Computer Aided Geometric Design 46 (2016) 16–29

Contents lists available at ScienceDirect

Computer Aided Geometric Design

www.elsevier.com/locate/cagd

# On degree elevation of T-splines ☆

Jingjing Zhang, Xin Li *

*University of Science and Technology of China, Hefei, Anhui, PR China*

ARTICLE INFO

*Article history:*
Received 29 July 2015
Received in revised form 4 May 2016
Accepted 19 May 2016
Available online 2 June 2016

*Keywords:*
T-splines
Analysis-suitable T-splines
Degree elevation

ABSTRACT

A degree elevation algorithm is presented for T-splines. We also provide two optimized degree elevation algorithms to restrict the resulting T-splines to be analysis-suitable.



## 1. Introduction

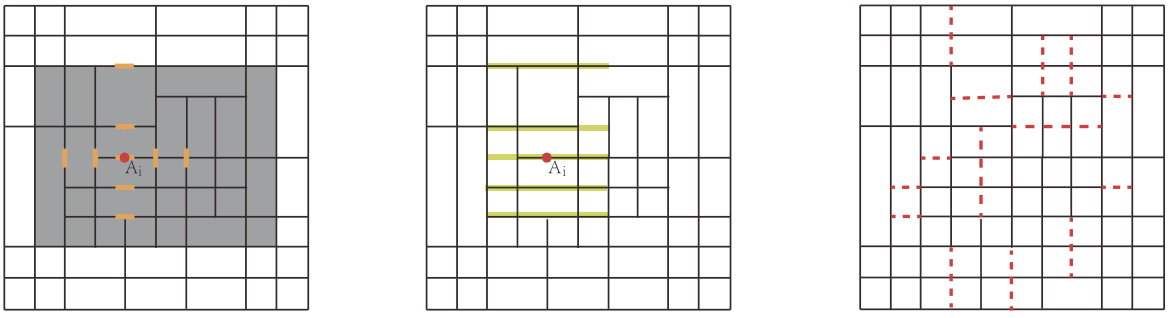
T-splines (Sederberg et al., 2003, 2004) address many limitations inherent in the NURBS representation, such as local refinement (Sederberg et al., 2004; Scott et al., 2012), watertightness via merging (Ipson, 2005; Sederberg et al., 2003) and trimmed NURBS conversion (Sederberg et al., 2008). T-splines have proved to be an important technology across several disciplines including industrial, architectural and engineering design, manufacturing and engineering analysis. Knot insertion and degree elevation algorithms are two fundamental algorithms which are used to enrich a spline space (Farin, 2002). Degree elevation is the process of raising the degree of a curve or a surface while keeping the shape unchanged. For NURBS, these issues have been well studied (Farin, 2002; Wang and Deng, 2007; Huang et al., 2005). For T-splines, the local refinement algorithm has also been well studied (Sederberg et al., 2004; Scott et al., 2012; Morgenstern and Peterseim, 2015). However, no previous articles address degree elevation for T-splines.

Another motivation for T-spline degree elevation is from the analysis community. T-splines are attractive not only in geometric modeling but also in iso-geometric analysis (IGA), which uses the smooth spline basis that defines the geometry as the basis for analysis. IGA is introduced in Hughes et al. (2005) and described in detail in Cottrell et al. (2009). The use of T-splines as a basis for IGA has gained widespread attention (Bazilevs et al., 2010; Scott et al., 2012, 2013; Borden et al., 2012; Benson et al., 2010; Veiga et al., 2011; Buffa et al., 2012; Dimitri et al., 2014; Liu et al., 2014; Schillinger et al., 2014). While the whole class of T-splines are not suitable as a basis for IGA because of possible linear dependence (Buffa et al., 2010), a mildly topological restricted subset of T-splines, analysis-suitable T-splines (AS T-splines), are optimized to meet the needs both for design and analysis (Li et al., 2012; Scott et al., 2012; Veiga et al., 2012, 2013; Li and Scott, 2014). B-spline based IGA uses the operations of h-refinement (knot insertion), p-refinement (degree elevation) and k-refinement (both h and p-refinement are preformed) (Cottrell et al., 2007; da Veiga et al., 2011). The k-refinement provides smoother functions

☆ This paper has been recommended for acceptance by Thomas Sederberg.

* Corresponding author. Tel.: +86 551 63607202.
*E-mail address:* lixustc@ustc.edu.cn (X. Li).

http://dx.doi.org/10.1016/j.cagd.2016.05.010
0167-8396/




a. A T-mesh b. The $hSK(\mathbf{A}_i)$ for one anchor $\mathbf{A}_i$ c. The elemental T-mesh

**Fig. 1.** A bi-degree (2,3) T-mesh. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

and increases the accuracy over the classical $C^0$-continuous p-refinement for the problems of structural vibrations (Cottrell et al., 2007). Thus, the development of k-refinement or even local k-refinement for T-splines is important both for IGA and geometric modeling.

Our goal is to develop algorithms for T-spline degree elevation. Given a bi-degree $(p, q)$ T-spline space $\mathbf{S}^{p,q}(\mathsf{T})$ defined on a T-mesh $\mathsf{T}$, the algorithm finds a new T-mesh $\hat{\mathsf{T}}$ such that $\mathbf{S}^{p,q}(\mathsf{T}) \subseteq \mathbf{S}^{p+1,q+1}(\hat{\mathsf{T}})$. If $\mathsf{T}$ is a tensor-product mesh, then the new mesh $\hat{\mathsf{T}}$ is also a tensor-product mesh by increasing the knot multiplicity, which is the mesh $D(\mathsf{T})$ defined in Section 4.3. However, this is not true for general T-splines because the relationship between $\mathsf{T}$ and $\hat{\mathsf{T}}$ is unknown. Thus, we first provide a recursive algorithm in Section 4 based on degree elevation of each blending function. But in the process of the algorithm, we need to insert some additional vertices and edges such that the blending functions after degree elevation correspond to a valid T-mesh. If we restrict the resulting T-spline to be analysis-suitable, then the relation between T-mesh $\mathsf{T}$ and $\hat{\mathsf{T}}$ is simplified (Theorem 5.3), which enables us to develop two optimized algorithms in Section 5. If the original T-mesh is also analysis-suitable, we can explicitly give the new T-mesh $\hat{\mathsf{T}}$ (Remark 5.4).

The paper is structured as follows. Section 2 provides the background on T-splines and AS T-splines. Section 3 recalls B-splines degree elevation. Section 4 presents a degree elevation algorithm for generic T-splines. Section 5 gives two optimized degree elevation algorithms. The last section is discussion.

## 2. T-splines

### 2.1. Index T-mesh

An index T-mesh (Bazilevs et al., 2010) $\mathsf{T}$ for a bi-degree $(p, q)$ T-spline is a collection of all the elements of a rectangular partition of the index domain $[0, c+p] \times [0, r+q]$, where all rectangle corners (or vertices) have integer coordinates. Denote the *active region* as a rectangle region $[[\frac{p+1}{2}], c+[\frac{p-1}{2}]] \times [[\frac{q+1}{2}], r+[\frac{q-1}{2}]]$, here $[d]$ is the maximal integers equal to or less than $d$. The active region carries the anchors that will be associated with the blending functions while the other indices will be needed for the definition of the local index vector when the anchor is close to the boundary. Fig. 1a shows the active region that is in gray for a bi-degree (2,3) T-mesh. There are three types of elements in a T-mesh:

- Vertex: vertex of a rectangle, denoted as $(\delta_i, \tau_i)$ or $\{\delta_i\} \times \{\tau_i\}$.
- Edge: a line segment connecting two vertices in the T-mesh and no other vertices lying in the interior, denoted as $[\delta_j, \delta_k] \times \{\tau_i\}$ or $\{\delta_i\} \times [\tau_j, \tau_k]$ for a horizontal or vertical edge.
- Face: a rectangle where no other edges and vertices in the interior, denoted as $[\delta_i, \delta_j] \times [\tau_k, \tau_l]$ or $(\delta_i, \delta_j) \times (\tau_k, \tau_l)$, where the second one is for an open face.

The valence of a vertex is the number of edges that touch it. While the algorithms can be extended to any T-mesh topology, for simplicity of the presentation, we do not allow valence one vertices and the only valence two vertices are the four corners. The valence three interior vertices are called T-junctions. We adopt the notations '$\vdash$', '$\dashv$', '$\perp$' and '$\top$' to indicate the four possible orientations for the T-junctions.

