Comput. Methods Appl. Mech. Engrg. 213–216 (2012) 206–222

Contents lists available at SciVerse ScienceDirect

Comput. Methods Appl. Mech. Engrg.

journal homepage: www.elsevier.com/locate/cma

Local refinement of analysis-suitable T-splines

M.A. Scott [a,*], X. Li [b], T.W. Sederberg [c], T.J.R. Hughes [a]

[a] Institute for Computational Engineering and Sciences, The University of Texas at Austin, Austin, TX 78712, USA
[b] University of Science and Technology of China, Anhui Province, Hefei 230026, PR China
[c] Department of Computer Science, Brigham Young University, Provo, UT 84602, USA

ARTICLE INFO

Article history:
Received 1 April 2011
Received in revised form 18 November 2011
Accepted 22 November 2011
Available online 6 December 2011

Keywords:
Isogeometric analysis
T-splines
Local refinement
Bézier extraction

ABSTRACT

We develop a local refinement algorithm for analysis-suitable T-splines which does not produce excessive propagation of control points. We then demonstrate its use as an adaptive framework for isogeometric analysis. Analysis-suitable T-splines are a class of T-splines which are linearly independent and form a partition of unity. These properties, coupled with local refinement, make this class of T-splines appealing as a basis for isogeometric analysis.

1. Introduction

Isogeometric analysis has emerged as an important alternative to traditional engineering design and analysis methodologies. Isogeometric analysis was introduced in [1] and later described in detail in [2]. In isogeometric analysis, the smooth geometric basis is used as the basis for analysis. Most of the early developments in isogeometric analysis focused on establishing the behavior of the smooth NURBS basis in analysis. It was demonstrated that smoothness offers important computational advantages over standard finite elements [3,4]. Areas of application of NURBS-based isogeometric analysis include turbulence [5–8], fluid–structure interaction [9–12], incompressibility [13–15], structural analysis [16,3], plates and shells [17–21], phase-field analysis [22,23], large deformation with mesh distortion [24], shape optimization [25–28], and electromagnetics [29]. This success has in turn stimulated efforts within the Computer Aided Geometric Design (CAGD) community to develop and integrate analysis-suitable geometric technologies and isogeometric analysis [30–37].

While smoothness is an important consideration, NURBS are severely limited by their tensor product construction. In traditional NURBS-based design, modeling a complicated engineering design often requires hundreds, if not thousands, of tensor product NURBS patches which are usually discontinuous across patch boundaries. Also, almost all NURBS models use trimming curves. For these reasons, a global geometric discretization, based on NURBS, is usually not suitable as a basis for analysis.

T-splines were introduced as a superior alternative to NURBS [38]. T-splines can model complicated designs as a single, watertight geometry. Additionally, NURBS are T-splines so existing technology based on NURBS extends to T-splines. Any trimmed NURBS model can be represented by a watertight trimless T-spline [39] and multiple NURBS patches can be merged into a single watertight T-spline [38,40]. Unlike NURBS, T-splines can be locally refined [41] without introducing a complex hierarchy of meshes [42]. In other words, all local refinement is done on one control mesh on a single hierarchical "level" and all control points have similar influence on the shape of the surface. These properties make T-splines an ideal technology for isogeometric discretizations and integrated design-through-analysis applications.

Initial investigations using T-splines as a basis for isogeometric analysis demonstrated that the T-spline basis possesses similar convergence properties to NURBS with far fewer degrees-of-freedom [43–45]. Additionally, T-splines possess a natural finite element structure which can be integrated seamlessly into existing finite element frameworks via Bézier extraction [46,47]. T-splines have since been applied to problems in fracture and damage [48,49], and shells [19].

Analysis-suitable T-splines were introduced in [50]. Analysis-suitable T-splines are a mildly restricted subset of T-splines. Analysis-suitable T-splines are linearly independent [50] and, if a minor boundary condition constraint is honored, form a partition of unity [51]. In this paper, we develop a highly localized refinement algorithm for analysis-suitable T-splines which meets the demands of *both* design and analysis.

This paper is organized as follows. Basic T-spline concepts are reviewed in Section 2. Analysis-suitable T-splines are then de-

* Corresponding author.
E-mail address: mscott@ices.utexas.edu (M.A. Scott).

0045-7825/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.cma.2011.11.022

scribed in Section 3. Section 4 presents a local refinement algorithm for analysis-suitable T-splines. The behavior and effectiveness of this algorithm is then demonstrated in Section 5. For simplicity, this paper focuses on bicubic T-spline surfaces, although the concepts generalize to arbitrary odd degree. T-splines of arbitrary degree are discussed in [43,52].

2. T-spline fundamentals

We present a brief overview of fundamental T-spline concepts focusing on those ideas required to understand local refinement. A more detailed description of T-splines from an isogeometric analysis perspective is presented in [47]. We base our developments on the physical domain $\Omega \subset \mathbb{R}^2$ shown in Fig. 1. Throughout this paper we use d_s to indicate the number of spatial dimensions. In all cases, the polynomial degree p is 3.

2.1. The T-mesh

The fundamental object of interest underlying T-spline technology is the T-mesh, denoted by T. For surfaces, the T-mesh is a mesh of polygonal elements. Each polygonal element is either a quadrilateral or an element with quadrilateral shape where one or more edges is split by T-junctions. A T-junction is analogous to a "hanging node" in finite elements. A control point, $\mathbf{P}_A \in \mathbb{R}^{d_s}$, and control weight, $w_A \in \mathbb{R}$, where the index A denotes a global control point number, is assigned to every vertex[1] in the T-mesh. A mesh for the domain Ω in Fig. 1 is shown in Fig. 2a. The black and red circles are T-mesh vertices, or, equivalently, control points. The T-junctions in Fig. 2a are the red circles $\mathbf{P}_{16}$, $\mathbf{P}_{17}$, $\mathbf{P}_{28}$, and $\mathbf{P}_{29}$.

To define a basis, a valid knot interval configuration must be assigned to the T-mesh. A knot interval [53] is a non-negative real number assigned to an edge. A valid knot interval configuration requires that the knot intervals on opposite sides of every element sum to the same value. A valid knot interval configuration for the T-mesh in Fig. 2a is shown in Fig. 2b.

2.2. The T-spline basis

Once a valid knot interval configuration has been assigned to a T-mesh, a T-spline basis can be constructed. For every vertex in the T-mesh, a T-spline basis function is constructed. To illustrate, we construct the T-spline basis function associated with $\mathbf{P}_{16}$. We note that a class of T-splines where the T-spline blending functions do, in fact, constitute a basis is described in Section 3.

2.2.1. Local knot interval vectors

A T-spline basis function is constructed from knot interval sequences inferred from the T-mesh in the neighborhood of the associated vertex. These knot interval sequences are called local knot interval vectors. A local knot interval vector is a sequence of knot intervals, $\Delta\Xi = \{\Delta\xi_1, \Delta\xi_2, \Delta\xi_3, \Delta\xi_4\}$.

For every vertex, A, in the T-mesh, we construct a set of local knot interval vectors, $\mathbf{\Delta\Xi}_A = \left\{\Delta\Xi_A^i\right\}_{i=1}^2$, where $\Delta\Xi_A^i = \left\{\Delta\xi_{A,1}^i, \Delta\xi_{A,2}^i, \Delta\xi_{A,3}^i, \Delta\xi_{A,4}^i\right\}$, by marching through the T-mesh in each topological direction, starting at vertex A, until 2 vertices or perpendicular edges are intersected. At each intersection, the knot interval distance traversed since the last intersection is placed in the local knot interval vector. If a T-mesh boundary is crossed before 2 knot intervals are intersected it is common to set the remaining knot intervals to zero. This creates an open knot vector structure along the boundary of the T-mesh. We note that this same procedure is applied to construct the local knot interval vectors for T-junctions.

Fig. 1. The domain $\Omega \subset \mathbb{R}^2$ for a bivariate, cubic ($p = 3$) T-spline.

A set of local knot vectors, $\mathbf{\Xi}_A = \left\{\Xi_A^i\right\}_{i=1}^2$, where $\Xi_A^i = \left\{\xi_{A,1}^i, \xi_{A,2}^i, \ldots, \xi_{A,5}^i\right\}$, can be derived from $\mathbf{\Delta\Xi}_A$ by selecting an origin $O \in \mathbb{R}$ and setting $\xi_{A,j}^i = O + \sum_{k=2}^{j} \Delta\xi_{A,k-1}^i$. Since a knot in a local knot vector corresponds to T-mesh topology (vertices or edges) encountered during basis function inference the T-mesh topology and knot interval configuration determine the *knot structure* of the underlying T-spline space. In the context of T-splines, the term knot is often also used to refer to the underlying T-mesh vertex or edge. In Fig. 3, the knot intervals used to construct the local knot interval vectors for T-mesh vertex $\mathbf{P}_{16}$ are shown. The knot interval vectors for $\mathbf{P}_{16}$ are given by

$$\mathbf{\Delta\Xi}_{16} = \begin{bmatrix} 1,1,2,0 \\ 0,2,2,2 \end{bmatrix}.$$

2.2.2. The local basis function domain

Using $\mathbf{\Delta\Xi}_A$ we define a local basis function domain, $\widehat{\Omega}_A \subset \mathbb{R}^2$, as

$$\widehat{\Omega}_A = \bigotimes_{i=1}^{2} \widehat{\Omega}_A^i, \tag{1}$$

where $\widehat{\Omega}_A^i = \left[0, \sum_{j=1}^{4} \Delta\xi_{A,j}^i\right] \subset \mathbb{R}$. A coordinate system, $\boldsymbol{\xi}_A = \left(\xi_A^1, \xi_A^2\right) = (\xi_A, \eta_A)$, called the *basis coordinate system*, is assigned to each local basis function domain.

We note that in addition to the basis coordinate systems in a T-mesh, it is often desirable to establish larger *knot coordinate systems* for a subset of the knot structure of a T-mesh. Using a knot coordinate system, multiple basis functions can be compared in a common coordinate system. Knot coordinate systems are used when computing the elements of the refinement operator, $\mathbf{M}$, as described in Sections 2.4.2 and 4.3.

2.2.3. T-spline basis functions

Over each local basis function domain $\widehat{\Omega}_A$ we define a single T-spline basis function, $N_A : \widehat{\Omega}_A \to \mathbb{R}^+ \cup \{0\}$. This is done by forming the tensor product of the univariate B-spline basis functions $\left\{N_A^{i,3}\left(\xi_A^i | \Xi_A^i\right)\right\}_{i=1}^2$ as

$$N_A(\boldsymbol{\xi}_A | \mathbf{\Xi}_A) \equiv \prod_{i=1}^{2} N_A^{i,3}\left(\xi_A^i | \Xi_A^i\right). \tag{2}$$

The univariate B-spline basis function, $N_A^{i,p} : \widehat{\Omega}_A^i \to \mathbb{R}^+ \cup \{0\}$, is defined using a recurrence relation, starting with the piecewise constant ($p = 0$) basis function

[1] In this paper, we use the term "vertex" and "control point" interchangeably. It should be noted, however, that a vertex usually only represents a topological mesh entity while a control point is the location of a vertex in physical space.

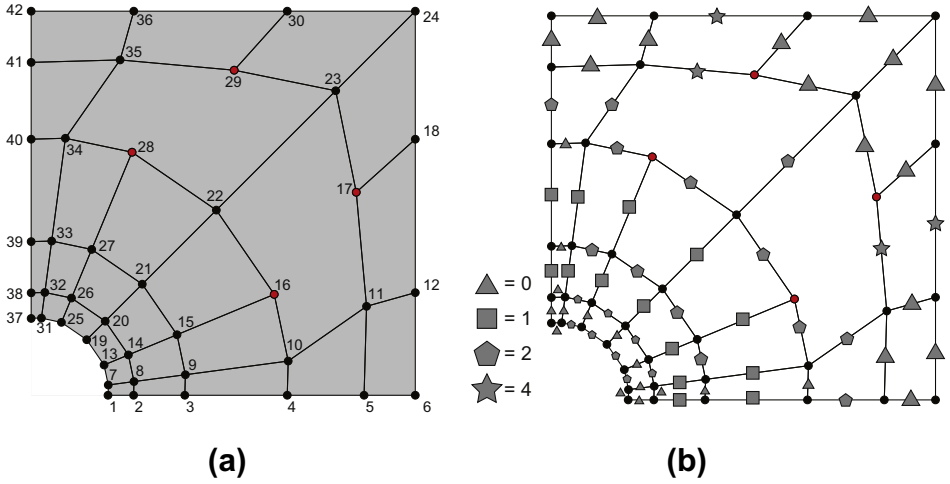


Fig. 2. The T-mesh and knot interval configuration defining the bicubic T-spline geometry in Fig. 1. (a) The T-mesh defining the bicubic T-spline geometry in Fig. 1. The red circles (16,17,28,29) are T-junctions. The indexing identifies the T-mesh control points. (b) A valid knot interval configuration for the bicubic T-mesh in (a). The triangles correspond to a knot interval of 0, the squares correspond to a knot interval of 1, the pentagons correspond to a knot interval of 2, and the stars correspond to a knot interval of 4. Notice that the knot intervals along opposing sides of each T-mesh element sum to the same value. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

