Received: 22 March 2021 | Revised: 28 June 2021 | Accepted: 20 August 2021

DOI: 10.1002/fld.5042

RESEARCH ARTICLE

WILEY

Four-dimensional elastically deformed simplex space-time meshes for domains with time-variant topology

Max von Danwitz[1,2] | **Patrick Antony**[2] | **Fabian Key**[2,3] | **Norbert Hosters**[2] | **Marek Behr**[2]

[1]Institute for Mathematics and Computer-Based Simulation (IMCS), University of the Bundeswehr Munich, Neubiberg, Germany

[2]Chair for Computational Analysis of Technical Systems (CATS), RWTH Aachen University, Aachen, Germany

[3]Institute of Lightweight Design and Structural Biomechanics, TU Wien, Vienna, Austria

Correspondence
Max von Danwitz, Institute for Mathematics and Computer-Based Simulation (IMCS), University of the Bundeswehr Munich, Werner-Heisenberg-Weg 39, D-85577 Neubiberg, Germany.
Email: max.danwitz@unibw.de

[Corrections added on 17 January 2022, after first online publication: Projekt DEAL funding statement has been added.]

Abstract

Considering the flow through biological or engineered valves as an example, there is a variety of applications in which the topology of a fluid domain changes over time. This topology change is characteristic for the physical behavior, but poses a particular challenge in computer simulations. A way to overcome this challenge is to consider the application-specific space-time geometry as a contiguous computational domain. In this work, we obtain a boundary-conforming discretization of the space-time domain with four-dimensional simplex elements (pentatopes). To facilitate the construction of pentatope meshes for complex geometries, the widely used elastic mesh update method is extended to four-dimensional meshes. In the resulting workflow, the topology change is elegantly included in the pentatope mesh and does not require any additional treatment during the simulation. The potential of simplex space-time meshes for domains with time-variant topology is demonstrated in a valve simulation, and a flow simulation inspired by a clamped artery.

KEYWORDS

compressible flow, elastic mesh update, four-dimensional meshes, mesh generation, pentatope discretization, simplex space-time, space-time finite elements, stabilized method, subsonic, topology changes

1 | INTRODUCTION AND PROBLEM DEFINITION

Computer simulations have been widely and successfully used to understand and predict physical behavior in biological, medical, technical, and many other applications. A central entity in simulation technology is the computational domain on which the solution is sought numerically. Commonly, the computational domain coincides with the spatial extent $\Omega \subset \mathbb{R}^3$ of the physical object under consideration. In many applications, this leads to a time-dependent spatial domain $\Omega(t)$ and—thinking for example of valves or bearings—there are also various applications where Ω changes its topology over time.

Several approaches have been developed to handle the time-variant topology of Ω. Our focus is on mesh-based approaches, which can be broadly organized in three groups, namely,

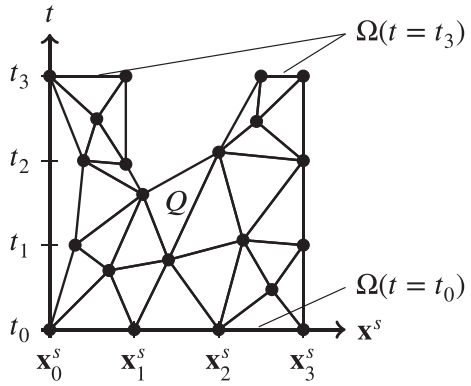


FIGURE 1 Simplex space-time (SST) mesh of spatial computational domain with time-variant topology

- non-boundary-conforming approaches,
- quasi-boundary-conforming approaches with residual gap,
- and boundary-conforming approaches.

Non-boundary-conforming (non-interface-fitted) approaches find a way to shift the complication of the topology change from the discretization to the integration. In the field of fluid-structure-contact interaction (FSCI), a number of methods have recently been developed. For example, Ager et al.[1] employ a fixed Eulerian mesh as fluid discretization and handle boundary motion and topology changes of the physical fluid domain with an approach based on CutFEM.[2] This includes a suitable method for the numerical integration on boundary-intersected elements,[3] a ghost penalty stabilization to overcome issues of very small cuts, and a Nitsche-based weak imposition of fluid boundary (coupling) conditions. In a separate development, Kapada et al.[4] propose a computational framework based on Cartesian hierarchical B-spline grids for the fluid discretization. Therein, integration over intersected elements is handled via sub-triangulation or uniform subdivision of the rectangular grid; a ghost penalty stabilization and Nitsche's method for fluid boundary conditions are used as well. Both aforementioned formulations bear similarities with the immersogeometric variational framework proposed by Kamensky et al.[5]

The second class of approaches uses a boundary-conforming discretization, but avoid the actual topology change of the fluid domain with a small residual gap. Ensuring the residual gap with a penalty force or a displacement restriction in the mesh motion, space-time,[6] or ALE methods[7] are employed to handle the deforming fluid domain.

Finally, boundary-conforming discretizations for spatial computational domains with topology changes can be obtained by reformulating the problem as a space-time problem with contiguous computational domain. For two-dimensional spatial computational domains, this approach has been implemented with unstructured finite volume[8] and finite element meshes.[9]

In this work, we follow the boundary-conforming approach and choose the space-time domain $Q \subset \mathbb{R}^4$ as computational domain. The concept is visualized in Figure 1 with the spatial domain collapsed onto the horizontal axis. Between t_1 and t_2, the spatial domain Ω splits into two unconnected parts. However, the computational space-time domain Q is contiguous. A boundary-conforming discretization with pentatope elements leads to a simplex space-time (SST) mesh, which is fully unstructured in space and time.

To generate such four-dimensional simplex meshes, several similar approaches have been developed recently. The robust meshing strategy proposed by Behr[10] extrudes a tetrahedral mesh resulting in a four-dimensional mesh of tensor product elements. These hyperprisms are split into pentatopes with an element-wise Delaunay triangulation of the perturbed nodal coordinates. In a further development, Karabelas and Neumüller[11] replace the element-wise Delaunay triangulation with a predefined decomposition of each hyperprism requiring a consistently numbered tetrahedral mesh. A third approach of Wang similarly employs a global node indexing scheme and extends it with a node insertion procedure to support local mesh density operations.[12] The above-mentioned strategies have in common that the four-dimensional mesh is based on an extruded tetrahedral mesh. An alternative approach to generate high-quality pentatope meshes could be based on Coxeter triangulations,[13] with the currently addressed issue of generating boundary-conforming meshes.[14]

In most cases, SST meshes are employed to facilitate adaptive mesh refinement in space and time. Suitable pentatope mesh refinement procedures have been explored by Neumüller and Steinbach[15] and Grande.[16] Anisotropic

four-dimensional mesh adaptation is pioneered by Caplan et al.[17,18] and successfully employed in the solution of the advection-diffusion equation.[19] Further, recent application examples of four-dimensional SST meshes from the field of mathematics deal with parabolic evolution problems[20,21] or a broader class of transient PDEs recast as constrained first-order system.[22] In the field of computational engineering science, adaptive temporal refinement of pentatope meshes is used for two-phase flow simulations[23]—also combined with complex material laws such as the Carreau-Yasuda-WLF model[24] or the $\mu(I)$-rheology[25]—as well as gas flow simulations in the piston ring-pack of internal combustion engines.[9]

In this work, we generate pentatope finite element meshes for spatial domains with time-variant topology. Therefore, we combine the extrusion based approach by Behr[10] with a four-dimensional extension of the elastic mesh update method (EMUM). Originally proposed as elastic grid approach by Lynch,[26] EMUM was refined and employed as automatic mesh moving scheme for the deforming spatial domain/stabilized space-time (DSD/SST) finite element formulation.[27] Since then, it has been widely used and become a standard technique to handle moving domains in fluid-structure interaction (FSI) simulations.[28-30] More recent FSI simulations relying on this approach are presented in the work of Spenke et al.,[31] La Spina et al.,[32] and Liu et al.[33] Furthermore, EMUM has been used in the context of free-surface flows[34] and to update finite element meshes according to prescribed boundary displacements.[35] The four-dimensional extension of EMUM (4DEMUM) presented in Section 3 allows us to obtain boundary-conforming pentatope meshes of complex geometries. The geometry may have holes, and does not have to have a tensor product shape in any dimension. This means that the pentatope mesh can account for a time-variant topology of the spatial domain. However, the mesh generation method is still limited to cases where the four-dimensional geometry can be obtained by extrusion of a (complex) three-dimensional geometry with subsequent elastic deformation (in the sense of 4DEMUM).

The flexibility of SST methods to elegantly include spatial topology changes in a contiguous space-time mesh has a certain computational cost. Although, a thorough analysis of the computational complexity of our method is beyond the scope of this work, we briefly address the topic of computational efficiency. In this regard, it is important to note, that SST methods can range from time-discontinuous methods (D-SST) with very few degrees of freedom in time direction to time-continuous methods (C-SST) with a fully coupled time domain. D-SST methods divide the space-time domain into space-time slabs that are then solved in a stepwise manner. Moreover, D-SST methods can be seen as replacement for linear multi-step time-discretization methods with a comparable efficiency. In the time-continuous case (C-SST), all degrees of freedom are connected in a single space-time mesh and are therefore assembled into one large equation system, which must be solved in each iteration of the nonlinear solver.

