Mollified finite element approximants of arbitrary order and smoothness

Eky Febrianto[a], Michael Ortiz[b], Fehmi Cirak[a,*]

[a]Department of Engineering, University of Cambridge, Cambridge, CB2 1PZ, UK
[b]Graduate Aerospace Laboratories, California Institute of Technology, Pasadena, CA 91125, USA

Abstract

The approximation properties of the finite element method can often be substantially improved by choosing smooth high-order basis functions. It is extremely difficult to devise such basis functions for partitions consisting of arbitrarily shaped polytopes. We propose the mollified basis functions of arbitrary order and smoothness for partitions consisting of convex polytopes. On each polytope an independent local polynomial approximant of arbitrary order is assumed. The basis functions are defined as the convolutions of the local approximants with a mollifier. The mollifier is chosen to be smooth, to have a compact support and a unit volume. The approximation properties of the obtained basis functions are governed by the local polynomial approximation order and mollifier smoothness. The convolution integrals are evaluated numerically first by computing the boolean intersection between the mollifier and the polytope and then applying the divergence theorem to reduce the dimension of the integrals. The support of a basis function is given as the Minkowski sum of the respective polytope and the mollifier. The breakpoints of the basis functions, i.e. locations with non-infinite smoothness, are not necessarily aligned with polytope boundaries. Furthermore, the basis functions are not boundary interpolating so that we apply boundary conditions with the non-symmetric Nitsche method as in immersed/embedded finite elements. The presented numerical examples confirm the optimal convergence of the proposed approximation scheme for Poisson and elasticity problems.

Keywords: finite elements, polytopic elements, mollifier, convolution, Voronoi diagrams

*Corresponding author
Email address:* `f.cirak@eng.cam.ac.uk` (Fehmi Cirak)

1. Introduction

Smooth high-order finite element approximants are often more efficient and, in general, integrate better with prevalent computer-aided geometric design (CAGD) descriptions [1–3]. The construction of mesh-based smooth high-order approximants is currently an active area of research as partly motivated by recent academic and industrial interest in isogeometric analysis. In most mesh-based approaches such approximants are defined as the tensor-products of univariate approximants. Such constructions do not generalise to unstructured meshes and auxiliary techniques are needed in the vicinity of the so-called extraordinary vertices where the tensor-product structure breaks. In CAGD a range of ingenious constructions has been conceived to generate smooth high-order approximants around the extraordinary vertices, see the monographs [4, 5] for an overview. Unfortunately, most of these constructions, including [6–13], target bivariate manifolds and do not generalise to the arbitrary variate case. Indeed, there are currently no sufficiently flexible and intuitive non-tensor-product arbitrary-variate constructions that can yield smooth polynomial high-order basis functions. By contrast, the proposed mollified approximation scheme over polytopic partitions is easy to construct, is polynomial and can have arbitrary order and smoothness.

Convolutional techniques are widely used in the analysis and numerics of partial differential equations. The convolution of a function with a mollifier, i.e. a kernel with a unit volume, yields a function that is smoother than the mollifier and the original function. This smoothing property is, for instance, used to recursively define uniform B-splines [14], to analyse non-smooth functions and partial differential equations [15–17], to postprocess finite element solutions [18–20] or to regularise optimisation problems [21–23]. Indeed, some of the classical meshless methods, like the smoothed particle hydrodynamics (SPH) [24, 25] and the reproducing kernel particle method (RPKM) [26], are defined via convolutions, see also the reviews [27–30]. In SPH and RPKM mollifiers are usually referred to as window, weight or influence functions. Different from our mollified approximation scheme, SPH is intrinsically restricted to low order approximants and the RPKM yields high-order and arbitrarily smooth approximants which are rational. The kernels derived in RPKM depend on the local node distribution and are determined so that they can exactly reproduce a polynomial of a given order. Although RPKM was conceived as a meshless method, it is possible to define its mesh-based cousins [31] and to blend it consistently with mesh-based B-spline basis functions [32, 33].

In the proposed mollified approximation scheme each non-overlapping polytopic cell has an independent local polynomial approximant of prescribed degree q^p. The local approximants are discontinuous across cell boundaries. The convolution of a local approximant with a C^k mollifier yields a smoother C^{k+1} approximant. The chosen mollifiers are compactly supported symmetric polynomials and have a unit volume. It is clear that the convolution of a global polynomial $f(\boldsymbol{x})$ of degree q^p with the chosen mollifiers gives a polynomial $\widehat{f}(\boldsymbol{x}) \neq f(\boldsymbol{x})$. However, it is straightforward to find a polynomial $g(\boldsymbol{x})$ of degree q^p such that $\widehat{g}(\boldsymbol{x}) \equiv f(\boldsymbol{x})$. This implies that the mollified approximants can exactly reproduce any global function $f(\boldsymbol{x})$ of degree q^p. Mollified basis functions for finite element analysis are defined by convolving the local approximant of each cell individually. At a given evaluation point the basis functions
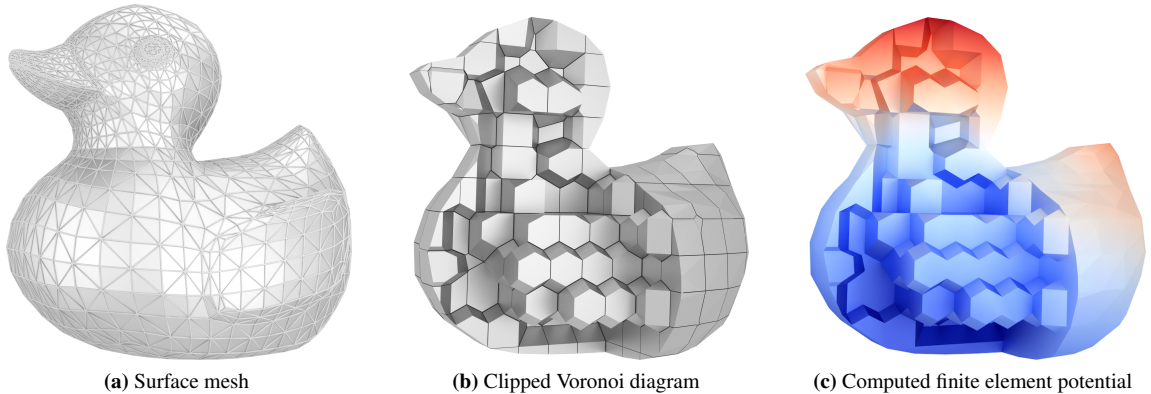

| (a) Surface mesh | (b) Clipped Voronoi diagram | (c) Computed finite element potential |

Figure 1: Illustrative three-dimensional finite element computation using mollified basis functions. The domain boundary is described with the triangular mesh in (a) and the domain is partitioned with the Voronoi tessellation in (b). The solution of a Poisson problem is shown in (c). Note that in (b) and (c) the cells intersected by the domain boundary have been clipped and others omitted for visualisation purposes.

are evaluated first by computing the intersection between the support of the mollifier and the cell. Subsequently, the convolution integral over the resulting intersection polytope is evaluated numerically, but exactly (up to round-off error). We apply the divergence theorem to reduce the dimension of the integrals, but other methods of integrating polynomials over polytopes can be used, see e.g. [34, 35]. The obtained basis functions consist of several C^{k+1} continuously joined polynomial pieces. It is worth emphasising that we use, in contrast to RPKM, a fixed kernel which does not depend on the local node distribution and its convolution with a polynomial $f(x)$ of degree $q^p > 0$ is not required to yield the same polynomial, i.e. $\widehat{f}(x) \neq f(x)$. Irrespectively, polynomials $f(x)$ of degree q^p are included in the space spanned by the mollified approximants and can be exactly reproduced.

The derived mollified basis functions can be used as usual in the finite element discretisation of partial differential equations. For ease and efficiency of implementation we assume that each of the polytopic cells representing a finite element is convex. The required geometric operations, like the intersection computations, are significantly simplified by the convexity assumption. We partition the problem domain into a set of convex polytopic cells using a Voronoi diagram, see Figure 1. The evaluation of the finite element integrals requires some care because the C^{k+1} continuous breakpoints of the basis functions are not aligned with cell boundaries. We use to this end the variationally consistent integration approach proposed in [29], which significantly reduces the number of required integration points. The required support of the mollified basis functions is the Minkowski sum of the mollifier support with the respective cell [36]. Furthermore, the present version of the mollified basis functions are non-boundary-interpolating so that the Dirichlet boundary conditions are applied weakly with the non-symmetric Nitsche method [37–40] as in immersed/embedded finite element methods, see e.g. [41].

The outline of this paper is as follows. In Section 2 we briefly review the convolution of univariate polynomials with a mollifier and characterise the properties of the resulting mollified polynomials. Subsequently we derive in Sections 3.1 and 3.2 first the univariate and then the multivariate mollified basis functions. The key difference between the two cases lies in the evaluation of the convolution integrals. In the univariate case the integrals are evaluated analytically and in the multivariate case numerically. The use of the derived mollified basis functions in finite element analysis, especially the integration and treatment of boundary conditions, is discussed in Section 4. Finally, in Section 5 we introduce several Poisson and elasticity examples to confirm the optimal convergence of the developed approach. The paper is supplemented by four appendices which provide convergence estimates and discuss implementation details.

2. Preliminaries

We consider the one-dimensional domain $\Omega \in \mathbb{R}^1$ partitioned into a set of n_c non-overlapping segments $\{\Omega_i\}$, referred to as cells, such that

$$\Omega = \bigcup_{i=1}^{n_c} \Omega_i \,. \tag{1}$$

On each cell Ω_i a compactly supported local polynomial is defined,

$$f_i(x) = \begin{cases} \boldsymbol{p}_i(x) \cdot \boldsymbol{\alpha}_i & \text{if } x \in \Omega_i \,, \\ 0 & \text{if } x \notin \Omega_i \end{cases} \,, \tag{2}$$

where the vector $\boldsymbol{p}_i(x)$ represents a polynomial basis of degree q^p and $\boldsymbol{\alpha}_i$ are its polynomial coefficients. See Figure 2 for an illustrative example. We choose in each cell the same polynomial basis $\boldsymbol{p}_i(x)$, although it is possible to change the type of basis and its polynomial degree. The sum of the local polynomials defined over the entire domain Ω is given by

$$f(x) = \sum_i \boldsymbol{p}_i(x) \cdot \boldsymbol{\alpha}_i \,. \tag{3}$$

Evidently, across the cell boundaries this function can be discontinuous, i.e. $f(x) \in C^{-1}$.

3

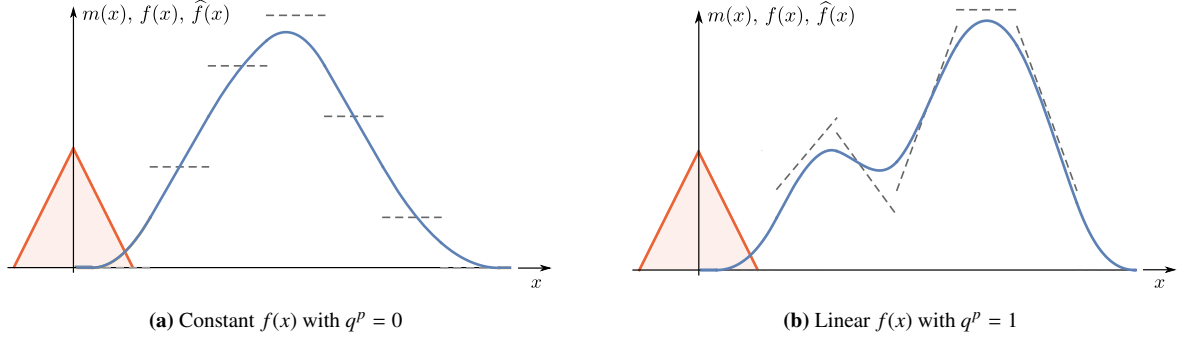**(a)** Constant $f(x)$ with $q^p = 0$ **(b)** Linear $f(x)$ with $q^p = 1$

Figure 2: Mollification of piecewise discontinuous functions $f(x)$ (black, dashed) with a linear mollifier $m(x)$ (red, solid). The resulting mollified functions $\widehat{f}(x)$ (blue, solid) are C^1 continuous.

