

# Aorta modeling with the element-based zero-stress state and isogeometric discretization

Kenji Takizawa<sup>1</sup> · Tayfun E. Tezduyar<sup>2</sup> · Takafumi Sasaki<sup>1</sup>

Received: 21 August 2016 / Accepted: 12 October 2016 / Published online: 5 November 2016  
© Springer-Verlag Berlin Heidelberg 2016

**Abstract** Patient-specific arterial fluid–structure interaction computations, including aorta computations, require an estimation of the zero-stress state (ZSS), because the image-based arterial geometries do not come from a ZSS. We have earlier introduced a method for estimation of the element-based ZSS (EBZSS) in the context of finite element discretization of the arterial wall. The method has three main components. 1. An iterative method, which starts with a calculated initial guess, is used for computing the EBZSS such that when a given pressure load is applied, the image-based target shape is matched. 2. A method for straight-tube segments is used for computing the EBZSS so that we match the given diameter and longitudinal stretch in the target configuration and the “opening angle.” 3. An element-based mapping between the artery and straight-tube is extracted from the mapping between the artery and straight-tube segments. This provides the mapping from the arterial configuration to the straight-tube configuration, and from the estimated EBZSS of the straight-tube configuration back to the arterial configuration, to be used as the initial guess for the iterative method that matches the image-based target shape. Here we present the version of the EBZSS estimation method with isogeometric wall discretization. With isogeometric discretization,

we can obtain the element-based mapping directly, instead of extracting it from the mapping between the artery and straight-tube segments. That is because all we need for the element-based mapping, including the curvatures, can be obtained within an element. With NURBS basis functions, we may be able to achieve a similar level of accuracy as with the linear basis functions, but using larger-size and much fewer elements. Higher-order NURBS basis functions allow representation of more complex shapes within an element. To show how the new EBZSS estimation method performs, we first present 2D test computations with straight-tube configurations. Then we show how the method can be used in a 3D computation where the target geometry is coming from medical image of a human aorta.

**Keywords** Patient-specific arterial FSI · Image-based geometry · Aorta · Zero-stress state · Estimated element-based zero-stress state · Isogeometric wall discretization

## 1 Introduction

Computational cardiovascular fluid mechanics and fluid–structure interaction (FSI) have seen major advances in the last 15 years. Some of the earliest patient-specific arterial FSI computations [1–4] were with a space–time (ST) FSI method, specifically the Deforming-Spatial-Domain/Stabilized ST (DSD/SST) method [5,6].

The majority of the cardiovascular fluid mechanics and FSI computations in the last 15 years have been with non-ST methods, with the Arbitrary Lagrangian–Eulerian (ALE) method [7] having the largest share in the computations reported (see, for example, [8–29]). A large number of computations with the ST methods were also reported in the last 15 years, including computations in the first 10 years for FSI

✉ Tayfun E. Tezduyar  
tezduyar@tafsm.org

Kenji Takizawa  
Kenji.Takizawa@tafsm.org

<sup>1</sup> Department of Modern Mechanical Engineering,  
Waseda University, 1-6-1 Nishi-Waseda, Shinjuku-ku,  
Tokyo 169-8050, Japan

<sup>2</sup> Mechanical Engineering, Rice University – MS 321, 6100  
Main Street, Houston, TX 77005, USA

of abdominal aorta [30], FSI of carotid artery [30], and FSI of cerebral aneurysms [31–37]. The work with ST methods reported in the last 5 years focused on even more challenging aspects of cardiovascular fluid mechanics and FSI, including comparative studies of cerebral aneurysms [38], stent treatment of cerebral aneurysms [39–42], heart valve flow computation [43–45], FSI analysis of thoracic aorta [46], and coronary arterial dynamics [47]. Advances in core methods for moving boundaries and interfaces (MBI) and FSI (see, for example, [20,21,48–52] and references therein) and development of special methods targeting cardiovascular MBI and FSI (see, for example, [20,37,41,42,53] and references therein) helped address a large number of computational challenges. In this article, we focus on a challenge related to how we use the image-based arterial geometry in patient-specific arterial FSI computations.

In patient-specific arterial FSI computations the image-based arterial geometry does not come from a zero-stress state (ZSS). Special methods targeting cardiovascular MBI and FSI include those designed to take that into account. The attempt to find a ZSS for the artery was first made in a 2007 conference paper [54], where the concept of estimated zero-pressure (EZP) arterial geometry was introduced. The method introduced in [54] for calculating an EZP geometry was also included in a 2008 journal paper on ST arterial FSI methods [31] as “a rudimentary technique” for addressing the issue. It was pointed out in [31,54] that quite often, the image-based geometries were used as arterial geometries corresponding to zero blood pressure, and that it would be more realistic to use the image-based geometry as the arterial geometry corresponding to the time-averaged value of the blood pressure. Given that arterial geometry at the time-averaged pressure value, an estimated arterial geometry corresponding to zero blood pressure needed to be constructed. Special methods developed to address the issue include the newer EZP versions [20,33,36,37,53] and the prestress technique introduced in [16], which was further refined in [18] and presented also in [20,53].

A method for estimation of the element-based ZSS (EBZSS) was introduced in [55] in the context of finite element discretization of the arterial wall. The method has three main components. 1. An iterative method, which starts with a calculated initial guess, is used for computing the EBZSS such that when a given pressure load is applied, the image-based target shape is matched. 2. A method for straight-tube segments is used for computing the EBZSS so that we match the given diameter and longitudinal stretch in the target configuration and the “opening angle.” 3. An element-based mapping between the artery and straight-tube is extracted from the mapping between the artery and straight-tube segments. This provides the mapping from the arterial configuration to the straight-tube configuration, and from the estimated EBZSS of the straight-tube configuration

back to the arterial configuration, to be used as the initial guess for the iterative method that matches the image-based target shape. The method was used successfully in [55] in test computations based on straight-tube configurations with single and three layers, and a curved-tube configuration with single layer. The method was used successfully also in [47] in coronary arterial dynamics computations with medical-image-based time-dependent anatomical models.

In a recent article [56], we have introduced the version of the EBZSS estimation method with isogeometric wall discretization. With isogeometric discretization, we can obtain the element-based mapping directly, instead of extracting it from the mapping between the artery and straight-tube segments. That is because all we need for the element-based mapping, including the curvatures, can be obtained within an element. With NURBS basis functions, we may be able to achieve a similar level of accuracy as with the linear basis functions, but using larger-size and much fewer elements. Higher-order NURBS basis functions allow representation of more complex shapes within an element. The 2D test computations with straight-tube configurations presented in [56] were intended to explain how the new EBZSS estimation method works and to demonstrate how it performs. In this expanded, journal version of that article, we also show how the method can be used in a 3D computation where the target geometry is coming from medical image of a human aorta.

In Sect. 2, we describe, in the context of isogeometric discretization, the Element-Based Total Lagrangian (EBTL) method, including the EBZSS concept. Section 3, extracted from [55], is an overview of the analytical relationship between the ZS and reference states of straight-tube segments, and here we call that relationship “straight-tube ZSS template”. The 2D test computations are presented in Sect. 4, and the 3D computation in Sect. 5. The concluding remarks are given in Sect. 6.

## 2 Element-Based Total Lagrangian (EBTL) method

In this section we provide an overview of the EBTL method [55], including the EBZSS concept, and describe the version of the method with NURBS wall discretization.

Let  $\Omega_0 \in \mathbb{R}^3$  be the material domain of a structure in the ZSS, and let  $\Gamma_0$  be its boundary. Let  $\Omega_t \in \mathbb{R}^3$ ,  $t \in (0, T)$ , be the material domain of the structure in the deformed state, and let  $\Gamma_t$  be its boundary. The structural mechanics equations based on the total Lagrangian formulation can be written as

$$\int_{\Omega_0} \mathbf{w} \cdot \rho_0 \frac{d^2 \mathbf{y}}{dt^2} d\Omega + \int_{\Omega_0} \delta \mathbf{E} : \mathbf{S} d\Omega - \int_{\Omega_0} \mathbf{w} \cdot \rho_0 \mathbf{f} d\Omega = \int_{(\Gamma_t)_h} \mathbf{w} \cdot \mathbf{h} d\Gamma. \quad (1)$$

Here,  $\mathbf{y}$  is the displacement,  $\mathbf{w}$  is the virtual displacement,  $\delta\mathbf{E}$  is the variation of the Green–Lagrange strain tensor,  $\mathbf{S}$  is the second Piola–Kirchhoff stress tensor,  $\rho_0$  is the mass density in the ZSS,  $\mathbf{f}$  is the body force per unit mass, and  $\mathbf{h}$  is the external traction vector applied on the subset  $(\Gamma_t)_h$  of the total boundary  $\Gamma_t$ .

### 2.1 EBZSS

In the EBTL method the ZSS is defined with a set of positions  $\mathbf{X}_0^e$  for each element  $e$ . Positions of nodes from different elements mapping to the same node in the mesh do not have to be the same. In the reference state,  $\mathbf{X}_{\text{REF}}$ , all elements are connected by nodes, and we measure the displacement  $\mathbf{y}$  from that connected state. The implementation of the method is simple. The deformation gradient tensor  $\mathbf{F}$  is evaluated for each element:

$$\mathbf{F}^e \equiv \frac{\partial \mathbf{x}}{\partial \mathbf{X}_0^e}, \tag{2}$$

$$= \frac{\partial (\mathbf{X}_{\text{REF}} + \mathbf{y})}{\partial \mathbf{X}_0^e}. \tag{3}$$

The deformation gradient tensors for different elements are on different states, but the terms in Eq. (1), including the second term, do not depend on the orientation. Therefore the rest of the process is the same as it is in the total Lagrangian formulation.

### 2.2 NURBS basis functions

In representation of the EBZSS with NURBS basis functions, we may be able to use larger and fewer elements compared to linear basis functions. Higher-order NURBS basis functions allow representation of more complex shapes within an element. Curvature representation requires at least quadratic NURBS, and to have a continuous curvature, cubic or higher-order NURBS is required.

