SIAM J. SCI. COMPUT.
Vol. 41, No. 3, pp. A1793–A1821

PARALLEL AND ROBUST PRECONDITIONING FOR SPACE-TIME ISOGEOMETRIC ANALYSIS OF PARABOLIC EVOLUTION PROBLEMS*

CHRISTOPH HOFER†, ULRICH LANGER‡, MARTIN NEUMÜLLER‡, AND RAINER SCHNECKENLEITNER§

Abstract. We propose and investigate new robust preconditioners for space-time Isogeometric Analysis (IgA) of parabolic evolution problems. These preconditioners are based on a time-parallel multigrid method. We consider a decomposition of the space-time cylinder into time-slabs which are coupled via a discontinuous Galerkin technique. The time-slabs provide the structure for the time-parallel multigrid solver. The most important part of the multigrid method is the smoother. We utilize the special structure of the involved operator to decouple its application into several spatial problems by means of generalized eigenvalue or Schur decompositions. Some of these problems have a symmetric saddle point structure, for which we present robust preconditioners. Finally, we present numerical experiments confirming the robustness of our space-time IgA solver.

Key words. parabolic evolution problems, space-time methods, isogeometric analysis, discontinuous Galerkin, solvers, robust preconditioners

AMS subject classifications. 65F08, 65M60

DOI. 10.1137/18M1208794

1. Introduction.

Time-dependent partial differential equations (PDEs) having parabolic type play an important role in the simulation of various physical processes, like heat conduction, diffusion, and 2D eddy-current problems in electromagnetics. They are often given as initial-boundary value problems (IBVP). The discretization of such problems is usually performed either by first discretizing in time by means of a time-stepping method and then in space, e.g., by finite elements or vice versa. The former method is often called Rothe's method [22], and the latter the vertical method of lines [37]. Both of the approaches are sequential in time. In order to treat such problems on massively parallel computers, different approaches are required to overcome the sequential structure. There exist various techniques for parallelization in time. In particular, we would like to mention time-parallel solvers for the huge all-at-once space-time system of algebraic equations which finally defines the efficiency of the complete space-time method. The parareal method [25, 12], time-parallel and space-time multigrid methods [13, 38, 28, 9, 11], space-time algebraic multigrid methods [32], space-time multilevel methods [2], Schur complement preconditioned inexact UZAWA methods [29], and block-circular preconditioned MINRES [27]. For the last two classes of solvers, the authors give a rigorous convergence analysis provided that the system has a uniform structure. In particular, the time-step must remain unchanged. We

*Submitted to the journal's Methods and Algorithms for Scientific Computing section August 21, 2018; accepted for publication (in revised form) March 19, 2019; published electronically June 4, 2019.

http://www.siam.org/journals/sisc/41-3/M120879.html

Funding: This work was supported by the Austrian Science Fund (FWF) under the grants W1214, project DK4, and NFN S117-03.

†Doctoral Program "Computational Mathematics", Johannes Kepler University, A-4040 Linz, Austria (christoph.hofer@dk-compmath.jku.at).

‡Institute of Computational Mathematics, Johannes Kepler University, A-4040 Linz, Austria (ulanger@numa.uni-linz.ac.at, neumueller@numa.uni-linz.ac.at).

§Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, A-4040 Linz, Austria (rainer.schneckenleitner@ricam.oeaw.ac.at).

refer to [10] for an overview of time-parallel methods and the corresponding references, and to [33] for an overview of space-time methods and solvers.

In the current work, we focus on space-time methods. More precisely, we consider the time as just another variable, say x_{d+1}, where $x_1, \dots, x_d$ are the d-dimensional spatial variables. The derivative in the time direction is then viewed as a strong convection term in the direction x_{d+1}. In order to provide a stable discretization, we use stabilization techniques developed for convection dominated elliptic convection-diffusion problems; see, e.g., [34]. To be more precise, we consider the Streamline-Upwind Petrov–Galerkin (SUPG) method introduced in [6]. We consider the linear parabolic IBVP. Find $u : \overline{Q} \to \mathbb{R}$ such that

$$\partial_t u - \Delta u = f \text{ in } Q, \quad u = 0 \text{ on } \Sigma, \quad \text{and} \quad u = u_0 \text{ on } \overline{\Sigma}_0, \tag{1.1}$$

as a typical parabolic model problem posed in the space-time cylinder $\overline{Q} = \overline{\Omega} \times \overline{J} = \overline{\Omega} \times [0,T] = Q \cup \Sigma \cup \overline{\Sigma}_0 \cup \overline{\Sigma}_T$, where ∂_t denotes the partial time derivative, Δ is the spatial Laplace operator, f is a given source function, u_0 are the given initial data, T is the final time, $J = (0,T)$ is the time interval, $Q = \Omega \times (0,T)$, $\Sigma = \partial\Omega \times (0,T)$, $\Sigma_0 := \Omega \times \{0\}$, $\Sigma_T := \Omega \times \{T\}$, and $\Omega \subset \mathbb{R}^d$ $(d = 1, 2, 3)$ denotes the spatial computational domain with the boundary $\partial\Omega$. In [23], time-upwind test functions were used to construct a stable single-patch discretization scheme in the Isogeometric Analysis (IgA) framework. This approach was extended in [15] to multiple patches in time, where each space-time patch Q_n is given as space-time-slab $Q_n = \Omega \times (t_{n-1}, t_n)$ corresponding to a decomposition $0 = t_0 < t_1 < \cdots < t_N = T$ of the time interval $[0,T]$. A discontinuous Galerkin (dG) technique was used for coupling the space-time-slabs in an appropriate way. Finally, the resulting huge linear system $\boldsymbol{L}_h \boldsymbol{u}_h = \boldsymbol{f}_h$ is solved by the time-parallel multigrid (MG) method introduced in [11]. The main new contribution of this paper is the smoothers that finally yield robust multigrid solvers. Moreover, such solvers could be used as preconditioners for the Generalized Minimal Residual (GMRES) solver.

IgA is a powerful methodology for discretizing PDEs. It was first introduced in [18] and its advantages have been highlighted in many publications; see, e.g., the monograph [7], the survey paper [3], and the references therein. The main idea is to use the same smooth higher-order splines for both representing the computational domain and approximating the solution of the PDE or the PDE system. The most common choices are B-Splines and Non-Uniform Rational B-Splines (NURBS). One of the strengths of IgA is the capability of creating high-order spline spaces, while keeping the number of degrees of freedom quite small.

The purpose of this paper is to investigate the efficient realization of the time-parallel MG method mentioned above. The special time-multipatch dG structure of the discretization leads to a block-bidiagonal matrix $\boldsymbol{L}_h = \text{blockbidiag}(-\boldsymbol{B}_n, \boldsymbol{A}_n)$, where the block-diagonal matrices $\boldsymbol{A}_n$, $i = 1, \dots, N$, and the block-subdiagonal matrices $\boldsymbol{B}_n$, $i = 2, \dots, N$, have tensor product representations. The most costly part of the MG method is the application of the smoother, which is of (inexact) damped block Jacobi type with a block-diagonal preconditioner $\boldsymbol{D}_h^{-1}$. The matrix $\boldsymbol{D}_h$ is formed by the diagonal blocks of $\boldsymbol{L}_h$, i.e., by $\boldsymbol{A}_n$. This paper investigates the efficient application of $\boldsymbol{A}_n^{-1}$ by utilizing its tensor product structure. We use ideas from [36, 30] to perform a decomposition of $\boldsymbol{A}_n$ into a series of spatial problems, for which we propose robust block preconditioners. These preconditioners are constructed by means of operator interpolation; see, e.g., [41, 5, 1]. Moreover, their application can be further accelerated by using domain decomposition or MG approaches in connection with parallelization in space.

The remainder of the paper is organized as follows. In section 2, we rephrase basic definitions and the stable space-time dG-IgA variational formulation. Section 3 is devoted to the construction of efficient smoothers used in the time-parallel MG solver, respectively, preconditioner. Numerical experiments confirming the theoretical results are presented in section 4. Finally, we draw some conclusions in section 5.

2. Preliminaries. In this section, we introduce the IgA concept, recall some important definitions, and state the space-time variational IgA scheme derived and analyzed in [15]. For a more detailed discussion of IgA, we refer to [7, 3]. We follow the notation used in [15].

2.1. Isogeometric analysis. Let $\hat{\Omega} := (0,1)^d$, $d = 1, 2, 3$, be the d-dimensional unit cube, which we refer to as the *parameter domain*. Let p_ι and N_ι, $\iota \in \{1, \dots, d\}$, denote the degree and the number of basis functions in the x_ι-direction. Moreover, let $\Xi_\iota = \{\xi_1 = 0, \xi_2, \dots, \xi_{n_\iota} = 1\}$, $n_\iota = N_\iota + p_\iota + 1$, be a partition of $[0,1]$, called the *knot vector*. With these ingredients we are able to define the B-Spline basis $\hat{N}_{i,p}$, $i \in \{1, \dots, N_\iota\}$ on $[0,1]$ via the Cox–De Boor algorithm; cf. [7]. The generalization to $\hat{\Omega}$ is realized by considering a tensor product, again denoted by $\hat{N}_{i,p}$, where $i = (i_1, \dots, i_d)$ and $p = (p_1, \dots, p_d)$ are multi-indices. For notational simplicity, we define $\mathcal{I} := \{(i_1, \dots, i_d) \,|\, i_\iota \in \{1, \dots, N_\iota\}\}$ as the set of multi-indices.

The computational domain Ω, also called *physical domain*, is parametrized by the B-Spline basis functions. It is given as an image of the parameter domain $\hat{\Omega}$ under the so-called *geometrical mapping* $G : \hat{\Omega} \to \mathbb{R}^d$, defined as

$$G(\xi) := \sum_{i \in \mathcal{I}} P_i \hat{N}_{i,p}(\xi),$$

with the control points $P_i \in \mathbb{R}^d$, $i \in \mathcal{I}$. In order to represent more complicated geometries Ω, multiple nonoverlapping domains (patches) $\Omega_n := G_n(\hat{\Omega}), n = 1, \dots, N$, are composed, where each patch is associated with a different geometrical mapping G_n. In the following, we refer to such domains $\overline{\Omega} := \bigcup_{n=1}^N \overline{\Omega}_n$ as *multipatch domains*.

In the IgA concept, the B-Splines are not only used for representing the geometry, but also as the basis for finite-dimensional spaces used for approximating the solution of the PDE. This motivates us to define the basis functions $N_{i,p} := \hat{N}_{i,p} \circ G^{-1}$ in the physical space by mapping the corresponding basis functions $\hat{N}_{i,p}$ defined in the parameter domain $\hat{\Omega}$. On each patch Ω_n, we now define the local IgA space $V_h^n := \operatorname{span}\{N_{i,p}\}_{i \in \mathcal{I}}$. The construction of global IgA space V_h depends on the formulation used. A continuous Galerkin formulation needs C^0 continuity, whereas a discontinuous Galerkin formulation only requests C^{-1} continuity across the patch boundaries. Below we will generalize this IgA concept from the spatial domain Ω to the space-time cylinder Q.

2.2. Space-time variational formulation. Let $J = (0, T)$ be the time interval with some final time $T > 0$. For later use, we define the space-time cylinder $Q = \Omega \times J$ and its boundary parts $\Sigma = \partial\Omega \times J$, $\Sigma_T = \Omega \times \{T\}$, and $\Sigma_0 = \Omega \times \{0\}$ such that $\partial Q = \Sigma \cup \overline{\Sigma}_0 \cup \overline{\Sigma}_T$. According to the usual definition of partial derivatives $\partial_x^{\boldsymbol{\alpha}} v = \partial^{|\alpha|} v / \partial x_1^{\alpha_1} \dots \partial x_d^{\alpha_d}$, $\alpha = (\alpha_1, \dots, \alpha_d)$, $|\alpha| = \alpha_1 + \dots + \alpha_d$, we now define the spatial gradient $\nabla_x v = (\partial_{x_1} v, \dots, \partial_{x_d} v)$, where $\partial_{x_i} v = \partial v / \partial x_i$, $i = 1, \dots, d$. Let ℓ and m be nonnegative integers. For functions defined in the space-time cylinder Q, we define the Sobolev spaces

$$H^{\ell,m}(Q) = \{v \in L_2(Q) : \partial_x^{\boldsymbol{\alpha}} v \in L_2(Q) \text{ for } 0 \le |\boldsymbol{\alpha}| \le \ell,\ \partial_t^i v \in L_2(Q),\ i = 0, \dots, m\},$$

where $\partial_t = \partial/\partial t$, and, in particular, the subspaces

$$H_0^{1,0}(Q) = \{v \in L_2(Q) : \nabla_x v \in [L_2(Q)]^d,\ v = 0 \text{ on } \Sigma\},$$
$$H_0^{1,1}(Q) = H_0^1(Q) = \{v \in L_2(Q) : \nabla_x v \in [L_2(Q)]^d,\ \partial_t v \in L_2(Q),\ v = 0 \text{ on } \Sigma\}, \text{ and}$$
$$H_{0,\bar{0}}^{1,1}(Q) = \{v \in H_0^1(Q) :\ v = 0 \text{ on } \Sigma_T\}.$$

We equip the above spaces with the norms

$$\|v\|_{H^{\ell,m}(Q)} = \Big(\sum_{|\boldsymbol{\alpha}| \le \ell} \|\partial_x^\alpha v\|_{L_2(Q)}^2 + \sum_{j=0}^{m} \|\partial_t^j v\|_{L_2(Q)}^2 \Big)^{\frac{1}{2}}$$

and seminorms

$$|v|_{H^{\ell,m}(Q)} = \Big(\sum_{|\boldsymbol{\alpha}| = \ell} \|\partial_x^\alpha v\|_{L_2(Q)}^2 + \|\partial_t^m v\|_{L_2(Q)}^2 \Big)^{\frac{1}{2}}.$$

Using the standard procedure and integration by parts with respect to both x and t, we can easily derive the following space-time variational formulation of (1.1): find $u \in H_0^{1,0}(Q)$ such that

$$a(u,v) = l(v) \quad \text{for all } v \in H_{0,\bar{0}}^{1,1}(Q) \tag{2.1}$$

with the bilinear form

$$a(u,v) = -\int_Q u(x,t)\partial_t v(x,t)\,dx\,dt + \int_Q \nabla_x u(x,t)\ \cdot\ \nabla_x v(x,t)\,dx\,dt$$

and the linear form

$$l(v) = \int_Q f(x,t)v(x,t)\,dx\,dt + \int_\Omega u_0(x)v(x,0)\,dx,$$

where the source $f \in L_2(Q)$ and the initial conditions $u_0 \in L_2(\Omega)$ are given.

