

## A computational framework for pharmaco-mechanical interactions in arterial walls using parallel monolithic domain decomposition methods

Balzani, Daniel; Heinlein, Alexander; Klawonn, Axel; Knepper, Jascha; Nurani Ramesh, Sharan; Rheinbach, Oliver; Saßmannshausen, Lea; Uhlmann, Klemens

**DOI**

[10.1002/gamm.202370002](https://doi.org/10.1002/gamm.202370002)

**Publication date**

2024

**Document Version**

Final published version

**Published in**

GAMM Mitteilungen

**Citation (APA)**

Balzani, D., Heinlein, A., Klawonn, A., Knepper, J., Nurani Ramesh, S., Rheinbach, O., Saßmannshausen, L., & Uhlmann, K. (2024). A computational framework for pharmaco-mechanical interactions in arterial walls using parallel monolithic domain decomposition methods. *GAMM Mitteilungen*, 47(1), Article e202370002. <https://doi.org/10.1002/gamm.202370002>

**Important note**

To cite this publication, please use the final published version (if applicable).  
Please check the document version above.









**Copyright**

Other than for strictly personal use, it is not permitted to download, forward or distribute the text or part of it, without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license such as Creative Commons.

**Takedown policy**

Please contact us and provide details if you believe this document breaches copyrights.  
We will remove access to the work immediately and investigate your claim.

# A computational framework for pharmaco-mechanical interactions in arterial walls using parallel monolithic domain decomposition methods

Daniel Balzani<sup>1</sup>  | Alexander Heinlein<sup>2</sup>  | Axel Klawonn<sup>3,4</sup>  | Jascha Knepper<sup>3,4</sup>  |  
 Sharan Nurani Ramesh<sup>1</sup>  | Oliver Rheinbach<sup>5,6</sup>  | Lea Saßmannshausen<sup>3</sup>  |  
 Klemens Uhlmann<sup>1</sup> 

<sup>1</sup>Civil and Environmental Engineering,  
Ruhr University Bochum, Bochum,  
Germany

<sup>2</sup>Delft Institute of Applied Mathematics,  
Delft University of Technology, Delft,  
The Netherlands

<sup>3</sup>Department of Mathematics and  
Computer Science, University of Cologne,  
Cologne, Germany

<sup>4</sup>Center for Data and Simulation Science,  
University of Cologne, Cologne, Germany

<sup>5</sup>Fakultät für Mathematik und Informatik,  
Technische Universität Bergakademie  
Freiberg, Freiberg, Germany

<sup>6</sup>Zentrum für effiziente  
Hochtemperaturstoffwandlung (ZeHS),  
Technische Universität Bergakademie  
Freiberg, Freiberg, Germany

## Correspondence

Axel Klawonn, University of Cologne,  
Department of Mathematics and  
Computer Science, Weyertal 86-90,  
50931 Köln, Germany.  
Email: [axel.klawonn@uni-koeln.de](mailto:axel.klawonn@uni-koeln.de)

## Funding information

Deutsche Forschungsgemeinschaft,  
Grant/Award Number: 465228106;  
NHR@FAU, Grant/Award Number:  
k105be

## Abstract

A computational framework is presented to numerically simulate the effects of antihypertensive drugs, in particular calcium channel blockers, on the mechanical response of arterial walls. A stretch-dependent smooth muscle model by Uhlmann and Balzani is modified to describe the interaction of pharmacological drugs and the inhibition of smooth muscle activation. The coupled deformation-diffusion problem is then solved using the finite element software FEDDLib and overlapping Schwarz preconditioners from the Trilinos package FROSch. These preconditioners include highly scalable parallel GDSW (generalized Dryja–Smith–Widlund) and RGDSW (reduced GDSW) preconditioners. Simulation results show the expected increase in the lumen diameter of an idealized artery due to the drug-induced reduction of smooth muscle contraction, as well as a decrease in the rate of arterial contraction in the presence of calcium channel blockers. Strong and weak parallel scalability of the resulting computational implementation are also analyzed.

## KEYWORDS

calcium channel blockers, domain decomposition methods, drug transport, finite element method, GDSW coarse space, hypertension, iterative solvers, overlapping Schwarz, RGDSW coarse space, scalable preconditioners, smooth muscle cells, structural mechanics

## 1 | INTRODUCTION

Approximately one in four adults worldwide has hypertension [17]. In contemporary societies, the mean blood pressure levels increase steadily with age, in stark contrast to preindustrial societies, where the levels changed little with age [47]. Furthermore, there has been a rapid increase in the prevalence of hypertension in the last few decades [42]. Hypertension

This is an open access article under the terms of the [Creative Commons Attribution-NonCommercial-NoDerivs](https://creativecommons.org/licenses/by-nc-nd/4.0/) License, which permits use and distribution in any medium, provided the original work is properly cited, the use is non-commercial and no modifications or adaptations are made.

© 2024 The Authors. *GAMM - Mitteilungen* published by Wiley-VCH GmbH.

is a major risk factor for cardiovascular disease and is associated with atherosclerosis, stroke, heart failure, and kidney disease. Atherosclerosis is a condition characterized by a reduction in the arterial lumen diameter and obstructed blood flow, due to the formation of fibrofatty lesions called atherosclerotic plaques. These plaques contain oxidized lipids, calcium deposits, inflammatory cells, and smooth muscle cells.

Hypertension and atherosclerosis are the two major causes of cardiovascular disease, which is the most common form of mortality in the world. Both these conditions are treated by antihypertensive drugs such as ACE inhibitors, angiotensin II receptor blockers, dihydropyridine calcium channel blockers (CCBs) and thiazide diuretics. Some of these drugs directly affect the arterial wall and can lead to changes in wall structure and function. For example, CCBs work by reducing the contractility of the arterial wall, thereby reducing blood pressure. However, it has been observed that when multiple drugs are used to treat hypertension, the risk of ischemic events increases with each drug class. In cases where three or more classes of drugs were necessary to successfully control hypertension, the risk of stroke was observed to be 2.5 times higher than in healthy normotensive people [33]. Additionally, the exact mechanism of the drug-induced mechanical response of the arterial wall in atherosclerotic arteries is not well understood. In light of the increasing prevalence of hypertension and the resulting uptake of antihypertensive drugs, there is a pressing need for a better understanding of pharmaco-mechanical interactions in arteries. To this end, numerical simulation of healthy and atherosclerotic arteries can be a valuable tool. Simulating the mechanical response entails an accurate description of the arterial wall material behavior, the drug transport process and the drug-artery interaction.

Arteries consist of three major components: endothelial cells, smooth muscle cells (SMCs) and the extracellular matrix (ECM). The thickness of the endothelium is negligible in comparison to the thickness of the artery and so is its load-carrying capacity; therefore it is not considered in our model. The ECM is composed of elastin and collagen fibers and supports the mechanical load. Vascular SMCs are arranged concentrically and their coordinated contraction and relaxation causes changes in luminal diameter. Contraction in SMCs is regulated by cytosolic free calcium concentration. A change in membrane potential due to factors like cell stretch, neurotransmitters, and hormones causes the opening of voltage-gated calcium channels, thereby enabling the movement of calcium ions into the cytosol [60]. The calcium forms a complex with calmodulin and activates the enzyme myosin light chain kinase (MLCK), which phosphorylates the regulatory light chain (RLC) of myosin, enabling contraction. The enzyme myosin light chain phosphatase (MLCP) is responsible for the dephosphorylation of myosin RLC. Enzymes such as Rho kinase inhibit the activity of MLCP and lead to a calcium-independent contraction mechanism [65]. Since antihypertensive drugs primarily work by interacting with SMC activation, it is important to have an SMC model that allows for a meaningful description of drug-tissue interaction. There are several models that describe the active response of arteries [9, 43, 44, 54, 63, 64, 68, 69]. The well-accepted cross-bridge phosphorylation model by Hai and Murphy [20] describes the influence of MLCK and MLCP on contraction. Yang et al. [68, 69] proposed an electro-chemo-mechanical model where the change in membrane potential due to the various ion channels is considered. In Böhl et al. [9], the calcium concentration was defined as a function of time, and arteries with calcium waves were simulated with a one-sided coupling. Uhlmann and Balzani [63] extended the model proposed by Murtada et al. [43] to include the influence of stretch on the action of MLCP and MLCK. Here, we adopt the active response model by Uhlmann and Balzani [63] and modify it to model the impact of dihydropyridine CCBs. In a previous work [45], we already investigated the effect of CCBs on the activity of MLCK, considering constant MLCP activity. CCBs work by blocking the voltage-gated calcium channels and thereby reducing the calcium influx into the cytosol. The arterial wall mass transport process, depending on the type of drug, is either advection or diffusion dominated and can therefore be modeled by a linear scalar advection-diffusion equation [1, 32, 34]. However, due to drug absorption and binding, and to account for complex patient-specific arteries, the three-dimensional reaction-advection-diffusion equation is necessary to comprehensively describe the drug transport phenomenon in arteries. In the case of hydrophobic drugs, diffusion-dominated transport is observed, whereas, in hydrophilic drugs, advection-dominated transport is observed [11, 35, 46]. Since we consider only hydrophobic CCBs, we may simplify the transport process and model it using the reaction-diffusion equation.

To simulate patient-specific arteries, a sufficient spatial resolution is required, leading to large number of degrees of freedom and ill-conditioned matrices. Moreover, for the interaction of the fluid with the arterial wall, strong coupling schemes are most suitable, and monolithic fluid–structure interaction (FSI) coupling schemes are most competitive in this regime; this behavior can be mathematically explained by the so-called *added-mass effect* [10]. This effect also explains instabilities typically resulting from the use of weakly coupled schemes, which are very efficient in other applications like aeroelasticity.

In this work, the effect of the antihypertensive drug in the structure is modeled and simulated; that is, we are concerned with coupled structure–chemistry interaction problem (SCI). A fully monolithic approach is taken; that is, the

multiphysics problem is assembled into a single system. To solve this problem, we then require robust parallel scalable preconditioners.

The remainder of the paper is organized as follows. In Section 2, the model for the arterial wall is introduced and in Section 3, the pharmaco-mechanical interaction is modeled. Next, in Section 4, the monolithic, two-level overlapping Schwarz preconditioners for the coupled system are presented. In Section 5, the software ecosystem defining our computational framework is described. Then, numerical results obtained using this software framework are presented. In Section 7, a conclusion is given.

## 2 | MODELING THE ARTERIAL WALL

In the present work, we restrict ourselves to resistance arteries, which are characterized by a high SMC content. Since we are interested in the vasoregulatory action of arteries, which is primarily a stretch-dependent, chemo-mechanically coupled problem, we neglect SMC activation due to other factors like the sympathetic nervous system and the endocrine system. We further restrict our model to the tunica media layer of the artery, which consists of a passive ECM and active smooth muscle cells. The passive behavior of the ECM is modeled using a hyperelastic material law described in Balzani et al. [4]. Two distinct crosswise helically arranged fiber directions are considered, both lying in the longitudinal-circumferential plane and symmetric about the circumferential axis. The active material is modeled using the phenomenological model by Uhlmann and Balzani [63], where the models of Hai and Murphy [20] and Muratada et al. [43] are extended to include the effects of stretch on the calcium-dependent and -independent contraction mechanisms.

### 2.1 | General continuum-mechanical model

Let  $\mathbf{X}$  and  $\mathbf{x}$  denote a material point in the initial configuration  $\mathcal{B}$  and the deformation in the current configuration  $\mathcal{S}$ , respectively. The motion of the point over time is defined by the map  $\mathbf{x} = \varphi(\mathbf{X}, t)$ . The associated deformation gradient  $\mathbf{F}$  and the right Cauchy–Green tensor  $\mathbf{C}$  are defined as

$$\mathbf{F} = \frac{\partial \mathbf{x}}{\partial \mathbf{X}}, \quad \mathbf{C} = \mathbf{F}^T \mathbf{F}. \quad (1)$$

We adopt an invariant-based framework for the constitutive model with two discrete fiber directions described by the referential unit vectors  $\mathbf{a}^{(f)}$ , where the strain energy density  $\Psi$  is additively decomposed as

$$\Psi = \Psi_{\text{p, isot}} + \sum_{f=1}^2 \Psi_{\text{p, ti}}^{(f)} + \sum_{f=1}^2 \Psi_{\text{a}}^{(f)}. \quad (2)$$

This additive decomposition is applicable, since only a weak interaction between the fiber families can be assumed. Here, the influence of the isotropic elastin matrix is described by  $\Psi_{\text{p, isot}}$ ; the material behavior of the passive collagen fibers is modeled as a transversely isotropic part in  $\Psi_{\text{p, ti}}^{(f)}$ , and the active, transversely isotropic material response is represented by  $\Psi_{\text{a}}^{(f)}$ . The principal and mixed invariants of the right Cauchy–Green tensor  $\mathbf{C}$  are defined as

$$I_1 = \text{tr}(\mathbf{C}), \quad I_2 = \frac{1}{2}(\text{tr}(\mathbf{C})^2 - \text{tr}(\mathbf{C}^2)), \quad I_3 = \det(\mathbf{C}), \quad I_4^{(f)} = \mathbf{C} : \mathbf{M}^{(f)}, \quad I_5^{(f)} = \mathbf{C}^2 : \mathbf{M}^{(f)}, \quad (3)$$

where  $\mathbf{M}^{(f)} = \mathbf{a}^{(f)} \otimes \mathbf{a}^{(f)}$  are the structural tensors. Using the formulation by Balzani et al. [4] we have the passive components

$$\begin{aligned} \Psi_{\text{p, isot}} &= \alpha_1 \left( I_1 I_3^{-1/3} - 3 \right) + \alpha_2 \left( I_3^{\alpha_3} + I_3^{-\alpha_3} - 2 \right), \\ \Psi_{\text{p, ti}}^{(f)} &= \alpha_4 \langle K_3^{(f)} - 2 \rangle^{\alpha_5}, \end{aligned} \quad (4)$$

where  $\alpha_1, \alpha_2, \alpha_3, \alpha_4$  and  $\alpha_5$  are material parameters and  $K_3^{(f)} = I_1 I_4^{(f)} - I_5^{(f)}$ . Furthermore,  $\langle \bullet \rangle = 1/2(\bullet + |\bullet|)$  defines the Macaulay brackets. Note that the material of the arterial wall is considered to be nearly incompressible and, therefore, the second term of  $\Psi_{\text{p, isot}}$  in (4) punishes deviations from the incompressible state.

