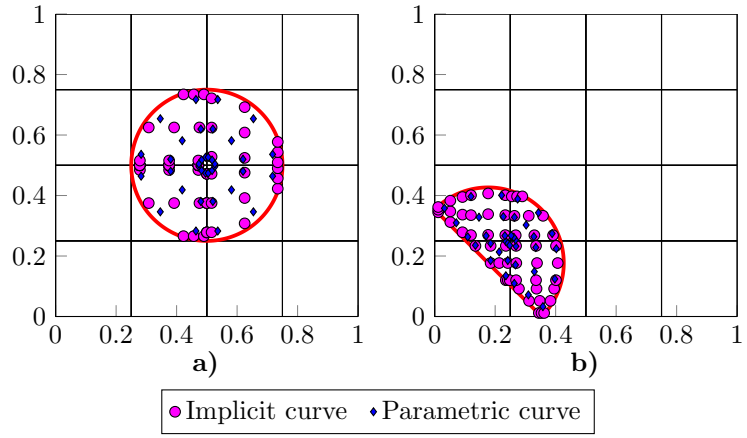

Figure 3: Quadrature points: a) circle example; b) semicircle example

up, one interface with no boundary conditions applied to it and one interface with boundary conditions applied to it.

4 NUMERICAL RESULTS

4.1 Area computation

Our first comparison study considers simply the computation of the area enclosed by an interface. In this test, we verify the accuracy by computing the relative error for a fixed interface while refining the background mesh. The first example is defined by a circle with radius $R = 0.2$ and centered at $(0.5, 0.5)$. In this example, the geometry is fixed using a single interface and the mesh is refined by h-refinement. Figure 3a shows the shape and quadrature points for a mesh of size $h = 0.25$, which results in 36 quadrature points for both interface types. While the relative