In the case of 1D parametric space, a curve segment can be represented by NURBS as

$$\mathbf{z}(\xi) = \sum_{a=1}^{n_{\text{en}}} R_a(\xi) \mathbf{z}_a, \tag{4}$$

where  $n_{\text{en}}$  is the number of control points in the element,  $\mathbf{z}_a$  is the position of the control point (node)  $a$ ,

$$R_a(\xi) = \frac{N_a(\xi)w_a}{\sum_{b=1}^{n_{\text{en}}} N_b(\xi)w_b}, \tag{5}$$

$N_a(\xi)$  is the B-spline basis function for point  $a$ , and  $w_a$  is the NURBS weight for point  $a$ . An equivalent form can be

obtained by using the homogeneous coordinates [57], where  $\mathbf{z}_a$  is augmented as

$$\mathbf{z}_a^w = \begin{bmatrix} w_a \mathbf{z}_a \\ w_a \end{bmatrix}. \tag{6}$$

With that, we represent the curve segment as

$$\mathbf{z}^w(\xi) = \sum_{a=1}^{n_{\text{en}}} N_a(\xi) \mathbf{z}_a^w, \tag{7}$$

and

$$\mathbf{z}(\xi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \frac{\mathbf{z}^w(\xi)}{\sum_{a=1}^{n_{\text{en}}} N_a(\xi)w_a}. \tag{8}$$

The forms given by Eqs. (4) and (8) are equivalent.

As proposed in [57], we represent the B-spline basis functions with the Bernstein basis functions  $B_b(\xi)$ :

$$N_a(\xi) = \sum_{b=1}^{n_{\text{en}}} C_{ab} B_b(\xi), \tag{9}$$

where  $C_{ab}$  denotes the components of the Bézier extraction operator. See [57] for how to obtain the operator from the B-spline knots. With that,

$$\mathbf{z}^w(\xi) = \sum_{a=1}^{n_{\text{en}}} \sum_{b=1}^{n_{\text{en}}} C_{ab} B_b(\xi) \mathbf{z}_a^w. \tag{10}$$

From that, we can first operate with  $C_{ab}$ , and obtain the Bézier control positions as

$$\hat{\mathbf{z}}_b = \sum_{a=1}^{n_{\text{en}}} \mathbf{z}_a^w C_{ab}. \tag{11}$$

The extension to multi-dimensional parametric spaces is straightforward.

Given the control position  $\mathbf{z}_a$  for a point  $a$ , the corresponding homogeneous coordinates  $\mathbf{z}_a^w$ , its Bézier representation  $\hat{\mathbf{z}}_a$ , and the Bézier extraction operator components  $C_{ab}$ , we move to an array notation where

$$\mathbf{Z} = [\mathbf{z}_a], \tag{12}$$

$$\mathbf{Z}^w = [\mathbf{z}_a^w], \tag{13}$$

$$\hat{\mathbf{Z}} = [\hat{\mathbf{z}}_a], \tag{14}$$

$$\mathbf{C} = [C_{ab}]. \tag{15}$$

Then, Eq. (11) can be written as

$$\hat{\mathbf{Z}} = \mathbf{Z}^w \mathbf{C}. \tag{16}$$

With that, the transformation from the Bézier representation to NURBS representation becomes

$$\mathbf{Z}^w = \hat{\mathbf{Z}}\mathbf{C}^{-1}. \tag{17}$$

### 2.3 EBZSS representation with NURBS basis functions

When we are designing a ZSS, we have the corresponding reference state  $\mathbf{X}_{\text{REF}}$ . Therefore we have  $(\mathbf{X}_{\text{REF}})_a$ ,  $w_a$ , and the Bézier extraction operator corresponding to the element. We obtain the homogeneous coordinates  $(\mathbf{X}_{\text{REF}})_a^w$  and then convert that to  $(\hat{\mathbf{X}}_{\text{REF}})_a$ . From that and the Bernstein basis functions, we can design the EBZSS as  $(\hat{\mathbf{X}}_0^e)_a$ . For implementation convenience, we convert the control points to  $(\mathbf{X}_0^e)_a$  by using Eq. (17).

*Remark 1* Although in the ZSS we could have element-based  $w_a$  values, that would in general require using different basis functions between the ZS and reference states. Here we do not consider that option.

### 2.4 An iterative method

With the EBZSS, under a given load we would like to reach a configuration that matches the target shape, and we take  $\mathbf{X}_{\text{REF}}$  as the target state. Here we assume that we have a reasonably good initial guess for the EBZSS, and explain the iterative method used in calculating the EBZSS that results in the target state associated with the given load. In doing that, we use many pieces of the method described in [55] for linear elements.

In our iterative method, we estimate  $\mathbf{F}$  from the  $i$ th solution. We use the notation

$$\mathbf{F}(\mathbf{x}, \mathbf{X}) = \mathbf{R}(\mathbf{x}, \mathbf{X})\mathbf{U}(\mathbf{x}, \mathbf{X}), \tag{18}$$

which is the polar decomposition of  $\mathbf{F}$  into rotation  $\mathbf{R}$  and right stretch tensor  $\mathbf{U}$ . The arguments in the tensors represent the numerator and denominator in the partial derivatives. With that,  $\mathbf{F}$  at  $(i + 1)$ th iteration is expressed as

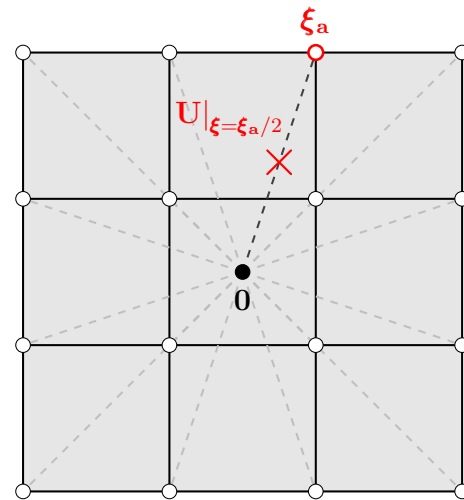
$$\mathbf{F}^{i+1} = \mathbf{R}(\mathbf{X}_{\text{REF}}, \mathbf{x}^i) \mathbf{F}^i \mathbf{R}((\mathbf{X}_0^e)^i, (\mathbf{X}_0^e)^{i+1}). \tag{19}$$

With the approximation

$$\mathbf{R}((\mathbf{X}_0^e)^i, (\mathbf{X}_0^e)^{i+1}) = \mathbf{I}, \tag{20}$$

we obtain

$$\mathbf{F}^{i+1} = \mathbf{R}(\mathbf{X}_{\text{REF}}, \mathbf{x}^i) \mathbf{F}^i, \tag{21}$$



**Fig. 1** The representative parametric position  $\xi_a$  assigned to the Bézier control point  $a$ , and the straight path from  $\mathbf{0}$  to  $\xi_a$ . When performing the integration with the midpoint rule, the evaluation point is  $\xi_a/2$

and from that we obtain

$$(\mathbf{F}^{i+1})^{-1} = (\mathbf{F}^i)^{-1} \mathbf{R}(\mathbf{x}^i, \mathbf{X}_{\text{REF}}), \tag{22}$$

$$= (\mathbf{U}^i)^{-1} \mathbf{R}((\mathbf{X}_0^e)^i, \mathbf{X}_{\text{REF}}). \tag{23}$$

In this article, in calculating  $(\mathbf{X}_0^e)^{i+1}$ , instead of doing the tensor evaluations at  $\xi = \mathbf{0}$ , we do integrations from  $\xi = \mathbf{0}$  to the corresponding positions:

$$(\mathbf{X}_0^e)^{i+1} \Big|_{\xi=\xi_a} - (\mathbf{X}_0^e)^{i+1} \Big|_{\xi=0} = \int_{\mathbf{X}_{\text{REF}}^e|_{\xi=0}}^{\mathbf{X}_{\text{REF}}^e|_{\xi=\xi_a}} (\mathbf{F}^{i+1})^{-1} d\mathbf{X}_{\text{REF}}. \tag{24}$$

Here  $\xi_a$  is the representative parametric position assigned to the Bézier control point for  $a$  (see Fig. 1), and we use a straight path from  $\mathbf{0}$  to  $\xi_a$ . The representative parametric positions are equally spaced. The first approximation here is performing the integration with the midpoint rule:

$$\begin{aligned} & (\mathbf{X}_0^e)^{i+1} \Big|_{\xi=\xi_a} - (\mathbf{X}_0^e)^{i+1} \Big|_{\xi=0} \\ & \approx (\mathbf{F}^{i+1})^{-1} \Big|_{\xi=\xi_a/2} \left( \mathbf{X}_{\text{REF}}^e \Big|_{\xi=\xi_a} - \mathbf{X}_{\text{REF}}^e \Big|_{\xi=0} \right). \end{aligned} \tag{25}$$

The second approximation is to assume that the relationship given by Eq. (25) between  $(\mathbf{X}_0^e)^{i+1} \Big|_{\xi=\xi_a}$  and  $\mathbf{X}_{\text{REF}}^e \Big|_{\xi=\xi_a}$  can also be used between the control points  $(\hat{\mathbf{X}}_0^e)_a^{i+1}$  and  $(\hat{\mathbf{X}}_{\text{REF}}^e)_a$ :

$$\begin{aligned}
 & (\hat{\mathbf{X}}_0^e)^{i+1} - (\mathbf{X}_0^e)^{i+1} \Big|_{\xi=0} \\
 & \approx \left( \mathbf{F}^{i+1} \right)^{-1} \Big|_{\xi=\xi_a/2} \left( (\hat{\mathbf{X}}_{\text{REF}}^e)_a - \mathbf{X}_{\text{REF}}^e \Big|_{\xi=0} \right). \tag{26}
 \end{aligned}$$

This is the new version of the “direct-update (DU)” process (see [55] for the original DU process). The “recursive-update (RU)” process is given as

$$\begin{aligned}
 & (\hat{\mathbf{X}}_0^e)^{i+1} - (\mathbf{X}_0^e)^{i+1} \Big|_{\xi=0} \\
 & \approx \left( \mathbf{F}^{i+1} \right)^{-1} \Big|_{\xi=\xi_a/2} \mathbf{F} \left( \mathbf{X}_{\text{REF}}, (\mathbf{X}_0^e)^i \right) \Big|_{\xi=\xi_a/2} \\
 & \quad \left( (\hat{\mathbf{X}}_0^e)^i - (\mathbf{X}_0^e)^i \Big|_{\xi=0} \right) \tag{27}
 \end{aligned}$$

$$\begin{aligned}
 & = \left( \mathbf{U}^i \right)^{-1} \Big|_{\xi=\xi_a/2} \mathbf{U} \left( \mathbf{X}_{\text{REF}}, (\mathbf{X}_0^e)^i \right) \Big|_{\xi=\xi_a/2} \\
 & \quad \left( (\hat{\mathbf{X}}_0^e)^i - (\mathbf{X}_0^e)^i \Big|_{\xi=0} \right). \tag{28}
 \end{aligned}$$

We note that the tensor–vector operations in Eqs. (26), (27) and (28) actually involve the augmented versions of the tensors, where the augmented version of a tensor  $\mathbf{F}$  is defined as

$$\begin{bmatrix} \mathbf{F} & \mathbf{0} \\ \mathbf{0}^T & 1 \end{bmatrix}, \tag{29}$$

and the augmented versions of the vectors  $(\mathbf{X}_0^e)^{i+1} \Big|_{\xi=0}$ ,  $\mathbf{X}_{\text{REF}}^e \Big|_{\xi=0}$  and  $(\mathbf{X}_0^e)^i \Big|_{\xi=0}$ , as defined by Eq. (7).