For simplicity, we only consider homogeneous Dirichlet boundary conditions on Σ. However, the method presented here can easily be generalized to other classes of boundary conditions as well as more general elliptic operators replacing the Laplace operator in our model problem. The space-time variational formulation (2.1) has a unique solution; see, e.g., [20, 21], where one can also find a priori estimates and regularity results. In particular, the solution u of (2.1) belongs to space $H_0^{\Delta,1}(Q) = W_{2,0}^{\Delta,1}(Q) = \{v \in H_0^1(Q) : \Delta_x v \in L_2(Q)\}$, and u continuously depends on t in the norm of the space $H_0^1(\Omega)$ provided that $f \in L_2(Q)$ and $u_0 \in H_0^1(\Omega)$; see Chapter III, Theorem 2.1 in [20]. This case of maximal parabolic regularity is considered throughout the paper.

2.3. Stable multipatch space-time dG-IgA discretization. We describe the space-time cylinder Q as a union of nonoverlapping time patches $Q_1, Q_2, \dots, Q_N$. We consider a partition $0 = t_0 < t_1 < \cdots < t_N = T$ of the time interval $\overline{J} = [0,T]$, and denote the subintervals by $J_n = (t_{n-1}, t_n)$. We now define the time-patches $Q_n = \Omega \times J_n$ and the faces $\Sigma_n = \overline{Q}_{n+1} \cap \overline{Q}_n = \Omega \times \{t_n\}$ between the time patches, where we identify Σ_T and Σ_N. In that way, we have the decomposition $\overline{Q} = \cup_{n=1}^N \overline{Q}_n$,

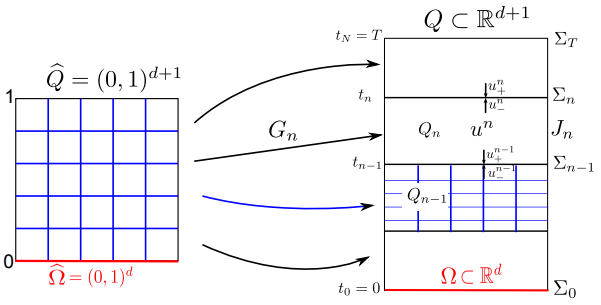

FIG. 1. *Decomposition of the space-time domain Q into time-slabs Q_n.*

where each space-time cylinder Q_n has a geometrical mapping $G_n : \overline{\widehat{Q}} := [0,1]^{d+1} \to \overline{Q}_n := \overline{\Omega} \times \overline{J}_n$. To keep the notation simple, in what follows, we will use the sup-index n to denote the restrictions to Q_n, e.g., $u^n := u|_{Q_n}$. An illustration is given in Figure 1.

Remark 2.1. We note that the spatial domain Ω can also be a multipatch domain. This leads to a representation of Q_n as union of nonoverlapping space-time patches $Q_{n,k}, k = 1, \dots, K$, i.e., $\overline{Q}_n = \cup_{k=1}^K \overline{Q}_{n,k}$. The corresponding bases are then coupled in a conforming way.

We denote the global discontinuous B-Spline space and the local continuous patch-wise B-Spline spaces by

$$V_{0h} = \{v_h \in L_2(Q) : v_h|_{Q_n} \in V_h^n \text{ for } n = 1, \dots, N, \text{ and } v_h|_\Sigma = 0\} \tag{2.2}$$

and

$$V_{0h}^n = \{v_h \in V_h^n, \text{ for } n = 1, \dots, N, \text{ and } v_h|_\Sigma = 0\}, \tag{2.3}$$

respectively. Notice that $v_h \in V_{0h}$ is, in general, discontinuous across Σ_n. We introduce the notation

$$v_{h,+}^n = \lim_{\varepsilon \to 0^+} v_h(t_n + \varepsilon),\ v_{h,-}^n = \lim_{\varepsilon \to 0^-} v_h(t_n + \varepsilon),\ [\![v_h]\!]^n = v_{h,+}^n - v_{h,-}^n,\ [\![v_h]\!]^0 = v_{h,+}^0,$$

where $[\![v_h]\!]^n$ denotes the jump of v_h across Σ_n for $n \geq 1$, and $[\![v_h]\!]^0 = v_{h,+}^0$ denotes the trace of v_h on Σ_0. For a smooth function u, we obviously have $[\![u]\!]^n = u_+^n - u_-^n = 0$ for $n \geq 1$, and $[\![u]\!]^0 = u|_{\Sigma_0}$.

We now consider the space-time-slab Q_n with the outer normal $\boldsymbol{n} = (\boldsymbol{n}_x, n_t) = (n_1, \dots, n_d, n_{d+1})$ to ∂Q_n. Let $v_h^n \in V_{0h}^n$ and $w_h^n = v_h^n + \theta_n h_n \partial_t v_h^n$ with some positive parameter θ_n, which will be defined later. We note that $w_h^n = 0$ on Σ. The choice of the time-upwind test function is motivated by SUPG [6]. Let the solution u of (2.1) belong to $H_0^{\Delta,1}(Q)$ that is ensured if $f \in L_2(Q)$ and $u_0 \in H_0^1(\Omega)$. Multiplying the parabolic PDE $\partial_t u - \Delta u = f$ by w_h^n, integrating over Q_n, and applying integration

by parts in the elliptic terms, we arrive at the variational identity

$$\begin{aligned}&\int_{Q_n} (\partial_t u\,(v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x u \cdot \nabla_x v_h^n + \theta_n\, h_n \nabla_x u \cdot \nabla_x \partial_t v_h^n)\, dx\, dt \\ &- \int_{\partial Q_n} n_x \cdot \nabla_x u (v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx + \int_{\Sigma_{n-1}} u_+^{n-1}\, v_{h,+}^{n-1}\, dx \\ &= \int_{Q_n} f\,(v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt + \int_{\Sigma_{n-1}} u_-^{n-1}\, v_{h,+}^{n-1}\, dx\end{aligned}$$

for $n = 1, \dots, N$, where we used that $u_-^{n-1} = u_+^{n-1} = u^{n-1}$ on every Σ_{n-1}. Furthermore, using $n_x = 0$ on Σ_n and $w_h = 0$ on Σ, we have

$$\begin{aligned}a_{Q_n}(u, v_h) :=& \int_{Q_n} (\partial_t u\,(v_h^n + \theta_n\, h_n \partial_t v_h^n) + \nabla_x u \cdot \nabla_x v_h^n + \theta_n\, h_n \nabla_x u \cdot \nabla_x \partial_t v_h^n)\, dx\, dt \\ &+ \int_{\Sigma_{n-1}} [\![u]\!]^{n-1}\, v_{h,+}^{n-1}\, dx = \int_{Q_n} f\,(v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt,\end{aligned}$$

for all $n = 2, \dots, N$, and

$$\begin{aligned}a_{Q_1}(u, v_h) :=& \int_{Q_1} (\partial_t u\,(v_h^1 + \theta_1\, h_1 \partial_t v_h^1) + \nabla_x u \cdot \nabla_x v_h^1 + \theta_1\, h_1 \nabla_x u \cdot \nabla_x \partial_t v_h^1)\, dx\, dt \\ &+ \int_{\Sigma_0} [\![u]\!]^0\, v_{h,+}^0\, dx = \int_{Q_1} f\,(v_h^1 + \theta_1\, h_1 \partial_t v_h^1)\, dx\, dt + \int_{\Sigma_0} u_0\, v_{h,+}^0\, dx.\end{aligned}$$

Summing over all Q_n, we conclude that

$$a_h(u, v_h) = l_h(v_h) \quad \forall v_h \in V_{0h}, \tag{2.4}$$

where

$$a_h(u, v_h) = \sum_{n=1}^{N} a_{Q_n}(u, v_h), \tag{2.5}$$

and

$$l_h(v_h) = \sum_{n=1}^{N} \int_{Q_n} f\,(v_h^n + \theta_n\, h_n \partial_t v_h^n)\, dx\, dt + \int_{\Sigma_0} u_0\, v_{h,+}^0\, dx.$$

Now, the space-time dG IgA variational scheme for (1.1) reads as follows: find $u_h \in V_{0h}$ such that

$$a_h(u_h, v_h) = l_h(v_h) \quad \forall v_h \in V_{0h}. \tag{2.6}$$

Motivated by the definition of the bilinear form $a_h(\cdot, \cdot)$ in (2.5), we introduce the mesh-dependent dG norm

$$\|v\|_h^2 = \sum_{n=1}^{N} \Big(\|\nabla_x v\|_{L_2(Q_n)}^2 + \theta_n\, h_n\, \|\partial_t v\|_{L_2(Q_n)}^2 + \frac{1}{2} \|[\![v]\!]^{n-1}\|_{L_2(\Sigma_{n-1})}^2 \Big) + \frac{1}{2} \|v\|_{L_2(\Sigma_N)}^2.$$

In the following, we recall some important properties of the IgA scheme (2.6), respectively the bilinear form (2.5). For the proofs, we refer to [15].

LEMMA 2.2. *The bilinear form $a_h(\cdot,\cdot)$, defined in (2.5), is V_{0h}-elliptic, i.e.,*

$$a_h(v_h, v_h) \geq C_e \|v_h\|_h^2 \quad \text{for } v_h \in V_{0h}, \tag{2.7}$$

where $C_e = 0.5$ for $\theta_n \leq C_{inv,0}^{-2}$, with the positive, h_n-independent constant $C_{inv,0}$ from the inverse inequality

$$\|v_h\|_{L_2(\Sigma_{n-1})}^2 \leq C_{inv,0} h_n^{-1} \|v_h\|_{L_2(Q_n)}^2$$

that holds for all $v_h \in V_h^n$, $n = 1, \ldots, N$.

The V_{0h}-ellipticity of the bilinear form $a_h(\cdot,\cdot)$ is crucial. It immediately implies uniqueness of the solution to (2.6), and uniqueness always yields existence in the finite-dimensional setting. Furthermore, we introduce the extended space $V_{0h,*} = H_0^{\Delta,1}(Q) + V_{0h}$ endowed with the norm

$$\|v\|_{h,*} := \left(\|v\|_h^2 + \sum_{n=1}^{N} (\theta_n h_n)^{-1} \|v\|_{L_2(Q_n)}^2 + \sum_{n=2}^{N} \|v_-^{n-1}\|_{L_2(\Sigma_{n-1})}^2 \right)^{\frac{1}{2}}. \tag{2.8}$$

Then the bilinear form $a_h(\cdot,\cdot)$ is bounded on $V_{0h,*} \times V_{0h}$ as stated in Lemma 2.3.

LEMMA 2.3. *The boundedness inequality*

$$|a_h(u, v_h)| \leq C_b \, \|u\|_{h,*} \|v_h\|_h \tag{2.9}$$

holds for all $u \in V_{0h,*}$ and $v_h \in V_{0h}$, where the constant $C_b = \max\{C_{inv,1}\, \theta_{max}, 2\}$, with $\theta_{max} = \max_n\{\theta_n\} \leq C_{inv,0}^{-2}$ and the positive, h_n-independent constant $C_{inv,1}$ from the inverse inequalities $\|\partial_t \partial_{x_i} v_h\|_{L_2(Q_n)}^2 \leq C_{inv,1} h_n^{-2} \|\partial_{x_i} v_h\|_{L_2(Q_n)}^2$ that holds for all $v_h \in V_h^n$, $n = 1, \ldots, N$, $i = 1, \ldots, d$.

The IgA scheme (2.6) is consistent. Indeed, consistency directly follows from the derivation of the scheme. Consistency yields Galerkin orthogonality $a_h(u - u_h, v_h) = 0$ for all $v_h \in V_{0h}$ that together with the V_{0h}-ellipticity (2.7) and the $V_{0h,*} \times V_{0h}$-boundedness (2.9) immediately implies the Cea-like discretization error estimate

$$\|u - u_h\|_h \leq \left(1 + \frac{C_b}{C_e}\right) \inf_{v_h \in V_{0h}} \|u - v_h\|_{h,*}, \tag{2.10}$$

from which one can derive convergence rate estimates under additional regularity assumptions imposed on the solution u. The following theorem directly follows from the approximation (quasi-interpolation) error estimate that was proven in [15].

THEOREM 2.4. *Let u and u_h solve (2.1) and (2.6), respectively. Then, under the regularity assumption that the solution u of (2.1) belongs to $V = H_0^{1,0}(Q) \cap H^{\ell,m}(Q)$ with some $\ell \geq 2$ and $m \geq 1$, there exists a positive generic constant C, which is independent of $h = \max\{h_n\}$, such that*

$$\|u - u_h\|_h \leq C(h^{\ell-1} + h^{m-\frac{1}{2}}) \, \|u\|_{H^{\ell,m}(Q)}. \tag{2.11}$$

Moreover, if $1 \leq m < \ell \leq p+1$, then $\|u - u_h\|_h \leq C h^{m-\frac{1}{2}} \|u\|_{H^{\ell,m}(Q)}$.

Remark 2.5. For the case of highly smooth solutions, i.e., $\min(\ell, m) \geq p+1$, estimate (2.11) takes the form $\|u - u_h\|_h \leq C\, h^p \, \|u\|_{H^{p+1,p+1}(Q)}$.

Remark 2.6. The case $\theta_n = 0$ can be handled by proving a discrete inf-sup condition as it was done for space-time finite element discretizations in [31].