Previous studies of SST methods[9,10] showed that for our current matrix-based implementation the total simulation time mainly consists of the time to assemble the linear equation system and the time required to solve it. While the assembly time is essentially the same for D-SST and C-SST, the solution time increases when transitioning from D-SST to C-SST. As it is typically cheaper to solve multiple small equation systems instead of one large one, the use of C-SST discretizations decreases the computational efficiency. However, C-SST discretizations also have several advantages that may be exploited to gain computational efficiency in the long run. The larger equation system also offers a way to increase parallelism as a form of parallel-in-time computations. Moreover, C-SST discretizations enable adaptive mesh refinement in the entire space-time domain[36] and mesh-based model order reduction (MOR) techniques for transient problems. Another promising approach to improve the computational efficiency of the method would be a matrix-free reimplementation of the solution algorithm. See Reference 37 for implementation details of an efficient matrix-free compressible Navier–Stokes solver for quadrilateral and hexahedral elements. With the current implementation, we recommend a discretization between the two extremes (D-SST and C-SST) for large applications with topology changes. Slabs must be chosen large enough to include the topology change, yet as small as possible to keep the linear equation systems small and ensure a reasonable computational efficiency.

Now, we proceed as follows. Section 2 presents a workflow for finite element simulations on pentatope meshes. In Section 3, we describe the strong and weak form for a finite element implementation of 4DEMUM. Then, Section 4 aims at experimentally validating the method, as well as demonstrating the particular potential of simulations on SST meshes for domains with time-variant topology. Finally, Section 5 contains concluding remarks and a brief outlook.

2 | SIMULATION WORK FLOW

To perform simulations on pentatope meshes, we follow the workflow given in Figure 2. The arrows correspond to eight steps that are taken, with each step completing a specific subtask in the workflow. When suitable, the arrows are labeled with the software employed to complete the task. The first four steps are considered as preprocessing, and produce the

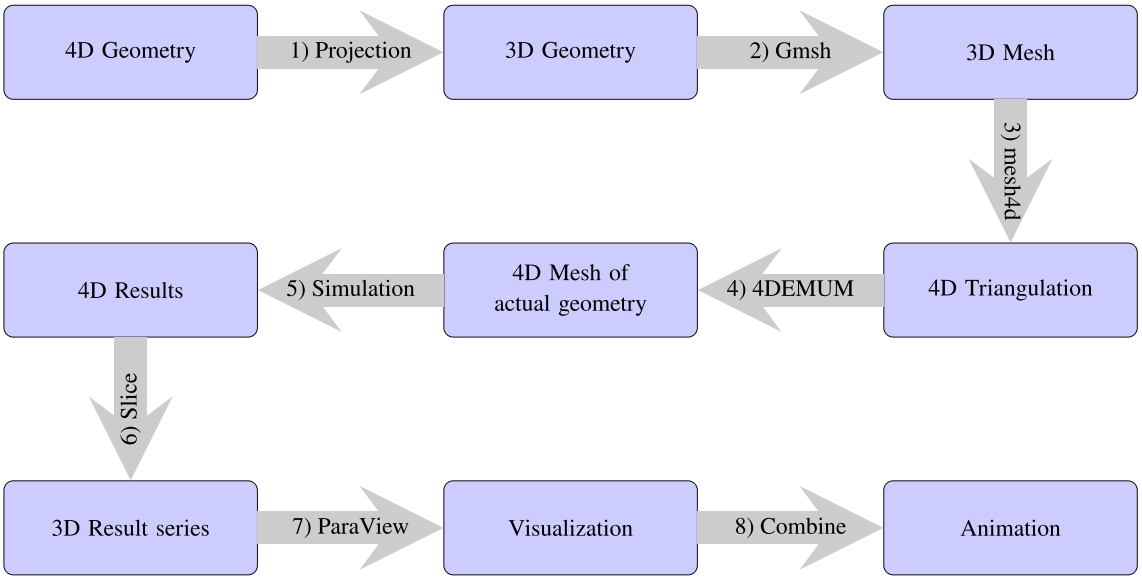


FIGURE 2 Workflow for four-dimensional SST simulations with topology change [Colour figure can be viewed at wileyonlinelibrary.com]

SST meshes covering the considered physical space-time domain Q. In detail, they are: First, identify a projection $\Omega_\# = \mathbf{p}(Q) \subset \mathbb{R}^3$, such that complex features of Q are included in $\Omega_\#$. In our work, this step is performed manually. Examples of such projections will be given in Section 4. As a second step, generate a tetrahedral mesh covering $\Omega_\#$, for example using Gmsh.[38] Then, following Behr,[10] extrude the mesh covering $\Omega_\#$ over an interval $I \subset \mathbb{R}$ resulting in a mesh of tensor-product elements, and apply an element-wise Delaunay triangulation to generate a pentatope mesh covering $Q_\# = \Omega_\# \times I$ (third step). The fourth step is to use 4DEMUM as described in Section 3 to deform the pentatope mesh to cover Q. Note that in contrast to previous work, the time dimension can be one of the initial three dimensions, and extrusion can be used to generate one of the spatial dimensions of the mesh.

With a boundary conforming discretization at hand, the fifth step is to perform the space-time simulation. Details of the simulation depend on the application example and are given in Section 4. In any case, the solution of the discrete problem is obtained on the unstructured space-time mesh. To visualize the results over time, a series of tetrahedral meshes covering the spatial domain at given time instances is generated. The node positions of these meshes are passed as query points to an efficient interpolation tool.[39] The tool identifies the pentatope of the space-time mesh which contains the query point and performs a linear barycentric interpolation on the element (step 6). Note that this interpolation is in accordance with the finite element approximation of the simulation. For domains with convex boundaries, some of the query points can be located slightly outside of the space-time mesh. In such cases, the solution is extrapolated from the element with the closest center. In step 7, the data on the tetrahedral meshes can be easily visualized, analyzed, and rendered with available tools, such as ParaView.[40] If desired, renderings of the result series can be combined in an animation with video editing software. An alternative visualization approach described by Karabelas and Neumüller[11] is based on the element-wise intersection of the pentatope mesh with a hyperplane.

3 | FOUR-DIMENSIONAL ELASTIC MESH UPDATE METHOD

The basic idea of the elastic mesh update method is to move the node positions according to the deformation of an elastic solid, while keeping the mesh connectivity unchanged. As outlined above, the method has been widely used to adjust two- or three-dimensional meshes to deforming boundaries. In the following, we extend the method to four-dimensional meshes.

Therefore, we consider a virtual four-dimensional linear elastic solid occupying the region $Q_\# \subset \mathbb{R}^4$. The assumption of a homogeneous, isotropic, linear elastic material behavior leads to a constitutive equation

$$\boldsymbol{\sigma} = \lambda \left[\mathrm{tr}\,(\boldsymbol{\varepsilon})\right] \mathbf{I} + 2\mu\boldsymbol{\varepsilon}, \tag{1}$$

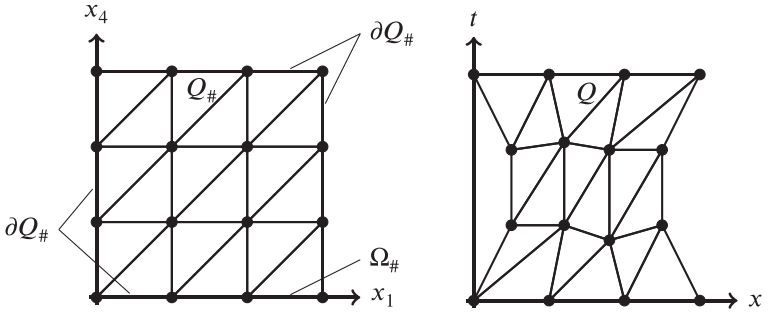


FIGURE 3 Four-dimensional elastic mesh update method

where $\boldsymbol{\sigma}$ is an equivalent of the Cauchy stress tensor and λ and μ correspond to the Lamé parameters. In this work, we choose $\lambda = \mu = 1$. Further, $\mathbf{I} \in \mathbb{R}^{4\times4}$ denotes the identity matrix. The trace $tr(\boldsymbol{\varepsilon})$ is the sum of the main diagonal entries of the strain tensor $\boldsymbol{\varepsilon}$. More precisely, the solid's deformation is measured with a linear strain model

$$\boldsymbol{\varepsilon} = \frac{1}{2}\left[\boldsymbol{\nabla}_\# \mathbf{d} + (\boldsymbol{\nabla}_\# \mathbf{d})^T\right]. \tag{2}$$

Therein, the vector $\mathbf{d}(\mathbf{x}_\#) \in \mathbb{R}^4$ holds the displacements in the four spatial directions. The displacements are considered to be functions of the coordinates $\mathbf{x}_\# = (x_1, x_2, x_3, x_4)^T \in Q_\#$. Accordingly, the gradient operator

$$\boldsymbol{\nabla}_\# = \left[\frac{\partial(.)}{\partial x_1}, \frac{\partial(.)}{\partial x_2}, \frac{\partial(.)}{\partial x_3}, \frac{\partial(.)}{\partial x_4}\right]^T \tag{3}$$

collects the respective partial derivatives. Choosing displacements as primary degrees of freedom in our model, the governing equation on the computational domain $Q_\#$ reads

$$\boldsymbol{\nabla}_\# \cdot \boldsymbol{\sigma}(\mathbf{d}) = \mathbf{0}, \quad \text{on} \quad Q_\#. \tag{4}$$

On the left of Figure 3, two dimensions of an example of $Q_\#$ are displayed. The domain boundary is denoted by $\partial Q_\#$. Let us assume that for each displacement degree of freedom d_i the domain boundary is split (individually) such that $\partial Q_\# = \partial Q_\#^D \cup \partial Q_\#^N$ and $\partial Q_\#^D \cap \partial Q_\#^N = \emptyset$. This allows to apply Dirichlet boundary conditions component-wise as

$$d_i = g_i \quad \text{on} \quad \partial Q_{\#\,i}^D \quad \text{for } i = 1, \dots, 4. \tag{5}$$

On boundary parts where no Dirichlet values are prescribed for one or more components, a homogeneous Neumann boundary condition is assumed for these components. Such a Neumann boundary condition allows the mesh nodes to slip freely along the boundary in directions orthogonal to the directions for which displacements are prescribed. This means no exterior tractions are influencing the node displacements. See Reference 41 for further details of a slip boundary condition with zero tangent stress. The combination of Equations (4) and (5) is the strong form of the continuous four-dimensional elastostatic problem. In case of coupled simulations, multiple mesh update steps may be performed, yet, in each step, a steady problem is solved.