## 2.2 | Stretch-dependent chemical kinetics model for smooth muscle cells

The SMC cytoskeleton is composed of a three-dimensional network of actin and myosin filaments connected at cytoplasmic dense bodies. SMC contraction is a result of relative sliding between the myosin and actin filaments. The myosin filaments contain distinct head regions that interact with actin to form attached cross-bridges. Contraction primarily occurs due to increased intracellular free calcium concentration ( $[Ca^{2+}]_i$ ). Calmodulin, a calcium-binding protein, binds with calcium ions to form a complex that activates the enzyme MLCK, which results in the phosphorylation of the 20-kDa light chain of myosin. This stimulates myosin-ATPase activity, enabling cross-bridge cycling and force generation, resulting in SMC contraction. The myosin regulatory light chain is dephosphorylated by the activity of the enzyme MLCP. Hai and Murphy [20] proposed a model where cross-bridges exist in four distinct states: free dephosphorylated (A), free phosphorylated (B), attached phosphorylated (C) and attached dephosphorylated (D). Their change is described by seven rate constants:  $k_1$  and  $k_6$  represent the rate of phosphorylation by MLCK,  $k_2$  and  $k_5$  represent the rate of dephosphorylation by MLCP,  $k_3$  is the rate of cross-bridge attachment,  $k_4$  and  $k_7$  are the rates of cross-bridge detachment. The kinetic model is thus governed by the following system of ordinary differential equations

$$\begin{bmatrix} \dot{n}_A \\ \dot{n}_B \\ \dot{n}_C \\ \dot{n}_D \end{bmatrix} = \begin{bmatrix} -k_1 & k_2 & 0 & k_7 \\ k_1 & -k_2 - k_3 & k_4 & 0 \\ 0 & k_3 & -k_4 - k_5 & k_6 \\ 0 & 0 & k_5 & -k_6 - k_7 \end{bmatrix} \begin{bmatrix} n_A \\ n_B \\ n_C \\ n_D \end{bmatrix}, \quad (5)$$

where  $n_A$ ,  $n_B$ ,  $n_C$  and  $n_D$  are the fractions of cross-bridges in the four aforementioned states, which fulfill the constraint  $n_A + n_B + n_C + n_D = 1$ . The rate of cross-bridge attachment ( $k_3$ ) and detachment ( $k_4$  and  $k_7$ ) are assumed to be constant, whereas the rates of phosphorylation  $k_1$  and  $k_6$  are given, as suggested first by Murtada et al. [43], by

$$k_{1/6} = \eta \frac{[Ca^{2+}]_i^2}{[Ca^{2+}]_i^2 + (Ca_{50})^2}, \quad (6)$$

where  $\eta$  is a material parameter and  $Ca_{50}$  is the half activation constant. Further, as proposed by Uhlmann and Balzani [63], the cytosolic calcium concentration is modeled as a stretch-dependent function

$$[Ca^{2+}]_i = \gamma_1 (\lambda - \bar{\lambda}_c)^2, \quad (7)$$

with  $\lambda = (I_4)^{1/2}$  being the stretch along the SMC longitudinal direction, and  $\gamma_1$  a material parameter. The authors modeled the reduction in intracellular calcium concentration due to various factors like calcium sequestration and calcium outflow by an evolution equation for  $\bar{\lambda}_c$

$$\dot{\bar{\lambda}}_c = \dot{\bar{\lambda}}_{c,\min} + \frac{\dot{\bar{\lambda}}_{c,\max} - \dot{\bar{\lambda}}_{c,\min}}{1 + e^{\gamma_2([Ca^{2+}]_{\text{tar}} - [Ca^{2+}]_i - \tau_c)}}, \quad (8)$$

where  $\dot{\bar{\lambda}}_{c,\min}$ ,  $\dot{\bar{\lambda}}_{c,\max}$ ,  $\gamma_2$ ,  $\tau_c$  are parameters and  $[Ca^{2+}]_{\text{tar}}$ , the target calcium concentration, is defined by a stretch-dependent function

$$[Ca^{2+}]_{\text{tar}} = \gamma_3 \frac{\lambda^2}{\lambda^2 + \lambda_{50,c}^2}, \quad (9)$$

with  $\gamma_3$  being a material parameter and  $\lambda_{50,c}$  the half activation stretch. Calcium sensitization, which is a calcium-independent contraction mechanism [53], occurs as a result of the action of the enzymes RhoA and Rho-Kinase (ROK). Uhlmann and Balzani [63] modeled the resulting reduction in the rate of dephosphorylation by evolution equations for  $k_{2/5}$

$$\begin{aligned}\dot{k}_{2/5} &= \dot{k}_{2/5,\min} \left( 1 - e^{-\zeta_1(k_{2/5} - k_{2/5,\min})} \right) + \frac{\dot{k}_{2/5,\max} - \dot{k}_{2/5,\min}}{1 + e^{\gamma_4(\lambda - \bar{\lambda}_p - \tau_p)}}, \\ \dot{\lambda}_p &= \dot{\lambda}_{p,\min} + \frac{\dot{\lambda}_{p,\max} - \dot{\lambda}_{p,\min}}{1 + e^{\gamma_5(k_{2/5,\text{tar}} - k_{2/5} - \tau_k)}} - \dot{\lambda}_{p,\max} e^{-\zeta_2(\lambda - \bar{\lambda}_p - \Delta \bar{\lambda}_{p,\min})},\end{aligned}\quad (10)$$

where  $\dot{k}_{2/5,\min}$  and  $\dot{k}_{2/5,\max}$  are the minimum and maximum rate of change of  $k_{2/5}$ , respectively, and  $\gamma_4$  and  $\tau_p$  are parameters. Here, we additionally use  $k_{2/5,\min}$  along with the penalty parameter  $\zeta_1$  to ensure a minimum rate of dephosphorylation. The time adaption of  $k_{2/5}$  after a stretch change is given by the second part of (10), wherein  $\dot{\lambda}_{p,\max}$ ,  $\dot{\lambda}_{p,\min}$ ,  $\gamma_5$  are parameters and  $\zeta_2$  is a penalty parameter which ensures a slow relaxation of the SMC by limiting  $\lambda - \bar{\lambda}_p$  to a minimum value of  $\Delta \bar{\lambda}_{p,\min}$ . Further, the target stretch-dependent MLCP activity is given as

$$k_{2/5,\text{tar}} = \gamma_6 \left( 1 - \frac{\lambda}{\lambda + \lambda_{50,p}} \right), \quad (11)$$

where  $\gamma_6$  is a material parameter and  $\lambda_{50,p}$  is the half activation stretch.

### 2.3 | Effect of pharmacological agents

Here, we are interested in modeling the effects of CCBs on intracellular calcium concentration. Clinically, dihydropyridine CCBs are effective vasodilators and are widely used as antihypertensives. While CCBs can affect other components of the active, and even the passive material response of the arterial wall to a certain degree, we focus here on their basic functionality, which leads to a reduction of the flow of calcium ions through L-type calcium channels located on the smooth muscle cell membrane. This effect can be modeled by extending the chemical kinetics model given in Section 2.2 by making the target calcium concentration  $[\text{Ca}^{2+}]_{\text{tar}}$  and the intracellular calcium concentration  $[\text{Ca}^{2+}]_i$  dependent on the concentration of the CCB. This is accomplished by making the parameters  $\gamma_1$  and  $\gamma_3$  in (7) and (9) functions of the CCB concentration  $c_{\text{CCB}}$

$$\begin{aligned}\gamma_1(c_{\text{CCB}}) &= \gamma_{1,\max} \left( 1 - p_1 \frac{c_{\text{CCB}}^2}{c_{\text{CCB}}^2 + c_{\text{CCB},50}^2} \right), \\ \gamma_3(c_{\text{CCB}}) &= \gamma_{3,\max} \left( 1 - p_3 \frac{c_{\text{CCB}}^2}{c_{\text{CCB}}^2 + c_{\text{CCB},50}^2} \right),\end{aligned}\quad (12)$$

wherein  $\gamma_{1,\max}$  and  $\gamma_{3,\max}$  are the maximum attainable values, whereas  $p_1$  and  $p_3$  are parameters and  $c_{\text{CCB},50}$  is the half activation CCB concentration. Currently, there is no experimental data available to support the correct choice of (12). Accordingly, we apply sigmoid functions, which are widely used to characterize biological processes with two limits, here, the maximal inflow of calcium into the cell and the suppression of calcium inflow. A more detailed adjustment of (12) is planned as future work based on experimental data.

### 2.4 | Smooth muscle cell activation model

From the work of Uhlmann and Balzani [63] we have the active arterial response

$$\Psi_a^{(f)} = \frac{\mu_a}{2} \left( n_C^{(f)} + n_D^{(f)} \right) (\lambda_e^{(f)} - 1)^2, \quad (13)$$

wherein,  $\mu_a$  is a stiffness constant,  $n_C^{(f)}$  and  $n_D^{(f)}$  are the fraction of myosin heads in states C and D, respectively. The stretch  $\lambda^{(f)}$  is multiplicatively split into an elastic  $\lambda_e^{(f)}$  and an active part  $\lambda_a^{(f)}$

$$\lambda^{(f)} = \lambda_e^{(f)} \lambda_a^{(f)}. \quad (14)$$



The rate of change of active strain is given by

$$\dot{\lambda}_a^{(f)} = \beta_1 \frac{P_a^{(f)} - P_c^{(f)}}{P_a^{(f)} - \beta_2}, \quad (15)$$

where  $\beta_1$  and  $\beta_2$  are material parameters,  $P_a^{(f)}$  and  $P_c^{(f)}$  are the active and driving stresses, respectively and are given by

$$\begin{aligned} P_a^{(f)} &= \mu_a \left( n_C^{(f)} + n_D^{(f)} \right) \left( \lambda_e^{(f)} - 1 \right), \\ P_c^{(f)} &= \kappa n_C^{(f)}, \end{aligned} \quad (16)$$

where  $\kappa$  is the maximum driving stress. The evolution equations of the smooth muscle model, that is, (8), (10), and (15), are solved by applying the backward Euler integration scheme. Further information can be found in Sect. 2.4 of the original publication [63].

### 3 | COUPLED DIFFUSION-DEFORMATION PROBLEM

The problem is described by the coupling of balance of linear momentum with a diffusion-reaction equation describing the local distribution of  $n$  different drug concentrations  $c_1, \dots, c_n$ . The set of partial differential equations is given as:

$$\text{Balance of linear momentum in Lagrangian setting :} \quad \rho_s \frac{\partial^2 \mathbf{u}}{\partial t^2} - \nabla \cdot \mathbf{P}(\mathbf{u}, c_1, \dots, c_n) = \rho_s \mathbf{b}, \quad (17)$$

$$\text{Diffusion-reaction of drug in Euler setting:} \quad \frac{\partial c_i}{\partial t} - \nabla_S \cdot \mathbf{D}_S \nabla_S c_i - r(c_i) = 0, \quad (18)$$

where  $\nabla$  and  $\nabla_S$  refer to derivatives with regard to  $\mathbf{X}$  and  $\mathbf{x}$ , respectively. The function  $r(c_i)$  describes, for example, the metabolism rate resulting in the reduction of drug  $i$ .  $\mathbf{P}$  is the first Piola–Kirchhoff stress tensor,  $\rho_s$  the density in the reference configuration [63], and  $\mathbf{D}_S$  the diffusion tensor in the current configuration. The first Piola–Kirchhoff stress tensor depends on the drug concentrations  $c_1, \dots, c_n$  as shown for  $c_{CCB}$  in (12). Accordingly, a change of the drug concentration influences the balance of linear momentum via  $\mathbf{P}$ . In our simulations, both processes occur simultaneously, namely the deformation due to blood pressure variations and the diffusion of drugs through the arterial wall.

The set of equations is solved using the finite element method. We apply the Galerkin method to obtain the weak form of (17) and (18) with test functions  $\mathbf{v}$  and  $v$  for the physical body in the reference configuration  $B \subset \mathbb{R}^3$ . For a more detailed derivation, we refer to [66]. Furthermore, only the effect of one antihypertensive is investigated in the presented simulations. Therefore, we do not include all drug concentrations  $c_1, \dots, c_n$  into the upcoming equations, but reduce them to one single drug concentration  $c$ . With  $\mathbf{C}_v = \nabla \mathbf{v}^T \mathbf{F} + \mathbf{F}^T \nabla \mathbf{v}$  and second Piola–Kirchhoff stress tensor  $\mathbf{S}(\mathbf{C}(\nabla \mathbf{u}), c)$  we obtain the weak form of the balance of linear momentum

$$G^u(\mathbf{u}, c, \mathbf{v}) := \int_B \rho_s \frac{\partial^2 \mathbf{u}}{\partial t^2} \cdot \mathbf{v} \, dX + \int_B \frac{1}{2} \mathbf{C}_v : \mathbf{S}(\mathbf{C}(\nabla \mathbf{u}), c) \, dX - \int_B \rho_s \mathbf{b} \cdot \mathbf{v} \, d\hat{x} - \int_{\partial B} \mathbf{v} \cdot \mathbf{T}_0 \, dX = 0. \quad (19)$$

Here, on the Neumann boundary  $\mathbf{T}_0 = \mathbf{P} \cdot \mathbf{n}_0$  describes the stresses in the outward normal direction. We apply deformation-dependent loads as described in Chap. 4.2.5 of [66], where the corresponding pressure values are inserted. Since blood flow is not considered in the presented simulations, these deformation-dependent pressure values aim to characterize the intravascular pressure on the arterial wall, which is usually applied by the blood flowing through the artery. Note that this generates another nonlinear component to the tangent matrix, which is included in the structure block.

Transforming the volume elements to the reference configuration  $dx = JdX$  with  $J = \det(\mathbf{F})$  with the deformation gradient  $\mathbf{F}$ , we obtain the weak form of the diffusion-reaction equation:

$$G^c(\mathbf{u}, c, v) := \int_B \frac{\partial c}{\partial t} \cdot v \, dX + \int_B \nabla c \cdot \mathbf{D} \cdot \nabla v - r(c) v J \, dX - \int_{\partial B} \mathbf{D} \cdot \nabla c \cdot \mathbf{n}_0 v J \, dX = 0, \quad (20)$$

with the diffusion tensor in the reference configuration obtained as pull-back operation  $\mathbf{D} := \mathbf{F}^{-1} \mathbf{D}_S \mathbf{F}^{-T}$ . Usually biological tissues are considered to be incompressible, despite the fact that significant volume changes have been reported. Therefore, we keep the Jacobian  $J$  in the weak form in our general representation here and, thus, consider the mechanical back-coupling in our implementation, although the material parameters in the volumetric energy are chosen to reflect a rather incompressible response. We plan to analyze the influence of volume changes on the coupling strategy in future work.

The coupling conditions reflect the influence of the drug on the deformation process and the extent of the deformation on the diffusion process. Generally, the diffusion process is influenced by the deformation via the change in element volume. More precisely, the determinant of the deformation gradient changes when the geometry deforms. For almost incompressible material the influence diminishes. The influence of drug concentration is reflected depending on the underlying modeling of the arterial wall. We have nonlinear dependencies between the governing equations. Thus, the equations are linearized using the Newton–Raphson scheme.

Then, we define the residual  $\mathcal{R}$  with  $G^u(\mathbf{u}, c, \mathbf{v})$  and  $G^c(\mathbf{u}, c, \mathbf{v})$  for Newton iterations  $k = 0, 1, \dots$  and the linearization  $\Delta$  of the weak forms  $G^c$  and  $G^u$  in the Jacobian  $\mathcal{J}$ , respectively.

$$\mathcal{R}(\mathbf{u}^k, c^k) := \begin{pmatrix} G^u(\mathbf{u}, c, \mathbf{v}) \\ G^c(\mathbf{u}, c, \mathbf{v}) \end{pmatrix} \Big|_{(\mathbf{u}^k, c^k)}, \quad \mathcal{J}(\mathbf{u}^k, c^k) := \begin{pmatrix} \Delta_u[G^u] & \Delta_c[G^u] \\ \Delta_u[G^c] & \Delta_c[G^c] \end{pmatrix} = \begin{pmatrix} \mathbf{K}^{uu} & \mathbf{K}^{uc} \\ \mathbf{K}^{cu} & \mathbf{K}^{cc} \end{pmatrix} \Big|_{(\mathbf{u}^k, c^k)}.$$

Ultimately, the Newton solve for the increments  $\delta \mathbf{u}^{k+1}$  and  $\delta c^{k+1}$  is defined as:

$$\mathcal{J}(\mathbf{u}^k, c^k) \begin{pmatrix} \delta \mathbf{u}^{k+1} \\ \delta c^{k+1} \end{pmatrix} = -\mathcal{R}(\mathbf{u}^k, c^k). \quad (21)$$

We can extend (21) with respect to the time discretization. Time steps are labeled with  $n$ . The residual components are included in  $\mathbf{r}^u$  and  $\mathbf{r}^c$ . The following system of equations is implicit with respect to the coupling of diffusion and deformation:

$$\begin{bmatrix} \mathbf{K}_{n+1,k}^{uu} & \mathbf{K}_{n+1,k}^{uc} \\ \mathbf{K}_{n+1,k}^{cu} & \mathbf{K}_{n+1,k}^{cc} \end{bmatrix} \begin{bmatrix} \delta \mathbf{u}^{n+1,k+1} \\ \delta c^{n+1,k+1} \end{bmatrix} + \begin{bmatrix} \mathbf{M}_{n+1,k}^u & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \delta \dot{\mathbf{u}}^{n+1,k+1} \\ \mathbf{0} \end{bmatrix} + \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{M}_{n+1,k}^c \end{bmatrix} \begin{bmatrix} \mathbf{0} \\ \delta \dot{c}^{n+1,k+1} \end{bmatrix} + \begin{bmatrix} \mathbf{r}_{n+1,k}^u \\ \mathbf{r}_{n+1,k}^c \end{bmatrix} = \mathbf{0}, \quad (22)$$

where  $M_{n+1,k}^u$  and  $M_{n+1,k}^c$  are mass matrices.