2.4. Efficient matrix assembly. We first recall the IgA variational problem (2.6), where we look for the IgA solution $u_h \in V_{0h} := V_{0h}^1 \times \cdots \times V_{0h}^N$ such that the IgA scheme (2.6) is fulfilled for all test functions $v_h \in V_{0h}$. The IgA bilinear form $a_h(u_h, v_h) = \sum_{n=1}^N a_{Q_n}(u_h, v_h)$ is given as the sum of all local bilinear forms $a_{Q_n}(u_h, v_h)$. The local bilinear form $a_{Q_n}(u_h, v_h)$ can be rewritten as follows:

$$\begin{aligned} a_{Q_n}(u_h, v_h) &= \int_{Q_n} \partial_t u_h^n \, (v_h^n + \theta_n \, h_n \partial_t v_h^n) + \nabla_x u_h^n \cdot \nabla_x (v_h^n + \theta_n \, h_n \partial_t v_h^n) \, dx \, dt \\ &\quad + \int_{\Sigma_{n-1}} [\![u_h]\!]^{n-1} \, v_{h,+}^{n-1} \, ds \\ &= \int_{Q_n} \partial_t u_h^n \, (v_h^n + \theta_n \, h_n \partial_t v_h^n) + \nabla_x u_h^n \cdot \nabla_x (v_h^n + \theta_n \, h_n \partial_t v_h^n) \, dx \, dt \\ &\quad + \int_{\Sigma_{n-1}} u_{h,+}^{n-1} \, v_{h,+}^{n-1} \, ds - \int_{\Sigma_{n-1}} u_{h,-}^{n-1} \, v_{h,+}^{n-1} \, ds \\ &=: b_{Q_n}(u_h^n, v_h^n) - \int_{\Sigma_{n-1}} u_{h,-}^{n-1} \, v_{h,+}^{n-1} \, ds, \end{aligned}$$

where $n = 1, \ldots, N$. For the local spaces V_{0h}^n defined by (2.3), we now introduce the simpler notation φ_j^n for the B-Spline basis functions such that $V_{0h}^n = \operatorname{span}\{\varphi_j^n\}_{j=1}^{N_n}$ for $n = 1, \ldots, N$. Once the basis is chosen, the IgA variational scheme (2.6) immediately results in the huge linear system

$$\boldsymbol{L}_h \vec{u}_h := \begin{pmatrix} \boldsymbol{A}_1 & & & \\ -\boldsymbol{B}_2 & \boldsymbol{A}_2 & & \\ & \ddots & \ddots & \\ & & -\boldsymbol{B}_N & \boldsymbol{A}_N \end{pmatrix} \begin{pmatrix} \vec{u}_1 \\ \vec{u}_2 \\ \vdots \\ \vec{u}_N \end{pmatrix} = \begin{pmatrix} \vec{f}_1 \\ \vec{f}_2 \\ \vdots \\ \vec{f}_N \end{pmatrix} =: \vec{f}_h, \tag{2.12}$$

with the matrices

$$\boldsymbol{A}_n[i,j] := b_{Q_n}(\varphi_j^n, \varphi_i^n) \quad \text{for } i,j = 1, \ldots, N_n$$

on the diagonal for $n = 1, \ldots, N$, and the matrices

$$\boldsymbol{B}_n[i,k] := \int_{\Sigma_{n-1}} \varphi_{k,-}^{n-1} \, \varphi_{i,+}^{n-1} \, ds \quad \text{for } k = 1, \ldots, N_{n-1} \text{ and } i = 1, \ldots, N_n$$

on the lower off diagonal for $n = 2, \ldots, N$. Moreover, for $n = 1, \ldots, N$, the right-hand sides are given by

$$\vec{f}_n[i] := l_h(\varphi_i^n), \quad i = 1, \ldots, N_n.$$

If the geometrical mappings $G_n : \overline{\widehat{Q}} \to \overline{Q}_n$, $n = 1, \ldots, N$, preserve the tensor product structure of the IgA basis functions φ_i^n, we can use this information to save assembling time and storage costs for the linear system (2.12). In this case, we can write the basis functions φ_i^n in the form

$$\varphi_i^n(x,t) = \phi_{i_x}^n(x) \psi_{i_t}^n(t) \quad \text{with } i_x \in \{1, \ldots, N_{n,x}\} \text{ and } i_t \in \{1, \ldots, N_{n,t}\},$$

where $N_n = N_{n,x} N_{n,t}$. Using this representation, we can rewrite the matrices $\boldsymbol{A}_n$ as

$$\boldsymbol{A}_n = \boldsymbol{K}_{n,t} \otimes \boldsymbol{M}_{n,x} + \boldsymbol{M}_{n,t} \otimes \boldsymbol{K}_{n,x}, \quad n = 1, \ldots, N, \tag{2.13}$$

with the standard mass and stiffness matrices with respect to space

$$\boldsymbol{M}_{n,x}[i_x, j_x] := \int_\Omega \phi^n_{j_x} \phi^n_{i_x}\, dx, \qquad \boldsymbol{K}_{n,x}[i_x, j_x] := \int_\Omega \nabla_x \phi^n_{j_x} \cdot \nabla_x \phi^n_{i_x}\, dx,$$

where $i_x, j_x = 1, \dots, N_{n,x}$, and the corresponding matrices with respect to time

$$\begin{aligned} \boldsymbol{K}_{n,t}[i_t, j_t] &:= \int_{t_{n-1}}^{t_n} \partial_t \psi^n_{j_t} (\psi^n_{i_t} + \theta_n h_n \partial_t \psi^n_{i_t})\, dt + \psi^n_{j_t}(t_{n-1}) \psi^n_{i_t}(t_{n-1}), \\ \boldsymbol{M}_{n,t}[i_t, j_t] &:= \int_{t_{n-1}}^{t_n} \psi^n_{j_t} (\psi^n_{i_t} + \theta_n h_n \partial_t \psi^n_{i_t})\, dt, \end{aligned} \tag{2.14}$$

with $i_t, j_t = 1, \dots, N_{n,t}$. The matrices on the off-diagonal $\boldsymbol{B}_n$, $n = 2, \dots, N$, can be rewritten in the form

$$\boldsymbol{B}_n := \boldsymbol{N}_{n,t} \otimes \widetilde{\boldsymbol{M}}_{n,x},$$

with the matrices

$$\widetilde{\boldsymbol{M}}_{n,x}[i_x, k_x] := \int_\Omega \phi^{n-1}_{k_x} \phi^n_{i_x}\, dx \quad \text{and} \quad \boldsymbol{N}_{n,t}[i_t, k_t] := \psi^{n-1}_{k_t}(t_{n-1}) \psi^n_{i_t}(t_{n-1}),$$

where $i_x = 1, \dots, N_{n,x}$, $k_x = 1, \dots, N_{n-1,x}$, $i_t = 1, \dots, N_{n,t}$, and $k_t = 1, \dots, N_{n-1,t}$.

3. Solvers for space-time problems. This section aims at the development of efficient solvers for the huge space-time system (2.12). Our new solver is based on the time-parallel multigrid method proposed in [11]; see also the Ph.D. thesis [28]. The key point in realizing the method efficiently is the application of the smoother, which is the most costly part of the algorithm. The goal is to utilize the structure of the involved matrix $\boldsymbol{A}_n^{-1}$, which then allows for a faster application. We mention that the approach presented in this section is not restricted to a continuous Galerkin discretization for the spatial domains. The approach immediately extends to a spatial coupling of the interface degrees of freedom (dofs) via a dG-IgA scheme; see, e.g, [24].

3.1. Space-time multigrid. In this section, we give a review of the time-parallel MG from [28]. Multigrid consists of three main ingredients: the coarse grid solver, the smoother, and the prolongation/restriction operators.

Concerning the restriction and prolongation operator, it is advantageous to consider coarsening in space and in time separately. The prolongation is then just defined as the adjoint operator. The coarsening in space is performed by classical MG transfer operators, whereas the coarsening in time is realized by combining two successive time-slabs to a single time-slab; see the illustration in Figure 2. For simplicity, we assume that the considered basis is identical on each patch, which greatly simplifies the calculation of the restriction operator. Certainly, it is possible to extend the method to the case where the bases are not identical. In the easiest version, we only perform coarsening in time, which is proven to converge without restrictions on the mesh-size h, the parameter θ, or polynomial degree p in case of A-stable time-stepping schemes; see [28]. Since the underlying method is A-stable without restrictions on the mesh-size h and the parameter θ at least for $p = 1$, we expect similar results as in [28] for higher degrees. The drawback of this strategy is that on the coarsest level, the number of dofs may still be quite large, especially if we consider relatively fine meshes for three-dimensional spatial domains. The more advanced strategy is to combine

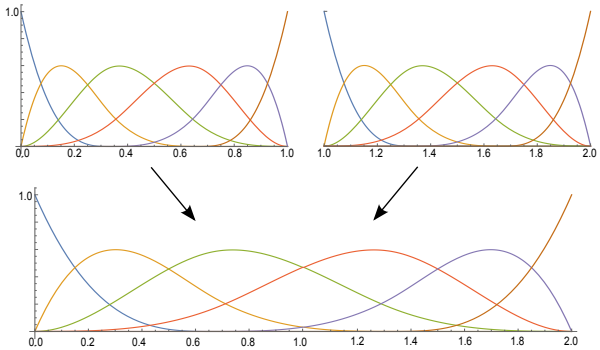

FIG. 2. *Illustration of time coarsening: two consecutive time-slabs are merged into a single time-slab.*

coarsening in space and time, but this has to be performed in a suitable way. In [28], for a one-dimensional spatial domain, it is shown that the factor $\tau_L h_L^{-2}$ must be large enough in order to guarantee convergence, where τ_L is the mesh-size on the time-slab on level L in the time direction and h_L the corresponding spatial mesh-size on level L. Hence, we have to fulfill an inequality of the form $\tau_L h_L^{-2} > c_{critical}$, where the threshold $c_{critical}$ is independent of h, τ, and L. Based on this inequality, we have to decide at each time-level whether we perform a full space-time coarsening or only a coarsening in time.

In this work, we are mostly interested in the smoother, which is an (inexact) damped block Jacobi smoother of the form

$$\boldsymbol{u}_h^{k+1} = \boldsymbol{u}_h^k + \omega \boldsymbol{D}_h^{-1} \left[\boldsymbol{f}_h - \boldsymbol{L}_h \boldsymbol{u}_h^k \right] \quad \text{for } k = 1, 2, \ldots.$$

We use the block-diagonal matrix $\boldsymbol{D}_h := \operatorname{diag}\{\boldsymbol{A}_n\}_{n=1}^N$ and the damping parameter $\omega = 0.5$; see also [11]. The application of the smoother can be accelerated by replacing the inverse of $\boldsymbol{D}_h$ by some approximation, i.e., an approximation $\hat{\boldsymbol{A}}_n^{-1}$ to $\boldsymbol{A}_n^{-1}$. The aim of this work is to find a procedure that allows an efficient application of $\hat{\boldsymbol{A}}_n^{-1}$ to a vector. In order to achieve this, we will heavily exploit the special tensor product structure of $\boldsymbol{A}_n$.

3.2. General construction of an approximation for $\boldsymbol{A}_n^{-1}$. In this section, for notational simplicity, we drop the subscript n when considering matrices and vectors defined on the space-time-slab Q_n. We recall the structure of the matrix $\boldsymbol{A}$,

$$\boldsymbol{A} = \boldsymbol{K}_t \otimes \boldsymbol{M}_x + \boldsymbol{M}_t \otimes \boldsymbol{K}_x,$$

where the matrices $\boldsymbol{M}_x$ and $\boldsymbol{K}_x$ are symmetric and positive definite, while the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ are nonsymmetric; cf. (2.13). The matrices $\boldsymbol{M}_x$ and $\boldsymbol{K}_x$ correspond to a d-dimensional spatial problem, whereas $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ are only related to a one-dimensional problem at one time-slab. Hence, the size of the latter two matrices is much smaller than the first two. The idea is to use already available preconditioners for symmetric and positive definite problems of the form $\boldsymbol{K}_x + \gamma \boldsymbol{M}_x$ with $\gamma > 0$ to construct efficient and robust preconditioners for $\boldsymbol{A}^{-1}$. The ideas of this section are based on the results developed in [36, 30].

We will achieve this by performing a decomposition of $\boldsymbol{M}_t^{-1} \boldsymbol{K}_t$ using one of the following three methods: *Diagonalization*, *Complex-Schur decomposition*, *Real-Schur decomposition*. Using these techniques, we obtain a decomposition of the form

$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{X}\boldsymbol{Z}\boldsymbol{X}^{-1}$, where the entries of the matrices $\boldsymbol{X}$ and $\boldsymbol{Z}$ are complex or real numbers, and $\boldsymbol{Z}$ has some sort of "simple" structure. A detailed specification will be presented in sections 3.3, 3.4, and 3.5.

By defining $\boldsymbol{Y} := (\boldsymbol{M}_t\boldsymbol{X})^{-1}$, we obtain the following representations:

$$\boldsymbol{M}_t = \boldsymbol{Y}^{-1}\boldsymbol{X}^{-1} \quad \text{and} \quad \boldsymbol{K}_t = \boldsymbol{Y}^{-1}\boldsymbol{Z}\boldsymbol{X}^{-1}.$$

Now we can rewrite $\boldsymbol{A}$ in the form

$$\begin{aligned}\boldsymbol{A} &= \boldsymbol{K}_t \otimes \boldsymbol{M}_x + \boldsymbol{M}_t \otimes \boldsymbol{K}_x \\ &= (\boldsymbol{Y}^{-1}\boldsymbol{Z}\boldsymbol{X}^{-1}) \otimes \boldsymbol{M}_x + (\boldsymbol{Y}^{-1}\boldsymbol{X}^{-1}) \otimes \boldsymbol{K}_x \\ &= (\boldsymbol{Y}^{-1} \otimes \boldsymbol{I}) \cdot (\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x) \cdot (\boldsymbol{X}^{-1} \otimes \boldsymbol{I}).\end{aligned}$$

Using the well-known fact that $(\boldsymbol{Y}^{-1} \otimes \boldsymbol{I})^{-1} = \boldsymbol{Y} \otimes \boldsymbol{I}$ and $(\boldsymbol{X}^{-1} \otimes \boldsymbol{I})^{-1} = \boldsymbol{X} \otimes \boldsymbol{I}$, we obtain

$$\boldsymbol{A}^{-1} = (\boldsymbol{X} \otimes \boldsymbol{I}) \cdot (\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} \cdot (\boldsymbol{Y} \otimes \boldsymbol{I}). \tag{3.1}$$

In the subsequent subsections, we will investigate the structure of the matrix $(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)$ for each of the three decomposition methods, and we will look for efficient ways of (approximate) inversion.

In the following, the generalized eigenvalues $\lambda_i := \alpha_i + \mathrm{i}\beta_i \in \mathbb{C}$ of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$, i.e.,

$$\boldsymbol{K}_t\boldsymbol{z}_i = \lambda_i\boldsymbol{M}_t\boldsymbol{z}_i, \tag{3.2}$$

with the eigenvector $\boldsymbol{z} := \boldsymbol{x}+\mathrm{i}\boldsymbol{y}$, will play an important role in constructing an efficient application of (3.1). First, for $0 < \theta_n \le C_{inv}^{-2}$, where C_{inv} denotes the constant from the inverse inequality

$$|v(t_{n-1})|^2 \le C_{inv}^2 h_n^{-1} \|v\|^2_{L_2(t_{n-1},t_n)} \; \forall v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t}, \tag{3.3}$$

we have the positiveness of the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$; see also [39] for an explicit formula of $C_{inv} = C_{inv}(p)$ in the case of polynomials of the degree p.

