Immersed b-spline (i-spline) finite element method for geometrically complex domains

R.A.K. Sanches, P.B. Bornemann, F. Cirak*

Department of Engineering, University of Cambridge,
Trumpington Street, Cambridge CB2 1PZ, U.K.

Abstract

A novel b-spline based immersed finite element method is introduced for the computation of geometrically and topologically complex problems. The geometry description and the finite element analysis rely on a block structured logically Cartesian mesh which encloses the domain of interest. A signed distance function is used for representing the domain on the Cartesian mesh, whereby the domain boundary is the zeroth level set of the signed distance function. Away from the domain boundaries, the standard b-spline basis functions are used for the finite element interpolation. Close to domain boundaries, a new approach has been developed for modifying the b-spline basis functions so that they locally interpolate the Dirichlet boundary conditions. The efficiency and robustness of the proposed approach is demonstrated with a number of one-, two- and three-dimensional linear boundary value problems.

Keywords: Finite elements, B-splines, Immersed methods, Cartesian meshes, Isogeometric analysis

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

Preprint submitted to Elsevier *May 14, 2011*

1. Introduction

B-splines and related interpolation functions, such as non-uniform rational b-splines (NURBS) are widely used in computer aided design (CAD). In contrast, in finite element analyses b-splines have been, until recently, only sparsely used. However, lately integrated geometric modelling and finite element analysis using b-splines gained momentum with the introduction of the *isogeometric analysis* paradigm by Hughes et al. [1]. In the spirit of isogeometric analysis, the *subdivision shells* introduced earlier by Cirak et al. [2] use subdivision surfaces which are historically motivated by b-splines, see e.g. [3, 4]. One key motivation for the use of b-spline basis functions throughout the geometric design and analysis workflow is the promise to side-step the error prone and often user guided generation of finite element meshes based on NURBS models. Furthermore, the use of the same basis functions can facilitate rapid data exchange between design and analysis models, which is, for example, crucial for design optimisation [5].

In addition to streamlining the design and analysis workflow, b-spline basis functions are in several ways more versatile and efficient than the Lagrangian interpolation functions presently used in finite elements, see [6] for an overview. For example, on tensor product meshes (also known as logically Cartesian or block-structured meshes), it is straightforward to define higher degree b-spline basis functions, to refine the b-spline basis functions and to increase or decrease their global smoothness. Importantly, most of these operations can be performed without altering the geometry of the described object. Moreover, one of the key properties of b-spline basis functions is that they are point-wise positive. This means, e.g., that all the components of a mass matrix are positive and, hence, the related lumped mass matrix is always positive definite. B-spline basis functions are also variation diminishing, which makes them less prone to numerical oscillations typically encountered with higher degree Lagrangian basis functions.

B-spline basis functions and most of the underlying theory is strictly tight to tensor product meshes (or more generally to shift invariant meshes, see e.g. [7, 8] for details). This results from the non-locality of the b-spline basis functions, which does not allow for a straightforward extension to unstructured meshes. Only in the two-dimensional manifold setting, there are the subdivision surfaces which generalise splines to unstructured meshes. However, an equivalent mathematically sound approach is presently not available in the three-dimensional setting. For many practically relevant geometries, logically Cartesian meshes are not flexible enough and for some geometries, like spherical objects, not possible. Therefore, it appears to be self-evident to combine b-spline basis functions with immersed boundary methods. Amongst the many published immersed boundary approaches, see e.g. [9, 10, 11], the key commonality is the use of Cartesian meshes which do not conform to physical domain boundaries. Close to boundaries auxiliary algorithms are used for enforcing Dirichlet and Neumann boundary conditions. Mostly, the auxiliary algorithms are derived from standard approaches for enforcing constraints in variational problems, such as the Lagrange multiplier, penalty or Nitsche methods.

The conventional algorithms for enforcing boundary conditions in immersed boundary methods are not directly applicable to an immersed b-spline finite element method. Due to the non-local nature of b-splines, they lead to a large number of basis functions close to the domain boundary which have only a small overlap with the physical domain. This has usually a detrimental effect on the numerical stability of the discretized problem, which has a negative impact on the robustness and accuracy of the overall approach. In the web-spline method of Höllig et al. [12, 13], which is a b-spline based immersed finite element method, this issue has been addressed by coupling the exterior to the interior basis functions avoiding thus very small effective supports. Furthermore, the web-spline method uses an approach introduced by Kantorovich and Krylov [14] for enforcing homogeneous (i.e., zero) Dirichlet boundary conditions. Although the web-spline method fails the patch test, it can be shown to be convergent. Furthermore, inhomogeneous (i.e., non-zero) Dirichlet boundary conditions are approximately considered by applying a suitable domain load close to the boundaries.

The immersed b-spline finite element method developed in this paper shares many advantages of the web-spline method and at the same time fulfils the patch test and is inherently robust. To this end, we follow a different approach by first deriving spline basis functions which are interpolating at the domain boundaries. From outset, b-spline basis functions outside the physical domain are omitted for interpolation purposes, which alleviates the stability issues associated with b-splines with a small support overlapping the physical domain. As a result of the proposed modifications, the convergence order of the proposed method is the same as for finite elements with linear polynomials.

The outline of the paper is as follows: In Section 2 b-splines are briefly recalled as they form the starting point from which the proposed basis functions emerge. Section 3 defines the boundary value problem and corresponding finite element context in case of b-splines on Cartesian grids. Section 4 provides the background on which the proposed basis functions are constructed inherently satisfying the Dirichlet boundary in a local way. The basis functions are applied to various one-, two- and three-dimensional examples in Section 5. Finally, conclusions are drawn in Section 6.

2. B-spline basis functions

In this section we provide a brief summary of b-spline basis functions as far as relevant for the present paper. A detailed presentation of b-splines can be found in standard textbooks, such as Piegl and Tiller [15], Rogers [16] or de Boor [17].

On a one-dimensional domain with the knots $\xi_0 = 0$, $\xi_1 = 1$, $\xi_2 = 2$, ..., the uni-variate b-spline basis functions can be given with the following well-known

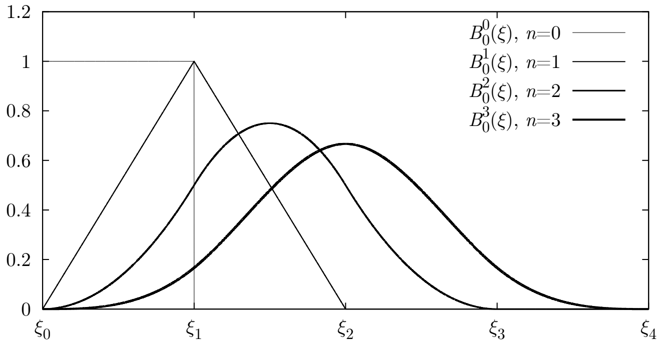


Figure 1: B-splines of polynomial degree zero, one, two and three; note the increasing smoothness and support size with increasing degree

recurrence relation

$$B_i^0(\xi) = \begin{cases} 1 & \text{if } \xi_i \leq \xi < \xi_{i+1} \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$B_i^n(\xi) = \frac{\xi - \xi_i}{\xi_{i+n} - \xi_i} B_i^{n-1}(\xi) + \frac{\xi_{i+n+1} - \xi}{\xi_{i+n+1} - \xi_{i+1}} B_{i+1}^{n-1}(\xi) \tag{2}$$

whereby n is the polynomial degree of the b-spline. From these equations, it can be deduced that the support of a b-spline basis function B_i^n of polynomial degree n reaches between $\xi_i \leq \xi \leq \xi_{i+n+1}$ (see also Figure 1). Furthermore, B_i^n is $(n-1)$-times continuously differentiable over the knots and is comprised of complete polynomials of degree n on its support. In this paper we use only uniform b-splines so that the distance between the knots is constant, i.e. $\Delta\xi = \xi_{i+1} - \xi_i = \text{const}$.

The uni-variate b-splines can be extended to several dimensions using the tensor product formalism:

$$B_{\boldsymbol{i}}^n(\boldsymbol{\xi}) = B_{i^1}^n(\xi^1) \times \ldots \times B_{i^m}^n(\xi^m) \quad \text{with} \quad \boldsymbol{\xi} = (\xi^1, \ldots, \xi^m) \in \mathbb{R}^m \tag{3}$$

denotes a general m-variate b-spline of degree $n \geq 0$ over the parameters ξ^i. Conveniently, the uni-directional indices $(i^1, \ldots, i^m)$ are combined in a unique multi-index $\boldsymbol{i}$. For example, combining two uni-variate b-splines in the two coordinate directions ξ^1 and ξ^2 yields a bi-variate b-spline $B_{\boldsymbol{i}}^n(\xi^1, \xi^2) = B_{i^1}^n(\xi^2) \times B_{i^2}^n(\xi^2)$. A bi-cubic b-spline $B_{\boldsymbol{i}}^3(\xi^1, \xi^2)$ is depicted in Fig. 2.

3. B-spline finite elements

In the present paper, we restrict ourselves to linear second order boundary value problems, namely elasticity and heat conduction. However, the presented

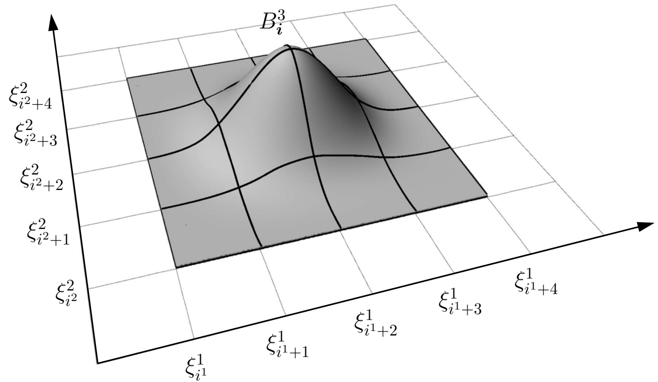


Figure 2: Bi-variate cubic b-spline basis function generated as the tensor product of two uni-variate cubic b-spline basis functions

overall approach can be easily generalised to other type of linear and nonlinear boundary value problems.

We consider the elasticity problem over a m-dimensional domain $\Omega \subset \mathbb{R}^m$, with the Neumann boundary Γ_N and the Dirichlet boundary Γ_D, which is given by

$$\begin{aligned} \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) + \boldsymbol{b} = \boldsymbol{0} &\quad \text{in } \Omega \\ \boldsymbol{\sigma}(\boldsymbol{u}) \cdot \boldsymbol{n} = \bar{\boldsymbol{t}} &\quad \text{on } \Gamma_\mathrm{N} \\ \boldsymbol{u} = \bar{\boldsymbol{u}} &\quad \text{on } \Gamma_\mathrm{D} \end{aligned} \tag{4}$$

where $\boldsymbol{u}$ are the displacements, $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{b}$ are the body forces, and $\bar{\boldsymbol{t}}$ and $\bar{\boldsymbol{u}}$ are the prescribed tractions and displacements, respectively. Equation (4) is closed with the usual linear elastic material relation $\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{C} : \boldsymbol{\varepsilon}(\boldsymbol{u})$ with the strain tensor $\boldsymbol{\varepsilon} = \frac{1}{2}\left(\nabla \boldsymbol{u} + (\nabla \boldsymbol{u})^\mathrm{T}\right)$ and the constitutive tensor $\boldsymbol{C}$. The related weak form can be stated as follows

$$\int_\Omega \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\varepsilon}(\boldsymbol{v})\, \mathrm{d}\Omega = \int_\Omega \boldsymbol{b} \cdot \boldsymbol{v}\, \mathrm{d}\Omega + \int_{\Gamma_\mathrm{N}} \bar{\boldsymbol{t}} \cdot \boldsymbol{v}\, \mathrm{d}\Gamma \tag{5}$$

where $\boldsymbol{v}$ are the test functions which have to be $\boldsymbol{v} = \boldsymbol{0}$ on the Dirichlet boundary Γ_D.