Finally, if we neglect the dynamic component and use a backward Euler scheme to discretize  $c$  in time, we obtain the following fully implicit scheme from (22):

$$\begin{bmatrix} \mathbf{K}_{n+1,k}^{uu} & \mathbf{K}_{n+1,k}^{uc} \\ \mathbf{K}_{n+1,k}^{cu} & \mathbf{K}_{n+1,k}^{cc} + \frac{1}{\Delta t} \mathbf{M}_{n+1,k}^c \end{bmatrix} \begin{bmatrix} \delta \mathbf{u}^{n+1,k+1} \\ \delta c^{n+1,k+1} \end{bmatrix} + \begin{bmatrix} \mathbf{r}_{n+1,k}^u \\ \mathbf{r}_{n+1,k}^c \end{bmatrix} = \mathbf{0}. \quad (23)$$

We use a  $\mathcal{P}_2 - \mathcal{P}_2$  finite element discretization. This is motivated by the fact that we allow small deviations from incompressibility and, thus, volumetric locking has not been observed, which is why we do not include any specialized mixed finite element formulation. In order to keep the required degrees of freedom for a desired level of accuracy small, we chose to use quadratic shape functions for both displacements and concentrations, although linear shape functions for the concentrations could be slightly more efficient considering the different magnitude in gradients per element for the two fields, at least in the boundary value problems analyzed here.

## 4 | TWO-LEVEL OVERLAPPING SCHWARZ PRECONDITIONERS

To efficiently solve the nonsymmetric system (23), we employ a preconditioned generalized minimal residual (GMRES) method [49]. The convergence of the (unpreconditioned) GMRES method will deteriorate when the mesh is refined. Hence, we will use preconditioning techniques to improve the convergence and to obtain parallel scalability for large problems. In particular, we use two-level overlapping Schwarz domain decomposition preconditioners with generalized



Dryja–Smith–Widlund (GDSW) coarse spaces [13, 15]. We use the FROSch (Fast and Robust Overlapping Schwarz) [25] domain decomposition solver package, which is part of the Trilinos [57, 58] library, for the implementation of these preconditioners; compare Section 5.2.

For the sake of simplicity, let us first describe the two-level Schwarz preconditioners just for a single-physics problem, for instance, for a diffusion problem

$$Kc = f,$$

resulting from a finite element discretization with a finite element space  $V$ . In Section 4.2, we discuss how to construct the preconditioners to block systems of the form (23). The preconditioners are then based on an overlapping domain decomposition of the computational domain  $\Omega$  into  $N$  overlapping subdomains  $\{\Omega'_i\}_{i=1, \dots, N}$ . In practice, we construct the overlapping domain decomposition from a nonoverlapping domain decomposition  $\{\Omega_i\}_{i=1, \dots, N}$  by recursively extending the subdomains by layers of finite elements; when extending the subdomains by  $k$  layers of elements with a width of  $h$ , the overlap has a width of  $\delta = kh$ . The overlapping subdomains then conform with the computational mesh on  $\Omega$ . Note that in our implementation (cf. Section 5.1), we construct the overlap algebraically, via the graph induced by the nonzero pattern of  $K$ . We will denote the size of the algebraically determined overlap with  $\hat{\delta} = k$ , for  $k$  layers of degrees of freedom. The initial nonoverlapping domain decomposition can be obtained using mesh partitioning algorithms, for instance, using the parallel ParMETIS library [37].

Let  $V_i$  be the finite element space on the local overlapping subdomain  $\Omega'_i$  with homogeneous zero boundary conditions on  $\partial\Omega'_i$ , and let  $R_i : V \rightarrow V_i$  be the restriction from the global to the local finite element space on  $\Omega'_i$ . The corresponding prolongation operators are then given by  $R_i^\top$ . Furthermore, we formally introduce a (global) coarse space  $V_0$  spanned by coarse basis functions corresponding to the columns of the matrix  $\Phi : V_0 \rightarrow V$ ; we will introduce specific coarse spaces in detail in Section 4.1.

The two-level additive overlapping Schwarz preconditioner reads

$$M^{-1} = \underbrace{\Phi K_0^{-1} \Phi^\top}_{\text{second level}} + \sum_{i=1}^N \underbrace{R_i^\top K_i^{-1} R_i}_{\text{first level}}, \quad (24)$$

where  $K_0$  is the coarse matrix,

$$K_0 = \Phi^\top K \Phi, \quad (25)$$

and  $K_i = R_i K R_i^\top$  are the local matrices. Note that the coarse level corresponds to solving a globally coupled coarse problem, whereas the first level corresponds to solving  $N$  decoupled local problems that can be solved concurrently in parallel computations. For more details on standard Schwarz preconditioners, we refer to [52, 59].

#### 4.1 | GDSW and RGDSW coarse spaces for overlapping Schwarz

The coarse level, which is determined by the choice of the coarse space, is essential for the numerical scalability of the solver. In particular, without a suitable coarse level, the convergence of the Krylov method with a one-level Schwarz preconditioner will increase with an increasing number of subdomains. Here, we will focus on coarse spaces that can be constructed from the parallel distributed system matrix, requiring little to no additional information.

In particular, we consider GDSW [13, 14] and reduced GDSW (RGDSW) [15] coarse spaces. Let us discuss both approaches in a common algorithmic framework. Both approaches start with a decomposition into nonoverlapping subdomains  $\{\Omega_i\}_{i=1, \dots, N}$ , as already mentioned in Section 4. First, we define the interface of the nonoverlapping domain decomposition as

$$\Gamma = \bigcup_{i \neq j} (\partial\Omega_i \cap \partial\Omega_j) \setminus \partial\Omega_D,$$

where  $\partial\Omega_D$  is the global Dirichlet boundary. Then, we decompose the interface  $\Gamma$  into components  $\gamma_k \subset \Gamma$  such that

$$\Gamma = \bigcup_k \gamma_k. \quad (26)$$

This decomposition may be disjoint ( $\gamma_i \cap \gamma_k = \emptyset$  for  $i \neq k$ ) or overlapping. A typical choice is to decompose the interface into faces, edges, and vertices, where (in three dimensions)

- a face is a set of nodes that belong to the same two subdomains,
- an edge is a set of at least two nodes that belong to the same (more than two) subdomains,
- a vertex is a single node that belongs to the same (more than two) subdomains.

Then, we define functions  $\phi_l^{\gamma_k}$  on  $\Gamma$  with  $\text{supp}(\phi_l^{\gamma_k}) \subset \gamma_k$  such that

$$N_\Gamma \subset \text{span}\{\phi_l^{\gamma_k}\}_{l,k},$$

where  $N_\Gamma$  is the restriction of the null space  $N$  of the global Neumann matrix to the interface  $\Gamma$ . The global Neumann matrix is obtained by assembling the original problem on  $\Omega$  but with  $\partial\Omega_D = \emptyset$ , that is, a Neumann condition on  $\partial\Omega$ . Then, we gather the functions as columns of a matrix

$$\Phi_\Gamma = \left( \phi_1^{\gamma_1} \quad \phi_2^{\gamma_1} \quad \dots \quad \phi_1^{\gamma_2} \quad \dots \right).$$

The matrix  $\Phi$  is obtained from computing the discrete energy-minimizing extension of the interface values  $\Phi_\Gamma$  to the interior of the subdomains

$$\Phi = \begin{bmatrix} \Phi_I \\ \Phi_\Gamma \end{bmatrix} = \begin{bmatrix} -\mathbf{K}_{II}^{-1} \mathbf{K}_{I\Gamma} \Phi_\Gamma \\ \Phi_\Gamma \end{bmatrix}, \quad (27)$$

where the index  $I$  indicates all degrees of freedom that do not correspond to the interface  $\Gamma$ , and  $\mathbf{K}_{II}$  and  $\mathbf{K}_{I\Gamma}$  are obtained by reordering and partitioning  $\mathbf{K}$  as follows:

$$\mathbf{K} = \begin{bmatrix} \mathbf{K}_{II} & \mathbf{K}_{I\Gamma} \\ \mathbf{K}_{\Gamma I} & \mathbf{K}_{\Gamma\Gamma} \end{bmatrix}.$$

As mentioned before, we will employ GDSW and RGDSW coarse spaces in this paper. Based on the algorithmic framework described above, they can be easily introduced.

In the GDSW coarse space, we choose the interface components  $\gamma_k^{\text{GDSW}}$  as faces, edges, and vertices. Then, the functions  $\phi_l^{\gamma_k^{\text{GDSW}}}$  are defined as the restrictions of a basis of the null space  $N$  to the interface components  $\gamma_k^{\text{GDSW}}$ . For example, in the case of a scalar diffusion problem, the null space consists only of constant functions, and we can set  $\phi_l^{\gamma_k^{\text{GDSW}}}$ ,  $l = 1$ , to one on  $\gamma_k^{\text{GDSW}}$  and zero elsewhere.

For a homogeneous two-dimensional diffusion problem with irregular subdomains as, for instance, obtained by METIS, the two-level Schwarz preconditioner (24) with the GDSW coarse space yields the condition number estimate

$$\kappa(\mathbf{M}^{-1}\mathbf{K}) \leq C \left( 1 + \frac{H}{\delta} \right) \left( 1 + \log \left( \frac{H}{h} \right) \right)^2;$$

compare [13]. Here,  $h$  is the typical diameter of the finite elements,  $H$  the diameter of the subdomains, and  $\delta$  is the overlap.

The construction of the RGDSW coarse space employed in this paper (“Option 1” in [15]) is based on a different, overlapping decomposition of the interface, generally resulting in a coarse space of lower dimension. In particular, let us consider the interface components  $\gamma_k^{\text{GDSW}}$  from the GDSW coarse space, which correspond to faces, edges, and vertices, and let  $S_k^{\text{GDSW}}$  be the set of subdomains that  $\gamma_k^{\text{GDSW}}$  belongs to. Then, we select all interface components  $\gamma_l^{\text{GDSW}}$  with

$$\nexists \gamma_l^{\text{GDSW}}, l \neq k : S_k^{\text{GDSW}} \subset S_l^{\text{GDSW}},$$

and we denote them by  $\hat{\gamma}_k^{\text{RGDSW}}$ . To satisfy (26), we choose

$$\gamma_k^{\text{RGDSW}} = \hat{\gamma}_k^{\text{RGDSW}} \cup \bigcup_{S_l^{\text{GDSW}} \subset S_k^{\text{RGDSW}}} \gamma_l^{\text{GDSW}},$$

that is, we add all edges, faces, and vertices that belong to a subset of the subdomains that  $\hat{\gamma}_k^{\text{RGDSW}}$  belongs to. To define the corresponding interface functions  $\phi_l^{\gamma_k^{\text{RGDSW}}}$ , we define a scaling function  $s : \Gamma \rightarrow \mathbb{R}$  with

$$s(n) = \left| \{ \gamma_k^{\text{RGDSW}} \mid n \in \gamma_k^{\text{RGDSW}} \} \right|,$$

for  $n \in \Gamma$ . Denoting by  $\hat{\phi}_l^{\gamma_k^{\text{RGDSW}}}$  restrictions of a basis of the null space  $N$  to the interface components  $\gamma_k^{\text{RGDSW}}$ , we define

$$\phi_l^{\gamma_k^{\text{RGDSW}}}(n) = \frac{\hat{\phi}_l^{\gamma_k^{\text{RGDSW}}}(n)}{s(n)},$$

for  $n \in \Gamma$ ; this means that in each node  $n$  we scale the restrictions of the basis functions of the null space by the inverse of the number of RGDSW interface components the node belongs to. For a simple homogeneous diffusion problem, the two-level Schwarz preconditioner in (24) with the RGDSW coarse space yields the condition number estimate

$$\kappa(\mathbf{M}^{-1}\mathbf{K}) \leq C \left( 1 + \frac{H}{\delta} \right) \left( 1 + \log \left( \frac{H}{h} \right) \right);$$

compare [15].

## 4.2 | Monolithic overlapping Schwarz preconditioners

For preconditioning block systems of the form (23), we employ monolithic Schwarz preconditioning techniques; compare [38, 39]. In particular, we use monolithic GDSW preconditioners

$$\mathcal{M}^{-1} = \phi \mathcal{K}_0^{-1} \phi^\top + \sum_{i=1}^N \mathcal{R}_i^\top \mathcal{K}_i^{-1} \mathcal{R}_i, \quad (28)$$

as introduced in [22, 23] for a block matrix

$$\mathcal{K} = \begin{bmatrix} \mathcal{K}_{u,u} & \mathcal{K}_{u,c} \\ \mathcal{K}_{c,u} & \mathcal{K}_{c,c} \end{bmatrix}. \quad (29)$$

In (28), the local subdomain matrices are given by  $\mathcal{K}_i = \mathcal{R}_i \mathcal{K} \mathcal{R}_i^\top$  with restriction operators

$$\mathcal{R}_i = \begin{bmatrix} \mathcal{R}_{u,i} & 0 \\ 0 & \mathcal{R}_{c,i} \end{bmatrix},$$

where  $\mathcal{R}_{u,i}$  and  $\mathcal{R}_{c,i}$  correspond to the restriction operators of the  $u$  and  $c$  degrees of freedom, respectively, for  $\Omega_i'$ . Also the coarse matrix  $\mathcal{K}_0 = \phi^\top \mathcal{K} \phi$ , has a block structure given by the matrix

$$\phi = \begin{bmatrix} \phi_{u,u_0} & \phi_{u,c_0} \\ \phi_{c,u_0} & \phi_{c,c_0} \end{bmatrix},$$

which has the coarse basis functions as its columns.

It is constructed analogously to (27) based on the null spaces of the Neumann matrices corresponding to the diagonal blocks of  $\mathcal{K}$ ; compare (29). See also [28] for monolithic Schwarz preconditioners for land ice problems, which have a similar matrix structure as (23).