In the actual computations, we start from the “ZSS template”:  $(\mathbf{X}_0^e)^0 = (\mathbf{X}_0^e)_{\text{TEMP}}$  (see Sect. 3). In the steady-state structural mechanics computations, it is reasonable to start from displacement  $\mathbf{y} = \mathbf{0}$ . However, it is unlikely for that to be a good match for the ZSS. To improve the convergence of the structural mechanics solution for  $i = 0$ , we use an incremental loading and modify the initial guess for the EBZSS based on that ramping:

$$\left( (\mathbf{X}_0^e)^0 \right)^j = (1 - t^j) \mathbf{X}_{\text{REF}} + t^j (\mathbf{X}_0^e)_{\text{TEMP}}. \tag{30}$$

Here  $0 < t^1 \leq t^2 \leq \dots \leq t^N = 1$ ,  $N$  is the number of nonlinear-iteration steps used in computing  $\mathbf{y}^0$ , and the iterations start with  $(\mathbf{y}^0)^0 = \mathbf{0}$ . We also ramp the load:

$$(\mathbf{h}^h)^j = t^j \mathbf{h}^h, \tag{31}$$

where  $\mathbf{h}^h$  is the target load. The ramping options include having a ramping profile where the  $t^j$  values change at every certain number of nonlinear-iteration steps. With that, we obtain the steady-state solution  $\mathbf{y}^0$  for  $(\mathbf{X}_0^e)^0 = (\mathbf{X}_0^e)_{\text{TEMP}}$

based on the full load. For  $i = 1$  and beyond,  $(\mathbf{X}_0^e)^i$  is calculated from Eq. (28), and the nonlinear iterations used in computing  $\mathbf{y}^i$  start with  $(\mathbf{y}^i)^0 = \mathbf{0}$ .

### 3 Modeling the artery ZSS: straight-tube ZSS template

An analytical relationship between the ZS and reference states of straight-tube segments was given in [55]. Here we will call that relationship “straight-tube ZSS template.” We describe the straight tube in the target state, which is here the reference state, with three lengths:  $\bar{\ell}$ ,  $h$  and  $L$ . They are the circumferential length of the arterial-wall midsurface, wall thickness, and the longitudinal length. The tube volume is

$$V = \bar{\ell} h L. \tag{32}$$

For an artery, beyond having a target shape under a given load, there are some significant properties. One of them is the opening angle,  $\phi$ , seen after a longitudinal cut, which we call the “LC state”. Figure 2 summarizes the template.

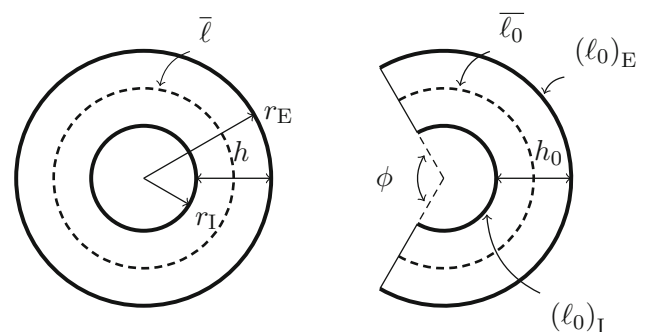
*Remark 2* We note that in general it is not necessary for the LC state to be a ZSS.

### 4 2D test computations

We define a parameter  $\alpha$ :

$$\bar{\ell}_0 = \alpha \ell_1, \tag{33}$$

where  $\ell_1$  is the circumferential length of the inner tube surface in the target state. The task of calculating  $\bar{\ell}_0$  becomes the task of calculating  $\alpha$ .



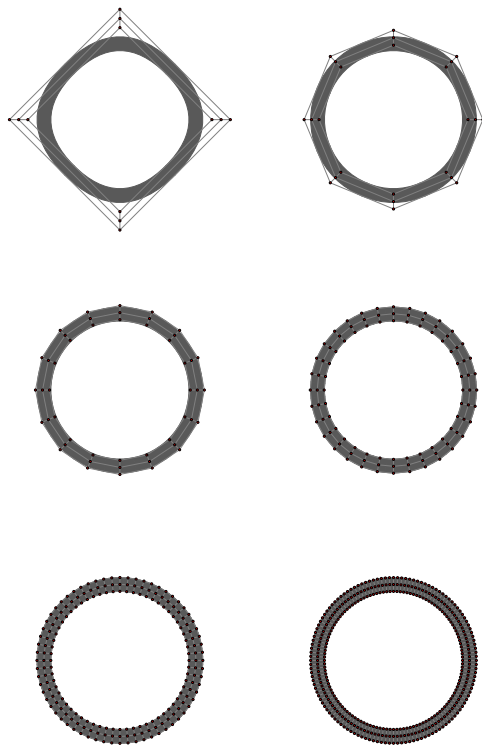
**Fig. 2** Straight tube in the target (left) and ZS (right) states. The dashed lines denote the arterial-wall midsurface in each state

### 4.1 Meshes

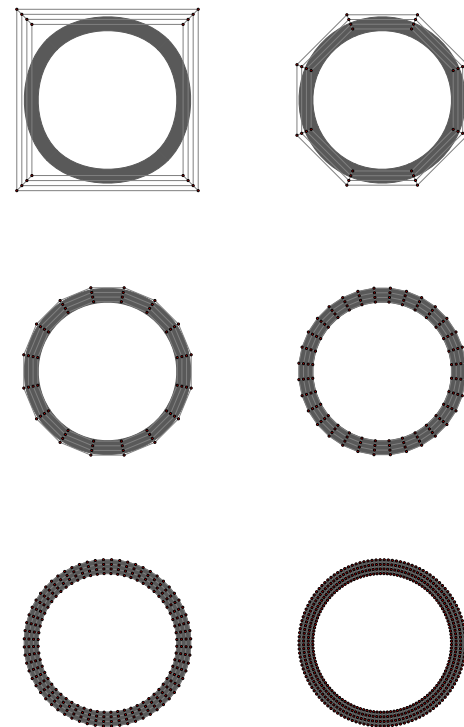
In the test computations here, we use quadratic and cubic NURBS basis functions. Although NURBS can represent a circular arc exactly, it cannot do that throughout a full circle while retaining the  $C^1$  continuity of the basis functions. Therefore, here we simplify the basis functions to uniform B-splines. Figures 3 and 4 show the meshes used. We evaluate how well the meshes represent the circular arcs. For that, we inspect the radius of curvature,  $\rho$ . Figures 5 and 6 show  $\rho/\bar{\rho}$  as a function of the circumferential parametric coordinate,  $\xi$ , where  $\bar{\rho}$  is the average radius.

*Remark 3* The highest shape function value is at  $\xi = 0$  for the quadratic B-splines, and at  $\xi = -1$  and  $\xi = 1$  for the cubic B-splines. From that, and from Figs. 5 and 6, we conclude that the radius of curvature will be lowest at points closest to the control points, and highest between those points.

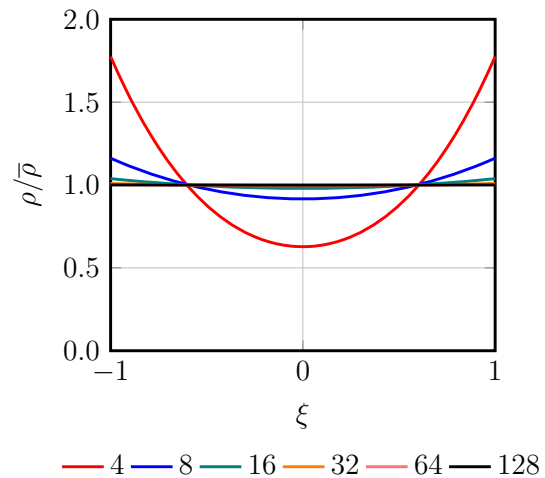
Figure 7 shows the standard deviation of  $\rho/\bar{\rho}$ , as a function of the arc angle  $\Delta\theta$  represented by an element, where  $\Delta\theta = 2\pi/n_{el}$ , and  $n_{el}$  is the number of elements. We note that both quadratic and cubic B-splines have second-order accuracy.



**Fig. 3** Quadratic B-spline meshes with 4, 8, 16, 32, 64, and 128 elements. The red circles are the control points and the gray part is the actual tube. (Color figure online)



**Fig. 4** Cubic B-spline meshes with 4, 8, 16, 32, 64, and 128 elements. The red circles are the control points and the gray part is the actual tube. (Color figure online)

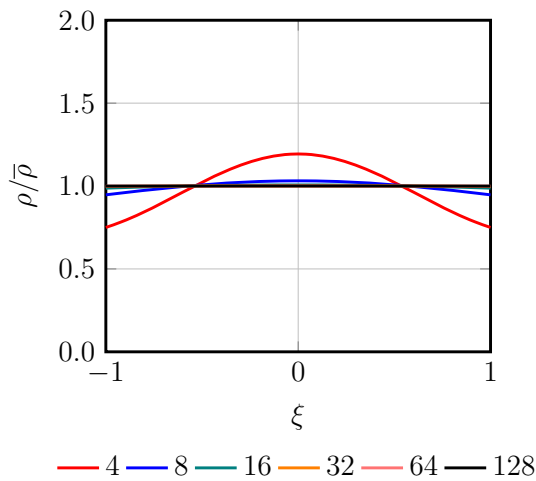


**Fig. 5** Representation of the radius of curvature within an element with quadratic B-splines. The curves are for the six meshes,  $\xi$  is the circumferential parametric coordinate, and  $\bar{\rho}$  is the average radius