An *anchor* is a point in the index T-mesh which corresponds to one blending function. The anchor corresponds to the vertex, the edge midpoint or the face center depending on the degrees. For the $i$-th anchor $\mathbf{A}_i$, we define a local index vector $\vec{\delta}_i \times \vec{\tau}_i$. The values of $\vec{\delta}_i = [\delta_i^0, \ldots, \delta_i^{p+1}]$ and $\vec{\tau}_i = [\tau_i^0, \ldots, \tau_i^{q+1}]$ are determined as follows. From the $i$-th anchor in the T-mesh, we shoot a ray in the $s$ and $t$ direction traversing the T-mesh and collect a total of $p+2$ and $q+2$ knot indices to form $\vec{\delta}_i$ and $\vec{\tau}_i$. If both $p$ and $q$ are odd, then the vertex $(\delta_i^{\frac{p+1}{2}}, \tau_i^{\frac{q+1}{2}})$ corresponds the anchor. If $p$ is even and $q$ is odd, then the horizontal edge $[\delta_i^{\frac{p}{2}}, \delta_i^{\frac{p+2}{2}}] \times \{\tau_i^{\frac{q+1}{2}}\}$ corresponds the anchor. If $p$ is odd and $q$ is even, then the vertical edge $\{\delta_i^{\frac{p+1}{2}}\} \times [\tau_i^{\frac{q}{2}}, \tau_i^{\frac{q+2}{2}}]$ corresponds the anchor. If both $p$ and $q$ are even, then the face $[\delta_i^{\frac{p}{2}}, \delta_i^{\frac{p+2}{2}}] \times [\tau_i^{\frac{q}{2}}, \tau_i^{\frac{q+2}{2}}]$ corresponds the

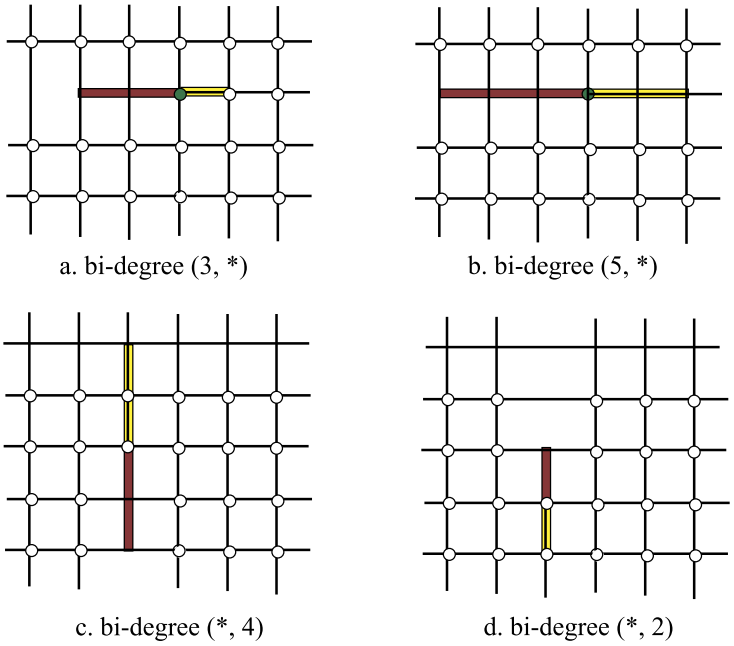



**Fig. 2.** The extensions for four different kinds of T-junctions.

anchor. For example, the local index vectors for the anchor $A_i$ are marked with orange as shown in Fig. 1a. Besides these, we have the following notations for an anchor $\mathbf{A}_i$ in the T-mesh,

- $hSK(\mathbf{A}_i)$ is the union of all the edge segments $[\delta_i^0, \delta_i^{p+1}] \times \{\tau_i^j\}$, $j = 0, 1, \ldots, q+1$;
- $vSK(\mathbf{A}_i)$ is the union of all the edge segments $\{\delta_i^j\} \times [\tau_i^0, \tau_i^{q+1}]$, $j = 0, 1, \ldots, p+1$;
- $SK(\mathbf{A}_i) = hSK(\mathbf{A}_i) \cup vSK(\mathbf{A}_i)$;
- An elemental T-mesh $\mathsf{T}_{elem} = \bigcup_{i=1}^{n_A} SK(\mathbf{A}_i)$, which is a T-mesh formed by all the edges in $SK(\mathbf{A}_i)$. Here $n_A$ is the number of anchors.

### 2.2. Extensions

T-junction extension $ext(T_i)$ is a very important topological issue to define analysis-suitable T-splines, which is a line segment associated with each T-junction $T_i$. For example, for a $i$-th T-junction $(\delta_i, \tau_i)$ of type $\vdash$, the extension for the T-junction is the line segment $[\underline{i}, \bar{i}] \times \{\tau_i\}$. $\underline{i}$ and $\bar{i}$ are determined such that the edges $[\underline{i}, \delta_i) \times \{\tau_i\}$ have $[\frac{p+1}{2}]$ intersections with the T-mesh and the edges $(\delta_i, \bar{i}] \times \{\tau_i\}$ have $[\frac{p}{2}]$ intersections with the T-mesh. For T-junction of type $\dashv$, we can similarly define the extension except the number of intersections is exchanged. Also, we can define the extensions for the other kinds of T-junctions $\perp$, $\top$, where we use degree $q$ instead of $p$. All these extension examples are illustrated in Fig. 2.

### 2.3. T-splines and analysis-suitable T-splines

In an index T-mesh, each index $\delta_i$ and $\tau_j$ corresponds to a knot value $s_{\delta_i}$ and $t_{\tau_j}$, which form two global knot vectors $\vec{\mathbf{s}} = [s_0, s_1, \ldots, s_{c+p}]$ and $\vec{\mathbf{t}} = [t_0, t_1, \ldots, t_{r+q}]$. The end condition knots for $\vec{\mathbf{s}}$ and $\vec{\mathbf{t}}$ may have multiplicity $p+1$ and $q+1$; all other knots are of multiplicity $\leq p$ and $\leq q$ respectively. With the knot vectors, each edge in the T-mesh can be assigned a knot interval which is the difference of the knot values for the two end vertices.

Now we are ready to define the blending function $T_i(s,t)$ associated with the $i$-th anchor,

$$T_i(s,t) = B[s_{\vec{\delta}_i}](s) B[t_{\vec{\tau}_i}](t) \tag{1}$$

where

$$s_{\vec{\delta}_i} = [s_{\delta_i^0}, s_{\delta_i^1}, \ldots, s_{\delta_i^{p+1}}], \qquad t_{\vec{\tau}_i} = [t_{\tau_i^0}, t_{\tau_i^1}, \ldots, t_{\tau_i^{q+1}}] \tag{2}$$

and $B[s_{\vec{\delta}_i}](s)$, $B[t_{\vec{\tau}_i}](t)$ are degree $p$ and $q$ B-spline basis functions defined in terms of knot vector $s_{\vec{\delta}_i}$ and $t_{\vec{\tau}_i}$.

A T-spline space $\mathbf{S}^{p,q}(\mathsf{T})$ is finally given as a linear space spanned by all these blending functions and a T-spline surface is defined as

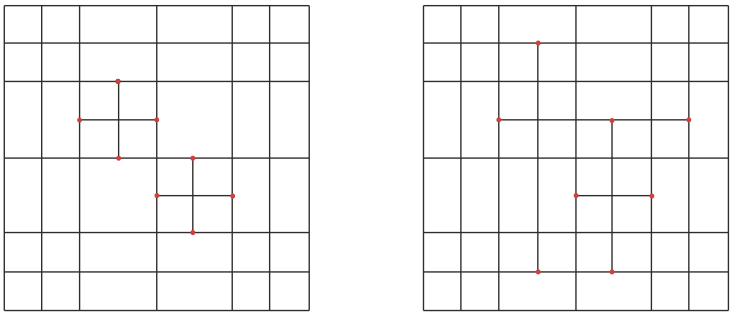


**Fig. 3.** The right T-mesh is a bi-cubic AS T-mesh while the left one is not.

$$\mathbf{T}(s,t) = \sum_{i=1}^{n_A} \mathbf{C}_i T_i(s,t) \tag{3}$$

where $\mathbf{C}_i = (\omega_i x_i, \omega_i y_i, \omega_i z_i, \omega_i) \in \mathbb{P}^3$ are homogeneous control points, $\omega_i \in \mathbb{R}$ are weights, $T_i(s,t)$ are blending functions.

**Definition 2.1.** For a bi-degree $(p,q)$ T-spline, a T-mesh is called analysis-suitable if the extensions for all the T-junctions $\vdash$ and $\dashv$ don't intersect the extensions for all the T-junctions $\perp$ and $\top$. A T-spline defined on an analysis-suitable T-mesh is called an analysis-suitable T-spline, for short AS T-spline. See Fig. 3.