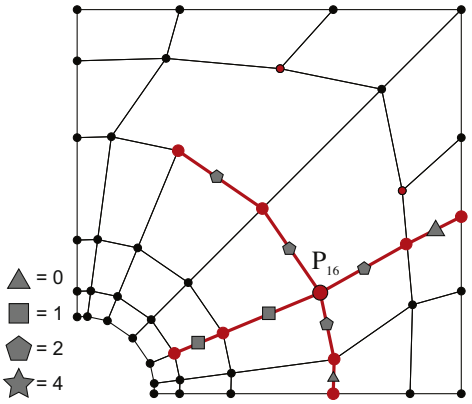


Fig. 3. Constructing the local knot interval vectors corresponding to T-mesh vertex $\mathbf{P}_{16}$.

$$N_A^{i,0}\left(\xi_A^i|\xi_{A,1}^i,\xi_{A,2}^i\right)=\begin{cases}1 & \text{if } \xi_{A,1}^i\leqslant\xi_A^i<\xi_{A,2}^i\\ 0 & \text{otherwise,}\end{cases}\tag{3}$$

where $\xi_{A,k}^i$ is the kth knot value in the local knot vector Ξ_A^i. For $p>0$, the basis function is defined using the Cox–de Boor recursion formula:

$$N_A^{i,p}\left(\xi_A^i|\xi_{A,1}^i,\xi_{A,2}^i,\ldots,\xi_{A,p+2}^i\right)=\frac{\xi_A^i-\xi_{A,1}^i}{\xi_{A,p+1}^i-\xi_{A,1}^i}N_A^{i,p-1}\left(\xi_A^i|\xi_{A,1}^i,\ldots,\xi_{A,p+1}^i\right)+\frac{\xi_{A,p+2}^i-\xi_A^i}{\xi_{A,p+2}^i-\xi_{A,2}^i}N_A^{i,p-1}\left(\xi_A^i|\xi_{A,2}^i,\ldots,\xi_{A,p+2}^i\right).\tag{4}$$

2.3. Bézier element construction

A Bézier element is a region of the T-spline surface in physical space bounded by knot lines. Each knot line in physical space is the image of a line of reduced continuity in at least one T-spline basis function. We call the collection of Bézier elements the Bézier mesh. The existence of T-junctions and zero knot intervals usually results in there not being a one-to-one correspondence between T-mesh elements and Bézier elements.

An elemental T-mesh, T_{elem}, is formed by augmenting T with all images of basis function knot lines that do not already correspond to an edge in T, then eliminating all elements for which the knot interval sum on any side is zero. The elements of T_{elem} are in one-to-one correspondence with the Bézier elements. Fig. 4a shows T_{elem} for the T-mesh in Fig. 2a. Dashed lines represent the edges which have been added.

To construct appropriate finite element paraphernalia for T-splines we use T_{elem} and Bézier extraction [47,46]. Bézier extraction builds a linear operator for each Bézier element. The linear transformation is defined by a matrix referred to as the extraction operator. The extraction operator maps a Bernstein polynomial basis defined on Bézier elements to the global T-spline basis. The transpose of the extraction operator maps the control points of the global T-spline to the Bézier control points. The idea is illustrated in Fig. 5 for a B-spline curve. This element form can then be integrated into existing finite element frameworks in a straightforward manner. See [47] for additional details.

The extracted Bézier mesh in physical space for the T-mesh in Fig. 2a is shown in Fig. 4b. Notice the one-to-one correspondence between elements in T_{elem} in Fig. 4a and the Bézier elements in Fig. 4b. On the other hand, a single T-mesh element may correspond to multiple Bézier elements, and other T-mesh elements may not correspond to any Bézier element. To demonstrate, the Bézier elements corresponding to the T-mesh in Fig. 2 are shown in Fig. 4b. Notice that Bézier elements 3 and 6 correspond to the same T-mesh element, as do Bézier elements 9 and 12. Each Bézier element in Fig. 4b is the image of a unique element in T_{elem} in Fig. 4a under the T-spline geometric map.

2.4. T-spline local refinement

The set of all T-splines with the same T-mesh topology, T, and knot interval configuration is called a *T-spline space* [41]. We denote a T-spline space by $\mathcal{T}$, where the number of T-spline control points in T is n. While the notation $\mathcal{T}^1\subseteq\mathcal{T}^2$ will be used in the conventional set-theoretic sense, the notation $\mathsf{T}^1\subseteq\mathsf{T}^2$ will indicate that T^2 can be created by adding vertices and edges to T^1, and appropriately modifying the knot intervals on any edges which are split. In the context of finite element analysis, vertices and edges are usually added by subdividing T-mesh elements.

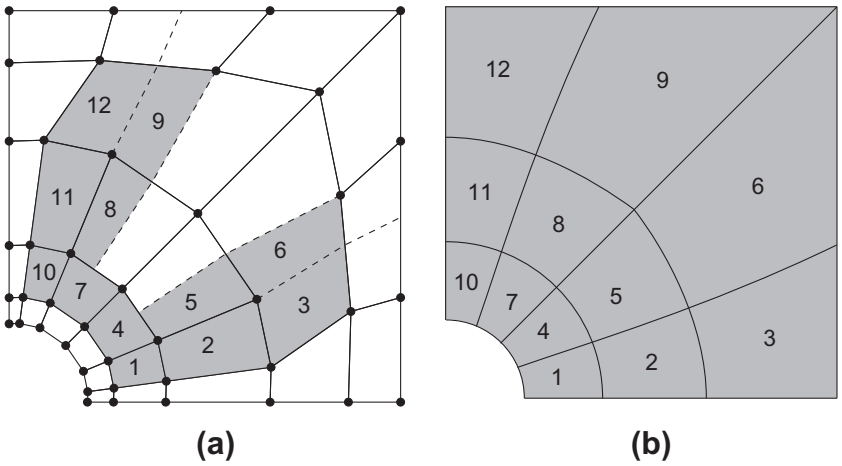


Fig. 4. The elemental T-mesh, T_{elem}, and extracted Bézier mesh in physical space corresponding to the T-mesh in Fig. 2a. (a) The elemental T-mesh, T_{elem}. The dashed lines are edges which have been added to T. Only the non-zero parametric area elements (the shaded elements) are included in T_{elem}. The indexing identifies the elements in T_{elem}. (b) The extracted Bézier mesh in physical space. Each element in (a) corresponds to a Bézier element in physical space. The indexing identifies the Bézier elements.

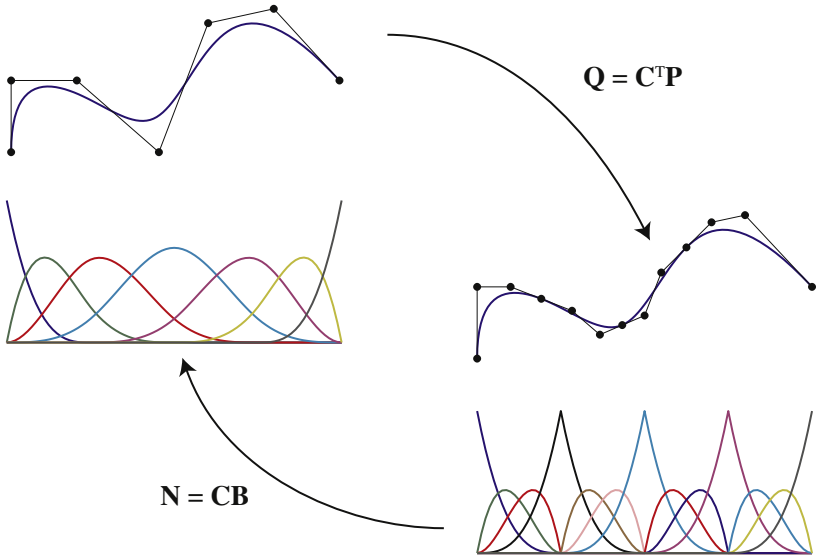


Fig. 5. Bézier extraction for a B-spline curve. B-spline basis functions and control points are denoted by **N** and **P**, respectively. Bernstein polynomials and control points are denoted by **B** and **Q**, respectively. The curve $T(\xi) = \mathbf{P}^T\mathbf{N}(\xi) = \mathbf{Q}^T\mathbf{B}(\xi)$. The extraction operator can be localized to the individual elements. Note that the Bernstein basis is the same for each element. Formation of element arrays can thus be standardized; see [47] for further details.

A central contribution of this paper (see Section 4.2) is to identify conditions under which

$$\mathsf{T}^1 \subseteq \mathsf{T}^2 \rightarrow \mathcal{T}^1 \subseteq \mathcal{T}^2. \tag{5}$$

If $\mathcal{T}^1 \subseteq \mathcal{T}^2, \mathcal{T}^1$ and $\mathcal{T}^2$ are said to be *nested* and $\mathcal{T}^2$ is a *local refinement* of $\mathcal{T}^1$. In Fig. 6, a T-mesh, T^1, (solid circles and lines) and corresponding T-spline space, $\mathcal{T}^1$, are locally refined through the addition of control points and edges (hollow circles and dashed edges). In this case, $\mathsf{T}^1 \subseteq \mathsf{T}^2 \rightarrow \mathcal{T}^1 \subseteq \mathcal{T}^2$.

2.4.1. Basis function refinement

In basis function refinement [41], knots are added to the local knot vector of a cubic B-spline basis function, $N(\xi|\Xi)$, where $\Xi = \{\xi_1, \xi_2, \ldots, \xi_5\}$, to form a knot vector, $\overline{\Xi}$, of length m. N can then be written as a linear combination of the $m - 4$ B-spline basis functions defined over substrings of length 5 in $\overline{\Xi}$.

For the case $m = 6$, $N(\xi|\Xi)$ is split by inserting a single knot $\bar{\xi}$ into Ξ where $\xi_i \leqslant \bar{\xi} \leqslant \xi_{i+1}$. This splits the basis function into two scaled basis functions:

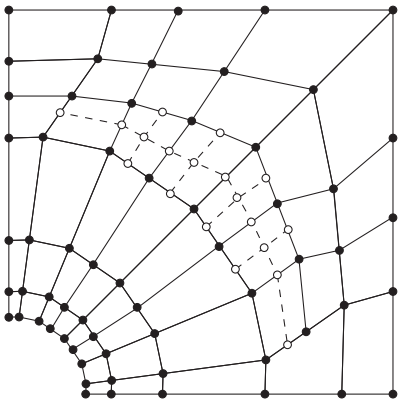


Fig. 6. A T-mesh, T^1, (solid circles and lines) and T-spline space, $\mathcal{T}^1$, is locally refined through the addition of control points and edges (hollow circles and dashed edges). In this case, $\mathsf{T}^1 \subseteq \mathsf{T}^2 \rightarrow \mathcal{T}^1 \subseteq \mathcal{T}^2$.

$$N(\xi|\xi_1,\ldots,\xi_5) = aN(\xi|\xi_1,\ldots,\xi_i,\bar{\xi},\xi_{i+1},\ldots,\xi_4) + bN(\xi|\xi_2,\ldots,\xi_i,\bar{\xi},\xi_{i+1},\ldots,\xi_5), \tag{6}$$

where

$$a = \begin{cases} \frac{\bar{\xi}-\xi_1}{\xi_4-\xi_1} & \text{for } \bar{\xi} < \xi_4 \\ 1 & \text{for } \bar{\xi} \geqslant \xi_4 \end{cases} \tag{7}$$

and

$$b = \begin{cases} \frac{\xi_5-\bar{\xi}}{\xi_5-\xi_2} & \text{for } \bar{\xi} > \xi_2 \\ 1 & \text{for } \bar{\xi} \leqslant \xi_2. \end{cases} \tag{8}$$

The refinement equations for the case $m > 6$ can be derived through repeated application of these equations.

A T-spline basis function, $N(\xi|\Xi)$, can undergo knot insertion in either parametric direction by inserting a knot into the corresponding local knot vector and then applying the refinement equations. This results in two scaled T-spline basis functions which sum to the original. Further knot insertion into these resultant scaled basis functions yields a set of scaled basis functions which also sum to the original.

2.4.2. The refinement operator **M**

If $\mathcal{T}^1 \subseteq \mathcal{T}^2$, each T-spline basis function, $N_A^1 \in \mathcal{T}^1$, can be expressed uniquely as a linear combination of the T-spline basis functions, $N_B^2 \in \mathcal{T}^2$, as

$$N_A^1 = \sum_{B=1}^{n_2} m_{A,B} N_B^2, \tag{9}$$

where n_2 is the number of control points in $\mathcal{T}^2$ and the $m_{A,B}$ are determined by knot insertion as described in Section 2.4.1. This relationship can be written in matrix–vector notation as

$$\mathbf{N}^1 = \mathbf{M}\mathbf{N}^2, \tag{10}$$

where $\mathbf{N}^1 = \left(N_1^1, N_2^1, \ldots, N_{n_1}^1\right)^T$ is the column vector of T-spline basis functions, $N_A^1 \in \mathcal{T}^1$, $\mathbf{N}^2 = \left(N_1^2, N_2^2, \ldots, N_{n_2}^2\right)^T$ is the column vector of T-spline basis functions, $N_B^2 \in \mathcal{T}^2$, and $\mathbf{M}$ is an $n_1 \times n_2$ matrix with elements $m_{A,B}$. We call $\mathbf{M}$ the refinement operator.