## 4.3 | Recycling strategies

To save computing time, information from previous time steps or Newton iterations can be reused. Specifically, three recycling strategies will be analyzed further: the reuse of the symbolic factorizations of local subdomain matrices  $\mathcal{K}_i$  and

**TABLE 1** Recycling strategies as described in Section 4.3.

Abbreviations	Recycling strategy
–	No recycling is used
(SF)	Reuse of symbolic factorizations of local subdomain matrices $\mathcal{K}_i$ and of the coarse matrix $\mathcal{K}_0$
(CB)	Reuse of the coarse basis $\Phi$
(CM)	Reuse of the entire coarse matrix $K_0$

of the coarse matrix  $\mathcal{K}_0$ —we denote this strategy by (SF)—the reuse of the coarse basis  $\Phi$ , which is used to assemble  $\mathcal{K}_0$  (cf. (25))—we denote this strategy by (CB)—and the reuse of the entire coarse matrix  $K_0$ , denoted by (CM); see also Table 1. Note that when recycling is used, the reuse of  $\mathcal{R}_i$  and  $\mathcal{R}_i^T$  is always included.

Generally, the reuse of a symbolic factorization is advantageous as the nonzero pattern of the system typically stays the same. Reusing the entire coarse matrix  $\mathcal{K}_0$  not only eliminates the time for setting it up but also the time required for its factorization. The reuse of the coarse basis and the coarse matrix, respectively, are more invasive strategies. They can further decrease the setup time but can also increase the iteration count, as the preconditioner no longer adapts exactly to the new Newton iteration or time step. Note, however, that we are still solving the correct tangent matrix system in each Newton step since only the preconditioner is affected.

## 5 | SOFTWARE ECOSYSTEM

For the large-scale simulations of arterial walls that are characterized by the models in Section 2, we use the different software libraries AceGen, Trilinos, FEDDLib, and FROSch together with a newly developed AceGen–FEDDLib interface. As the main software package, we use the library FEDDLib [55] (**F**inite **E**lement and **D**omain **D**ecomposition **L**ibrary), and for some functionalities, we make use of other libraries. In particular, we employ data services, parallel linear algebra, solvers, and preconditioners to solve our systems efficiently from the open-source software library Trilinos [57]. For the implementation of the solid material models, we use the commercial symbolic software AceGen, which is based on Mathematica. More specifically, we generate the code that implements the material models using AceGen and call it through an interface between AceGen and FEDDLib.

Through an interface between AceGen and FEDDLib, which has been developed within the present project, the assembled structure–chemical interaction system can be passed along. FEDDLib then uses the solvers and preconditioners provided by Trilinos to solve the system. We especially focus on the Trilinos package FROSch (**F**ast and **R**obust **O**verlapping **S**chwarz), which includes the parallel implementation of the two-level overlapping Schwarz preconditioner as described in Section 4.

### 5.1 | FEDDLib and Trilinos

FEDDLib [55] is a C++-based, object-oriented finite element library built on top of the Trilinos software infrastructure. Trilinos [57, 58] is a software framework containing mathematical software libraries for the solution of large-scale, complex multiphysics engineering and scientific problems; compare [30]. It is organized as a collection of (mostly) interoperable packages. The packages are categorized in the six product areas *Data Services*, *Linear Solvers*, *Nonlinear Solvers*, *Discretizations*, *Framework*, and *Product Manager*.

The main design principle of FEDDLib is to reach outstanding parallel efficiency and robustness for complex single and multiphysics applications by closely integrating finite element discretization and domain decomposition-based solvers. In particular, the finite element assembly in FEDDLib provides the data structures required to make use of the full potential of state-of-the-art domain decomposition algorithms; in this context, the main focus is on overlapping Schwarz solvers in Trilinos' domain decomposition solver package FROSch [25].

The main components of FEDDLib can be categorized into *linear algebra*, *mesh handling*, *finite element assembly*, *boundary condition handling*, *linear and nonlinear solvers*, *time-stepping*, *parallel I/O*, and *application-specific classes*; an overview over the functionality in the year 2020 is given in the PhD thesis of C. Hochmuth [31]. Afterwards, further significant developments have been carried out, for instance, on adaptive mesh refinement by one of the authors [50]

and an interface for AceGen generated code, which is briefly discussed in Sections 5.3 and 5.4. Application problems currently implemented in FEDDLib range from simple scalar elliptic problems, such as diffusion, to complex multiphysics problems, such as fluid–structure interaction with nonlinear models for the fluid and solid.

FEDDLib provides wrappers and interfaces to many packages of Trilinos. First, the parallel linear algebra in FEDDLib is done via wrappers for the parallel linear algebra classes in Trilinos (product area *Data Services*); in particular, FEDDLib uses Xpetra, which in turn is a lightweight interface for both the Epetra and Tpetra linear algebra frameworks in Trilinos. Whereas Epetra is the older linear algebra framework from the origins of Trilinos, Tpetra is the current linear algebra framework. One of the main advancements of Tpetra compared with Epetra is the integration of the performance portability libraries Kokkos [16, 61, 62] and Kokkos Kernels [48]. This also enables the efficient use of GPUs; see, for instance, [67] for the application of FROSch on GPUs via Kokkos and Kokkos Kernels.

Therefore, we mostly use the newer Tpetra package, and we will focus on the software stack based on Tpetra in our discussion here. In particular, FEDDLib provides interfaces to Trilinos' nonlinear solver package NOX (product area *Nonlinear Solvers*) as well as several packages for linear solvers and preconditioners (product area *Linear Solvers*), which we access via the unified solver interface Stratimikos [5], which in turn makes use of the interoperability framework Thyra [6]. In particular, we employ the Belos and Amesos2 [7] packages for iterative and direct solvers, respectively; note that the direct solvers—except for the serial direct solver KLU—are not part of Trilinos and can therefore only be called via the interface Amesos2. For preconditioning, we mostly use the domain decomposition package FROSch [25] and the block-preconditioning package Teko [12]; other available options are Ifpack2, which combines one-level Schwarz methods with inexact factorization preconditioning techniques, and the algebraic multigrid package MueLu [8]. In Section 5.2, we will discuss the FROSch package in more detail since it implements the Schwarz domain decomposition preconditioners discussed in Section 4 and investigated in our Section 6 on numerical results. Finally, the toolbox package Teuchos (product area *Framework*), which provides smart pointers, parameter lists, and XML parsers, as well as the graph partitioning and load-balancing package Zoltan2 (product area *Data Services*) for mesh partitioning are employed.

## 5.2 | FROSch

The Trilinos package FROSch (Fast and Robust Overlapping Schwarz) [25] provides a highly scalable, parallel framework for overlapping Schwarz domain decompositions solvers; the first version of the implementation was described in [21, 26] and the extension to monolithic GDSW and RGDSW preconditioners in [22].

The design of the FROSch package is based on the concept of combining Schwarz operators in additive or multiplicative ways, resulting in different types of Schwarz preconditioners; compare [25, 26, 59]. The Schwarz preconditioners are constructed algebraically, that is, based on the sparsity pattern of the parallel-distributed system matrix. For the first level, the overlapping subdomains are constructed from nonoverlapping subdomains by extending the subdomains by layers of adjacent degrees of freedom. Adjacency is defined based on nonzero off-diagonal entries in the system matrix.

For the coarse level, we employ extension-based coarse spaces, such as GDSW and RGDSW; compare Section 4.1. The construction described in Section 4.1 is algebraic except for the fact that the null space of the Neumann matrix is required. In particular, for elasticity problems, the null space is spanned by the rigid body modes, and the (linearized) rotations cannot be constructed algebraically; they have to be either provided by the user or constructed using coordinates of the finite element nodes. For more details on the algebraic construction of GDSW-type coarse spaces for elasticity problems, see [24].

FROSch preconditioners can be easily constructed via Trilinos' unified solver interface Stratimikos using only the parallel-distributed system matrix and a list of parameters defining the specific setup of the preconditioners. FROSch is based on the Xpetra wrappers, which wrap the Epetra and Tpetra linear algebra frameworks of Trilinos; compare the discussion in Section 5.1. If Tpetra is used, the performance portability libraries Kokkos and Kokkos Kernels can be employed, enabling the use on, for instance, GPU architecture [67]. Due to its algebraic implementation, FROSch preconditioners can be constructed recursively, resulting in multi-level Schwarz preconditioners; see [29].

## 5.3 | AceGen

AceGen [40] is a Mathematica package that is used to automatically derive mathematical expressions required in numerical procedures. AceGen exploits the symbolic and algebraic capabilities of Mathematica by combining them with a hybrid

symbolic-automatic differentiation technique. AceGen uses automatic code generation and simultaneous optimization of expressions. Furthermore, AceGen's ability to generate code for multiple languages can be leveraged to generate finite element code to run simulations with, for example, FEAP (Finite Element Analysis Program), Elfen, Abaqus, and the Mathematica-based finite element environment AceFEM. Using the AceGen–AceFEM combination, new finite elements can be rapidly developed and tested. Here, we use AceGen to generate finite element code with consistent linearizations, which is important for the optimal performance of the Newton algorithm.

## 5.4 | Interface

The newly developed AceGen interface enables the use of AceGen-generated AceFEM (AceGen–AceFEM) finite elements in external software libraries, by providing C++ finite element classes that can be used to compute element-level quantities such as residuals, stiffness matrices, history variables and postprocessing quantities. The interface contains a set of functions that handle the manipulation of data structures that are required by the AceGen–AceFEM finite elements. It greatly simplifies the use of the generated finite elements by providing a unified object-oriented interface and obscures low-level memory management, which tends to be error-prone. The interface theoretically also enables the use of the various finite elements available in the AceShare finite element library (which contains numerous community-contributed finite elements), in C++-based finite element codes.

Accordingly, the corresponding FEDDLib assembly routines have been modified to include the externally generated AceGen–AceFEM finite elements through the AceGen interface, enabling their use in high-performance computing environments. An object-oriented, factory-based concept enables specific assembly routines to be built on top of a generalized base class, which defines the interface. Depending on the specific problem at hand, the base class' pure virtual functions, most importantly `getRhs()` and `getJacobian()`, are overridden in the derived class, with the corresponding finite element assembly routines. Mainly, FEDDLib only interacts with the base class to retrieve the element's residual vector (`getRhs()`) and Jacobian matrix (`getJacobian()`), that are necessary to setup the linearized Newton system as depicted in (21). The elementwise system matrix and vector—for the structure–chemical interaction, the Jacobian would be the 2x2 block system—is then assembled into the parallelly distributed `Tpetra::CrsMatrix` and `Tpetra::MultiVector` data types. These are in turn passed along to the linear solvers and preconditioners in Trilinos.

## 6 | NUMERICAL RESULTS

For the numerical analysis of the coupling of the equations of linear momentum and reaction-diffusion (Section 3), we primarily focus on two aspects: the pharmaco-mechanical effects induced by the presence of medication and the weak and strong parallel scalability of the solvers.

As described in Section 3 for time discretization we apply the backward Euler scheme. As we neglect the dynamic component in the structural part, we employ different loading protocols for our numerical tests. Here, the load steps coincide with the time steps. For the discretization of (23), we use tetrahedra with piecewise polynomials of degree two for both the displacement and the concentration.

Except for the weak scaling results in Section 6.2, all meshes are unstructured. We use Gmsh [19] to generate the unstructured meshes, and we partition them with METIS. The parallel results were obtained on the Fritz supercomputer at Friedrich-Alexander-Universität Erlangen-Nürnberg. It has 992 compute nodes, each with two Intel Xeon Platinum 8360Y “Ice Lake” processors (36 cores per chip, 2.4 GHz base frequency) and 256 GB of DDR4 RAM per node.

We use the Trilinos-based Newton solver NOX as the nonlinear solver (cf. Section 5.1), which offers, among other features, different globalization and forcing term strategies for inexact Newton. If  $J_{F_k} d_k = -F_k$  denotes the linear (non-symmetric) system to be solved (see (21)) in the  $k$ th Newton step, we use GMRES [49] (without restart; choosing a low restart value was not robust or did not even converge) to find an approximate solution  $\tilde{d}_k$  such that

$$\left\| J_{F_k} \tilde{d}_k + F_k \right\|_{l_2} \leq \eta_k \|F_k\|_{l_2}, \quad (30)$$

where the forcing term (or tolerance)  $\eta_k = 10^{-6}$  is not changed between Newton steps. We employ no globalization strategies. In our implementation, this corresponds to restricting the line search in NOX to a full Newton step. Newton's method is stopped if the  $l_2$ -norm of the update  $\tilde{d}_k$  or the relative  $l_2$ -norm of the Newton residual (i.e.,  $\|F_k\|_{l_2} / \|F_0\|_{l_2}$ )

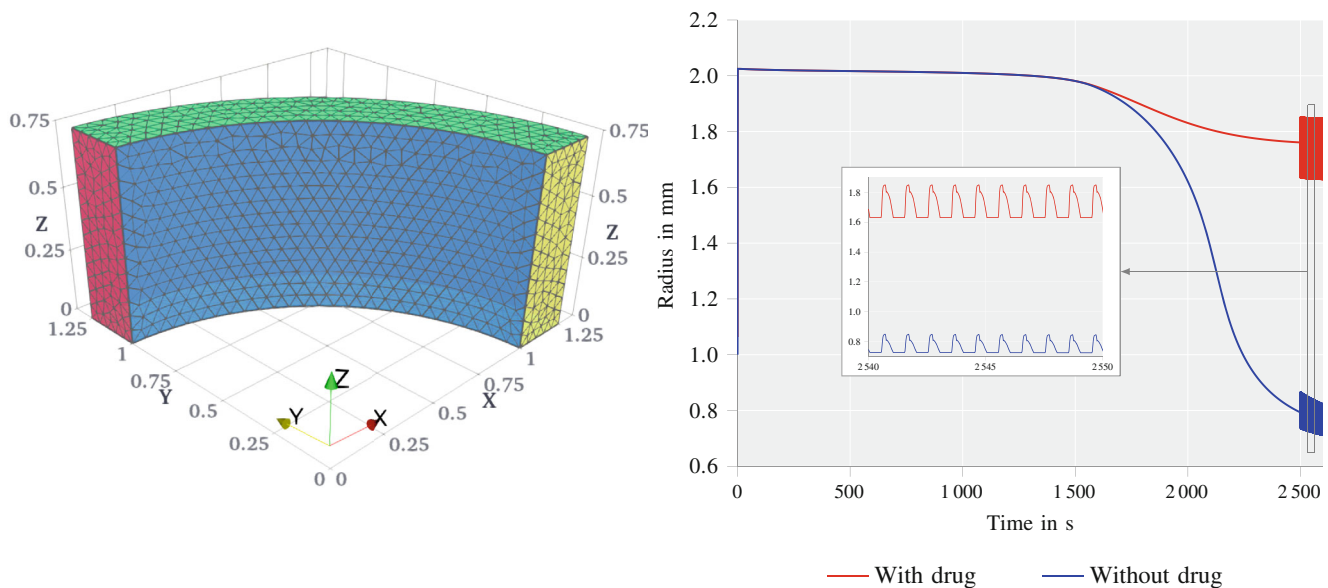


is smaller than  $10^{-8}$ . The initial vector for the Newton iteration is the solution from the previous time step; the initial vector for GMRES is zero. The GMRES method is preconditioned with the two-level additive overlapping Schwarz method (Section 4), where the coarse space is either GDSW or RGDSW. As direct solvers to construct the preconditioner, we use Intel MKL Pardiso [51] on the subdomains and SuperLU\_DIST [41] on the coarse level. SuperLU\_DIST is well-suited for the coarse solve, as we can execute it in parallel. Here, we use five processor cores to solve the coarse problem. Intel MKL Pardiso is used without threading. For the numerical results of this paper, we have always chosen the algebraically determined overlap  $\hat{\delta}$  as one layer of degrees of freedom.