The smoothness of $f(x)$ is increased by convolving it with a mollifier $m(x)$. The mollifier is chosen such that it has the following properties

$$m(x) \geq 0 \quad \forall x \in \Omega \tag{4a}$$

$$\operatorname{supp} m(x) = (-h_m/2, h_m/2) \tag{4b}$$

$$\int_\Omega m(x)\, \mathrm{d}x = 1 . \tag{4c}$$

That is, the mollifier is positive, has a unit volume and a finite support of size h_m. In addition we require that the mollifier is symmetric, i.e. $m(x) = m(-x)$, and that it has certain smoothness properties, as yet to be specified. The mollification of $f(x)$ is defined with the convolution

$$\widehat{f}(x) = m(x) * f(x) = \int_\Omega m(x - y) f(y)\, \mathrm{d}y = \int_\Omega m(y) f(x - y)\, \mathrm{d}y . \tag{5}$$

The equality of both integrals can be shown by a simple substitution. We usually use the first integral expression in the following. Furthermore, we choose polynomial mollifiers $m(x)$ of degree q^m and $f(x)$ is, as stated above, of degree q^p. The mollified function $\widehat{f}(x)$ has monomials up to degree $q^m + q^p + 1$.

If the derivative of the mollifier $m(x)$ exist, the derivative of the mollified function $\widehat{f}(x)$ is given by

$$\frac{\mathrm{d}}{\mathrm{d}x} \widehat{f}(x) = \int_\Omega \frac{\mathrm{d}m(x - y)}{\mathrm{d}x} f(y)\, \mathrm{d}y . \tag{6}$$

The higher order derivatives are computed similarly. Considering that $f(x) \in C^{-1}$ is discontinuous and the mollifier is $m(x) \in C^k$ we can deduce for the mollified smooth function $\widehat{f}(x) \in C^{k+1}$.

Finally, the cell-wise definition of $f(x)$ introduced in (5) and (6) yields

$$\widehat{f}(x) = \sum_i \alpha_i \cdot \int_{\Omega_i} m(x - y) p_i(y)\, \mathrm{d}y \tag{7}$$

and

$$\frac{\mathrm{d}}{\mathrm{d}x} \widehat{f}(x) = \sum_i \alpha_i \cdot \int_{\Omega_i} \frac{\mathrm{d}m(x - y)}{\mathrm{d}x} p_i(y)\, \mathrm{d}y . \tag{8}$$

The function $\widehat{f}(x)$ is composed of infinitely smooth polynomial pieces that are smoothly connected, i.e. C^{k+1}, across a finite number of breakpoints. In general the location of the breakpoints does not coincide with the cell

4

boundaries. However, as is known from B-splines, on uniformly partitioned domains, it is possible to choose the support size of the mollifier such that the breakpoints fall on the cell boundaries [14, 42]. In the illustrative example in Figure 2 the influence of the choice of the local polynomial $f(x)$ on the mollified function $\widehat{f}(x)$ is demonstrated. In this example the convolution integral (7) has been evaluated analytically.

A final remark concerns the reproduction of polynomials with mollified functions. It can be shown that it is possible to find for a given polynomial $f(x)$ of degree q^p a polynomial $g(x)$ of the same degree which yields after mollification $\widehat{g}(x) = m(x) * g(x) \equiv f(x)$. Specifically, the mollification of a polynomial

$$g(x) = \alpha_0 + \alpha_1 x + \alpha_2 x^2 + \alpha_3 x^3 + \dots \tag{9}$$

is given by

$$\widehat{g}(x) = \alpha_0 + \alpha_1 x + \alpha_2(x^2 + m_2) + \alpha_3(x^3 + 3xm_2) + \dots , \tag{10}$$

where m_s are the moments of the mollifier defined as

$$m_s = \int_\Omega m(x)x^s \, \mathrm{d}x . \tag{11}$$

After rearranging the terms in (10) according to the powers of x it is easy to see how to choose a function $g(x)$ which is after mollification equal to the given function $f(x)$. This implies that the polynomials of degree q^p are included in the space spanned after mollification.

3. Smooth piecewise basis functions

We now use the mollification approach to derive basis functions on one- and multi-dimensional domains. Again the domain $\Omega \in \mathbb{R}^d$ is assumed to be partitioned into a set of non-overlapping convex polytopes $\{\Omega_i\}$, which are in the present paper obtained from a Voronoi diagram. In the following we refer to the polytopes as cells. The mollification approach yields a set of basis functions for each cell. The convolution integrals for obtaining the basis functions are evaluated analytically in the one-dimensional, i.e. univariate, case with $d = 1$ and numerically in the multi-dimensional, i.e. multivariate, case with $d \geq 2$. Convergence estimates for the obtained mollified basis are provided in Appendix A.

3.1. Univariate basis functions

To set the stage for multivariate basis functions, we first consider the derivation of univariate basis functions. The mollified basis functions belonging to a cell Ω_i are defined according to the mollification (7) by

$$f^m(x) = \sum_i \boldsymbol{\alpha}_i \cdot \boldsymbol{N}_i(x) , \tag{12}$$

where the vector of mollified basis functions $\boldsymbol{N}_i(x)$ are defined with

$$\boldsymbol{N}_i(x) = \int_{\Omega_i} m(x - y)\boldsymbol{p}_i(y) \, \mathrm{d}y . \tag{13}$$

Note that the local polynomial basis is outside the cell Ω_i zero and the integration domain is restricted to Ω_i. As a local polynomial basis $\boldsymbol{p}_i(y)$ different basis choices are possible, such as the monomial, Lagrange or Bernstein. While this choice has no influence on the approximation quality of the resulting mollified basis, it affects the interpretability of the coefficients $\boldsymbol{\alpha}_i$ and the conditioning of the resulting finite element system matrices. In our examples we use in each cell Ω_i the scaled and shifted monomial basis
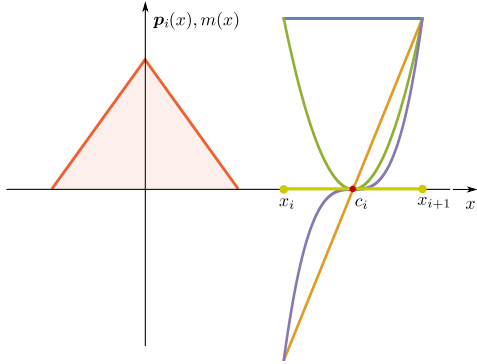
$$\boldsymbol{p}_i(x) = \begin{pmatrix} 1 & x & x^2 & \dots & x^{q^p} \end{pmatrix} \quad \text{with } x = \frac{2(x - c_i)}{h} , \tag{14}$$
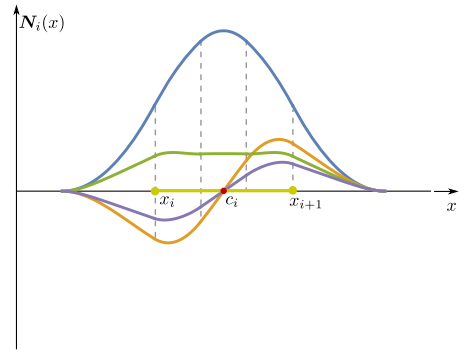
5
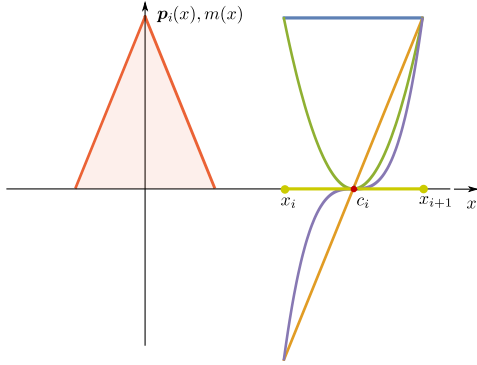
(a) Mollifier (narrow) and local monomial basis
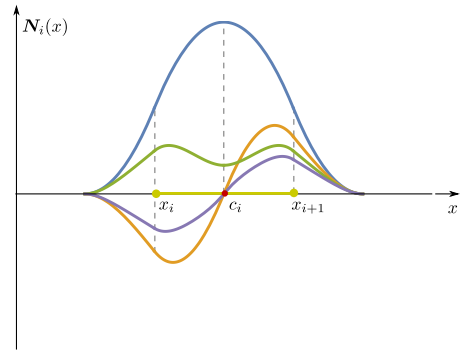
(b) Basis functions obtained with the mollifier in (a)

(c) Mollifier (wide) and local monomial basis

(d) Basis functions obtained with the mollifier in (b)

Figure 3: Univariate mollified basis functions with a bilinear mollifier ($q^m = 1$) and a cubic scaled and shifted monomial basis ($q^p = 3$) on a cell $\Omega_i = (x_{i+1}, x_i)$. The mollified basis functions in (b) are obtained with the narrow mollifier in (a) and the ones in (d) with the wide mollifier in (c). The dashed lines in (b) and (d) indicate the C^1 continuous breakpoints of the obtained basis functions.

where q^p is its degree, c_i is the centre of the cell and h is the average length of all the cells $\{\Omega_i\}$ in the domain. The scaling by $2/h$ ensures that the obtained mollified basis functions have a similar maximum value. It is possible to apply a different scaling factor and to choose the scaling in each cell differently.

For any given point $x \in \Omega$ in the domain the mollified basis functions $N_i(x)$ are evaluated by computing the convolution integral (13). Evidently, when the chosen local polynomial basis $p_i(x)$ is a monomial basis the mollified basis functions are simply the moments of the mollifier. The derivatives of the mollified basis functions are computed according to (6). In Figure 3 the mollified basis functions for a cubic monomial basis with $q^p = 3$ and a piecewise linear mollifier $q^m = 1$ with two different support sizes h_m are shown. In each case the support size of the mollified basis functions is $h_m + h_{c,i}$, with the mollifier size h_m and the cell size $h_{c,i} = x_{i+1} - x_i$. The basis functions consist of several polynomial pieces that are C^1 continuously connected at the breakpoints. The number of breakpoints in each cell depends on the number and arrangement of the breakpoints in the mollifier. Note, although not shown in the figure, the basis functions of the neighbouring cells are non-zero in the considered cell as well. The breakpoints of those neighbouring basis functions may not coincide with the breakpoints of the shown basis functions.

3.2. Multivariate basis functions

Without loss of generality we focus in the following on trivariate basis functions. As in the univariate case the basis functions for a cell Ω_i are given by

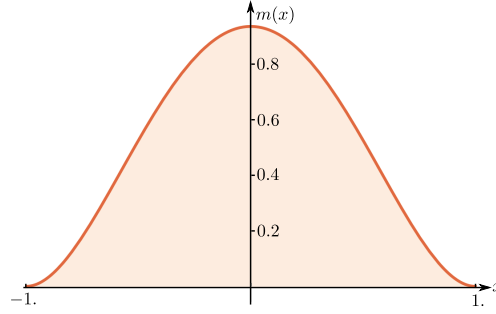$$N_i(x) = \int_{\Omega_i} m(x - y) p_i(y) \, dy. \tag{15}$$

Figure 4: C^1-continuous quartic spline mollifier with the support size $h_m = 2$.