3. Analysis-suitable T-splines

Analysis-suitable T-splines form a practically useful subset of T-splines. Analysis-suitable T-splines maintain the important mathematical properties of the NURBS basis while providing an efficient and highly localized refinement capability. All T-splines possess the following properties:

- The basis constitutes a partition of unity [51] (see Section 3.4.)
- Each basis function is non-negative.
- An affine transformation of an analysis-suitable T-spline is obtained by applying the transformation to the control points. We refer to this as affine covariance. This implies that all "patch tests" (see [54]) are satisfied *a priori*.
- *They obey the convex hull property.*
- *They can be locally refined.*

While most (but not all [55]) T-splines are also linearly independent, analysis-suitable T-splines are *always* linearly independent for any choice of knot intervals [50]. Analysis-suitable T-spline spaces are defined over a mildly restricted set of allowable T-mesh topologies. This topological restriction can be described elegantly in terms of *T-junction extensions*.

3.1. T-junction extensions

A T-junction extension is normally composed of a face and edge extension (if one exists.) We define face and edge extensions to be closed, directed line segments which originate at a T-junction. An extended T-mesh T_{ext} is formed by adding all T-junction extensions to a T-mesh T. The extended T-mesh, T_{ext}, for the T-mesh in Fig. 7a is shown in Fig. 7b and c, respectively. The dotted black arrows are face extensions and the dashed red arrows are edge extensions. The T-junctions are denoted by large circles.

Fig. 7b shows the T-junction extensions in the *index space* [43] of the T-mesh. The index space is created by plotting the knots in the T-mesh at equally spaced intervals, regardless of their actual values. The index space point of view is useful for developing algorithms, as well as for building intuition. For example, in the index space it is easy to identify the knot lines at which the support of any given function will begin or end. Additionally, the direction of traversal and orientation of a T-junction extension can be uniquely established in the index space of the T-mesh.

Fig. 7c shows the set of T-junction extensions drawn on the T-mesh in physical space. We often drop the distinction between the index and physical space representation for T-junction extensions and use the physical space representation.

A T-junction extension is formed in a manner similar to what was described for the construction of local knot interval vectors in Section 2.2.1. A face extension is created by marching from the T-junction, in the direction of a missing edge (thus spanning T-mesh faces), until two perpendicular edges or vertices are intersected. The direction of an extension is always away from its T-junction. An edge extension is then formed *only* if an edge is attached to the T-junction in the opposite direction. If so, the extension is formed by marching in the opposite direction of the face extension until the edge's opposite vertex is encountered (thus spanning a T-mesh edge). Since T-junction extensions are closed line segments, a horizontal and vertical extension can intersect either on the interior of both extensions or at the endpoint of one extension or both extensions.

As an example, consider T-junction E in Fig. 7b. The face extension points to the right and intersects the two vertical edges corresponding to indices 5 and 6 along the bottom. Since T-junction E is connected to an edge in the direction opposite the face extension, we also form an edge extension along that edge. The edge extension points to the left and intersects the vertical edge corresponding to index 2 along the bottom. The T-junction extension for T-junction E is composed of the face *and* edge extension.

3.2. The extension graph

Intersecting T-junction extensions in an extended T-mesh T_{ext} can be visualized using an undirected graph. We call this graph the extension graph and denote it by $E(\mathsf{T}_{ext})$. Each node in E corresponds to a single T-junction extension in T_{ext}. If two extensions in T_{ext} intersect then an edge is drawn between the corresponding nodes in E. The extension graph for the extended T-mesh in Fig. 7b is shown in Fig. 8a. In this case there are five intersections represented by the five edges in the graph.

3.3. Analysis-suitable definition

An analysis-suitable T-spline is one whose extended T-mesh is analysis-suitable. An analysis-suitable extended T-mesh is one where no T-junction extensions intersect. In other words, $E(\mathsf{T}_{ext})$ is an empty graph (no edges in the graph). We denote an analysis-suitable T-spline space by $\mathcal{T}_s$ and analysis-suitable T-mesh by T_s. The T-mesh in Fig. 7a is not analysis-suitable. This can be seen

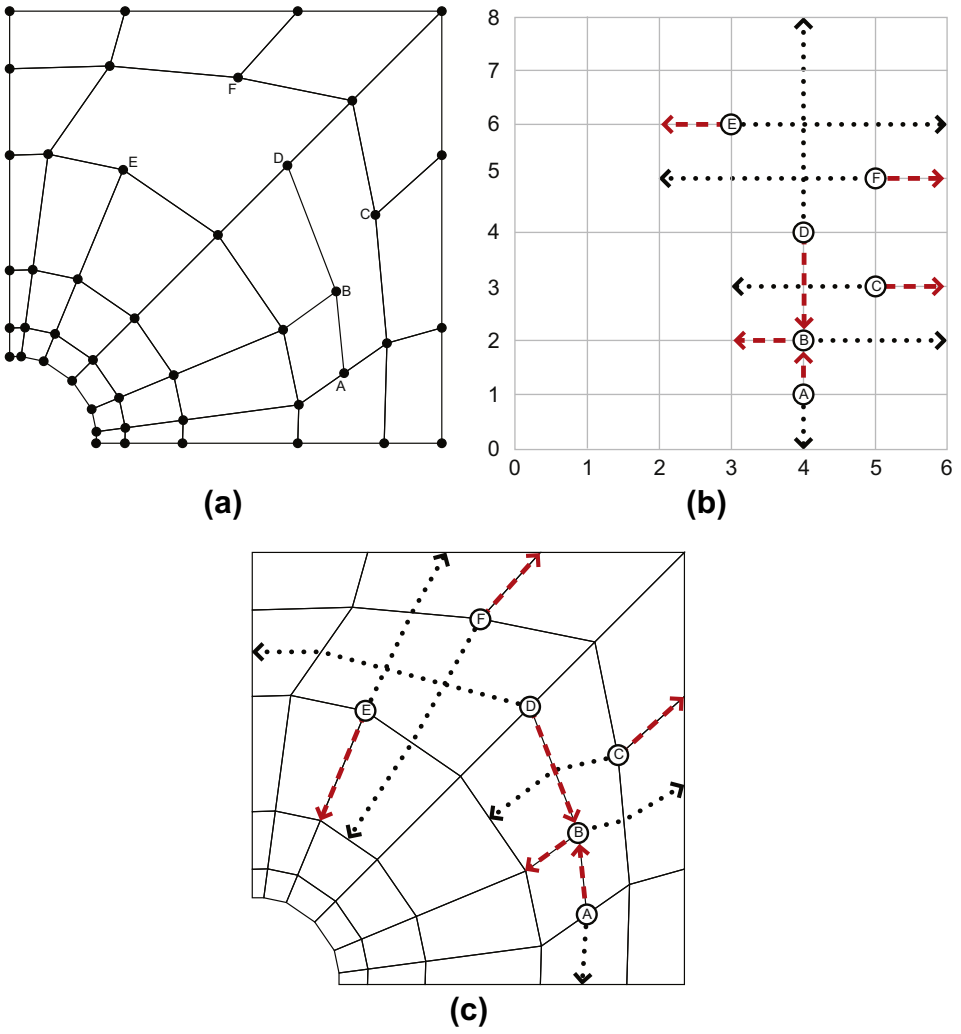


Fig. 7. The extended T-mesh, T_{ext}, in the index and physical space. The dotted arrows (in black) represent face extensions and the dashed arrows (in red) represent edge extensions. The T-junctions are denoted by large circles. (a) A T-mesh T with six T-junctions. (b) The extended T-mesh formed from T and the T-junction extensions in index space. The indexing of the knot structure of T is along the bottom and left. (c) The extended T-mesh in physical space. Note that the directionality of each extension is always determined in the index space of the T-mesh. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

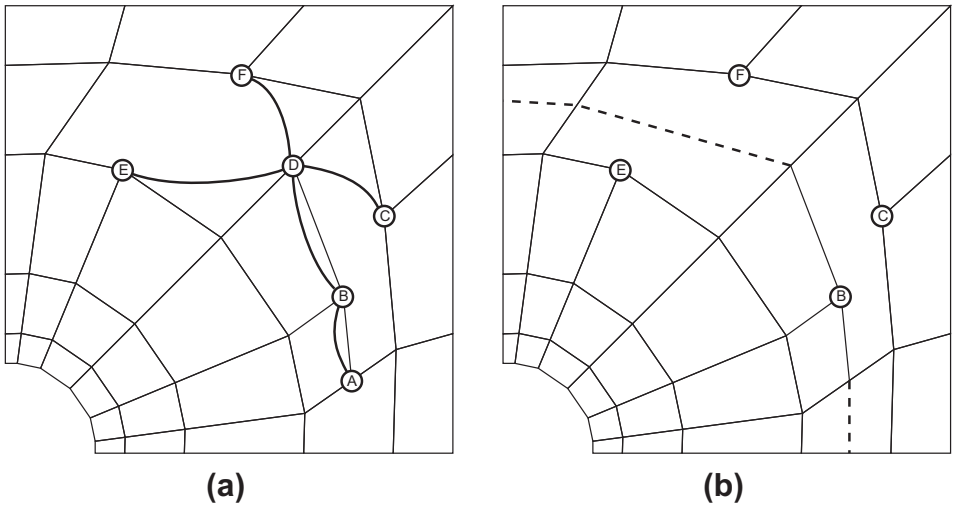


Fig. 8. The extension graph, $\mathsf{E}(\mathsf{T}_{ext})$. (a) The extension graph $E(\mathsf{T}_{ext})$ corresponding to T_{ext} in Fig. 7b and c. The five edges in the graph correspond to the five intersections between T-junction extensions in T_{ext}. (b) The T-mesh in Fig. 7a can be made analysis-suitable by adding the bold dashed edges. The extension graph for this new T-mesh is empty (no edges.)

by inspecting the extension graph in Fig. 8a which has five edges. By adding the dashed edges in Fig. 8b the T-mesh becomes analysis-suitable because its extension graph is empty.

3.4. Partition of unity

An analysis-suitable T-spline basis forms a partition of unity. In other words, $\sum_{A=1}^{n} N_A = 1$. The partition of unity property is important for both geometry and analysis because it assures affine covariance and exact satisfaction of all patch tests. In the context of T-splines, the partition of unity property can be described in two ways. The equation for *any* T-spline surface is

$$T = \frac{\sum_{A=1}^{n} \mathbf{P}_A w_A N_A}{\sum_{A=1}^{n} w_A N_A}, \tag{11}$$

where the $\mathbf{P}_A$ are control points, w_A are weights, and N_A are blending functions. This equation can also be written as

$$T = \sum_{A=1}^{n} \frac{w_A N_A}{\sum_{B=1}^{n} w_B N_B} \mathbf{P}_A = \sum_{A=1}^{n} R_A \mathbf{P}_A \tag{12}$$

in which case the R_A may be *rational* T-spline blending functions. It is clear that the R_A always sum to one, regardless of the choice of w_A. In general, affine covariance only requires that the R_A form a partition of unity, not the N_A. However, it is shown in [51] that for an analysis-suitable T-spline with all $w_A = 1$, $R_A \equiv N_A$. Thus, when we say that analysis-suitable T-splines form a partition of unity, we mean that both the N_A and R_A sum to one. This stronger notion of partition of unity is also a property of NURBS.

3.5. The analysis-suitable elemental T-mesh

For analysis-suitable T-splines, the elemental T-mesh, T_{elem} (see Section 2.3), can be formed by simply adding the face extensions to T [51]. This greatly simplifies the construction of T_{elem} since it is *not* necessary to inspect the knot lines in the T-spline basis.

4. Local refinement of analysis-suitable T-splines

The T-spline local refinement algorithm presented in [41] may add many superfluous control points to a T-mesh during refinement. Additionally, using this algorithm to locally refine an analysis-suitable T-spline often results in a refined T-spline which is *not* analysis-suitable.

This behavior can be attributed to the generality of the algorithm, which is designed to operate on *any* T-spline. No assumptions are made about the topological characteristics of the underlying T-mesh or space. By restricting ourselves to analysis-suitable T-splines, however, we can leverage the structure of the T-mesh to develop a simple local refinement algorithm which only introduces a minimal number of superfluous control points and preserves the properties of an analysis-suitable space.

4.1. Analysis-suitable nesting theory

We say T^2_{ext} is a refinement of T^1_{ext} (denoted $\mathsf{T}^1_{ext} \hat{\subseteq} \mathsf{T}^2_{ext}$) if $\mathsf{T}^1 \subseteq \mathsf{T}^2$ and no face extension endpoint in T^2_{ext} corresponds to a point in the interior of a face extension in T^1_{ext} of the same directionality. Then, if $\mathcal{T}^1_s$ and $\mathcal{T}^2_s$ are analysis-suitable T-spline spaces, $\mathsf{T}^1_{ext} \hat{\subseteq} \mathsf{T}^2_{ext} \rightarrow \mathcal{T}^1_s \subseteq \mathcal{T}^2_s$ (A proof of this result should be forthcoming in [56]). Note that we compare face extensions in the index space of T^2_{ext}.