LEMMA 3.1. *Let $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ be given by* (2.14)*, and let the constant $C_{inv} > 0$ be defined according to* (3.3)*. If $\theta_n > 0$, then the matrix $\boldsymbol{K}_t$ is positive definite, i.e., $\boldsymbol{v}^T\boldsymbol{K}_t\boldsymbol{v} > 0$ for all $\boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$, and if $\theta_n < 2C_{inv}^{-2}$, then the matrix $\boldsymbol{M}_t$ is positive definite.*

Proof. We first consider the matrix $\boldsymbol{K}_t$. We can write $\boldsymbol{v}^T\boldsymbol{K}_t\boldsymbol{v}$ in the following way:

$$\begin{aligned}\boldsymbol{v}^T\boldsymbol{K}_t\boldsymbol{v} = (\boldsymbol{K}_t\boldsymbol{v}, \boldsymbol{v}) &= \int_{t_{n-1}}^{t_n} (v'(t)v(t) + \theta_n h_n (v'(t))^2)\, dt + |v(t_{n-1})|^2 \\ &= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\int_{t_{n-1}}^{t_n} (v^2)'(t)\, dt + |v(t_{n-1})|^2 \\ &= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}|v(t_n)|^2 - \frac{1}{2}|v(t_{n-1})|^2 + |v(t_{n-1})|^2 \\ &= \theta_n h_n \|v'\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}(|v(t_n)|^2 + |v(t_{n-1})|^2) > 0\end{aligned}$$

for all $v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$. Using (3.3), we similarly obtain

$$\begin{aligned}\boldsymbol{v}^T \boldsymbol{M}_t \boldsymbol{v} = (\boldsymbol{M}_t \boldsymbol{v}, \boldsymbol{v}) &= \int_{t_{n-1}}^{t_n} (v(t)^2 + \theta_n h_n v'(t) v(t))\, dt \\ &= \|v\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\theta_n h_n (|v(t_n)|^2 - |v(t_{n-1})|^2) \\ &\geq \left(1 - \frac{C_{inv}^2 \theta_n}{2}\right) \|v\|^2_{L_2(t_{n-1},t_n)} + \frac{1}{2}\theta_n h_n |v(t_n)|^2 > 0\end{aligned}$$

for all $v \leftrightarrow \boldsymbol{v} \in \mathbb{R}^{N_t} \setminus \{\boldsymbol{0}\}$. ∎

Next we are going to investigate the generalized eigenvalues in (3.2). More precisely, we want to find conditions under which the real part α is positive. However, for a generalized eigenvalue problem $\boldsymbol{A}\boldsymbol{z} = \lambda \boldsymbol{B}\boldsymbol{z}$, this does not follow from the positivity of $\boldsymbol{A}$ and $\boldsymbol{B}$ as the following example shows.

Example 3.1. Let the matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ be given by

$$\boldsymbol{A} = \begin{bmatrix} 5 & -2 \\ 13 & 18 \end{bmatrix} \quad \text{and} \quad \boldsymbol{B} = \begin{bmatrix} 4 & 10 \\ -10 & 9 \end{bmatrix}.$$

For the spectra, we have $\sigma(\boldsymbol{A}) = \{9 \pm 2\sqrt{5}\mathrm{i}\}$ and $\sigma(\boldsymbol{B}) = \{\frac{13}{2} \pm 5\sqrt{15}\mathrm{i}\}$. However, the generalized eigenvalues are $\sigma(\boldsymbol{B}^{-1}\boldsymbol{A}) = \{-\frac{103}{272} \pm \sqrt{4435}\mathrm{i}\}$.

Let $\boldsymbol{z}$ be the eigenvector to the eigenvalue $\lambda = \alpha + \mathrm{i}\beta$, i.e., $(\boldsymbol{A} - \lambda \boldsymbol{B})\boldsymbol{z} = 0$. Multiplying from the left with $(\boldsymbol{x} - \mathrm{i}\boldsymbol{y})^T$ yields

$$(\boldsymbol{x} - \mathrm{i}\boldsymbol{y})^T (\boldsymbol{A} - (\alpha + \mathrm{i}\beta)\boldsymbol{B})(\boldsymbol{x} + \mathrm{i}\boldsymbol{y}) = 0.$$

Separating the real and imaginary part, we obtain

$$\begin{aligned}&\alpha(\boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y}) - \beta(\boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}) = \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{A}\boldsymbol{y} \\ &\alpha(\boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}) + \beta(\boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y}) = \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y}.\end{aligned} \tag{3.4}$$

Introducing the abbreviations $a := \boldsymbol{x}^T \boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{A}\boldsymbol{y}$, $b := \boldsymbol{x}^T \boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T \boldsymbol{B}\boldsymbol{y}$, $c := \boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y}$, and $d := \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y}$, we can rewrite this system in the compact form

$$\begin{bmatrix} b & -c \\ c & b \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = \begin{bmatrix} a \\ d \end{bmatrix},$$

and α is then given by the formula

$$\alpha = \frac{1}{b^2 + c^2}(ab + cd). \tag{3.5}$$

We can easily observe the statements of the following lemma.

LEMMA 3.2. *Let $\boldsymbol{A}$ and $\boldsymbol{B}$ be positive definite matrices; then the following statements hold:*

1. *$a > 0$ and $b > 0$.*
2. *If $\beta = 0$, i.e., the eigenvalue $\lambda \in \mathbb{R}$, then $\lambda = \alpha > 0$.*
3. *If either $\boldsymbol{A}$ or $\boldsymbol{B}$ is symmetric, then $\alpha > 0$.*

If $\boldsymbol{A}$ is only nonnegative, then these inequalities hold with $\geq$ instead of $>$.

Proof. The positivity of a and b immediately follows from the definition. If the eigenvalue λ is real, i.e., $\beta = 0$, we obtain from the first equation of (3.4) that $\alpha = a/b > 0$. If either $\boldsymbol{A}$ or $\boldsymbol{B}$ is symmetric, then either d or c is zero. Hence, by (3.5), α is positive. ∎

Let us now consider the special case of $\boldsymbol{A} = \boldsymbol{K}_t$ and $\boldsymbol{B} = \boldsymbol{M}_t$. For notational simplicity we consider the interval $[0, T]$. First, we observe that

$$c = \boldsymbol{x}^T(\boldsymbol{B} - \boldsymbol{B}^T)\boldsymbol{y} = \theta h \int_0^T y'(t)x(t) - x'(t)y(t)\,dt,$$
$$d = \boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y} = \int_0^T x'(t)y(t) - y'(t)x(t)\,dt.$$

Hence, it follows that $c = -\theta h d$. This relation leads to the following formula for α:

$$\alpha = \frac{1}{b^2 + c^2}(ab - \theta h d^2). \tag{3.6}$$

The problem then reduces to checking the relation $ab - \theta h d^2 > 0$, which then reads

$$(\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y})(\boldsymbol{x}^T\boldsymbol{B}\boldsymbol{x} + \boldsymbol{y}^T\boldsymbol{B}\boldsymbol{y}) - \theta h(\boldsymbol{x}^T(\boldsymbol{A} - \boldsymbol{A}^T)\boldsymbol{y})^2 > 0 \tag{3.7}$$

for the eigenvector $\boldsymbol{z} = \boldsymbol{x} + \mathrm{i}\boldsymbol{y}$ corresponding to $\lambda = \alpha + \mathrm{i}\beta$. Rewriting (3.7) in terms of functions, we get the relation

$$\begin{aligned}
&\left(\theta_n h_n \|x'\|^2 + \frac{1}{2}(|x(T)|^2 + |x(0)|^2) + \theta_n h_n \|y'\|^2 + \frac{1}{2}(|y(T)|^2 + |y(0)|^2)\right) \\
&\cdot \left(\|x\|^2 + \frac{1}{2}\theta_n h_n(|x(T)|^2 - |x(0)|^2) + \|y\|^2 + \frac{1}{2}\theta_n h_n(|y(T)|^2 - |y(0)|^2)\right) \\
&- \theta h \left(\int_0^T x'(t)y(t) - y'(t)x(t)\,dt\right)^2 > 0.
\end{aligned}$$

Unfortunately, in this work we cannot give a complete characterization of the conditions under which the last inequality holds.

Let us consider the special case $\theta = 0$. We note that $\boldsymbol{v}^T\boldsymbol{K}_t\boldsymbol{v} = \frac{1}{2}(|v(t_{n-1})|^2 + |v(t_n)|^2)$, which then only defines a seminorm. Hence, discrete coercivity is not valid. Therefore, this case is not covered by the analysis presented in [15]. For its analysis, we refer to [31], where an inf-sup condition and error estimates are proven. The matrix $\boldsymbol{M}_t$ is symmetric and $\boldsymbol{v}^T\boldsymbol{M}_t\boldsymbol{v} = \|v\|_{L^2}^2$. From this fact, we can deduce the following statement by means of Lemma 3.2.

PROPOSITION 3.3. *Let $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ be as defined above with $\theta = 0$. Then $\alpha \geq 0$.*

Remark 3.4. In the condition number analysis of the following subsections, we consider matrices of the form $\boldsymbol{K}_x + \alpha\boldsymbol{M}_x$, which are required to be positive definite. Therefore, the positivity of α can be relaxed in the case $|\Gamma_D| > 0$.

Remark 3.5. A more detailed investigation of (3.4) shows that

$$\alpha = 0 \Longleftrightarrow x(0) = x(T) = y(0) = y(T) = 0 \tag{3.8}$$

for the eigenvector $\boldsymbol{z} = \boldsymbol{x} + \mathrm{i}\boldsymbol{y}$ corresponding to $\alpha + \mathrm{i}\beta$.

For the case $p = 1$, we can even show that the property $x(0) = x(T) = y(0) = y(T) = 0$ cannot hold for an eigenvector corresponding to a purely imaginary eigenvalue. Considering a uniform knot vector in $[0, 1]$ with B-Splines of degree $p = 1$ and $N_t \geq 3$, we see that

$$\boldsymbol{K}_t = \frac{1}{2}\begin{bmatrix} 1 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 1 \end{bmatrix} \quad \text{and} \quad \boldsymbol{M}_t = C_n \begin{bmatrix} 2 & 1 & & & \\ 1 & 4 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 4 & 1 \\ & & & 1 & 2 \end{bmatrix},$$

where $C_n > 0$ depends on N_t. Rewriting $\boldsymbol{K}_t \boldsymbol{z} = \mathrm{i}\beta \boldsymbol{M}_t \boldsymbol{z}$ as recurrence relation for $\boldsymbol{z} = [z_1, z_2, \ldots, z_{N_t-1}, z_{N_t}]$, we obtain the equations

$$\begin{aligned} z_1 + z_2 &= \mathrm{i}\beta(2z_1 + z_2), \\ -z_{i-1} + z_{i+1} &= \mathrm{i}\beta(z_{i-1} + 4z_i + z_{i+1}), \quad i = 2, \ldots, N_t - 1, \\ -z_{N_t-1} + z_{N_t} &= \mathrm{i}\beta(z_{N_t-1} + 2z_{N_t}), \end{aligned} \tag{3.9}$$

where we include C_n and $1/2$ into the eigenvalue $\mathrm{i}\beta$. In order to ensure that $\boldsymbol{z} = [0, z_2, \ldots, z_{N_t-1}, 0]$ is an eigenvector, it must fulfill

$$z_2 = \mathrm{i}\beta z_2 \Leftrightarrow (1 - \mathrm{i}\beta) z_2 = 0,$$

which results from the first line of (3.9). Since $(1 - \mathrm{i}\beta)$ cannot be zero, it follows that $z_2 = 0$. Considering now the second line of (3.9) and assuming $z_1 = \cdots = z_j = 0$, we observe that

$$z_{j+1} = \mathrm{i}\beta z_{j+1} \Leftrightarrow (1 - \mathrm{i}\beta) z_{j+1} = 0$$

for $i = j$. Therefore, $z_{j+1} = 0$. By induction, it follows that $z = 0$. Hence, it cannot be an eigenvector.

In the case of $p > 1$, the matrices $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$ have more than one off-diagonal and such a relation would not follow so easily. Numerical experiments in section 4.2 indicate that the real part of λ is positive for the case $p > 1$, too.

Remark 3.6. Let us consider the case $|\Gamma_D| > 0$. From Remark 3.4, Proposition 3.3, and the continuous dependence of α on θ, we obtain that $\boldsymbol{K}_x + \alpha \boldsymbol{M}_x$ must be positive definite for sufficiently small θ.

Remark 3.7. Numerical experiments performed for various values of θ, p, and h_n in section 4.2 indicate that the generalized eigenvalues λ_i have a positive real part α provided that the real part of the eigenvalues of $\boldsymbol{M}_t$ is positive. Moreover, in the practical implementation, one has to compute the eigenvalues λ_i anyway. Therefore, we always have an a posteriori control on the positivity of α. If it happens that $\alpha \leq 0$, then we have to use a smaller θ.

3.3. Diagonalization. If the matrix $\boldsymbol{M}_t^{-1} \boldsymbol{K}_t$ is diagonalizable, the eigenvalue decomposition allows us to write

$$\boldsymbol{M}_t^{-1} \boldsymbol{K}_t = \boldsymbol{X} \boldsymbol{D} \boldsymbol{X}^{-1}, \tag{3.10}$$

where $\boldsymbol{D} = \mathrm{diag}(\lambda_i)$, $\lambda_i \in \mathbb{C}$, is a diagonal matrix with possibly complex eigenvalues

on the diagonal, and $\boldsymbol{X} \in \mathbb{C}^{N_t \times N_t}$ denotes the matrix of the possibly complex eigenvectors. Due to the fact that the matrix $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is nonsymmetric, the eigenvectors do not form an orthogonal basis, i.e., $\boldsymbol{X}^{-1} \neq \boldsymbol{X}^*$. An efficient calculation can be performed by means of solving the generalized eigenvalue problem $\boldsymbol{K}_t x = \lambda \boldsymbol{M}_t x$.

Thanks to (3.10), the matrix $(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}$ from (3.1) then takes the form

$$(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} = (\boldsymbol{D} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1} = \operatorname{diag}_{i=1,\dots,N_t}((\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)^{-1}).$$

Therefore, only N_t problems of the form $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)$ have to be solved, independently of each other. We have to distinguish two cases: the first case where the eigenvalue λ_i is a positive real number, and the second one where λ_i is a complex number.

In the first case, we consider $\lambda_i = \alpha_i \in \mathbb{R}^+$. In this case the matrix $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$ is symmetric and positive definite (SPD). This allows for many efficient exact and inexact solution strategies. Indeed, linear algebraic systems of IgA equations with an SPD system matrix $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$ can efficiently be solved by different robust solution strategies like MG [35, 17, 8], Domain Decomposition type methods [14, 4], or Fast Diagonalization type methods [30, 36].