The vector $\boldsymbol{p}_i(\boldsymbol{x})$ contains the scaled and shifted multivariate monomial basis of degree q^p. In this paper, the mollifier

$$m(\boldsymbol{x}) = m(x^{(1)}) \cdot m(x^{(2)}) \cdot m(x^{(3)}) \tag{16}$$

is composed of C^1-continuous quartic splines

$$m(x) = \begin{cases} \frac{15}{8h_m}\left(1 - 8\left(\frac{x}{h_m}\right)^2 + 16\left(\frac{x}{h_m}\right)^4\right) & \text{if } |x| < h_m/2 \\ 0 & \text{if } |x| \geq -h_m/2 \end{cases} \; ; \tag{17}$$

see Figure 4. It is easy to verify that $m(-h_m/2) = m(h_m/2) = 0$ and $\mathrm{d}m(-h_m/2)/\mathrm{d}x = \mathrm{d}m(h_m/2)/\mathrm{d}x = 0$. Hence, the mollifier is C^1 continuous. The continuity of the mollifier can be increased by forcing more derivatives to be zero at $x = -h_m/2$ and $x = h_m/2$, which can be achieved either by choosing a higher order polynomial, using a non-polynomial function or introducing more breakpoints. Obviously all approaches increase the cost of evaluating the convolution integrals.

The availability of efficient evaluation techniques for the multi-dimensional convolution integral (15) are vital to the proposed approach. We note that the integrand is a polynomial and that it is only non-zero on the intersection, i.e. boolean intersection, of the support of the mollifier and the considered cell, i.e.,

$$\omega_i := \square_{\boldsymbol{x}} \cap \Omega_i \,, \tag{18}$$

where $\square_{\boldsymbol{x}} = \operatorname{supp} m(\boldsymbol{x} - \boldsymbol{y})$ denotes the support of the mollifier centred at the evaluation point $\boldsymbol{x}$, see Figure 5. The intersection domain ω_i is convex because both $\square_{\boldsymbol{x}}$ and Ω_i are convex. Computing the intersection of polytopes and the integration of polynomials over polytopes are recurring tasks in computer graphics and many robust and efficient algorithms and implementations are available. In Appendix C we introduce one such algorithm for determining the intersection between a cell and a mollifier. One possible approach to evaluate the integrals on ω_i is first to tesselate it and then to integrate over the obtained simplices using Gaussian quadrature. Considering that the integrands are polynomials and the tessellation consists of affinely mapped simplices, the integration can be performed exactly (up to round-off error) using a sufficient number of quadrature points. A more elegant approach is to reduce the domain integrals to line integrals by repeated application of the divergence theorem, see e.g. [43]. We briefly sketch the conversion of volume integrals to surface integrals for completeness. The polytope ω_i consists of a set of uniquely oriented faces $\{\gamma_{i,j}\}$, i.e. all the respective normals $\boldsymbol{n}_{i,j}$ point outside the domain. To integrate an arbitrary polynomial $f(\boldsymbol{x})$ it is first integrated, e.g. in the $x^{(1)}$ direction,

$$\tilde{f}(\boldsymbol{x}) = \int f(\boldsymbol{x}) \, \mathrm{d}x^{(1)} \,. \tag{19}$$

The divergence theorem applied to this new function yields

$$f(\boldsymbol{x}) = \int_{\omega_i} \nabla \cdot \begin{pmatrix} \tilde{f}(\boldsymbol{x}) \\ 0 \\ 0 \end{pmatrix} \mathrm{d}\omega_i = \sum_j n_{i,j}^{(1)} \int_{\gamma_{i,j}} \tilde{f}(\boldsymbol{x}) \, \mathrm{d}\gamma_{i,j} \,, \tag{20}$$

7

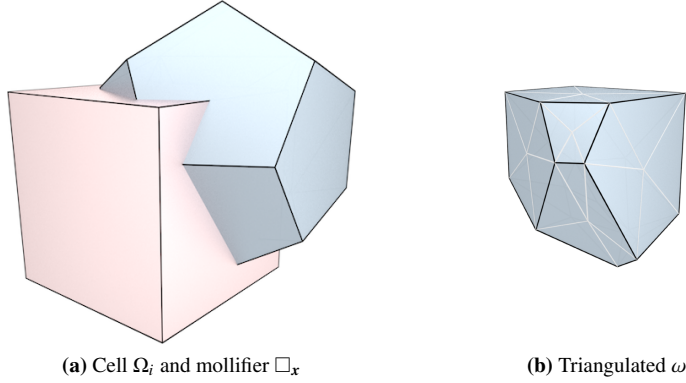**(a)** Cell Ω_i and mollifier $\square_x$ **(b)** Triangulated ω_i

Figure 5: Evaluation of mollified basis functions $N_i(x)$ by numerically computing the convolution integral (15) at a point $x \in \Omega$. (a) The support of the mollifier $\square_x$ centred at x (in red) and the cell (in blue). (b) The integration domain for the convolution integral is the boolean difference $\omega_i = \square_x \cap \Omega_i$.
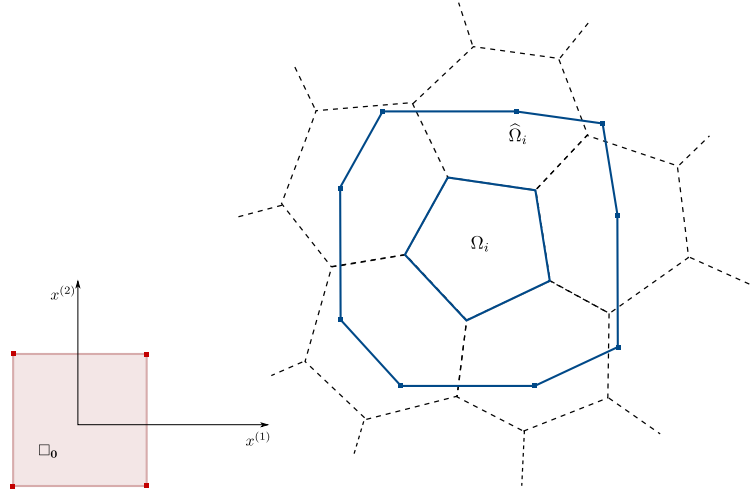

Figure 6: A cutout from a Voronoi tesselation and the support of the mollifier. The mollified basis functions $N_i(x)$ of the highlighted cell Ω_i are supported on the larger polygonal domain $\widehat{\Omega}_i = \text{supp } N_i(x)$. The polygonal domain is the Minkowski sum, i.e. $\widehat{\Omega}_i = \Omega_i \oplus \square_0$, of the cell with the mollifier support $\square_0$.

where the surface normal $n_{i,j}$ of the face $\gamma_{i,j}$ is constant. It is possible to stop at this point and to numerically evaluate the surface integrals after triangulating the faces $\gamma_{i,j}$. The number of quadrature points on each face is chosen to integrate exactly polynomials of degree $q^m + q^p$. However, it is also possible to reapply the divergence theorem to reduce the surface integrals to line integrals, which can then be analytically evaluated. In our implementation we evaluate (20) numerically by triangulating the faces $\gamma_{i,j}$, as indicated in Figure 5b. Note that the change from volume to surface integrals already yields a significant reduction in the number of integration points.

As in the univariate case, the support of a mollified basis function $N_i(x)$ is larger than its respective cell. In finite element computations also the support of a basis function is required, which is given by the Minkowski addition

$$\widehat{\Omega}_i = \Omega_i \oplus \square_0 = \{y + z \,|\, y \in \square_0, \; z \in \Omega_i\}. \tag{21}$$

In Appendix D we introduce the algorithm used for computing the Minkowski sum of two convex polyhedra.

8

4. Finite element discretisation with mollified basis functions

The smoothness and approximation properties of the mollified basis functions make them ideal for finite element analysis. In the following, we briefly outline the discretisation of a Poisson equation using the mollified basis functions. As in the previous sections we assume that a partitioning of the domain consisting of convex polytopes is given. We generate such a partitioning using a Voronoi diagram from a given implicit, i.e. level set, or parametric, i.e. surface mesh, description of the domain boundary. A set of points is placed within and outside the domain to generate the Voronoi diagram. The points outside the domain ensure that the mollifier's support is fully covered by local polynomials when the mollifier is placed on the domain boundary. The Voronoi diagram and the respective mollified basis functions do not conform to the domain boundaries. Therefore, the boundary conditions are applied weakly on cells cut by the boundary, as in immersed or embedded finite elements.

Notwithstanding this, it appears to be possible to continuously shrink the mollifier support when approaching the boundaries from the inside. In the limit on the boundary the mollifier becomes a Dirac delta and the interpolation within the domain becomes independent of the outside of the domain. This idea has, however, not been further pursued in this paper in order to focus on other aspects of the method.

The Poisson equation on a domain Ω is given by

$$
\begin{aligned}
-\nabla \cdot \nabla u &= s && \text{in } \Omega \\
u &= \bar{u} && \text{on } \Gamma_D \\
\boldsymbol{n} \cdot \nabla u &= \bar{t} && \text{on } \Gamma_N
\end{aligned}
\tag{22}
$$

where $\bar{u}$ is the prescribed solution field on the Dirichlet boundary Γ_D and $\bar{t}$ is the prescribed flux on the Neumann boundary Γ_N with the outward normal $\boldsymbol{n}$. The weak formulation of the Poisson equation can be stated according to Nitsche [37] as: Find $u \in H^1(\Omega)$ such that

$$
\underbrace{\int_\Omega \nabla u \cdot \nabla v \, d\Omega}_{a(u,v)} = \underbrace{\int_\Omega s v \, d\Omega + \int_{\Gamma_N} \bar{t} v \, d\Gamma}_{b(v)} \underbrace{- \gamma \int_{\Gamma_D} (u - \bar{u}) v \, d\Gamma}_{\gamma p(u,v)} + \underbrace{\int_{\Gamma_D} \left((u - \bar{u}) \boldsymbol{n} \cdot \nabla v + (\boldsymbol{n} \cdot \nabla u) v \right) d\Gamma}_{l(u,v)} .
\tag{23}
$$

The stabilisation parameter can be set to $\gamma = 0$ when the sign of the last term $(\boldsymbol{n} \cdot \nabla u)v$ is reversed, as proposed in a number of papers [38–40]. In our computations we use this so-called non-symmetric Nitsche method which has a non-symmetrical system matrix. The trial and test functions are discretised with the mollified basis functions

$$
u^h(\boldsymbol{x}) = \sum_{i=1}^{n_c} N_i(\boldsymbol{x}) \cdot \boldsymbol{\alpha}_i \quad \text{and} \quad v^h(\boldsymbol{x}) = \sum_{j=1}^{n_c} N_j(\boldsymbol{x}) \cdot \boldsymbol{\beta}_j \,,
\tag{24}
$$

where n_c is the number of the polytopic cells in the mesh.

Introducing the interpolation equations (24) into the weak form (23) yields a linear system of equations with the unknowns $\boldsymbol{\alpha}_i$, which are the coefficients of the local monomial bases in the cells. For instance, the bilinear form $a(u^h, v^h)$ becomes after discretisation

$$
a(u^h, v^h) = \sum_{i=1}^{n_c} \sum_{j=1}^{n_c} \boldsymbol{\alpha}_i^\top \int_\Omega \nabla N_i^\top \cdot \nabla N_j^\top \, d\Omega \, \boldsymbol{\beta}_j \,.
\tag{25}
$$

As usual, the domain integral is evaluated numerically after splitting it into cell contributions

$$
a(u^h, v^h) = \sum_{k=1}^{n_c} \left(\sum_{i=1}^{n_c} \sum_{j=1}^{n_c} \boldsymbol{\alpha}_i^\top \int_{\Omega_k} \nabla N_i^\top \cdot \nabla N_j^\top \, d\Omega_k \, \boldsymbol{\beta}_j \right) .
\tag{26}
$$

We evaluate the integral over a cell Ω_k by first decomposing it into tetrahedra and then applying standard Gauss integration. A cell is tetrahedralised by introducing additional nodes at its centre and face centres. At each integration point the mollified basis functions are evaluated as described in Section 3. Although the sketched integration of the

weak form (26) is straightforward, Gauss integration unfortunately requires too many quadrature points because of the breakpoints of the basis functions within the cells. Using too few quadrature points usually leads to suboptimal convergence rates.