Nestedness between two T-spline spaces, $\mathcal{T}^1_s$ and $\mathcal{T}^2$ (not necessarily analysis-suitable), can be visualized using a simple modification of the extension graph described in Section 3.2. We call this graph the *coupled extension graph*, $E(\mathsf{T}^{1\rightarrow 2}_{ext})$. $\mathsf{T}^{1\rightarrow 2}_{ext}$, called a *coupled extended T-mesh*, is constructed by adding the face extensions of T^1_{ext} to T^2_{ext}.

To construct $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ we augment $E(\mathsf{T}^2_{ext})$ by adding an additional edge to the graph if a T-junction face extension endpoint in T^2_{ext} is in the interior of a face extension from T^1_{ext} of the same directionality. In that case, a "loop edge" is drawn from the corresponding node in $E(\mathsf{T}^2_{ext})$ to itself creating a small loop in the graph. If T^2_{ext} is analysis-suitable ($E(\mathsf{T}^2_{ext})$ is empty) then the only edges which exist in $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ are "loop edges." The non-existence of loop edges is a necessary condition that $\mathsf{T}^1_{ext} \hat{\subseteq} \mathsf{T}^2_{ext}$. If $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ is an empty graph then $\mathcal{T}^2$ is analysis-suitable and $\mathcal{T}^1_s \subseteq \mathcal{T}^2$. The *weight* of a coupled extension graph is the number of edges in the graph and is denoted by $W(E)$. The weight of a node is the number of edges touching the node.

Fig. 9 illustrates the construction of a coupled extension graph and its use in determining nestedness. An analysis-suitable extended T-mesh T^1_{ext} is shown in Fig. 9a and T^2_{ext} is shown Fig. 9b. We construct the coupled extended T-mesh $\mathsf{T}^{1\rightarrow 2}_{ext}$ in Fig. 9c. Notice that in this case the only face extension from T^1_{ext} which is visible (a light gray dotted arrow in a box) corresponds to T-junction extension B in Fig. 9a. This indicates that the endpoint of T-junction extension C in T^2_{ext} (see Fig. 9b) is in the interior of T-junction extension B in T^1_{ext} (see Fig. 9a.) Fig. 9d shows $E(\mathsf{T}^{1\rightarrow 2}_{ext})$. Since the graph is not empty, $\mathcal{T}^1_s \not\subseteq \mathcal{T}^2$. In fact, the edges between different nodes means the underlying extension graph $E(\mathsf{T}^2_{ext})$ is not empty which implies that $\mathcal{T}^2$ is not analysis-suitable. The loop edge which begins and ends at node C indicates that $\mathsf{T}^1_{ext} \hat{\not\subseteq} \mathsf{T}^2_{ext}$.

4.2. A local refinement algorithm

The example in Fig. 9 motivates a simple approach to local refinement. First, create $\mathsf{T}^2 \supseteq \mathsf{T}^1_s$. As in standard finite element analysis, this is usually done by subdividing a set of T-mesh elements in T^1_s. The T-mesh elements are often selected so as to reduce error in the finite element solution. If $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ is not empty, we must add some additional control points and edges to T^2 to cause $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ to be empty.

To illustrate, Fig. 10a shows the T-mesh T^2 from Fig. 9b, where $\mathcal{T}^1_s \not\subseteq \mathcal{T}^2$ eventhough $\mathsf{T}^1_s \subseteq \mathsf{T}^2$. This can be seen by inspecting the coupled extension graph in Fig. 9d. To ensure nestedness, additional refinement of T^2 must be performed. Fig. 10b–d shows three possible analysis-suitable refinements of T^2 where the resulting coupled extension graph $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ is empty. The dashed edges and open circles in Fig. 10 are T-mesh edges and vertices, respectively, added during local refinement. The minimum of the three refinements is shown in Fig. 10d, where only 6 vertices and 8 edges have been added. We note that Fig. 10b is a NURBS refinement.

This example raises the question of how to devise an algorithm for automatically finding the fewest additional control points and edges that will cause $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ to be empty. Finding the minimal number is an NP-hard problem, so for efficiency, our algorithm uses the following greedy strategy that only provides an approximate minimum. The following steps constitute our analysis-suitable local refinement algorithm:

1. Create $\mathsf{T}^2 \supseteq \mathsf{T}^1_s$.
2. Using T^1_s and T^2, form $E(\mathsf{T}^{1\rightarrow 2}_{ext})$.
3. Of all possible insertion edges, add one into T^2 for which the weight of the resulting $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ is smallest. An *insertion edge* has a vertex that is a T-junction in T^2 and the corresponding node in the coupled extension graph has non-zero weight (see Fig. 11.)
4. Repeat Step 3 until the weight of $E(\mathsf{T}^{1\rightarrow 2}_{ext})$ is zero.
5. Compute the refinement operator $\mathbf{M}$, if desired. See Section 4.3.

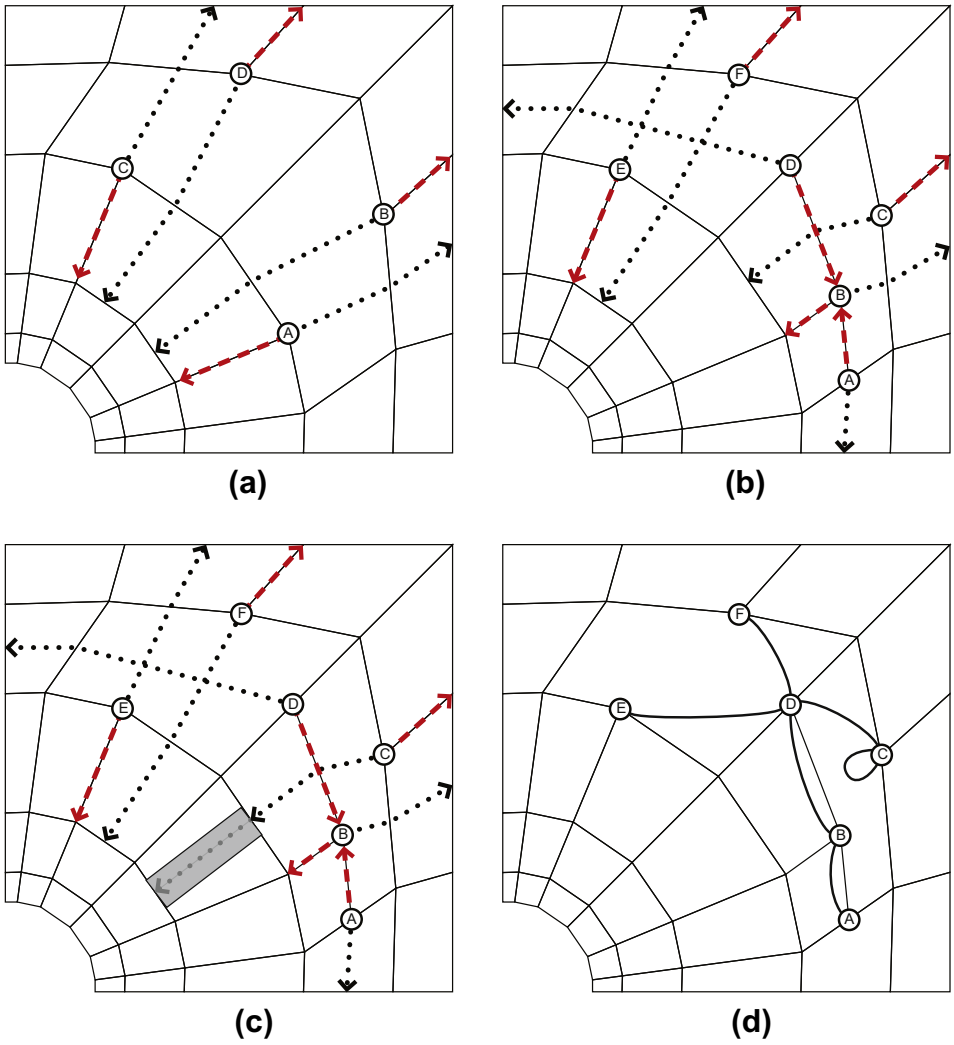


Fig. 9. Determining nestedness. (a) An analysis-suitable extended T-mesh T^1_{ext}. (b) An extended T-mesh T^2_{ext} (not analysis-suitable). (c) Superimposing the face extensions from (a) on T^2_{ext} to form $\mathsf{T}^{1\to 2}_{ext}$. Notice that the only face extension from T^1_{ext} which is visible is in the light gray box. (d) The coupled extension graph for (c). The graph contains edges which implies that $\mathcal{T}^1_s \not\subseteq \mathcal{T}^2$.

Only one insertion edge is inserted during each iteration of the refinement algorithm. Also, there are cases for which the weight will stay the same or increase after the addition of the optimal insertion edge. It should be noted that the algorithm will always terminate, because in the limit, if all T-junctions are extended all the way to a boundary edge, a NURBS is created.

We first demonstrate Steps 1–4 of analysis-suitable local refinement on a simple example. Step 5 is explained in detail in Section 4.3. We begin with the analysis-suitable T-mesh T^1_s shown in Fig. 12a. In Fig. 12b, T^1_s is refined by subdividing several T-mesh elements. The coupled extension graph $E(\mathsf{T}^{1\to 2}_{ext})$ can then be constructed from the coupled extended T-mesh $\mathsf{T}^{1\to 2}_{ext}$ as shown in Fig. 12d and c, respectively. Notice that the weight of the graph is 4 so nesting is not assured between $\mathcal{T}^1_s$ and $\mathcal{T}^2$.

We now begin to add insertion edges to T^2 until $E(\mathsf{T}^{1\to 2}_{ext})$ is empty. The result of the first iteration of the algorithm is shown in Fig. 13. The input coupled extension graph is shown in Fig. 13a. The corresponding set of insertion edges is shown in Fig. 13b. Notice that insertion edges are only created for T-junctions where the weight of the corresponding node in $E(\mathsf{T}^{1\to 2}_{ext})$ is nonzero. Each insertion edge is then added to T^2 as a T-mesh edge and the change in graph weight ΔW and total resulting graph weight W are computed as shown in the Table of Fig. 13. The shaded cells are the ΔWs which must be computed during this iteration. Since this is the first iteration all must be computed. The insertion edge with the largest ΔW is then selected and inserted into the T-mesh. For this iteration, insertion edge K (the bold dashed line in Fig. 13b) is inserted into T^2. Notice that in this case insertion edge E could also have been selected. Both have a ΔW of 2.

The second and final iteration of the algorithm is shown in Fig. 14. The coupled extension graph is shown in Fig. 14a. Notice the presence of the new T-mesh edge in T^2 which corresponds to insertion edge K from the previous step. The current insertion edges are shown in Fig. 14b. There are now only three since the previous iteration decoupled nodes A, J, and K. The current values of ΔW and W are shown in the Table of Fig. 14. None of the ΔW cells are shaded which indicates that the values are saved from a previous iteration and not computed. In general, only a small number of ΔWs need to be computed during each step of local refinement. The insertion edge with the largest ΔW is E which, when inserted, drives the total weight of the graph to zero thus terminating the refinement process. The final refined T-mesh is shown in Fig. 15. The new edges are the dashed lines. If desired, the refinement operator **M** can be computed as described in Section 4.3.

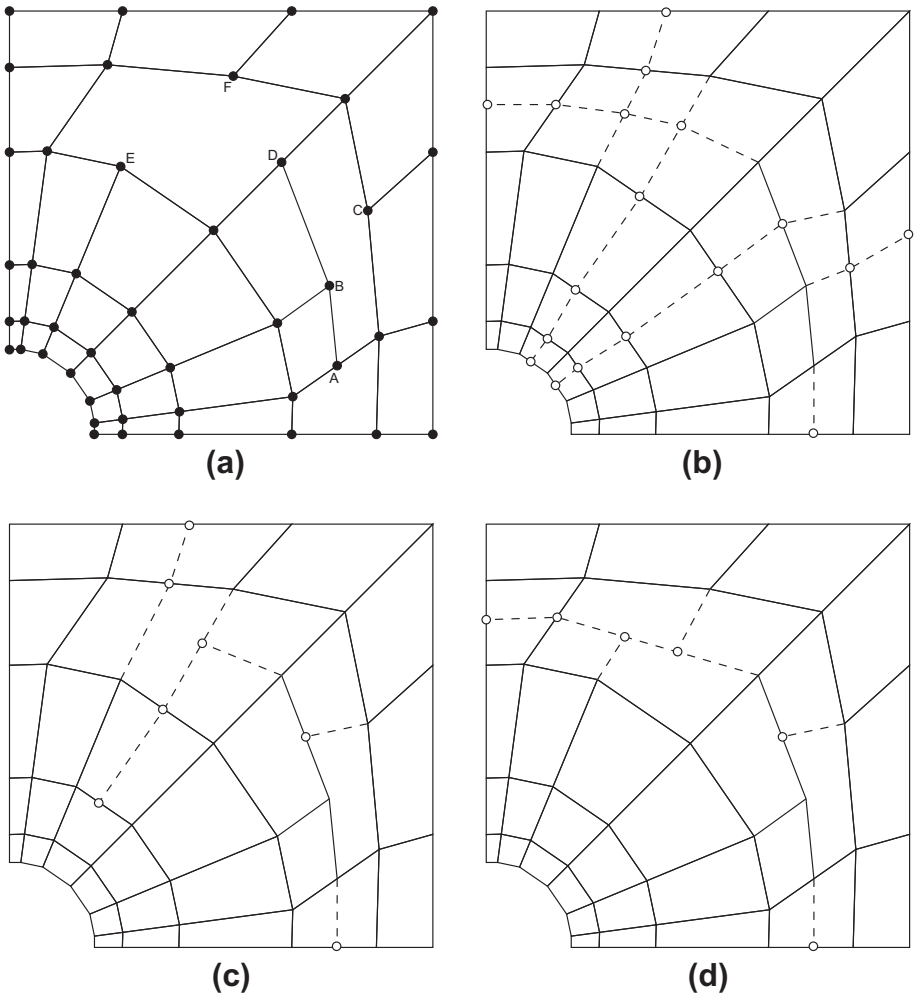


Fig. 10. Several analysis-suitable local refinements for the example in Fig. 9. The dashed edges and open circles are T-mesh edges and vertices, respectively, added during local refinement. (a) The T-mesh T^2 from Fig. 9b. (b) 18 vertices and 19 edges have been added. (c) 7 vertices and 8 edges have been added. (d) 6 vertices and 8 edges have been added.