Analysis-suitable T-splines are optimized to meet the needs of both design and analysis. AS T-splines have the following properties: local linear independence (Li et al., 2012; Veiga et al., 2012, 2013; Li, 2015), partition of unity (Li and Scott, 2014; Zhang and Li, 2015), local refinement (Sederberg et al., 2003; Scott et al., 2012), dual basis (Veiga et al., 2012, 2013), characterization and approximation (Li and Scott, 2014). Besides these, one interesting property for AS T-meshes is given in the following lemma from Veiga et al. (2012, 2013), Li (2015). This property tells us that an AS T-mesh can be created from its elemental T-mesh by decreasing extensions.

**Lemma 2.2.** *For an AS T-mesh* $\mathsf{T}$, $\mathsf{T}_{elem} = \mathsf{T} \cup \{ext(T_i) | \text{for all T-junctions } T_i\}$.

## 3. Degree elevation of B-splines curves

This section recalls degree elevation for a B-spline curve and a tensor-product B-spline surface.

Given a knot vector $\vec{s} = [s_0, s_1, \ldots, s_{n+p+1}]$, $s_i \le s_{i+1}$, $i = 0, 1, \ldots, n+p$, a set of degree $p$ B-spline basis functions $B^p[\vec{s}_i](s)$ can be defined in terms of the local knot vector $\vec{s}_i = [s_i, \ldots, s_{i+p+1}]$. If we rewrite the knot vector by getting rid of the multiplicities as $\{s_{i_0} < s_{i_1} < \ldots, s_{i_m}\}$, here $\mu_k = i_{k+1} - i_k$ is the multiplicities of knots $u_{i_k}$. Then the B-spline space can also be defined as

$$\mathbf{S}^p[\vec{s}] := \left\{ f(s) \middle| f(s)|_{[u_{i_k}, u_{i_{k+1}}]} \in \mathbb{P}_p,\ f \text{ is } C^{p-\mu_k} \text{ at } u_{i_k} \right\},$$

where $\mathbb{P}_p$ is the space of all the degree $p$ polynomials.

In order to degree elevate a B-spline curve, we can directly degree elevate each curve segment and keep the continuity at each interior knot, which means that we need to increase the multiplicity of each knot by one to get a new knot vector $\{s_{i_0} < s_{i_1+1} < \ldots, s_{i_m+m}\}$. The new degree $p+1$ basis functions can be defined in terms of the new knot vector. And the new control points can be computed via blossom or the fast degree elevation algorithm (Huang et al., 2005; Farin, 2002).

The same idea can be generalized to perform degree elevation of a tensor-product B-spline surface directly. However, this is not the situation for T-splines because the T-spline space is general a linear space spanned by the blending functions defined from the knots and the T-mesh. There is no characterization for the general T-spline space in terms of the piecewise polynomial space with some continuity constrains. In the following sections, we will provide a recursive algorithm to degree elevation of general T-splines and two optimized degree elevation algorithms if we restrict the resulting T-spline to be analysis-suitable.

## 4. Degree elevation of T-splines

In this section, we provide a recursive degree elevation for general T-splines.

### 4.1. Blending function refinement

For a degree $d$ B-spline basis function $B[\vec{s}](s)$, where $\vec{s}=[s_0,s_1,\ldots,s_{d+1}]$, insert a single knot $s^*$ into $\vec{s}$, then $B[\vec{s}](s)$ can be written into a linear combination of two degree $d$ B-splines:

$$B[\vec{s}](s)=c_1B[\vec{s_l}](s)+c_2B[\vec{s_r}](s), \tag{4}$$

where $\vec{s_l}=[s_0,\ldots,s_i,s^*,s_{i+1},\ldots,s_d]$, $\vec{s_r}=[s_1,\ldots,s_i,s^*,s_{i+1},\ldots,s_{d+1}]$,

$$c_1=\begin{cases}\frac{s^*-s_0}{s_d-s_0} & \text{if } s^*<s_d\\ 1 & \text{if } s^*\geq s_d\end{cases}\quad c_2=\begin{cases}\frac{s_{d+1}-s^*}{s_{d+1}-s_1} & \text{if } s^*>s_1\\ 1 & \text{if } s^*\leq s_1\end{cases} \tag{5}$$

If $\vec{s}=[s_0,s_1,\ldots,s_{d+1}]$ is a knot vector which is a subsequence of another knot vector $\tilde{\vec{s}}$ of length $n$, then $B[\vec{s}](s)$ can be written as a linear combination of $n-d-1$ B-spline basis functions defined over substrings of length $d+2$ in $\tilde{\vec{s}}$. The coefficients can be computed by repeating the above equation.

### 4.2. Degree elevation for a blending function

For a degree $p$ B-spline basis function $B^p[\vec{s}](s)$ defined on knot vector $\vec{s}$, where

$$\vec{s}=[s_0,s_1,\ldots,s_{p+1}]=[\underbrace{s_0,\ldots,s_0}_{\omega_0},\underbrace{s_{z_0+1},\ldots,s_{z_0+1}}_{\omega_1},\ldots,\underbrace{s_{z_{m-1}+1},\ldots,s_{z_{m-1}+1}}_{\omega_m}],$$

$z_r=\sum_{j=0}^{r}\omega_j$, then, $B^p[\vec{s}](s)$ can be written as a linear combination of $(m+1)$ degree $(p+1)$ B-spline basis functions,

$$B^p[\vec{s}](s)=\sum_{i=0}^{m}\alpha_iB^{p+1}[\vec{s_i}](s), \tag{6}$$

here, $\vec{s_i}$ is a substring of length $d+3$ in

$$\vec{s^2}=[s_0^2,s_1^2,\ldots,s_{p+m+2}^2]=[\underbrace{s_0,\ldots,s_0}_{\omega_0+1},\underbrace{s_{z_0+1},\ldots,s_{z_0+1}}_{\omega_1+1},\ldots,\underbrace{s_{z_{m-1}+1},\ldots,s_{z_{m-1}+1}}_{\omega_m+1}].$$

We will compute each coefficient $\alpha_i$ in (6) by using the fast degree elevation algorithm described in Huang et al. (2005), which includes the following main steps:

1. Compute $P^j_{j-(p+1-\omega_0)}(p+1-\omega_0\leq j\leq p)$ and $P^j_{z_r+j-(p+1-\omega_0)}(p+1-\omega_r\leq j\leq p)$ according to the following equation.

$$P_i^j=\begin{cases}1 & \text{if } i=0 \text{ and } j=0\\ \frac{P_{i+1}^{j-1}-P_i^{j-1}}{s_{i+p+1-j}-s_i} & \text{if } j>0, s_{i+p+1}>s_i\\ 0 & \text{if } j>0, s_{i+p+1}=s_i\end{cases} \tag{7}$$

2. Compute $\alpha^j_{j-(p+1-\omega_0)}$, $p+1-\omega_0\leq j\leq p$, $\alpha^j_{z_r+r+j-(p+1-\omega_0)}$, $p+1-\omega_r\leq j\leq p$ via the following equations:

$$\alpha^j_{j-(p+1-\omega_0)}=(\frac{p+1-j}{p+1})P^j_{j-(p+1-\omega_0)},\ p+1-\omega_0\leq j\leq p$$
$$\alpha^j_{z_r+r+j-(p+1-\omega_0)}=(\frac{p+1-j}{p+1})P^j_{z_r+j-(p+1-\omega_0)},\ p+1-\omega_r\leq j\leq p$$
$$\alpha^p_{z_r+r-(p-\omega_0)}=\alpha^p_{z_r+r-(p+1-\omega_0)}.$$

3. Compute $\alpha_i=\alpha_i^0$ via the following equation:

$$\alpha_i^j=\begin{cases}\frac{\omega_0}{p+1}\prod_{l=1}^{p+1-\omega_0}(s^2_{p+2-l}-s_0^2) & \text{if } i=0 \text{ and } j=0\\ \alpha_{i-1}^j+(s^2_{i+p-j}-s^2_{i-1})\alpha_{i-1}^{j+1} & \text{if } j>0, s_{i+p+1}>s_i\end{cases} \tag{8}$$