Based on the extruded mesh covering $Q_\#$, we approximate the displacements with the vector-valued functions

$$\mathbf{d}^h \in \left[H_h^1\right]^4, \tag{6}$$

where H_h^1 is a H^1-conformal finite element approximation space based on linear basis functions. Furthermore, we consider a suitable representation g^h of the Dirichlet boundary data g_i and a projection $\mathbf{P}$ that selects from $\mathbf{d}^h$ the degrees of freedom which are fixed by Dirichlet boundary conditions. Subsequently, the trial and test function spaces read

$$\mathcal{S}_h = \left\{\mathbf{d}^h \in \left[H_h^1\right]^4 \;\middle|\; \mathbf{P}\mathbf{d}^h = g^h\right\}, \qquad \mathcal{V}_h = \left\{\mathbf{w}^h \in \left[H_h^1\right]^4 \;\middle|\; \mathbf{P}\mathbf{w}^h = \mathbf{0}\right\}. \tag{7}$$

Using the definitions above, a discretized weak form of the elastostatic problem can be stated as: Find $\mathbf{d}^h \in \mathcal{S}_h$ such that for all $\mathbf{w}^h \in \mathcal{V}_h$

$$0 = \int_{Q_\#} \nabla_\# \mathbf{w}^h \ : \ \boldsymbol{\sigma}\left(\mathbf{d}^h\right) \, dQ = \int_{Q_\#} \nabla_\# \mathbf{w}^h \ : \left\{ \lambda \mathrm{tr}\left(\nabla_\# \mathbf{d}^h\right) \mathbf{I} + \mu \left[\nabla_\# \mathbf{d}^h + \left(\nabla_\# \mathbf{d}^h\right)^T \right] \right\} dQ. \tag{8}$$

As usual, the divergence theorem was used (here in combination with homogeneous Neumann boundary conditions) to reduce the smoothness required of $\mathbf{d}^h$ in Equation (8) in comparison to Equation (4). The above weak form results in a linear equation system that can be solved to obtain the displacements $\mathbf{d}^h$. Finally, the new position of a node with coordinates $\mathbf{x} = [x, y, z, t]^T \in Q$ is obtained by adding the displacements $\mathbf{d}^h(\mathbf{x}_\#)$ to the original node position $\mathbf{x}_\#$ in the extruded mesh

$$\mathbf{x} = \mathbf{x}_\# + \mathbf{d}^h(\mathbf{x}_\#). \tag{9}$$

In this equation, we assign $[x_1, x_2, x_3, x_4]^T$ to $[x, y, z, t]^T$. Two dimensions of a resulting mesh are shown in Figure 3 on the right. Note that the elastic mesh update couples the displacements in all four directions, such that interior mesh nodes can move in any direction, even if nonzero boundary displacements are applied only to displacements in specific directions. Furthermore, we want to remark, that from a modeling point of view, the numerical values for the parameters λ and μ can be chosen independently as the method does not attempt to model a physical material, but rather considers a virtual solid to perform a mesh update.

4 | APPLICATION EXAMPLES

For the physical simulations in the following application examples, we distinguish between space and time coordinates. The spatial gradient operator, $\nabla_\mathbf{x} = \frac{\partial(\cdot)}{\partial \mathbf{x}^s} = \left[\frac{\partial(.)}{\partial x}, \frac{\partial(.)}{\partial y}, \frac{\partial(.)}{\partial z}\right]^T$, collects the partial derivatives with respect to the three spatial coordinates. In the following two examples, we consider the compressible Navier–Stokes equations

$$\frac{\partial \rho}{\partial t} + \nabla_\mathbf{x} \cdot (\rho \mathbf{u}) = 0 \quad \text{on } Q, \tag{10}$$

$$\frac{\partial (\rho \mathbf{u})}{\partial t} + \nabla_\mathbf{x} \cdot [(\rho \mathbf{u}) \otimes \mathbf{u}] + \nabla_\mathbf{x} \, p - \nabla_\mathbf{x} \cdot \boldsymbol{\tau} = \mathbf{0} \quad \text{on } Q, \tag{11}$$

$$\frac{\partial (\rho e)}{\partial t} + \nabla_\mathbf{x} \cdot (\rho e \mathbf{u}) + \nabla_\mathbf{x} \cdot (p \mathbf{u}) - \nabla_\mathbf{x} \cdot (\boldsymbol{\tau} \mathbf{u}) + \nabla_\mathbf{x} \cdot \mathbf{q} = 0 \quad \text{on } Q. \tag{12}$$

Here $\rho(\mathbf{x}^s, t)$, $\mathbf{u}(\mathbf{x}^s, t)$, $p(\mathbf{x}^s, t)$, $\boldsymbol{\tau}(\mathbf{x}^s, t)$, $e(\mathbf{x}^s, t)$, and $\mathbf{q}(\mathbf{x}^s, t)$ are density, velocity vector, pressure, viscous stress tensor, total energy per unit mass, and heat flux vector, respectively. For convenience, the conservation variables are collected in the vector $\mathbf{U} = [\rho, \rho\mathbf{u}, \rho e]^T$. We consider an ideal, calorically perfect gas, with the specific gas constant $R = 287$ J/(kg K), the ratio of specific heats $\gamma = 1.4$, and a Prandtl number of $\mathrm{Pr} = \frac{\nu}{\kappa} \frac{\gamma R}{\gamma - 1} = 0.71$. With these assumptions, we can perform a change of variables from the conservation variables $\mathbf{U} = \mathbf{U}(\mathbf{Y})$ to the pressure-primitive variables $\mathbf{Y} = \left[p, \mathbf{u}, T\right]^T$. The motivation to use pressure-primitive variables as primary unknowns is two-fold. First, formulations based on pressure-primitive variables behave well for flow conditions approaching the incompressible limit[42] and second, they are very convenient to prescribe boundary conditions since given pressure values can be directly applied as Dirichlet boundary condition.[9]

Based in the pressure-primitive variables, the governing equations above are formulated as generalized advective-diffusive system

$$\mathbf{Res}(\mathbf{Y}) \ := \mathbf{A}_0 \mathbf{Y}_{,t} + \left(\mathbf{A}_i^{\mathrm{adv\setminus p}} + \mathbf{A}_i^{\mathrm{p}} + \mathbf{A}_i^{\mathrm{sp}} \right) \mathbf{Y}_{,i} - (\mathbf{K}_{ij} \mathbf{Y}_{,j})_{,i} = \mathbf{0}, \quad i, j = 1, \ \ldots \ , n_{\mathrm{sd}}. \tag{13}$$

Therein, partial time derivatives are denoted with $(\cdot)_{,t}$, partial derivatives in each of the $n_{\mathrm{sd}} = 3$ spatial directions are denoted with $(\cdot)_{,i}$, and the Einstein summation convention applies to repeated indices. Details of the generalized advection matrices $\mathbf{A}$ and diffusion matrices $\mathbf{K}$ can be found in earlier work.[9,43]

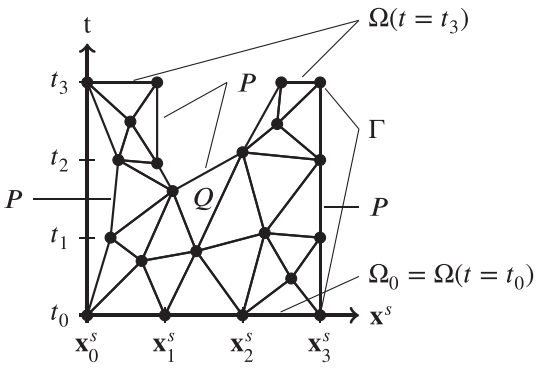


FIGURE 4 Physical space-time domain and labeled boundary regions

To find a numerical solution to the compressible Navier–Stokes equations with space-time finite elements, a discretization of the physical space-time domain $Q \subset \mathbb{R}^{n_{\text{sd}}+1}$ is required. An example of such a discretization can be seen in Figure 4. In practice, a procedure is required to numerically integrate over Q, over the spatial computational domain $\Omega_0 \subset \mathbb{R}^{n_{\text{sd}}}$ at the initial time t_0, and over P which is the temporal evolution of the spatial domain boundary $\Gamma \subset \mathbb{R}^{n_{\text{sd}}-1}$. For each degree of freedom, the space-time boundary P consist of non-overlapping Dirichlet regions P_D and Neumann regions P_N.