The second case, where $\lambda_i = \alpha_i + \mathrm{i}\beta_i \in \mathbb{C}$ with $\alpha_i, \beta_i \in \mathbb{R}, \alpha_i > 0$, is more difficult to handle. We note that $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)^* \neq \boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$. Separating the real and imaginary parts, we can rewrite the complex system $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z} = \boldsymbol{h}$ as a real system with a real block system matrix of doubled size:

$$\begin{aligned} & (\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z} = \boldsymbol{h} \\ \iff & \begin{bmatrix} \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x & -\beta_i \boldsymbol{M}_x \\ \beta_i \boldsymbol{M}_x & \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\ \iff & \underbrace{\begin{bmatrix} \boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x & \beta_i \boldsymbol{M}_x \\ \beta_i \boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha_i \boldsymbol{M}_x) \end{bmatrix}}_{=:\overline{\boldsymbol{A}}_i} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix}, \end{aligned}$$

where $\boldsymbol{z} = \boldsymbol{x} + \mathrm{i}\boldsymbol{y}$ and $\boldsymbol{h} = \boldsymbol{f} + \mathrm{i}\boldsymbol{g}$. The matrix $\overline{\boldsymbol{A}}_i \in \mathbb{R}^{2N_x \times 2N_x}$ is symmetric, but indefinite. We are now looking for a robust preconditioner for $\overline{\boldsymbol{A}}_i$. The construction of the preconditioner is motivated by operator interpolation techniques; see, e.g., [41, 5, 1].

THEOREM 3.8. *Let $\boldsymbol{K}_x$ and $\boldsymbol{M}_x$ be symmetric and positive definite matrices, and let α and β be real numbers with $\alpha > 0$. Furthermore, we define the block matrices*

$$\overline{\boldsymbol{A}} := \begin{bmatrix} \boldsymbol{K}_x + \alpha \boldsymbol{M}_x & \beta \boldsymbol{M}_x \\ \beta \boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) \end{bmatrix}, \tag{3.11}$$

$$\boldsymbol{P} := \begin{bmatrix} \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x & 0 \\ 0 & \boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x \end{bmatrix}. \tag{3.12}$$

Then the condition number estimate $\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) := \|\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}\|_{\boldsymbol{P}} \|\boldsymbol{A}^{-1}\overline{\boldsymbol{P}}\|_{\boldsymbol{P}} \leq \sqrt{2}$ holds, where $\|\cdot\|_{\boldsymbol{P}}$ denotes the matrix norm that is associated with the $\boldsymbol{P}$-energy vector norm $\|\cdot\|_{\boldsymbol{P}} := (\boldsymbol{P}\cdot,\cdot)^{0.5}$ generated by the SPD matrix $\boldsymbol{P}$.

In order to prove this theorem, one could either use results from interpolation theory and obtain an additional factor $(\sqrt{5}+1)/(\sqrt{5}-1)$ in the condition number bound (see, e.g., [40]) or directly consider the generalized eigenvalue problem $\overline{\boldsymbol{A}}x = \lambda \boldsymbol{P}x$, which gives $\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) \leq \sqrt{2}$. The proof follows the procedure outlined in Remark 9 in [41], and will be presented for the matrices obtained from a Real Schur decomposition; see Theorem 3.9. The result above follows analogously.

We note that both block-diagonal entries of $\boldsymbol{P}$ are identical, and the matrix $\boldsymbol{K}_x + (\alpha + |\beta|)\boldsymbol{M}_x$ is symmetric and positive definite. This opens various possibilities for preconditioning based on standard techniques for symmetric and positive definite matrices. We can efficiently solve the linear system with system matrix $\overline{\boldsymbol{A}}$ by means of MINRES preconditioned by $\boldsymbol{P}^{-1}$. We can even use a spectral equivalent approximation $\hat{\boldsymbol{P}}^{-1}$ such that $c\hat{\boldsymbol{P}}^{-1} \leq \boldsymbol{P}^{-1} \leq C\hat{\boldsymbol{P}}^{-1}$, with constants c and C, independent of α and β. Note that, although the number N_t might be small, e.g., less than 30, we have to solve for each i an independent spatial problem, possibly with a huge number of dofs. Therefore, this approach allows for a further parallelization by applying $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)^{-1}$ in parallel for $i = 1, \ldots, N_t$.

Note that the eigenvalue decomposition (3.10) has to be calculated on each time-slab only for a matrix of size $N_t \times N_t$. This greatly reduces the complexity of the operations. Moreover, often the matrices K_t and M_t are identical on each time-slab, so the decomposition can be reused.

Unfortunately, this approach has a severe drawback. Due to the fact that the matrix $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is nonsymmetric, the matrix $\boldsymbol{X}$ of eigenvectors is not unitary and, therefore, $\kappa(\boldsymbol{X}) \neq 1$. Actually, numerical tests in section 4.1 show that, for large B-Spline degree or small h_t, we observe large condition numbers of the matrix $\boldsymbol{X}$. In that case, we cannot correctly apply (3.1) and the algorithm fails. This problem can be circumvented either by splitting the time-slab, hence reducing the number of dofs, or by using the Complex or Real-Schur decomposition, as presented in the subsequent two subsections.

3.4. Complex-Schur decomposition. In this section, we investigate an alternative technique for decomposing $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$. The Complex-Schur decomposition provides a decomposition of the form

$$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{Q}\boldsymbol{T}\boldsymbol{Q}^*, \tag{3.13}$$

where $\boldsymbol{Q} \in \mathbb{C}^{N_t \times N_t}$, and $\boldsymbol{T} \in \mathbb{C}^{N_t \times N_t}$ is an upper triangular matrix with $T_{ii} = \lambda_i$. The advantage of the (Complex) Schur decomposition is the fact that we obtain a unitary matrix $\boldsymbol{Q}$. Hence, $\kappa(\boldsymbol{Q}) = 1$, but the diagonal matrix $\boldsymbol{D}$ in the decomposition (3.10) is now replaced by the upper triangular matrix $\boldsymbol{T}$ in the decomposition (3.13), By means of (3.13), the matrix $\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x$ from (3.1) takes the form

$$(\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x) = \begin{bmatrix} \boldsymbol{K}_x + \lambda_1 \boldsymbol{M}_x & T_{12}\boldsymbol{M}_x & \cdots & \\ 0 & \boldsymbol{K}_x + \lambda_2 \boldsymbol{M}_x & T_{23}\boldsymbol{M}_x & \\ \vdots & 0 & \ddots & T_{N_t N_t - 1}\boldsymbol{M}_x \\ 0 & \cdots & 0 & \boldsymbol{K}_x + \lambda_{N_t}\boldsymbol{M}_x \end{bmatrix}.$$

The application of $(\boldsymbol{T} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}$ to some vector f can be performed in a staggered way as presented in Algorithm 3.1.

Algorithm 3.1 Calculation of $\boldsymbol{z} = (\boldsymbol{T} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x)^{-1}\boldsymbol{h}$.

for $i = N_t, N_t - 1, \ldots, 1$ **do**
  $\hat{\boldsymbol{h}} = \boldsymbol{h}_i$
  for $j = i + 1, i + 2, \ldots, N_t$ **do**
    $\hat{\boldsymbol{h}} = \hat{\boldsymbol{h}}$- $T_{ij}\boldsymbol{z}_j$
  end for
  SOLVE $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z}_i = \hat{\boldsymbol{h}}$, where $\lambda_i = T_{ii}$.
end for
return $\boldsymbol{z}$

In order to solve the linear systems $(\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x)\boldsymbol{z}_i = \hat{\boldsymbol{h}}_i$, $i = 1, \ldots, N_t$, in Algorithm 3.1 we can use the techniques developed in the previous subsection. This decomposition method provides a well-conditioned transformation matrix Q. However, the linear systems cannot be solved independently of each other. We note that this method and the eigenvalue decomposition require complex arithmetic, which is more expensive than the real one. In the following subsection, we investigate the Real-Schur decomposition, which eliminates the need for having complex arithmetic.

3.5. Real-Schur decomposition. In this subsection, we look at the decomposition of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ by means of the Real-Schur decomposition of the form

$$\boldsymbol{M}_t^{-1}\boldsymbol{K}_t = \boldsymbol{Q}\boldsymbol{T}\boldsymbol{Q}^T, \tag{3.14}$$

where $\boldsymbol{Q} \in \mathbb{R}^{N_t \times N_t}$. The matrix $\boldsymbol{T} \in \mathbb{R}^{N_t \times N_t}$ is an upper quasi-triangular matrix, i.e., the diagonal consists of 1×1 and 2×2 blocks. The values of the 1×1 blocks correspond to the real eigenvalues, while the 2×2 blocks correspond to the complex eigenvalues of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$.

By additionally performing a Givens rotation, the 2×2 block can be transformed to the structure $\left[\begin{smallmatrix} \alpha & \beta_1 \\ \beta_2 & \alpha \end{smallmatrix}\right]$, where $\alpha, \beta_1, \beta_2 \in \mathbb{R}$ and $\beta_1 \neq \beta_2 \neq 0$. The eigenvalues of this matrix are given by $\alpha \pm \sqrt{\beta_1\beta_2}$. Due to the fact that the eigenvalues have to be complex and the real part has to be positive, we obtain that $\alpha > 0$ and β_1 and β_2 have different signs. Therefore, we can write the eigenvalues as $\alpha \pm \mathrm{i}\sqrt{|\beta_1\beta_2|}$.

Using this decomposition, the matrix $\boldsymbol{Z} \otimes \boldsymbol{M}_x + \boldsymbol{I} \otimes \boldsymbol{K}_x$ appearing in (3.1) has a structure, which is similar to that of the Complex-Schur decomposition. The corresponding system of linear algebraic equations can again be solved in a staggered way as presented in Algorithm 3.1. We have to adapt the algorithm in the following way. If the diagonal block is a 2×2 block, then we have to work with two-block vectors and a 2×2 block matrix. It remains to investigate the solution strategy for

the 2×2 block matrix. As already mentioned, the 2×2 block of T is nonsymmetric. Hence, the 2×2 block matrix is also nonsymmetric and is given in the following way:

$$\begin{bmatrix} \boldsymbol{K}_x + \alpha \boldsymbol{M}_x & \beta_1 \boldsymbol{M}_x \\ \beta_2 \boldsymbol{M}_x & \boldsymbol{K}_x + \alpha \boldsymbol{M}_x \end{bmatrix}.$$

The structure of the matrix is very similar to $\overline{\boldsymbol{A}}$ in Theorem 3.8 up to the nonsymmetry, which originates just from the different scalings β_1 and β_2 as well as their different sign. By a proper rescaling, we can transform this linear system into an equivalent system with a symmetric, but indefinite system matrix:

$$\begin{aligned} & \begin{bmatrix} \boldsymbol{K}_x + \alpha \boldsymbol{M}_x & \beta_1 \boldsymbol{M}_x \\ \beta_2 \boldsymbol{M}_x & \boldsymbol{K}_x + \alpha \boldsymbol{M}_x \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\ \iff & \begin{bmatrix} \boldsymbol{K}_x + \alpha \boldsymbol{M}_x & -\beta_1 \boldsymbol{M}_x \\ \beta_2 \boldsymbol{M}_x & -(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} = \begin{bmatrix} \boldsymbol{f} \\ \boldsymbol{g} \end{bmatrix} \\ \iff & \underbrace{\begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) & -\beta_1|\beta_2| \boldsymbol{M}_x \\ |\beta_1|\beta_2 \boldsymbol{M}_x & -|\beta_1|(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) \end{bmatrix}}_{=:\overline{\boldsymbol{A}}} \begin{bmatrix} \boldsymbol{x} \\ -\boldsymbol{y} \end{bmatrix} = \begin{bmatrix} |\beta_2| \boldsymbol{f} \\ |\beta_1| \boldsymbol{g} \end{bmatrix}. \end{aligned}$$

We note that β_1 and β_2 have different signs. Hence, $\beta_1|\beta_2| = -\beta_2|\beta_1|$. Motivated by the construction of the preconditioner in the case of the eigenvalue decomposition, we can come up with an optimal preconditioner. The following theorem presents this optimal preconditioner for the matrix $\overline{\boldsymbol{A}}$.

THEOREM 3.9. *Let $\boldsymbol{K}_x$ and $\boldsymbol{M}_x$ be symmetric and positive matrices, and let α, β_1, β_2 be real numbers with $\alpha > 0$. Furthermore, we define the block matrices*

$$\begin{aligned} \overline{\boldsymbol{A}} &:= \begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) & -\beta_1|\beta_2| \boldsymbol{M}_x \\ |\beta_1|\beta_2 \boldsymbol{M}_x & -|\beta_1|(\boldsymbol{K}_x + \alpha \boldsymbol{M}_x) \end{bmatrix}, \\ \boldsymbol{P} &:= \begin{bmatrix} |\beta_2|(\boldsymbol{K}_x + (\alpha + \sqrt{|\beta_1\beta_2|}) \boldsymbol{M}_x) & 0 \\ 0 & |\beta_1|(\boldsymbol{K}_x + (\alpha + \sqrt{|\beta_1\beta_2|}) \boldsymbol{M}_x) \end{bmatrix}. \end{aligned}$$

Then the condition number estimate $\kappa_P(\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}) \leq \sqrt{2}$ holds.