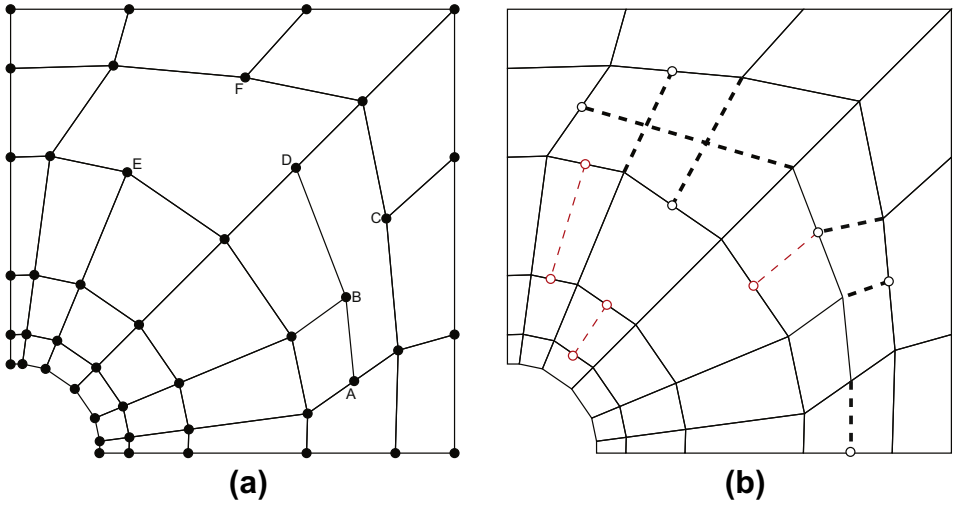


Fig. 11. T-mesh edges which may be inserted during Step 3 of the local refinement algorithm. (a) The T-mesh T^2. (b) The thick black dashed lines are edges which can be inserted into T^2. Notice that each of these edges has a vertex which is a T-junction in T^2 and the corresponding node in the coupled extension graph has non-zero weight (see Fig. 9d.) The thick black dashed lines are called insertion edges. The thin red dashed lines are examples of T-mesh edges which are not inserted to form a refined T-mesh because they do not satisfy the requirements of an insertion edge. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

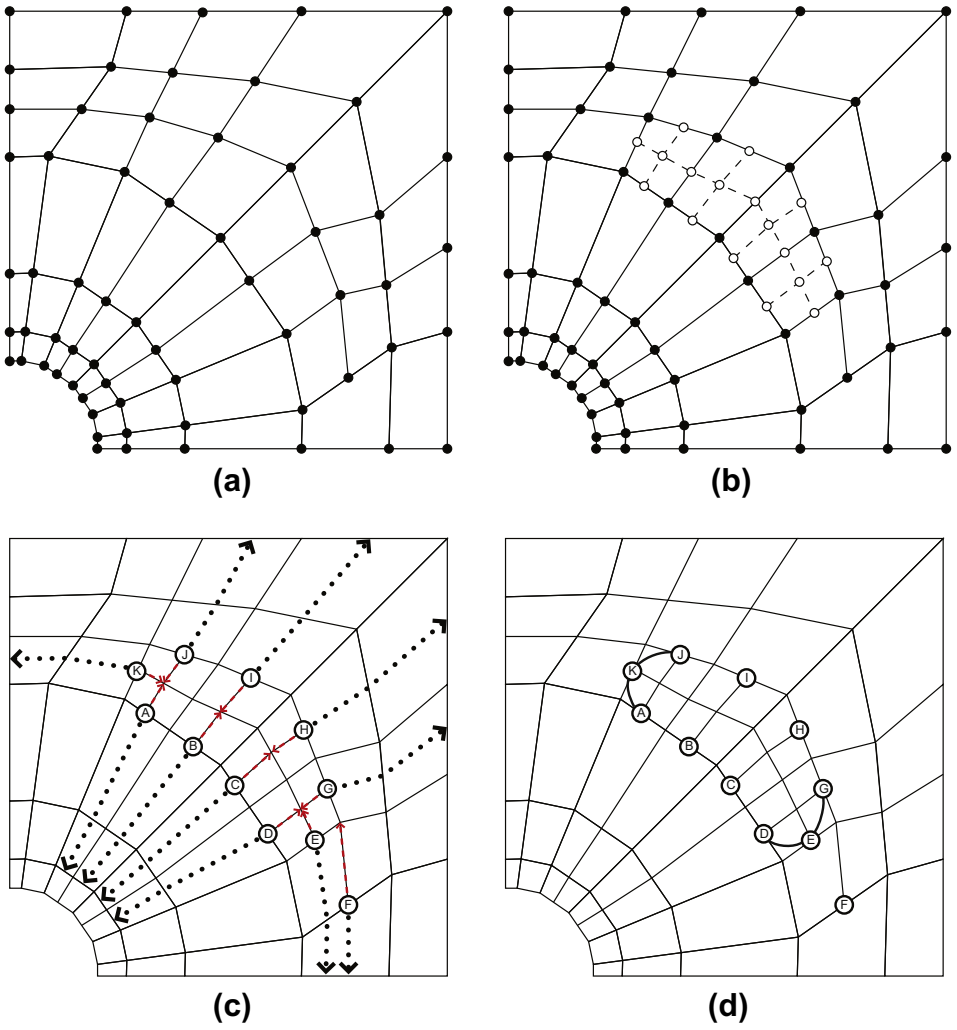


Fig. 12. Initialization of a simple analysis-suitable local refinement example. (a) The initial analysis-suitable T-mesh before refinement. (b) Four T-mesh elements are subdivided to form T^2. (c) The coupled extended T-mesh $\mathrm{T}^{1\to 2}_{ext}$ corresponding to (a) and (b). (d) The coupled extension graph $E(\mathrm{T}^{1\to 2}_{ext})$. The graph is not empty so nesting does not necessarily hold.

4.3. Computing the refinement operator **M**

We now describe how the elements, $m_{A,B}$, of a refinement operator, **M**, are computed. We recall that Steps 1–4 of the analysis-suitable local refinement algorithm in Section 4.2 guarantee that $\mathcal{T}^1_s \subseteq \mathcal{T}^2_s$ and the existence of **M**. We note that all knot comparisons between basis functions are made in a common knot coordinate system defined in the index space of T^2_s (see Section 2.2.2).

Before proceeding, we define several important index space concepts. For a basis function, $N^k_A \in \mathcal{T}^k_s$, we can construct the *index vectors*, $\mathbf{\Gamma}_{A,k} = \left\{ I^i_{A,k} \right\}^2_{i=1}$, where $I^i_{A,k} = \left\{ \tau^{k,i}_{A,1}, \tau^{k,i}_{A,2}, \ldots, \tau^{k,i}_{A,5} \right\}$ and $\tau^{k,i}_{A,j}$ is the index of $\xi^i_{A,j}$ in the index space of T^k. Using $\mathbf{\Gamma}_{A,k}$ we can then define the *index domain* $\Omega^I_{A,k} \subset \mathbb{R}^2$ as

$$\Omega^I_{A,k} = \bigotimes_{i=1}^{2} \Omega^{I,i}_{A,k}, \tag{13}$$

where $\Omega^{I,i}_{A,k} = \left[\tau^{k,i}_{A,1}, \tau^{k,i}_{A,5} \right] \subset \mathbb{R}$.

We first initialize all entries in **M** to zero. Then, for each $N^1_A \in \mathcal{T}^1_s$ and $N^2_B \in \mathcal{T}^2_s$ such that $\Omega^I_{B,2} \subseteq \Omega^I_{A,2}$, we insert $\xi^i_{B,j}$ into Ξ^i_A if $\tau^{2,i}_{B,j} \notin I^i_{A,2}, i = 1,2, j = 1,2,\ldots,5$. This constructs refined local knot vectors, $\overline{\mathbf{\Xi}}_A = \left\{ \overline{\Xi}^i_A \right\}^2_{i=1}$. We then apply basis function refinement (see Section 2.4.1) using $\overline{\mathbf{\Xi}}_A$ to generate the linear combination

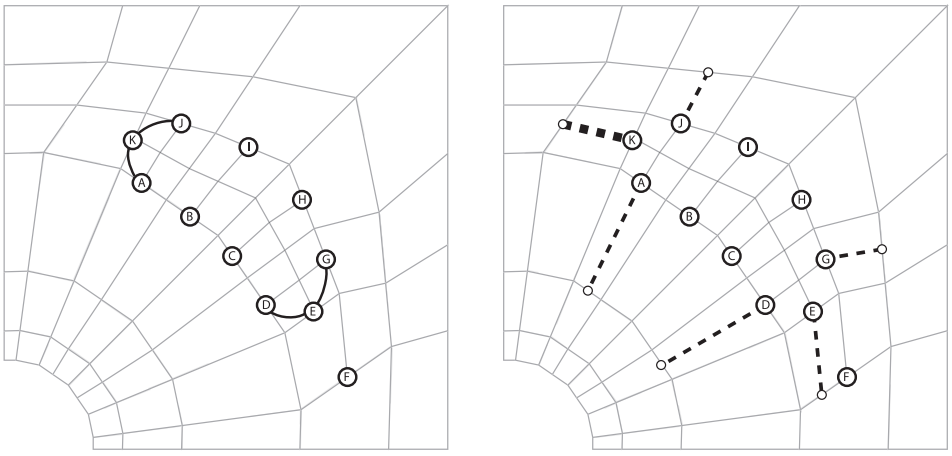
$$N^1_A = \sum_{k=1}^{m} c_{A,k} N_k. \tag{14}$$

If there exists an N_k such that $N_k \equiv N^2_B$, then we set $m_{A,B} = c_{A,k}$. Note that the index vectors and domains for N^1_A are constructed using the index space of T^2_s. This is possible since $\mathrm{T}^1_s \subseteq \mathrm{T}^2_s$.

To illustrate, Fig. 16a shows a T-mesh, T^1_s, where additional vertices and edges (open circles and dashed lines) have been added during the first four steps of the local refinement algorithm in Section 4.2 to form T^2_s. As a result $\mathcal{T}^1_s \subseteq \mathcal{T}^2_s$. The knot intervals used in this example are shown next to the corresponding edges. The index space representation is shown in Fig. 16b. For simplicity we choose a *global* knot coordinate system with an origin at (1,0) in the index space of T^2_s. We note that, in practice, the knot coordinate system can be defined by the subset of knots in T^2 which define N^1_A and N^2_B.

We now compute $m_{A,B}$, where the basis functions $N^1_A \in \mathcal{T}^1_s$ and $N^2_B \in \mathcal{T}^2_s$ are associated with the vertices labeled A and B in Fig. 16. In this case, the index vectors, with respect to the index space of T^2_s, are

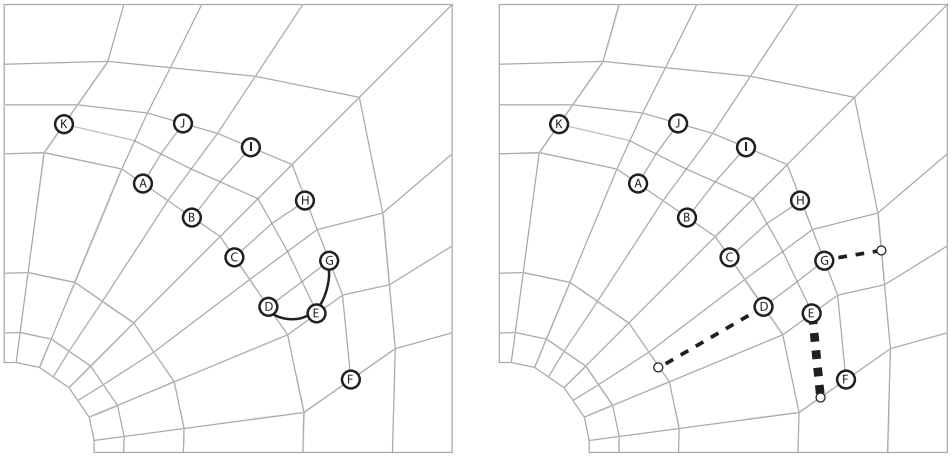
$$\mathbf{\Gamma}_{A,2} = \begin{bmatrix} 1,2,3,5,6 \\ 0,1,2,4,5 \end{bmatrix} \tag{15}$$



(a) Input graph (weight = 4) (b) Insertion edges

	Insertion edge					
	A	D	E	G	J	K
ΔW	1	1	2	1	1	2
W	3	3	2	3	3	2

Fig. 13. The first local refinement iteration for the example in Fig. 12. (a) The input coupled extension graph. This graph has a weight of 4. (b) The corresponding insertion edges. ΔW and W for each insertion edge are shown in the table. ΔW measures the change in graph weight after the edge is inserted into T^2. W is the total graph weight after the edge is inserted into T^2. The shaded cells indicate ΔW values which were computed during this iteration. Insertion edge K minimizes the graph weight and is inserted into the T-mesh as a T-mesh edge. Notice that in this case insertion edge E could also have been selected. Both have a ΔW of 2.