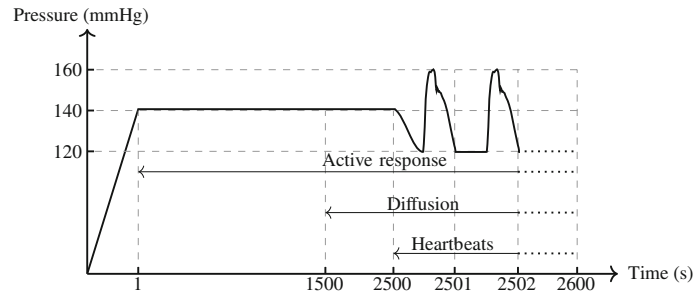
## 6.1 | Pharmaco-mechanical results

For the analysis of pharmaco-mechanical effects, we consider an idealized arterial segment with inner radius 1 mm, outer radius 1.25 mm and axial length of 0.75 mm; see Figure 1 (left). This geometry is chosen to match a realistic wall thickness of a middle cerebral rat artery, but is not matched to real measurements. However, an idealized geometry enables a more characteristic investigation of fundamental aspects of the overall strategy than considering a real, patient-specific artery [2], where results could be rather specific. The two fiber directions are crosswise helically wound and lie at  $30^\circ$  to the circumferential axis on the axial-circumferential plane, at every material point. The simulation protocol is given in Figure 2.

Between 0 and 1 s, we have a linear increase in pressure to 140 mmHg, which is then held constant until 2 500 s. Since simulating heartbeats that have systolic and diastolic pressures ranging from 120 to 160 mmHg from the beginning would be computationally expensive, we use a constant average pressure of 140 mmHg instead. During this first second of the simulation, the initial values for the fractions of the cross-bridges in their four states are set to  $n_A = 1$ ,  $n_B = 0$ ,  $n_C = 0$ , and  $n_D = 0$ . Accordingly, there is no initial active material response, since  $n_C$  and  $n_D$  are set to 0 (see (16)). Starting from 1 s, the active response of the material is turned on, in the consequence of which the evolution equations stated in Section 2 are solved using the backward Euler scheme at every time step, which leads to a change of the fractions  $n_C$  and  $n_D$  to positive values. It takes over 2500 s for the various internal variables to reach a steady state, which may be understood as corresponding to the physiological range. We observe an initial slow contraction, followed by a rapid contraction phase, as



**FIGURE 1** Left: Arterial segment (quarter of tube) with inner radius 1 mm, wall thickness 0.25 mm and length 0.75 mm. Dirichlet boundary conditions: fixed in  $x$  direction on  $x = 0$  plane (red), fixed in  $y$  direction on  $y = 0$  plane (yellow), fixed in  $z$  direction on  $z = 0$  and  $z = 0.75$  planes (green). On the interior wall (blue), pressure is applied. Right: Comparison of the response of the arterial radius with and without drug diffusion, with a detailed extract for the time between 2540 and 2550 s. Our material model and the protocol in Figure 2 are used; the drug diffusion is turned off for one of the simulations. The oscillations in the vessel radius from 2500 s onwards are caused by the onset of heartbeats.



**FIGURE 2** Loading protocol for long-term calculations with active response, diffusion process, and 100 heartbeats. Loading phase from  $t = 0$  s to  $t = 1$  s with 20 load steps. A constant pressure of 140 mmHg is applied until  $t = 2500$  s with  $\Delta t = 0.25$  s. Active response starts at  $t = 1$  s. Drug diffusion starts at  $t = 1500$  s. Starting at  $t = 2500$  s, a total of 100 heartbeats are simulated with  $\Delta t = 0.02$  s. The simulation ends at  $t = 2600$  s. The time-dependent change of the loading during the heartbeats was generated based on the flow rate profile from [3], which is based on data from [56].

**TABLE 2** Passive material parameters (adopted from [63]).

Parameter	$\alpha_1$	$\alpha_2$	$\alpha_3$	$\alpha_4$	$\alpha_5$
Value	11.52507 kPa	151.73775 kPa	2.75662	1.27631 kPa	3.08798

**TABLE 3** Parameters for the chemical model (adopted from [63]).

Parameter	$k_3$	$k_4$	$k_7$	$Ca_{50}$	$\gamma_2$	$\gamma_3$	$\lambda_{50, c}$	$\bar{\lambda}_{c, start}$	$\lambda_{a, start}$
Value	$0.134 \text{ s}^{-1}$	$0.00166 \text{ s}^{-1}$	$0.000066 \text{ s}^{-1}$	$0.4 \mu\text{M}$	$50 \mu\text{M}^{-1}$	$0.9 \mu\text{M}$	1.2	1.0	1.0
Parameter	$\gamma_4$	$\zeta_1$	$\gamma_5$	$\zeta_2$	$\Delta\bar{\lambda}_{p, min}$	$\gamma_6$	$\lambda_{50, p}$	$\bar{\lambda}_{p, start}$	$\lambda_{e, start}$
Value	200	100 s	50 s	1000	-0.00001	$1.5 \text{ s}^{-1}$	1.0	1.0	1.0

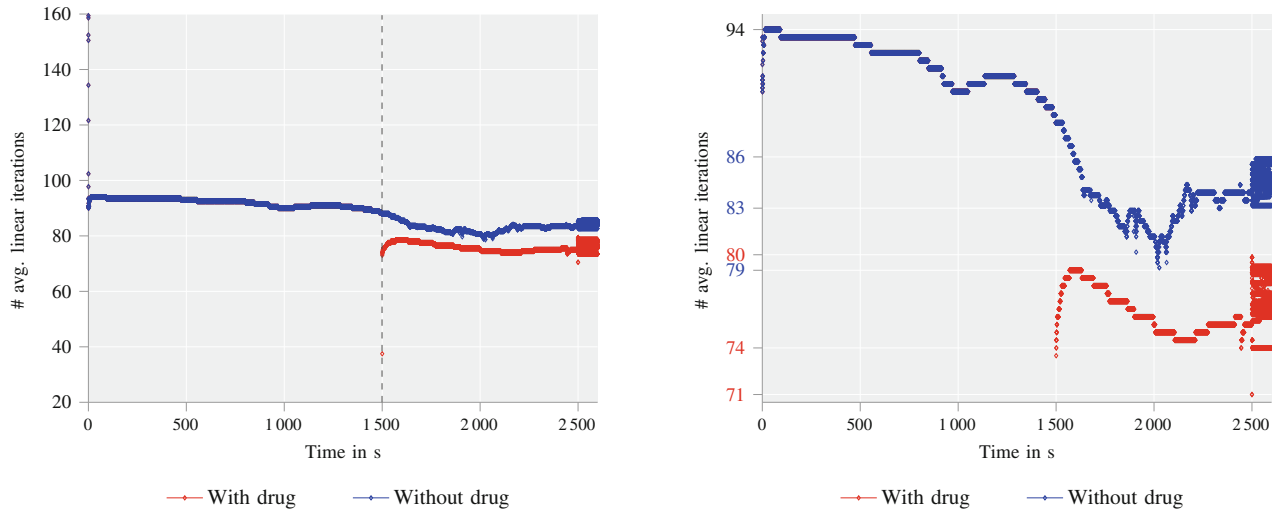
**TABLE 4** Parameters for the active chemo-mechanical material model (adopted from [63]).

Parameter	$\eta$	$\gamma_1$	$\dot{\bar{\lambda}}_{c, max}$	$\dot{\bar{\lambda}}_{c, min}$	$\dot{k}_{2/5, max}$	$\dot{k}_{2/5, min}$
Value	$0.1624 \text{ s}^{-1}$	$0.5131 \mu\text{M}$	$0.0443 \text{ s}^{-1}$	$-0.0443 \text{ s}^{-1}$	$0.0009735 \text{ s}^{-2}$	$-0.0010694 \text{ s}^{-2}$
Parameter	$\dot{\bar{\lambda}}_{p, max}$	$\dot{\bar{\lambda}}_{p, min}$	$\mu_a$	$\kappa$	$\beta_1$	$k_{2/5, start}$
Value	$0.0000699 \text{ s}^{-1}$	$-0.0002323 \text{ s}^{-1}$	11.857 kPa	148.262 kPa	$0.001006 \text{ s}^{-1}$	$1.82758 \text{ s}^{-1}$

the fraction of myosin heads in state C increases. Drug diffusion begins at 1500 s, signifying CCB intake. In this simulation, we assume an isotropic diffusion  $\mathbf{D} = D\mathbf{I}$  with the second-order identity tensor  $\mathbf{I}$  and a diffusion coefficient of  $D = 6 \cdot 10^{-5}$ . While in reality, an anisotropic diffusion (with higher diffusion along the fiber directions) is more realistic, an isotropic diffusion tensor is sufficient to show the functionality of the proposed material model. The Dirichlet boundary condition emulating drug inflow is implemented by switching the concentration from zero to one at  $t = 1500$  s. Starting from 2500 s, at which point there is mature but incomplete diffusion of drugs, 100 heartbeats are simulated. This part of the simulation shows the reaction of the material model on a changing intravascular pressure, which can be assumed in a real artery.

The material parameters for the passive material model, the chemical model, and the active chemo-mechanical material model are taken from Uhlmann and Balzani [63]. These material parameters were fitted by the authors to experimental results from Johnson et al. [36], who investigated the contraction of a middle cerebral rat artery by applying a sequence of intravascular pressure with increasing pressure values. The material parameters of the passive response are listed in Table 2, whereas the material parameters for the active part of the material model are given in Tables 3 and 4.

Figure 1 (right) shows the behavior of the artery with and without the influence of CCBs. In the simulation without medicinal effects, we see that the radius of the artery decreases with time and, in the end, is smaller than the radius of the initial unloaded artery. Additionally, it can be seen that the radius of the artery shows an ongoing decrease until



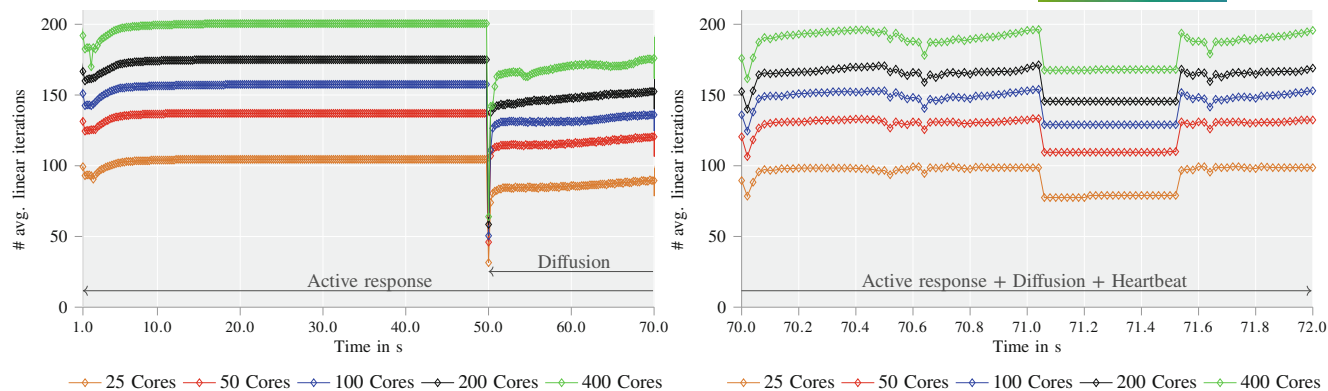
**FIGURE 3** Left and Right: Average number of linear iterations of loading protocol in Figure 2 and geometry in Figure 1 (left). Drug influence is switched off for one of the simulations. Computations on 50 processor cores with 50 000 degrees of freedom. Two-level overlapping Schwarz preconditioner with RGDSW coarse space and (SF) recycling. Before 1500 s, the iteration counts for the simulation with the influence of drugs are identical to the one without drugs. Right: Detailed view between 70 and 95 average number of iterations. The grid lines of the y axis mark rounded maxima and minima of the iteration count with respect to the sections [1500, 2500] and [2500, 2600]. We observe a range of approximately 20% (between 74 and 94) in iteration counts for simulations with and approximately 15% (between 79 and 94) without drug influence, ignoring outliers in the loading phase. The range of iteration counts increases in the heart-beat phase of the experiment, mainly due to the increased deformation caused by the softening of the material. With and without the influence of drugs, respectively, we observe 74 to 80 and 83 to 86 average number of linear iterations.

2500 s. This means that the steady state for all internal variables of the smooth muscle model is not reached at 2500 s yet. The change of the load to heartbeats leads to additional stretch of the smooth muscle cells, which results in a further contraction of the stretch-dependent material model. This behavior is consistent with the results of [63]. In the simulation with calcium channel-blocker influence, we observe, as expected, that the radius of the artery decreases at a much slower rate, owing to the lower  $\text{Ca}^{2+}$  availability. Additionally, we see that with increasing drug concentration, the arterial wall softens. This can be observed in Figure 1 (right) in the heartbeat phase of the simulation, where the magnitude of radius change between the systolic and diastolic pressure is higher in the simulation with drug influence. The vasodilatory action of CCBs is satisfactorily captured by the model.

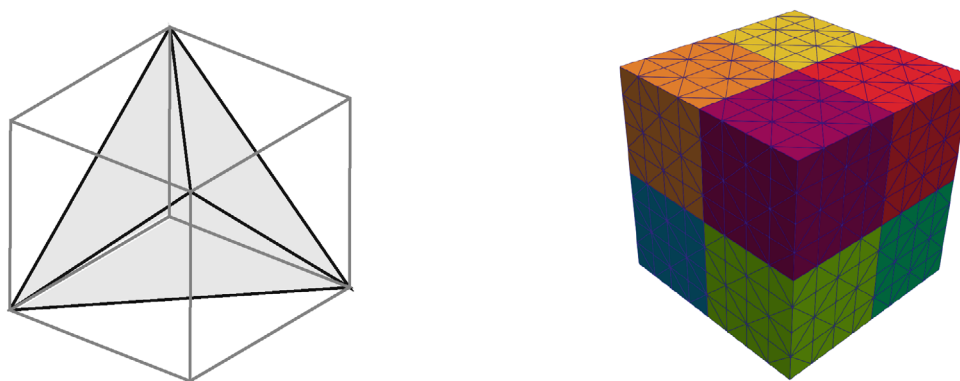
For further numerical tests and to optimize our solver, the average number of linear iterations per time step throughout the simulation are relevant and are shown in Figure 3. Except for higher iteration counts in the loading phase from 0 to 1 s and one outlier when the diffusion process starts, the iteration count is fairly constant and even decreases over time. Considering this tendency, it suffices to analyze not the whole experiment but an abridged one. We note that, when enabling the diffusion process at  $t = 1500$  s, we observe a significant reduction in the number of linear iterations and, at the same time, an extreme increase in the magnitude of the nonlinear residual  $\|F_k\|_2$  by more than 14 orders of magnitude; see also Figure 4. We presume that there is a relation between those two effects, which we plan to further investigate in future work.

## 6.2 | Weak parallel scalability

To analyze numerical and parallel scalability, we employ structured meshes for which the number of degrees of freedom per subdomain remains constant, but the number of subdomains increases. For perfect scalability, the number of iterations and time-to-solution should stay asymptotically constant with an increase in the number of subdomains and processor cores. A voxel mesh of the unit cube is first created and subsequently refined with 5 tetrahedra per voxel; see Figure 5 (left). The mesh of the cube is then partitioned into subcubes as depicted in Figure 5 (right). We choose  $5 \cdot 6^3$  tetrahedra per subdomain, which results in 7924 degrees of freedom per nonoverlapping subdomain. For the overlapping



**FIGURE 4** Iteration counts for different phases of loading as depicted in Figure 8. Geometry in Figure 1. RGDSW (reduced generalized Dryja–Smith–Widlund) coarse space with (SF) recycling strategy. Compare with the results in Figure 3, where a coarser mesh was used.