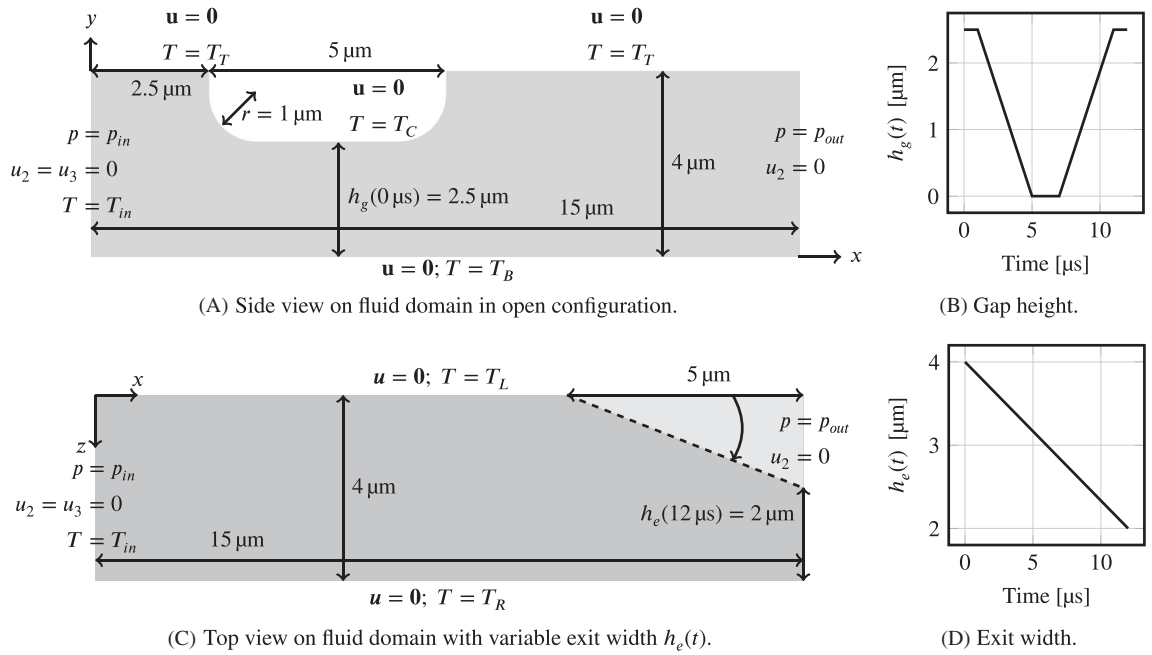
Based on the domains introduced above and for admissible test and trial function spaces V_h and S_h, the weak form of Equation (13) can be stated as: For given initial conditions $\mathbf{Y}_0^h$, find $\mathbf{Y}^h \in S_h$ such that for all $\mathbf{W}^h \in V_h$

$$
\begin{aligned}
0 = &\int_Q \mathbf{W}^h \cdot \left[\mathbf{A}_0 \mathbf{Y}^h_{,t} + \left(\mathbf{A}_i^{\text{adv\textbackslash p}} + \mathbf{A}_i^{\text{sp}}\right) \mathbf{Y}^h_{,i}\right] dQ \\
&+ \int_Q \mathbf{W}^h_{,i} \cdot \left[\mathbf{K}_{ij} \mathbf{Y}^h_{,j} - \mathbf{A}_i^{\text{p}}\, \mathbf{Y}^h\right] dQ \\
&- \int_{P_N} \mathbf{W}^h \cdot \mathbf{H}(\mathbf{Y}^h)\, dP \\
&+ \int_{\Omega_0} \mathbf{W}^h \cdot \left[\mathbf{U}(\mathbf{Y}^h) - \mathbf{U}(\mathbf{Y}_0^h)\right] d\Omega \\
&+ \int_Q \left[(\hat{\mathbf{A}}_m)^T \mathbf{W}^h_{,m}\right] \cdot \boldsymbol{\tau}_{\text{SUPG}}\, \mathbf{Res}(\mathbf{Y}^h)\, dQ, \\
&i,j = 1, \ldots, n_{\text{sd}}, \quad m = 1, \ldots, n_{\text{sd}} + 1.
\end{aligned} \tag{14}
$$

The first integral collects terms of the residual that are multiplied with the test function vector and integrated over Q. In the second integral, the derivatives with respect to the spatial coordinates are shifted to the test function vector, leading to the third integral over the Neumann part of the space-time boundary P_N. Therein, the boundary normal fluxes $\mathbf{H}$ are evaluated with the outwards pointing surface normal $\mathbf{n} = [n_1, n_2, n_3]^T$ as

$$
\mathbf{H} = \begin{bmatrix} 0 \\ -pn_1 + \tau_{1i} n_i \\ -pn_2 + \tau_{2i} n_i \\ -pn_3 + \tau_{3i} n_i \\ -q_i n_i \end{bmatrix}. \tag{15}
$$

The initial condition $\mathbf{Y}_0^h$ is weakly enforced with the integral over Ω_0. The weak form is completed with a SUPG operator to overcome instabilities of the pure Galerkin formulation which occur in convection-dominated flow simulations. For more details on the space-time finite element scheme for the compressible Navier–Stokes equations, the reader is referred to the earlier publication.[9]



(A) Side view on fluid domain in open configuration.

(B) Gap height.

(C) Top view on fluid domain with variable exit width $h_e(t)$.

(D) Exit width.

FIGURE 5 Valve test case. Fluid domain geometry specifications and boundary conditions

TABLE 1 Valve test case

T_{in} [K]	T_T [K]	T_C [K]	T_B [K]	$T_L = T_R$ [K]	p_{in} [Pa]	p_{out} [Pa]
373.15	383.15	403.15	393.15	$T_B - \frac{10.0y}{4\ \mu m}$	1.3×10^5	1.0×10^5

Note: Boundary conditions.

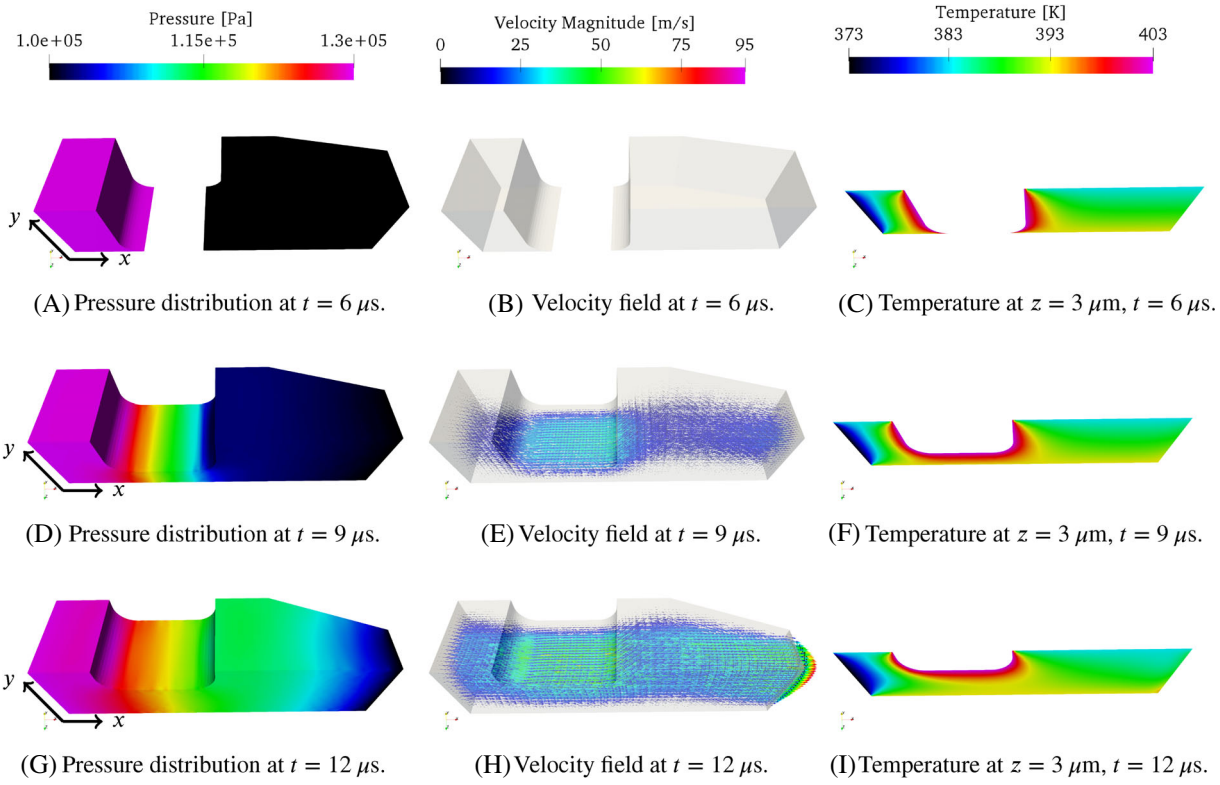
4.1 | Transient gas flow through valve

In the following, we present a transient three-dimensional simulation of gas flow through a valve. The fluid domain consist of a square channel with 4 µm sides in the *y*-*z*-plane and a length of 15 µm in *x*-direction. In the course of the simulation, the position of the rounded valve clamp determines the topology of the spatial computational domain (see Figure 5A). During the valve cycle, the clamp is first lowered and later lifted again. This leads to a splitting of the fluid domain at 5 µs and a reconnection at 7 µs. Additionally, the exit cross-section of the channel deforms from the initial square geometry into a rectangular cross-section with half the size. Detailed geometry specifications and boundary conditions are collected in Figure 5. Along the solid walls, no-slip boundary conditions are enforced, that is, the velocity is set to zero and a wall temperature is prescribed. On the left open boundary, the pressure and temperature are given; on the right open boundary a pressure value is set. Prescribed temperature and pressure values are summarized in Table 1. The gas viscosity is modeled using Sutherland's relation

$$\nu = \nu_{\text{ref}} \frac{T_{\text{ref}} + C}{T + C} \left(\frac{T}{T_{\text{ref}}} \right)^{\frac{3}{2}}, \tag{16}$$

with $\nu_{\text{ref}} = 21.7 \times 10^{-6}$ Pa s, $T_{\text{ref}} = 373.15$ K, $C = 120$ K.