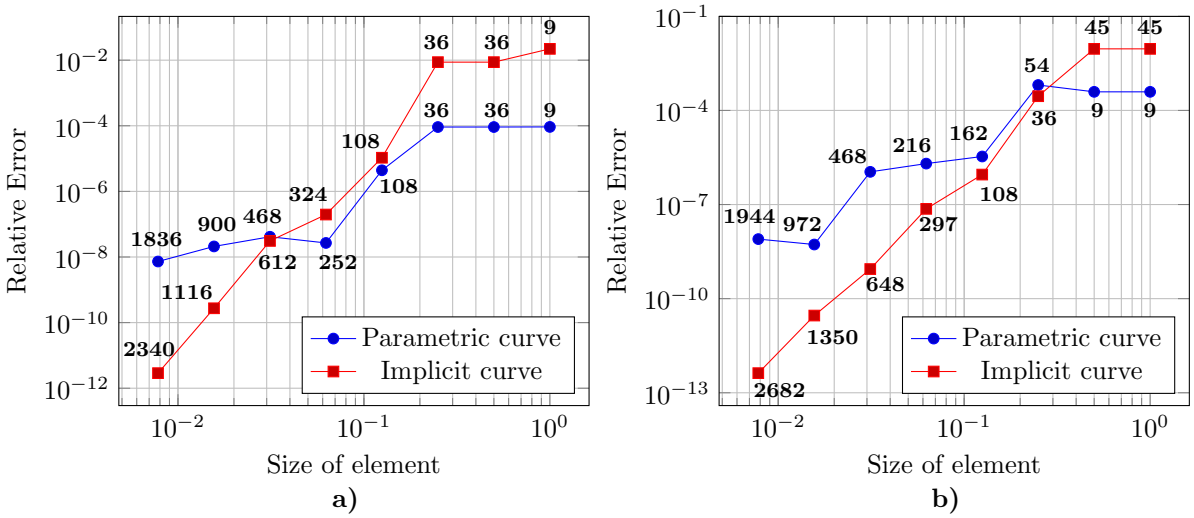

Figure 4: Convergence study: a) circle example; b) semicircle example

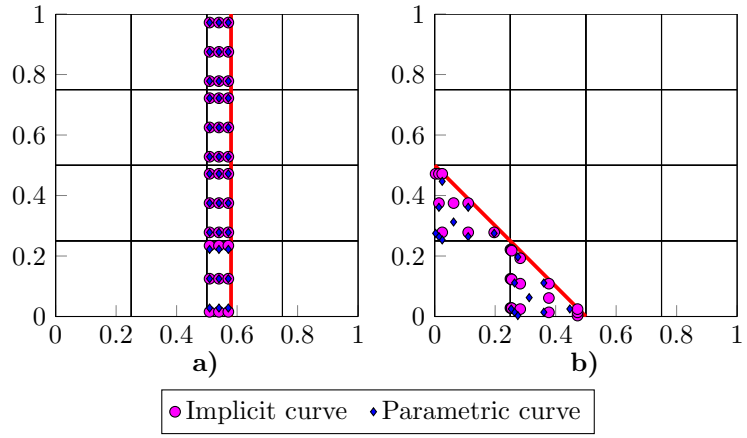

Figure 5: Quatrature points: a) Line ; b) Triangle

error with respect to the size of the elements is shown in Figure 4a. The second example is a semicircle, which is built using two interfaces. In particular, the interface using implicit curves is defined by the intersection of a straight line $y = -x + 0.3536$ with a circle with radius $R = 0.25$ and centered at (0.1768,0.1768). The mesh is refined by h-refinement and Figure 3b shows the quadrature points obtained for the same element size as the circle example. Figure 4b shows the relative error and the number of quadrature points needed for each size of mesh.

4.2 Moving interfaces

Another aspect verified in this part of the work is the robustness of the approaches. For this, we look at different situations where it might be difficult to deal with the integration. This is done by setting a mesh of size $h = 0.25$ and moving an interface around, trying to find positions or situations where the codes are crashing. We investigate two situations: The first one is defined