**FIGURE 5** Partition of a cube into five tetrahedra (left) and corresponding structured partition of the unit cube into eight subdomains, each containing  $5 \cdot 4^3$  tetrahedra (right). The orientation of a partition into five tetrahedra needs to be switched from one cube to a neighboring cube such that a checkerboard pattern is obtained. This pattern is visible on the right by looking at the finite element edges.

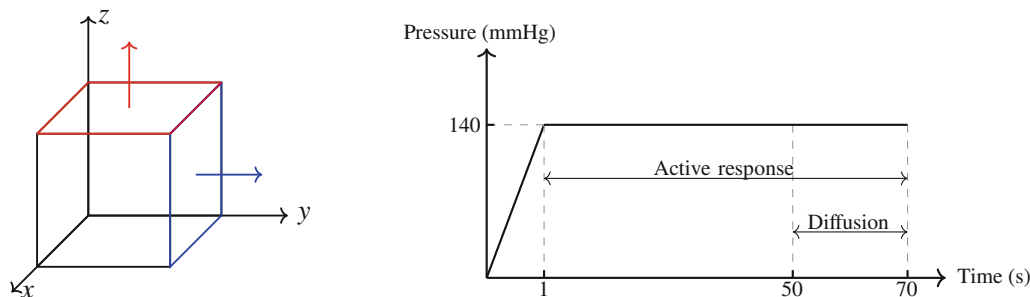
subdomains, we have a minimum of 11 884 and maximum of 167 516 degrees of freedom per subdomain. The number of subdomains or processor cores is increased from 216 to 1000 to 4096, which produces systems with approximately 1.3, 6.2, and 25.2 million degrees of freedom, respectively.

We consider a simplified setup of an axial pulling of the unit cube (Figure 6, left) with the presented material model from Section 2 and model settings from Tables 2 to 4. The unit cube has a side length of 1 mm. The boundary conditions for the displacement are prescribed as follows: At  $(0, 0, 0)$ , the cube is fixed in all directions. The three faces adjacent to  $(0, 0, 0)$  are fixed in their respective normal directions. The cube is then pulled at the top face, corresponding to the  $z = 1$  plane, in  $z$  direction and at the right face, corresponding to the  $y = 1$  plane, in  $y$  direction. For diffusion, a Dirichlet boundary condition emulating drug inflow is prescribed at the back face, corresponding to the  $x = 0$  plane.

For the weak scaling tests, we choose a relatively simple loading protocol as shown in Figure 6. From 0 to 1 s, a load is built up and then kept constant at 140 mmHg. In comparison to Section 6.1, we continue to use an isotropic diffusion tensor, but choose a larger diffusion coefficient of  $D = 6 \cdot 10^{-3}$  to increase the influence of the diffusion, which is diminished due to the shorter simulation time. Note that the different diffusion coefficients lead to only a negligible variation in the number of iterations.

For this setup, we will test different coarse spaces (cf. Section 4.1) and recycling strategies as defined in Section 4.3. Generally, we can expect that with more extensive reuse of coarse space information, the setup cost of the preconditioner will decrease while the linear iteration count and, thus, the runtime of the solver will increase. Therefore, we need to consider carefully which configuration yields the most time gain.

Using the GDSW coarse space, the method does not scale well (cf. Table 5): this is mainly a result of increasing setup times for an increasing number of subdomains since the GDSW coarse space is too large to be solved efficiently by a



**FIGURE 6** Setup for weak scaling tests on unit cube (side length: 1 mm). Left: Pressure load on  $z = 1$  plane and  $y = 1$  plane. Pulling in normal directions of the respective planes. Boundary conditions for the displacement: Fixed at  $(0, 0, 0)$  in all directions. The three faces adjacent to  $(0, 0, 0)$  are fixed in their respective normal directions. Right: Loading protocol for weak scaling with diffusion process and active response. Loading from 0 to 1 s with 20 load steps. Active response from 1 s until the end of 70 s with  $\Delta t = 0.25$  s. Diffusion starts at  $t = 50$  s.

**TABLE 5** Weak scaling results for two-level monolithic preconditioners with GDSW (generalized Dryja–Smith–Widlund) and RGDSW (reduced GDSW) coarse space for structure-chemical interaction system with overlap  $\hat{\delta} = 1$ .

# Cores	Coarse space	Recycling strategy	Avg. # Its.	Setup	Solve	Total
216	GDSW	(SF)	123.7 (2.1)	1138 s	1839 s	2997 s
	RGDSW	(SF)	152.5 (2.1)	789 s	1827 s	<b>2616 s</b>
1000	GDSW	(SF)	164.1 (2.1)	2414 s	6297 s	8711 s
	RGDSW	(SF)	184.7 (2.1)	1006 s	3449 s	<b>4435 s</b>

Notes: Simulation of 296 time steps. Loading protocol and geometry in Figure 6. The best total time is highlighted in boldface.

Abbreviations: Avg. # Its.: avg. # its. of linear solver (nonlinear solver); Setup: setup of preconditioner; Solve: linear solver time.

**TABLE 6** Coarse space dimension for the two-level monolithic overlapping Schwarz preconditioners with a GDSW and RGDSW coarse space for the structure–chemical interaction system and different number of cores. Geometry in Figure 6.

Coarse space	216	1000	4096
GDSW	7430	38 826	169 740
RGDSW	875	5103	23 625

direct solver. Notably, with 4096 subdomains, the computational time exceeds a 24-h limit at the Fritz supercomputer. A three-level variant may accelerate the solution; compare [27, 29]. The weak parallel scalability is significantly better for the RGDSW coarse space, since the smaller coarse space dimension (cf. Table 6) reduces the time, for instance, for the factorization of the coarse matrix and the application of the factorization.

Also for RGDSW, the parallel efficiency obtained for this complex nonlinear coupled problem is significantly below the efficiency routinely reported for domain decomposition applied to single physics problems. This is to some extent a result of an increase in GMRES iterations within the range of processor cores considered here. Specifically, when increasing the number of processor cores, we notice a significant increase in the orthogonalization time of GMRES: for instance, for the (SF)+(CM) combination of recycling strategies (cf. Table 7), we obtain 486, 1136 s, and 1 978 s for 216, 1 000, and 4 096 cores, respectively. The significant increase in total time cannot be exclusively explained by the increasing number of GMRES iterations. As we do not observe this effect in our strong scalability studies in Section 6.3, where the increase in GMRES iterations is even larger, we suspect that this effect also relates to increasing Tpetra communication times. However, the main part of the increase in the solve time can be attributed to the increasing dimension of the coarse space respectively the coarse matrix. Despite the significant reduction compared to the classical GDSW coarse space, the coarse space dimension for RGDSW still increases from 875 for 216 cores to 23 625 for 4096 cores. Whether numerical scalability is obtained for a larger number of subdomains and cores remains to be tested. A hotspot analysis using more detailed subtimers should be performed in the future.



**TABLE 7** Weak scaling results for two-level monolithic preconditioners with RGDSW (reduced generalized Dryja–Smith–Widlund) coarse space for structure-chemical interaction system with overlap  $\hat{\delta} = 1$ .

RGDSW coarse space					
# Cores	Recycling strategy	Avg. # Its.	Setup	Solve	Total
216	–	152.5 (2.1)	1207 s	1990 s	3197 s
	(SF)	152.5 (2.1)	788 s	1827 s	<b>2616 s</b>
	(SF)+(CB)	193.7 (2.1)	562 s	2362 s	2924 s
	(SF)+(CM)	183.8 (2.1)	706 s	2096 s	2802 s
1000	–	184.7 (2.1)	1841 s	4331 s	6172 s
	(SF)	184.7 (2.1)	1006 s	3449 s	<b>4435 s</b>
	(SF)+(CB)	239.0 (2.1)	794 s	5049 s	5843 s
	(SF)+(CM)	210.2 (2.1)	824 s	3976 s	4799 s
4096	–	219.9 (2.1)	2736 s	8844 s	11 580 s
	(SF)	219.9 (2.1)	1483 s	6934 s	<b>8417 s</b>
	(SF)+(CB)	286.5 (2.1)	1217 s	8827 s	10 044 s
	(SF)+(CM)	245.6 (2.1)	860 s	7807 s	8666 s

Notes: Simulation of 296 time steps. Loading protocol and geometry in Figure 6. The best total time is highlighted in boldface.

Abbreviations: Avg. # Its.: avg. # its. of linear solver (nonlinear solver); Setup: setup of preconditioner; Solve: linear solver time.

**TABLE 8** Results for different recycling strategies (cf. Section 4.3) and the two-level monolithic preconditioners with a GDSW (generalized Dryja–Smith–Widlund) coarse space for the structure-chemical interaction system with an overlap of  $\hat{\delta} = 1$ .

GDSW coarse space					
# Cores	Recycling strategy	Avg. # Its.	Setup	Solve	Total
216	–	123.7 (2.1)	1931 s	2001 s	3932 s
	(SF)	123.7 (2.1)	1138 s	1839 s	<b>2997 s</b>
	(SF)+(CB)	176.5 (2.1)	811 s	2594 s	3405 s
	(SF)+(CM)	201.8 (2.1)	805 s	3542 s	4347 s

Note: Simulation of 296 time steps. Loading protocol and geometry in Figure 6. The best total time is highlighted in boldface.

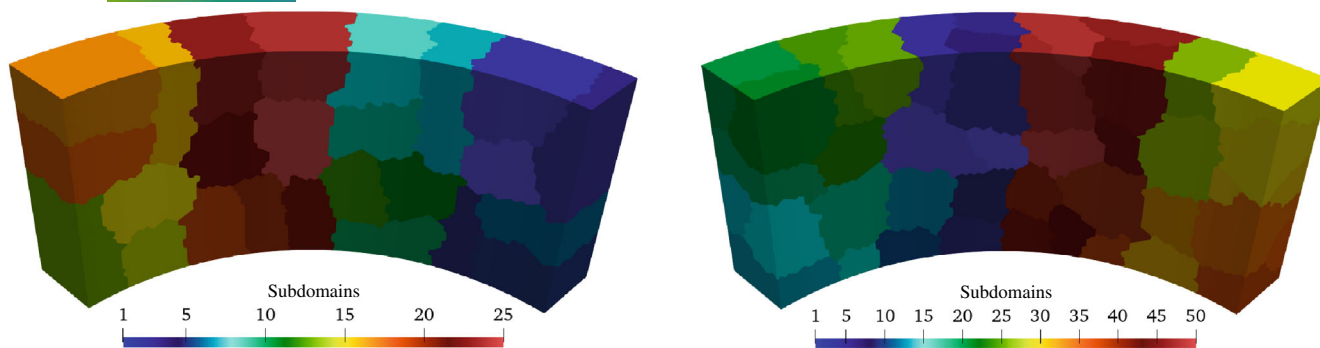
Abbreviations: Avg. # Its.: avg. # its. of linear solver (nonlinear solver); Setup: setup of preconditioner; Solve: linear solver time.

In Tables 8 and 7, we compare the different recycling strategies introduced in Section 4.3. According to these results, the (SF) strategy performs best. It decreases the setup time significantly by 35–45% compared to no use of recycling, and yet the average iteration counts do not increase. Note that, despite the equal number of iterations, the solve time using the (SF) recycling strategy is lower than for strategy using no recycling. We have planned to carry out an in-depth performance analysis to understand the source of this performance gain and other performance related aspects. The reuse of the coarse basis (CB) and the coarse matrix (CM) are much more invasive strategies. Nevertheless, in the considered examples, they outperform the no-reuse strategy for the RGDSW coarse space. For the GDSW coarse space, the (CM)+(SF) strategy performed worst with respect to iteration count and solver time. With the RGDSW coarse space, the (CB)+(SF) strategy delivered the worst results with respect to iteration counts and solver time. The results for the (CM)+(SF) strategy with the RGDSW coarse space are comparable to the (SF) strategy as it saves further time in the preconditioner setup, even though the iteration count increases.

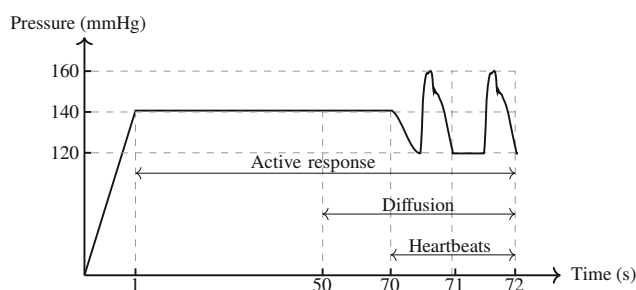
### 6.3 | Strong parallel scalability

For the analysis of strong scaling, we keep the problem size constant while increasing the number of processor cores from 25 to 400. We consider the arterial segment used in Section 6.1 (cf. Figures 1 and 7) and construct a mesh such that





**FIGURE 7** Unstructured partition of mesh with METIS into 25 and 50 subdomains. Geometry in Figure 1. With 25 subdomains (left), the partition is almost two-dimensional with respect to the interior arterial wall. With 50 subdomains (right), we observe a slight partition of the domain in the radial direction.



**FIGURE 8** Loading protocol for strong scaling tests with active response, diffusion process, and two heartbeats. Loading from  $t = 0$  s to  $t = 1$  s with 40 load steps. A constant pressure of 140 mmHg is applied until  $t = 70$  s with  $\Delta t = 0.25$  s. Starting at  $t = 70$  s two heartbeats are simulated with  $\Delta t = 0.02$  s. The time-dependent change of the loading during the heartbeats was generated based on the flow rate profile from [3], which is based on data from [56].

the coupled system (23) has 2.5 million degrees of freedom. The geometry is partitioned via METIS into unstructured subdomains; see Figure 7. We apply an abridged loading protocol, similar to the weak scaling analysis. Here, we additionally simulate two heartbeats; see Figure 8.

For the preconditioner, the RGDSW coarse space is employed, and we reuse the symbolic factorizations (SF) as described in Section 4.3, since this configuration provided the best results for weak scaling. Let us note that, for a relatively small number of processor cores, GDSW provides competitive results compared to RGDSW; see the results in Table 5 for 216 cores.

As before in Section 6.1 (cf. Figure 3), we observe a decline in the average number of linear iterations when the diffusion process starts; see also the discussion at the end of Section 6.1.

Results for GDSW and RGDSW are given in Table 9; the results for RGDSW are also plotted in Figure 9 (right). For RGDSW, we initially see an efficiency of over 90%, while it decreases considerably for a larger number of processors. Overall, the performance of the RDGSW coarse space is relatively robust. Especially the setup of the preconditioner scales well with an increasing number of subdomains. For GDSW, in Table 9 we clearly see the scalability limit stemming from the size of the coarse space when scaling from 200 to 400 cores. For GDSW, the coarse problem has 40 905 d.o.f., which is more than four times larger than the coarse space of RGDSW. For GDSW, we would need to make use of an additional level (see, e.g., [27, 29]) to obtain better results. Similarly, RGDSW may profit from an additional level, even for the moderate numbers of cores considered here.