Proof. The proof follows the lines from Remark 9 in [41], which gives a sharper bound than using interpolation theory as in [40]. For notational simplicity, we introduce the abbreviations $\mathcal{K} := \boldsymbol{K}_x + \alpha \boldsymbol{M}_x$ and $\mathcal{M} := \boldsymbol{M}_x$. We now investigate the generalized eigenvalue problem $\overline{\boldsymbol{A}}\boldsymbol{u} = \lambda \boldsymbol{P}\boldsymbol{u}$, which reads

(3.15)
$$\begin{bmatrix} |\beta_2|\mathcal{K} & -\beta_1|\beta_2|\mathcal{M} \\ |\beta_1|\beta_2\mathcal{M} & -|\beta_1|\mathcal{K} \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix} = \lambda \begin{bmatrix} |\beta_2|(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}) & 0 \\ 0 & |\beta_1|(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}) \end{bmatrix} \begin{bmatrix} \boldsymbol{x} \\ \boldsymbol{y} \end{bmatrix}.$$

At first we consider the following generalized eigenvalue problem:

$$\mathcal{K}\boldsymbol{z} = \mu(\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M})\boldsymbol{z}.$$

Due to the fact that $\mathcal{K}$ and $\mathcal{M}$ are symmetric, there exists a basis $\{\boldsymbol{e}_1, \boldsymbol{e}_2, \ldots, \boldsymbol{e}_{N_x}\}$ of eigenvectors, which are orthonormal with respect to the inner product generated by $\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}$, and corresponding eigenvalues μ_j. Since $\mathcal{K}$ is dominated by $\mathcal{K} + \sqrt{|\beta_1\beta_2|}\mathcal{M}$ and due to their positivity, we have that $\mu_j \in [0, 1]$. Therefore,

we can express $\boldsymbol{x}$ and $\boldsymbol{y}$ as a linear combination of $\boldsymbol{e}_j$ with coefficients $\hat{x}_j$ and $\hat{y}_j$, respectively. Moreover, $\mathcal{M}\boldsymbol{z}$ fulfills the following identity:

$$\begin{aligned}\mathcal{M}\boldsymbol{z} &= (|\beta_1\beta_2|)^{-1/2}(\sqrt{|\beta_1\beta_2|}\mathcal{M}+\mathcal{K})\boldsymbol{z} - (|\beta_1\beta_2|)^{-1/2}\mathcal{K}\boldsymbol{z}\\ &= (|\beta_1\beta_2|)^{-1/2}(\sqrt{|\beta_1\beta_2|}\mathcal{M}+\mathcal{K})\boldsymbol{z} - (|\beta_1\beta_2|)^{-1/2}\mu(\mathcal{K}+\sqrt{|\beta_1\beta_2|}\mathcal{M})\boldsymbol{z}\\ &= (|\beta_1\beta_2|)^{-1/2}(1-\mu)(\sqrt{|\beta_1\beta_2|}\mathcal{M}+\mathcal{K})\boldsymbol{z}.\end{aligned}$$

Using the expansion of $\boldsymbol{x}$ and $\boldsymbol{y}$ into the eigenvectors $\{\boldsymbol{e}_j\}$, system (3.15) decomposes into the 2×2 systems

$$\begin{bmatrix} |\beta_2|\mu_j & -\beta_1|\beta_2||\beta_1\beta_2|^{-1/2}(1-\mu_j)\\ |\beta_1|\beta_2|\beta_1\beta_2|^{-1/2}(1-\mu_j) & -|\beta_1|\mu_j\end{bmatrix}\begin{bmatrix}\hat{x}_j\\ \hat{y}_j\end{bmatrix} = \lambda\begin{bmatrix}|\beta_2| & 0\\ 0 & |\beta_1|\end{bmatrix}\begin{bmatrix}\hat{x}_j\\ \hat{y}_j\end{bmatrix}.$$

Since there exists at least one pair $(\hat{x}_j,\hat{y}_j)$ which is nonzero, the determinant of the system matrix must be zero, i.e.,

$$\det\left(\begin{bmatrix} |\beta_2|\mu_j & -\beta_1|\beta_2||\beta_1\beta_2|^{-1/2}(1-\mu_j)\\ |\beta_1|\beta_2|\beta_1\beta_2|^{-1/2}(1-\mu_j) & -|\beta_1|\mu_j\end{bmatrix} - \lambda\begin{bmatrix}|\beta_2| & 0\\ 0 & |\beta_1|\end{bmatrix}\right) = 0,$$

which reduces to $|\beta_1\beta_2|(\lambda^2-\mu_j^2) - (\beta_1\beta_2)^2|\beta_1\beta_2|^{-1}(1-\mu_j)^2 = 0$, where we used that $-\beta_1|\beta_2| = |\beta_1|\beta_2 \neq 0$. We immediately obtain that $|\lambda| = \sqrt{\mu_j^2+(1-\mu_j)^2}$ for $\mu_i \in [0,1]$, and it follows that $\frac{1}{\sqrt{2}} \leq |\lambda| \leq 1$, which gives the desired bound on the condition number of $\boldsymbol{P}^{-1}\overline{\boldsymbol{A}}$. ∎

Now we can again use the MINRES preconditioned by $\boldsymbol{P}$ as iterative solver for systems with the system matrix $\overline{\boldsymbol{A}}$, and we obtain a robust method. Moreover, due to the use of real arithmetic, this approach is usually more efficient than that using the Complex-Schur decomposition.

As already noted at the end of section 3.3, the Complex- or Real-Schur decomposition of an $N_t \times N_t$ matrix has to be calculated on each time-slab. For completeness, the setup phase and the application of $\boldsymbol{v} = \boldsymbol{D}_h^{-1}\boldsymbol{u}$ are summarized in Algorithm 3.2 and Algorithm 3.3, respectively.

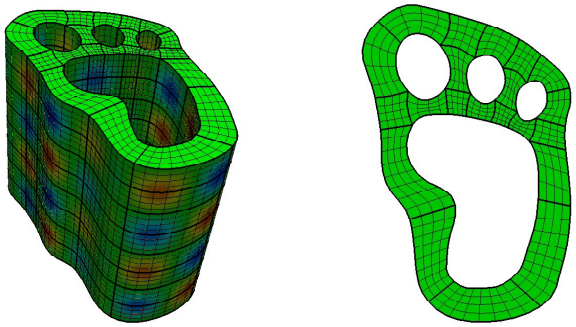

FIG. 3. *Left picture shows the space-time domain $Q = \Omega \times (0,T)$ with eight time-slabs; the colors indicate the contours of* (4.1). *Right picture presents the spatial domain Ω, consisting of* 21 *patches. (Figure in color online.)*

Algorithm 3.2 Setup for application of $\boldsymbol{D}_h^{-1}\boldsymbol{u} = \operatorname{diag}(\boldsymbol{A}_n^{-1})\boldsymbol{u}$.

Algorithm stores: $\boldsymbol{X}_n, \boldsymbol{Y}_n, (\boldsymbol{D}_n$ or $\boldsymbol{T}_n)$, solvers for $(\boldsymbol{K}_{n,x} + \gamma_{n,i}\boldsymbol{M}_{n,x})^{-1}$, $\gamma_{n,i} > 0, (\boldsymbol{P}_{n,i}^{diag})^{-1}, (\boldsymbol{P}_{n,i}^{cSchur})^{-1}, (\boldsymbol{P}_{n,i}^{rSchur})^{-1}$ for $n = 1, \dots, N$, $i = 1, \dots, N_{n,t}$
for $n = 1, \dots, N$ **do**
  if method = diagonalization **or** method = complex Schur **then**
    if method = diagonalization **then**
      Calculate $(\boldsymbol{X}_n, \boldsymbol{D}_n) : \boldsymbol{M}_{n,t}^{-1}\boldsymbol{K}_{n,t} = \boldsymbol{X}_n\boldsymbol{D}_n\boldsymbol{X}_n^{-1}$
    else if method = complex Schur **then**
      Calculate $(\boldsymbol{Q}_n, \boldsymbol{T}_n) : \boldsymbol{M}_{n,t}^{-1}\boldsymbol{K}_{n,t} = \boldsymbol{Q}_n\boldsymbol{T}_n\boldsymbol{Q}_n^T$
      $\boldsymbol{X}_n = \boldsymbol{Q}_n$
    end if
    for $i = 1, \dots, N_{n,t}$ **do**
      $\lambda_{n,i} = \alpha_{n,i} + \mathrm{i}\beta_{n,i} = (\boldsymbol{D}_n)_{ii}$ or $(\boldsymbol{T}_n)_{ii}$
      if $\alpha_{n,i} \le 0$ **then**
        restart with smaller θ_n
      end if
      if $\lambda_{n,i} \in \mathbb{R}$ **then**
        Setup solver for $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})^{-1}$
      else
        Setup solver $\boldsymbol{L}_{n,i}^{-1}$ for $(\boldsymbol{K}_{n,x} + (\alpha_{n,i} + |\beta_{n,i}|)\boldsymbol{M}_{n,x})^{-1}$
        $(\boldsymbol{P}_{n,i}^{cSchur})^{-1}$ or $(\boldsymbol{P}_{n,i}^{diag})^{-1} = \begin{bmatrix} \boldsymbol{L}_{n,i}^{-1} & 0 \\ 0 & \boldsymbol{L}_{n,i}^{-1} \end{bmatrix}$
        Reuse this for the complex conjugate $\overline{\lambda}_{n,i}$
      end if
    end for
  else if method = real Schur **then**
    Calculate $(\boldsymbol{Q}_n, \boldsymbol{T}_n) : \boldsymbol{M}_{n,t}^{-1}\boldsymbol{K}_{n,t} = \boldsymbol{Q}_n\boldsymbol{T}_n\boldsymbol{Q}_n^T$
    Givens rotations: 2×2 diagonal blocks of $\boldsymbol{T}_n \to \begin{bmatrix} \alpha_{n,i} & \beta_{1,n,i} \\ \beta_{2,n,i} & \alpha_{n,i} \end{bmatrix}, \tilde{\boldsymbol{Q}}_n$
    $\boldsymbol{X}_n = \tilde{\boldsymbol{Q}}_n$
    loop All diagonal blocks of $\boldsymbol{T}_n$
      if $\alpha_{n,i} \le 0$ **then**
        restart with smaller θ_n
      end if
      if 1×1 block **then**
        Setup solver for $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})^{-1}$
      else
        Setup solver $\boldsymbol{L}_{n,i}^{-1}$ for $(\boldsymbol{K}_{n,x} + (\alpha_{n,i} + \sqrt{|\beta_{1,n,i}\beta_{2,n,i}|})\boldsymbol{M}_{n,x})^{-1}$
        $(\boldsymbol{P}_{n,i}^{rSchur})^{-1} = \begin{bmatrix} |\beta_{2,n,i}|^{-1}\boldsymbol{L}_{n,i}^{-1} & 0 \\ 0 & |\beta_{1,n,i}|^{-1}\boldsymbol{L}_{n,i}^{-1} \end{bmatrix}$
      end if
    end loop
  end if
  $\boldsymbol{Y}_n = (\boldsymbol{M}_{n,t}\boldsymbol{X}_n)^{-1}$
end for

Algorithm 3.3 Application of $\boldsymbol{v} = \boldsymbol{D}_h^{-1}\boldsymbol{u} = \mathrm{diag}(\boldsymbol{A}_n^{-1})\boldsymbol{u}$.

Notation: $\boldsymbol{h}_i = \boldsymbol{f}_i + \mathrm{i}\boldsymbol{g}_i, \hat{\boldsymbol{h}}_i = \hat{\boldsymbol{f}}_i + \mathrm{i}\hat{\boldsymbol{g}}_i, \boldsymbol{z}_i = \boldsymbol{x}_i + \mathrm{i}\boldsymbol{y}_i$,
for $n = 1, \dots, N$ **do**
  $\boldsymbol{u}_n$ = all entries from $\boldsymbol{u}$ corresponding to slab n
  $\boldsymbol{h}_n = (\boldsymbol{Y}_n \otimes \boldsymbol{I}_n)\boldsymbol{u}_n$
  if method = diagonalization **then**
    for $i = 1, \dots, N_{n,t}$ **do**
      $\boldsymbol{h}_i$ = restriction of $\boldsymbol{h}_n$ corresponding to $\psi_i^n(t)$
      if $\lambda_{n,i} \in \mathbb{R}$ **then**
        Solve $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{x}_i = \boldsymbol{f}_i$
        Solve $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{y}_i = \boldsymbol{g}_i$
      else
        Solve with preconditioner $\boldsymbol{P}_{n,i}^{diag}$:
        $$\begin{bmatrix} \boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x} & \beta_{n,i}\boldsymbol{M}_x \\ \beta_{n,i}\boldsymbol{M}_x & -(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x}) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{y}_i \end{bmatrix} = \begin{bmatrix} \boldsymbol{f}_i \\ \boldsymbol{g}_i \end{bmatrix}$$
        $\boldsymbol{y}_i = -\boldsymbol{y}_i$
      end if
    end for
    build up $\boldsymbol{z}_n$ from $\boldsymbol{z}_1, \dots, \boldsymbol{z}_{N_{n,t}}$
  else if method = complex Schur **then**
    $\boldsymbol{z}_n$ = apply Algorithm 3.1 to $\boldsymbol{h}_n$, where
    function SOLVE $(\boldsymbol{K}_{n,x} + \lambda_i\boldsymbol{M}_{n,x})\boldsymbol{z}_i = \hat{\boldsymbol{h}}_i$:
      if $\lambda_{n,i} \in \mathbb{R}$ **then**
        Solve $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{x}_i = \hat{\boldsymbol{f}}_i$
        Solve $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{y}_i = \hat{\boldsymbol{g}}_i$
      else
        As for method = diagonalization
      end if
    end function
  else if method = real Schur **then**
    $\hat{\boldsymbol{h}}_i$ *and* $\boldsymbol{z}_i$ *are real vectors or* 2×1 *block vectors with components* $\hat{\boldsymbol{f}}_i, \hat{\boldsymbol{g}}_i, \boldsymbol{x}_i, \boldsymbol{y}_i$
    $\boldsymbol{z}_n$ = apply Algorithm 3.1 to $\boldsymbol{h}_n$, where
    function SOLVE $(\boldsymbol{K}_{n,x} + \lambda_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{z}_i = \hat{\boldsymbol{h}}_i$:
      if 1×1 block **then**
        Solve $(\boldsymbol{K}_{n,x} + \alpha_{n,i}\boldsymbol{M}_{n,x})\boldsymbol{z}_i = \hat{\boldsymbol{h}}_i$
      else
        Obtain $\beta_{1,n,i}, \beta_{2,n,i}, \alpha_{n,i}$ from $\boldsymbol{T}_n$
        $\beta_1 = \beta_{1,n,i}, \beta_2 = \beta_{2,n,i}, \alpha = \alpha_{n,i}$
        Solve with preconditioner $\boldsymbol{P}_{i,n}^{rSchur}$:
        $$\begin{bmatrix} |\beta_2|(\boldsymbol{K}_{n,x} + \alpha\boldsymbol{M}_{n,x}) & -\beta_1|\beta_2|\boldsymbol{M}_{n,x} \\ |\beta_1|\beta_2\boldsymbol{M}_{n,x} & -|\beta_1|(\boldsymbol{K}_{n,x} + \alpha\boldsymbol{M}_{n,x}) \end{bmatrix} \begin{bmatrix} \boldsymbol{x}_i \\ \boldsymbol{y}_i \end{bmatrix} = \begin{bmatrix} |\beta_2|\hat{\boldsymbol{f}}_i \\ |\beta_1|\hat{\boldsymbol{g}}_i \end{bmatrix}$$
        $\boldsymbol{y}_i = -\boldsymbol{y}_i$
      end if
    end function
  end if
  $\boldsymbol{v}_n = (\boldsymbol{X}_n \otimes \boldsymbol{I}_n)\boldsymbol{z}_n$
end for
build up $\boldsymbol{v}$ from $\boldsymbol{v}_1, \dots, \boldsymbol{v}_N$

4. Numerical examples. In this section, we test the proposed preconditioners on the three (2+1)-dimensional space-time cylinder Q illustrated in Figure 3. The two-dimensional spatial domain Ω consists of 21 spatial subdomains (volumetric patches). For each time-slab, we use conforming B-Splines of degree p. The right-hand side f and the initial condition u_0 are given by

$$f(x,y,t) = \pi \sin(\pi x + 1)\sin(\pi y)(\cos(\pi t + 2) + 2\pi \sin(\pi t + 2)),$$
$$u_0(x,y) = \sin(2)\sin(\pi x + 1)\sin(\pi y).$$

At the outer boundary of the domain we impose Dirichlet boundary conditions, given by

$$g(x,y,t) = \sin(\pi x + 1)\sin(\pi y)\sin(\pi t + 2), \tag{4.1}$$

and at the inner boundaries (holes), we impose homogeneous Neumann conditions.