Following the workflow outlined in Section 2, we start with a projection of four-dimensional space-time geometry of the valve into the *x*-*y*-*t*-hyperplane. The resulting three-dimensional *x*-*y*-*t*-domain corresponds to the temporal evolution of the side view shown in Figure 5A and includes the topology change of the flow domain. We discretize the *x*-*y*-*t*-domain with 177,708 tetrahedral elements. This is sufficient to resolve a transient two-dimensional flow field.[9] In the next step, the mesh is extruded in *z*-direction and triangulated with 14,216,640 pentatopes, such that both sides of the square channel cross-section are split into 20 line elements. To account for the closing exit, we identify $[x, y, z, t]^T$ with $[x_1, x_2, x_3, x_4]^T$ and



(A) Pressure distribution at $t = 6\ \mu$s. (B) Velocity field at $t = 6\ \mu$s. (C) Temperature at $z = 3\ \mu$m, $t = 6\ \mu$s.

(D) Pressure distribution at $t = 9\ \mu$s. (E) Velocity field at $t = 9\ \mu$s. (F) Temperature at $z = 3\ \mu$m, $t = 9\ \mu$s.

(G) Pressure distribution at $t = 12\ \mu$s. (H) Velocity field at $t = 12\ \mu$s. (I) Temperature at $z = 3\ \mu$m, $t = 12\ \mu$s.

FIGURE 6 Pressure, velocity, and temperature in the closed, half open, and fully opened valve [Colour figure can be viewed at wileyonlinelibrary.com]

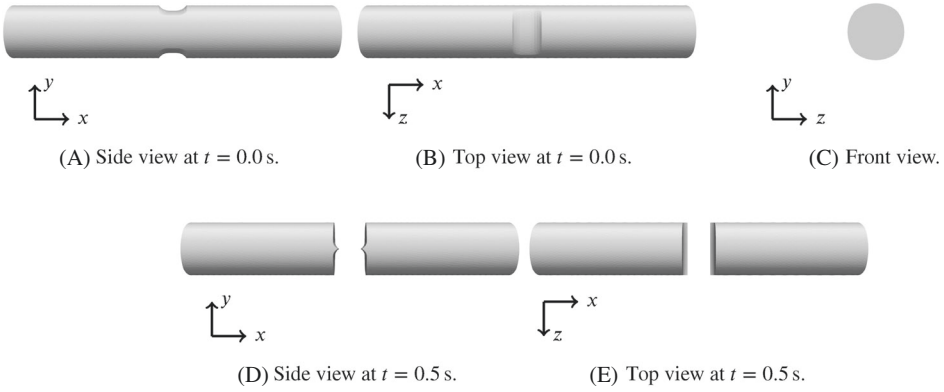
apply 4DEMUM with

$$g^h = \begin{bmatrix} 0 \\ 0 \\ 2\ \mu\text{m} \\ 0 \end{bmatrix} \cdot \mathcal{H}\left(x_1 - 10\ \mu\text{m}\right) \left(\frac{x_1}{5\ \mu\text{m}} - 2\right) \cdot \left(1 - \frac{x_3}{4\ \mu\text{m}}\right) \cdot \frac{x_4}{12\ \mu\text{m}} \tag{17}$$

prescribed as Dirichlet boundary condition on the domain boundary. Therein, $\mathcal{H}(x)$ denotes the Heaviside function. After the mesh update, the fourth dimension of the resulting mesh is interpreted as time and the SST flow simulation is performed. The simulation took 30 min of wall-clock time on 240 processing elements (cores of the RWTH compute cluster CLAIX 2018) using a distributed memory parallelization based on MPI.

The results displayed in Figure 6 show the pressure, velocity, and temperature distribution in the closed, half open, and fully opened valve. At $t = 6$ µs, the valve clamp divides the fluid domain into two parts with distinct pressure values (Figure 6A). In consequence, there are no significant flow velocities in the entire computational domain (Figure 6B). When the valve is open, a gas flow from the high pressure inlet to the low pressure outlet is observed (Figure 6E). For the half open configuration (Figure 6D), the pressure gradient is concentrated below the valve clamp. In contrast, the pressure variation is evenly distributed along the channel axis in x-direction for the fully opened valve (Figure 6G). Additional to the valve clamp motion, the outlet cross-section is reduced in the course of the simulation. In Figure 6H, it can be clearly seen that the reduced cross-section of the closing exit accelerates the flow to the maximum velocity of 94 m/s. The temperature field on the plane $z = 3$ µm is shown in Figure 6C,F,I. In Figure 6I, the influence of the flow on the temperature distribution can be seen in form of a convective heat transport in flow direction. At $t = 12$ µs, we observe a laminar flow with maximum Reynolds and Mach numbers

$$\text{Re} = \frac{\rho\, u_{\max}\, h_e(12\mu\text{s})}{\mu_{\text{ref}}} \approx 10 \quad \text{and} \quad \text{Ma} = \frac{u_{\max}}{\sqrt{\gamma R T}} \approx 0.24. \tag{18}$$



(A) Side view at $t = 0.0$ s. (B) Top view at $t = 0.0$ s. (C) Front view.

(D) Side view at $t = 0.5$ s. (E) Top view at $t = 0.5$ s.

FIGURE 7 Geometry of clamped artery test case

TABLE 2 Clamped artery test case

T_{in} [K]	T_{out} [K]	T_W [K]	p_{in} [Pa]	p_{out} [Pa]	ν [Pa s]
310.15	310.15	$307.15 + 3 \cdot \frac{\lvert x \rvert}{3\ \text{cm}\rvert}$	1.173×10^5	1.12×10^5	4×10^{-3}

Note: Pressure and temperature boundary conditions, viscosity.

The density variations between 0.89 and 1.2 kg/m^3 underline the necessity to consider compressibility effects in this test case.

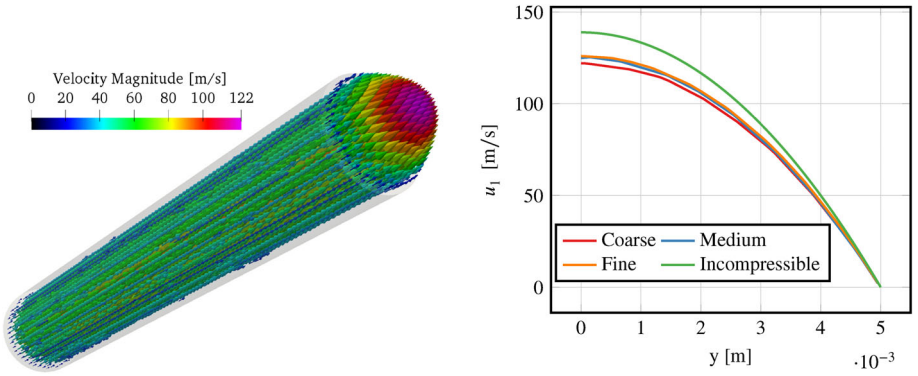
4.2 | Flow inspired by clamped artery

As further application example, we simulate a transient flow through an artery which is temporarily sealed by a clamp and reopened. See Figure 7 for several views on the problem geometry in initial and clamped configuration. The considered artery section is 6 cm long and has an approximately circular cross-section with a diameter of 1 cm. We select an approximately circular cross-section to facilitate the use of 4DEMUM (see also Section 4.2.2). The clamp center is located at $x = 0$ cm. Regarding the fluid domain, we assume that the artery volume is displaced by the clamp and returns to the initial approximately circular shape as the clamp is removed. As time frame for the closing and opening, we choose the duration of one cardiac cycle approximated by 1 s. For the first 0.2 s, the artery is in its initial shape. Over the next 0.2 s, the clamp is applied and seals the artery from 0.4 s until 0.6 s. From 0.6 s until 0.8 s, the artery is reopened and for the last 0.2 s it is again in its initial shape.

The flow through the deforming domain is driven by a pressure gradient of 40 mmHg, which corresponds to the difference between the minimum and maximum aortic pressure during a cardiac cycle. On the left open boundary, a pressure $p_{in} = 1\,\text{atm} + 120\,\text{mmHg} \approx 1.173 \times 10^5$ Pa is prescribed. On the right open boundary, a pressure $p_{out} = 1\,\text{atm} + 80\,\text{mmHg} \approx 1.12 \times 10^5$ Pa is prescribed. We assume that the fluid enters the domain at normal body temperature $T_{in} = 310.15$ K. The colder clamp leads to a temperature variation of 3 K along the artery wall. The wall temperature is denoted by T_W. A summary of the corresponding boundary conditions is given in Table 2. Regarding the velocity degrees of freedom, we set the tangential velocity components on the open boundaries to zero, as well as the velocity degrees of freedom on the arterial wall. A constant fluid viscosity of $\nu = 0.04$ poise $= 4 \times 10^{-3}$ P s is used in this test case. The gas constant, ratio of specific heats and Prandtl number are chosen as in the previous example, leading to a fictitious fluid with a density of roughly 1 kg/m^3. In Sections 4.2.1 and 4.2.2, we proceed with two preparatory examples that lead to the simulation of the application case presented in Section 4.2.3. The preparatory examples do not require a four-dimensional space-time discretization. They can also be solved by conventional methods with a separate discretization of space and time domain.