For discretizing the weak form with shape functions, the problem domain Ω is embedded into a slightly larger domain ω, which is suitable for a Cartesian mesh (see Figure 3). As discussed, we use the Cartesian mesh in order to facilitate the finite element discretization with tensor product b-splines. The b-spline basis functions on the Cartesian mesh provide a means to interpolate the displacements and test functions

$$\boldsymbol{u}^\mathrm{c}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i} \in \mathcal{C}} B_{\boldsymbol{i}}(\boldsymbol{\xi})\, \boldsymbol{u}_{\boldsymbol{i}}\,, \quad \boldsymbol{v}^\mathrm{c}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i} \in \mathcal{C}} B_{\boldsymbol{i}}(\boldsymbol{\xi})\, \boldsymbol{v}_{\boldsymbol{i}} \quad \text{with } \boldsymbol{\xi} \in \square \tag{6}$$

where $B_{\boldsymbol{i}}(\boldsymbol{\xi})$ are the b-spline basis functions of polynomial degree $n = 3$ which span the space $\mathcal{U}^c = P^3(\bar{\square}, \mathbb{R}^m)$. We focus on cubic b-splines and imply $n = 3$

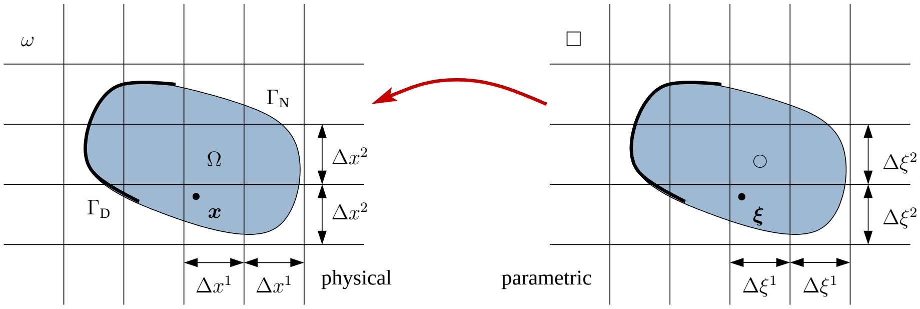


Figure 3: Cartesian domain ω and the physical domain Ω and its Dirichlet and Neumann boundaries Γ_{D} and Γ_{N}, respectively; the corresponding domains in the parametric domain are denoted with $\square$ and $\bigcirc$

throughout the following text. In the definition of $B_{\boldsymbol{i}}(\boldsymbol{\xi})$ we assume a renumbering of the basis functions defined in Eq. (3) of the form $B_{\boldsymbol{i}}(\boldsymbol{\xi}) = B^3_{\boldsymbol{j}}(\boldsymbol{\xi})$ with $\boldsymbol{j} = (i^1 - 2, \ldots, i^m - 2)$, so that $B_{\boldsymbol{i}}(\boldsymbol{\xi})$ is associated to the knot $\boldsymbol{\xi}_{\boldsymbol{i}}$ underneath its maximum. Thus, the displacements $\boldsymbol{u}_{\boldsymbol{i}}$ and test functions $\boldsymbol{v}_{\boldsymbol{i}}$ can be interpreted as vectors associated to node $\boldsymbol{i}$ of the Cartesian mesh. The set $\mathcal{C}$ is the collection of indices $\boldsymbol{i}$ and the total number of Cartesian nodes is $|\mathcal{C}|$.

Although the physical domain Ω is contained within ω, and thus the displacement field on Ω can be addressed in general, it is evident that the b-spline displacement interpolation does not conform to the physical domain boundary Γ. Hence, it is not directly suitable for discretizing the weak form in Equation (5). This is especially notable for the enforcement of Dirichlet boundary conditions. However, as will be discussed in forthcoming Section 4, it is possible to modify the b-spline basis functions so that they can accommodate the boundary.

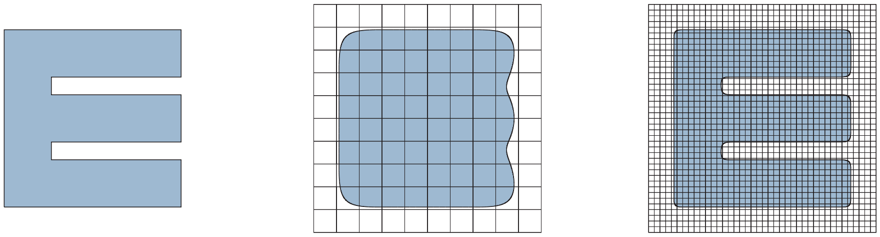
The weak form, Eq. (5), is expressed on the physical domain Ω and includes derivatives with respect to the physical coordinate $\boldsymbol{x} \in \Omega$. Thus a geometry approximation $\boldsymbol{x}(\boldsymbol{\xi})$ is required which can be established with the Cartesian b-splines at hand:

$$\boldsymbol{x}^{\mathrm{c}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}\in\mathcal{C}} B_{\boldsymbol{i}}(\boldsymbol{\xi})\, \boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}} \tag{7}$$

with nodal coordinates $\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}$ underneath the maxima of the basis functions. Due to the uniformity of the Cartesian b-splines, the nodal coordinate $\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}$ is determined by a linear mapping of the respective knot $\boldsymbol{\xi}_{\boldsymbol{i}}$. The linear mapping accounts for the physical dimensions of the Cartesian domain. For instance, the one-dimensional linear map is given by

$$x^c_i = \frac{x_{\max} - x_{\min}}{\max_j \xi_j - \min_j \xi_j}\, \xi_i + x_{\min} \tag{8}$$

with $\omega = (x_{\min}, x_{\max})$ and $\square = (\min_j \xi_j, \max_j \xi_j)$. In higher dimensions the mapping is applied to each coordinate component.



(a) Geometrically complex domain

(b) Coarse mesh

(c) Fine mesh

Figure 4: Level set representation of the domain shown in 4(a) on a coarse and a fine mesh 4(b) and 4(c), respectively

4. Immersed b-spline basis functions (i-splines)

4.1. Implicit shape representation

Before moving on to the discussion of the modified b-spline basis functions, we first elaborate on the representation of the physical domain on the Cartesian mesh. The proposed modification of the b-spline basis functions requires to identify all the cells close to the boundary Γ and if they are inside or outside the physical domain Ω. To this end, a computationally efficient and scalable approach is to use a signed distance function (or, level set function)

$$\phi(\boldsymbol{x}, \Gamma) = \begin{cases} \text{distance}(\boldsymbol{x}, \Gamma) & \text{if } \boldsymbol{x} \in \Omega \\ 0 & \text{if } \boldsymbol{x} \in \Gamma \\ -\text{distance}(\boldsymbol{x}, \Gamma) & \text{otherwise} \end{cases} \tag{9}$$

A discrete representation of the signed distance function can be obtained by combining the b-spline basis functions with the signed distance values at the nodes $\phi_{\boldsymbol{i}} = \phi(\boldsymbol{x}^{\text{c}}_{\boldsymbol{i}}, \Gamma)$ yielding

$$\phi(\boldsymbol{\xi}) = \sum B_{\boldsymbol{i}}(\boldsymbol{\xi}) \phi_{\boldsymbol{i}} \tag{10}$$

The interpolated level set $\phi(\boldsymbol{\xi}) = 0$ deviates in general from the original shape, because Equation (10) is an interpolation of the boundary based on a scalar field approximation using cubic b-spline basis functions. The interpolated boundary consists of the trace of the cubic tensor-product b-splines. For instance, this results in a smoothed boundary if the original boundary is given in terms of segments or facets. This effect is highlighted in Figure 4. Figure 4 reveals exemplary the captured level of details depending on the refinement of the Cartesian mesh. The arms of the original ‘E’ in Figure 4(a) are suppressed on the coarse mesh, Figure 4(b), but resolved on the (sufficiently) fine mesh, Figure 4(c). The filtering adds robustness to the overall method as only details are resolved in the dimension of the mesh size.

In contrast to the usual parametric mesh based boundary representations (using segments or facets), level set based representations are more suitable for problems with large deformations and topology changes. There are efficient and scalable algorithms for converting a mesh based representation into an implicit representation (e.g., closest point transform [18, 19]) and vice versa (e.g., marching cubes [20]).

Signed distance functions, also known as R-functions, allow very easy combination of body entities using Rvachev's Boolean operations, see for instance Rvachev et al. [21, 22] or Höllig [13]. The resulting signed distance function describes after combining several entities again a signed distance function, whose zero level set determines the resulting body shape. The combination can be intersections, unions etc. For instance, the subtraction of body Ω_{B} from Ω_{A} is achieved with $\phi_{\mathrm{A}\setminus\mathrm{B}} = \phi_{\mathrm{A}} - \phi_{\mathrm{B}} - \sqrt{\phi_{\mathrm{A}}^2 + \phi_{\mathrm{B}}^2}$ in which ϕ_{A} and ϕ_{B} are the signed distance functions of the initial bodies and the level set $\phi_{\mathrm{A}\setminus\mathrm{B}} = 0$ describes the boundary of the resulting body. Figure 5 shows the subtraction of a cross from a cube.

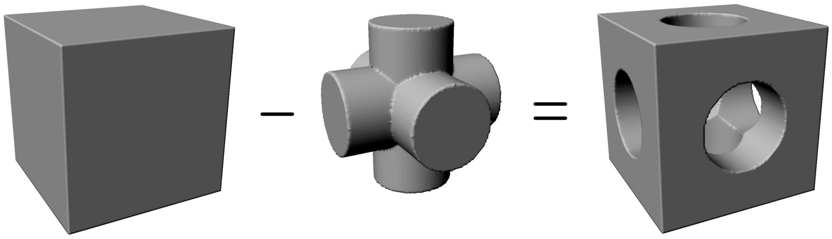


Figure 5: Boolean signed distance function operation

4.2. Normalised weighted b-spline (i-spline) basis functions

As discussed, the b-spline basis functions defined over the Cartesian mesh do not conform to the boundary of the physical domain. In particular, they are non-interpolating at Dirichlet boundaries and, hence, cannot be used for directly discretizing the weak form in Eq. (5). The imposition of Dirichlet boundary conditions cannot be carried out at the nodes as the approximate trial space does not satisfy the Dirichlet boundary conditions. In the following, we develop an approach for modifying the basis functions associated with the Cartesian mesh so that they are interpolating at the boundaries. First, we define similar to Höllig et al. [13] or Kantorovich et al. [14], a weight function which is zero outside the domain, one inside the domain and has a smooth transition zone in between

$$w(\boldsymbol{\xi}) = \begin{cases} 1 & \text{for } \phi(\boldsymbol{\xi}) > \delta \\ 1 - \left(1 - \frac{\phi(\boldsymbol{\xi})}{\delta}\right)^p & \text{for } 0 \leq \phi(\boldsymbol{\xi}) \leq \delta \\ 0 & \text{for } \phi(\boldsymbol{\xi}) < 0 \end{cases} \tag{11}$$

where $\phi(\boldsymbol{\xi})$ is the signed distance, δ is a transition length and p is an integer with $p \geq 1$ which controls the smoothness of the weight function inside the domain. Inside the domain, the weight function gradient is non-zero and C^{p-1}-continuous including at $\phi(\boldsymbol{\xi}) = \delta$. Advantageously, if we take $p \geq n$, the weight does not interfere with the C^{n-1}-continuity of the b-splines.

4.2.1. Classification of cells and nodes

All the Cartesian mesh cells are tagged as *physical*, *fictitious* or *boundary* depending on their position with respect to the physical domain. This classification is performed by computing for each cell $\square_e$ the minimum and maximum signed distance, respectively. The collected different cell types form sub-domains of the Cartesian domain, i.e.

$$\square_{\text{physical}} = \{\square_e \in \square \,|\, \min_{\boldsymbol{\xi} \in \square_e} \phi(\boldsymbol{\xi}) \geq 0\} \tag{12a}$$

$$\square_{\text{fictitious}} = \{\square_e \in \square \,|\, \max_{\boldsymbol{\xi} \in \square_e} \phi(\boldsymbol{\xi}) < 0\} \tag{12b}$$

$$\square_{\text{boundary}} = \square \setminus (\square_{\text{physical}} \cup \square_{\text{fictitious}}) \tag{12c}$$

The classification is summarised in Figure 6. This classification is unique as it is based on the signed distance function and not on the parametric representation of the physical domain (e.g., via a surface mesh).