In the same way, we can get $B^q[\vec{t}](t)$ written as a linear combination of $(q+1)$-degree B-spline basis functions:

$$B^q[\vec{t}](t)=\sum_{i=0}^{n}\beta_iB^{q+1}[\vec{t_i}](t) \tag{9}$$

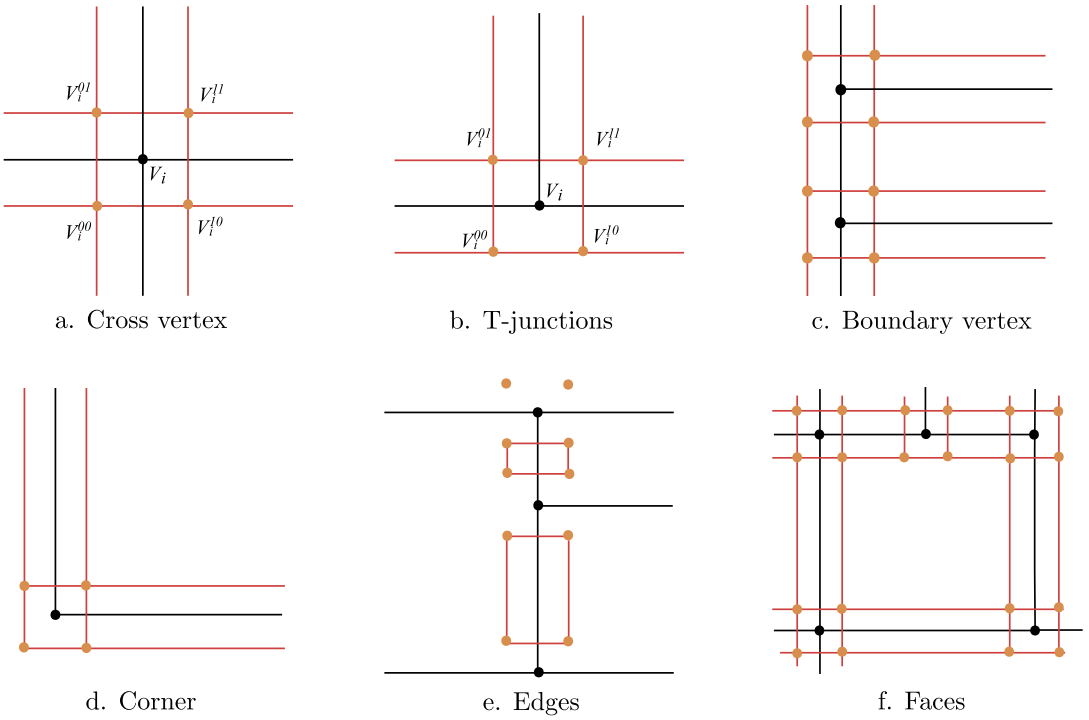



**Fig. 4.** Rules to create $D(\mathsf{T})$, $\mathsf{T}$ in black and $D(\mathsf{T})$ in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Thus, we have

$$\mathbf{T}(s,t) = B^p[\vec{s}](s)B^q[\vec{t}](t) \tag{10}$$

$$= \sum_{i=0}^{m}\sum_{j=0}^{n}\alpha_i\beta_j B^{p+1}[\vec{s_i}](s)B^{q+1}[\vec{t_j}](t) \tag{11}$$

$$= \sum_{i=0}^{N} C_i\hat{T}_i^*(s,t) \tag{12}$$

where $N=(m+1)\times(n+1)-1$, $\hat{T}_i^*(s,t)$ is a blending function of degree $(p+1,q+1)$ and $\hat{T}_i^*(s,t) = B^{p+1}[\vec{s_{j_1}}](s)B^{q+1}[\vec{t_{j_2}}](t)$, $j_1=[\frac{i}{n+1}]$, $j_2 = i \mod (n+1) = i - j_1(n+1)$.

### 4.3. Dual T-meshes

For a T-mesh $\mathsf{T}$ with associated knot intervals, if there are no zero knot intervals in the interior, then the dual T-mesh $D(\mathsf{T})$ is constructed as follows.

1. Vertex:
   - Split each interior vertex $V_i$ into four vertices $V_i^{jk}$, $j,k=1,2$. If the vertex is valence four, then the four new vertices are also valance four. If the vertex is a T-junction, then the four new vertices are two valence four vertices and two T-junctions of the same type. These two cases are illustrated in Figs. 4a and 4b, where the original vertices are shown as black points, and the new vertices in $D(\mathsf{T})$ are colored orange.
   - Split each boundary vertex into two valence four vertices and two boundary vertices, see Fig. 4c as an example. If the boundary vertex is on the left side of the T-mesh, then the new vertices $V_i^{00}$ and $V_i^{01}$ are boundary vertices. Split each corner into one corner, two boundary vertices and one valence four vertex, see Fig. 4d.
2. Faces: Each vertex, edge, and face in $\mathsf{T}$ maps to a face in $D(\mathsf{T})$, as follows:
   - A v-face in $D(\mathsf{T})$, corresponding to each vertex in $\mathsf{T}$, has corners $V_i^{jk}$, $j,k=1,2$, see Fig. 4a–d.
   - An e-face in $D(\mathsf{T})$, corresponding to each edge in $\mathsf{T}$, is created by connecting the vertices associated with the two end vertices of the edge.

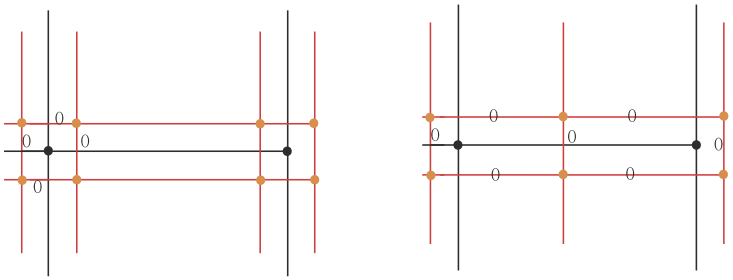


**Fig. 5.** Rules for $D(\mathsf{T})$ with multiple knots in $\mathsf{T}$, $\mathsf{T}$ in black and $D(\mathsf{T})$ in red. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

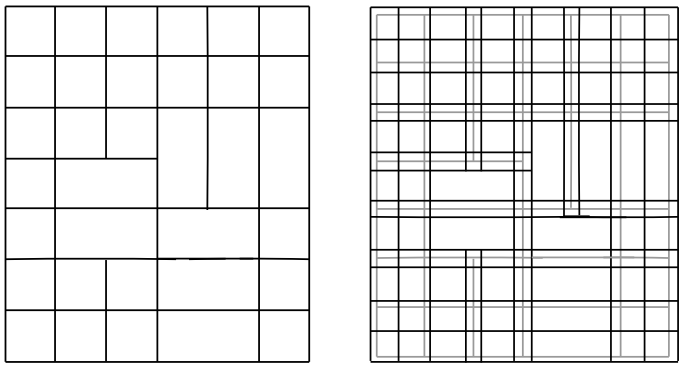



**Fig. 6.** A bi-degree $(2, 3)$ T-mesh $\mathsf{T}$ and its dual T-mesh $D(\mathsf{T})$, where is in black in the right figure while the original T-mesh is marked as gray.

- An f-face in $D(\mathsf{T})$, corresponding to each face in $\mathsf{T}$, is created by connecting the four valence-four vertices associated with the four vertices of the original face. Noticed that we also need to connect the T-junctions in the edge of an f-face.

3. Knot intervals:
   - For a v-face, the knot intervals for the four edges are all zeros;
   - For an e-face corresponding to a horizontal edge, the knot intervals for the two horizontal edges are the same as the original knot interval for the original edge and the knot intervals for the two vertical edges are zeros;
   - For an f-face, the knot intervals are replicated from the original edges of the face.

If zero knot intervals exist in the interior, the rules for creating the dual T-mesh are modified slightly. As illustrated in Fig. 5, a face that has zero knot intervals in one direction maps to an edge, and a face that has zero knot intervals in both directions maps to a vertex. Fig. 6a presents bi-degree $(2, 3)$ T-mesh $\mathsf{T}$ and its dual T-mesh $D(\mathsf{T})$.

### 4.4. Degree elevation algorithm

The degree elevation on a blending function $T_i(s,t)$ in $\mathbf{T}(s,t)$ with a knot vector $\vec{s_i} \times \vec{t_i}$ can be written as a linear combination of a set of bi-degree $(p+1, q+1)$ blending functions $\hat{T}^*_{i_j}(s,t)$,

$$T_i(s,t) = \sum_{j=1}^{n_i} c_i^{i_j} \hat{T}^*_{i_j}(s,t) \tag{13}$$

From (13),

$$\mathbf{T}(s,t) = \sum_{i=1}^{n_A} \sum_{j=1}^{n_i} c_i^{i_j} \mathbf{C}_i \hat{T}^*_{i_j}(s,t) \tag{14}$$

The degree elevation algorithm finds a T-mesh $\hat{\mathsf{T}}$ such that each function $\hat{T}^*_{i_j}(s,t)$ belongs to the bi-degree $(p+1, q+1)$ T-spline space $\mathbf{S}^{p+1,q+1}(\hat{\mathsf{T}})$. According to the discussion in section 4.2 and 4.3, the dual T-mesh is always a sub-mesh of $\hat{\mathsf{T}}$.