Considering the initial configuration with $p = (p_x, p_t) = (2,2)$ and without additional refinement, we have 272 dofs in the spatial direction (where 50 get removed due to the Dirichlet boundary conditions) and three dofs per slab in the time direction. In the following a uniform refinement in the x- or t-direction is abbreviated by "ref. x" and "ref. t," respectively.

The calculations presented in subsections 4.1, 4.2, and 4.3 were performed on a Desktop PC with an Intel Xeon CPU E5-1650 v2 @ 3.50GHz and 16 GB main memory. The computations in the subsequent subsection 4.4 were already done on the cluster Radon1 which we will introduce below in more detail.

We use the C++ library G+Smo for describing the geometry and performing the numerical tests; see also [19, 26].

In the following, we first investigate the condition number of the matrix $\boldsymbol{X}$ of eigenvectors of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$, which is an important quantity in the diagonalization approach. Second, we perform tests on how the smallest real eigenvalue of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ depends on θ, p, and h_t, because all of the proposed preconditioners require its strict positivity and a theoretical proof is still missing. Next, we present the robustness with respect to h and p of the proposed preconditioner $\boldsymbol{P}$ used in the Complex- and Real-Schur approaches. Finally, we combine the space-time MG introduced in [11] with our realization of the smoother.

4.1. Condition number of eigenvectors. Here, we study the condition number κ of the generalized eigenvectors of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$. Due to the nonsymmetry of $\boldsymbol{K}_t$ and $\boldsymbol{M}_t$, we do not obtain an orthogonal basis of eigenvectors. Hence, the condition number is not 1. Actually, it can be quite large. We report on the condition number for different p and N_t in Table 1. We observe that the condition number grows exponentially with p and N_t. Further numerical tests show the same behavior also for different values of $\theta \geq 0$. We conclude that for small p or small number of dofs in the time direction, the approach presented in section 3.3 may still be feasible.

4.2. Smallest real part of the eigenvalue of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$. In section 3.2, we observed the necessity that the smallest real part of the eigenvalues of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is positive. In this section, we present numerical studies for different p, h, and θ, where we fix the time interval to $[0,1]$. The results are summarized in Table 2, where the entries with $*$ indicate that the matrix $\boldsymbol{M}_t$ had at least one eigenvalue with negative real part. Consequently, the smallest real part of the generalized eigenvalues was also negative. We observe that if $\boldsymbol{M}_t > 0$, then also the real part of $\boldsymbol{M}_t^{-1}\boldsymbol{K}_t$ is positive. The positive real part of the eigenvalues for the $p = 1$ and $\theta = 0$ is in agreement

TABLE 1
Spectral condition number of $\boldsymbol{X}$ for the settings $\theta = 0.1$ and $|t_{n+1} - t_n| = 0.1$.

$N_t - p \setminus p$	2	3	4	5	6	7	8
2	64	309	362	766	1706	3907	9501
4	481	1036	3037	9419	41959	39323	73946
8	2869	16118	39693	74370	180054	472758	1e+06
16	34332	188263	463148	1e+06	6e+06	3e+07	1e+08
32	701306	2e+06	1e+07	6e+07	4e+08	7e+09	1e+10
64	5e+07	4e+07	3e+08	3e+09	6e+10	3e+11	1e+12
128	2e+08	1e+09	1e+10	3e+11	2e+13	5e+13	4e+14

TABLE 2
Smallest real part of generalized eigenvalues $\boldsymbol{K}_t x = \lambda \boldsymbol{M}_t x$ for different B-Spline degrees p, θ and number of dofs. The $*$ indicates that the matrix $\boldsymbol{M}_t$ has at least one eigenvalue with negative real part.

	2 refinements in t							4 refinements in t						
$\theta \backslash p$	1	2	3	4	5	6	7	1	2	3	4	5	6	7
0	1.5	2.4	3.2	3.8	4.3	4.7	5.0	0.2	0.5	0.9	1.5	2.1	2.7	3.4
0.01	1.6	2.5	3.2	3.6	4.0	4.4	4.9	0.7	0.7	1.1	1.6	2.2	2.8	3.3
0.1	2.5	2.9	3.2	3.6	4.0	4.5	5.2	4.8	2.9	2.7	3.0	3.4	3.6	4.1
1	4.1	4.5	4.7	*	*	*	*	12.4	12.0	9.2	*	*	*	*
10	4.6	5.2	5.2	*	*	*	*	6.7	11.8	*	*	*	*	*
	6 refinements in t							8 refinements in t						
$\theta \backslash p$	1	2	3	4	5	6	7	1	2	3	4	5	6	7
0	0.01	0.03	0.06	0.1	0.1	0.2	0.2	0.0008	0.002	0.004	0.006	0.009	0.01	0.02
0.01	1.9	1.0	0.8	0.7	0.6	0.6	0.6	7.7	4.0	3.0	2.5	2.0	1.8	1.6
0.1	18.6	9.9	7.4	6.0	5.1	4.5	4.0	34.8	33.8	29.5	23.8	20.0	17.2	15.1
1	34.2	35.1	33.8	*	*	*	*	34.8	34.4	34.5	*	*	*	*
10	11.4	17.4	*	*	*	*	*	29.0	32.2	*	*	*	*	*

with Remark 3.5. Moreover, for $\theta = 0$ and increasing p we observe even an increase of the smallest real part of the eigenvalues; cf. Proposition 3.3 and Remark 3.5. The numerical tests indicate that, for sufficiently small θ, the smallest real part of the generalized eigenvalues stays positive.

4.3. Condition number of preconditioned $\boldsymbol{K}_x + \lambda \boldsymbol{M}_x$. The aim of this section is to verify the optimal condition number bound presented in Theorems 3.8 and 3.9. To do so, we report on the maximum number of MINRES-iterations in order to solve $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$, where $\lambda_i \in \mathbb{C}$ are the generalized eigenvalues of $(\boldsymbol{K}_t, \boldsymbol{M}_t)$. We use zero initial guess, and a reduction of the initial residual by 10^{-10}. We choose $\theta = 0.1$. In Table 3, we investigate the robustness of the preconditioners from Theorems 3.8 and 3.9. We observe that the number of iterations stays bounded for various p and h.

4.4. Application to space-time MG. This section deals with the use of the iterative methods developed in section 3 as smoothers in the space-time MG method. The realization of the preconditioner P is performed via a sparse direct solver. We use the PARDISO solver from the Intel MKL for performing the LU factorizations. We compare the three different approaches, presented in subsections 3.3, 3.4, and 3.5, with the exact realization of $\boldsymbol{A}_n^{-1}$ via the sparse direct solver PARDISO. For approximating $\boldsymbol{A}_n^{-1}$ via MINRES, we use zero initial guess and a reduction of the initial residuum by 10^{-4}. In Tables 4 and 5 we report on the single-core computation time of the MG algorithm to set up the data-structures and solve the system via the MG iteration. The setup time includes the required LU factorizations, but not the

TABLE 3
Maximum number of MINRES-iterations to solve $\boldsymbol{K}_x + \lambda_i \boldsymbol{M}_x$, $i = 1, \dots, n_t$, resulting from the Complex- and Real-Schur decomposition. Refinement is performed uniformly in x and t.

	C-Schur decomp.					R-Schur decomp.				
ref. x and $t \setminus p$	2	3	4	5	6	2	3	4	5	6
0	23	22	26	26	26	21	24	25	26	25
1	25	24	24	27	26	26	26	28	27	27
2	25	25	25	27	27	28	28	28	27	28
3	24	26	26	27	27	28	28	28	28	28
4	25	25	26	27	26	28	28	26	26	26

assembling of the matrices, which is implemented as prescribed in section 2.4. The assembling does not give a significant contribution to the overall time. For the MG iteration, we use zero initial guess and a reduction of the initial residuum by 10^{-8}. We choose $\theta = 0.1$ and the polynomial degree $p = 3$ for both the space and the time directions. Moreover, we fix the number of dofs in the time direction of a time-slab, but increase the number of time-slabs, this gives 11 dofs in the time direction per time-slab. Hence in the first run with 19206 dofs, we have 873 spatial dofs. In Table 4, we use a fixed slablength $|t_n - t_{n+1}| = 0.1$, increasing the time interval with every refinement, whereas in Table 5 we refer to results obtained on the fixed time interval $(0, 5)$. In the latter case, this leads to a slablength of $5/N$, where N is the number of time-slabs. The time-parallel MG method uses coarsening in space as well as in time.

We observe that the LU factorization of $\boldsymbol{A}_n$ needs a quite large amount of time, whereas the setup time is almost negligible for the three preconditioners proposed. The little increase in the solution time definitely pays off by the small setup time. In addition, the Real-Schur decomposition almost provides the same solution time as the direct solver. Because of the complex arithmetic of the Diagonalization or the Complex-Schur decomposition, their computational effort doubles, which we also observe in the numerical test. Finally, due to the quite accurate approximation of $\boldsymbol{A}_n^{-1}$ (up to 10^{-4}), we do not observe a significant deterioration of the MG iteration numbers. It took around 12 MINRES-iterations to reach the desired tolerance of 10^{-4}. With almost the same settings we address the peak memory consumption of the different approaches in Table 6. The only difference in the settings compared to Table 4 is that we now do one additional refinement in space and also increase the number of time-slabs accordingly. We notice that the Diagonalization requires a bit more memory at its peak in terms of megabytes (MB) than the Real-Schur decomposition. Those methods require the least amount of memory compared to the Complex-Schur decomposition or the Direct approach which needs by far the highest amount of memory. In fact, the Direct method is the only method which exceeds the border of one gigabyte for 108108 and 706200 dofs, respectively. The memory requirement is measured in terms of the proportional set size (PSS) of a process. This is the amount of unshared memory unique to that process along with a proportion of its shared memory divided by the number of other processes sharing that memory. In conclusion, one obtains significant advantages in memory consumption and performance of the overall MG method by utilizing the tenor product structure of $\boldsymbol{A}_n$.

4.5. Parallelization in space and time. We conclude the numerical examples with studies regarding the parallelization in space and time. The results are obtained

TABLE 4

Comparison of the Eigenvalue, Complex-Schur, and Real-Schur decomposition with a sparse direct solver used for approximating $\boldsymbol{A}_n^{-1}$. The timings are given in seconds.

#dofs	ref.		#slabs	MG-It	Direct		Diag	
	x	t			Setup	Solving	Setup	Solving
19206	1	3	2	8	4.9	1.4	0.06	3.4
108108	2	3	4	8	52.7	11.4	0.4	25.4
706200	3	3	8	7	1213.7	105.0	4.4	198.3
#dofs	ref.		#slabs	MG-It	C-Schur		R-Schur	
	x	t			Setup	Solving	Setup	Solving
19206	1	3	2	8	0.08	3.3	0.06	1.9
108108	2	3	4	8	0.6	24.8	0.4	13.4
706200	3	3	8	7	6.2	198.3	4.4	104.9

TABLE 5

Comparison of the Eigenvalue, Complex-Schur, and Real-Schur decomposition with a sparse direct solver used for approximating $\boldsymbol{A}_n^{-1}$ in the time interval $(0, 5)$. The timings are given in seconds.

#dofs	ref.		#slabs	MG-It	Direct		Diag	
	x	t			Setup	Solving	Setup	Solving
19206	1	3	2	7	3.5	1.2	0.04	2.6
108108	2	3	4	7	53.0	10.0	0.4	22.2
706200	3	3	8	7	1211.7	104.0	4.4	204.0
#dofs	ref.		#slabs	MG-It	C-Schur		R-Schur	
	x	t			Setup	Solving	Setup	Solving
19206	1	3	2	7	0.04	2.8	0.04	1.6
108108	2	3	4	7	0.4	22.3	0.4	12.0
706200	3	3	8	7	4.6	204.1	4.4	108.1

on the RADON1[1] cluster at Linz. Each node is equipped with 2x Xeon E5-2630v3 "Haswell" CPU (8 Cores, 2.4Ghz, 20MB Cache) and 128 GB RAM. We consider a similar setup as in section 4.4, i.e., we use B-Splines of degree 3, perform two refinements in the spatial direction, three in the time direction, θ is given by 0.1, and a fixed time interval $(0, 5)$, i.e., a slablength of $5/N$, where N is the number of slabs. We perform coarsening in space and time. The systems with the system matrix $(\boldsymbol{K}_x + \gamma \boldsymbol{M}_x)$, $\gamma \in \mathbb{R}^+$ are solved by means of PARDISO, i.e., a sparse direct solver. For the time-parallel MG method, we use a reduction of the initial residuum by 10^{-8} as stopping criterion, while a relative tolerance of 10^{-4} is used for the linear system in (3.1). First, we consider only parallelization in time and we use the same number of processors and time-slabs, i.e., each time-slab is associated with one processor. In Table 7, we report on the weak scaling results using the Real-Schur decomposition to realize the application of $\boldsymbol{A}_n^{-1}$. We observe an excellent weak scaling with just a minor increase in the computation time. The last two columns present serial results, where (2.12) is solved by a time-stepping like approach.

As the last example, we present results, where we perform parallelization in space and time simultaneously. We consider the same setting as above, but we perform an additional uniform splitting of each patch into four new patches, giving in total 84 patches for the spatial domain and only two refinements in the time direction. In order to enable parallelization in the spatial direction, we utilize the parallel p-robust multipatch MG method with minor adaptions due to the presence of the mass matrix; see [16, 35]. We solve the spatial problems corresponding to real eigenvalues by

[1] https://www.ricam.oeaw.ac.at/hpc/

TABLE 6

Comparison of the Eigenvalue, Complex-Schur, and Real-Schur decomposition with a sparse direct solver used for approximating $\boldsymbol{A}_n^{-1}$. The peak memory requirements are given in MB.