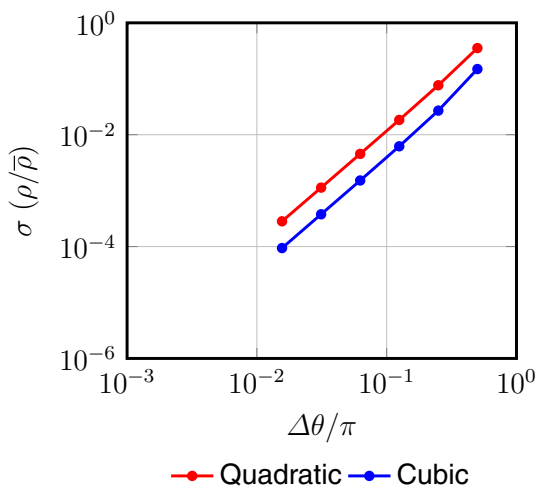
### 4.2 Curvature matching in the ZSS

Curvature matching in the ZSS is done by first converting the B-spline element in the target state to Bézier representation. After that, for the specified  $\phi$ , we generate element configurations with the objective of having a constant radius of curvature in the ZSS. Then we convert that back to B-spline representation. Figures 8 and 9 show examples of the





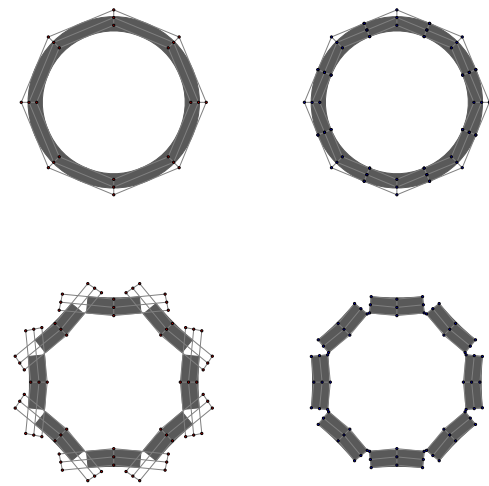
**Fig. 6** Representation of the radius of curvature within an element with cubic B-splines. The curves are for the six meshes,  $\xi$  is the circumferential parametric coordinate, and  $\bar{\rho}$  is the average radius



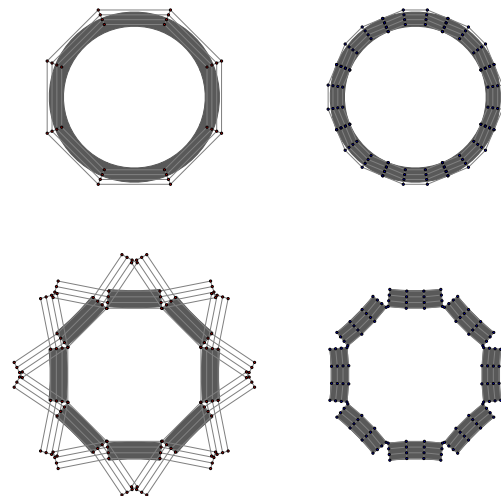
**Fig. 7** Standard deviation of  $\rho/\bar{\rho}$  as a function of the arc angle  $\Delta\theta$  represented by a B-spline element, where  $\bar{\rho}$  is the average radius

process for  $\phi = 5\pi/2$ , which results in Bézier elements with  $\Delta\theta = (2\pi - \phi)/n_{el} = -\pi/(2n_{el})$ , and for a given value of  $\alpha$ . The negative value implies that the outer surface is smaller than the inner surface.

In the case of quadratic Bézier functions, we choose the middle control point to be on the tangents to the inner surface at the two other control points. This determines the curvature in the ZSS. For cubic Bézier functions, we have an additional degree of freedom, and we use that by choosing the control points to be also at equally-spaced angular positions. Figure 10 shows quadratic convergence of the standard deviation of  $\rho_0/\bar{\rho}_0$  for the Bézier elements, where  $\bar{\rho}_0$  is the average radius. We note from Figs. 7 and 10 that Bézier elements yield the same representation quality as the B-spline elements for quadratic functions, and slightly better quality for cubic functions.



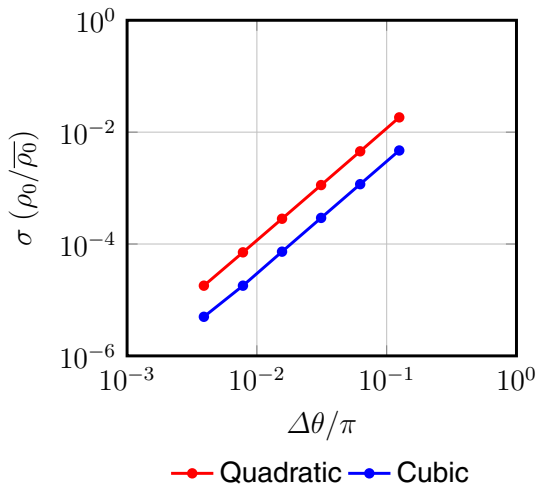
**Fig. 8** Curvature matching in the ZSS. Quadratic basis functions with eight elements. The B-spline mesh in the target state (*top left*) is converted to Bézier representation (*top right*). From that, for the specified  $\phi$  and for a given value of  $\alpha$ , we generate element configurations with the objective of having constant radius of curvature in the ZSS (*bottom right*). Then we convert that back to B-spline representation (*bottom left*)



**Fig. 9** Curvature matching in the ZSS. Cubic basis functions with eight elements. The B-spline mesh in the target state (*top left*) is converted to Bézier representation (*top right*). From that, for the specified  $\phi$  and for a given value of  $\alpha$ , we generate element configurations with the objective of having constant radius of curvature in the ZSS (*bottom right*). Then we convert that back to B-spline representation (*bottom left*)

### 4.3 Computational results

For a range of  $\alpha$  values, we go through the process described in Sect. 4.2 and, using the B-spline elements with curvature matching in the ZSS, we do structural mechanics computations. We obtain the steady-state solutions corresponding to a constant pressure value of  $p_0 = 92$  mm Hg. In these computations, the arterial wall is made of hypere-



**Fig. 10** Curvature matching in the ZSS. Standard deviation of  $\rho_0/\bar{\rho}_0$  as a function of the arc angle  $\Delta\theta$  represented by a Bézier element, where  $\bar{\rho}_0$  is the average radius

lastic (Fung) material. The material constants  $D_1$  and  $D_2$  are  $2.6447 \times 10^3 \text{ N/m}^2$  and 8.365, and the penalty Poisson’s ratio is 0.45. We note that, because of the process in Sect. 4.2, the inner-surface shapes obtained from the computations will be at the best as good as those in the upper-left picture in Figs. 8 and 9, for whatever the number of elements are.

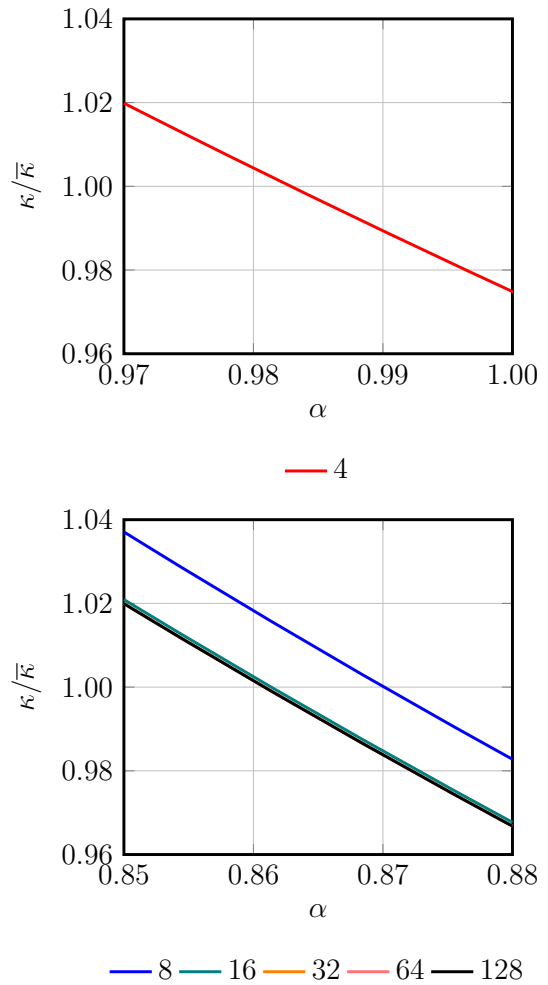
The structural mechanics computations generate a relationship between the curvature in the deformed state and  $\alpha$ , and from that we select the  $\alpha$  value that matches the curvature in the target state. Figures 11 and 12 show the average curvature of the inner surface in the deformed state as a function of  $\alpha$  for the six meshes with quadratic and cubic B-splines. For quadratic functions, with the mesh made of four elements, the  $\alpha$  value that matches the target curvature is very different than the value obtained with the other meshes. For cubic functions, except for the mesh made of four elements, curves for all the meshes coincide.

With all the meshes and the  $\alpha$  values displayed in Fig. 13, we compute the steady-state structural mechanics solutions to examine the stretches. We evaluate the stretches at the  $4 \times 4$  Gaussian quadrature points (see Fig. 14).

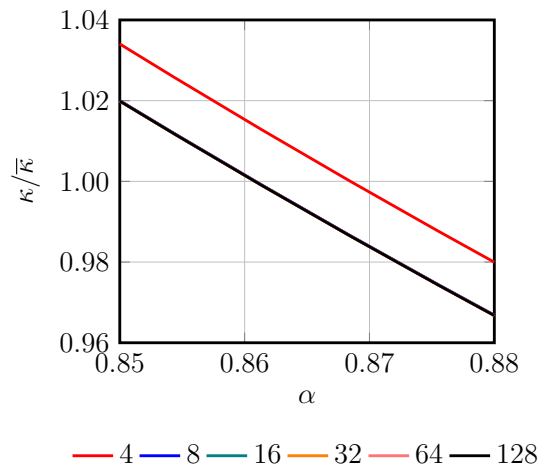
*Remark 4* In the structural mechanics computations with quadratic functions, we use  $3 \times 3$  quadrature points.

*Remark 5* The integration points are defined over the ZSS.