The principal difficulties encountered in efficient and accurate integration of the weak form (26) are very similar to those encountered in meshless methods. In the variationally consistent integration techniques for meshless methods the test functions are modified to satisfy a consistency condition when integrated numerically, see e.g. [44, 45]. If the basis functions can reproduce a solution $u(x)$ of polynomial degree q^p, the finite element scheme must be able exactly to solve such a problem even when the integrals are evaluated numerically. Assuming a problem with the solution of polynomial degree q^p and inserting it into the discretised weak form yields for each cell a consistency condition for integration. To satisfy this consistency condition in a cell j the gradient of the test functions is modified as

$$\nabla \widetilde{N}_j^{\mathsf{T}}(x) = \nabla N_j^{\mathsf{T}}(x) + \Lambda_j \psi_j(x) \otimes \mathbf{1}\,, \tag{27}$$

where $\mathbf{1}$ is a vector of all ones and the vector $\psi_j(x)$ contains a monomial basis of degree $q^p - 1$ with the yet to be determined matrix of coefficients Λ_j. The support of the basis functions $\psi_j(x)$ is chosen to be same as of the basis functions $N_j(x)$. The matrix of coefficients Λ_j is determined by solving per basis function one small linear equation system so that variational consistency condition is satisfied. This equation system contains the integrals of the basis functions over their supports, which is given by the Minkowski sum of the mollifier support and the cell. See [44] for further details.

In the cells cut by the domain boundary the element integrals are evaluated only over parts of the cell which lie inside the domain. The respective integration domains are obtained by clipping the cells, see Appendix B. The resulting polyhedron is tetrahedralised with the same approach used for a non-clipped cell. As in immersed, or embedded domain, methods the faces of the tetrahedra can be projected to the curved domain boundaries if higher-order boundary approximation is needed, see e.g. [46–48].

5. Examples

We introduce in this section several examples of increasing complexity to experimentally verify the convergence of the proposed mollified finite element approach. In the one-dimensional problems both the convolution and finite element integrals are evaluated analytically, whereas in multi-dimensional problems both integrals are evaluated numerically. In multi-dimensional problems only the quartic spline mollifier (17) consisting of a single polynomial with no internal breakpoints is used. The mollified basis functions contain monomials of up to degree $q^p + q^m + 1$, although they are only complete up to degree q^p, and are non-zero over several cells. Therefore, it is not obvious how many quadrature points to choose in each of the integration triangles used for evaluating the finite element integrals. In our present computations we determine a stable number of quadrature points by successively increasing the number of quadrature points until we have a stable solution. In two dimensional problems we choose for linear basis functions ($q^p = 1$) three integration points for domain integrals and five for boundary integrals, and for quadratic ($q^p = 2$) we choose four and five integration points respectively. In all problems the Dirichlet boundary conditions are enforced with the parameter-free non-symmetric Nitsche method.

5.1. One-dimensional Poisson problem

As a first example we consider the solution of the one-dimensional Poisson-Dirichlet problem $-\mathrm{d}^2 u / \mathrm{d}x^2 = s$ on the domain $\Omega = (0, 1)$. The source term $s(x)$ is chosen such that the solution is equal to

$$u(x) = \sin(3\pi x). \tag{28}$$

The initial coarse mesh consisting of $n_c = 6$ cells is chosen to be non-uniform. The cell sizes, starting from the left, are $h_{c,1} = 0.15$, $h_{c,2} = 0.2$, $h_{c,3} = 0.15$, $h_{c,4} = 0.15$, $h_{c,5} = 0.2$ and $h_{c,6} = 0.15$. In addition to these cells, each domain boundary is padded with an extra ghost cell to ensure that the obtained mollified basis functions have the same approximation properties over the entire domain. We obtain finer meshes by repeated bisectioning of all cells.
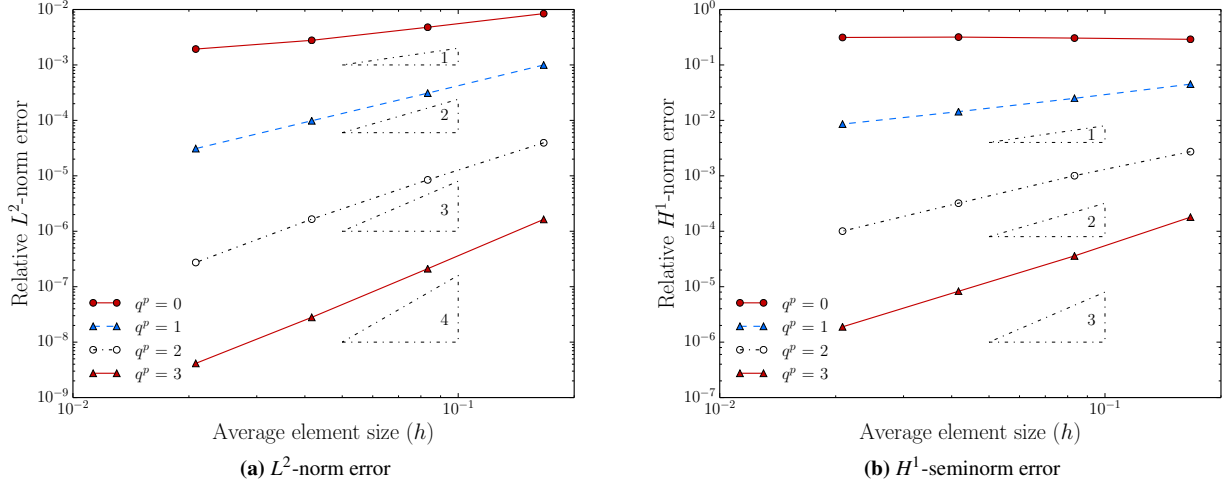
(a) L^2-norm error

(b) H^1-seminorm error

Figure 7: One-dimensional Poisson problem. Convergence with normalised linear B-spline mollifier and local polynomial basis of degrees $q^p \in \{0, 1, 2, 3\}$.

In the following set of experiments we study the influence of the choice of the local polynomial basis $\boldsymbol{p}_i(x)$ and the mollifier $m(x)$ on the convergence of the finite element solution. Firstly, we take in turn different local polynomials of degrees $q^p \in \{0, 1, 2, 3\}$ and a normalised linear B-spline, i.e. hat function, mollifier with a support width of

$$h_m = 2\left(\max_j h_{c,j}\right). \tag{29}$$

Note that the normalisation of the mollifier is essential to ensuring that it integrates as required to one. Figure 7 shows the optimal convergence of the mollified finite element approach in the L^2 norm and H^1 seminorm with $q^p + 1$ and q^p respectively. These convergence rates are in agreement with the analytic estimates provided in Appendix A.

Next, we investigate the influence of the mollifier support width on the convergence order while keeping the normalised linear B-spline mollifier. The mollifier width is chosen according to

$$h_m = 2\chi\left(\max_j h_{c,j}\right) \quad \text{with } \chi \in \{1,\ 1.25,\ 1.5\}. \tag{30}$$

The increase in mollifier size leads to an increase in the support size of the mollified basis functions, which results in an increase of the number of non-zero basis functions in a cell. The obtained optimal convergence rates for $q^p = 2$ are shown in Figure 8. The increase in mollifier width leads to a somewhat decrease in the convergence constants, but the optimal support size appears to depend on the specific problem considered. The results for higher order local polynomials are similar and have been not included here.

Finally, we study the effect of the mollifier smoothness on finite element convergence. The normalised B-spline mollifiers are of degree $q^m \in \{1, 2, 3\}$ and the local polynomial is of degree $q^p = 2$. The mollifier width factor is chosen as $\chi = 1$. Note that the B-spline mollifiers are $C^{q^m - 1}$ continuous so that the obtained mollified basis functions are C^{q^m} continuous. Figure 9 shows that an increase of the kernel degree and smoothness does not have an effect on the optimal convergence rate, but leads to a significant decrease in convergence constants.

5.2. Two-dimensional examples

5.2.1. Poisson problem on a square domain

We consider next the Poisson-Dirichlet problem $-\nabla \cdot \nabla u = s$ on a square domain $\Omega = (0,\ 1) \times (0,\ 1)$. The domain Ω is partitioned into n_c cells using the Voronoi diagram of n_c non-uniformly distributed points, see Figure 10. Starting

11

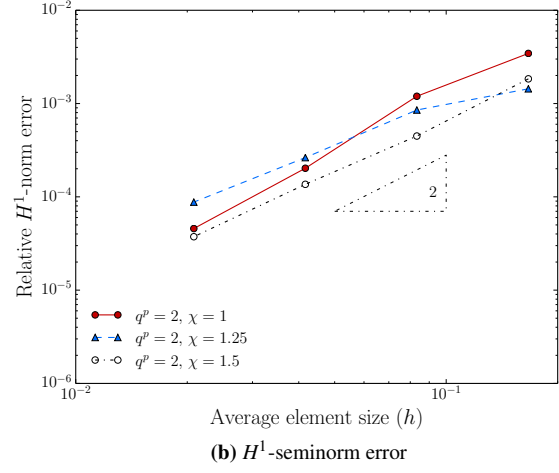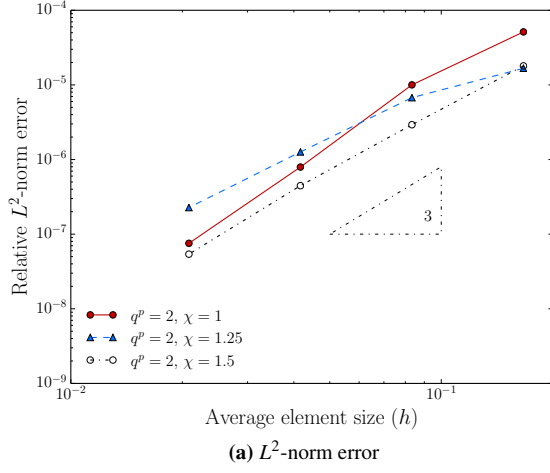**(a)** L^2-norm error

(b) H^1-seminorm error

Figure 8: One-dimensional Poisson problem. Convergence with local polynomial basis of degree $q^p = 2$ and normalised linear B-spline mollifier with different support sizes of $\chi \in \{1, 1.25, 1.5\}$.
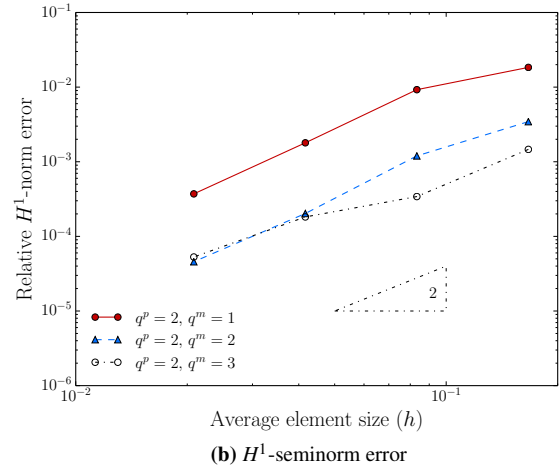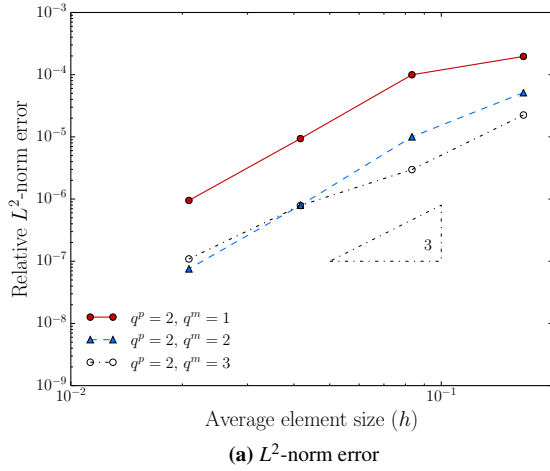