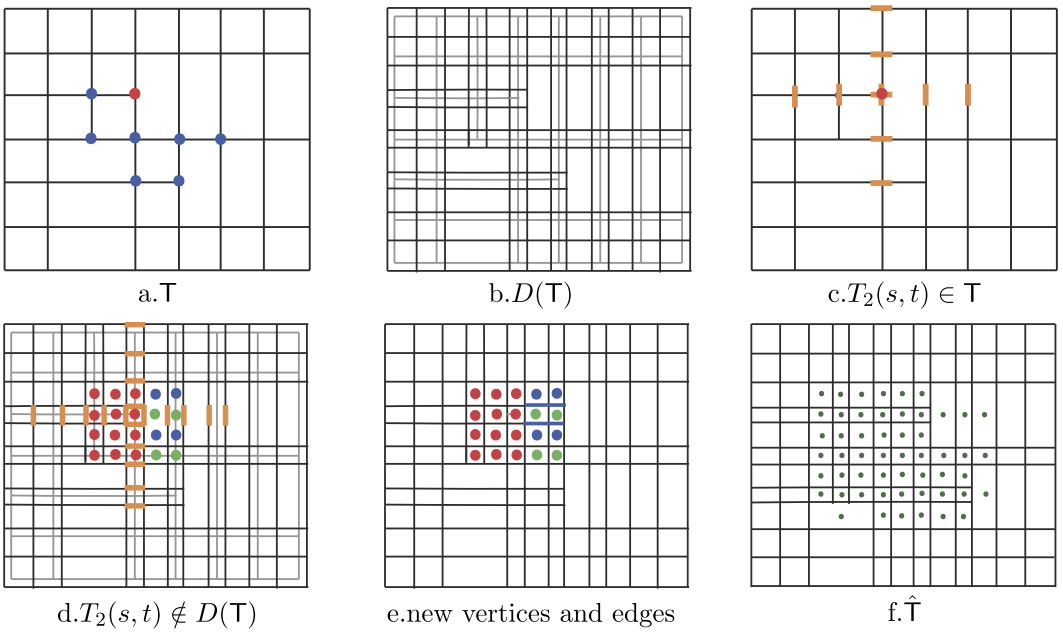



**Fig. 7.** An example for a bi-cubic T-spline. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

Based on these ideas, Algorithm 1 presents a recursive algorithm for degree elevation. An important key to understanding the algorithm is that the blending functions and anchors are tightly coupled, every anchor corresponds to a blending function, and the blending function's knot vectors are uniquely determined from the position of the anchor.

**Algorithm 1:** T-spline degree elevation algorithm.

**Input**: A bi-degree $(p, q)$ T-spline $\mathbf{T}(s,t) = \sum_{i=1}^{n_A} \mathbf{C}_i T_i(s,t)$ defined on a T-mesh $\mathsf{T}$;
**Output**: A bi-degree $(p+1, q+1)$ T-spline defined on a T-mesh $\hat{\mathsf{T}}$ with control points $\hat{\mathbf{C}}$ such that it is identical to $\mathbf{T}(s,t)$;

Degree elevate each blending function as in Section 4.2;
Add all the functions after degree elevation into a list $L$;
Create the dual T-mesh $D(\mathsf{T})$ as discussed in Section 4.3;
**repeat**
  Remove any blending function $B$ from list $L$;
  **if** $D(\mathsf{T})$ *has a knot* $k$ *(in either parameter direction) that lies in the support of* $B$ *but that is not contained in the knot vectors for* $B$ **then**
  Split $B$ into two blending functions by inserting a knot at $k$ (Section 4.1)
  Add those two functions into the list $L$.
  **if** $B$ *has a knot that is not dictated* **then**
  Add an appropriate vertex and edge associated with the knot into the T-mesh $D(\mathsf{T})$.
**until** $L$ *is empty*;

We illustrate the algorithm with an bi-cubic example in Fig. 7. The two global knot vectors are $\vec{s} = \{s_0, s_1, \ldots, s_7\}$ and $\vec{t} = \{t_0, t_1, \ldots, t_7\}$. There are eight anchors which are marked in blue. In order to degree elevate the T-spline, we first degree elevate each blending function. For example, for the blending function $T_2(s,t) = B[\vec{s_2}](s)B[\vec{t_2}](t)$ at $(s_3, t_4)$, where $\vec{s_2} = [s_1, s_2, s_3, s_4, s_5]$ and $\vec{t_2} = [t_2, t_3, t_4, t_5, t_6]$, it can be written into a linear combination of 20 bi-quartic basis functions $\hat{T}_j^*(s,t)$, where the knot vector is a subsequence of knot vectors from the degree elevation of the local knot vector $\vec{s_2}$ and $\vec{t_2}$. However, only 12 basis functions (marked in red in Fig. 7d) are coupled with the dual T-mesh $D(\mathsf{T})$ without the violations. There are 4 basis functions which correspond anchors are not in $D(\mathsf{T})$ (marked in blue in Fig. 7d) and the other 4 basis functions have knots that are not in the mesh $D(\mathsf{T})$ (marked in green). So we need to insert some edges and vertices (blue edges in Fig. 7e) into $D(\mathsf{T})$. The process is ended with the T-mesh in Fig. 7e.

This algorithm is always guaranteed to terminate because the only blending function refinements and edges (vertices) insertions must involve knot values that initially exist in the T-mesh. In the worst case, the algorithm would extend all partial rows of T-junctions to cross the entire surface. After creating the T-mesh $\hat{\mathsf{T}}$, it is obvious that every blending function in (14) can be represented as the linear combination of some blending functions $\hat{T}_i(s,t)$ in the resulting T-mesh $\hat{\mathsf{T}}$:

$$\hat{T}_{i_j}^*(s,t) = \sum_{k=1}^{\hat{n}_A} \alpha_k \hat{T}_k(s,t). \tag{15}$$

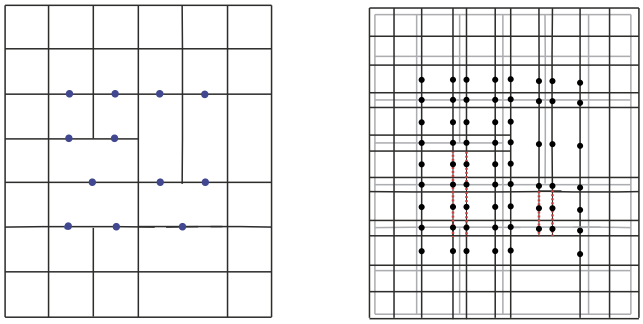


**Fig. 8.** Degree elevation of a bi-degree $(2,3)$ T-spline defined on the left T-mesh, which is the bi-degree $(3,4)$ T-spline defined on the right T-mesh (the anchors are marked as circle and black). The red edges are added from the dual T-mesh. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

Thus,

$$\sum_{i=1}^{n_A} \mathbf{C}_i T_i(s,t) = \sum_{i=1}^{n_A}\sum_{j=1}^{n_i}\sum_{k=1}^{\hat{n}_A} \mathbf{C}_i \alpha_k c_i^{i_j} \hat{T}_k(s,t) = \sum_{k=1}^{\hat{n}_A} \hat{\mathbf{C}}_k \hat{T}_k(s,t) \tag{16}$$

where $\hat{\mathbf{C}}_k = \sum_{i=1}^{n_A}\sum_{j=1}^{n_i} c_i^{i_j}\alpha_k \mathbf{C}_i$.

Fig. 8 shows another bi-degree $(2,3)$ T-spline degree elevation example. The resulting T-mesh is illustrated on the right. We can see that we also need to insert some additional vertices and edges (red edges) into the dual T-mesh.

## 5. Analysis suitable degree elevation

Because AS T-splines possess many desirable good properties for geometric modeling and iso-geometric analysis, in this section, we develop two algorithms such that the T-spline after degree elevation is analysis-suitable.

### 5.1. Theorem foundation

The basic theoretical foundation for the two degree elevation algorithms in the next two sections is based on the following two lemmas. And we will use them to construct a T-mesh for which T-spline space is a super space of the original T-spline space.