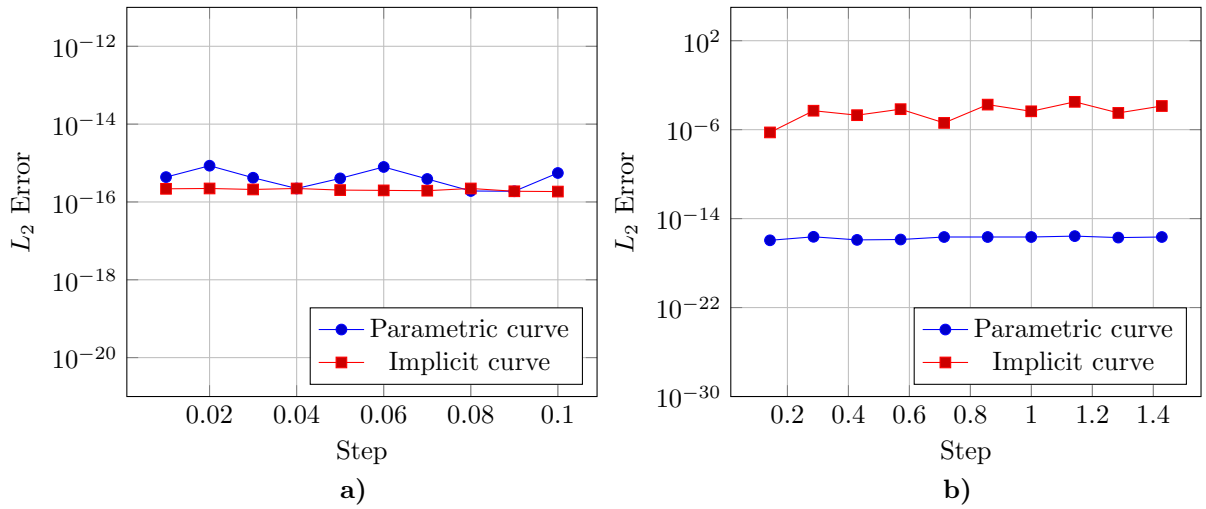

Figure 6: Robustness study: a) line; b) triangle

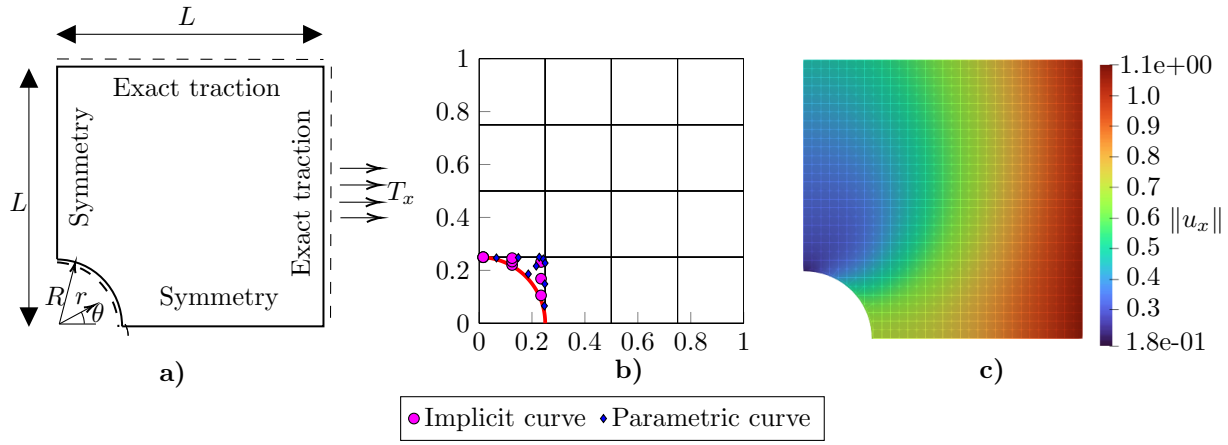

Figure 7: Infinite plate with a hole: a) problem scheme; b) quadrature points; c) displacement magnitude

by a straight vertical line moving along the horizontal direction, as shown in Figure 5a. While the second is given by a triangle in which the node $(0, 0.5)$ rotates clockwise with respect to the origin, while the nodes at $(0, 0)$ and $(0.5, 0)$ remain fixed, as shown in Figure 5b. Figure 6 shows the relative error with respect to the given steps for both examples. For the line example, the step represents the distance with respect to the first position of the line, which is at $x = 0.5$, and for the triangle example, the step represent the angle between the line that connects the origin to the moving node and its original position, the line that connects the origin to the node $(0, 0.5)$. As it is possible to see, none of the two approaches fails for any of the investigated steps.

4.3 Elasticity benchmarks

The 2D elasticity problems are solved using the open-source code for Isogeometric Analysis GeoPDEs [7] with quadrature points and weights provided by Algoim [5] and Ginkgo [2] .The geometries are defined by the intersection between a B-spline background mesh and one of the two types of an interface. And it is considered $E = 1$ and $\nu = 0.3$.

4.3.1 Infinite plate with a hole

The first example is the well-known benchmark of an infinite plate with a hole subjected to a traction $T_x = 10$ at infinity, which can be modeled by a quarter of the geometry due to symmetry. To do this, the background mesh $\Omega = [0, 1] \times [0, 1]$ is cut by a circle with radius $R = 0.25$, as shown in Figure 7a, and exact traction is applied at the top and right side of the mesh, while symmetry boundary conditions are applied at the bottom and left side of the mesh. The analytical solution [3] is given as

$$u_x = T_x \cdot \frac{R_i}{8\mu}\left[\frac{r}{R_i}A + 2\frac{R_i}{r}(A+B) - 2\frac{R_i^3}{r^3}B\right] \tag{3}$$

$$u_y = T_x \cdot \frac{R_i}{8\mu}\left[\frac{r}{R_i}C + 2\frac{R_i}{r}(D+E) - 2\frac{R_i^3}{r^3}E\right], \tag{4}$$