(a) L^2-norm error

(b) H^1-seminorm error

Figure 9: One-dimensional Poisson problem. Convergence with local polynomial basis of degree $q^p = 2$ and normalised B-spline mollifiers of degrees $q^m \in \{1, 2, 3\}$.
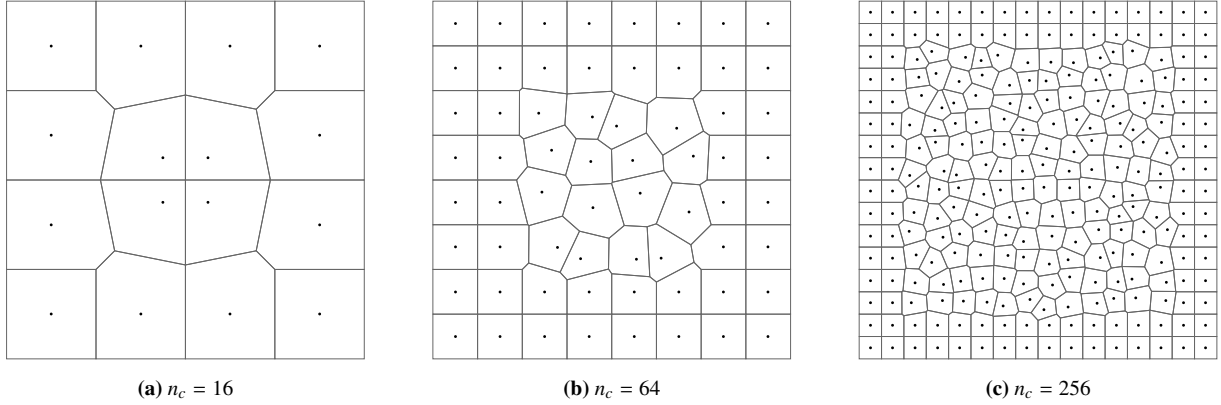
(a) $n_c = 16$ **(b)** $n_c = 64$ **(c)** $n_c = 256$

Figure 10: Poisson problem on a square domain. Three different partitionings of the domain using the Voronoi diagram of n_c non-uniformly distributed points.

from a set of uniformly distributed points we introduce non-uniformity by randomly perturbing their coordinates by $\kappa \sqrt{1/n_c}$ with $\kappa \sim \mathcal{U}(-0.15h_m/2, +0.15h_m/2)$. Only the coordinates of points farther than a certain distance from the boundaries are perturbed.

As in the one-dimensional case the domain is padded with an extra layer of ghost cells (not shown in Figure 10) to ensure that the mollified basis functions are complete close to the boundaries. Depending on the number of Voronoi cells n_c the width of the C^1 continuous quartic spline mollifier is chosen with

$$h_m = 2 \left(\frac{1}{n_c} \right)^{\frac{1}{2}} . \tag{31}$$

We firstly perform a patch test to verify that the mollified finite element method in combination with variationally consistent integration can exactly solve problems with polynomial degree q^p. To this end, we consider on the mesh shown in Figure 10b two problems with the exact solutions $u(\boldsymbol{x}) = x + 2y$ and $u(\boldsymbol{x}) = x + 2y + x^2 + 2xy + y^2$. Solving the linear problem using the linear mollified basis functions with $q^p = 1$ leads to an L^2 norm error of 8.635×10^{-14} and an H^1 seminorm error of 2.902×10^{-12}. The corresponding errors for the quadratic problem using quadratic mollified basis functions with $q^p = 2$ are 7.060×10^{-12} and 3.198×10^{-10}. This clearly confirms that the mollified finite element method satisfies the patch test.

With the consistency of the method confirmed, we proceed to establish its convergence under mesh refinement. The source term $s(x)$ is now chosen such that the solution is equal to

$$u(\boldsymbol{x}) = \sin\left(\pi x^{(1)}\right) \sin\left(\pi x^{(2)}\right) . \tag{32}$$

The used mollified basis functions are the C^2 continuous linear and quadratic basis functions with $q^p = 1$ and $q^p = 2$, respectively. Figure 11 shows the convergence of the errors in L^2 norm and H^1 seminorm as the mesh is refined. Note that the refined meshes are not nested so that some small kinks in the convergence curves may be expected. The average convergence rates are, however, close to optimal, as indicated by the dashed triangles in Figure 11.

5.2.2. Elastic plate with a hole

As a two-dimensional problem with a non-trivial geometry we compute the infinite elastic plate with a hole subjected to uniaxial tension. The radius of the hole is $R = 0.25$ and the applied uniaxial traction in the vertical direction is $\sigma_\infty = 10^6$. The Young's modulus of the material is $E = 70 \cdot 10^6$ and its Poisson's ratio is $\nu = 0.3$. This problem has a known closed-form analytic solution [49]. Therefore, we discretise only the plate of size $L = 1$ shown in Figure 12 and apply Dirichlet boundary conditions over its entire boundary.

The initial mesh consists of a Voronoi diagram of 36 non-uniformly distributed points, see Figure 13. The refined meshes are obtained by subdividing the cells by introducing new vertices on the cell and edge centres. This refinement
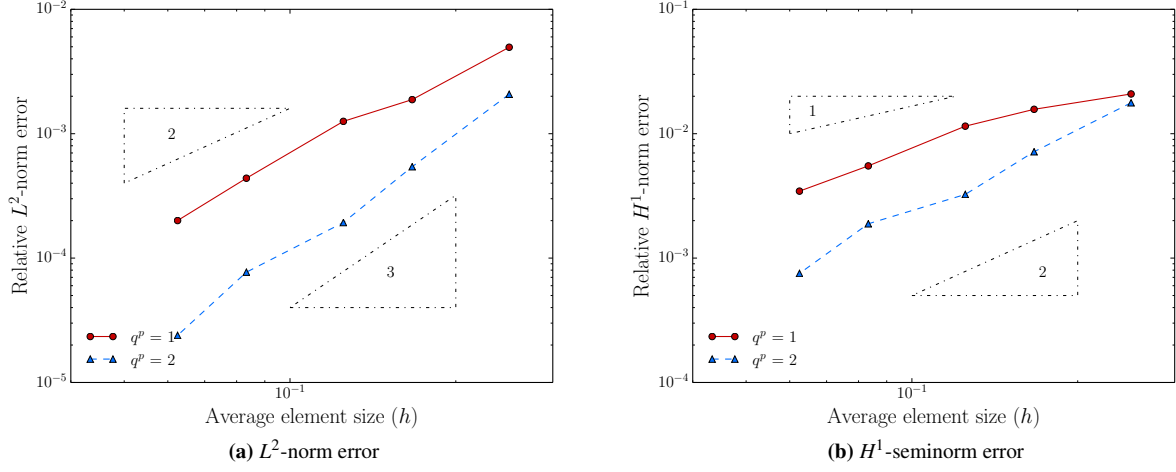
13

(a) L^2-norm error

(b) H^1-seminorm error

Figure 11: Poisson problem on a square domain. Convergence with quartic spline mollifier and local polynomial basis of degree $q^p = 1$ and $q^p = 2$.
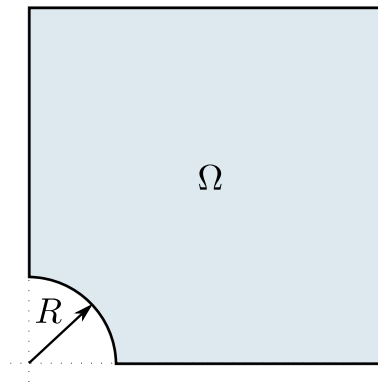

Figure 12: Geometry and boundary conditions of the elastic plate with a hole.

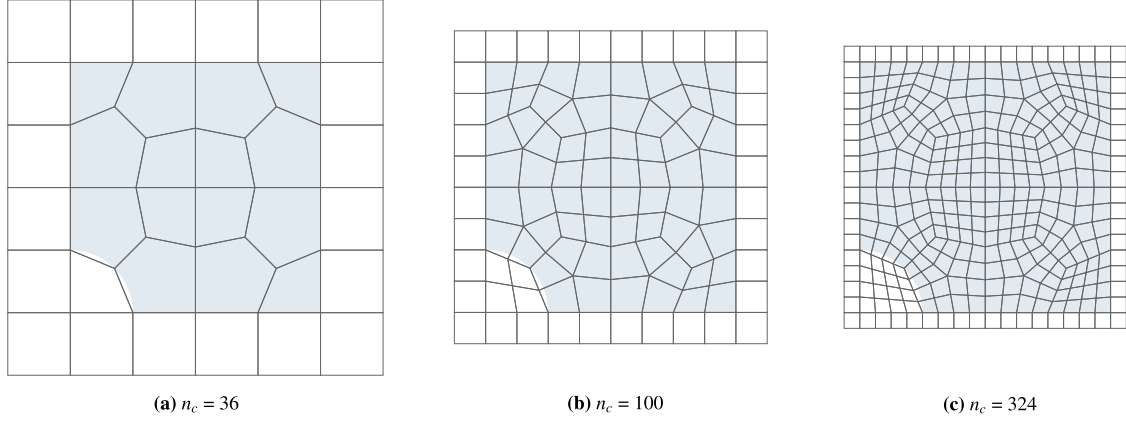**(a)** $n_c = 36$ **(b)** $n_c = 100$ **(c)** $n_c = 324$

Figure 13: Elastic plate with a hole. Three different partitionings of the plate with a hole (highlighted in blue). The elements outside the domain are the ghost cells. The coarse mesh in (a) is a Voronoi diagram which is refined to obtain (b) and (c) by introducing new vertices on the cell and edge centres.
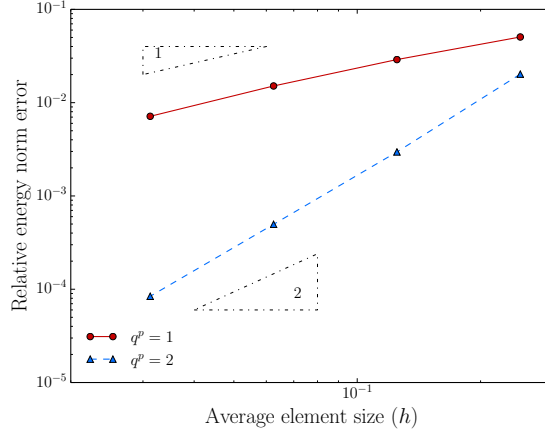

Figure 14: Elastic plate with a hole. Convergence of the relative energy norm error with quartic spline mollifier and local polynomial basis of degree $q^p = 1$ and $q^p = 2$.

ensures that the meshes are nested. As can be inferred from Figure 13 along the circular boundary the mesh edges are not aligned with the boundary. In the respective cells cut by the boundary the finite element integrals are evaluated only over the cell areas inside the domain. The cut-cells for integration are obtained with the clipping process introduced in Appendix B. To achieve a higher order approximation the edges of the triangles used for integration are curved by introducing additional nodes on the faces. As in standard finite elements, to achieve an optimally convergent method the boundary geometry has to be approximated with the same polynomial order as the used mollified basis functions.