Because of the circular symmetry, the stretches should depend only on the radial position. Here we check that by examining their values along different radial lines in the element, which are called “inner” and “outer” in Fig. 14. Figure 15 shows, for the meshes with eight elements, the radial stretches. For the meshes with 16 or more elements,

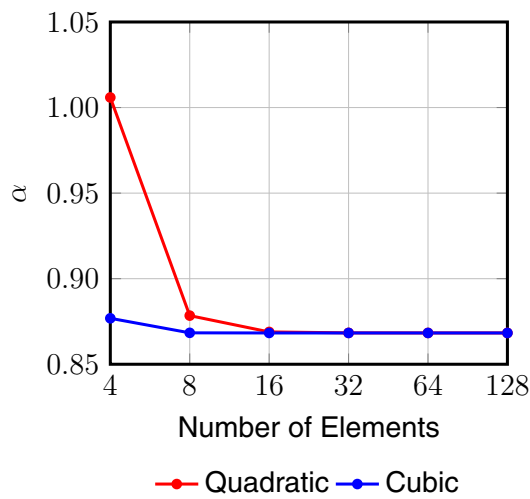


**Fig. 11** Average curvature in the deformed state as a function of  $\alpha$  for the six meshes with quadratic B-splines, where  $\bar{\kappa}$  is the curvature in the target state

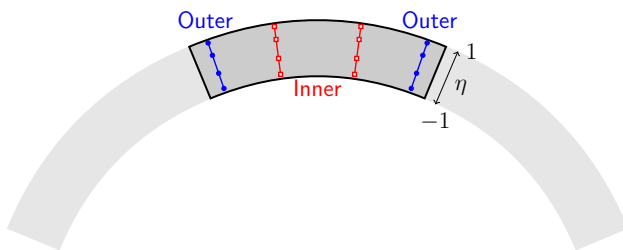


**Fig. 12** Average curvature in the deformed state as a function of  $\alpha$  for the six meshes with cubic B-splines, where  $\bar{\kappa}$  is the curvature in the target state





**Fig. 13** For all the meshes, the  $\alpha$  value that matches the target curvature



**Fig. 14** Schematic display of the integration points

the radial stretches along the inner and outer lines are basically indistinguishable.

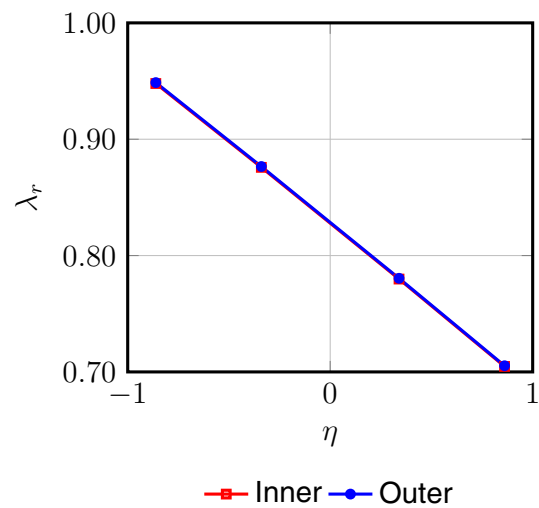
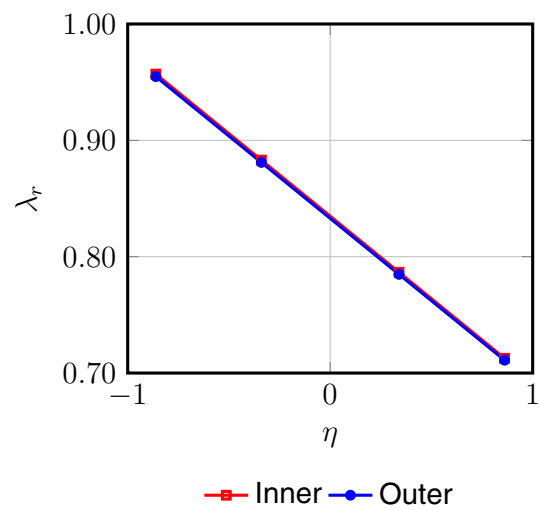
*Remark 6* The precise definitions of the radial and circumferential directions for the stretch components are based on the parametric coordinates.

Figure 16 shows, for the meshes with 16 elements, the circumferential stretches. For the meshes with 32 or more elements, the circumferential stretches along the inner and outer lines are basically indistinguishable.

Table 1 shows, for the mesh with 128 elements and cubic B-splines, the stretch values at the integration points. We assume the values in Table 1 to be the actual values and calculate based on that the relative error for the other meshes. Figure 17 shows the relative error in the radial and circumferential stretches at all eight integration points for all those other meshes.

### 5 3D computation

Here we show how the method we have presented can be used in a 3D computation where the target geometry is coming from medical image of a human aorta. The first step is to generate a mesh for the target geometry, and we do the volume mesh generation in a special fashion, taking into account

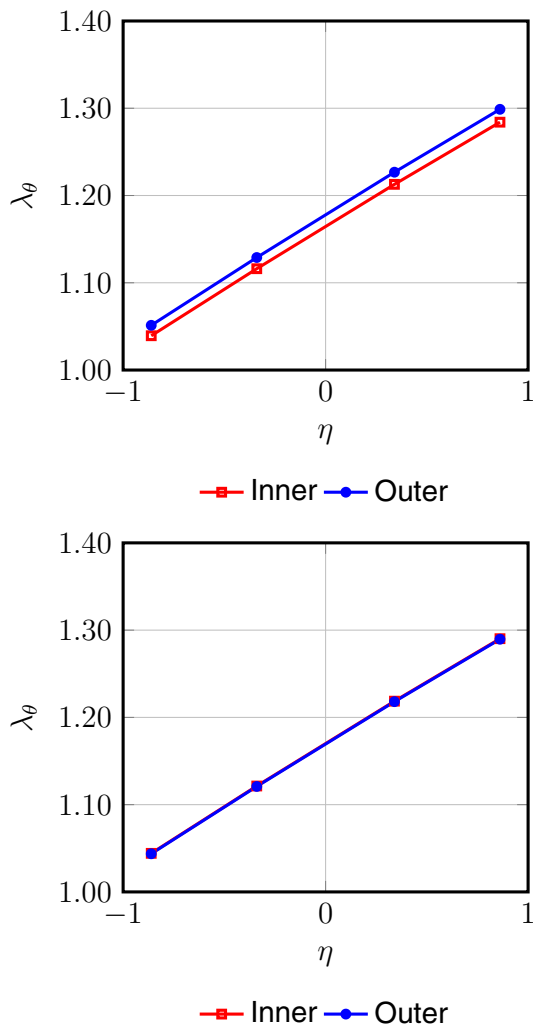


**Fig. 15** Radial stretches for the meshes with eight elements and quadratic (top) and cubic (bottom) B-splines

the radial stretch. As seen in Fig. 15, in the 2D computations  $\lambda_r$  was linear in  $\eta$ , the parametric coordinate in the radial direction in the ZSS. In that computational setting, in the ZSS, the parametric and physical coordinates in the radial direction also had a linear relationship. Consequently, in the ZSS,  $\lambda_r$  was linear in the physical coordinate in the radial direction. Based on these considerations, we do the volume mesh generation for the target geometry with the objective of having in the ZSS a linear relationship between the parametric and physical coordinates in the radial direction. In this example, we use cubic B-splines.

#### 5.1 Surface and volume mesh generation for the target geometry

The surface geometry can be obtained from the medical images using techniques such as those described in [37]. Then, we generate a surface mesh as shown in Fig. 18.



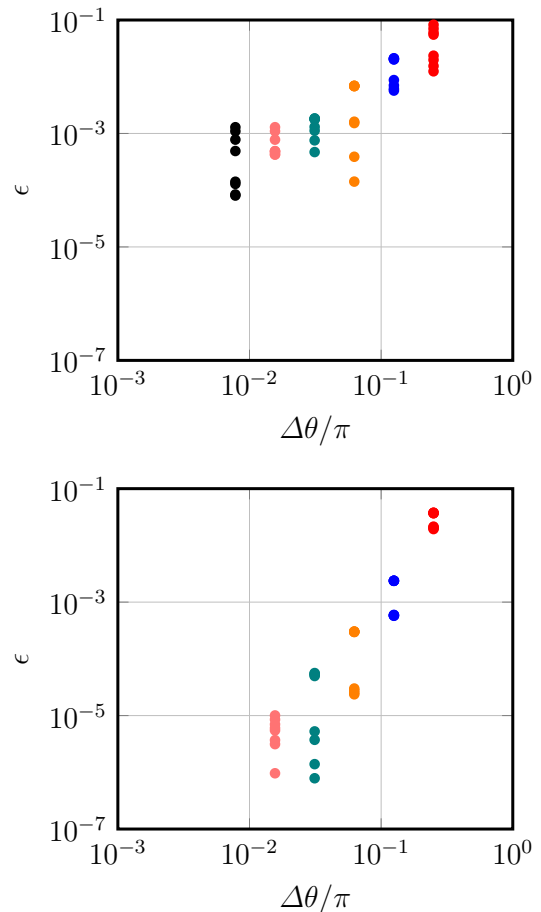
**Fig. 16** Circumferential stretches for the meshes with 16 elements and quadratic (*top*) and cubic (*bottom*) B-splines

**Table 1** Stretch values at the integration points for the mesh with 128 elements and cubic B-splines

$\eta$	$\lambda_r$	$\lambda_\theta$
-0.86114	0.9481	1.044
-0.33998	0.8760	1.121
0.33998	0.7800	1.218
0.86114	0.7049	1.289

The values along the inner and outer lines are identical for the number of digits displayed

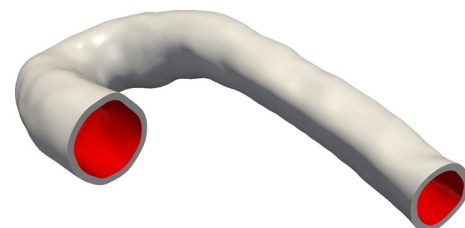
Next we generate the volume mesh. We have only one element in the radial direction, with four control points radially. The control points of the inner surface of the artery are the control points of the surface mesh in Fig. 18. To locate the control points of the outer surface, for each control point of the inner surface, we take a distance equal to the wall thickness in the direction normal to the surface at the collocation point corresponding to that control point. Figure 19 shows



**Fig. 17** Relative error in the radial and circumferential stretches at all eight integration points for the meshes with quadratic (*top*) and cubic (*bottom*) B-splines. The colors represent different meshes, with the same color convention used in Fig. 11. The relative error is calculated based on the value obtained from the mesh with 128 elements and cubic B-splines. (Color figure online)



**Fig. 18** Surface geometry



**Fig. 19** Volume geometry

the volume geometry. In this example, the inner diameter at the inlet is 27.3 mm, and use a constant wall thickness of 2.5 mm. The two interior control points do not effect the arterial volume geometry. However, as mentioned above, we determine their locations in a special fashion, based on considerations related to the radial stretch. We will explain that in the next subsection.

### 5.2 Mesh generation for $(\mathbf{X}_0^e)^0$

To obtain  $(\mathbf{X}_0^e)^0$ , we use the straight-tube ZSS template with the data obtained in Sect. 4. We first calculate the principal curvature directions for the elements of the inner surface. Figure 20 shows the directions of maximum principal curvature. As expected, in most places, the direction of maximum principal curvature coincides with the circumferential direction. Figure 21 shows the distribution of the maximum principal curvature.