**Lemma 5.1.** *If* $\mathsf{T}$ *is a bi-degree* $(p,q)$ *analysis-suitable T-mesh, then* $D(\mathsf{T})$ *is a bi-degree* $(p+1,q+1)$ *analysis-suitable T-mesh.*

**Proof.** In $\mathsf{T}$, each index $\delta_i$ corresponds two indices $D^1(\delta_i)$ and $D^2(\delta_i)$ in $D(\mathsf{T})$. It is obvious that if $\delta_i < \delta_j$, then $D^1(\delta_i), D^2(\delta_i) < D^1(\delta_j), D^2(\delta_j)$. For each T-junction extensions $[\underline{\delta_i}, \overline{\delta_i}] \times \{\tau_i\}$ in $\mathsf{T}$, it is also corresponding two T-junction extensions, denoted as $[\underline{\delta_i^*}, \overline{\delta_i^*}] \times \{D^1(\tau_i)\}$ and $[\underline{\delta_i^*}, \overline{\delta_i^*}] \times \{D^2(\tau_i)\}$. It is also obvious that $[\underline{\delta_i^*}, \overline{\delta_i^*}] \subseteq [D^1(\underline{\delta_i}), D^2(\overline{\delta_i})]$. Because $\mathsf{T}$ is analysis-suitable, so for any two T-junction extensions, $[\underline{\delta_i}, \overline{\delta_i}] \times \{\tau_i\}$ and $\{\delta_j\} \times [\underline{\tau_j}, \overline{\tau_j}]$, don't intersect. Each T-junction in $\mathsf{T}$ corresponds to two new T-junctions of the same type in $D(\mathsf{T})$. Based on the above observation, the extensions of the four T-junctions in $D(\mathsf{T})$ don't intersect either. Thus, $D(\mathsf{T})$ is also analysis-suitable. □

**Lemma 5.2.** *If* $\mathsf{T}^1 \subseteq \mathsf{T}^2$, $\mathsf{T}^1_{elem} \subseteq \mathsf{T}^2_{elem}$ *and* $\mathsf{T}^2$ *is analysis-suitable, then* $\mathbf{S}(\mathsf{T}^1) \subseteq \mathbf{S}(\mathsf{T}^2)$.

**Proof.** For the simplicity of the paper, we omit the proof here. Because you can reach this lemma from Li and Scott (2014) for the bi-cubic case, which can be generalized to arbitrary degree AS T-splines very easily. □

### 5.2. AS T-mesh conversion algorithm

In this section, we give the AS T-mesh conversion algorithm, i.e., for a T-mesh $\mathsf{T}$, we find an AS T-mesh $\mathsf{T}^{as}$ with as less as possible vertices such that $\mathsf{T}_{elem} \subseteq \mathsf{T}^{as}_{elem}$. Finding the minimal number is an NP-hard problem. Thus for efficiency, our algorithm uses the following greedy strategy that only provides an approximate minimum.

Similar as Scott et al. (2012), for each T-junction $\rho_i$ at $(\delta_i, \tau_i)$ in the mesh $\mathsf{T}_{elem}$, if it is $\dashv$, we define its extension $e_i$ as an edge-line segment $[\underline{i}, \delta_i] \times \{\tau_i\}$ such that the edges $[\underline{i}, \delta_i) \times \{\tau_i\}$ have $(p+1)$ intersections with the T-mesh $\mathsf{T}_{elem}$. If it is $\vdash$, we define its extension $e_i$ as edge-line segment $[\delta_i, \overline{i}] \times \{\tau_i\}$ such that the edges $(\delta_i, \overline{i}] \times \{\tau_i\}$ have $(p+1)$ intersections with $\mathsf{T}_{elem}$. The cases for T-junction of $\top$ and $\perp$ can be defined similarly.

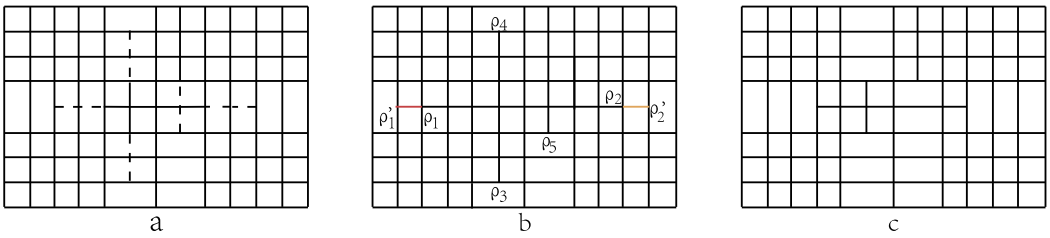



**Fig. 9.** An example for AS T-mesh conversion algorithm. (For interpretation of the colors in this figure, the reader is referred to the web version of this article.)

1. Denote all T-junctions in $\mathsf{T}_{elem}$ by $N_T = \{\rho_1, \rho_2, \ldots, \rho_{n_T}\}$ and their extensions by $E_T = \{e_1, e_2, \ldots, e_{n_T}\}$.
2. If all the edge-line segments in $E_T$ of different directions have no intersections, then go to step 4;
3. Else, let $n_{inter}$ be the number of intersection. For each T-junction $\rho_i$, extend one bay to create a new T-mesh $\mathsf{T}^i_{elem}$, denote the number of intersections for T-junction extensions by $n^i_{inter}$. For all $n^i_{inter}$, find the index $k$ such that $n^k_{inter}$ is minimal. If there are more than one such index, choose the first one. Now, we use $\mathsf{T}^k_{elem}$ to replace $\mathsf{T}_{elem}$ and repeat Step 2 to 3.
4. Suppose the final T-mesh is $\mathsf{T}^{as}_{elem}$, then, decrease the extensions from $\mathsf{T}^{as}_{elem}$ to get $\mathsf{T}^{as}$.

Fig. 9 illustrates a bi-cubic T-spline example. Fig. 9a shows a bi-cubic T-mesh $\mathsf{T}$ (solid edges) and its elemental mesh $\mathsf{T}_{elem}$ (with dotted edges). There are five T-junctions $N_T = \{\rho_1, \rho_2, \rho_3, \rho_4, \rho_5\}$ in $\mathsf{T}_{elem}$. In Fig. 9b, the edge-line segments of $\rho_1$ and $\rho_3$, $\rho_1$ and $\rho_4$, $\rho_2$ and $\rho_5$ insert. First we extend $\rho_1$ one bay (red edge), there are only one intersection which is minimal. So, in the first iterator, we extend $\rho_1$. And in the second iterator, we extend $\rho_2$ (yellow edge) and no extensions insert. Then we get the elemental mesh of an AS T-mesh, $\mathsf{T}^{as}_{elem}$ as shown in Fig. 9b. The final AS T-mesh $\mathsf{T}^{as}$ is shown in Fig. 9c.

### 5.3. Analysis-suitable degree elevation

Algorithm 2 presents an algorithm for analysis-suitable degree elevation.

**Algorithm 2:** Analysis-suitable T-spline degree elevation algorithm one.

**Input**: A bi-degree $(p, q)$ T-spline $\mathbf{T}(s,t) = \sum_{i=1}^{n_A} \mathbf{C}_i T_i(s,t)$ defined on a T-mesh $\mathsf{T}$;
**Output**: A bi-degree $(p+1, q+1)$ T-spline defined on an analysis-suitable T-mesh $\mathsf{T}^{as1}$ with some control points such that it is identical to $\mathbf{T}(s,t)$;

Create the elemental T-mesh $\mathsf{T}_{elem}$;
1 Create the dual T-mesh $D(\mathsf{T}_{elem})$ as discussed in Section 4.3;
2 Create an analysis-suitable T-mesh $\mathsf{T}^{as1}$ such that $D(\mathsf{T}_{elem})$ is a sub-mesh of $\mathsf{T}^{as1}$'s elemental T-mesh using the algorithm in Section 5.2;
Compute the new control points by basis functions refinement in (4) and degree elevation for one blending function in (5) for $\mathsf{T}^{as1}$.

In fact, in the Algorithm 2, we can achieve the degree elevation if we exchange the order of Line 1 and 2. Then, we get the following degree elevation Algorithm 3. According to the discussion in Section 5.2, for both analysis-suitable T-spline degree elevation algorithms, we can guarantee that the resulting T-splines are analysis-suitable and the resulting spline spaces are super space of the original T-spline space (Theorem 5.3).

**Algorithm 3:** Analysis-suitable T-spline degree elevation algorithm two.

**Input**: A bi-degree $(p, q)$ T-spline $\mathbf{T}(s,t) = \sum_{i=1}^{n_A} \mathbf{C}_i T_i(s,t)$ defined on a T-mesh $\mathsf{T}$;
**Output**: A bi-degree $(p+1, q+1)$ T-spline defined on an analysis-suitable T-mesh $\mathsf{T}^{as2}$ with some control points such that it is identical to $\mathbf{T}(s,t)$;

Create the elemental T-mesh $\mathsf{T}_{elem}$;
1 Create an analysis-suitable T-mesh $\mathsf{T}^{as}$ such that $\mathsf{T}_{elem}$ is a sub-mesh of its elemental T-mesh using the algorithm in Section 5.2;
2 Create the dual T-mesh $D(\mathsf{T}^{as}_{elem})$ as discussed in Section 4.3;
3 Decrease extensions of $D(\mathsf{T}^{as}_{elem})$ to get $\mathsf{T}^{as2}$.
Compute the new control points by basis functions refinement in (4) and degree elevation for one blending function in (5) for $\mathsf{T}^{as2}$.

**Theorem 5.3.** *In Algorithm 2 and Algorithm 3, the resulting T-splines are analysis-suitable and the resulting spline spaces are super spaces of the original T-spline space, i.e.* $\mathbf{S}(\mathsf{T}) \subseteq \mathbf{S}(\mathsf{T}^{as1})$, $\mathbf{S}(\mathsf{T}) \subseteq \mathbf{S}(\mathsf{T}^{as2})$.