To analyse this problem we again use C^2 continuous linear and quadratic basis functions with $q^p = 1$ and $q^p = 2$, respectively. Figure 14 shows the convergence of the errors in the energy norm. It is apparent that optimal convergence rates are achieved. A final comment concerns the possibly very small contributions of basis functions cut by the boundary to the system matrix. To this end, several approaches have been developed in immersed/embedded finite elements [41, 50, 51]. A particularly simple approach is simply to scale the relevant basis functions according to their support size within the domain, i.e.

$$\frac{|\operatorname{supp}(N_i) \cap \Omega|}{|\operatorname{supp}(N_i)|} . \tag{33}$$

15

5.3. Three-dimensional example

As an illustrative three-dimensional example with a complex boundary we consider the solution of a Poisson-Dirichlet problem on the domain contained within the Stanford bunny, see Figure 15. The geometry of the bunny is given as a triangle mesh with 66272 facets. The volume mesh shown in Figure 15c is created in several steps. Firstly we introduce within the bounding box of the bunny a set of uniformly distributed $20 \times 20 \times 20$ points with each 0.08 apart. The head is then refined by introducing additional $6 \times 7 \times 9$ points with each 0.05 apart. Finally, one of the ears is refined by adding $6 \times 7 \times 8$ points with each 0.04 apart. We then generate the Voronoi diagram of all the points and iteratively relax it to achieve a more even distribution of cell sizes. During the iterative relaxation the new position of each point is recomputed by convolving nodal coordinates with a box function. This relaxation is equivalent to standard Laplace smoothing of meshes. After the relaxation the Voronoi diagram is clipped with the technique described in Appendix B. The final mesh consists of 897 cells.

We solve on the generated polytopic mesh a Poisson-Dirichlet problem with a source term $s(x)$ such that the solution is equal to

$$u = \cos\left(\pi x^{(1)}\right) \sin\left(\pi x^{(2)}\right) \cos\left(\pi x^{(3)}\right) . \tag{34}$$

The isocontours of the computed solution are shown in Figure 15d. For the employed mollified basis functions we use a local polynomial basis with $q^p = 1$ and a quartic spline mollifier with a support size of $h_m = 0.16$, which is twice the coarse cell size. The discretisation has in total 9032 basis functions. The cut-cells are stabilised as described earlier by scaling the basis functions according to (33).

6. Conclusions

We introduced the mollified basis functions of arbitrary order and smoothness and verified their excellent finite element approximation properties with a selected set of examples. In the two- and three-dimensional examples we chose the Voronoi diagram of a given set of points as the partitioning of the domain. The mollified basis functions are obtained by convolving cell-wise defined local polynomial approximants with a compactly supported smooth mollifier with a unit volume. We integrate the convolution integrals exactly (up to round-off errors) by first determining the geometry of the polytopic integration domain and then reducing the dimension of the integrals using the divergence theorem. In determining the polytopic integration domain we consider the intersection of a single cell with the mollifier and make use of polytope clipping and convex-hull computation algorithms. Efficient and robust implementations of both algorithms are available in most development platforms, including Matlab, Mathematica and Python, and in high-performance geometry libraries [52]. Because the mollified basis functions are not boundary conforming, we enforce boundary conditions with standard immersed/embedded finite element techniques. The obtained polynomial basis functions may have breakpoints, i.e. points or lines of reduced continuity, within the cells. Therefore, we evaluate the finite element integrals with a variationally consistent approach originally developed for meshless methods. As shown numerically and analytically the mollified basis functions in combination with the proposed finite element implementation can pass the patch test and achieve optimal rates of convergence. Finally, while all the operations in evaluating the mollified basis functions are of geometric nature, in meshless methods, like RPKM, a dense local matrix must be inverted. Evidently, this matrix can become very large especially in 3D when high-order polynomials are used. The algorithmic complexity of the geometric operations is at most log-linear and the complexity of the matrix inversion is cubic. Therefore, we conjecture that the mollified finite element approximants are more efficient for high-order polynomials and higher dimensions.

There are several promising applications and extensions of the proposed mollified approximation scheme worth mentioning. Clearly, it is straightforward to apply h-, p- and hp-refinement. A given Voronoi diagram can be h-refined by incrementally adding new points and updating the Voronoi diagram. For p-refinement it is sufficient to choose in each cell the order of the polynomial approximant differently. A-priori and a-posteriori estimators are crucial to making efficient use of h-, p- and hp-refinement in applications. Furthermore, in our present implementation the mollifier support size is uniform throughout the domain. As our preliminary experiments indicate, it is possible to vary the mollifier support within a domain. This can be, for instance, used in creating boundary interpolating approximants by continuously shrinking the mollifier support size to zero while approaching the boundary. An alternative approach to easing the enforcement of boundary conditions is to blend mollified basis functions with finite elements similar

16
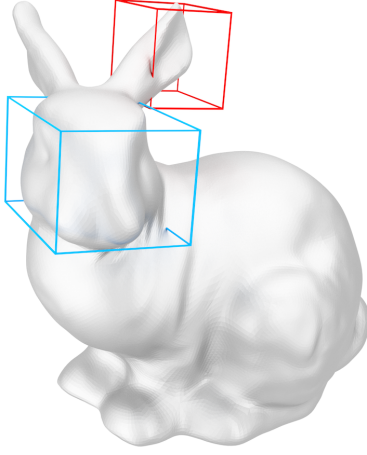
(a) Fine surface mesh

(b) Surface of the clipped Voronoi diagram
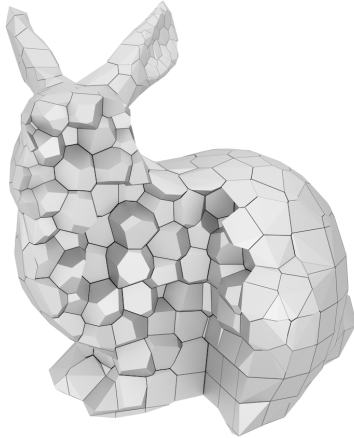
(c) Voronoi diagram
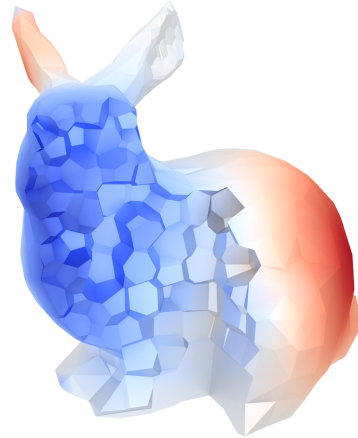
(d) Finite element solution

Figure 15: Poisson-Dirichlet problem on the domain contained within the Stanford bunny. The two boxes in (a) indicate the locations where additional nodes are introduced to refine the mesh. The mesh in (c), with some of the cells omitted, shows the unstructured Voronoi diagram used in the mollified finite element computation (d).

to blending techniques developed for meshless methods [53, 54]. Lastly, returning to our original motivation in developing smooth basis functions for isogeometric analysis, it is appealing to develop mollified blending techniques for B-spline patches meeting at an extraordinary vertex. The convolutional definition of the B-splines can be used to derive mollified approximants which reduce to B-splines away from the mesh boundaries.

Appendix A. Convergence estimates

We make use of convolution and polynomial approximation estimates to derive convergence estimates for the proposed mollified finite element approximation scheme [17, 55]. In doing so, we adopt a multiindex notation and denote derivatives with $D_j = \partial/\partial x_j$. If the multiindex $\alpha = (\alpha_1, \dots, \alpha_d)$ is an d-tuple of nonnegative integers α_j, then $D^\alpha = D_1^{\alpha_1} \cdots D_n^{\alpha_d}$ is a differential operator of order $|\alpha| = \alpha_1 + \cdots + \alpha_d$, with the convention that $D^{(0,\dots,0)} u = u$. We denote the norms of the Lebesgue and Sobolev spaces $L^p(\Omega)$ and $W^{p,k}(\Omega)$ with

$$
\begin{aligned}
\|f\|_{L^p} &:= \left(\int_\Omega |f(\boldsymbol{x})|^p \, \mathrm{d}\boldsymbol{x} \right)^{1/p} \\
\|f\|_{W^{p,k}} &:= \left(\sum_{|\alpha| \le k} \|D^\alpha f\|_{L^p}^p \right)^{1/p} \qquad |f|_{W^{p,k}} := \left(\sum_{|\alpha| = k} \|D^\alpha f\|_{L^p}^p \right)^{1/p} .
\end{aligned}
\tag{A.1}
$$

In this Appendix we assume that the mollifier is C^∞ continuous. For instance, we may take

$$
\rho(\boldsymbol{x}) = \begin{cases} \exp[-1/(1 - |\boldsymbol{x}|^2)]/I_n, & \text{if } |\boldsymbol{x}| < 1 \\ 0, & \text{otherwise}, \end{cases}
\tag{A.2}
$$

where the scalar $I_n > 0$ is chosen so that $\rho(\boldsymbol{x})$ has unit volume. The scaled mollifier with the support size $|\square_{\boldsymbol{0}}| = h_m$ is given by

$$
m(\boldsymbol{x}) = \rho(2\boldsymbol{x}/h_m)
\tag{A.3}
$$

so that

$$
\|D^\alpha m\|_{L^1} \le \frac{C}{h_m^{|\alpha|}}
\tag{A.4}
$$

for any multiindex α.

As shown, e.g. in [17], the derivatives of the convolution satisfy for any pair of multiindices α and β the relation

$$
D^{\alpha+\beta}(m * u) = D^\alpha m * D^\beta u
\tag{A.5}
$$

and the Young's inequality for convolutions reads

$$
\|f * u\|_{L^p} \le \|f\|_{L^1} \|u\|_{L^p}
\tag{A.6}
$$

yielding the estimate

$$
\|m * u\|_{L^p} \le \|u\|_{L^p} .
\tag{A.7}
$$

Moreover, according to the Bramble-Hilbert lemma, for a $u \in W^{p,k+1}(\Omega)$, $k \ge 1$, $1 \le p \le \infty$, there is a $v_h \in P_k$ (polynomials of degree less than or equal to k in one variable) such that

$$
\|v_h - u\|_{L^p(\Omega)} \le C h^{k+1} |u|_{W^{p,k+1}(\Omega)} .
\tag{A.8}
$$

Let us now extend u and v_h by zero outside the domain Ω and denote their mollifications with

$$
\widehat{v}_h = m * v_h \quad \text{and} \quad \widehat{u} = m * u .
\tag{A.9}
$$

Then, for $l < k$, from (A.5), (A.7), (A.4) and (A.8) we have

$$\|\widehat{v_h} - \widehat{u}\|_{W^{p,l}(\Omega)} \leq \frac{C}{h_m^l}\|v_h - u\|_{L^p(\Omega)} \leq \frac{Ah^{k+1}}{h_m^l}|u|_{W^{p,k+1}(\Omega)}, \tag{A.10}$$

where $A > 0$ is a constant. The difference between a function and its mollification can be bounded with a standard approximation theorem for convolutions ([17], Theorem 5.33) which reads

$$\|\widehat{u} - u\|_{W^{p,l}(\Omega)} \leq Bh_m^{k+1-l}|u|_{W^{p,k+1}(\Omega)}, \tag{A.11}$$

where $B > 0$ is a constant. Combining the preceding two estimates we obtain

$$\begin{aligned}\|\widehat{v_h} - u\|_{W^{p,l}(\Omega)} &\leq \|\widehat{v_h} - \widehat{u}\|_{W^{p,l}(\Omega)} + \|\widehat{u} - u\|_{W^{p,l}(\Omega)} \\ &\leq \left(A\frac{h^{k+1}}{h_m^l} + Bh_m^{k+1-l}\right)|u|_{W^{p,k+1}(\Omega)}.\end{aligned} \tag{A.12}$$

This bound is minimised by taking

$$h_m = \left(\frac{lA}{(k+1-l)B}\right)^{1/(k+1)} h \equiv h_m^*, \tag{A.13}$$

which defines the optimal mollifier support size h_m^* in dependence of the mesh size h. Inserting the optimal h_m^* into the estimate (A.12) gives

$$\|\widehat{v_h}^* - u\|_{W^{p,l}(\Omega)} \leq Ch^{k+1-l}|u|_{W^{p,k+1}(\Omega)}, \tag{A.14}$$

where $\widehat{v_h}^*$ denotes the mollification of the polynomial $v_h \in P_k$ with a mollifier with support size l h_m^*. We note that the estimate (A.14) provides control over l-th order derivatives, whereas the initial estimate (A.8) does not.