Using those two directions and their cross product, we create an “ideal element” as shown in Fig. 22. We see the direction of maximum principal curvature as the circumferential direction  $\theta$ , the other principal direction as the longitudinal direction  $z$ , and the cross-product direc-

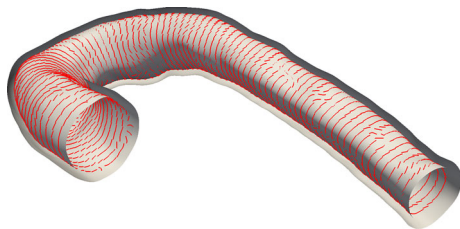


Fig. 20 Directions of maximum principal curvature

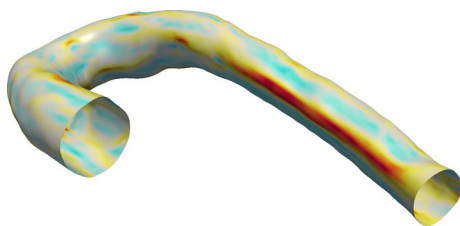


Fig. 21 Distribution of the maximum principal curvature

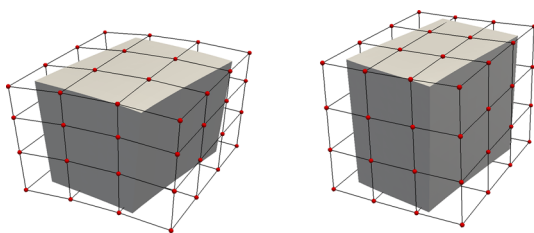


Fig. 22 Physical and ideal elements. The grey colored shape represents the physical element, and the lines and points represent the corresponding ideal Bézier control element. Target state (left) and ZSS (right)

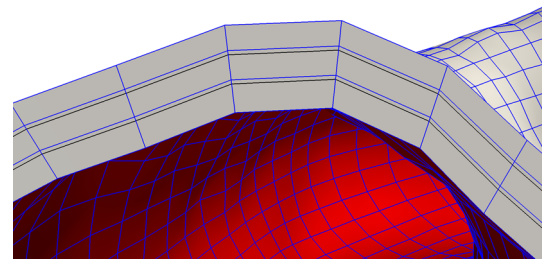


Fig. 23 Control mesh in the target state, with the blue lines representing the mesh we use, which is based on the control-point spacing we chose in the radial direction, and with the black lines representing equal spacing of control points in the radial direction. (Color figure online)

tion as the radial direction  $r$ . The ideal element is defined with its inner surface having those principal curvatures in the  $\theta$  and  $z$  directions, and expanding in all three directions to cover the physical element. To define the EBZSS for the ideal element, we use the straight-tube ZSS template with the data obtained in Sect. 4. The curvature in the circumferential direction comes from  $\phi = 5\pi/2$  and  $\alpha = 0.870$ , and for the  $z$  direction we assume  $\lambda_z = 1.0$  (no stretch) in this example. What we do in the radial direction is based on having the  $\lambda_r$  distribution from Fig. 15. To that end, as mentioned in the earlier parts of this section, we prefer to have a linear relationship between the parametric and physical coordinates in the radial direction in the ZSS. For that reason, the control points are equally spaced in the radial direction in the ZSS, but not in the target state.

With the ideal elements from the target state and ZSS, by using a least-squares projection, we map the physical element from the target state to the ZSS. Here we face a second choice to make, and that is for the control-point spacing in the radial direction for the physical element in the target state and ZSS. We choose the spacing in the target state to be proportional to the spacing for the ideal element in the target state, leading to a distribution in the ZSS that is not far from being equally spaced. Figure 23 shows the control mesh in the target state.

*Remark 7* We expect that having proportionality between the spacings for the physical and ideal elements will facilitate accurate representation of the radial stretch.

Figure 24 shows the elements of  $(\mathbf{X}_0^e)^0$  obtained with the process we have described. From  $(\mathbf{X}_0^e)^0$  and  $\mathbf{X}_{REF}$ , we calculate the circumferential, longitudinal and radial stretches, defined based on the parametric coordinates in the ideal element, with the three directions corresponding to the directions of maximum and minimum principal curvatures and the cross product of the two directions. Figures 25, 26 and 27 show those stretches.

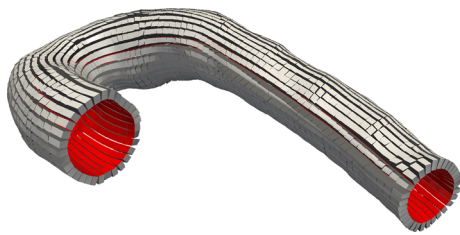


Fig. 24 Elements of  $(\mathbf{X}_0^e)^0$

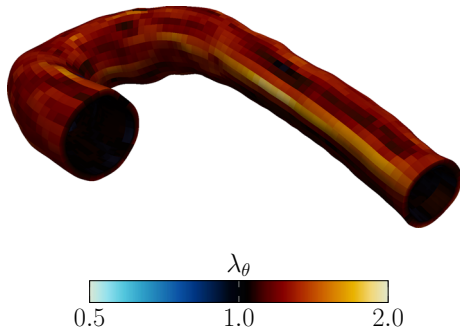


Fig. 25 Circumferential stretch from  $(\mathbf{X}_0^e)^0$  and  $\mathbf{X}_{REF}$

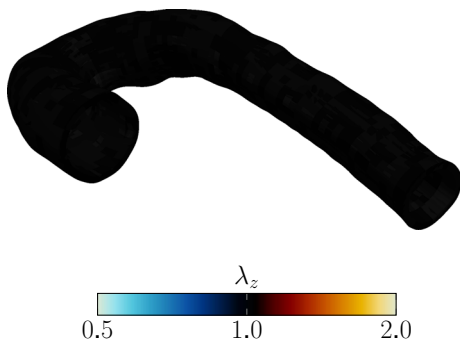


Fig. 26 Longitudinal stretch from  $(\mathbf{X}_0^e)^0$  and  $\mathbf{X}_{REF}$

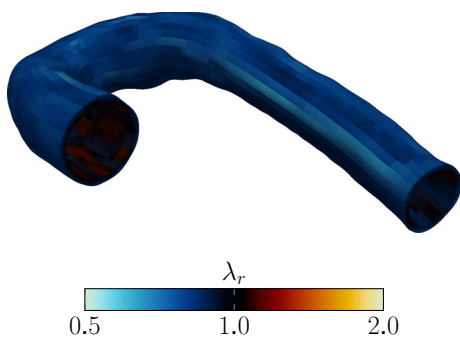


Fig. 27 Radial stretch from  $(\mathbf{X}_0^e)^0$  and  $\mathbf{X}_{REF}$

### 5.3 Iterations to calculate $\mathbf{X}_0^e$

In computation of  $\mathbf{y}^0$  from  $(\mathbf{X}_0^e)^0$ , we use the ramping method described in Sect. 2.4, for reasons also described in that section. Figure 28 shows  $(\mathbf{X}_0^e)^0$  and  $\mathbf{y}^0$ . Figures 29, 30, 31 and

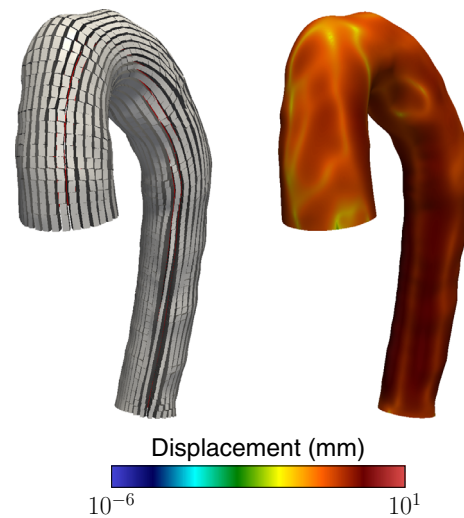


Fig. 28 Elements of  $(\mathbf{X}_0^e)^0$  (left) and  $\mathbf{y}^0$  (right)

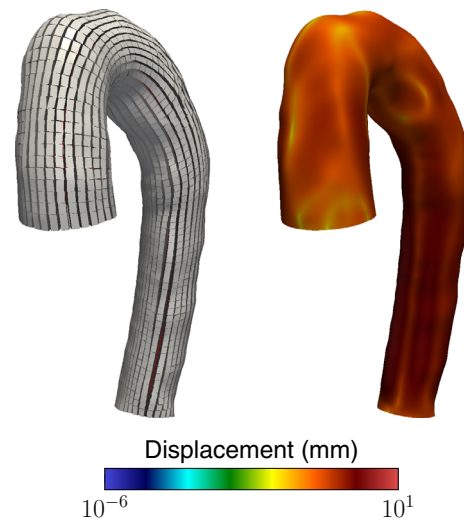
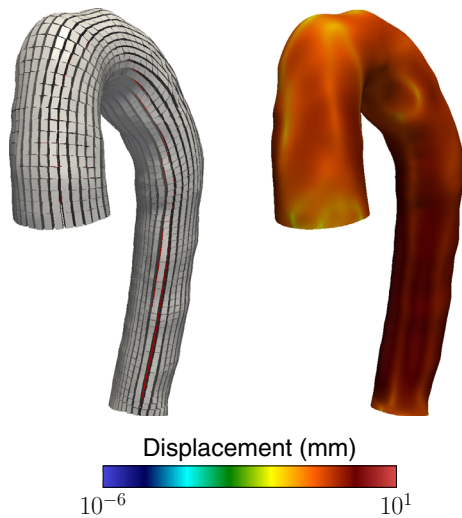


Fig. 29 Elements of  $(\mathbf{X}_0^e)^1$  (left) and  $\mathbf{y}^1$  (right)

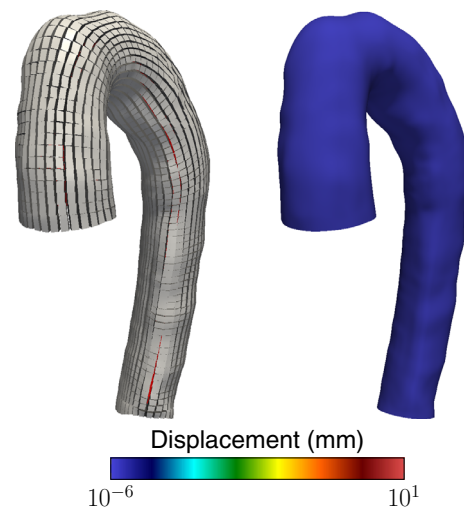
32 show  $(\mathbf{X}_0^e)^i$  and  $\mathbf{y}^i$  for  $i = 1, 2, 300,$  and  $1000$ . Figure 33 shows the maximum of  $\|\mathbf{y}^i\|$  as a function of  $i$ . There are some very-low-magnitude oscillations, but the solution converges very well.