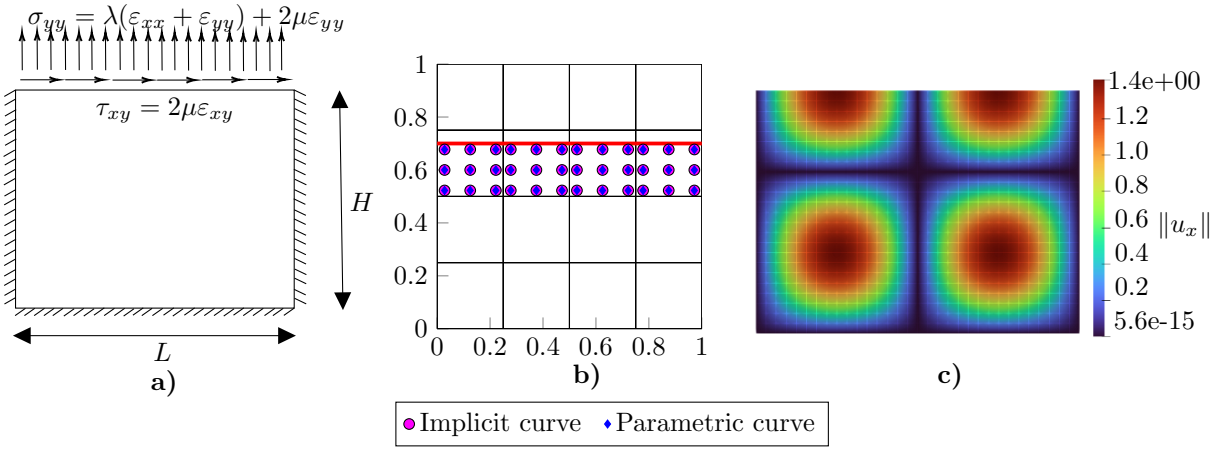

Figure 8: Plane square: a) problem scheme; b) quadrature points; c) displacement magnitude

where

$$A = (k+1)\cos(\theta); B = \cos(3\theta); C = (k-3)\sin(\theta)$$
$$D = (1-k)\sin(\theta); E = \sin(3\theta); k = (3-4\nu); \mu = \frac{E}{2(1+\nu)}.$$

Figure 7b shows the position of the quadrature points on the cut element, obtained with the tool Algoim [5], when the interface is defined by an implicit curve, and by the tool Ginkgo [2], when the interface is defined by a parametric curve. In Figure 7c, the magnitude of the component u_x of the displacement vector $\mathbf{u} = [u_x, u_y]^T$ is shown. The graph in Figure 9a shows the comparison of the error with the size of the element, from which it is possible to see that solving the problem with an element trimmed by an implicit curve gives as good results as when solving the problem defining by a parametric interface. In both approaches we see almost the same accuracy.

4.3.2 Square plate

In this example, we considered a square background mesh $\Omega = [0,1] \times [0,1]$ trimmed at the top, in the position $y = 0.75$, by a horizontal straight line, as shown in Figure 8a and 8b, which also shows the quadrature points obtained from both approaches and from the respective open-source codes (Algoim [5] or Ginkgo [2]). The boundary conditions and the source term are manufactured in a such way that the exact solution is given by

$$u_x = \sin(2\pi x)\sin(2\pi y) \tag{5}$$
$$u_y = \sin(2\pi x)\sin(2\pi y) \tag{6}$$

Figure 8c shows the magnitude of the component u_x of the displacement field obtained from this problem. Figure 9b presents the graphs of convergence for different element sizes. Both approaches give similar accuracy, which is the same behavior identified on the previous example of an infinite plate with a hole. Hence, we can conclude that both approaches are well-suited for numerical approximation schemes.