As in standard finite element approximation theory, see e.g. [55], applying estimate (A.14) cell-wise and considering their sum yields global convergence estimates. Subsequently, it is straightforward to confirm the optimal convergence of the proposed mollified finite elements as already suggested by our numerical experiments.

Appendix B. Clipped Voronoi diagrams

We briefly review the properties of Voronoi diagrams and sketch the generation of clipped Voronoi diagrams that approximately fill a given domain Ω. For a more detailed discussion see, e.g., [36, 56, 57]. For a set of points $\{c_i\}_{i=1}^{n_c}$ in $\mathbb{R}^d$ the Voronoi diagram is defined by a set of cells $\{\Omega_i\}_{i=1}^{n_c}$ such that

$$\Omega_i = \left\{x \in \mathbb{R}^d | \ |x - c_i| \leq |x - c_j| \quad \forall i \neq j\right\}. \tag{B.1}$$

As indicated in Figure B.16, the cells are convex, are either bounded or unbounded and have planar faces. There are a number of efficient software libraries available for generating Voronoi diagrams, such as the Voro++ [58] library (for 3D) or Mathematica (for 2D) used in this work. To obtain a Voronoi diagram that approximately fills a given domain Ω the cells intersected by the boundary are clipped. To implement the clipping process we assume that the domain Ω is described implicitly with a signed distance function

$$\phi(x) = \begin{cases} \min_{y \in \Gamma} |x - y| & \text{if } x \in \Omega \\ 0 & \text{if } x \in \Gamma \\ -\min_{y \in \Gamma} |x - y| & \text{otherwise}, \end{cases} \tag{B.2}$$

where Γ is the boundary of the domain Ω. Domains that are described with a parametric polygon mesh can first be converted to an implicit signed distance function representation using standard algorithms, see e.g. [41].
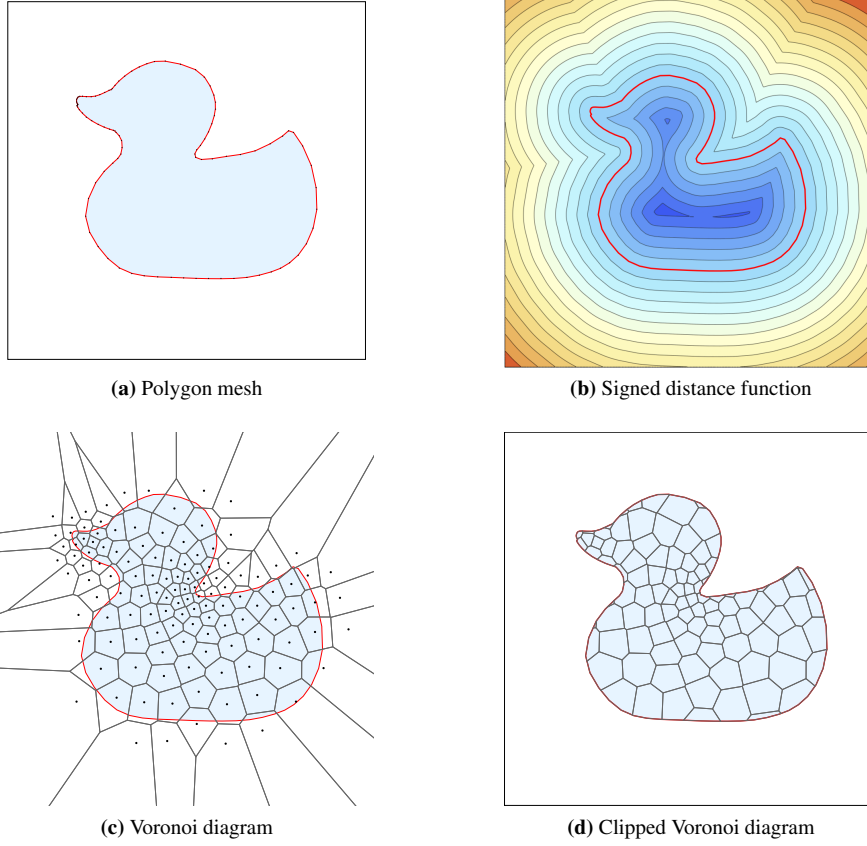
(a) Polygon mesh

(b) Signed distance function

(c) Voronoi diagram

(d) Clipped Voronoi diagram

Figure B.16: Clipped Voronoi diagram of a domain described by a polygonal mesh.

The minimal data structure for representing a Voronoi cell Ω_i consists of its vertices $\{\boldsymbol{\eta}_{i,j}\}$ and orientated faces $\{\gamma_{i,j}\}$, with all the face normals $\boldsymbol{n}_{i,j}$ pointing, e.g., outside the cell. The cells cut by the boundary are determined by evaluating the level set function at the vertices and checking whether

$$\min_j \phi(\boldsymbol{\eta}_{i,j}) \cdot \max_j \phi(\boldsymbol{\eta}_{i,j}) < 0 \tag{B.3}$$

is satisfied. Each of the cut cells is clipped by performing the following steps:

1. Deduce the set of cell edges from $\{\boldsymbol{\eta}_{i,j}\}$ and $\{\gamma_{i,j}\}$.
2. Determine the intersection points between edges and the boundary $\phi(\boldsymbol{x}) = 0$ with a bisectioning algorithm.
3. Introduce new vertices at the points determined in step 2.
4. Generate a new clipped cell by determining the convex hull of vertices inside the domain and the vertices on the domain boundary.

In finite element computations the clipped cells represent the integration domain for element integrals for the cells touching the domain boundary. The clipping process introduced yields only convex clipped cells with planar boundaries. For domains with curved boundaries this limits the overall accuracy of finite element method to first order even when higher order mollified basis functions are used. In this setting, a standard approach to achieving higher order accuracy in immersed finite elements is to curve the planar faces by introducing additional vertices on the faces, which is clearly also applicable to mollified finite elements.

Appendix C. Intersection of a convex cell with a box

The intersection between a convex cell and a box is required to evaluate the mollified basis functions. The box represents the support of the mollifier and is centred at the given evaluation point. The intersection of two convex solids is frequently required in computer graphics and a range of efficient and robust algorithms is available. Our specific approach is motivated by the more general algorithm presented in [59]. The key idea is that the intersection of a cell and a box can be determined by clipping the cell in turn by the six half-spaces defining the box. Each half-space is defined by a plane, or its respective normal, and has an inside and outside. Furthermore, recall that the intersection of two convex solids is always convex.

As mentioned in Appendix B, in our implementation, a cell Ω_i is represented by its vertices $\{\boldsymbol{\eta}_{i,j}\}$ and orientated faces $\{\gamma_{i,j}\}$. With this in mind, the sequence of steps in computing the convex polytope representing the intersection domain is as follows:

1. Deduce the set of cell edges from $\{\boldsymbol{\eta}_{i,j}\}$ and $\{\gamma_{i,j}\}$.
2. Determine the intersection points between the edges and the six half planes in turn while keeping track of the vertices inside the half-spaces.
3. Generate a new polyhedron by determining the convex hull of vertices inside the domain and the intersection points on the edges.

In meshes with a large number of cells it is usually more efficient first to identify the small set of cells which are possibly intersected by a given box. Subsequent intersection computations have to be applied only to the few identified cells. The relevant cells can be efficiently identified with a standard hierarchical bounding volume tree, see e.g. [60].

Appendix D. Minkowski sum of two polytopes

The support of the mollified basis functions N_i corresponding to the cell Ω_i is obtained as the Minkowski sum of the cell with the support of the mollifier. For an in-depth introduction to Minkowski sums see, e.g., [36]. The Minkowski sum of two sets Ω_i, $\square_0 \in \mathbb{R}^2$ is defined by

$$\widehat{\Omega}_i = \Omega_i \oplus \square_0 = \{\boldsymbol{x} + \boldsymbol{y} \,|\, \boldsymbol{x} \in \Omega_i, \boldsymbol{y} \in \square_0\}. \tag{D.1}$$

The domain $\widehat{\Omega}_i$ resulting from the Minkowski sum may be visualised as that obtained by sliding the centre of $\square_0$ along the boundaries of Ω_i, see Figure D.17. Recall here that the domain $\square_0$ is centred at the origin of the coordinate axis as implied by the subscript $\boldsymbol{0}$. It is easy to show that $\widehat{\Omega}_i$ is convex because both Ω_i and $\square_0$ are convex. We use this to devise a simple algorithm for computing the Minkowski sum. That is, we first generate a set of points by sliding the domain $\square_0$ along the boundaries of Ω_i, which we subsequently combine with a convex hull algorithm to obtain $\widehat{\Omega}_i$. In generating the set of points it is sufficient to place $\square_{\boldsymbol{\eta}_{i,j}}$ at the vertices $\boldsymbol{\eta}_{i,j}$ of the domain Ω_i and to take successively the union of the vertices of $\square_{\boldsymbol{\eta}_{i,j}}$, see Figure D.17 .

References

[1] F. Cirak, M. Ortiz, P. Schröder, Subdivision surfaces: A new paradigm for thin-shell finite-element analysis, International Journal for Numerical Methods in Engineering 47 (2000) 2039–2072.

[2] T. J. R. Hughes, J. A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Computer Methods in Applied Mechanics and Engineering 194 (2005) 4135–4195.

[3] J. A. Evans, Y. Bazilevs, I. Babuska, T. J. R. Hughes, n-Widths, sup-infs, and optimality ratios for the k-version of the isogeometric finite element method, Computer Methods in Applied Mechanics and Engineering 198 (2009) 1726–1741.

[4] G. Farin, Curves and Surfaces for CAGD, Academic Press, 2002.

[5] J. Peters, U. Reif, Subdivision Surfaces, Springer, 2008.

[6] M. A. Scott, R. N. Simpson, J. A. Evans, S. Lipton, S. P. A. Bordas, T. J. R. Hughes, T. W. Sederberg, Isogeometric boundary element analysis using unstructured T-splines, Computer Methods in Applied Mechanics and Engineering (2013) 197–221.

[7] Y. Jia, Y. Zhang, G. Xu, X. Zhuang, T. Rabczuk, Reproducing kernel triangular B-spline-based FEM for solving PDEs, Computer Methods in Applied Mechanics and Engineering 267 (2013) 342–358.