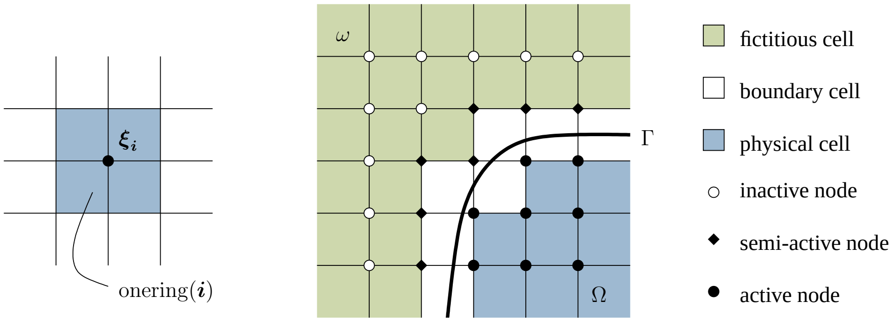


Figure 6: One-ring, and cell and node classification

The purpose of the cell tags is to identify and modify the basis functions which will be used for discretizing the weak form. To this purpose, all the Cartesian nodes and, hence, the associated basis functions are tagged as *active*, *inactive* or *semi-active*. This classification is based on the tags of all the cells which surround a node $\boldsymbol{i}$, i.e. the *one-ring of the node* denoted as onering($\boldsymbol{i}$). A two-dimensional one-ring is depicted in Figure 6 on the left. We group all active node indices (or related basis functions) in $\mathcal{A}$, all inactive in $\mathcal{I}$ and all semi-active in $\mathcal{S}$ which are given by

$$\mathcal{A} = \{\boldsymbol{i} \in \mathcal{C} \,|\, \operatorname{onering}(\boldsymbol{i}) \subset (\square_{\text{physical}} \cup \square_{\text{boundary}})\} \tag{13a}$$

$$\mathcal{I} = \{\boldsymbol{i} \in \mathcal{C} \,|\, \operatorname{onering}(\boldsymbol{i}) \subset \square_{\text{fictitious}}\} \tag{13b}$$

$$\mathcal{S} = \mathcal{C} \setminus (\mathcal{A} \cup \mathcal{I}) \tag{13c}$$

The node classification is shown in Figure 6. It is observable that an active node contains only physical *or* boundary cells in its one-ring. However, a semi-active node is characterised by boundary *and* fictitious cells in its one-ring. The remaining inactive nodes are only surrounded by fictitious cells.

4.2.2. Construction of i-spline basis functions

The preceding definitions are used for defining a weighting function $z_{\boldsymbol{i}}$ associated with each node (or, basis function) $\boldsymbol{i}$ of the Cartesian mesh. Depending on the classification, the nodal weighting function is set to

$$z_{\boldsymbol{i}}(\boldsymbol{\xi}) = \begin{cases} w(\boldsymbol{\xi}) & \text{if } \boldsymbol{i} \in \mathcal{A}, \text{ i.e. node is active} \\ 1 & \text{if } \boldsymbol{i} \in \mathcal{S}, \text{ i.e. node is semi-active} \\ 0 & \text{if } \boldsymbol{i} \in \mathcal{I}, \text{ i.e. node is inactive} \end{cases} \tag{14}$$

The weight function $z_{\boldsymbol{i}}$ is a scalar field attached to each Cartesian node (or basis function) $\boldsymbol{i}$. The application of the weighting functions does not modify the b-splines $z_{\boldsymbol{i}}B_{\boldsymbol{i}}$ in the core of the domain Ω, because w attains the value 1 there. However, the b-splines are altered whose support is intersected by the boundary. Those reaching their maximum *inside* of Ω are forced to vanish on the boundary and outside of Ω. The semi-active b-splines, which reach their maximum *just outside* of Ω, are kept unchanged. The basis functions being even farther outside are removed. The removal includes basis functions that have only small support in the physical domain, which may lead to badly conditioned systems and stability problems.

Finally, the nodally weighted b-splines are adapted to yield a set of basis functions interpolating at the immersed boundary, i.e. which are 1 on the boundary. This scaling is achieved by normalising the weighted b-splines $z_{\boldsymbol{i}}B_{\boldsymbol{i}}$. The normalisation of the weighted b-splines yields

$$N_{\boldsymbol{i}}(\boldsymbol{\xi}) = \frac{z_{\boldsymbol{i}}(\boldsymbol{\xi})\ B_{\boldsymbol{i}}(\boldsymbol{\xi})}{\sum_{\boldsymbol{j}} z_{\boldsymbol{j}}(\boldsymbol{\xi})\ B_{\boldsymbol{j}}(\boldsymbol{\xi})}\,. \tag{15}$$

The normalisation does not only ensure $\big(\sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\big)|_{\boldsymbol{\xi}\in\partial\bigcirc} = 1$, but also establishes a partition of unity. For the sake of brevity, these rationalised weighted b-splines are abbreviated with *i-splines*. Only the active and semi-active i-splines are of interest, since inactivity leads to $N_{\boldsymbol{i}} = 0$ for any $\boldsymbol{i} \in \mathcal{I}$.

The derived rationalised weighted b-splines span the approximate test space

$$\mathcal{V}^{\text{h}} = \{\boldsymbol{v}^{\text{h}} \in C^{n-1}(\bar{\bigcirc}, \mathbb{R}^m) \,|\, \boldsymbol{v}^{\text{h}}(\boldsymbol{\xi}) = \textstyle\sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}_{\text{N}}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\boldsymbol{v}_{\boldsymbol{i}}\} \tag{16}$$

on the closed parametric domain $\bar{\bigcirc}$ of the body, which would satisfy homogeneous Dirichlet boundary conditions. The semi-active i-splines are separated in two subsets $\mathcal{S}_{\mathrm{D}} \cup \mathcal{S}_{\mathrm{N}} = \mathcal{S}$. The basis functions $N_{\boldsymbol{i}} \in \mathcal{S}_{\mathrm{D}}$ achieve their maxima on the Dirichlet boundary. The remaining semi-active basis functions are in $\mathcal{S}_{\mathrm{N}}$. Satisfaction of in-homogeneous boundary conditions can be added to the approximate trial space

$$\mathcal{U}^{\mathrm{h}} = \mathcal{V}^{\mathrm{h}} \oplus \{\boldsymbol{u}^{\mathrm{h}} \in C^{n-1}(\bar{\bigcirc}, \mathbb{R}^m) \,|\, \boldsymbol{u}^{\mathrm{h}}(\boldsymbol{\xi}) = \textstyle\sum_{\boldsymbol{i}\in\mathcal{S}_{\mathrm{D}}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\boldsymbol{u}_{\boldsymbol{i}}\} \tag{17}$$

using the semi-active basis functions in $\mathcal{S}_{\mathrm{D}}$. The latter allow to impose the Dirichlet boundary conditions at the nodes $\boldsymbol{i} \in \mathcal{S}_{\mathrm{D}}$ like in ordinary finite elements based on Lagrangian polynomials, or by performing, a least-square fit with respect to the Dirichlet data. Similar to Lagrangian finite elements, an overlap region $\mathrm{supp}_{\boldsymbol{i}\in\mathcal{S}_{\mathrm{D}}}(N_{\boldsymbol{i}}) \cap \mathrm{supp}_{\boldsymbol{j}\in\mathcal{S}_{\mathrm{N}}}(N_{\boldsymbol{j}}) \neq \emptyset$ exists. In this region the prescribed displacements blend with the unknown displacements.

In summary, the i-spline basis is used to interpolate the displacements $\boldsymbol{u}$ and test functions $\boldsymbol{v}$, cf. Equation (6),

$$\boldsymbol{u}^{\mathrm{h}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\,\boldsymbol{u}_{\boldsymbol{i}} \quad \text{and} \quad \boldsymbol{v}^{\mathrm{h}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}_{\mathrm{N}}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\,\boldsymbol{v}_{\boldsymbol{i}}\,, \tag{18}$$

since this approach allows to enforce the Dirichlet boundary conditions in a local way.

Illustration of one-dimensional i-splines. The construction process of the i-spline basis is illustrated with a one-dimensional model problem. Figures 7 to 9 show the construction steps of the modified basis functions for a one-dimensional domain $\bigcirc = (0.9, 5.4)$ on the Cartesian domain $\square = (-3, 9)$.

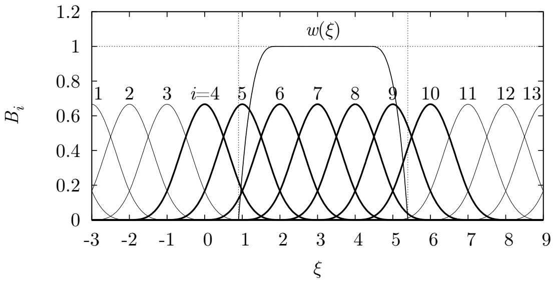


Figure 7: Model problem 1D: Cubic b-splines and a smooth weight function $w(\xi)$ (with $\delta = 1$, $p = 3$)

In Figure 7 the ordinary cubic b-splines are depicted, in which the relevant basis functions $i = 5, \ldots, 10$ in the immersed domain $\bigcirc = (0.9, 5.4)$ are drawn with thicker lines. The weighting function $w(x)$ with $\delta = 1$ and $p = 3$ is shown as well. After multiplication with the weight functions $z_i(\xi)$ the shape of the functions $i = 5, \ldots, 9$ are modified, cf. Figure 8. The support is reduced of the

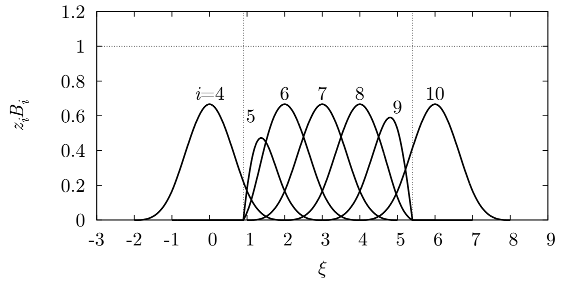


Figure 8: Model problem 1D: Weighted basis functions

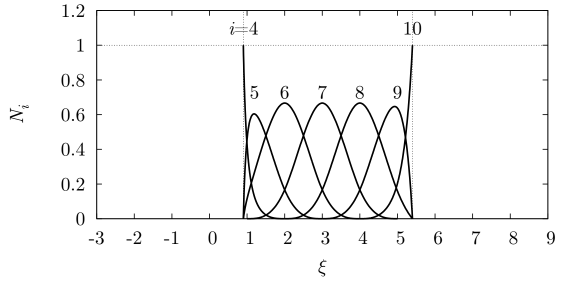
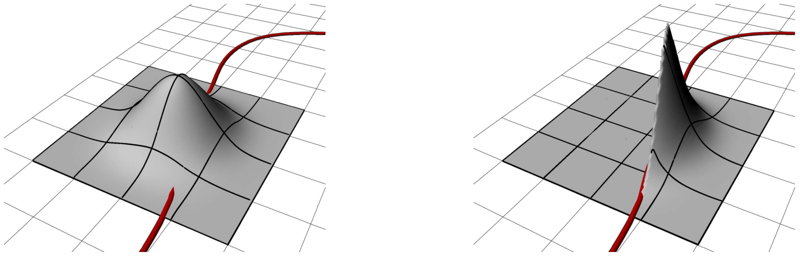


Figure 9: Model problem 1D: I-spline basis functions computed with the proposed approach. Note that the modified basis functions are interpolating at the boundaries.

functions close the boundary, namely z_5B_5, z_6B_6, z_8B_8 and z_9B_9. The basis functions z_4B_4 and $z_{10}B_{10}$ are kept unaltered. The support of at least one Cartesian integration cell is clearly observable. The two basis functions B_3 and B_{11}, which have a small support in the physical domain are removed. Finally, the normalisation elevates the semi-active basis functions z_4B_4 and $z_{10}B_{10}$ to one on the immersed boundary, see Figure 9. The process also scales the active basis functions $z_5B_5, \ldots, z_9B_9$. Although the central basis function N_7 is slightly changed in the depicted situation, it is straightforward to imagine a large enough region, in which core basis functions would remain literally unchanged.