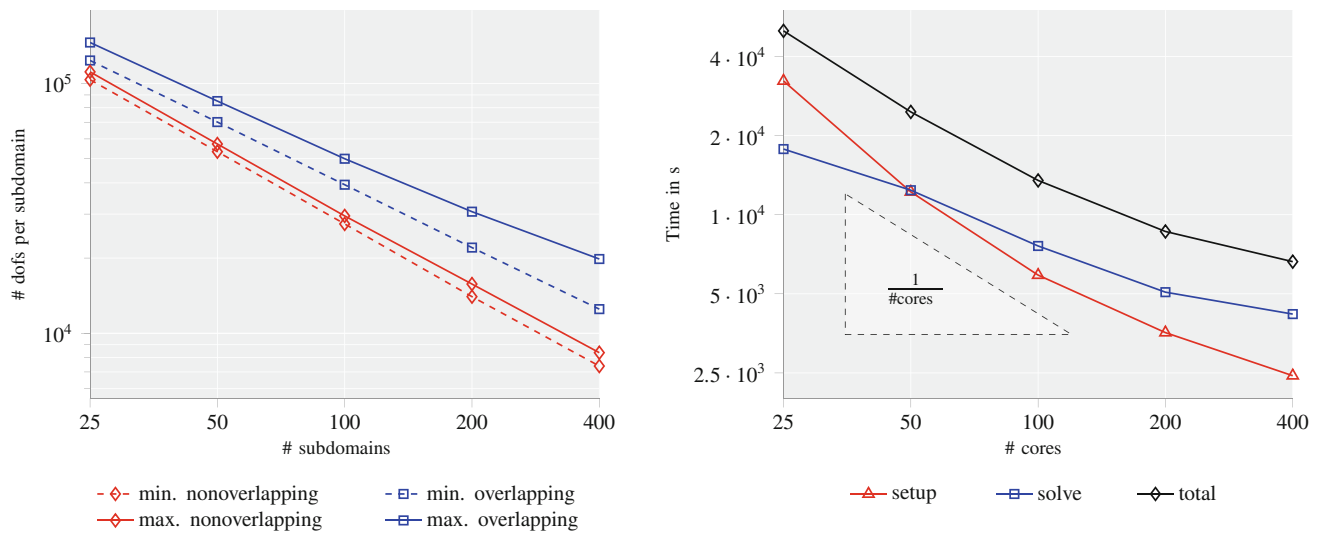
Before we discuss several factors that influence the performance, we consider the timings for GDSW. As the coarse space grows more rapidly with the number of subdomains than the RGDSW coarse space, we expect a lower efficiency. Indeed, the efficiency deteriorates much more quickly. Despite fewer iterations than using the RGDSW coarse space for 400 subdomains, the time taken by the linear solver is 10 834 s larger; we conclude that the coarse solve takes up a large portion of the linear solver time. In total, 13 590 s are spent in 186 200 applications of the preconditioner.

**TABLE 9** Strong scaling results for two-level monolithic overlapping Schwarz preconditioners with RGDSW (reduced generalized Dryja–Smith–Widlund) coarse space and overlap  $\hat{\delta} = 1$ .

# Cores	Dim. coarse space	Avg. # Its.	Setup	Solve	Total
<b>RGDSW coarse space</b>					
25	203	97.9 (2.4)	32 270 s	17 770 s	50 040 s
50	602	131.8 (2.4)	12 210 s	12 390 s	24 600 s
100	1645	151.6 (2.4)	5890 s	7603 s	13 498 s
200	3927	169.1 (2.4)	3559 s	5072 s	8631 s
400	9450	193.3 (2.4)	2440 s	4186 s	6626 s
<b>GDSW coarse space</b>					
25	613	79.6 (2.4)	35 570 s	14 850 s	50 420 s
50	3153	119.8 (2.4)	20 550 s	12 590 s	33 140 s
100	8096	143.5 (2.4)	12 050 s	8918 s	20 968 s
200	18 134	159.9 (2.4)	9650 s	7406 s	17 056 s
400	40 905	185.5 (2.4)	11 370 s	15 020 s	26 390 s

Notes: Average linear iteration count over all time steps with nonlinear solver iterations in parenthesis. Unstructured mesh partition with METIS. Simulation of 416 time steps. Loading protocol in Figure 8, geometry in Figure 1.

Abbreviations: Avg. # Its.: avg. # its. of linear solver (nonlinear solver); Setup: setup of preconditioner; Solve: linear solver time.



**FIGURE 9** Degrees of freedom per subdomain versus number of subdomains (left). Minimum and maximum number of degrees of freedom per nonoverlapping and overlapping subdomain, respectively. Unstructured mesh partition with METIS; compare Figure 7. Strong scaling plot for RGDSW with time in seconds versus number of cores for setup, solve, and total time (right); compare Table 9. The black dashed triangle shows the slope for ideal scalability.

There are several properties that influence the performance of the two-level additive Schwarz method. For example, if the size of local problems decreases, generally, the number of iterations increases, which is also evident from Table 9. A larger number of iterations not only requires more applications of the system matrix in GMRES but also of the preconditioner. Moreover, the complexity of the orthogonalization routine within GMRES increases quadratically with the number of iterations. Similarly, since the coarse space size increases with the number of subdomains (cf. Table 6), the (at worst) cubic complexity of a direct solver results in significantly longer runtimes, which can, to some degree, be mitigated by using a parallel sparse direct solver. On the other hand, the subdomain problems decrease in size and, thus, profit from

the superlinear complexity of direct solvers. Furthermore, there are hardware-related factors like cache effects, which we will not elaborate here.

Instead, we discuss how the mesh partition can influence the performance. The partition of the mesh into 25 subdomains is almost two-dimensional in the sense that there is usually only a single subdomain in the radial direction; see Figure 7 (left). This effect is diminished with a partition into 50 subdomains (see Figure 7 (right)); it significantly increases coupling between subdomains for the largest test case of 400 subdomains. The additional coupling has an influence on the performance and weakens strong scalability.

For good load balancing, subdomains should ideally have the same number of nodes. Otherwise, results for strong scaling may be impacted by idle cores. As we can see in Figure 9 (left), the METIS-computed mesh partition for the nonoverlapping subdomains differs only slightly with respect to the minimum (red dashed line) and maximum (red solid line) subdomain size. However, locally, we do not solve nonoverlapping but overlapping subdomain problems. Figure 9 (left) shows that the minimum (blue dashed line) and maximum (blue solid line) overlapping subdomain sizes diverge with an increasing number of subdomains. A reason may be the shape of the subdomains. Even if the nonoverlapping subdomains are identical in size, a different shape can significantly influence the size of the corresponding overlapping subdomain. For example, the ratio of nodes in the overlapping to the nodes in the nonoverlapping subdomain is large for a thin beam and small for a cube or sphere.

This is not the only source that can lead to a deterioration in performance. With an increasing number of subdomains (for a fixed mesh), the total number of nodes in overlapping subdomains increases as well. As a result, we cannot obtain perfect scalability. We consider, for example, a cube as a nonoverlapping subdomain with  $10 \times 10 \times 10$  nodes. If we prescribe an overlap of one layer of additional nodes, we obtain  $12 \times 12 \times 12 = 1728$  nodes. By decomposing the cube into  $2 \times 2 \times 2$  subcubes, we obtain overlapping subdomains of size  $7 \times 7 \times 7 = 343$ , in total  $8 \times 343 = 2744$  nodes, an increase of 58.8% with respect to the initial overlapping cube. Consequently—disregarding other factors—we cannot obtain perfect strong scalability.

A further reduction of the computing time could be obtained by a temporal homogenization of the diffusion process, which can also be combined with a parallel-in-time integration; compare [18]. We might consider this in future investigations.

## 7 | CONCLUSIONS

Motivated by a need to better understand pharmaco-mechanical interactions in arteries, we have developed a computational framework to describe the effect of CCBs on the mechanical response. We adopted the material model of the arterial wall from Uhlmann and Balzani [63] and extended it to include pharmacological effects. The transmural drug transport was modeled using the reaction-diffusion equation and the resulting coupled system of equations was discretized using the finite element method. The resulting nonlinear finite element system is solved with a Newton method combined with a load stepping strategy where the tangent problems are solved using a parallel monolithic domain decomposition approach. Numerical results analyzing strong and weak parallel scalability for this strategy are presented showing the simulation capability of this approach. The pharmaco-mechanical model was implemented in AceGen and then transferred to FEDDLib using a newly developed interface. The numerical results show that this approach works well. For the scenarios considered here, scalability limits are already visible, especially for the GDSW coarse space. A detailed performance analysis needs to be performed as a next step. The size of the coarse space indicates that we will need to move to a three-level method. Simulation results for an arterial section under physiological loading conditions show that the proposed model is able to qualitatively capture the vasodilatory effects of CCBs. An accurate simulation of patient-specific arteries and stress distributions would require additional considerations such as residual stresses, advection-based drug transport and realistic fiber distributions. Residual stresses can be included in our model using methods such as the open angle method or anisotropic growth models, whereas realistic fiber orientations may be obtained using arterial remodeling models. However, since these factors do not play a pivotal role in the pharmaco-mechanical interaction, their effects were not considered in this work and may be included in future studies.

## AUTHOR CONTRIBUTIONS

The order of the list of authors of this paper is alphabetical. Their contributions are as follows: **Daniel Balzani**: Conceptualization, supervision, methodology, writing, review; **Alexander Heinlein**: Methodology, writing, programming,

review; **Axel Klawonn**: Conceptualization, supervision, methodology, writing, review; **Jascha Knepper**: Methodology, writing, review; **Sharan Nurani Ramesh**: Methodology, writing, programming; **Oliver Rheinbach**: Conceptualization, supervision, methodology, writing, review; **Lea Saßmannshausen**: Methodology, writing, programming; **Klemens Uhlmann**: Writing, model formulation.









## ACKNOWLEDGMENTS

Financial funding from the Deutsche Forschungsgemeinschaft (DFG) through the Priority Program 2311 “Robust coupling of continuum-biomechanical in silico models to establish active biological system models for later use in clinical applications—Co-design of modeling, numerics and usability,” project ID 465228106, is greatly appreciated. The authors gratefully acknowledge the scientific support and HPC resources provided by the Erlangen National High Performance Computing Center (NHR@FAU) of the Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) under the NHR project k105be. NHR funding is provided by federal and Bavarian state authorities. NHR@FAU hardware is partially funded by the German Research Foundation (DFG)—440719683. Open Access funding enabled and organized by Projekt DEAL.

## CONFLICT OF INTEREST STATEMENT

The authors declare no potential conflict of interests.

## ORCID

**Daniel Balzani**  <https://orcid.org/0000-0002-1422-4262>  
**Alexander Heinlein**  <https://orcid.org/0000-0003-1578-8104>  
**Axel Klawonn**  <https://orcid.org/0000-0003-4765-7387>  
**Jascha Knepper**  <https://orcid.org/0000-0002-8769-2235>  
**Sharan Nurani Ramesh**  <https://orcid.org/0000-0001-5586-9924>  
**Oliver Rheinbach**  <https://orcid.org/0000-0002-9310-8533>  
**Lea Saßmannshausen**  <https://orcid.org/0009-0005-9538-5461>  
**Klemens Uhlmann**  <https://orcid.org/0000-0002-5340-6800>

## REFERENCES

- [1] L. Ai and K. Vafai, A coupling model for macromolecule transport in a stenosed arterial wall, *Int. J. Heat Mass Transf.* **49** (2006), no. 9-10, 1568–1591.
- [2] D. Balzani, D. Böse, D. Brands, R. Erbel, A. Klawonn, O. Rheinbach, and J. Schröder, Parallel simulation of patient-specific atherosclerotic arteries for the enhancement of intravascular ultrasound diagnostics, *Eng. Comput.* **29** (2012), no. 8, 888–906.
- [3] D. Balzani, S. DeParis, S. Fausten, D. Forti, A. Heinlein, A. Klawonn, A. Quarteroni, O. Rheinbach, and J. Schröder, Numerical modeling of fluid-structure interaction in arteries with anisotropic polyconvex hyperelastic and anisotropic viscoelastic material models at finite strains, *Int. J. Numer. Meth. Biomed. Eng.* **32** (2015), no. 10, e02756.
- [4] D. Balzani, P. Neff, J. Schröder, and G. A. Holzapfel, A polyconvex framework for soft biological tissues. Adjustment to experimental data, *Int. J. Solids Struct.* **43** (2006), no. 20, 6052–6070.
- [5] R. A. Bartlett. *Stratimikos wrapper package*, STRATIMIKOS; 001978MLTPL00, Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA. 2006 <https://www.osti.gov/biblio/1245170>.
- [6] R. A. Bartlett. *Thyra linear operators and vectors: overview of interfaces and support software for the development and interoperability of abstract numerical algorithms*, SAND2007-5984. 2007.
- [7] E. Bavier, M. Hoemmen, S. Rajamanickam, and H. Thornquist, Amesos2 and Belos: Direct and iterative solvers for large sparse linear systems, *Sci. Program.* **20** (2012), no. 3, 241–255.
- [8] L. Berger-Vergiat, C. A. Glusa, J. J. Hu, M. Mayr, A. Prokopenko, C. M. Siefert, R. S. Tuminaro, and T. A. Wiesner. *MueLu User's Guide*, SAND2019-0537, Sandia National Laboratories. 2019.
- [9] M. Böl, A. Schmitz, G. Nowak, and T. Siebert, A three-dimensional chemo-mechanical continuum model for smooth muscle contraction, *J. Mech. Behav. Biomed. Mater.* **13** (2012), 215–229.
- [10] P. Causin, J. F. Gerbeau, and F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid–structure problems, *Comput. Methods Appl. Mech. Eng.* **194** (2005), no. 42, 4506–4527.
- [11] C. J. Creel, M. A. Lovich, and E. R. Edelman, Arterial paclitaxel distribution and deposition, *Circ. Res.* **86** (2000), no. 8, 879–884.
- [12] E. C. Cyr, J. N. Shadid, and R. S. Tuminaro, Teko: A block preconditioning capability with concrete example applications in Navier-stokes and MHD, *SIAM J. Sci. Comput.* **38** (2016), no. 5, S307–S331.
- [13] C. R. Dohrmann, A. Klawonn, and O. B. Widlund, Domain decomposition for less regular subdomains: overlapping Schwarz in two dimensions, *SIAM J. Numer. Anal.* **46** (2008), no. 4, 2153–2168.