### 5.4 Observations on the results

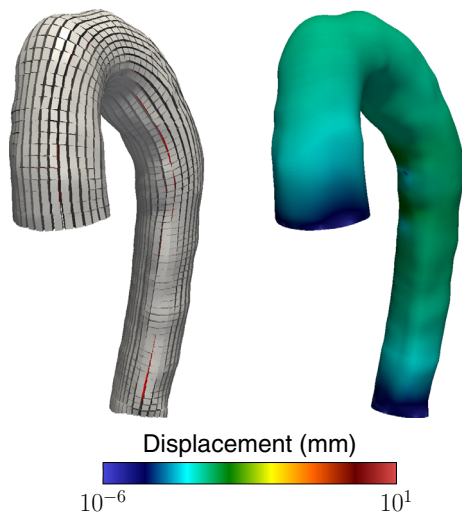
From converged  $\mathbf{X}_0^e$  and  $\mathbf{X}_{REF}$ , we calculate the circumferential, longitudinal and radial stretches, defined based on the parametric coordinates in the ideal element in the target state, with the three directions corresponding to the directions of maximum and minimum principal curvatures and the cross product of the two directions. The stretches are shown in Figs. 34, 35 and 36. In most places the stretches are very comparable to what we obtained from  $(\mathbf{X}_0^e)^0$  and  $\mathbf{X}_{REF}$  (see Figs. 25, 26 and 27). However, there are some differences



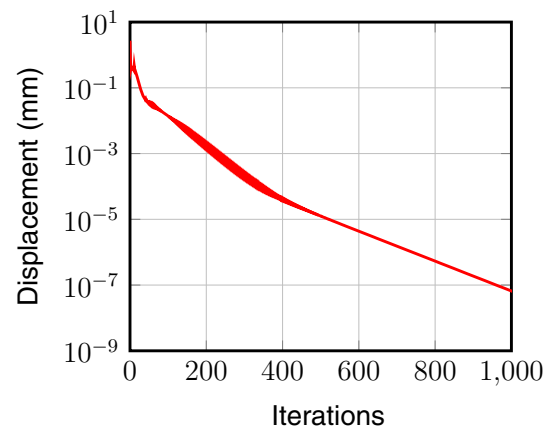
**Fig. 30** Elements of  $(X_0^\epsilon)^2$  (left) and  $y^2$  (right)



**Fig. 32** Elements of  $(X_0^\epsilon)^{1000}$  (left) and  $y^{1000}$  (right)



**Fig. 31** Elements of  $(X_0^\epsilon)^{300}$  (left) and  $y^{300}$  (right)

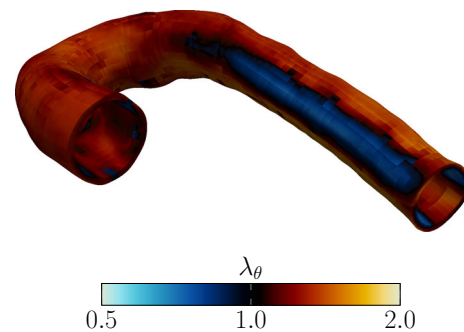


**Fig. 33** Maximum of  $\|y^i\|$  as a function of  $i$  (EBZSS iteration counter)

locally, such as at the high-curvature region of the descending aorta. The high curvature might be due to the surrounding tissues. We also note the template used in the computation was based on the straight part of the aorta. Improving our method for determining the EBZSS might require more observed data, and additional boundary conditions representing the surrounding tissues.

### 6 Concluding remarks

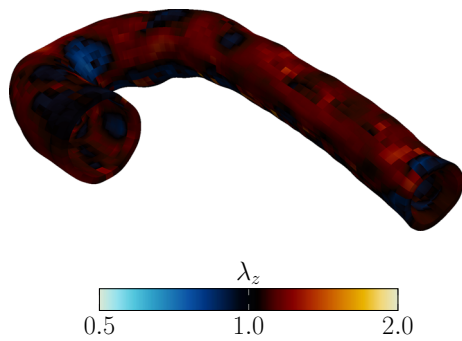
We have presented the version of the EBZSS estimation method with isogeometric wall discretization and have successfully applied the method to patient-specific modeling of a human aorta. The EBZSS estimation method, originally introduced in the context of finite element wall discretiza-



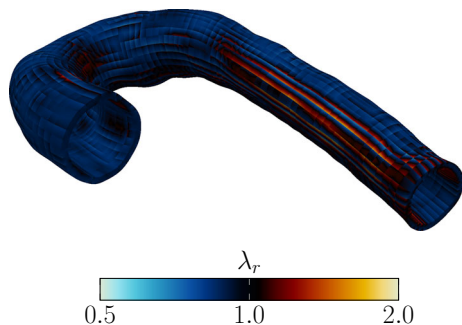
**Fig. 34** Circumferential stretch from converged  $X_0^\epsilon$  and  $X_{REF}$

tion, will help us estimate the ZSS in patient-specific arterial FSI computations, where the image-based arterial geometry does not come from a ZSS. The method consists of three main components. 1. An iterative method, which starts with a calculated initial guess, is used for computing the EBZSS such that under a given load, the image-based target shape





**Fig. 35** Longitudinal stretch from converged  $\mathbf{X}_0^e$  and  $\mathbf{X}_{REF}$



**Fig. 36** Radial stretch stretch from converged  $\mathbf{X}_0^e$  and  $\mathbf{X}_{REF}$

is matched. 2. A method for straight-tube segments is used for computing the EBZSS so that the given diameter and longitudinal stretch in the target configuration are matched together with the opening angle. 3. An element-based mapping between the artery and straight-tube configurations. This provides the mapping from the arterial configuration to the straight-tube configuration, and from the estimated EBZSS of the straight-tube configuration back to the arterial configuration, to be used as the initial guess for the iterative method that matches the target shape. With isogeometric discretization, we obtain the element-based mapping directly, instead of extracting it from the mapping between the artery and straight-tube segments. That is because all we need for the element-based mapping, including the curvatures, can be obtained within an element. With NURBS basis functions, we may be able to achieve a similar level of accuracy as with the linear basis functions, but using larger-size and much fewer elements. Higher-order NURBS basis functions allow representation of more complex shapes within an element. To demonstrate how the new EBZSS estimation method performs, we first presented 2D test computations with straight-tube configurations, carried out with quadratic and cubics basis functions and meshes made of different number of elements. Then we showed how the method can be used in a 3D computation where the target geometry is coming from medical image of a human aorta.

**Acknowledgements** This work was supported in part by JST-CREST; Grant-in-Aid for Scientific Research (S) 26220002 from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT); Rice–Waseda research agreement.

## References

1. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2004) Computation of cardiovascular fluid–structure interactions with the DSD/SST method. In: Proceedings of the 6th world congress on computational mechanics (CD-ROM), Beijing
2. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2004) Influence of wall elasticity on image-based blood flow simulations. *Trans Jpn Soc Mech Eng Ser A* 70:1224–1231. doi:10.1299/kikaia.70.1224 (in Japanese)
3. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Computer modeling of cardiovascular fluid–structure interactions with the deforming-spatial-domain/stabilized space–time formulation. *Comput Methods Appl Mech Eng* 195:1885–1895. doi:10.1016/j.cma.2005.05.050
4. Torii R, Oshima M, Kobayashi T, Takagi K, Tezduyar TE (2006) Fluid–structure interaction modeling of aneurysmal conditions with high and normal blood pressures. *Comput Mech* 38:482–490. doi:10.1007/s00466-006-0065-6
5. Tezduyar TE (1992) Stabilized finite element formulations for incompressible flow computations. *Adv Appl Mech* 28:1–44. doi:10.1016/S0065-2156(08)70153-4
6. Tezduyar TE (2003) Computation of moving boundaries and interfaces and stabilization parameters. *Int J Numer Methods Fluids* 43:555–575. doi:10.1002/flid.505
7. Hughes TJR, Liu WK, Zimmermann TK (1981) Lagrangian–Eulerian finite element formulation for incompressible viscous flows. *Comput Methods Appl Mech Eng* 29:329–349
8. Bazilevs Y, Calo VM, Zhang Y, Hughes TJR (2006) Isogeometric fluid–structure interaction analysis with applications to arterial blood flow. *Comput Mech* 38:310–322
9. Bazilevs Y, Calo VM, Tezduyar TE, Hughes TJR (2007)  $\gamma$ - $\beta$  discontinuity-capturing for advection-dominated processes with application to arterial drug delivery. *Int J Numer Methods Fluids* 54:593–608. doi:10.1002/flid.1484
10. Bazilevs Y, Calo VM, Hughes TJR, Zhang Y (2008) Isogeometric fluid–structure interaction: theory, algorithms, and computations. *Comput Mech* 43:3–37
11. Isaksen JG, Bazilevs Y, Kvamsdal T, Zhang Y, Kaspersen JH, Waterloo K, Romner B, Ingebrigtsen T (2008) Determination of wall tension in cerebral artery aneurysms by numerical simulation. *Stroke* 39:3172–3178
12. Bazilevs Y, Gohean JR, Hughes TJR, Moser RD, Zhang Y (2000) 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 Eng* 198(2009):3534–3550
13. Bazilevs Y, Hsu M-C, Benson D, Sankaran S, Marsden A (2009) Computational fluid–structure interaction: methods and application to a total cavopulmonary connection. *Comput Mech* 45:77–89
14. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Liang X, Kvamsdal T, Brekken R, Isaksen J (2010) A fully-coupled fluid–structure interaction simulation of cerebral aneurysms. *Comput Mech* 46:3–16
15. Sugiyama K, Ii S, Takeuchi S, Takagi S, Matsumoto Y (2010) Full Eulerian simulations of biconcave neo-Hookean particles in a Poiseuille flow. *Comput Mech* 46:147–157
16. Bazilevs Y, Hsu M-C, Zhang Y, Wang W, Kvamsdal T, Hentschel S, Isaksen J (2010) Computational fluid–structure interaction:



- methods and application to cerebral aneurysms. *Biomech Model Mechanobiol* 9:481–498
17. Bazilevs Y, del Alamo JC, Humphrey JD (2010) From imaging to prediction: emerging non-invasive methods in pediatric cardiology. *Progr Pediatr Cardiol* 30:81–89
  18. Hsu M-C, Bazilevs Y (2011) Blood vessel tissue prestress modeling for vascular fluid–structure interaction simulations. *Finite Elem Anal Des* 47:593–599
  19. Yao JY, Liu GR, Narmoneva DA, Hinton RB, Zhang Z-Q (2012) Immersed smoothed finite element method for fluid–structure interaction simulation of aortic valves. *Comput Mech* 50:789–804
  20. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Computational fluid–structure interaction: methods and applications. Wiley, Cambridge ISBN 978-0470978771
  21. Bazilevs Y, Takizawa K, Tezduyar TE (2013) Challenges and directions in computational fluid–structure interaction. *Math Models Methods Appl Sci* 23:215–221. doi:10.1142/S0218202513400010
  22. Long CC, Marsden AL, Bazilevs Y (2013) Fluid–structure interaction simulation of pulsatile ventricular assist devices. *Comput Mech* 52:971–981. doi:10.1007/s00466-013-0858-3
  23. Esmaily-Moghadam M, Bazilevs Y, Marsden AL (2013) A new preconditioning technique for implicitly coupled multidomain simulations with applications to hemodynamics. *Comput Mech* 52:1141–1152. doi:10.1007/s00466-013-0868-1
  24. Long CC, Esmaily-Moghadam M, Marsden AL, Bazilevs Y (2014) Computation of residence time in the simulation of pulsatile ventricular assist devices. *Comput Mech* 54:911–919. doi:10.1007/s00466-013-0931-y
  25. Yao J, Liu GR (2014) A matrix-form GSM–CFD solver for incompressible fluids and its application to hemodynamics. *Comput Mech* 54:999–1012. doi:10.1007/s00466-014-0990-8
  26. Long CC, Marsden AL, Bazilevs Y (2014) Shape optimization of pulsatile ventricular assist devices using FSI to minimize thrombotic risk. *Comput Mech* 54:921–932. doi:10.1007/s00466-013-0967-z
  27. Hsu M-C, Kamensky D, Bazilevs Y, Sacks MS, Hughes TJR (2014) Fluid–structure interaction analysis of bioprosthetic heart valves: significance of arterial wall deformation. *Comput Mech* 54:1055–1071. doi:10.1007/s00466-014-1059-4
  28. Hsu M-C, Kamensky D, Xu F, Kiendl J, Wang C, Wu MCH, Mineroff J, Reali A, Bazilevs Y, Sacks MS (2015) Dynamic and fluid–structure interaction simulations of bioprosthetic heart valves using parametric design with T-splines and fung-type material models. *Comput Mech* 55:1211–1225. doi:10.1007/s00466-015-1166-x
  29. Kamensky D, Hsu M-C, Schilling D, Evans JA, Aggarwal A, Bazilevs Y, Sacks MS, Hughes TJR (2015) An immersed-geometric variational framework for fluid–structure interaction: application to bioprosthetic heart valves. *Comput Methods Appl Mech Eng* 284:1005–1053
  30. Tezduyar TE, Sathe S, Cragin T, Nanna B, Conklin BS, Pausewang J, Schwaab M (2007) Modeling of fluid–structure interactions with the space–time finite elements: arterial fluid mechanics. *Int J Numer Methods Fluids* 54:901–922. doi:10.1002/flid.1443
  31. Tezduyar TE, Sathe S, Schwaab M, Conklin BS (2008) Arterial fluid mechanics modeling with the stabilized space–time fluid–structure interaction technique. *Int J Numer Methods Fluids* 57:601–629. doi:10.1002/flid.1633
  32. Tezduyar TE, Schwaab M, Sathe S (2009) Sequentially-coupled arterial fluid–structure interaction (SCAFSI) technique. *Comput Methods Appl Mech Eng* 198:3524–3533. doi:10.1016/j.cma.2008.05.024
  33. Takizawa K, Christopher J, Tezduyar TE, Sathe S (2010) Space–time finite element computation of arterial fluid–structure interactions with patient-specific data. *Int J Numer Methods Biomed Eng* 26:101–116. doi:10.1002/cnm.1241
  34. Tezduyar TE, Takizawa K, Moorman C, Wright S, Christopher J (2010) Multiscale sequentially-coupled arterial FSI technique. *Comput Mech* 46:17–29. doi:10.1007/s00466-009-0423-2
  35. Takizawa K, Moorman C, Wright S, Christopher J, Tezduyar TE (2010) Wall shear stress calculations in space–time finite element computation of arterial fluid–structure interactions. *Comput Mech* 46:31–41. doi:10.1007/s00466-009-0425-0
  36. Takizawa K, Moorman C, Wright S, Purdue J, McPhail T, Chen PR, Warren J, Tezduyar TE (2011) Patient-specific arterial fluid–structure interaction modeling of cerebral aneurysms. *Int J Numer Methods Fluids* 65:308–323. doi:10.1002/flid.2360
  37. Tezduyar TE, Takizawa K, Brummer T, Chen PR (2011) Space–time fluid–structure interaction modeling of patient-specific cerebral aneurysms. *Int J Numer Methods Biomed Eng* 27:1665–1710. doi:10.1002/cnm.1433
  38. Takizawa K, Brummer T, Tezduyar TE, Chen PR (2012) A comparative study based on patient-specific fluid–structure interaction modeling of cerebral aneurysms. *J Appl Mech* 79:010908. doi:10.1115/1.4005071
  39. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2012) Patient-specific computer modeling of blood flow in cerebral arteries with aneurysm and stent. *Comput Mech* 50:675–686. doi:10.1007/s00466-012-0760-4
  40. Takizawa K, Schjodt K, Puntel A, Kostov N, Tezduyar TE (2013) Patient-specific computational analysis of the influence of a stent on the unsteady flow in cerebral aneurysms. *Comput Mech* 51:1061–1073. doi:10.1007/s00466-012-0790-y
  41. Takizawa K, Bazilevs Y, Tezduyar TE, Long CC, Marsden AL, Schjodt K (2014) ST and ALE-VMS methods for patient-specific cardiovascular fluid mechanics modeling. *Math Models Methods Appl Sci* 24:2437–2486. doi:10.1142/S0218202514500250
  42. Takizawa K, Bazilevs Y, Tezduyar TE, Long CC, Marsden AL, Schjodt K (2014) Patient-specific cardiovascular fluid mechanics analysis with the ST and ALE-VMS methods. In: Idelsohn SR (ed) *Numerical simulations of coupled problems in engineering of computational methods in applied sciences*. Springer, Basel, pp 71–102
  43. Takizawa K, Tezduyar TE, Buscher A, Asada S (2014) Space–time interface-tracking with topology change (ST-TC). *Comput Mech* 54:955–971. doi:10.1007/s00466-013-0935-7
  44. Takizawa K (2014) Computational engineering analysis with the new-generation space–time methods. *Comput Mech* 54:193–211. doi:10.1007/s00466-014-0999-z
  45. Takizawa K, Tezduyar TE, Buscher A, Asada S (2014) Space–time fluid mechanics computation of heart valve models. *Comput Mech* 54:973–986. doi:10.1007/s00466-014-1046-9
  46. Suito H, Takizawa K, Huynh VQH, Sze D, Ueda T (2014) FSI analysis of the blood flow and geometrical characteristics in the thoracic aorta. *Comput Mech* 54:1035–1045. doi:10.1007/s00466-014-1017-1
  47. Takizawa K, Torii R, Takagi H, Tezduyar TE, Xu XY (2014) Coronary arterial dynamics computation with medical-image-based time-dependent anatomical models and element-based zero-stress state estimates. *Comput Mech* 54:1047–1053. doi:10.1007/s00466-014-1049-6
  48. Takizawa K, Tezduyar TE (2012) Space–time fluid–structure interaction methods. *Math Models Methods Appl Sci* 22(supp02):1230001. doi:10.1142/S0218202512300013
  49. Takizawa K, Bazilevs Y, Tezduyar TE, Hsu M-C, Øiseth O, Mathisen KM, Kostov N, McIntyre S (2014) Engineering analysis and design with ALE-VMS and space–time methods. *Arch Comput Methods Eng* 21:481–508. doi:10.1007/s11831-014-9113-0
  50. Takizawa K, Bazilevs Y, Tezduyar TE, Hsu MC, Øiseth O, Mathisen KM, Kostov N, McIntyre S (2014) Computational engineering analysis and design with ALE-VMS and ST methods. In: Idelsohn SR (ed) *Numerical simulations of coupled prob-*

- lems in engineering of computational methods in applied sciences. Springer, Basel
51. Bazilevs Y, Takizawa K, Tezduyar TE (2015) New directions and challenging computations in fluid dynamics modeling with stabilized and multiscale methods. *Math Models Methods Appl Sci* 25:2217–2226. doi:[10.1142/S0218202515020029](https://doi.org/10.1142/S0218202515020029)
  52. Takizawa K, Tezduyar TE (2016) New directions in space–time computational methods. In: Bazilevs Y, Takizawa K (eds) *Advances in computational fluid–structure interaction and flow simulation: new methods and challenging computations, modeling and simulation in science, engineering and technology*. Springer, New York, pp 159–178
  53. Takizawa K, Bazilevs Y, Tezduyar TE (2012) Space–time and ALE–VMS techniques for patient-specific cardiovascular fluid–structure interaction modeling. *Arch Comput Methods Eng* 19:171–225. doi:[10.1007/s11831-012-9071-3](https://doi.org/10.1007/s11831-012-9071-3)
  54. Tezduyar TE, Cragin T, Sathe S, Nanna B (2007) FSI computations in arterial fluid mechanics with estimated zero-pressure arterial geometry. In: Onate E, Garcia J, Bergan P, Kvamsdal T (eds) *Marine 2007*. CIMNE, Barcelona
  55. Takizawa K, Takagi H, Tezduyar TE, Torii R (2014) Estimation of element-based zero-stress state for arterial FSI computations. *Comput Mech* 54:895–910. doi:[10.1007/s00466-013-0919-7](https://doi.org/10.1007/s00466-013-0919-7)
  56. Takizawa K, Tezduyar TE, Sasaki T (2016) Estimation of element-based zero-stress state in arterial FSI computations with isogeometric wall discretization. Springer, New York
  57. Borden MJ, Scott MA, Evans JA, Hughes T (2011) Isogeometric finite element data structures based on Bézier extraction of NURBS. *Int J Numer Methods Eng* 87:15–47