[8] M. Majeed, F. Cirak, Isogeometric analysis using manifold-based smooth basis functions, Computer Methods in Applied Mechanics and Engineering 316 (2017) 547–567.
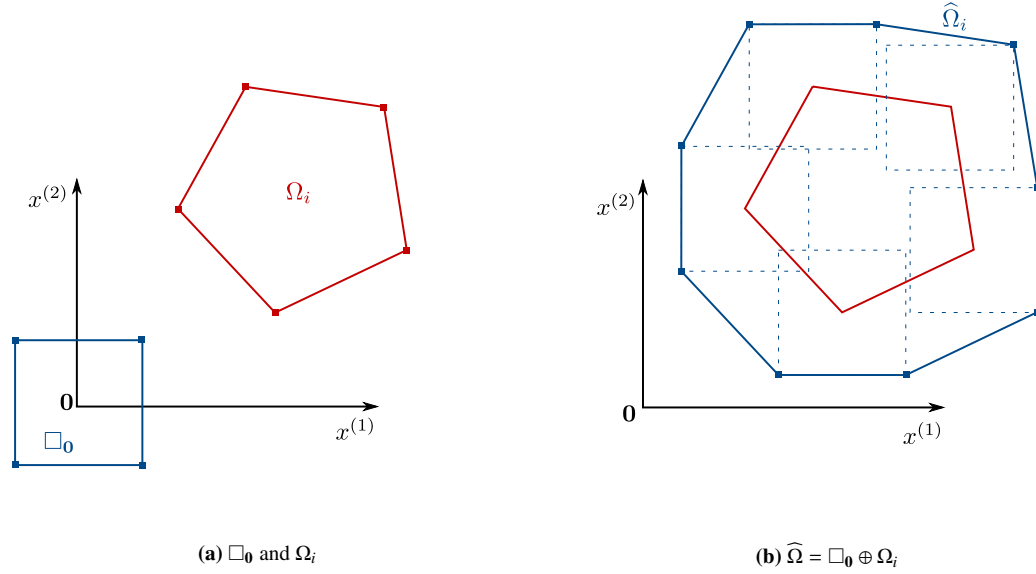
(a) $\square_{\mathbf{0}}$ and Ω_i **(b)** $\widehat{\Omega} = \square_{\mathbf{0}} \oplus \Omega_i$

Figure D.17: Minkowski sum of the mollifier support $\square_{\mathbf{0}}$ and the cell Ω_i. The dashed $\square_{\eta_{i,j}}$ in (b) indicate the sliding of the centre of $\square_{\mathbf{0}}$ along the boundaries of Ω_i.

[9] D. Toshniwal, H. Speleers, T. J. R. Hughes, Smooth cubic spline spaces on unstructured quadrilateral meshes with particular emphasis on extraordinary points: Geometric design and isogeometric analysis considerations, Computer Methods in Applied Mechanics and Engineering 327 (2017) 411–458.

[10] D. Toshniwal, H. Speleers, R. R. Hiemstra, T. J. R. Hughes, Multi-degree smooth polar splines: A framework for geometric modeling and isogeometric analysis, Computer Methods in Applied Mechanics and Engineering 316 (2017) 1005–1061.

[11] Q. Zhang, M. Sabin, F. Cirak, Subdivision surfaces with isogeometric analysis adapted refinement weights, Computer-Aided Design 102 (2018) 104–114.

[12] M. Kapl, G. Sangalli, T. Takacs, Construction of analysis-suitable G^1 planar multi-patch parameterizations, Computer-Aided Design 97 (2018) 41–55.

[13] Q. Zhang, F. Cirak, Manifold-based isogeometric analysis basis functions with prescribed sharp features, Computer Methods in Applied Mechanics and Engineering 359 (2020) 112659.

[14] C. De Boor, B(asic)-Spline Basics., Tech. Rep., Madison mathematics research center, Winsconsin University, 1986.

[15] S. Hilbert, A Mollifier Useful for Approximations in Sobolev Spaces and Some Applications to Approximating Solutions of Differential Equations, Mathematics of Computation 27 (1973) 81–89.

[16] L. C. Evans, Partial differential equations, American Mathematical Society, 1998.

[17] R. A. Adams, J. J. F. Fournier, Sobolev Spaces, Elsevier Science, 2003.

[18] V. Thomée, High order local approximations to derivatives in the finite element method, Mathematics of Computation 31 (1977) 652–660.

[19] J. H. Bramble, A. H. Schatz, Higher order local accuracy by averaging in the finite element method, Mathematics of Computation 31 (1977) 94–111.

[20] H. Mirzaee, J. King, J. K. Ryan, R. M. Kirby, Smoothness-increasing accuracy-conserving filters for discontinuous Galerkin solutions over unstructured triangular meshes, SIAM Journal on Scientific Computing 35 (2013) A212–A230.

[21] O. Sigmund, J. Petersson, Numerical instabilities in topology optimization: a survey on procedures dealing with checkerboards, mesh-dependencies and local minima, Structural optimization 16 (1998) 68–75.

[22] C. Le, T. Bruns, D. Tortorelli, A gradient-based, parameter-free approach to shape optimization, Computer Methods in Applied Mechanics and Engineering 200 (2011) 985–996.

[23] K.-U. Bletzinger, A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape, Structural and Multidisciplinary Optimization 49 (2014) 873–895.

[24] L. B. Lucy, A numerical approach to the testing of the fission hypothesis, The Astronomical Journal 82 (1977) 1013–1024.

[25] R. A. Gingold, J. J. Monaghan, Smoothed particle hydrodynamics: theory and application to non-spherical stars, Monthly notices of the royal astronomical society 181 (1977) 375–389.

[26] W. K. Liu, S. Jun, Y. F. Zhang, Reproducing kernel particle methods, International Journal for Numerical Methods in Fluids 20 (1995) 1081–1106.

[27] S. Li, W. K. Liu, Meshfree and particle methods and their applications, Applied Mechanics Reviews 55 (2002) 1–34.

[28] M. A. Bessa, J. T. Foster, T. Belytschko, W. K. Liu, A meshfree unification: reproducing kernel peridynamics, Computational Mechanics 53 (2014) 1251–1264.

[29] J.-S. Chen, M. Hillman, S.-W. Chi, Meshfree methods: progress made after 20 years, Journal of Engineering Mechanics 143 (2017) 04017001: 1–37.

[30] A. Huerta, T. Belytschko, S. Fernández-Méndez, T. Rabczuk, X. Zhuang, M. Arroyo, Meshfree methods, Encyclopedia of Computational Mechanics (2017) 1–38.

[31] W. K. Liu, W. Han, H. Lu, S. Li, J. Cao, Reproducing kernel element method. Part I: Theoretical formulation, Computer Methods in Applied Mechanics and Engineering 193 (2004) 933–951.

[32] D. Wang, H. Zhang, A consistently coupled isogeometric–meshfree method, Computer Methods in Applied Mechanics and Engineering 268 (2014) 843–870.

[33] N. Valizadeh, Y. Bazilevs, J. S. Chen, T. Rabczuk, A coupled IGA–Meshfree discretization of arbitrary order of accuracy and without global geometry parameterization, Computer Methods in Applied Mechanics and Engineering 293 (2015) 20–37.

[34] E. B. Chin, J. B. Lasserre, N. Sukumar, Numerical integration of homogeneous functions on convex and nonconvex polygons and polyhedra, Computational Mechanics 56 (2015) 967–981.

[35] Y. Sudhakar, J. M. De Almeida, W. A. Wall, An accurate, robust, and easy-to-implement method for integration over arbitrary polyhedra: application to embedded interface methods, Journal of Computational Physics 273 (2014) 393–415.

[36] M. de Berg, O. Cheong, M. van Kreveld, M. Overmars, Computational Geometry: Algorithms and Applications, Springer, 3rd edn., 2010.

[37] J. Nitsche, Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind, Abhandlungen aus dem Mathematischen Seminar der Universität Hamburg 36 (1971) 9–15.

[38] J. T. Oden, I. Babuŝka, C. E. Baumann, A discontinuous hp finite element method for diffusion problems, Journal of Computational Physics 146 (1998) 491–519.

[39] T. Boiveau, E. Burman, A penalty-free Nitsche method for the weak imposition of boundary conditions in compressible and incompressible elasticity, IMA Journal of Numerical Analysis 36 (2015) 770–795.

[40] D. Schillinger, I. Harari, M.-C. Hsu, D. Kamensky, S. K. F. Stoter, Y. Yu, Y. Zhao, The non-symmetric Nitsche method for the parameter-free imposition of weak boundary and coupling conditions in immersed finite elements, Computer Methods in Applied Mechanics and Engineering 309 (2016) 625–652.

[41] T. Rüberg, F. Cirak, Subdivision-stabilised immersed b-spline finite elements for moving boundary flows, Computer Methods in Applied Mechanics and Engineering 209–212 (2012) 266–283.

[42] M. Sabin, Analysis and Design of Univariate Subdivision Schemes, Springer, 2010.

[43] B. Mirtich, Fast and accurate computation of polyhedral mass properties, Journal of Graphics Tools 1 (1996) 31–50.

[44] J.-S. Chen, M. Hillman, M. Rüter, An arbitrary order variationally consistent integration for Galerkin meshfree methods, International Journal for Numerical Methods in Engineering 95 (2013) 387–418.

[45] M. Hillman, J. S. Chen, Y. Bazilevs, Variationally consistent domain integration for isogeometric analysis, Computer Methods in Applied Mechanics and Engineering 284 (2015) 521 – 540.

[46] K. W. Cheng, T.-P. Fries, Higher-order XFEM for curved strong and weak discontinuities, International Journal for Numerical Methods in Engineering 82 (2010) 564–590.

[47] L. Kudela, N. Zander, T. Bog, S. Kollmannsberger, E. Rank, Efficient and accurate numerical quadrature for immersed boundary methods, Advanced Modeling and Simulation in Engineering Sciences 2.

[48] H. Xiao, E. Febrianto, Q. Zhang, F. Cirak, An Immersed Discontinuous Galerkin Method for Compressible Navier-Stokes Equations on Unstructured Meshes, International Journal for Numerical Methods in Fluids 91 (2019) 487–508.

[49] S. P. Timoshenko, Theory of elasticity, McGraw-Hill Higher Education, 3rd edn., 1970.

[50] F. de Prenter, C. V. Verhoosel, E. H. van Brummelen, Preconditioning immersed isogeometric finite element methods with application to flow problems, Computer Methods in Applied Mechanics and Engineering 348 (2019) 604–631.

[51] C. Gürkan, A. Massing, A stabilized cut discontinuous Galerkin framework for elliptic boundary value and interface problems, Computer Methods in Applied Mechanics and Engineering 348 (2019) 466–499.

[52] N. Ray, D. Sokolov, S. Lefebvre, B. Lévy, Meshless Voronoi on the GPU, ACM Transactions on Graphics (TOG) 37 (2018) 1–12.

[53] A. Huerta, S. Fernández-Méndez, Enrichment and coupling of the finite element and meshless methods, International Journal for Numerical Methods in Engineering 48 (2000) 1615–1636.

[54] A. Rosolen, M. Arroyo, Blending isogeometric analysis and local maximum entropy meshfree approximants, Computer Methods in Applied Mechanics and Engineering 264 (2013) 95–107.

[55] A. Ern, J. L. Guermond, Theory and Practice of Finite Elements, Springer New York, 2004.

[56] F. Aurenhammer, Voronoi Diagrams — A Survey of a Fundamental Geometric Data Structure, ACM Computing Surveys 23 (1991) 345–405.

[57] Q. Du, V. Faber, M. Gunzburger, Centroidal Voronoi Tessellations: Applications and Algorithms, SIAM Review 41 (1999) 637–676.

[58] C. Rycroft, Voro++: A three-dimensional Voronoi cell library in C++, Tech. Rep., Lawrence Berkeley National Lab, Berkeley, CA, 2009.

[59] D. E. Muller, F. P. Preparata, Finding the intersection of two convex polyhedra, Theoretical Computer Science 7 (1978) 217–236.

[60] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, K. Zikan, Efficient collision detection using bounding volume hierarchies of k-DOPs, IEEE Transactions on Visualization & Computer Graphics 4 (1998) 21–36.