4.2.1 | Steady pipe simulation

To validate our compressible Navier–Stokes finite element formulation for these settings, we perform—in a first step—a flow simulation through a straight circular duct. The results are collected in Figure 8. The velocity distribution is visualized



(A) Velocity distribution in circular pipe.

(B) Velocity in axial direction at $x = 2\,\text{cm}$, $z = 0\,\text{cm}$.

FIGURE 8 Compressible Poiseuille flow in circular pipe [Colour figure can be viewed at wileyonlinelibrary.com]

with glyphs showing a paraboloid in qualitative agreement with the Poiseuille paraboloid obtained for an incompressible Hagen–Poiseuille flow. Note that neither on the inflow nor on the outflow boundary the velocity component in normal direction is prescribed. For the pressure driven incompressible flow through a straight circular pipe, there is the well-known analytical solution for the velocity component in axial direction u_a,[44] which reads

$$u_a = -\frac{1}{4\mu}\frac{\partial p}{\partial x}\left(r_0^2 - r^2\right) = 138.9\ \text{m/s}\left(1 - \left(\frac{r}{5\ \text{mm}}\right)^2\right) \tag{19}$$

for the considered configuration with $r_0 = 5\,\text{mm}$ and $r^2 = y^2 + z^2$.

After a first simulation on a coarse mesh with 111,744 tetrahedral elements, the simulation is repeated on a medium and fine mesh, with 206,568 and 652,090 elements, respectively. Figure 8B shows the velocity profiles at $x = 2\,\text{cm}, z = 0\,\text{cm}$ in comparison to u_a. One can observe that the compressible flow solutions are flatter than the incompressible solution, but follow a similar progression. Focusing on the numerical solutions, only slight variations of the centerline axial velocity are observed—122.03 m/s on the coarse mesh (3.1% smaller than fine), 124.91 m/s on the medium mesh (0.8% smaller than fine), and 125.98 m/s on the fine mesh. The small changes indicate that the coarse mesh resolution is sufficient to obtain a solution within the range of engineering accuracy.

4.2.2 | Transient simulation of straight channel

As second step, we perform a transient simulation on a straight duct with approximately circular cross-section. Starting point for the SST mesh generation is an unstructured tetrahedral x-y-t-mesh. The temporal resolution of the mesh is comparable to the one shown in Figure 12, yet the influence of the clamp is excluded for now. Next, the tetrahedral mesh is extruded in z-direction, such that the cross-section in the y-z-plane forms a square. In z-direction, 14 nodes are added during the extrusion, such that the spatial resolution of the resulting mesh is comparable to the coarse pipe mesh discussed above.

To obtain the approximately circular cross-section, we perform the 4DEMUM with the space-time coordinates $[x, y, z, t]^T$ identified with $[x_1, x_2, x_3, x_4]^T$. Further, the extruded mesh is shifted and scaled, such that $x_1 \in [-6, 6]$, $x_2 \in [-1, 1]$, $x_3 \in [-1, 1]$, $x_4 \in [0, 6]$. On the pentatope mesh boundary $\partial Q^D_{\#}$, we prescribe the displacements

$$g^h = 0.9\begin{bmatrix} 0 \\ \left(\sqrt{1 - \frac{1}{2}x_3^2} - 1\right)x_2 \\ \left(\sqrt{1 - \frac{1}{2}x_2^2} - 1\right)x_3 \\ 0 \end{bmatrix}. \tag{20}$$

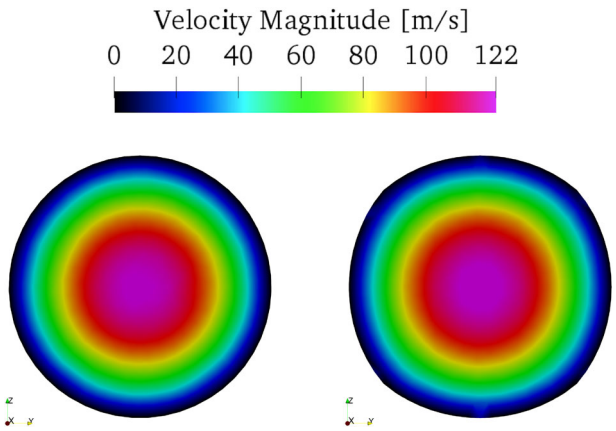
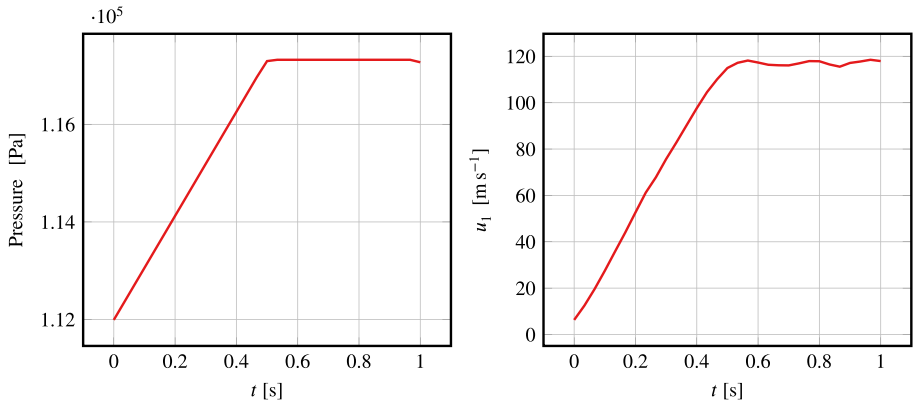


FIGURE 9 Velocity field at $x = 3$ cm of steady circular pipe simulation (left) and transient approximately circular pipe simulation at $t = 0.9$ s (right) [Colour figure can be viewed at wileyonlinelibrary.com]



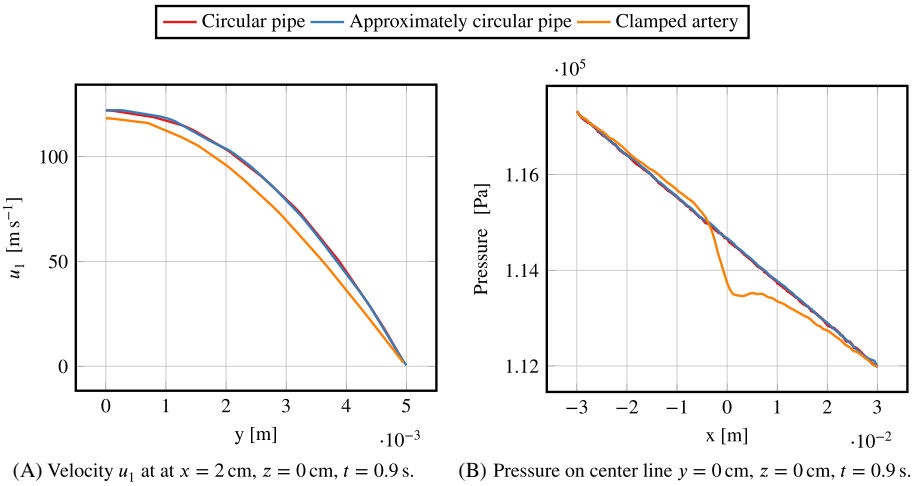
(A) Prescribed inflow pressure. (B) Velocity evolution at $x = -3$ cm, $y = 0$ cm, $z = 0$ cm.

FIGURE 10 Transient pressure-driven compressible flow in approximately circular pipe [Colour figure can be viewed at wileyonlinelibrary.com]

This boundary condition g^h maps the square cross-section in the x_2-x_3-plane onto an approximately circular shape (see Figure 9 on the right). In a final step before the transient simulation, the mesh is shifted and rescaled such that $x \in [-3\,\text{cm}, 3\,\text{cm}]$, $y \in [-0.5\,\text{cm}, 0.5\,\text{cm}]$, $z \in [-0.5\,\text{cm}, 0.5\,\text{cm}]$, $t \in [0\,\text{s}, 1\,\text{s}]$. The shifts are performed to allow for relatively simple expressions in the boundary conditions of 4DEMUM, and at the same time allow the fluid simulation to start at $t = 0$.

In this second simulation, a transient feature is introduced by ramping up the inflow pressure from the initial value p_{out} to p_{in} over the first 0.5 s. For the subsequent 0.5 s, the pressure value is kept constant as shown in Figure 10A. The flow velocity at the center of the inflow (Figure 10B) closely follows the temporal evolution of the pressure value. Note that the computed flow velocity is slightly larger than zero at $t = 0$, which is in line with the weakly enforced initial condition in our formulation (Equation 14). Overall, the transient nature of the problem is properly captured in the computed flow field.

With this test case, we also want to explain our choice of an approximately circular cross-section. A perfectly circular cross-section introduces an expected complication in 4DEMUM, that is, the elements formerly in the corners of the square cross-section attain very large dihedral angles and eventually lead to an invalid mesh. We want to point out, that this is a characteristic of the elastic mesh update method independent of the mesh dimension. Also, in case of two- and three-dimensional meshes, elements formerly in the corners of a square cross-section attain large dihedral angles, when the mesh is deformed into a mesh with a circular cross-section. The complication is avoided by introducing the prefactor 0.9 in Equation (20) to obtain a mesh with an approximately circular cross section in the y-z-plane.



(A) Velocity u_1 at at $x = 2\,\text{cm}$, $z = 0\,\text{cm}$, $t = 0.9\,\text{s}$. (B) Pressure on center line $y = 0\,\text{cm}$, $z = 0\,\text{cm}$, $t = 0.9\,\text{s}$.