#dofs	ref. x	ref. t	#slabs	Direct Peak requirement	Diag Peak requirement
108108	3	3	4	2231.3	108.5
706200	4	3	8	21791.6	753.3
5061936	5	3	16	Out of memory	7000.2

#dofs	ref. x	ref. t	#slabs	C-Schur Peak requirement	R-Schur Peak requirement
108108	3	3	4	130.6	103.0
706200	4	3	8	937.8	712.6
5061936	5	3	16	8907.4	6686.9

TABLE 7

Weak scaling results for parallelization in time for the time interval $(0, 5)$; timings are given in seconds. Real-Schur decomposition is used to realize the application of $\boldsymbol{A}_n^{-1}$.

#dofs	#procs	#slabs	MG-It.	Setup	Solving	Serial setup/solving	
54054	2	2	7	0.11	3.2	0.2	0.4
108108	4	4	7	0.12	3.9	0.4	0.8
216216	8	8	7	0.11	4.0	0.8	1.63
432432	16	16	7	0.11	4.7	1.6	3.3
864864	32	32	7	0.11	4.8	3.3	6.5
1729728	64	64	8	0.12	5.8	6.6	13.13
3459456	128	128	9	0.12	7.4	13.6	26.0
6918912	256	256	9	0.2	10.7	36.5	50.2
13837824	512	512	10	0.3	17.2	82.4	98.1

means of a preconditioned CG method. The problems corresponding to the complex eigenvalues are solved by a MINRES-method, where we use five MG cycles for the construction of the preconditioner $\hat{\boldsymbol{P}}^{-1}$ and two pre- and postsmoothing steps in each MG cycle. In Table 8, we report on the weak scaling results using the Real-Schur decomposition to realize the application of $\boldsymbol{A}_n^{-1}$. The numbers c_x and c_t denote the used number of cores in (the) space and (the) time direction, respectively, and c_{total} refers to the total number of used cores. We again observe a good weak scaling, where the computation time even decreases a bit, which is due to caching effects in the parallel MG method. Nevertheless, the total computation time is much higher than in Table 7 due to the fact that the direct solvers perform very well on a relatively small spatial problem in two dimensions. Moreover, we perform additional refinements in the spatial direction and have a larger number of patches. The last two columns present results for the time-stepping like approach, having only parallelization in space according to c_x.

5. Conclusions. In this work, we presented a decomposition of a nonsymmetric linear system arising from a space-time formulation into a series of symmetric linear systems, which are easier to solve. These spatial problems are part of the time-parallel MG method introduced in [28]. They are either symmetric and positive definite or have a symmetric saddle point structure. For the latter, we presented robust preconditioners. The advantage of the proposed decompositions consists of the availability of well-established preconditioners for SPD problems. The approach extends to a spatial coupling of the interface dofs via a dG-IgA scheme; see, e.g, [24].

TABLE 8

Weak scaling results for parallelization in space and time in the interval $(0, 5)$*; timings are given in seconds. Real-Schur decomposition is used to realize the application of* $\boldsymbol{A}_n^{-1}$.

#dofs	ref. x	#sl.	c_{total}	c_x	c_t	it	Setup	Solving	Serial setup/solving $c_t = 1$	
42028	1	4	2	1	2	8	0.46	179.6	0.74	23.2
84056	1	8	4	1	4	9	0.46	212.6	1.36	47.0
193368	2	8	16	4	4	9	0.50	140.1	1.25	27.1
386736	2	16	32	4	8	9	0.50	144.4	2.27	54.2
1092784	3	16	128	16	8	9	0.56	133.4	1.82	49.2
2185568	3	32	256	16	16	10	0.56	153.6	3.3	99.0
4371136	3	64	512	16	32	10	0.63	152.7	6.2	196.7

Acknowledgment. The authors would like to express their thanks to the anonymous referees for their helpful hints and valuable suggestions.

REFERENCES

[1] R. A. ADAMS AND J. J. F. FOURNIER, *Sobolev Spaces*, 2nd ed., Pure Appl. Math. (Amst.) 140, Elsevier/Academic Press, Amsterdam, 2003.

[2] R. ANDREEV AND C. TOBLER, *Multilevel preconditioning and low rank tensor iteration for space-time simultaneous discretizations of parabolic PDEs*, Numer. Linear Algebra Appl., 22 (2015), p. 317–337, https://doi.org/10.1002/nla.1951.

[3] L. BEIRÃO DA VEIGA, A. BUFFA, G. SANGALLI, AND R. VÁZQUEZ, *Mathematical analysis of variational isogeometric methods*, Acta Numer., 23 (2014), pp. 157–287, https://doi.org/10.1017/S096249291400004X.

[4] L. BEIRÃO DA VEIGA, L. F. PAVARINO, S. SCACCHI, O. B. WIDLUND, AND S. ZAMPINI, *Adaptive selection of primal constraints for isogeometric BDDC deluxe preconditioners*, SIAM J. Sci. Comput., 39 (2017), pp. A281–A302, https://doi.org/10.1137/15M1054675.

[5] J. BERGH AND J. LÖFSTRÖM, *Interpolation Spaces. An Introduction*, Grundlehren der Mathematischen Wissenschaften 223, Springer-Verlag, Berlin, New York, 1976.

[6] A. N. BROOKS AND T. J. R. HUGHES, *Streamline upwind/Petrov-Galerkin formulation for convection dominated flows with particular emphasis on the incompressible Navier-Stokes equations*, Comput. Methods Appl. Mech. Engrg., 32 (1982), pp. 199–259.

[7] J. A. COTTRELL, T. J. R. HUGHES, AND Y. BAZILEVS, *Isogeometric Analysis, Toward Integration of CAD and FEA*, John Wiley & Sons, Chichester, West Sussex, UK, 2009.

[8] M. DONATELLI, C. GARONI, C. MANNI, S. SERRA-CAPIZZANO, AND H. SPELEERS, *Symbol-based multigrid methods for Galerkin B-spline isogeometric analysis*, SIAM J. Numer. Anal., 55 (2017), pp. 31–62, https://doi.org/10.1137/140988590.

[9] R. FALGOUT, S. FRIEDHOFF, T. KOLEV, S. MACLACHLAN, AND J. SCHRODER, *Parallel time integration with multigrid*, SIAM J. Sci. Comput., 36 (2014), pp. C635–C661, https://doi.org/10.1137/130944230.

[10] M. J. GANDER, 50 *years of time parallel time integration*, in Multiple Shooting and Time Domain Decomposition, Contrib. Math. Comput. Sci., 9, T. Carraro, M. Geiger, S. Körkel, and R. Rannacher, eds., Springer, Cham, 2015, pp. 69–114; available online at www.unige.ch/~gander/Preprints/50YearsTimeParallel.pdf.

[11] M. GANDER AND M. NEUMÜLLER, *Analysis of a new space-time parallel multigrid algorithm for parabolic problems*, SIAM J. Sci. Comput., 38 (2016), pp. A2173–A2208, https://doi.org/10.1137/15M1046605.

[12] M. GANDER AND S. VANDEWALLE, *Analysis of the parareal time-parallel time-integration method*, SIAM J. Sci. Comput., 29 (2007), pp. 556–578, https://doi.org/10.1137/05064607X.

[13] W. HACKBUSCH, *Parabolic multigrid methods*, in Computing Methods in Applied Sciences and Engineering VI, R. Glowinski and J.-L. Lions, eds., North-Holland, Amsterdam, 1984, pp. 189–197.

[14] C. HOFER, *Parallelization of continuous and discontinuous Galerkin dual-primal isogeometric tearing and interconnecting methods*, Comput. Math. Appl., 74 (2017), pp. 1607–1625, https://doi.org/10.1016/j.camwa.2017.06.051.

[15] C. Hofer, U. Langer, M. Neumüller, and I. Toulopoulos, *Time-multipatch discontinuous Galerkin space-time isogeometric analysis of parabolic evolution problems*, Electron. Trans. Numer. Anal., 49 (2018), pp. 126–150.

[16] C. Hofer and S. Takacs, *A parallel multigrid solver for multi-patch isogeometric analysis*, in Advanced Finite Element Methods with Applications. Selected Papers from the 30th Chemnitz Finite Element Symposium 2017, T. Apel, U. Langer, A. Meyer, and O. Steinbach, eds., Lect. Notes Comput. Sci. Eng. 128, Springer, Cham, 2019, pp. 191–206.

[17] C. Hofreither and S. Takacs, *Robust multigrid for isogeometric analysis based on stable splittings of spline spaces*, SIAM J. Numer. Anal., 4 (2017), pp. 2004–2024, https://doi.org/10.1137/16M1085425.

[18] T. J. R. Hughes, J. A. Cottrell, and Y. Bazilevs, *Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement*, Comput. Methods Appl. Mech. Engrg., 194 (2005), pp. 4135–4195.

[19] B. Jüttler, U. Langer, A. Mantzaflaris, S. E. Moore, and W. Zulehner, *Geometry + simulation modules: Implementing isogeometric analysis*, PAMM, 14 (2014), pp. 961–962, https://doi.org/10.1002/pamm.201410461.

[20] O. A. Ladyzhenskaya, *The Boundary Value Problems of Mathematical Physics*, Nauka, Moscow, 1973 (in Russian); Appl. Math. Sci. 49, Springer-Verlag, New York, 1985 (in English).

[21] O. A. Ladyzhenskaya, V. A. Solonnikov, and N. N. Uraltseva, *Linear and Quasilinear Equations of Parabolic Type*, Nauka, Moscow, 1967 (in Russian); Transl. Math. Mongr. 23, AMS, Providence, RI, 1968 (in English).

[22] J. Lang, *Adaptive Multilevel Solution of Nonlinear Parabolic PDE Systems. Theory, Algorithm, and Applications*, Lect. Notes Comput. Sci. Eng. 16, Springer-Verlag, Berlin, Heidelberg, 2001.

[23] U. Langer, S. Moore, and M. Neumüller, *Space-time isogeometric analysis of parabolic evolution equations*, Comput. Methods Appl. Mech. Engrg., 306 (2016), pp. 342–363, https://doi.org/10.1016/j.cma.2016.03.042.

[24] U. Langer and I. Toulopoulos, *Analysis of multipatch discontinuous Galerkin IgA approximations to elliptic boundary value problems*, Comput. Vis. Sci., 17 (2015), pp. 217–233, https://doi.org/10.1007/s00791-016-0262-6.

[25] J.-L. Lions, Y. Maday, and G. Turinici, *A parareal in time discretization of PDEs*, C.R. Acad. Sci. Paris, Sér. I Math., 332 (2001), pp. 661–668.

[26] A. Mantzaflaris, C. Hofer, et al., *G+Smo (Geometry + Simulation Modules)* v0.8.1, http://gs.jku.at/gismo, 2015.

[27] E. McDonald, J. Pestana, and A. Wathen, *Preconditioning and iterative solution of all-at-once systems for evolutionary partial differential equations*, SIAM J. Sci. Comput., 40 (2018), pp. A1012–A1033, https://doi.org/10.1137/16M1062016.

[28] M. Neumüller, *Space-Time Methods: Fast Solvers and Applications*, Monographic Series TU Graz: Computation in Engineering and Science 20, TU Graz, 2013, https://doi.org/10.3217/978-3-85125-290-3.

[29] M. Neumüller and I. Smears, *Time-parallel iterative solvers for parabolic evolution equations*, SIAM J. Sci. Comput., 41 (2019), p. C28–C51, https://doi.org/10.1137/18M1172466.

[30] G. Sangalli and M. Tani, *Isogeometric preconditioners based on fast solvers for the Sylvester equation*, SIAM J. Sci. Comput., 38 (2016), pp. A3644–A3671, https://doi.org/10.1137/16M1062788.

[31] O. Steinbach, *Space-time finite element methods for parabolic problems*, Comput. Methods Appl. Math., 15 (2015), pp. 551–566, https://doi.org/10.1515/cmam-2015-0026.

[32] O. Steinbach and H. Yang, *Comparison of algebraic multigrid methods for an adaptive space-time finite-element discretization of the heat equation in 3D and 4D*, Numer. Linear Algebra Appl., 25 (2018), e2143, https://doi.org/10.1002/nla.2143.

[33] O. Steinbach and H. Yang, *Space–time finite element methods for parabolic evolution equations: Discretization, a posteriori error estimation, adaptivity and solution*, in Space-Time Methods: Applications to Partial Differential Equations, Radon Series on Computational and Applied Mathematics 25, U. Langer and O. Steinbach, eds., Walter de Gruyter, Berlin, 2019.

[34] M. Stynes, *Steady–state convection–diffusion problems*, Acta Numer., 14 (2005), pp. 445–508.

[35] S. Takacs, *Robust approximation error estimates and multigrid solvers for isogeometric multi-patch discretizations*, Math. Models Methods Appl. Sci., 28 (2018), pp. 1899–1928, https://doi.org/10.1142/S021820251850046X.

[36] M. Tani, *A preconditioning strategy for linear systems arising from nonsymmetric schemes in isogeometric analysis*, Comput. Math. Appl., 74 (2017), pp. 1690–1702, https://doi.org/10.1016/j.camwa.2017.06.013.

[37] V. Thomeé, *Galerkin Finite Element Methods for Parabolic Problems*, 2nd ed., Springer Ser. Comput. Math. 25, Springer-Verlag, Berlin, 2006.

[38] S. Vandewalle, *Parallel Multigrid Wavefrom Relaxation for Parabolic Problems*, Teubner Skripten zur Numerik, Teubner, 1993.

[39] T. Warburton and J. Hesthaven, *On the constants in hp-finite element trace inverse inequalities*, Comput. Methods Appl. Mech. Engrg., 192 (2003), pp. 2765–2773, https://doi.org/10.1016/S0045-7825(03)00294-9.

[40] M. Wolfmayr, *Multiharmonic Finite Element Analysis of Parabolic Time-Periodic Simulation and Optimal Control Problems*, Ph.D. thesis, Johannes Kepler University, Institute of Computational Mathematics, Linz, Austria, 2014, http://www.numa.uni-linz.ac.at/Teaching/PhD/Finished/wolfmayr.

[41] W. Zulehner, *Nonstandard norms and robust estimates for saddle point problems*, SIAM J. Matrix Anal. Appl., 32 (2011), pp. 536–560, https://doi.org/10.1137/100814767.