(a) Input graph (weight = 2) (b) Insertion edges

	Insertion edge		
	D	E	F
ΔW	1	2	1
W	1	0	1

Fig. 14. The second and final local refinement iteration for the example in Fig. 12. (a) The input coupled extension graph. This graph has a weight of 2. (b) The corresponding insertion edges. ΔW and W for each insertion edge are shown in the Table. ΔW measures the change in graph weight after the edge is inserted into T^2. W is the total graph weight after the edge is inserted into T^2. Since no ΔW cells are shaded all values are saved from the previous refinement step (see Fig. 13.) Insertion edge E drives the graph weight to zero and is inserted into the T-mesh as a T-mesh edge to complete the refinement process.

and

$$\mathbf{\Gamma}_{B,2} = \begin{bmatrix} 2,3,4,5,6 \\ 0,1,2,3,4 \end{bmatrix}. \tag{16}$$

The index domain, $\Omega^I_{A,2} = [1,6] \otimes [0,5]$, is the union of the dark and lightly shaded rectangles in Fig. 16b and the index domain, $\Omega^I_{B,2} = [2,6] \otimes [0,4]$, is the lightly shaded rectangle.

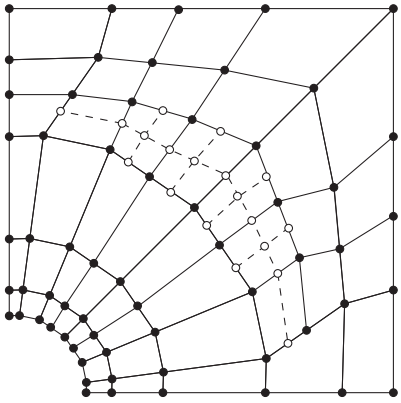


Fig. 15. The final refined T-mesh for the example in Fig. 12. The new edges and vertices are the dashed lines and open circles, respectively.

In this coordinate system, the local knot vectors for N_A^1 and N_B^2 are

$$\Xi_A = \begin{bmatrix} 0,1,2,4,4 \\ 0,0,2,4,4 \end{bmatrix} \tag{17}$$

and

$$\Xi_B = \begin{bmatrix} 1,2,3,4,4 \\ 0,0,2,4,4 \end{bmatrix}. \tag{18}$$

Obviously, $\Omega_{B,2}^l \subseteq \Omega_{A,2}^l$, and since $\tau_{B,3}^{2,1} = 4$ and $\tau_{B,4}^{2,2} = 3$ are not in $\Gamma_{A,2}^1$ and $\Gamma_{A,2}^2$ we insert the corresponding knots $\xi_{B,3}^1 = 3$ and $\xi_{B,4}^2 = 4$ into Ξ_A^1 and Ξ_A^2. This results in the refined local knot vectors

$$\overline{\Xi}_A = \begin{bmatrix} 0,1,2,3,4,4 \\ 0,0,2,4,4,4 \end{bmatrix}. \tag{19}$$

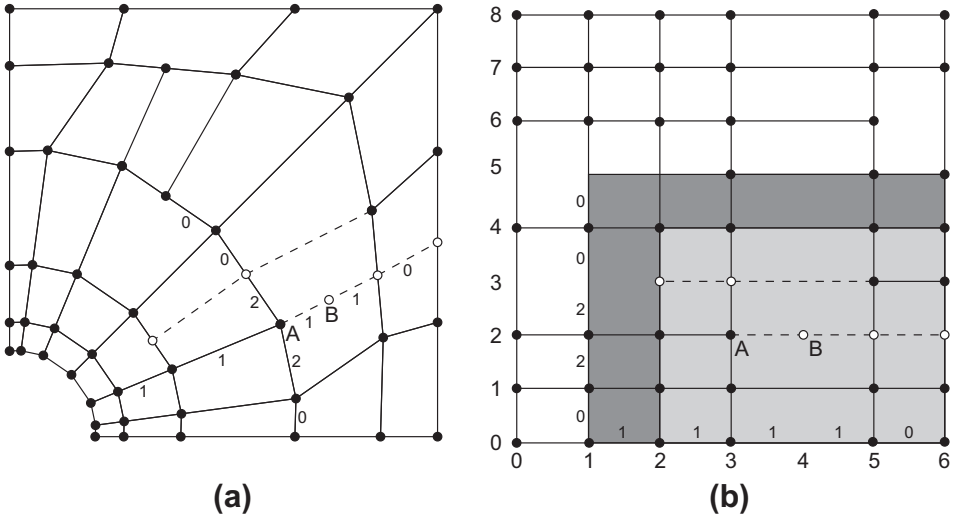


Fig. 16. Computing the refinement coefficient, $m_{A,B}$. (a) The T-mesh, T_s^1, consists of the solid circles and lines. The topology (control points and edges) added during the topology phase of analysis-suitable local refinement (steps 1–4 in Section 4.2) are denoted by the open circles and dashed lines, respectively. The topology phase ensures that $\mathcal{T}_s^1 \subseteq \mathcal{T}_s^2$. (b) The index space of T_s^2. The index domain $\Omega_{A,2}^l = [1,6] \otimes [0,5]$ is the union of the dark and lightly shaded rectangles and the index domain $\Omega_{B,2}^l = [2,6] \otimes [0,4]$ is the lightly shaded rectangle. The origin of the common knot coordinate system is (1,0) in the index space. The knot intervals for the common knot coordinate system are shown in (a) and (b).

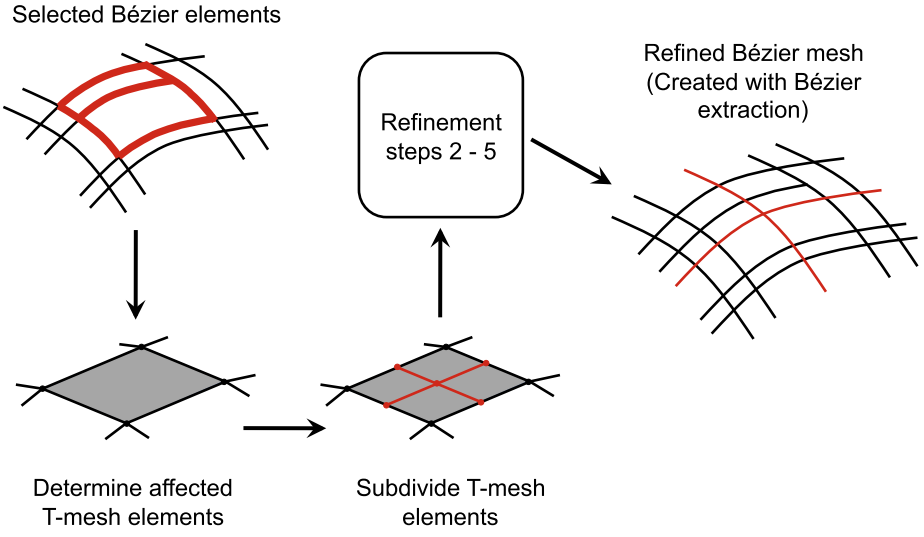


Fig. 17. An analysis-suitable local refinement framework.

Applying the refinement equations, (6)–(8), to $\overline{\Xi}_A$, in each univariate direction, and taking the tensor product of the resulting coefficients results in

$$N_A^1 = c_{A,1}N_1 + c_{A,2}N_2 + c_{A,3}N_3 + c_{A,4}N_4, \tag{20}$$

$$= \frac{3}{4}N_1 + \frac{1}{4}N_3, \tag{21}$$

where

$$\Xi_1 = \begin{bmatrix} 0,1,2,3,4 \\ 0,0,2,4,4 \end{bmatrix} \tag{22}$$

and

$$\Xi_3 = \begin{bmatrix} 1,2,3,4,4 \\ 0,0,2,4,4 \end{bmatrix}. \tag{23}$$

Since $N_3 \equiv N_B^2$, we have that $m_{A,B} = \frac{1}{4}$.

5. Applying analysis-suitable local refinement

We now explore the application and behavior of analysis-suitable T-splines and local refinement. In this example, we use analysis-suitable local refinement to transform an initial coarse

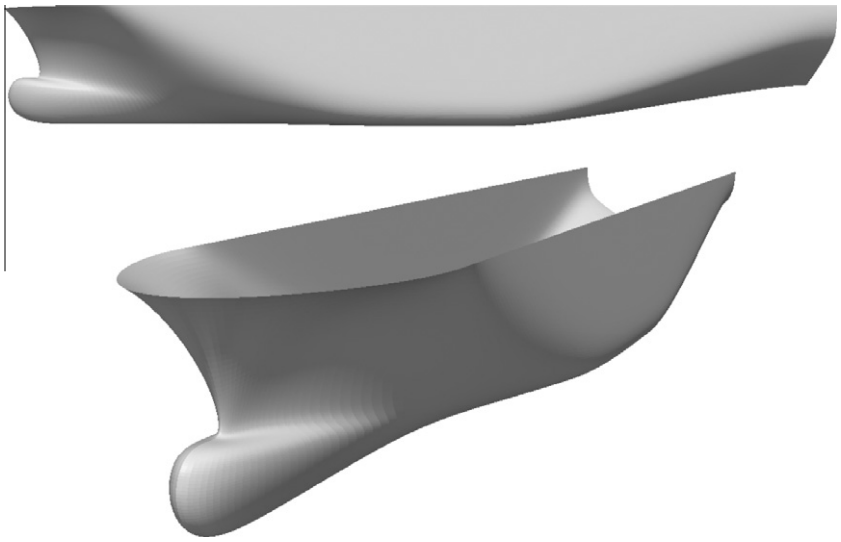


Fig. 18. A T-spline container ship hull. The surface is C^2-continuous everywhere.

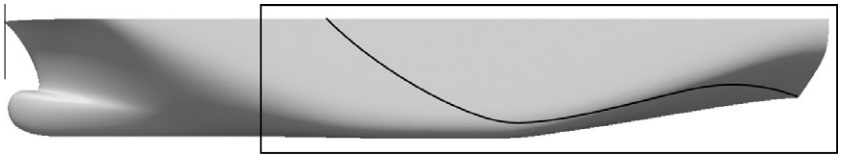


Fig. 19. The regions of the container ship hull where analysis-suitable local refinement will be performed. First, refinement will be performed in the rectangular region followed by highly localized refinement along the curve.