**Proof.** $\mathsf{T}_{elem} \subseteq D(\mathsf{T}_{elem})$. According to step 2 in Algorithm 2, $D(\mathsf{T}_{elem})$ is a sub-mesh of $\mathsf{T}^{as1}$'s elemental T-mesh, $D(\mathsf{T}_{elem}) \subseteq \mathsf{T}^{as1}_{elem}$. Then $\mathsf{T}_{elem} \subseteq \mathsf{T}^{as1}_{elem}$. According to Lemma 5.2, $\mathbf{S}(\mathsf{T}) \subseteq \mathbf{S}(\mathsf{T}^{as1})$.

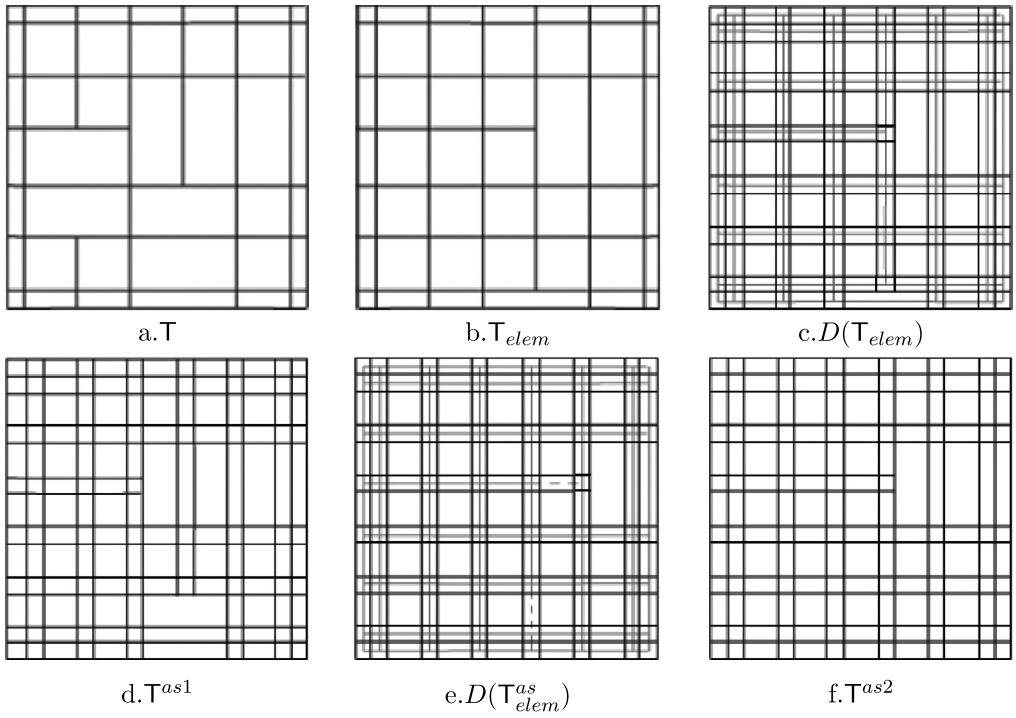



**Fig. 10.** The two AS algorithms for a (2,3) T-spline.

According to Lemma 5.1, $T^{as2}$ is a bi-degree $(p+1, q+1)$ analysis-suitable T-mesh. According to step 2 in Algorithm 3, $T_{elem}$ is a sub-mesh of $T^{as}$'s elemental T-mesh, $T_{elem} \subseteq T^{as}_{elem}$. And $T^{as}_{elem} \subseteq D(T^{as}_{elem})$. So $T_{elem} \subseteq D(T^{as}_{elem})$. According to Lemma 5.2, $\mathbf{S}(T) \subseteq \mathbf{S}(T^{as2})$. □

### 5.4. Comparison

Although the two algorithms are very similar and they both create a super space (Theorem 5.3), they will produce different resulting T-splines for most general T-meshes. First, we show a very simple example and illustrate all the intermedial T-meshes created by the two algorithms in Fig. 10.

Fig. 12 and Fig. 13 show the other two examples that the second AS degree elevation algorithm produces less control points. In Fig. 12, the number of the control points in original T-mesh is 127, while the number of control points in the resulting mesh by the first AS degree elevation algorithm is 924, and that by the second algorithm is 913. In the Fig. 13, the number of the original control points is 167, and the numbers of the control points by the two AS degree elevation algorithm are 996 and 866 respectively. In our experiences, the second algorithm create the T-splines with less control points on more examples with similar efficiency. However, if the initial T-spline is an analysis-suitable T-spline, then the resulting T-splines created by both algorithms will be exactly the same, see Remark 5.4 for more details.

**Remark 5.4.** If the original T-mesh T is analysis-suitable, then the resulting T-splines created by the two AS degree elevation algorithms are the same. Actually, the resulting T-mesh can be explicitly defined. Because $D(T_{elem})$ is the elemental T-mesh of an analysis-suitable T-mesh, so the $T^{as1}$'s elemental T-mesh in Line 2 of Algorithm 2 is $D(T_{elem})$. And the T-mesh $T^{as}$ in Line 1 of Algorithm 3 is T. Thus the two algorithms give exactly the same resulting T-meshes and the T-mesh can be directly given because we don't need the greedy optimized algorithm to convert some T-meshes into analysis-suitable. Fig. 11 illustrates an example for the process.

## 6. Results and conclusion

We conclude by presenting some numerical experimentations about these three degree elevation algorithms for T-splines. The key issue to compare the algorithms is the number of resulting anchors. Thus all the examples are shown with the index T-meshes in this section.

The first example is a T-spline defined on a T-mesh refined from a tensor-product T-mesh along the diagonal faces. Using the general T-spline degree elevation (Fig. 12b), just similar as the behavior of local refinement algorithm in Sederberg et al. (2004), the resulting T-spline is almost a tensor-product B-spline. However, for both analysis-suitable T-spline degree elevation algorithms, the resulting T-splines are very reasonable (Fig. 12c and d).

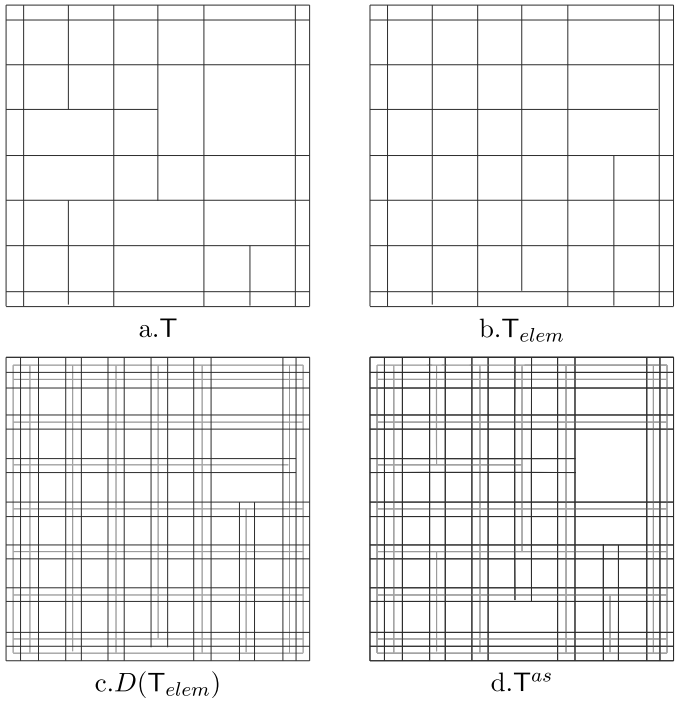



**Fig. 11.** Degree elevation for an AS T-spline.

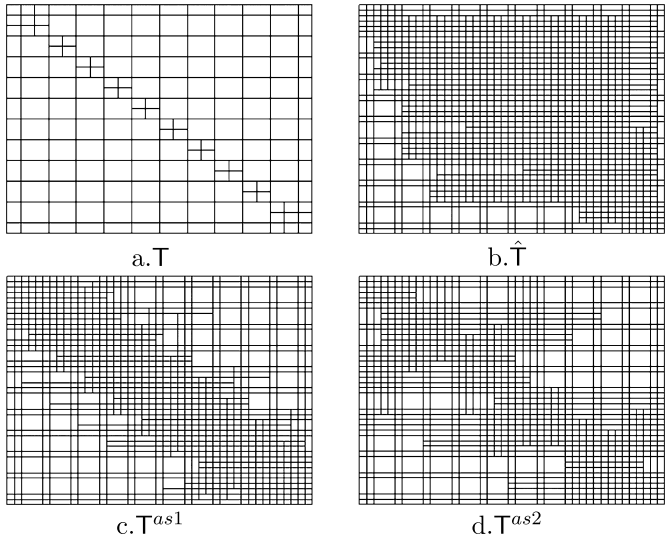



**Fig. 12.** The three degree elevation algorithms on a bi-cubic T-spline defined on the T-mesh in a.

The second example is a T-spline defined on a T-mesh which is refined a tensor-product mesh with random faces. We can see the similar behavior through this example. The resulting T-spline with general T-spline degree elevation algorithm (Fig. 13b) is also almost a tensor-product B-spline, while the analysis-suitable T-spline degree elevation algorithms preserve the resulting T-splines in the localized regions (Fig. 13c and d).