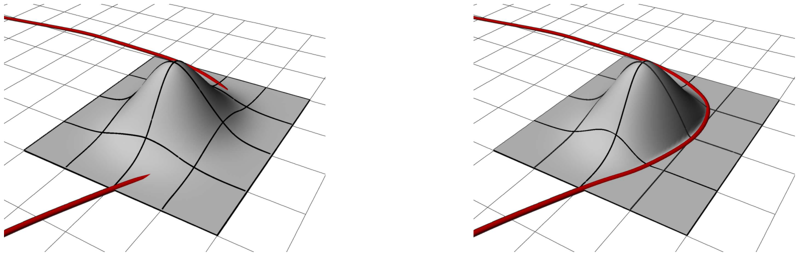
Illustration of two-dimensional i-splines. The overall modification of a semi-active and an active two-dimensional bi-cubic b-spline is epitomised in Figures 10 and 11. In Figure 10(a) and 11(a) two bi-cubic b-splines are shown whose supports intersect with the immersed two-dimensional domain. The immersed boundary is highlighted with a red string. The red string is not visible, where the b-spline overlaps the boundary. The b-spline in Fig. 10(a) achieves its maximum just outside the immersed domain, whereas the b-splines depicted in Fig. 11(a) achieves it in the interior. The former b-spline is a semi-active basis function which is altered to match the boundary, leading to the i-spline presented in Fig. 10(b). The active b-spline of Fig. 11(a) is also adapted to the boundary, cf. Fig. 11(b). The basis function is only C^0-continuous across the



(a) Semi-active b-spline

(b) Semi-active i-spline

Figure 10: Semi-active original bi-cubic b-splines function and its i-spline counterpart



(a) Active b-spline

(b) Active i-spline

Figure 11: Active original bi-cubic b-splines functions and its i-spline counterpart

boundary. This can be noticed looking at the kinks of the superimposed grid.

4.2.3. Properties of i-splines

We want to summarise a few important properties of the rationalised weighted b-splines of the relevant active and semi-active kind.

- *Positivity and single maximum.* The original uniform cubic b-spline $B_{\boldsymbol{i}}$ is positive on its support and is characterised by a single maximum at the centre of its support. The weight function w, Eq. (11), shows the same behaviour on $\bigcirc$. The multiplication of these two functions with non-decreasing first derivative leads again to a function with a single maximum. Therefore, the weighted spline $z_{\boldsymbol{i}} B_{\boldsymbol{i}}$ is positive and assumes a single maximum for $\boldsymbol{i} \in \mathcal{A} \cup \mathcal{S}$ in the immersed domain. Finally, the i-spline $N_{\boldsymbol{i}}$ behaves identically as the sum of positive $z_{\boldsymbol{i}} B_{\boldsymbol{i}}$ is positive as well.

- *Local support.* The i-splines are constructed by multiplying the corresponding b-splines with a scalar field. The b-splines possess local supports, thus the i-splines must have $\operatorname{supp}(N_{\boldsymbol{i}}) \subseteq \operatorname{supp}(B_{\boldsymbol{i}})$ for $\boldsymbol{i} \in \mathcal{A} \cup \mathcal{S}$.

- *Continuity.* An appropriate choice of the weight function Eq. (11) together with interpolated signed distance function Eq. (10) preserves the C^2-continuity of the cubic b-splines. However, in general a different choice of either the weight or signed distance function may lead to only continuous basis functions.

- *Local satisfaction of the Dirichlet boundary conditions.* This relation is explained in Section 4.2.
- *Partition of unity.* The basis functions form a partition of unity by construction. The normalisation approach of Eq. (15) provides inherently

$$\sum_{\boldsymbol{i}} N_{\boldsymbol{i}} = \sum_{\boldsymbol{i}} \left(\frac{z_{\boldsymbol{i}} B_{\boldsymbol{i}}}{\sum_{\boldsymbol{j}} z_{\boldsymbol{j}} B_{\boldsymbol{j}}} \right) = \frac{\sum_{\boldsymbol{i}} z_{\boldsymbol{i}} B_{\boldsymbol{i}}}{\sum_{\boldsymbol{j}} z_{\boldsymbol{j}} B_{\boldsymbol{j}}} = 1 \tag{19}$$

- *Linear independence.* The b-splines $B_{\boldsymbol{i}}$ are linearly independent. The i-splines $N_{\boldsymbol{i}}$ are effectively created by multiplication with $W_{\boldsymbol{i}}(\boldsymbol{\xi}) = \frac{z_{\boldsymbol{i}}}{\sum_{\boldsymbol{j}} z_{\boldsymbol{j}} B_{\boldsymbol{j}}}$. Since $W_{\boldsymbol{i}} > 0$ and $W_{\boldsymbol{i}} \neq W_{\boldsymbol{j}}$ the i-splines must be linearly independent too.

4.3. Iso-parametric geometry representation of the immersed domain

The i-spline basis is not a polynomial basis, therefore, it does not approximate linear functions over the parametric domain $\bigcirc$. This means the displacement field $\boldsymbol{u}^{\mathrm{h}}(\boldsymbol{x}) = \boldsymbol{u}^{\mathrm{h}}(\boldsymbol{\xi}(\boldsymbol{x}^{\mathrm{c}}))$ with the introduced inverted geometry approximation $\boldsymbol{\xi}(\boldsymbol{x}^{\mathrm{c}}) = [\boldsymbol{x}^{\mathrm{c}}(\boldsymbol{\xi})]^{-1}$ due to Equation (7) does not contain linear displacement fields. However, this can be remedied by changing the geometry approximation using an *iso-parametric approach.* The iso-parametric approach together with the partition of unity property enables the approximation of constant and linear functions, cf. Appendix A.

Several alternatives are available to create an iso-parametric geometry description, i.e.

$$\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i} \in \mathcal{A} \cup \mathcal{S}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\, \boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}} \tag{20}$$

in which *suitable* nodal positions $\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}$ occur. In Section 4.3.1 a simple, yet well working, approach is presented based on closest points on the boundary. This method was implemented and employed for the test examples. For the sake of completeness, a few alternatives to this approach are shortly summarised in Sections 4.3.2 and 4.3.3.

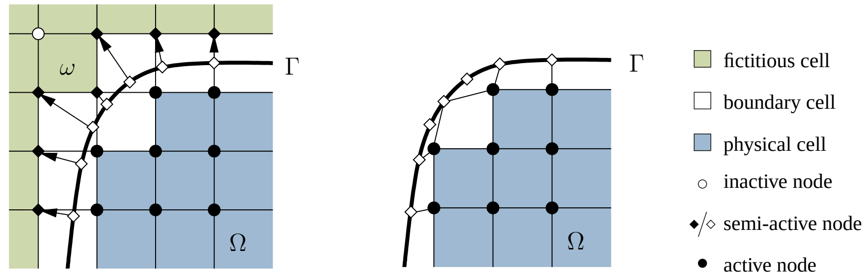
4.3.1. Closest points on boundary

This approach assigns to a semi-active node $\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}$ its closest point $\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}$ on the boundary Γ. The closest point is the new coordinate associated to $N_{\boldsymbol{i}}$, with $\boldsymbol{i} \in \mathcal{S}$, i.e.

$$\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}} = \begin{cases} \min_{\boldsymbol{x} \in \Gamma} \|\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}} - \boldsymbol{x}\| & \text{if } \boldsymbol{i} \in \mathcal{S} \\ \boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}} & \text{if } \boldsymbol{i} \in \mathcal{A} \end{cases} \tag{21}$$

All remaining basis functions $\boldsymbol{i} \in \mathcal{A}$ use simply the Cartesian grid coordinate $\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}$. The closest points $\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}$ can be conveniently estimated with

$$\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}} \approx \boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}} + \phi(\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}, \Gamma)\, \boldsymbol{n}(\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}) \quad \text{where} \quad \boldsymbol{n}(\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}) = -\frac{\nabla\phi(\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}, \Gamma)}{\|\nabla\phi(\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}, \Gamma)\|}\,, \tag{22}$$



(a) Closest points of semi-active Cartesian nodes

(b) Mesh after change to iso-parametric node coordinates

Figure 12: Mesh modification for establishing an iso-parametric approach

taking advantage of the implicit shape definition based on the signed distance function.

Figure 12 sketches the new iso-parametric node coordinates in relation to the Cartesian coordinates. The closest points of the semi-active nodes on the boundary are highlighted in Fig. 12(a). The iso-parametric mesh possesses non-rectangular cells to account for the boundary cells, cf. 12(b). Since the inactive and semi-active Cartesian nodes are not needed to approximate the geometry, it is convenient to store the coordinates of iso-parametric semi-active nodes on the Cartesian semi-active counterparts which simplifies the implementation.

It should be highlighted that the change to the iso-parametric geometry does not modify the i-spline basis functions. The i-splines are still defined on the Cartesian grid subject to the immersed domain. Thus the change in geometry approximation affects only the integration and differentiation in the usual iso-parametric sense, e.g.

$$N_{\boldsymbol{i},\boldsymbol{x}} = N_{\boldsymbol{i},\boldsymbol{\xi}} \cdot (\boldsymbol{x}^{\mathrm{h}}_{,\boldsymbol{\xi}})^{-1} \quad \text{with } \boldsymbol{\xi} \in \bigcirc\,, \tag{23}$$

with respect to the physical domain.

4.3.2. Least squares fitting

A straight-forward, different approach to determine a set of iso-parametric node coordinates $\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}$ is a least squares fitting. Looking at the square of the difference in the geometry field approximations leads to the least-squares functional

$$R(\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}) = \int_{\bigcirc} \left(\boldsymbol{x}^{\mathrm{h}} - \boldsymbol{x}^{\mathrm{c}}\right)^2 \mathrm{d}\bigcirc = \int_{\bigcirc} \Big(\sum_{\boldsymbol{i}\in\mathcal{C}} \left(N_{\boldsymbol{i}}\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}} - B_{\boldsymbol{i}}\boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}}\right)\Big)^2 \mathrm{d}\bigcirc\,. \tag{24}$$

The minimisation of this functional with respect to the nodal positions $\boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}$ results in $|\mathcal{A}\cup\mathcal{S}|$ vector equations

$$\frac{\partial R}{\partial \boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}}} = \mathbf{0} \qquad \Longrightarrow \qquad \sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}} \int N_{\boldsymbol{j}} N_{\boldsymbol{i}}\, \mathrm{d}\bigcirc\, \boldsymbol{x}^{\mathrm{h}}_{\boldsymbol{i}} = \sum_{\boldsymbol{i}\in\mathcal{C}} \int N_{\boldsymbol{j}} B_{\boldsymbol{i}} \mathrm{d}\bigcirc\, \boldsymbol{x}^{\mathrm{c}}_{\boldsymbol{i}} \tag{25}$$

The method has the drawback to violate the interpolation on the boundary, since the residuum is minimised on the whole body domain, contradicting the original target. The linear system can be reduced by setting $\boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{h}} = \boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{c}}$ for all active nodes $\boldsymbol{i} \in \mathcal{A}$ only computing the semi-active node coordinates. Alternatively, only the geometry difference over the domain boundary could be considered.

4.3.3. At maxima

Another possibility to determine an appropriate set of coordinates $\boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{h}}$ can be based on the maxima of the i-splines, i.e.

$$\boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{h}} = \sum_{\boldsymbol{j}} B_{\boldsymbol{j}}(\boldsymbol{\xi}_{\boldsymbol{i}}^{\max})\boldsymbol{x}_{\boldsymbol{j}}^{\mathrm{c}} \quad \text{with} \quad \boldsymbol{\xi}_{\boldsymbol{i}}^{\max} = \max_{\boldsymbol{\xi} \in \bigcirc} N_{\boldsymbol{i}}(\boldsymbol{\xi}) \,. \tag{26}$$

The maximum of an i-spline can be easily detected and it is a priori known that only a single maximum exists for each i-spline on the domain $\bar{\bigcirc}$. In terms of computing the maxima, it appears convenient to convert (26) in a constraint minimisation problem,

$$\boldsymbol{\xi}_{\boldsymbol{i}}^{\max} = \min_{\substack{\boldsymbol{\xi} \in \operatorname{supp} B_{\boldsymbol{i}} \\ \phi(\boldsymbol{\xi}) \geq 0}} \big(- N_{\boldsymbol{i}}(\boldsymbol{\xi}) \big) \,, \tag{27}$$

which limits the search region by incorporating the effective domain with the signed distance function. Suitable algorithms can be found, e.g., in Nocedal and Wright [23]. The parametric maximum $\boldsymbol{\xi}_{\boldsymbol{i}}$ of the related original cubic b-spline provides a good initial guess for iterative algorithms.