FIGURE 11 Comparison between circular pipe, approximately circular pipe, and artery geometry [Colour figure can be viewed at wileyonlinelibrary.com]

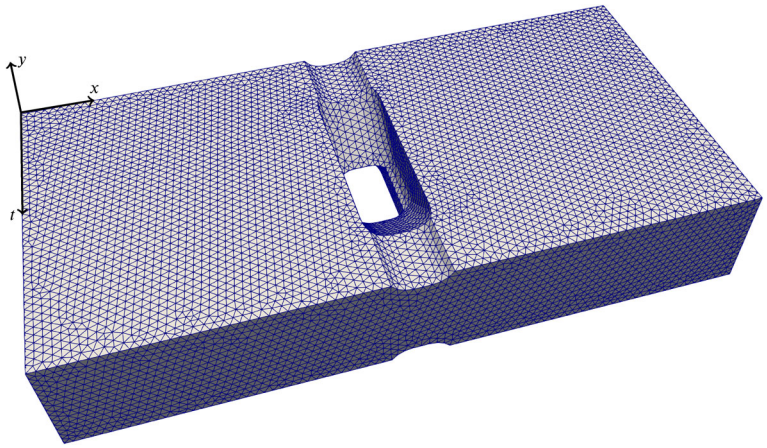


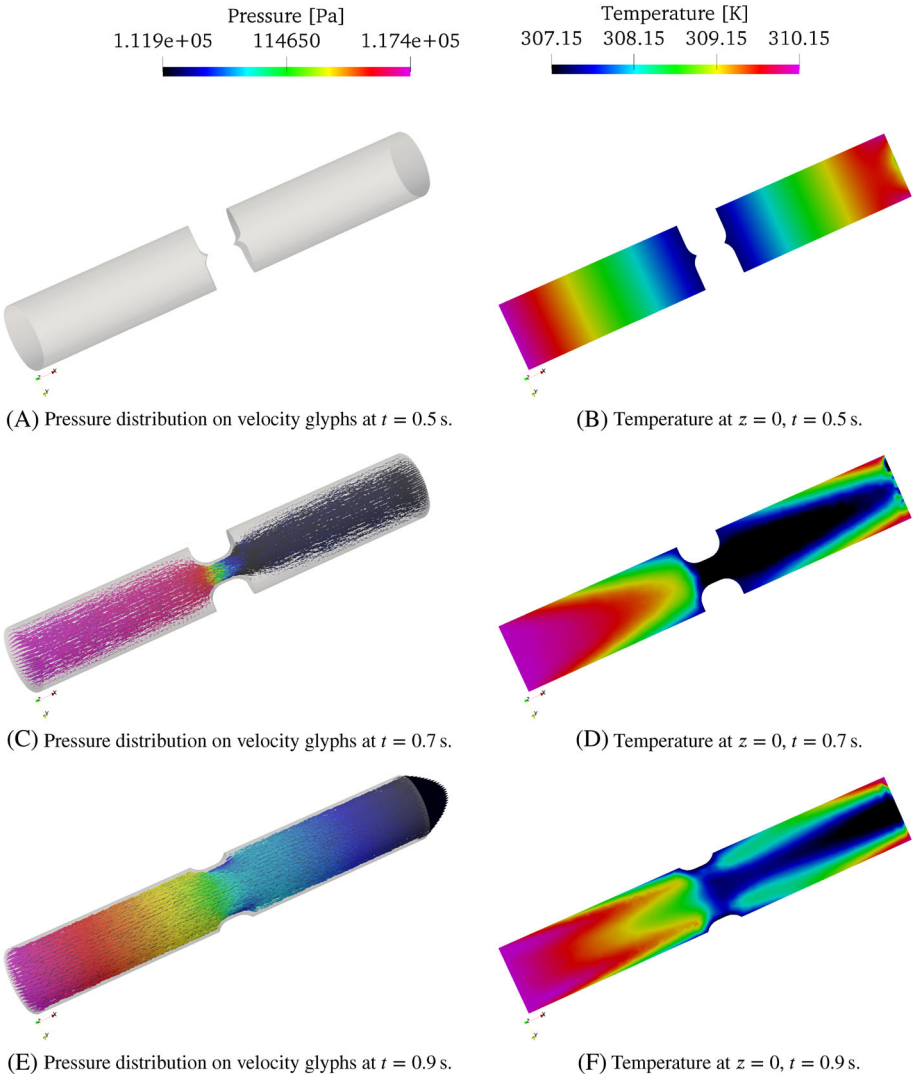
FIGURE 12 Clamped artery test case. *x*-*y*-*t*-mesh [Colour figure can be viewed at wileyonlinelibrary.com]

Figure 9 compares the velocity field computed for the approximately circular pipe at $x = 3\,\text{cm}$, $t = 0.9\,\text{s}$ with the velocity field found in the steady computation of the circular pipe (Section 4.2.1). Note that the solution of the circular pipe is computed on a tetrahedra mesh, whereas the solution for the pipe with approximately circular cross-section is computed on a pentatope mesh. On both meshes, the solution shows concentric circles around the peak velocity at the center. Disturbances of the flow field by the approximately circular cross-section are completely absorbed by the boundary layer, so that the "missing 10%" toward the perfectly circular cross-section have hardly any influence on the bulk flow field.

A more quantitative comparison is presented in Figure 11. The parabolic velocity profile in radial direction (Figure 11A) as well as the linear pressure decay along the pipe axis (Figure 11B) are obtained independently of the approximation of the circular cross-section. Therefore, we consider an approximately circular cross-section of the artery in the following simulation.

4.2.3 | Transient simulation with topology change

As third and final step, we consider the transient simulation with topology change of the spatial computational domain. The topology change caused by the clamp is included in the tetrahedral *x*-*y*-*t*-mesh shown in Figure 12. We find the



(A) Pressure distribution on velocity glyphs at $t = 0.5$ s. (B) Temperature at $z = 0$, $t = 0.5$ s.

(C) Pressure distribution on velocity glyphs at $t = 0.7$ s. (D) Temperature at $z = 0$, $t = 0.7$ s.

(E) Pressure distribution on velocity glyphs at $t = 0.9$ s. (F) Temperature at $z = 0$, $t = 0.9$ s.

FIGURE 13 Pressure, velocity, and temperature in the fully clamped, partially, and fully opened artery [Colour figure can be viewed at wileyonlinelibrary.com]

underlying three-dimensional domain as projection of the four-dimensional space-time geometry onto $z = 0$. The further pentatope mesh generation steps of extrusion, connectivity generation, and elastic deformation are performed as in Section 4.2.2; the employed boundary displacements are given in Equation (20). The final pentatope mesh has 711,645 nodes and 13,602,232 elements. Subsequently, the finite element flow simulation was performed on 240 cores using a distributed memory parallelization based on MPI. It took 16 min of wall clock-time.

Figure 13 presents the simulation results at $t = 0.5$, 0.7, and 0.9 s. For the fully clamped case, we obtain two separate domains and negligible flow velocities (Figure 13A). In the absence of flow, the temperature distribution in the fluid is an interpolation of the temperature prescribed on the domain boundaries (Figure 13B). Note that on the rightmost boundary ($x = 3$ cm), no temperature is prescribed, because this part is an outflow boundary during most of the simulation. However, from $t = 0.5$ s until $t = 0.7$ s, back-flow across this boundary introduces a small disturbance in the temperature field.

When the artery is reopened to roughly half of the total diameter (at $t = 0.7$ s), a strong pressure gradient across the clamp region accelerates the flow in this area (Figure 13C). Furthermore at $t = 0.7$ s, the temperature distribution is strongly influenced by the flow field as well as the cooler clamp (Figure 13D). In the open configuration (Figure 13E,F), we observe a linear pressure decrease from the inflow to the outflow and a parabolic velocity profile everywhere except for the clamp region. Based on the center line velocity of $u_{\mathrm{cl}} = 118.28$ m/s at $x = 2$ cm and $t = 0.9$ μs and the diameter $D = 1$ cm, the Reynolds and Mach number are evaluated as

$$\mathrm{Re} = \frac{\rho\, u_{\mathrm{cl}}\, D}{\mu} \approx 380 \quad \text{and} \quad \mathrm{Ma} = \frac{u_{\mathrm{cl}}}{\sqrt{\gamma R T}} \approx 0.34. \tag{21}$$

The comparison to the velocity profile of the approximately circular pipe in Figure 11A shows that the flow speed computed for the clamped artery is slower at $x = 2\,\mathrm{cm}$, $z = 0\,\mathrm{cm}$, $t = 0.9\,\mathrm{s}$. This is to be expected, as the flow driven by the same pressure gradient has to overcome the additional obstacle of the clamp region (Figure 11B).

5 | CONCLUSION AND OUTLOOK

In this article, we presented the generation and application of four-dimensional SST meshes that allow for a boundary-conforming discretization of spatial domains with time-variant topology. To produce pentatope meshes of complex geometries, the elastic mesh update method was extended to four dimensions and included in the simulation workflow. Regarding the workflow, we described the steps from the space-time geometry to the SST mesh of the physical domain, as well as the post-processing steps to visualize the solution on the four-dimensional mesh as series of data sets on three-dimensional meshes. The workflow was successfully applied to two test cases featuring geometries of a valve and a clamped artery. The transient three-dimensional flow solutions validate the mesh generation in the sense that proper pentatope finite element meshes are obtained on which finite element flow simulations can be performed.

As an outlook, the enhanced meshing capabilities open up a path to parallel-in-time computations on complex domains. We applied domain-decomposition not only to the spatial domain, but to the complete space-time domain. Another promising application area is FSCI simulations with topology changes on boundary-conforming meshes. For this application, the topology changes have to be included in the SST mesh, however, spatial and temporal position of the topology changes can be determined in the course of a coupled FSI simulation and adjusted using 4DEMUM.