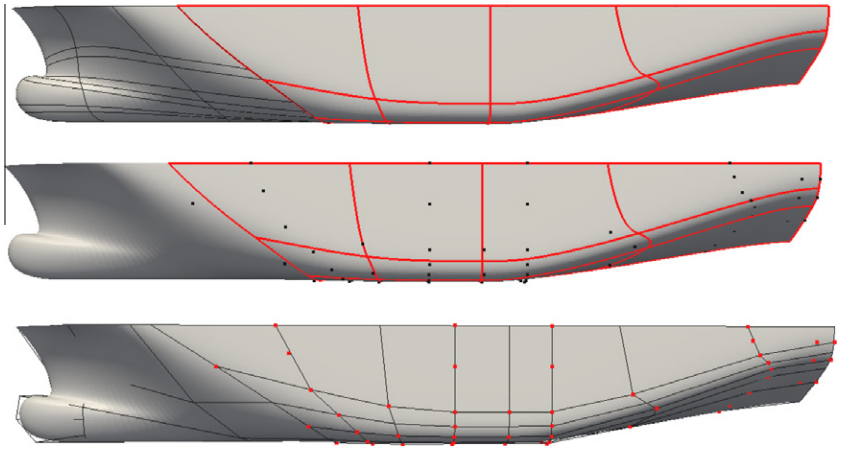


Fig. 20. The first iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown in the middle. The refined T-mesh is shown on the bottom. The new control points are highlighted in red. (Note: Some of the T-mesh edges are hidden behind the ship hull.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

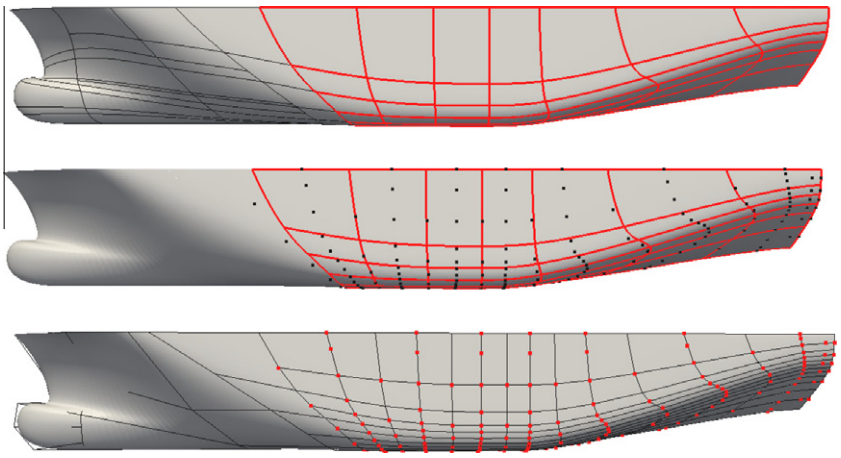


Fig. 21. The second iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown in the middle. The refined T-mesh is shown on the bottom. The new control points are highlighted in red. (Note: Some of the T-mesh edges are hidden behind the ship hull.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

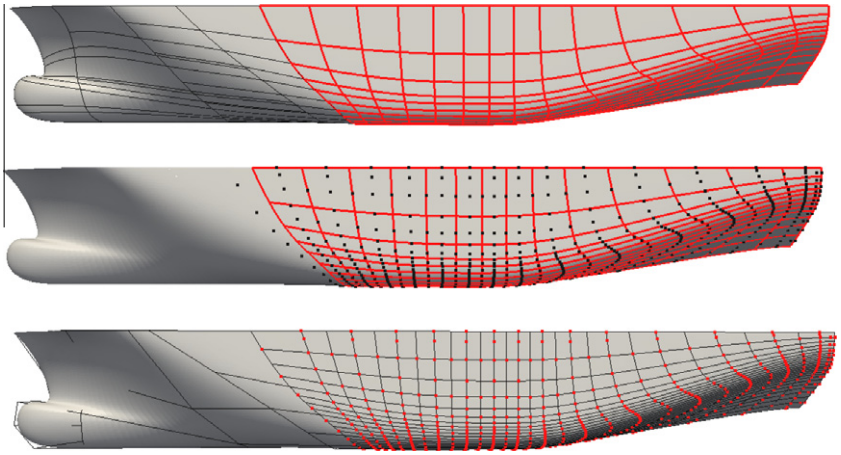


Fig. 22. The third iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown in the middle. The refined T-mesh is shown on the bottom. The new control points are highlighted in red. (Note: Some of the T-mesh edges are hidden behind the ship hull.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

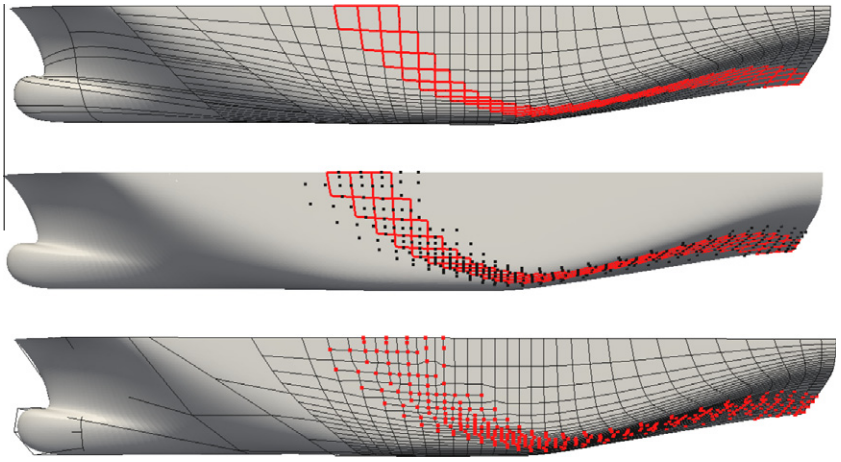


Fig. 23. The fourth iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown in the middle. The refined T-mesh is shown on the bottom. The new control points are highlighted in red. Notice that analysis-suitable local refinement remains localized to the Bézier elements selected for refinement. (Note: Some of the T-mesh edges are hidden behind the ship hull.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

T-spline ship hull design into an analysis-suitable model. The analysis-suitable model can then be used directly in isogeometric analysis by way of Bézier extraction [47]. We note that this same approach can also be used as an adaptive finite element solution strategy. A demanding application of adaptive T-spline local refinement in the context of a phase-field fracture model is described in [57].

In Fig. 17, we schematically illustrate the process used to perform analysis-suitable local refinement. First, a set of Bézier elements, generated using Bézier extraction, is flagged by the user or finite element solver. Next, the Bézier elements are used to identify corresponding T-mesh elements. We recall that several Bézier elements may correspond to a single T-mesh element as described in Section 2.3 and shown in Fig. 17 on the left. The selected T-mesh elements are then refined to generate T^2 as described in Section 4.2. In this case, the T-mesh elements are simply subdivided. Once the selected T-mesh elements are subdivided, Steps 2–5 of the local refinement algorithm presented in Section 4.2 are applied. This generates the final refined analysis-suitable T-spline space. Bézier extraction is then performed resulting in a new set of Bézier elements. This process is then repeated until the resolution of the T-spline is sufficient for the application.

The T-spline container ship hull in Fig. 18 is first designed using the T-spline plugin for Rhino3d [58]. It is then transformed into an analysis-suitable model using local refinement. T-splines are a popular technology in ship hull design because an entire hull can

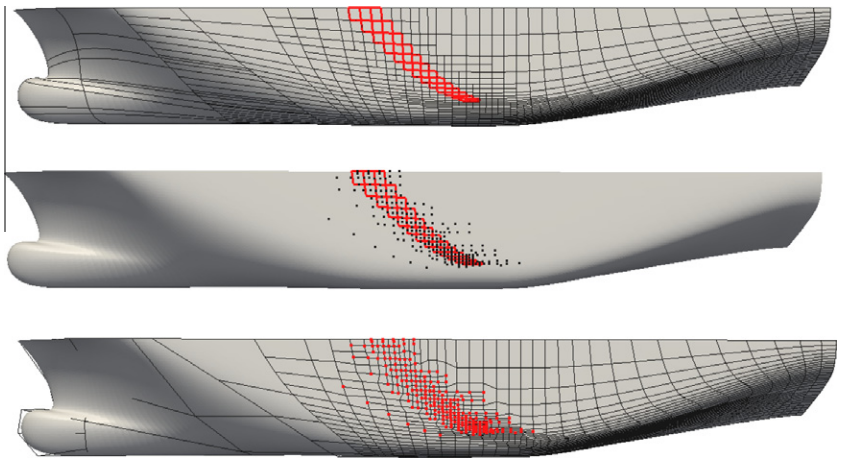


Fig. 24. The fifth iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown in the middle. The refined T-mesh is shown on the bottom. The new control points are highlighted in red. Notice that analysis-suitable local refinement remains localized to the Bézier elements selected for refinement. (Note: Some of the T-mesh edges are hidden behind the ship hull.) (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

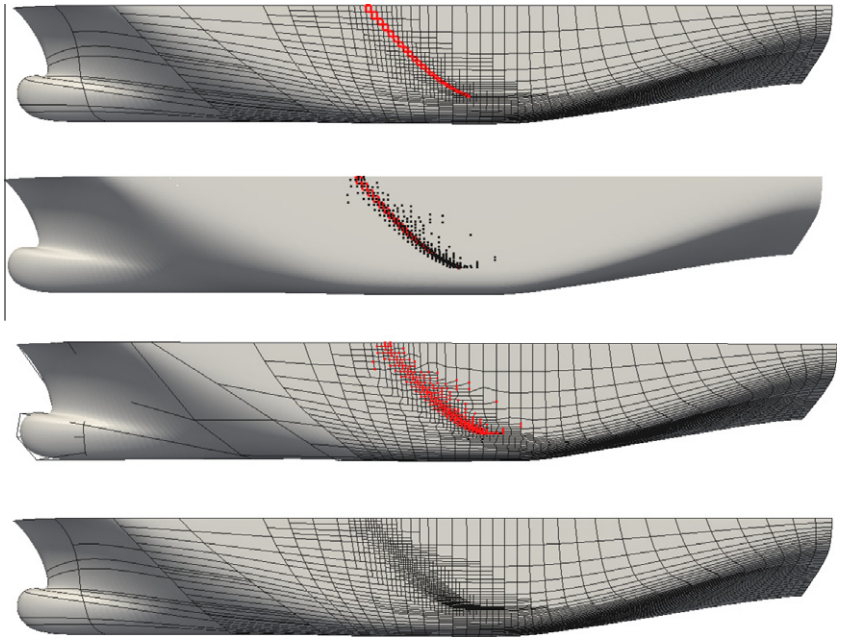


Fig. 25. The sixth and final iteration of analysis-suitable local refinement of the container ship hull in Fig. 18. The Bézier elements are shown on the top with those elements selected for refinement highlighted in red. The control points added during analysis-suitable local refinement are shown below that. The refined final T-mesh is then shown. The new control points are highlighted in red. (Note: Some of the T-mesh edges are hidden behind the ship hull.) The final Bézier element mesh is shown on the bottom. The refinements form a nested sequence of C^2-continuous spline spaces. The geometry of the hull is unchanged during the refinements. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

be modeled by a single watertight surface with a minimal number of control points [59]. In analysis, however, far more degrees-of-freedom are often required to capture the physical phenomenon of interest. In other words, the initial T-mesh must undergo additional refinements to create models that satisfy the needs of analysis.

To demonstrate the pertinent ideas, we assume that the final analysis-suitable model of the hull must be sufficiently resolved to capture the response of the ship in the two regions outlined in Fig. 19. This will require refinements in the region of the hull corresponding to the rectangle followed by highly localized refinements along the region corresponding to the curve. Six iterations of refinement are performed as shown in Figs. 20–25. The initial T-spline of the hull contains just 75 control points and 36 Bézier elements.

The first iteration of local refinement of the ship hull is shown in Fig. 20. A set of Bézier elements is selected for refinement as shown on the top of Fig. 20. The refinement framework described in Fig. 17 is then applied. First, the selected Bézier elements are used to select corresponding T-mesh elements. These T-mesh elements are subdivided and the local refinement algorithm described in Section 4.2 is applied to the resulting subdivided T-mesh. The control points added during local refinement are shown in the middle of Fig. 20. Notice that these control points remain localized to the region of selected Bézier elements. The refined control mesh is shown on the bottom of Fig. 20 with the new control points highlighted. The refined set of Bézier elements is then extracted from the refined T-mesh as shown in Fig. 21 on the top. We note that the transpose of the refinement operator, $\mathbf{M}^T$ (see Sections 2.4.2 and 4.3), is used to update control point positions after each refinement step. This ensures that the geometry and its parameterization are preserved exactly.

The next five iterations of local refinement are shown in Figs. 21–25. In Figs. 21 and 22 the rectangular region in Fig. 19 undergoes additional local refinement. In Figs. 23–25 highly localized refinement is performed along the curve in Fig. 19. Notice that the refinement pattern follows the curve without excessive propagation of control points while preserving the C^2-continuous analysis-suitable T-spline basis. The fully resolved analysis-suitable model has 1722 control points and 1922 Bézier elements. The final Bézier mesh is shown in Fig. 25 on the bottom and the final T-mesh is shown immediately above it. The sequence of C^2-continuous T-spline spaces is nested and the initial geometry is exactly preserved throughout.

6. Conclusion

Analysis-suitable T-splines address mathematical and practical shortcomings observed when general T-spline spaces are used as a basis for isogeometric analysis. Specifically, analysis-suitable T-splines are linearly independent, form a partition of unity, and can be locally refined without excessive propagation. We have developed an analysis-suitable local refinement algorithm for this class of T-splines, and implemented it for bicubic T-spline surfaces.

We have also developed an efficient adaptive framework which combines analysis-suitable T-splines, Bézier extraction, and analysis-suitable local refinement. The sequence of refined spaces is nested and exactly preserves the initial geometry. The basis of the refined T-spline spaces maintains the smoothness of the initial basis. We have demonstrated its effectiveness on a real-world example of a ship hull design. The procedures described provide a powerful methodology for instantiating the vision of isogeometric analysis. In future work, we plan to describe the analysis-suitable treatment of "extraordinary points.".

Acknowledgements

This work was supported by Grants from the Office of Naval Research (N00014-08-C-0920 and N00014-08-1-0992), the National Science Foundation (CMI-0700807), SINTEF (UTA10-000374), the National Science Foundation of China (11031007 and 60903148), the Chinese Universities Scientific Fund, the Chinese Academy of Science (Startup Scientific Research Foundation), and SRF for ROCS SE. M.A. Scott was partially supported by an ICES CAM Graduate Fellowship. This support is gratefully acknowledged.