4.3.4. Numerical comparisons

In Figure 13, the results of the presented identification methods of the iso-parametric nodes are depicted for the one-dimensional model problem on $\bar{\bigcirc} = [0.9, 5.4]$. The closest point approach is denoted with CP, least squares fitting with LS and underneath the maxima with AM. The reference solution is $x = \xi$. The least squares fitting violates the boundary which can be observed by

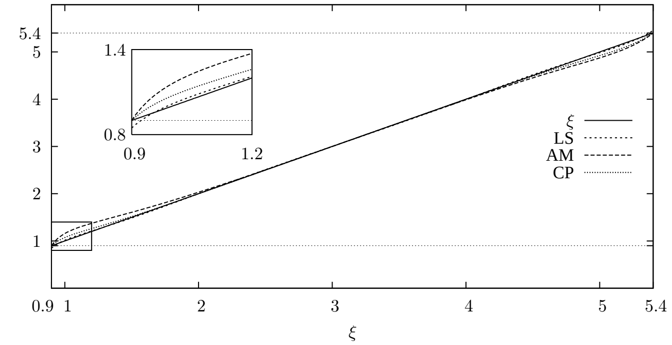


Figure 13: Iso-parametric geometry approximation of 1D model problem

looking at the magnified, boxed region. The AM curve satisfies the boundary, but exhibits relatively large deviation from the linear reference curve. Overall, the closest point approach appears to produce the best result. It ought to be highlighted, in general, all presented approximations are viable since they provide a strictly positive gradient $x_{,\xi}$. In essence, the positivity and the single maximum of the i-splines behaves rather good-natured towards modification of the nodal Cartesian coordinates as long as $x_i^{\mathrm{h}} < x_{i+1}^{\mathrm{h}}$.

Although, the two-dimensional iso-parametric mesh shown in Figure 12 is based on the closest point method, the least square fitting or AM technique would result in similarly distributed iso-parametric meshes. The shape of the iso-parametric mesh is a consequence of the iso-parametric geometry approximation based on i-splines rather than of the method to determine the iso-parametric node coordinates.

4.4. Quadrature

The integrals in the weak form (5) are given on the physical body domain Ω and its Neumann boundary Γ_{N}. Due to the (iso-)parametric description of the geometry the integrals are mapped to the parametric body domain $\bigcirc$. This is a sub-domain on which the cubic b-splines are described; the b-splines and the i-splines share the parameters $\boldsymbol{\xi}$. For instance, the integral of the external body forces results in

$$\begin{aligned}\int_{\Omega} \boldsymbol{v}^{\mathrm{h}} \cdot \boldsymbol{b}(\boldsymbol{x})\,\mathrm{d}\Omega &= \int_{\bigcirc} \boldsymbol{v}^{\mathrm{h}}(\boldsymbol{\xi}) \cdot \boldsymbol{b}(\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi}))\ \det(\boldsymbol{x}^{\mathrm{h}}_{,\boldsymbol{\xi}})\,\mathrm{d}\bigcirc \\ &= \sum_{e} \sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}_{\mathrm{N}}} \boldsymbol{v}_{\boldsymbol{i}} \cdot \int_{\bigcirc_e} N_{\boldsymbol{i}}(\boldsymbol{\xi})\,\boldsymbol{b}(\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi}))\ \det(\boldsymbol{x}^{\mathrm{h}}_{,\boldsymbol{\xi}})\,\mathrm{d}\bigcirc\,,\end{aligned} \tag{28}$$

in which the parametric body domain $\bigcirc = \bigcup_e \bigcirc_e$ is the collection of the integration cells $\bigcirc_e$. The cells are sketched on the left in Figure 14.

Although it is possible to integrate on the physical cells, for which $\bigcirc_e = \square_e$, directly with an ordinary Gauss quadrature, this cannot be performed on boundary cells characterised by $\bigcirc_e \subset \square_e$. However, the iso-parametric geometry mapping allows to consider each boundary cell as a re-shaped Cartesian cell. Additional element-wise parameters $\boldsymbol{s}_e$ are introduced on each element, cf. Figure 14, and mapped to $\boldsymbol{\xi}$ using the cubic b-splines

$$\boldsymbol{\xi}(\boldsymbol{s}_e) = \sum_{\boldsymbol{i}\in\mathcal{K}_e} B_{\boldsymbol{i}}(\boldsymbol{s}_e)\,\boldsymbol{\xi}_{\boldsymbol{i}}^{\mathrm{h}} \quad \text{with} \quad \boldsymbol{\xi}_{\boldsymbol{i}}^{\mathrm{h}} = \boldsymbol{\xi}_{\boldsymbol{i}} + \boldsymbol{G}\cdot\left(\boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{h}} - \boldsymbol{x}_{\boldsymbol{i}}^{\mathrm{c}}\right) \quad \text{and} \quad \boldsymbol{s}_e \in \square_e\,, \tag{29}$$

where $\mathcal{K}_e = \{\boldsymbol{i} \in \mathcal{A}\cup\mathcal{S} \,|\, \mathrm{supp}(B_{\boldsymbol{i}}) \cap \square_e \neq \emptyset\}$ and in two dimensions $\boldsymbol{G} = \mathrm{diag}(\frac{\Delta\xi^1}{\Delta x^1}, \frac{\Delta\xi^2}{\Delta x^2})$ with physical and parametric Cartesian mesh sizes Δx^1, Δx^2 and $\Delta\xi^1 = \Delta\xi^2 = 1$, respectively, see also Fig. 3. The one- and three-dimensional case follow analogously. Alternatively, the cubic b-splines can be replaced by linear Lagrangian polynomials, however, the numerical examples led to better results using cubic b-splines. The map $\boldsymbol{\xi}(\boldsymbol{s}_e)$ behaves good-natured

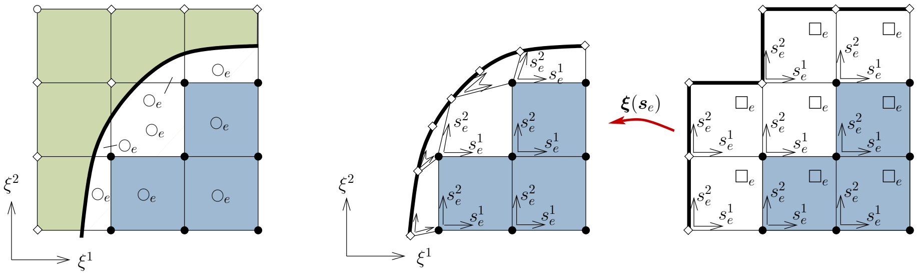


Figure 14: Parametric coordinates for integration

due to the smoothly interpolated level set, Eq. (10), which prevents pathologically shaped cells. Degenerated cells, i.e. cells taking on the shape of a triangle for instance, are tolerated by the mapping. The latter parameters are only used to integrate on the correct domain, they are not necessary to express physical gradients, because the basis functions are given over $\boldsymbol{\xi}$. Therefore, the element-wise parameters merely enable an individual Gauss quadrature on the re-shaped element domain. The integral of the body forces yields

$$\int_\Omega \boldsymbol{v}^{\mathrm{h}} \cdot \boldsymbol{b}(\boldsymbol{x}) \,\mathrm{d}\Omega = \sum_e \sum_{\boldsymbol{i}\in\mathcal{A}\cup\mathcal{S}_{\mathrm{N}}} \boldsymbol{v}_{\boldsymbol{i}} \cdot \sum_g \left(N_{\boldsymbol{i}}(\boldsymbol{\xi})\, \boldsymbol{b}(\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi}))\, \det(\boldsymbol{x}^{\mathrm{h}}_{,\boldsymbol{\xi}})\right)\Big|_{\boldsymbol{\xi}=\tilde{\boldsymbol{\xi}}_g} \tilde{\omega}_g \,, \quad (30)$$

with the Gauss points and weights of the re-shaped rectangle

$$\tilde{\boldsymbol{\xi}}_g = \boldsymbol{\xi}(\tilde{\boldsymbol{s}}_g) \quad \text{and} \quad \tilde{\omega}_g = \det(\boldsymbol{\xi}_{,\boldsymbol{s}_e})\, \tilde{w}_g \,. \quad (31)$$

Here, $\tilde{\boldsymbol{s}}_g$ and $\tilde{w}_g$ are the well-known Gauss points and weights of a rectilinear domain.

Integration over the boundary of the body to incorporate Neumann boundary conditions is carried out over the element directions of $\boldsymbol{s}_e$ which are tangential to the boundary. In Figure 14, these edges are highlighted with a thick line.

5. Examples

We proceed to establish the convergence characteristics and robustness of the proposed method by means of one-, two- and three-dimensional elasticity and heat transfer boundary value problems. The following error norms are used for the subsequent convergence tests

$$e^{\mathrm{h}}_{L_2} = \|\boldsymbol{u}^{\mathrm{h}} - \boldsymbol{u}\|_{L_2(\Omega)} = \sqrt{\int_\Omega \left(\boldsymbol{u}^{\mathrm{h}} - \boldsymbol{u}\right) \cdot \left(\boldsymbol{u}^{\mathrm{h}} - \boldsymbol{u}\right) \mathrm{d}\Omega} \quad (32)$$

and

$$e^{\mathrm{h}}_{H_1} = |\boldsymbol{u}^{\mathrm{h}} - \boldsymbol{u}|_{H_1(\Omega)} = \sqrt{\int_\Omega \left(\nabla\boldsymbol{u}^{\mathrm{h}} - \nabla\boldsymbol{u}\right) : \left(\nabla\boldsymbol{u}^{\mathrm{h}} - \nabla\boldsymbol{u}\right) \mathrm{d}\Omega} \quad (33)$$

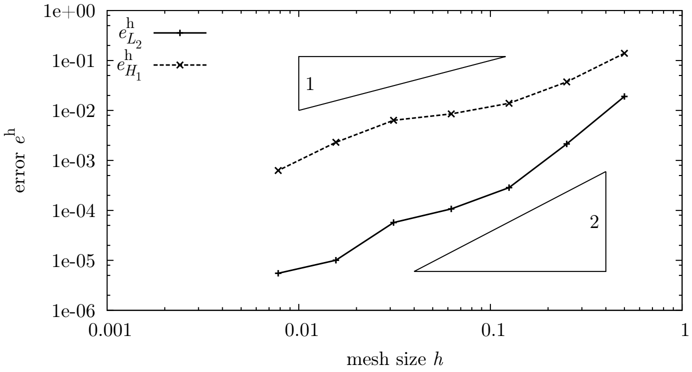


Figure 15: Convergence of one-dimensional bar

5.1. One-dimensional bar

The convergence of the proposed approach is first studied with a one-dimensional bar

$$\begin{aligned} \frac{\mathrm{d}^2 u}{\mathrm{d}x^2} &= -x \quad \text{in } \Omega = (a,b) \\ u &= 0 \qquad \text{on } \Gamma_\mathrm{D} = \{a,b\} \end{aligned} \tag{34}$$

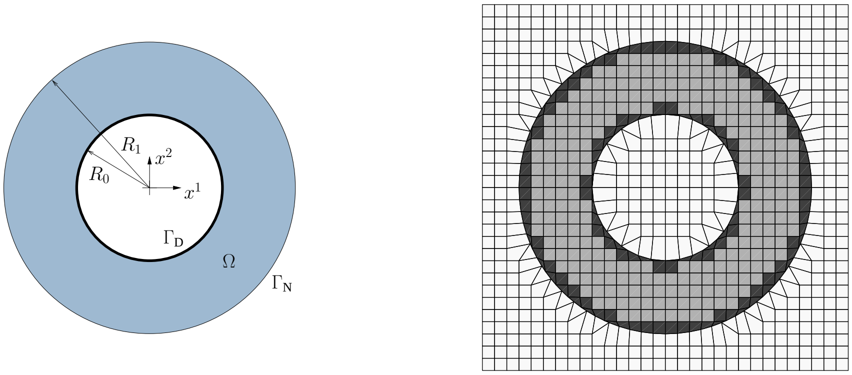
with $a = 0.9$ and $b = 5.4$ and the analytical solution

$$u(x) = -\frac{1}{6}\big(x^3 - (b^2 + ab + a^2)x + ab^2 + a^2 b\big) \tag{35}$$

On this simple domain, a parabola is used as a weight function. It is zero on the boundary, smooth, and positive (with a maximum of 1) inside the domain.

$$w(\xi) = \frac{4}{(b-a)^2}\big(\xi - a\big)\big(b - \xi\big) \tag{36}$$

The iso-parametric mesh points are determined with the closest point approach described in Section 4.3.1. This means that the left and right semi-active basis functions are simply associated with boundary points a and b, respectively. In computations, 10 Gauss-Legendre points per cell were used in order to exclude the influence of integration errors on the convergence. Figure 15 shows the convergence of the error between the numerical and analytical solution with decreasing mesh size. The error $e^\mathrm{h}_{L_2}$ decays about quadratically and the error $e^\mathrm{h}_{H_1}$ decays about linearly. This is in line with the expected convergence rate of a method, which can interpolate linears and constants exactly. However, the optimal convergence rate of a complete cubic polynomial interpolation is not achieved, which can be explained with the rational i-spline basis functions close to the boundaries.