ACKNOWLEDGMENTS

Computing resources were provided by JARA-Jülich Aachen Research Alliance and RWTH Aachen University IT Center. Open access funding enabled and organized by Projekt DEAL.

DATA AVAILABILITY STATEMENT

The data that support the findings of this study are available from the corresponding author upon reasonable request.

ORCID

Max von Danwitz https://orcid.org/0000-0002-2814-0027

REFERENCES

1. Ager C, Schott B, Vuong AT, Popp A, Wall WA. A consistent approach for fluid-structure-contact interaction based on a porous flow model for rough surface contact. *Int J Numer Methods Eng*. 2019;119(13):1345-1378.
2. Burman E, Claus S, Hansbo P, Larson MG, Massing A. CutFEM: discretizing geometry and partial differential equations. *Int J Numer Methods Eng*. 2015;104(7):472-501.
3. Sudhakar Y, Moitinho de Almeida J, Wall WA. An accurate, robust, and easy-to-implement method for integration over arbitrary polyhedra: application to embedded interface methods. *J Comput Phys*. 2014;273:393-415.
4. Kadapa C, Dettmer W, Perić D. A stabilised immersed framework on hierarchical b-spline grids for fluid-flexible structure interaction with solid-solid contact. *Comput Methods Appl Mech Eng*. 2018;335:472-489.
5. Kamensky D, Hsu MC, Schillinger D, et al. An immersogeometric variational framework for fluid-structure interaction: application to bioprosthetic heart valves. *Comput Methods Appl Mech Eng*. 2015;284:1005-1053.
6. Sathe S, Tezduyar TE. Modeling of fluid-structure interactions with the space–time finite elements: contact problems. *Comput Mech*. 2008;43(1):51-60.
7. Bogaers AE, Kok S, Reddy BD, Franz T. An evaluation of quasi-Newton methods for application to FSI problems involving free surface flow and solid body contact. *Comput Struct*. 2016;173:71-83.
8. Rendall TC, Allen CB, Power ED. Conservative unsteady aerodynamic simulation of arbitrary boundary motion using structured and unstructured meshes in time. *Int J Numer Methods Fluids*. 2012;70(12):1518-1542.
9. von Danwitz M, Karyofylli V, Hosters N, Behr M. Simplex space-time meshes in compressible flow simulations. *Int J Numer Methods Fluids*. 2019;91:29-48.
10. Behr M. Simplex space-time meshes in finite element simulations. *Int J Numer Methods Fluids*. 2008;57(9):1421-1434.

11. Karabelas E, Neumüller M. Generating admissible space-time meshes for moving domains in (d + 1) dimensions. In: Langer U, Steinbach O, eds. *Space-Time Methods*. De Gruyter; 2019:185-206.
12. Wang L. *Discontinuous Galerkin Methods on Moving Domains with Large Deformations*. PhD thesis. University of California, Berkeley; 2015.
13. Choudhary A, Kachanovich S, Wintraecken M. Coxeter triangulations have good quality. *Math Comput Sci*. 2020;14(1):141-176.
14. Boissonnat JD, Kachanovich S, Wintraecken M. Triangulating submanifolds: an elementary and quantified version of Whitney's method. *Discr Comput Geom*. 2020;66:386-434. https://doi.org/10.1007/s00454-020-00250-8
15. Neumüller M, Steinbach O. Refinement of flexible space-time finite element meshes and discontinuous Galerkin methods. *Comput Vis Sci*. 2011;14(5):189-205.
16. Grande J. Red-green refinement of simplicial meshes in *D* dimensions. *Math Comput*. 2019;88(316):751-782.
17. Caplan PC, Haimes R, Darmofal DL, Galbraith MC. Anisotropic geometry-conforming d-simplicial meshing via isometric embeddings. *Proc Eng*. 2017;203:141-153.
18. Caplan PC, Haimes R, Darmofal DL, Galbraith MC. Four-dimensional anisotropic mesh adaptation. *Comput Aid Des*. 2020;129:102915.
19. Caplan PCD. *Four-Dimensional Anisotropic Mesh Adaptation for Spacetime Numerical Simulations*. PhD thesis. Massachusetts Institute of Technology, Cambridge; 2019.
20. Langer U, Neumüller M, Schafelner A. Space-time finite element methods for parabolic evolution problems with variable coefficients. In: Apel T, Langer U, Meyer A, Steinbach O, eds. *Advanced Finite Element Methods with Applications: Selected Papers from the 30th Chemnitz Finite Element Symposium 2017*. Springer International Publishing; 2019:247-275.
21. Steinbach O, Yang H. Space-time finite element methods for parabolic evolution equations: discretization, a posteriori error estimation, adaptivity and solution. In: Langer U, Steinbach O, eds. *Space-Time Methods*. De Gruyter; 2019:207-248.
22. Voronin K, Lee CS, Neumüller M, Sepulveda P, Vassilevski PS. Space-time discretizations using constrained first-order system least squares (CFOSLS). *J Comput Phys*. 2018;373:863-876.
23. Karyofylli V, Frings M, Elgeti S, Behr M. Simplex space-time meshes in two-phase flow simulations. *Int J Numer Methods Fluids*. 2018;86:218-230.
24. Karyofylli V, Wendling L, Make M, Hosters N, Behr M. Simplex space-time meshes in thermally coupled two-phase flow simulations of mold filling. *Comput Fluids*. 2019;192:104261.
25. Gesenhues L, Behr M. Simulating dense granular flow using the $\mu(I)$-rheology within a space-time framework. *Int J Numer Methods Fluids*. 2021;93:1-16. https://doi.org/10.1002/fld.5014
26. Lynch DR. Unified approach to simulation on deforming elements with application to phase change problems. *J Comput Phys*. 1982;47(3):387-411.
27. Johnson AA, Tezduyar TE. Mesh update strategies in parallel finite element computations of flow problems with moving boundaries and interfaces. *Comput Methods Appl Mech Eng*. 1994;119(1):73-94.
28. Hübner B, Walhorn E, Dinkler D. A monolithic approach to fluid-structure interaction using space-time finite elements. *Comput Methods Appl Mech Eng*. 2004;193(23–26):2087-2104.
29. Bazilevs Y, Calo VM, Hughes TJ, Zhang Y. Isogeometric fluid-structure interaction: theory, algorithms, and computations. *Comput Mech*. 2008;43(1):3-37.
30. Bazilevs Y, Takizawa K, Tezduyar TE. *Computational Fluid-Structure Interaction: Methods and Applications*. John Wiley & Sons; 2013.
31. Spenke T, Hosters N, Behr M. A multi-vector interface quasi-Newton method with linear complexity for partitioned fluid-structure interaction. *Comput Methods Appl Mech Eng*. 2020;361:112810.
32. La Spina A, Förster C, Kronbichler M, Wall WA. On the role of (weak) compressibility for fluid-structure interaction solvers. *Int J Numer Methods Fluids*. 2020;92(2):129-147.
33. Liu J, Yang W, Lan IS, Marsden AL. Fluid-structure interaction modeling of blood flow in the pulmonary arteries using the unified continuum and variational multiscale formulation. *Mech Res Commun*. 2020;107:103556.
34. Zwicke F, Eusterholz S, Elgeti S. Boundary-conforming free-surface flow computations: Interface tracking for linear, higher-order and isogeometric finite elements. *Comput Methods Appl Mech Eng*. 2017;326:175-192.
35. Wendling L, Behr M, Hopf A, Kraemer F, Weber C, Turner P. CFD simulation of oil jets for piston cooling applications comparing the level set and the volume of fluid method. *SAE Int J Adv Current Pract Mob*. 2019;1(2019-01-0155):550-561.
36. Langer U, Schafelner A. Space-time finite element methods for parabolic initial-boundary value problems with non-smooth solutions. In: Lirkov I, Margenov S, eds. *International Conference on Large-Scale Scientific Computing*. Springer; 2019:593-600.
37. Fehn N, Wall WA, Kronbichler M. A matrix-free high-order discontinuous Galerkin compressible Navier-stokes solver: a performance comparison of compressible and incompressible formulations for turbulent incompressible flows. *Int J Numer Methods Fluids*. 2019;89:71-102.
38. Geuzaine C, Remacle JF. Gmsh: a 3-D finite element mesh generator with built-in pre-and post-processing facilities. *Int J Numer Methods Eng*. 2009;79(11):1309-1331.
39. Fernández-Fernández JA. Mesh projector; 2019. github.com/JoseAntFer/MeshProjector
40. Ayachit U. *The ParaView Guide: A Parallel Visualization Application*. Kitware Inc; 2015.
41. Behr M. On the application of slip boundary condition on curved boundaries. *Int J Numer Methods Fluids*. 2004;45:43-51.
42. Hauke G, Hughes TJR. A comparative study of different sets of variables for solving compressible and incompressible flows. *Comput Methods Appl Mech Eng*. 1998;153(1):1-44.

43. Xu F, Moutsanidis G, Kamensky D, et al. Compressible flows on moving domains: stabilized methods, weakly enforced essential boundary conditions, sliding interfaces, and application to gas-turbine modeling. *Comput Fluids*. 2017;158:201-220.
44. White FM. *Viscous Fluid Flow*. McGraw-Hill; 2006.

How to cite this article: von Danwitz M, Antony P, Key F, Hosters N, Behr M. Four-dimensional elastically deformed simplex space-time meshes for domains with time-variant topology. *Int J Numer Meth Fluids*. 2021;93(12):3490-3506. https://doi.org/10.1002/fld.5042