References

[1] T.J.R. Hughes, J.A. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4135–4195.
[2] J.A. Cottrell, T.J.R. Hughes, Y. Bazilevs, Isogeometric Analysis: Toward Integration of CAD and FEA, Wiley, Chichester, 2009.
[3] J.A. Cottrell, A. Reali, Y. Bazilevs, T.J.R. Hughes, Isogeometric analysis of structural vibrations, Comput. Methods Appl. Mech. Engrg. 195 (2006) 5257–5296.
[4] J.A. Evans, Y. Bazilevs, I. Babuška, T.J.R. Hughes, *n*-Widths, sup-infs, and optimality ratios for the *k*-version of the isogeometric finite element method, Comput. Methods Appl. Mech. Engrg. 198 (21–26) (2009) 1726–1741.
[5] I. Akkerman, Y. Bazilevs, V. Calo, T.J.R. Hughes, S. Hulshoff, The role of continuity in residual-based variational multiscale modeling of turbulence, Comput. Mech. 41 (2008) 371–378.
[6] Y. Bazilevs, V.M. Calo, J.A. Cottrell, T.J.R. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Comput. Methods Appl. Mech. Engrg. 197 (2007) 173–201.
[7] Y. Bazilevs, I. Akkerman, Large eddy simulation of turbulent Taylor–Couette flow using isogeometric analysis and residual-based variational multiscale method, J. Comput. Phys. 229 (2010) 3402–3414.
[8] Y. Bazilevs, C. Michler, V.M. Calo, T.J.R. Hughes, Isogeometric variational multiscale modeling of wall-bounded turbulent flows with weakly enforced boundary conditions on unstretched meshes, Comput. Methods Appl. Mech. Engrg. 199 (13-16) (2010) 780–790.
[9] Y. Bazilevs, V.M. Calo, Y. Zhang, T.J.R. Hughes, Isogeometric fluid-structure interaction analysis with applications to arterial blood flow, Comput. Mech. 38 (2006) 310–322.
[10] Y. Bazilevs, V.M. Calo, T.J.R. Hughes, Y. Zhang, Isogeometric fluid–structure interaction: Theory, algorithms, and computations, Comput. Mech. 43 (2008) 3–37.
[11] Y. Zhang, Y. Bazilevs, S. Goswami, C. Bajaj, T.J.R. Hughes, Patient-specific vascular NURBS modeling for isogeometric analysis of blood flow, Comput. Methods Appl. Mech. Engrg. 196 (2007) 2943–2959.
[12] Y. Bazilevs, J.R. Gohean, T.J.R. Hughes, R.D. Moser, Y. Zhang, Patient-specific isogeometric fluid-structure interaction analysis of thoracic aortic blood flow due to implantation of the Jarvik 2000 left ventricular assist device, Comput. Methods Appl. Mech. Engrg. 198 (45–46) (2009) 3534–3550.
[13] F. Auricchio, L.B. da Veiga, C. Lovadina, A. Reali, The importance of the exact satisfaction of the incompressibility constraint in nonlinear elasticity: Mixed FEMs versus NURBS-based approximations, Comput. Methods Appl. Mech. Engrg. 199 (2010) 314–323.
[14] F. Auricchio, L.B. da Veiga, A. Buffa, C. Lovadina, A. Reali, G. Sangalli, A fully "locking-free" isogeometric approach for plane linear elasticity problems: A stream function formulation, Comput. Methods Appl. Mech. Engrg. 197 (2007) 160–172.
[15] T. Elguedj, Y. Bazilevs, V.M. Calo, T.J.R. Hughes, $\bar{B}$ and $\bar{F}$ projection methods for nearly incompressible linear and non-linear elasticity and plasticity using higher-order NURBS elements, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2732–2762.
[16] J.A. Cottrell, T.J.R. Hughes, A. Reali, Studies of refinement and continuity in isogeometric analysis, Comput. Methods Appl. Mech. Engrg. 196 (2007) 4160–4183.
[17] D.J. Benson, Y. Bazilevs, M.C. Hsu, T.J.R. Hughes, Isogeometric shell analysis: The Reissner–Mindlin shell, Comput. Methods Appl. Mech. Engrg. 199 (5–8) (2010) 276–289.
[18] D.J. Benson, Y. Bazilevs, M.C. Hsu, T.J.R. Hughes, A large deformation, rotation-free, isogeometric shell, Int. J. Numer. Methods Eng. 200 (2011) 1367–1378.
[19] D.J. Benson, Y. Bazilevs, E. De Luycker, M.C. Hsu, M.A. Scott, T.J.R. Hughes, T. Belytschko, A generalized finite element formulation for arbitrary basis functions: From isogeometric analysis to XFEM, Int. J. Numer. Methods Engrg. 83 (2010) 765–785.
[20] R. Echter, M. Bischoff, Numerical efficiency, locking and unlocking of NURBS finite elements, Comput. Methods Appl. Mech. Engrg. 199 (5-8) (2010) 374–382.
[21] J. Kiendl, Y. Bazilevs, M.C. Hsu, R. Wuechner, K.U. Bletzinger, The bending strip method for isogeometric analysis of Kirchhoff–Love shell structures comprised of multiple patches, Comput. Methods Appl. Mech. Engrg. 199 (37-40) (2010) 2403–2416.

[22] H. Gomez, V.M. Calo, Y. Bazilevs, T.J.R. Hughes, Isogeometric analysis of the Cahn–Hilliard phase-field model, Comput. Methods Appl. Mech. Engrg. 197 (2008) 4333–4352.
[23] H. Gomez, T.J.R. Hughes, X. Nogueira, V.M. Calo, Isogeometric analysis of the isothermal Navier–Stokes–Korteweg equations, Comput. Methods Appl. Mech. Engrg. 199 (25-28) (2010) 1828–1840.
[24] S. Lipton, J.A. Evans, Y. Bazilevs, T. Elguedj, T.J.R. Hughes, Robustness of isogeometric structural discretizations under severe mesh distortion, Comput. Methods Appl. Mech. Engrg. 199 (5–8) (2010) 357–373.
[25] W.A. Wall, M.A. Frenzel, C. Cyron, Isogeometric structural shape optimization, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2976–2988.
[26] X. Qian, Full analytical sensitivities in NURBS based isogeometric shape optimization, Comput. Methods Appl. Mech. Engrg. 199 (29–32) (2010) 2059–2071.
[27] A.P. Nagy, M.M. Abdalla, Z. Gurdal, Isogeometric sizing and shape optimization of beam structures, Comput. Methods Appl. Mech. Engrg. 199 (17–20) (2010) 1216–1230.
[28] A.P. Nagy, M.M. Abdalla, Z. Gurdal, On the variational formulation of stress constraints in isogeometric design, Comput. Methods Appl. Mech. Engrg. 199 (41–44) (2010) 2687–2696.
[29] A. Buffa, G. Sangalli, R. Vazquez, Isogeometric analysis in electromagnetics: B-splines approximation, Comput. Methods Appl. Mech. Engrg. 199 (17–20) (2010) 1143–1152.
[30] E. Cohen, T. Martin, R.M. Kirby, T. Lyche, R.F. Riesenfeld, Analysis-aware modeling: Understanding quality considerations in modeling for isogeometric analysis, Comput. Methods Appl. Mech. Engrg. 199 (5–8) (2010) 334–356.
[31] J. Lu, Circular element: Isogeometric elements of smooth boundary, Comput. Methods Appl. Mech. Engrg. 198 (30-32) (2009) 2391–2402.
[32] H. Kim, Y. Seo, S. Youn, Isogeometric analysis for trimmed CAD surfaces, Comput. Methods Appl. Mech. Engrg. 198 (37–40) (2009) 2982–2995.
[33] P. Costantini, C. Manni, F. Pelosi, M.L. Sampoli, Quasi-interpolation in isogeometric analysis based on generalized B-splines, Comput. Aid. Geometric Des. 27 (8) (2010) 656–668.
[34] M. Aigner, C. Heinrich, B. Juettler, E. Pilgerstorfer, B. Simeon, A.V. Vuong, Swept volume parameterization for isogeometric analysis, in: E.R. Hancock, R.R. Martin, M.A. Sabin (Eds.), Mathematics of Surfaces XIII, Lecture Notes in Computer Science, vol. 5654, 2009, pp. 19–44.
[35] T. Martin, E. Cohen, R.M. Kirby, Volumetric parameterization and trivariate B-spline fitting using harmonic functions, Comput. Aid. Geometric Des. 26 (6) (2009) 648–664.
[36] W. Wang, Y. Zhang, Wavelets-based NURBS simplification and fairing, Comput. Methods Appl. Mech. Engrg. 199 (5-8) (2010) 290–300, doi:10.1016/j.cma.2009.04.003.
[37] W. Wang, Y. Zhang, M.A. Scott, T.J.R. Hughes, Converting an unstructured quadrilateral mesh to a standard T-spline surface, Comput. Mech. 48 (2011) 477–498.
[38] T.W. Sederberg, J. Zheng, A. Bakenov, A. Nasri, T-splines and T-NURCCs, ACM Trans. Graph. 22 (2003) 477–484. <http://doi.acm.org.ezproxy.lib.utexas.edu/10.1145/882262.882295>.
[39] T.W. Sederberg, G.T. Finnigan, X. Li, H. Lin, Watertight trimmed NURBS, ACM Trans. Graph. 27 (2008) 1–79. <http://doi.acm.org.ezproxy.lib.utexas.edu/10.1145/1360612.1360678>.
[40] H. Ipson, T-spline merging, Master's thesis, Brigham Young University (April 2005).
[41] T.W. Sederberg, D.L. Cardon, G.T. Finnigan, N.S. North, J. Zheng, T. Lyche, T-spline simplification and local refinement, ACM Trans. Graph. 23 (2004) 276–283. <http://doi.acm.org.ezproxy.lib.utexas.edu/10.1145/1015706.1015715>.
[42] D.R. Forsey, R.H. Bartels, Hierarchical B-spline refinement, ACM SIGGRAPH Comput. Graph. 22 (4) (1988) 205–212.
[43] Y. Bazilevs, V.M. Calo, J.A. Cottrell, J.A. Evans, T.J.R. Hughes, S. Lipton, M.A. Scott, T.W. Sederberg, Isogeometric analysis using T-splines, Comput. Methods Appl. Mech. Engrg. 199 (5-8) (2010) 229–263.
[44] M. Dörfel, B. Jüttler, B. Simeon, Adaptive isogeometric analysis by local h-refinement with T-splines, Comput. Methods Appl. Mech. Engrg. 199 (5–8) (2009) 264–275.
[45] L. Beirão da Veiga, A. Buffa, D. Cho, G. Sangalli, Isogeometric analysis using T-splines on two-patch geometries, Comput. Methods Appl. Mech. Engrg. 200 (21–22) (2011) 1787–1803.
[46] M.J. Borden, M.A. Scott, J.A. Evans, T.J.R. Hughes, Isogeometric finite element data structures based on Bézier extraction of NURBS, Int. J. Numer. Methods Engrg. 87 (2011) 15–47.
[47] M.A. Scott, M.J. Borden, C.V. Verhoosel, T.W. Sederberg, T.J.R. Hughes, Isogeometric finite element data structures based on Bézier Extraction of T-splines, Int. J. Numer. Methods Engrg. 88 (2011) 126–156.
[48] C.V. Verhoosel, M.A. Scott, R. de Borst, T.J.R. Hughes, An isogeometric approach to cohesive zone modeling, Int. J. Numer. Methods Engrg., in press, doi:10.1002/nme.3061.
[49] C.V. Verhoosel, M.A. Scott, T.J.R. Hughes, R. de Borst, An isogeometric analysis approach to gradient damage models, Int. J. Numer. Methods Engrg. 86 (2011) 115–134.
[50] X. Li, J. Zheng, T.W. Sederberg, T.J.R. Hughes, M.A. Scott, On linear independence of T-spline blending functions, Comput. Aid. Geometric Des. 29 (2012) 63–76.
[51] X. Li, J. Zheng, T.W. Sederberg, On T-spline classification, in preparation.
[52] G.T. Finnigan, Arbitrary degree T-splines, Master's thesis, Brigham Young University (August 2008).
[53] T.W. Sederberg, J. Zheng, X. Song, Knot intervals and multi-degree splines, Comput. Aid. Geometric Des. 20 (2003) 455–468.
[54] T.J.R. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Dover Publications, Mineola, NY, 2000.
[55] A. Buffa, D. Cho, G. Sangalli, Linear independence of the T-spline blending functions associated with some particular T-meshes, Comput. Methods Appl. Mech. Engrg. 199 (23–24) (2010) 1437–1445.
[56] X. Li, M.A. Scott, On the nesting theory of T-splines, in preparation.
[57] M.J. Borden, M.A. Scott, C.V. Verhoosel, C.M. Landis, T.J.R. Hughes, Phase-field modeling of dynamic fracture using isogeometric analysis, Comput. Methods Appl. Mech. Engrg., submitted for Publication.
[58] T-Splines, Inc., 2011. <http://www.tsplines.com/rhino/>.
[59] M.T. Sederberg, T.W. Sederberg, T-splines: A technology for marine design with minimal control points, 2010. <http://www.tsplines.com/technicalpapers.html>.