(a) Geometry

(b) Mesh

Figure 16: Ring geometry and its discretization with 30 × 30 elements

5.2. Ring

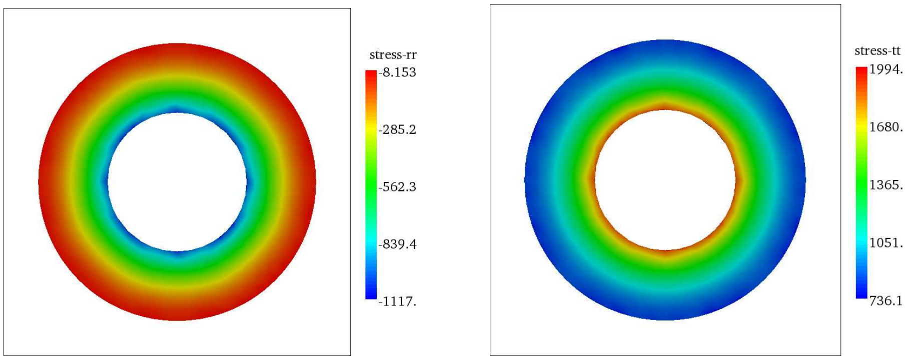
As a two-dimensional example, we consider the simple case of a linear elastic disk with an external radius $R_1 = 1$ and an internal radius $R_0 = 0.5$, cf. Figure 16(a). The disk is subjected to an internal radial displacement of $\bar{u}_r = 0.1$ and an external boundary traction of zero. A body force is not applied. The Young's modulus is $E = 10000$ and the Poisson's ratio $\nu = 0$. A plane stress state is assumed. The analytic solution for this radially symmetric example in Cartesian components is

$$\boldsymbol{u} = \begin{bmatrix} u_r \cos\theta \\ u_r \sin\theta \end{bmatrix} \quad \text{with} \quad u_r = \frac{R_0 \bar{u}_r}{R_0^2 + R_1^2}\Big(\frac{R_1^2}{r} + r\Big) \quad \text{and} \quad \left\{ \begin{array}{l} r = \sqrt{(x^1)^2 + (x^2)^2} \\ \theta = \tan(x^2/x^1) \end{array} \right. \tag{37}$$

which can be found in standard textbooks on elasticity, such as Timoshenko [24].

The size of the Cartesian domain has been chosen as 2.5 × 2.5 so that it is slightly larger than the outer disk diameter, see Figure 16(b); in which a 30 × 30 (iso-parametric) mesh is drawn as well. The cells indicated with a light gray are 252 physical cells, whereas the dark gray marks the 144 boundary cells. The remaining white cells are fictitious. The numerical integration was performed with 3 × 3 Gauss–Legendre quadrature points. The radial and tangential stresses are depicted in Figure 17(a) and 17(b). The rotational symmetry of the stress components is clearly reproduced.

One key component of the i-splines is the weight function w which specifically forces basis functions to vanish on the boundary. The employed weight function, Eq. (11), possesses two parameters: the power p and the transition length δ. The power was fixed to $p = 3$ in relation to the smoothness of cubic b-splines. The transition length was altered in dependence of the Cartesian mesh size h and the error in the solution was monitored for different meshes. The outcome is presented in Figure 18. Increasingly short transition lengths have a negative influence on the solution quality. In absolute terms the smallest



(a) Radial stress

(b) Tangential stress

Figure 17: Computed stress contours

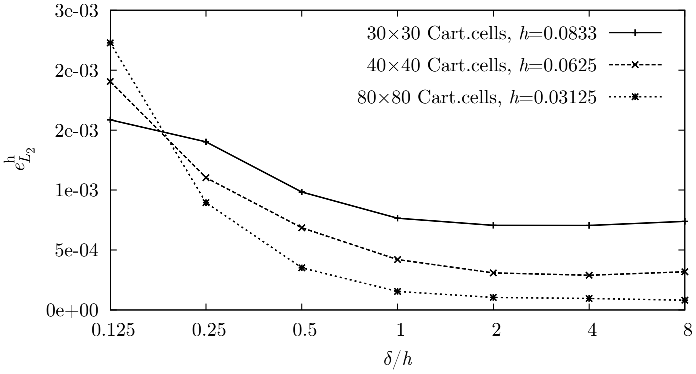


Figure 18: Influence of weight function transition length

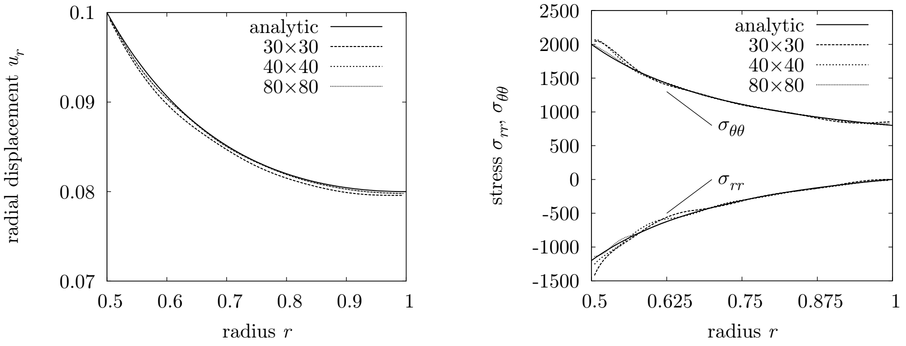


Figure 19: Radial displacement and polar stresses on horizontal line

computed transition length amounts to only $\delta = 1.04 \cdot 10^{-2}$ (mesh 30×30) or $\delta = 3.9 \cdot 10^{-3}$ (mesh 80×80) compared to the external radius of $R_1 = 1$. This implies a predominantly local adjustment of the original b-splines at the immersed boundary. A moderate increase of the transition length to $\delta \approx h$ improves the solution quality while ensuring a local modification of the basis functions at the boundary.

Figure 19 depicts the radial displacement and polar stresses over the horizontal axis, cf. Figure 16. The radial displacement u_r appeared very close to the analytic solution, Eq. (37). Although, the stress evolutions σ_{rr} and $\sigma_{\theta\theta}$ exhibited slight deviations on the inner radius, the solution appeared faithful otherwise.

A series of different mesh refinements was computed to study the convergence rate in the $e^{\mathrm{h}}_{L_2}$ and $e^{\mathrm{h}}_{H_1}$ errors, see Figure 20. The coarsest mesh is 20×20 with $h = 0.125$ and the finest amounts to 140×140; the transition length was chosen $\delta = 2h$ and a 4×4 Gauss quadrature was employed. Figure 20 presents the evolution of the $e^{\mathrm{h}}_{L_2}$ and $e^{\mathrm{h}}_{H_1}$ error versus the Cartesian mesh size h. The convergence rate of $e^{\mathrm{h}}_{L_2}$ appears to be about $O(h^2)$ and the convergence rate of $e^{\mathrm{h}}_{H_1}$ about $O(h^1)$. Due to the used rational basis functions better rates of convergence cannot be expected, but those of linears, which are made possible with the iso-parametric approach. Moreover, these rates of convergence depend on a sufficiently accurate quadrature of the rational basis functions especially of the boundary cells. More quadrature points are needed in the boundary cells to achieve an adequate rate of convergence for fine meshes. The presented results were computed by using a Riemann sum with up to 6×6 sub-cells at each boundary cell.

A well known shortcoming of many immersed finite element methods is the occurrence of numerical instabilities due to the cut cells at the boundaries. To study the susceptibility of the proposed approach towards such instabilities, we investigated the condition number of the stiffness matrix for a fixed grid and moving disk. For this study a fixed grid with 30×30 cells was chosen and the ring was moved on a 30°-inclined line with respect to the horizontal axis. The

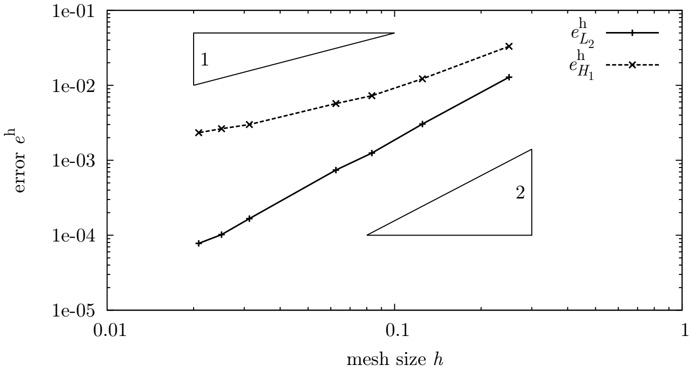


Figure 20: Convergence of linear-elastic ring

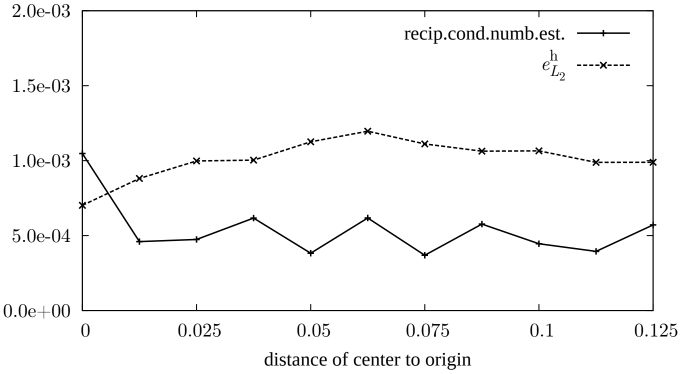


Figure 21: Condition number and $e^h_{L_2}$ error of moving disk

initial position is the origin and 10 increments with a distance of 0.0125 are applied. The increments are significantly smaller than the Cartesian mesh size of $h = 0.08\bar{3}$. During the different positions of the ring in the Cartesian mesh, not only the mesh location changed but the number of physical and boundary cells (definition see Figure 6) as well. The estimated *reciprocal* condition number of the stiffness matrix is plotted in Figure 21 at the eleven investigated locations. The condition number was estimated with SuperLU [25] using a 1-norm approach following Higham [26]. Remarkably, the condition number was only slightly dependent on the position of the ring, hence on the number and geometry of cut cells. In Figure 21 the error $e^{\mathrm{h}}_{L_2}$ at various locations is plotted as well. Its magnitude did not vary much and exhibited only a weak connection to the different body locations.