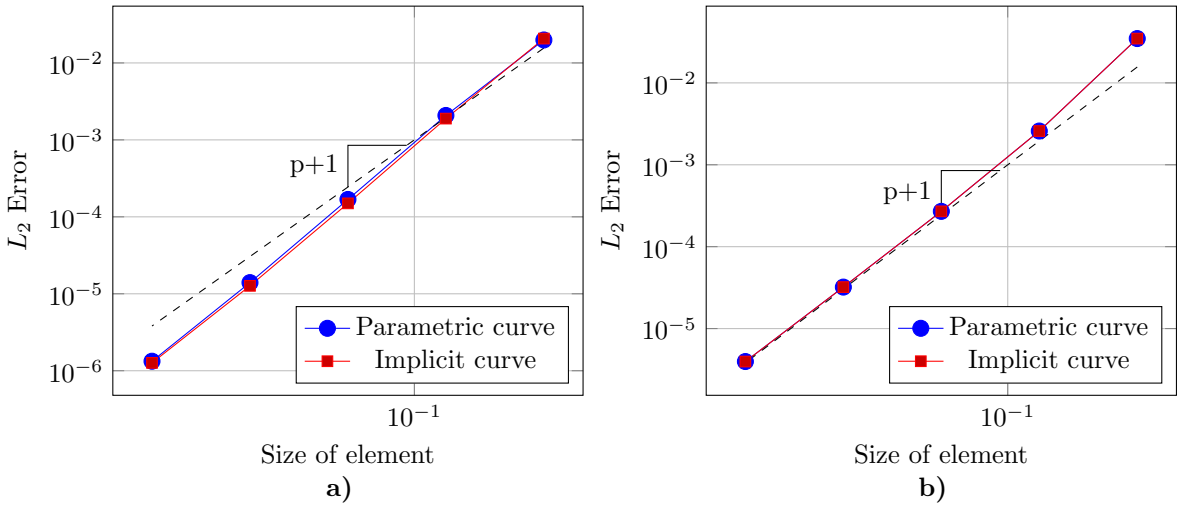

Figure 9: Convergence study for p=2: a) infinite plate with a hole; b) square plate

5 CONCLUSIONS

In this work, we compare two situations of cut elements that occur due to the intersection of embedded interfaces in a background mesh. The two approaches differ in the interface definition, that is, either using parametric or implicit curves. We employ the open-source codes Algoim [5] and Ginkgo [2] to provide the quadrature points and weights over the cut elements. Numerical examples of area computation are defined, from which we can observe similar behaviour in terms of convergence and the number of quadrature points used. Furthermore, using the open-source code GeoPDEs, we apply both approaches to solve 2D elasticity problems with a background mesh defined by B-splines. In this situation, we can compare the behaviour of these procedures not only when the integrand is equal to 1 (area computation problem) but also to observe the effects of the definition of the interface in the solution of a different problem (2D elasticity). In all examples, we observe almost the same accuracy. Therefore, the examples show similar answers for both approaches, which yield that the choice between the type of interface depends mainly on the application of the problem, the advantage of using one rather than the other, and the interface definition available but not on the numerical aspects of the integration.

ACKNOWLEDGEMENTS

The work of Guilherme Henrique Teixeira is supported by the joint DFG/FWF Collaborative Research Centre CREATOR (CRC – TRR361/F90) at TU Darmstadt, TU Graz and JKU Linz/Johann Radon Institute for Computational and Applied Mathematics, Linz.

REFERENCES

[1] G. Allaire, F. Jouve, and A. Toader. A level-set method for shape optimization. *Comptes Rendus. Mathématique*, 334:1125–1130, 2002. doi: 10.1016/S1631-073X(02)02412-3.

[2] P. Antolin, A. Buffa, and M. Martinelli. Isogeometric analysis on v-reps: First results. *Computer Methods in Applied Mechanics and Engineering*, 355:976–1002, 2019. doi: 10.

1016/j.cma.2019.07.015.

[3] L. Coradello. *Accurate isogeometric methods for trimmed shell structures.* PhD thesis, École polytechnique federale de Lausanne, 2021.

[4] B. Marussig and T. J.R. Hughes. A review of trimming in isogeometric analysis: Challenges, data exchange and simulation aspects. *Archives of Computational Methods in Engineering*, 25:1059–1127, 2018. doi: 10.1007/s11831-017-9220-9.

[5] R. I. Saye. High-order quadrature on multi-component domains implicitly defined by multivariate polynomials. *Journal of Computational Physics*, 448, 2022. doi: 10.1016/j.jcp.2021.110720.

[6] D. Schillinger and M. Ruess. The finite cell method: A review in the context of higher-order structural analysis of cad and image-based geometric models. *Archives of Computational Methods in Engineering*, 22:391–455, 2015. doi: 10.1007/s11831-014-9115-y.

[7] R. Vázquez. A new design for the implementation of isogeometric analysis in octave and matlab: Geopdes 3.0. *Computers and Mathematics with Applications*, 72:523–554, 2016. doi: 10.1016/j.camwa.2016.05.010.

[8] X. Wei, B. Marussig, P. Antolin, and A. Buffa. Immersed boundary-conformal isogeometric method for linear elliptic problems. *Computational Mechanics*, 68:1385–1405, 2021. doi: 10.1007/s00466-021-02074-6.

View publication stats