- [14] C. R. Dohrmann, A. Klawonn, and O. B. Widlund, “A family of energy minimizing coarse spaces for overlapping Schwarz preconditioners,” *Domain Decomposition Methods in Science and Engineering XVII, Lecture Notes on Computer Science Engineering*, Vol **60**, Springer, Berlin, 2008, pp. 247–254. [https://doi.org/10.1007/978-3-540-75199-1\\_28](https://doi.org/10.1007/978-3-540-75199-1_28).
- [15] C. R. Dohrmann and O. B. Widlund, On the design of small coarse spaces for domain decomposition algorithms, *SIAM J. Sci. Comput.* **39** (2017), no. 4, A1466–A1488.
- [16] H. C. Edwards, C. R. Trott, and D. Sunderland, Kokkos: Enabling manycore performance portability through polymorphic memory access patterns, *J. Parallel Distrib. Comput.* **74** (2014), no. 12, 3202–3216.
- [17] M. H. Forouzanfar, P. Liu, G. A. Roth, M. Ng, S. Biryukov, L. Marczak, L. Alexander, K. Estep, K. Hassen Abate, T. F. Akinyemiju, R. Ali, N. Alvis-Guzman, P. Azzopardi, A. Banerjee, T. Bärnighausen, A. Basu, T. Bekele, D. A. Bennett, S. Biadgilign, F. Catalá-López, V. L. Feigin, J. C. Fernandes, F. Fischer, A. A. Gebru, P. Gona, R. Gupta, G. J. Hankey, J. B. Jonas, S. E. Judd, Y. H. Khang, A. Khosravi, Y. J. Kim, R. W. Kimokoti, Y. Kokubo, D. Kolte, A. Lopez, P. A. Lotufo, R. Malekzadeh, Y. A. Melaku, G. A. Mensah, A. Misganaw, A. H. Mokdad, A. E. Moran, H. Nawaz, B. Neal, F. N. Ngalesoni, T. Ohkubo, F. Pourmalek, A. Rafay, R. K. Rai, D. Rojas-Rueda, U. K. Sampson, I. S. Santos, M. Sawhney, A. E. Schutte, S. G. Sepanlou, G. T. Shifa, I. Shiue, B. A. Tedla, A. G. Thrift, M. Tonelli, T. Truelsen, N. Tsilimparis, K. N. Ukwaja, O. A. Uthman, T. Vasankari, N. Venketasubramanian, V. V. Vlassov, T. Vos, R. Westerman, L. L. Yan, Y. Yano, N. Yonemoto, M. E. S. Zaki, and C. J. L. Murray, Global burden of hypertension and systolic blood pressure of at least 110 to 115 mm hg, 1990-2015, *Jama* **317** (2017), no. 2, 165–182.
- [18] S. Frei and A. Heinlein. *Towards parallel time-stepping for the numerical simulation of atherosclerotic plaque growth*, ArXiv:2203.06526 [math.NA]. 2023 <https://doi.org/10.48550/arXiv.2203.06526>.
- [19] C. Geuzaine and J.-F. Remacle, Gmsh: A 3-D finite element mesh generator with built-in pre- and post-processing facilities, *Int. J. Numer. Methods Eng.* **79** (2009), no. 11, 1309–1331.
- [20] C.-M. Hai and R. A. Murphy, Cross-bridge phosphorylation and regulation of latch state in smooth muscle, *Am. J. Phys. Cell Phys.* **254** (1988), no. 1, C99–C106.
- [21] A. Heinlein. *Parallel overlapping Schwarz preconditioners and multiscale discretizations with applications to fluid-structure interaction and highly heterogeneous problems*, Ph.D. thesis, Universität zu Köln. 2016.
- [22] A. Heinlein, C. Hochmuth, and A. Klawonn, Monolithic overlapping Schwarz domain decomposition methods with GDSW coarse spaces for incompressible fluid flow problems, *SIAM J. Sci. Comput.* **41** (2019), no. 4, C291–C316.
- [23] A. Heinlein, C. Hochmuth, and A. Klawonn, Reduced dimension GDSW coarse spaces for monolithic Schwarz domain decomposition methods for incompressible fluid flow problems, *Int. J. Numer. Methods Eng.* **121** (2020), no. 6, 1101–1119.
- [24] A. Heinlein, C. Hochmuth, and A. Klawonn, “Fully algebraic two-level overlapping Schwarz preconditioners for elasticity problems,” F. J. Vermolen, and C. Vuik (eds.) *Numerical Mathematics and Advanced Applications ENUMATH 2019, Lecture Notes on Computer Science Engineering*, Vol **139**, Springer, Cham, 2019, pp. 531–539. [https://doi.org/10.1007/978-3-030-55874-1\\_52](https://doi.org/10.1007/978-3-030-55874-1_52).
- [25] A. Heinlein, A. Klawonn, S. Rajamanickam, and O. Rheinbach, “FROSch: a fast and robust overlapping Schwarz domain decomposition preconditioner based on Xpetra in Trilinos,” R. Haynes, S. MacLachlan, X.-C. Cai, L. Halpern, H. H. Kim, A. Klawonn, and O. Widlund, (eds.) *Domain Decomposition Methods in Science and Engineering XXV*, Springer International Publishing, Cham, 2020, pp. 176–184.
- [26] A. Heinlein, A. Klawonn, and O. Rheinbach, A parallel implementation of a two-level overlapping Schwarz method with energy-minimizing coarse space based on Trilinos, *SIAM J. Sci. Comput.* **38** (2016), no. 6, C713–C747.
- [27] A. Heinlein, A. Klawonn, O. Rheinbach, and F. Röver, “A three-level extension of the GDSW overlapping Schwarz preconditioner in three dimensions,” R. Haynes, S. MacLachlan, X.-C. Cai, L. Halpern, H. H. Kim, A. Klawonn, and O. Widlund, (eds.) *Domain Decomposition Methods in Science and Engineering XXV, Lecture Notes in Computer Science Engineering*, Vol **138**, Springer, Cham, 2017, pp. 185–192. [https://doi.org/10.1007/978-3-030-56750-7\\_20](https://doi.org/10.1007/978-3-030-56750-7_20). DD2018.
- [28] A. Heinlein, M. Perego, and S. Rajamanickam, FROSch preconditioners for land ice simulations of Greenland and Antarctica, *SIAM J. Sci. Comput.* **44** (2022), no. 2, B339–B367.
- [29] A. Heinlein, O. Rheinbach, and F. Röver, Parallel scalability of three-level FROSch preconditioners to 220000 cores using the theta supercomputer, *SIAM J. Sci. Comput.* **45** (2023), no. 3, S173–S198 Special Section: 2021 Copper Mountain Conference.
- [30] M. A. Heroux, R. A. Bartlett, V. E. Howle, R. J. Hoekstra, J. J. Hu, T. G. Kolda, R. B. Lehoucq, K. R. Long, R. P. Pawlowski, E. T. Phipps, A. G. Salinger, H. K. Thornquist, R. S. Tuminaro, J. M. Willenbring, A. Williams, and K. S. Stanley, An overview of the Trilinos project, *Trans. Math. Softw.* **31** (2005), no. 3, 397–423.
- [31] C. Hochmuth. *Parallel overlapping Schwarz preconditioners for incompressible fluid flow and fluid-structure interaction problems*, Ph.D. thesis, Universität zu Köln. 2020 <https://kups.ub.uni-koeln.de/11345/>.
- [32] S. S. Hossain, S. F. Hossainy, Y. Bazilevs, V. M. Calo, and T. J. Hughes, Mathematical modeling of coupled drug and drug-encapsulated nanoparticle transport in patient-specific coronary artery walls, *Comput. Mech.* **49** (2012), 213–242.
- [33] G. Howard, M. Banach, M. Cushman, D. C. Goff, V. J. Howard, D. T. Lackland, J. McVay, J. F. Meschia, P. Muntner, S. Oparil, M. Rightmyer, and H. A. Taylor, Is blood pressure control for stroke prevention the correct goal? The lost opportunity of preventing hypertension, *Stroke* **46** (2015), no. 6, 1595–1600.
- [34] C.-W. Hwang, D. Wu, and E. R. Edelman, Physiological transport forces govern drug distribution for stent-based delivery, *Circulation* **104** (2001), no. 5, 600–605.
- [35] C.-W. Hwang, D. Wu, and E. R. Edelman, Impact of transport and drug properties on the local pharmacology of drug-eluting stents, *Int. J. Cardiovasc. Interv.* **5** (2003), no. 1, 7–12.



- [36] R. P. Johnson, A. F. El-Yazbi, K. Takeya, E. J. Walsh, M. P. Walsh, and W. C. Cole, Ca<sup>2+</sup> sensitization via phosphorylation of myosin phosphatase targeting subunit at threonine-855 by rho kinase contributes to the arterial myogenic response, *J. Physiol.* **587** (2009), no. 11, 2537–2553.
- [37] G. Karypis, K. Schloegel, and V. Kumar, *ParMETIS: Parallel graph partitioning and sparse matrix ordering library*, University of Minnesota, Minneapolis, MN, 1997.
- [38] A. Klawonn and L. F. Pavarino, Overlapping Schwarz methods for mixed linear elasticity and Stokes problems, *Comput. Methods Appl. Mech. Eng.* **165** (1998), no. 1-4, 233–245.
- [39] A. Klawonn and L. F. Pavarino, A comparison of overlapping Schwarz methods and block preconditioners for saddle point problems, *Numer. Linear Algebra Appl.* **7** (2000), no. 1, 1–25.
- [40] J. Korelc and P. Wriggers, *Automation of finite element methods*, Springer, Cham, 2016.
- [41] X. S. Li and J. W. Demmel, SuperLU\_DIST: a scalable distributed-memory sparse direct solver for unsymmetric linear systems, *ACM Trans. Math. Softw.* **29** (2003), no. 2, 110–140.
- [42] K. T. Mills, J. D. Bundy, T. N. Kelly, J. E. Reed, P. M. Kearney, K. Reynolds, J. Chen, and J. He, Global disparities of hypertension prevalence and control: a systematic analysis of population-based studies from 90 countries, *Circulation* **134** (2016), no. 6, 441–450.
- [43] S. C. Murtada, A. Arner, and G. A. Holzapfel, Experiments and mechanochemical modeling of smooth muscle contraction: significance of filament overlap, *J. Theor. Biol.* **297** (2012), 176–186.
- [44] S.-I. Murtada, M. Kroon, and G. A. Holzapfel, A calcium-driven mechanochemical model for prediction of force generation in smooth muscle, *Biomech. Model. Mechanobiol.* **9** (2010), no. 6, 749–762.
- [45] S. Nurani Ramesh, K. Uhlmann, L. Saßmannshausen, O. Rheinbach, A. Klawonn, A. Heinlein, and D. Balzani, First steps towards modeling the interaction of cardiovascular agents and smooth muscle activation in arterial walls, *Proc. Appl. Math. Mech.* **22** (2023), no. 1, e202200133.
- [46] B. M. O’Connell, T. M. McGloughlin, and M. T. Walsh, Factors that affect mass transport from drug eluting stents into the artery wall, *Biomed. Eng. Online* **9** (2010), no. 1, 1–16.
- [47] S. Oparil, M. C. Acelayado, G. L. Bakris, D. R. Berlowitz, R. Cifková, A. F. Dominiczak, G. Grassi, J. Jordan, N. R. Poulter, A. Rodgers, and P. K. Whelton, Hypertension, *Nature Rev. Dis. Primers* **4** (2018), no. 1, 1–21.
- [48] S. Rajamanickam, S. Acer, L. Berger-Vergiat, V. Dang, N. Ellingwood, E. Harvey, B. Kelley, C. R. Trott, J. Wilke, I. Yamazaki, and K. Kernels. Performance portable sparse/dense linear algebra and graph kernels, *arXiv:2103.11991 [cs]*. 2021. 10.48550/arXiv.
- [49] Y. Saad and M. H. Schultz, GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems, *J. Sci. Stat. Comput.* **7** (1986), no. 3, 856–869.
- [50] L. Saßmannshausen. *Adaptive Finite Elemente in 2D und 3D — Fehlerschätzer, Verfeinerung und parallele Implementierung*, Master’s thesis, Universität zu Köln, 2021. English title: Adaptive finite elements in 2D and 3D — Error estimators, refinement, and parallel implementation.
- [51] O. Schenk and K. Gärtner, Solving unsymmetric sparse systems of linear equations with PARDISO, *Futur. Gener. Comput. Syst.* **20** (2004), no. 3, 475–487.
- [52] B. F. Smith, P. E. Bjørstad, and W. D. Gropp, *Domain decomposition*, Cambridge University Press, Cambridge, 1996.
- [53] A. P. Somlyo and A. V. Somlyo, Ca<sup>2+</sup> sensitivity of smooth muscle and nonmuscle myosin II: Modulated by G proteins, kinases, and myosin phosphatase, *Physiol. Rev.* **83** (2003), no. 4, 1325–1358.
- [54] J. Stålhand, A. Klarbring, and G. A. Holzapfel, A mechanochemical 3D continuum model for smooth muscle contraction under finite strains, *J. Theor. Biol.* **268** (2011), no. 1, 120–130.
- [55] The FEDDLib team, *FEDDLib (Finite Element and Domain Decomposition Library)*, GitHub Repository. 2023 <https://github.com/FEDDLib/FEDDLib>.
- [56] The HeMoLab team, *HeMoLab*. 2014 <http://hemolab.lncc.br/adan-web>.
- [57] The Trilinos Project Team, Trilinos, GitHub Repository. 2023 <https://github.com/trilinos/Trilinos>.
- [58] The Trilinos Project Team. *The Trilinos project website*. 2023 <https://trilinos.github.io>.
- [59] A. Toselli and O. Widlund, *Domain decomposition methods—Algorithms and theory*, Springer series in computational mathematics, Vol **34**, Springer-Verlag, Berlin, 2005. <https://doi.org/10.1007/b137868>.
- [60] R. M. Touyz and C. Delles, *Textbook of vascular medicine*, Springer, Cham, 2019.
- [61] C. Trott, L. Berger-Vergiat, D. Poliakoff, S. Rajamanickam, D. Lebrun-Grandie, J. Madsen, N. Al Awar, M. Gligoric, G. Shipman, and G. Womeldorff, The Kokkos ecosystem: comprehensive performance portability for high performance computing, *Comput. Sci. Eng.* **23** (2021), no. 5, 10–18.
- [62] C. R. Trott, D. Lebrun-Grandié, D. Arndt, J. Ciesko, V. Dang, N. Ellingwood, R. Gayatri, E. Harvey, D. S. Hollman, D. Ibanez, N. Liber, J. Madsen, J. Miles, D. Poliakoff, A. Powell, S. Rajamanickam, M. Simberg, D. Sunderland, B. Turcksin, and J. Wilke, Kokkos 3: Programming model extensions for the exascale era, *IEEE Trans. Parallel Distrib. Syst.* **33** (2022), no. 4, 805–817.
- [63] K. Uhlmann and D. Balzani, Chemo-mechanical modeling of smooth muscle cell activation for the simulation of arterial walls under changing blood pressure, *Biomech. Model. Mechanobiol.* **22** (2023), no. 3, 1049–1065.
- [64] K. Uhlmann, A. Zahn, and D. Balzani, “Simulation of arterial walls: Growth, fiber reorientation, and active response,” *Solid (Bio)mechanics: Challenges of the Next Decade*, Springer, Cham, 2022, pp. 181–209.
- [65] R. C. Webb, Smooth muscle contraction and relaxation, *Adv. Physiol. Educ.* **27** (2003), no. 4, 201–206.
- [66] P. Wriggers, *Nonlinear finite element methods*, Springer Science & Business Media, Berlin, Heidelberg, 2008.



- [67] I. Yamazaki, A. Heinlein, and S. Rajamanickam. *An experimental study of two-level Schwarz domain decomposition preconditioners on GPUs*, arXiv:2304.04876 [cs, math]. 2023 <https://doi.org/10.48550/arXiv.2304.04876>.
- [68] J. Yang, J. W. Clark Jr., R. M. Bryan, and C. Robertson, The myogenic response in isolated rat cerebrovascular arteries: smooth muscle cell model, *Med. Eng. Phys.* **25** (2003), no. 8, 691–709.
- [69] J. Yang, J. W. Clark Jr., R. M. Bryan, and C. S. Robertson, The myogenic response in isolated rat cerebrovascular arteries: vessel model, *Med. Eng. Phys.* **25** (2003), no. 8, 711–717.

**How to cite this article:** D. Balzani, A. Heinlein, A. Klawonn, J. Knepper, S. Nurani Ramesh, O. Rheinbach, L. Saßmannshausen, and K. Uhlmann, *A computational framework for pharmaco-mechanical interactions in arterial walls using parallel monolithic domain decomposition methods*, *GAMM-Mitteilungen*. **47** (2024), e202370002. <https://doi.org/10.1002/gamm.202370002>