5.3. Three-quarter disk

The three-quarter disk is a two-dimensional geometry characterised by a re-entrant corner. The system is depicted in Figure 22 on the left. The radius of the disk is $R = 1$. For instance, Evans and Hossen [27] describe a boundary value problem on the three-quarter disk for which they provide an analytical solution $u(\boldsymbol{x})$ subject to mixed Dirichlet and Neumann boundary conditions. The boundary value problem is

$$\begin{aligned} \nabla \cdot \nabla u(\boldsymbol{x}) &= -(16(x^1)^2 + 1)u(\boldsymbol{x}) + 4\cos(2(x^1)^2 - (x^2)) && \text{on } \Omega\,, \\ \boldsymbol{n} \cdot \nabla u(\boldsymbol{x}) &= 0 && \text{on } \Gamma_{\mathrm{N}}\,, \\ u(\boldsymbol{x}) &= \sin(2(x^1)^2 - (x^2)) && \text{on } \Gamma_{\mathrm{D}}\,, \end{aligned} \tag{38}$$

with the Cartesian coordinates $\boldsymbol{x} = (x^1, x^2)$. The Neumann boundary is $\Gamma_{\mathrm{N}} = \{\boldsymbol{x} \,|\, x^1 = 0, -1 < x^2 < 0\}$. The remaining boundary is the Dirichlet boundary Γ_{D}, cf. Fig. 22. The analytic solution amounts to

$$u(\boldsymbol{x}) = \sin(2(x^1)^2 - (x^2)) \tag{39}$$

The three-quarter disk was immersed into a slightly larger Cartesian square with edge length $\ell = 1.25$, see Figure 22 on the left. The computed solution $u(\boldsymbol{x})$ and its gradient are compared to their analytical counterparts on the bisectrix $x^1 = x^2$ in Figure 23. The results are drawn for several mesh refinements. The solutions were computed with a transition length $\delta = 2h$ and 3×3 Gauss–Legendre quadrature points per cell. The solution $u(\boldsymbol{x})$, on the left of Fig. 23, agreed very well with the analytic solution for the investigated meshes. The computed gradient deviated mildly at the re-entrant corner $(x^1, x^2) = (0, 0)$, but it stayed close to the analytic gradient in the remaining range.

Finally, we want to take a look at the condition number of the system matrix for different orientations of the three-quarter disk. Different angular orientations lead to completely dissimilar boundary cells and thus i-splines. To study the condition number, the disk was rotated in steps of 5° around the x^3-axis starting with the x^1-axis being initially horizontal as shown in Fig. 22. A 41×41 Cartesian mesh, 3×3 Gauss points and a transition length $\delta = 2h$ were used.

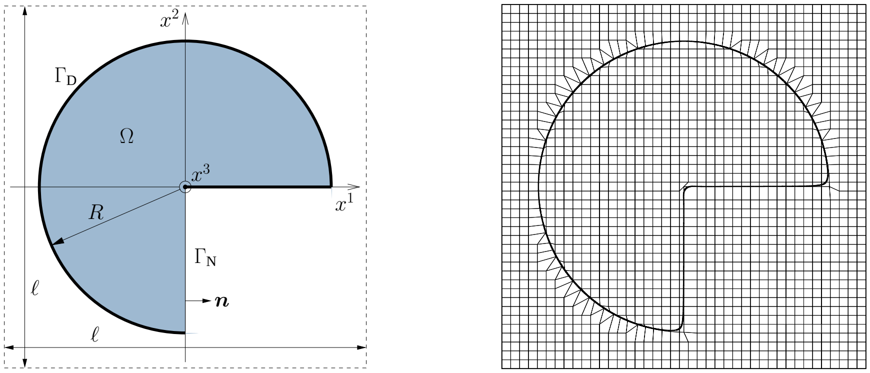


Figure 22: Three-quarter disk: System, level set and iso-parametric mesh

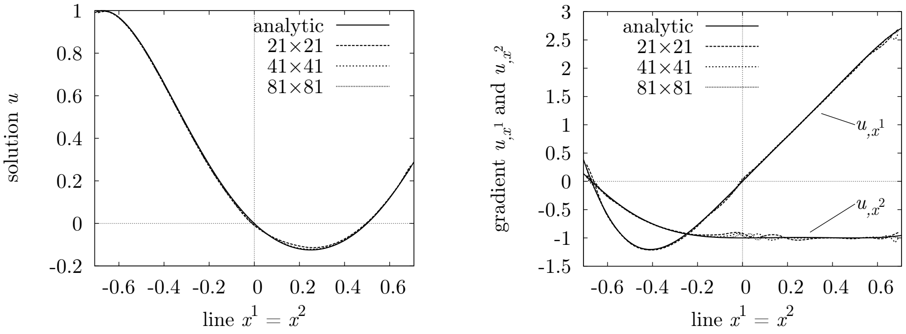


Figure 23: Solution and its gradient over line $x^1 = x^2$

The results for the angles ranging from 0° to 50° are depicted in Figure 24. The reciprocal condition number of the matrix, cf. SuperLU [25], remained essentially unaffected by the superimposed rotation. In Figure 24 the $e^{\mathrm{h}}_{L_2}$ error is given as well over the various angles. The error depended also only moderately on the investigated angles. The evolution of the condition number as well as the $e^{\mathrm{h}}_{L_2}$ error indicate the robustness of the i-spline basis towards the location, number and size of the cut cells at the immersed boundary.

5.4. Spherical cavity

This example verifies by employing a non-trivial three-dimensional geometry the ability of the proposed method to recover linear displacement states. Figure 25 shows the three-dimensional system which consists of a cube with edge length $a = 1$ of which a sphere of radius $R_0 = 0.8$, centred at one cube vertex, is subtracted. The displacements $\boldsymbol{u}(\boldsymbol{x})$ (with $\boldsymbol{x} = (x^1, x^2, x^3)$) on the boundary of the body are prescribed with the solution of an isotropic, linear

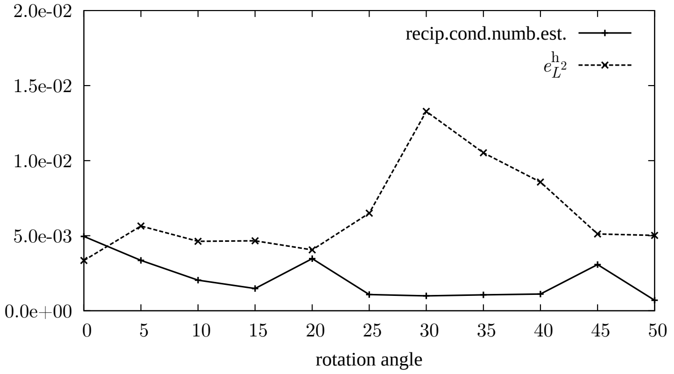


Figure 24: Condition number and $e^{\mathrm{h}}_{L_2}$ error at several rotation angles

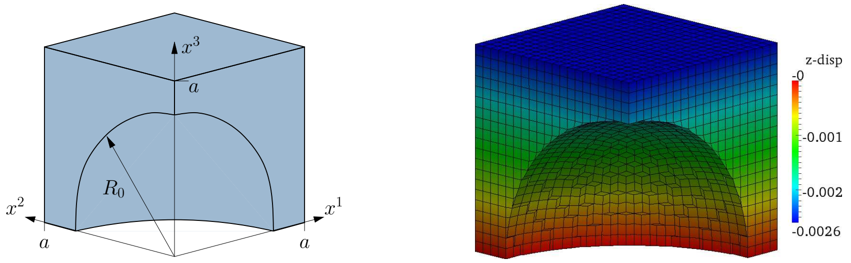


Figure 25: Spherical cavity: System and x^3-displacement of $30 \times 30 \times 30$ Cartesian mesh

elastic continuum subject to uni-axial stress in the x^3-direction, i.e.

$$\boldsymbol{u} = \frac{\sigma_0}{4\mu^2 + 6\lambda\mu} \begin{bmatrix} -\lambda\, x^1 \\ -\lambda\, x^2 \\ 2(\mu+\lambda)\, x^3 \end{bmatrix} \quad \text{and } \mu = \frac{E}{2(1+\nu)}\,,\, \lambda = \frac{\nu E}{(1+\nu)(1-2\nu)} \tag{40}$$

which can be found in many textbooks on linear elasticity, e.g. Timoshenko and Goodier [24]. The given uni-axial stress was set to $\sigma_0 = 10$; the solution is a linear displacement field in every space direction due to a Poisson ratio $\nu = 0.3$ and a Young modulus $E = 10^4$.

The linear displacement field was recovered on the given geometry, Figure 25, by the presented i-spline approach. To compute the solution, the body was embedded in a Cartesian mesh $[-0.25, 1.25]^3$. In Figure 26 the displacement components along the line $x^1 = x^2 = x^3$ are compared to the analytic solution. The x^1-displacement is denoted with u^1 and the x^3-displacement with u^3; both agreed perfectly with the linear analytic solution. The meshes were integrated with 3×3 Gauss points and the transition length was $\delta = 0.2$. The linear evolution of the displacement components can be clearly observed. Since the i-spline basis recovered exactly the linear displacements on this non-trivial geometry, it appears adequate to claim the method satisfies the constant stress

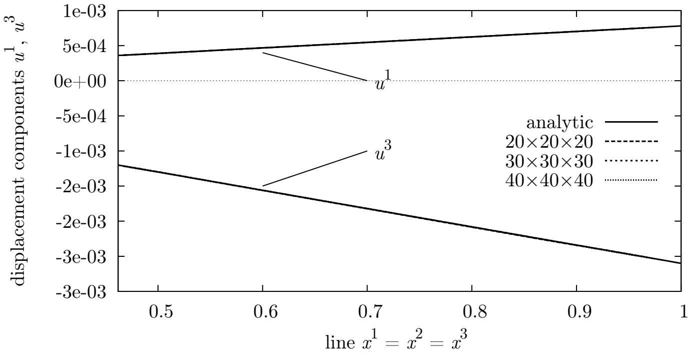


Figure 26: Spherical cavity: Solution on line $x^1 = x^2 = x^3$

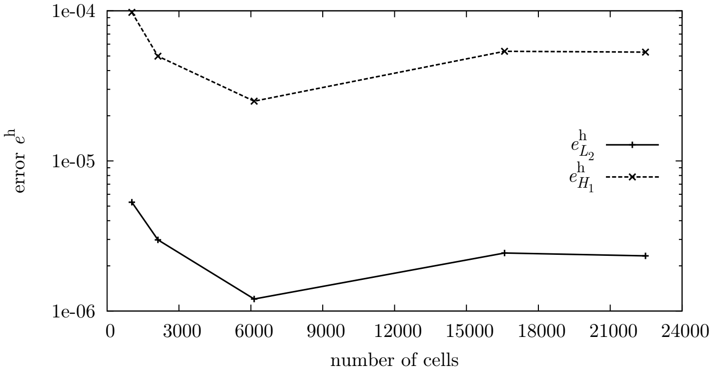


Figure 27: Spherical cavity: Convergence diagram

patch test. A convergence diagram is presented in Figure 27. The abscissa shows the number of cells containing the immersed body rather than the total number of Cartesian cells, i.e. the physical and boundary cells following definition Eq. (12). Since the analytic solution was recovered by the method at hand, the errors were low on the domain and did not differ much for the various mesh sizes, as expected.

5.5. Assorted three-dimensional examples with complex geometries

Finally, the presented finite element approach was applied to a few complex three-dimensional immersed geometries. They are focused on the ability to robustly handle qualitatively complex geometries; quantitative comparisons are not considered.

Figure 28 shows a three-dimensional horse embedded in a Cartesian domain with the dimensions $[-0.05, 0.05] \times [-0.1, 0.1] \times [-0.08, 0.08]$. The Cartesian domain was divided in $30 \times 60 \times 40$ cells. The horse geometry, i.e. its surface,

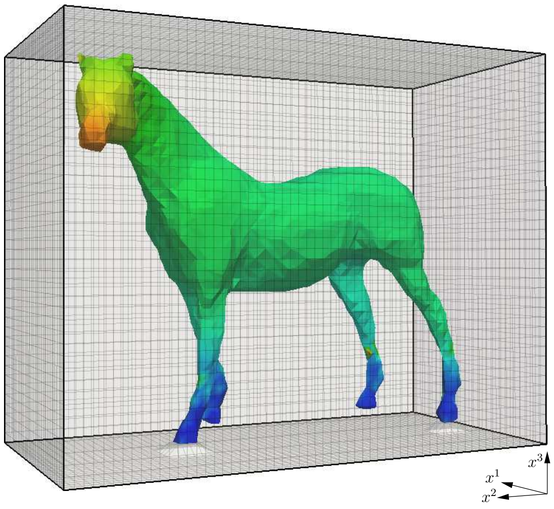


Figure 28: Horse

was constructed of 666 triangles. In Figure 28 the determined level set surface is presented on which the computed x^2-displacements are mapped; the x^2-axis follows the horse back. The displacement field stems from assuming an isotropic linear elastic material with $E = 10^4$ and $\nu = 0.3$ and pulling the head of the horse with a prescribed displacement of 0.02 along the x^2-axis. The hooves are fixed.