In this paper, we present an degree elevation algorithm for arbitrary bi-degree T-splines. We also develop two analysis-suitable T-splines degree elevation algorithms. Similar as the behavior of local refinement algorithm, the degree elevation for general T-splines also suffers the problem of global propagation. But if we restrict the resulting T-spline to be analysis-suitable, then the propagation will be kept in a much more localized region. A very interesting topic of future work is to develop the iso-geometric analysis application with the current degree elevation algorithm and local refinement algorithm. Another interesting topic is to define multi-degree T-splines such that we can develop the associated local degree elevation

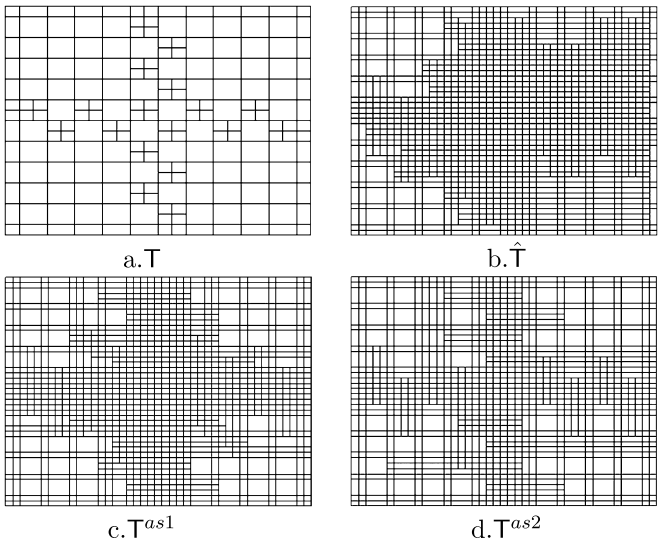



**Fig. 13.** The three degree elevation algorithms on a bi-cubic T-spline defined on the T-mesh in a.

algorithm or local k-refinement algorithm by combining the existing local refinement algorithm. This is also left as future work.

## Acknowledgements

The authors are supported by the NSFC (No. 11031007, No. 60903148, No. 11371341), a NKBRPC (2011CB302400), the Chinese Universities Scientific Fund, SRF for ROCS SE, and the Youth Innovation Promotion Association CAS.

## References

Bazilevs, Y., Calo, V.M., Cottrell, J.A., Evans, J.A., Hughes, T.J.R., Lipton, S., Scott, M.A., Sederberg, T.W., 2010. Isogeometric analysis using T-splines. Comput. Methods Appl. Mech. Eng. 199 (5–8), 229–263.

Benson, D.J., Bazilevs, Y., De Luycker, E., Hsu, M.C., Scott, M.A., Hughes, T.J.R., Belytschko, T., 2010. A generalized finite element formulation for arbitrary basis functions: from isogeometric analysis to XFEM. Int. J. Numer. Methods Eng. 83, 765–785.

Borden, M.J., Verhoosel, C.V., Scott, M.A., Hughes, T.J.R., Landis, C.M., 2012. A phase-field description of dynamic brittle fracture. Comput. Methods Appl. Mech. Eng. 217–220, 77–95.

Buffa, A., Cho, D., Sangalli, G., 2010. Linear independence of the T-spline blending functions associated with some particular T-meshes. Comput. Methods Appl. Mech. Eng. 199 (23–24), 1437–1445.

Buffa, A., Cho, D., Kumar, M., 2012. Characterization of T-splines with reduced continuity order on T-meshes. Comput. Methods Appl. Mech. Eng. 201–204, 112–126.

Cottrell, J.A., Hughes, T.J.R., Reali, A., 2007. Studies of refinement and continuity in isogeometric structural analysis. Comput. Methods Appl. Mech. Eng. 196, 4160–4183.

Cottrell, J.A., Hughes, T.J.R., Bazilevs, Y., 2009. Isogeometric Analysis: Toward Integration of CAD and FEA. Wiley, Chichester.

da Veiga, L.B., Buffa, A., Rivas, J., Sangalli, G., 2011. Some estimates for h-p-k-refinement in isogeometric analysis. Numer. Math. 118 (2), 271–305.

Dimitri, R., Lorenzis, L.D., Scott, M.A., Wriggers, P., Taylor, R.L., Zavarise, G., 2014. Isogeometric large deformation frictionless contact using T-splines. Comput. Methods Appl. Mech. Eng. 269, 394–414.

Farin, G., 2002. NURBS Curves and Surfaces: From Projective Geometry to Practical Use, fourth edition. A.K. Peters, Ltd., Natick, MA.

Huang, Q.-X., Hu, S.-M., Martin, R.R., 2005. Fast degree elevation and knot insertion for B-spline curves. Comput. Aided Geom. Des. 22 (2), 183–197.

Hughes, T.J.R., Cottrell, J.A., Bazilevs, Y., 2005. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. Comput. Methods Appl. Mech. Eng. 194, 4135–4195.

Ipson, H., 2005. T-spline merging. Master's thesis. Brigham Young University.

Li, X., 2015. Some properties for analysis-suitable T-splines. J. Comput. Math. 33 (4), 427–440.

Li, X., Scott, M.A., 2014. Analysis-suitable T-splines: characterization, refineability and approximation. Math. Models Methods Appl. Sci. 24 (06), 1141–1164.

Li, X., Zheng, J., Sederberg, T.W., Hughes, T.J.R., Scott, M.A., 2012. On the linear independence of T-splines blending functions. Comput. Aided Geom. Des. 29, 63–76.

Liu, L., Zhang, Y., Hughes, T.J.R., Scott, M.A., Sederberg, T.W., 2014. Volumetric T-spline construction using boolean operations. Eng. Comput. 30, 425–439.

Morgenstern, P., Peterseim, D., 2015. Analysis-suitable adaptive T-mesh refinement with linear complexity. Comput. Aided Geom. Des. 34, 50–66.

Schillinger, D., Dede, L., Scott, M.A., Evans, J.A., Borden, M.J., Rank, E., Hughes, T.J.R., 2014. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline cad surfaces. Comput. Methods Appl. Mech. Eng. 249–252, 116–150.

Scott, M.A., Li, X., Sederberg, T.W., Hughes, T.J.R., 2012. Local refinement of analysis-suitable T-splines. Comput. Methods Appl. Mech. Eng. 213–216, 206–222.

Scott, M.A., Simpson, R.N., Evans, J.A., Lipton, S., Bordas, S.P.A., Hughes, T.J.R., Sederberg, T.W., 2013. Isogeometric boundary element analysis using unstructured T-splines. Comput. Methods Appl. Mech. Eng. 254, 197–221.

Sederberg, T.W., Zheng, J., Bakenov, A., Nasri, A., 2003. T-splines and T-NURCCSs. ACM Trans. Graph. 22 (3), 477–484.

Sederberg, T.W., Cardon, D.L., Finnigan, G.T., North, N.S., Zheng, J., Lyche, T., 2004. T-spline simplification and local refinement. ACM Trans. Graph. 23 (3), 276–283.

Sederberg, T.W., Finnigan, G.T., Li, X., Lin, H., Ipson, H., 2008. Watertight trimmed NURBS. ACM Trans. Graph. 27 (3), 79.
Veiga, L.B., Buffa, A., Sangalli, D.C.G., 2011. Isogeometric analysis using T-splines on two-patch geometries. Comput. Methods Appl. Mech. Eng. 200, 1787–1803.
Veiga, L.B., Buffa, A., Sangalli, D.C.G., 2012. Analysis-suitable T-splines are dual-compatible. Comput. Methods Appl. Mech. Eng. 249–252, 42–51.
Veiga, L.B., Buffa, A., Sangalli, G., Vazquez, R., 2013. Analysis-suitable T-splines of arbitrary degree: definition and properties. Math. Models Methods Appl. Sci. 23, 1979–2003.
Wang, G., Deng, C., 2007. On the degree elevation of B-spline curves and corner cutting. Comput. Aided Geom. Des. 24 (2), 90–98.
Zhang, J., Li, X., 2015. On the linear independence and partition of unity of arbitrary degree analysis-suitable T-splines. Commun. Math. Stat. 3 (3), 353–364.