In the next example, the deformation of a hollow duck was computed, Figure 29. The outer duck surface was described by a faceted surface made of 4096 triangles. The core was straightforwardly removed by computing a new signed distance function based on the outer signed distance, which was found with the help of the CPT algorithm [19]. The inner signed distance was obtained by shrinking the outer surface by 0.16 and negating the result. Then, the hollow duck level set was achieved by combining the two signed distances as suggested in Rvachev et al. [21, 22] or Höllig [13]. The duck was modelled with Hooke's material with the (arbitrary) material constants $E = 100$ and $\nu = 0.3$. The duck has approximately the dimensions $3.9 \times 4.2 \times 4.6$. The duck was enclosed in a Cartesian domain of $[-2.0, 2.0] \times [0.0, 4.5] \times [-3.0, 1.75]$ with $45 \times 45 \times 45 = 91125$ cells of which 13775 were identified as physical and 323 as boundary cells. The quadrature was based on $2 \times 2 \times 2$ Gauss-Legendre points. Figure 29 highlights the x^2-displacements found for a prescribed displacement of the duck beak in x^3-direction of -0.5 and a fixed support of the duck bottom side. The x^2-displacements are shown on the exterior surface on the left in Fig. 29 and on the interior surface on the right in Fig. 29.

Another interesting geometry is the lower part of the human heart shown in Figure 30. The part consists of the endocardium, epicardium and the connection at the base resulting topology-wise in a first order doughnut. The geometry of

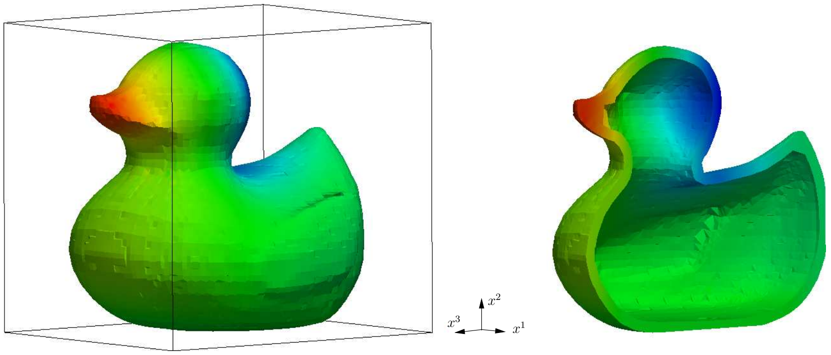


Figure 29: Hollow duck

van Oosterom and van Dam was enhanced with subdivision resulting in an increase of 1026 to 16334 triangles. The physical dimensions of the heart piece are ca. 96.3 × 109.7 × 99.3 mm. The computational Cartesian domain was set to $[-10, 90] \times [-43, 72] \times [-68, 37]$ using $40 \times 45 \times 45$ cells. The i-spline basis utilises 7601 physical and 10116 boundary cells. The connection at the base was kept fixed and the wall at the end of the ‘bridge’ was pulled sideways in x^1-direction. The snapshots in Figure 30 are colorised with the computed x^2-displacements ($E = 100$, $\nu = 0.3$). The left snapshot of Fig. 30 is based on the level set geometry, whereas the right on the iso-parametric geometry. The boxes in Fig. 30 indicate the Cartesian domain.

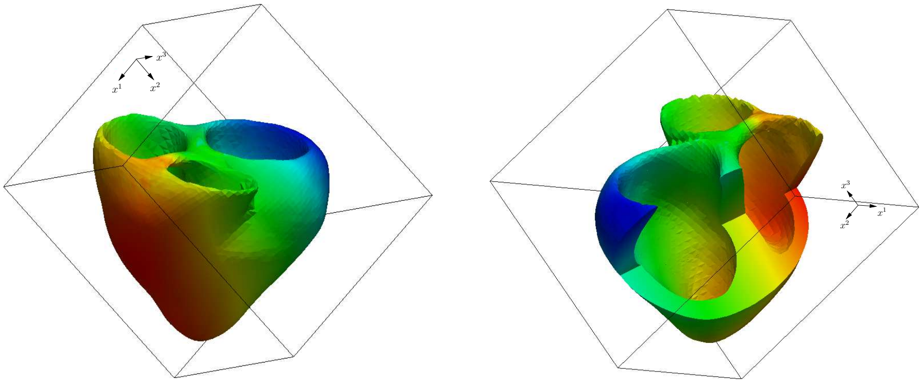


Figure 30: Lower part of human heart

6. Conclusions

A new immersed Cartesian b-spline finite element approach is proposed, which exactly satisfies the Dirichlet and Neumann boundary conditions. This is achieved by being able to separate the basis functions on the Dirichlet boundary from the remaining basis functions, similarly to the well-known Lagrangian finite element basis functions. The separation is based on a combination of introducing a weight function, eliminating basis functions with small support and normalising the weighted b-splines. The method inherits almost all of the b-spline basis function properties and, hence, has better accuracy and is more efficient than the conventional Lagrange type basis functions (see, e.g., Evans et al. [28]).

Furthermore, it can be shown that the method satisfies the constant stress patch test. Although the resulting basis functions are not necessarily polynomial, an iso-parametric geometry description is put forward enabling the exact interpolation of linear functions. Several techniques to obtain an iso-parametric set of node coordinates are summarised, and an intriguing simple approach, i.e. looking for the closest point on the immersed boundary, was found robust in many computed examples.

The method was applied to elliptic second-order boundary value problems with mixed boundary conditions. Various two- and three-dimensional examples were calculated for which the computed solutions compared very well to the analytic solutions. As demonstrated the method was capable to compute complex three-dimensional geometries.

The use of a three-dimensional Cartesian block-structured background mesh requires much computational resources. Therefore, as a future work, an adaptive refinement to account for details of the boundary is desirable. On the other hand, the latter can be achieved using mesh refinement strategies natural to b-splines.

Acknowledgement

The research reported here was supported in part by the Brazilian National Council for Scientific and Technological Development (CNPq) and the EU through the Marie Curie Actions Program (IAPP) as part of the CASOPT project. The horse mesh in Fig. 28 is courtesy of R.W. Sumner and J. Popovich (MIT). The heart mesh in Fig. 30 is courtesy of van Oosterom and van Dam (http://www.physionet.org/challenge/2007/data/case0003-tri/).

Appendix A. Constant and linear approximation due to iso-parametric approach

The proposed method can approximate linear functions, e.g. displacements, exactly due to its inherent partition of unity property, c.f. Section 4.2.3, and the iso-parametric geometry approximation.

A linear function $f(\boldsymbol{x}) = a\boldsymbol{x} + b$ is approximated with its discrete values $f_{\boldsymbol{i}} = f(\boldsymbol{x}_{\boldsymbol{i}})$ using the i-spline basis functions $N_{\boldsymbol{i}}(\boldsymbol{x})$ leading to

$$f^{\mathrm{h}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi}) f_{\boldsymbol{i}} = \sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi}) \big(a\boldsymbol{x}_{\boldsymbol{i}} + b \big) \tag{A.1}$$

Although this approximation is not in general linear over $\boldsymbol{\xi}$, it is still linear over the iso-parametric approximation of the coordinate $\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi}) = \sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\boldsymbol{x}_{\boldsymbol{i}}$. The preserved linearity over $\boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi})$ can be seen by expanding Equation (A.1)

$$f^{\mathrm{h}}(\boldsymbol{\xi}) = a \underbrace{\sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi})\boldsymbol{x}_{\boldsymbol{i}}}_{= \boldsymbol{x}^{\mathrm{h}}(\boldsymbol{\xi})} + b \underbrace{\sum_{\boldsymbol{i}} N_{\boldsymbol{i}}(\boldsymbol{\xi})}_{= 1} , \tag{A.2}$$

which yields the desired result $f^{\mathrm{h}}(\boldsymbol{x}^{\mathrm{h}}) = a\boldsymbol{x}^{\mathrm{h}} + b$ after dropping the parametric coordinates.

References

[1] T. Hughes, J. 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.

[2] 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.

[3] J. Peters, U. Reif, Subdivision surfaces, Springer Verlag, 2008.

[4] J. Warren, H. Weimer, Subdivision methods for geometric design: a constructive approach, Morgan Kaufmann, 2001.

[5] F. Cirak, M. Scott, E. Antonsson, M. Ortiz, P. Schröder, Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision, Computer-Aided Design 34 (2002) 137–148.

[6] J. Cottrell, T. Hughes, Y. Bazilevs, Isogeometric analysis: toward integration of CAD and FEA, John Wiley & Sons Ltd., 2009.

[7] H. Prautzsch, W. Boehm, M. Paluszny, Bezier and b-spline techniques, Springer Verlag, 2002.

[8] C. De Boor, K. Höllig, S. Riemenschneider, Box splines, Springer Verlag, 1993.

[9] R. Glowinski, T.-W. Pan, J. Periaux, A fictitious domain method for Dirichlet problem and applications, Computer Methods in Applied Mechanics and Engineering 111 (1994) 283–303.

[10] C. Peskin, The immersed boundary method, Acta Numerica 11 (2002) 479–517.

[11] T. Belytschko, C. Parimi, N. Moes, N. Sukumar, S. Usui, Structured extended finite element methods for solids defined by implicit surfaces, International Journal for Numerical Methods in Engineering 56 (2003) 609–635.

[12] K. Höllig, U. Reif, J. Wipper, Weighted extended b-spline approximation of dirichlet problems, SIAM Journal on Numerical Analysis 39 (2002) 442–462.

[13] K. Höllig, Finite element methods with b-splines, SIAM, Philadelphia, 2003.

[14] L. W. Kantorovich, W. I. Krylov, Approximate methods of higher analysis, Interscience Publishers, New York, translated from the 4th Rusian edition, 1964.

[15] L. Piegl, W. Tiller, The NURBS book, Springer, 1997.

[16] D. Rogers, An introduction to NURBS, Academic Press, 2001.

[17] C. de Boor, A practical guide to splines, Springer-Verlag, New York, revised edition, 2001.

[18] S. Mauch, Efficient algorithms for solving static Hamilton-Jacobi equations, Ph.D. thesis, California Institute of Technology, 2003.

[19] F. Cirak, R. Deiterding, S. Mauch, Large-scale fluid-structure interaction simulation of viscoplastic and fracturing thin-shells subjected to shocks and detonations, Computers & Structures 85 (2007) 1049–1065.

[20] W. Lorensen, H. E. Cline, Marching cubes: a high resolution 3d surface construction algorithm, Computer Graphics 21 (1987) 163–169.

[21] V. L. Rvachev, T. I. Sheiko, R-functions in boundary value problems in mechanics, Applied Mechanics Reviews 48 (1995) 151–188.

[22] V. L. Rvachev, T. I. Sheiko, V. Shapiro, I. Tsukanov, On completeness of RFM solution structures, Computational Mechanics 25 (2000) 305–317.

[23] J. Nocedal, S. J. Wright, Numerical optimization, Springer, 2nd edition, 2000.

[24] S. Timoshenko, J. N. Goodier, Theory of elasticity, McGraw-Hill Higher Education, 3rd edition, 1970.

[25] J. W. Demmel, J. R. Gilbert, X. S. Li, SuperLU users' guide, Technical Report LBNL-44289, Lawrence Berkeley National Lab, Berkeley, CA, U.S.A., 2009.

[26] N. J. Higham, Accuracy and stability of numerical algorithms, SIAM, Philadelphia, PA, U.S.A., 1996.

[27] D. J. Evans, K. A. A. Hossen, The numerical solution of problems involving singularities by the finite element method, International Journal of Computer Mathematics 19 (1986) 339–379.

[28] J. Evans, Y. Bazilevs, I. Babuska, T. 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.