Journal of Computational Physics 229 (2010) 642–659

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Scalable parallel methods for monolithic coupling in fluid–structure interaction with application to blood flow modeling ☆

Andrew T. Barker [a,*], Xiao-Chuan Cai [b]

[a] Department of Applied Mathematics, University of Colorado, 526 UCB, Boulder, CO 80309-0526, United States
[b] Department of Computer Science, University of Colorado, 430 UCB, Boulder, CO 80309-0430, United States

ARTICLE INFO

Article history:
Received 21 November 2008
Received in revised form 24 September 2009
Accepted 3 October 2009
Available online 14 October 2009

Keywords:
Fluid–structure interaction
Blood flow
Mesh movement
Restricted additive Schwarz
Domain decomposition
Parallel computing

ABSTRACT

We introduce and study numerically a scalable parallel finite element solver for the simulation of blood flow in compliant arteries. The incompressible Navier–Stokes equations are used to model the fluid and coupled to an incompressible linear elastic model for the blood vessel walls. Our method features an unstructured dynamic mesh capable of modeling complicated geometries, an arbitrary Lagrangian–Eulerian framework that allows for large displacements of the moving fluid domain, monolithic coupling between the fluid and structure equations, and fully implicit time discretization. Simulations based on blood vessel geometries derived from patient-specific clinical data are performed on large supercomputers using scalable Newton–Krylov algorithms preconditioned with an overlapping restricted additive Schwarz method that preconditions the entire fluid–structure system together. The algorithm is shown to be robust and scalable for a variety of physical parameters, scaling to hundreds of processors and millions of unknowns.

1. Introduction

In this work, we develop a scalable parallel framework for simulating fluid–structure interaction. Fluid–structure interaction is an important concern in many applications, including aircraft design and civil engineering, but the application area we focus on here is the simulation of blood flow in human arteries.

Artery disease is the leading cause of death in developed nations, and by some accounts is the leading cause of death worldwide [35]. Artery disease is closely associated with flow properties of the blood and with the interaction between the blood and the vessel wall. In particular, areas of turbulence, flow recirculation, or places where the artery wall is subject to low or oscillating shear stress are at higher risk for plaque formation and disease. Accurate modeling of these flow characteristics might enable better prediction of when and where artery disease will occur and lead to more accurate, less invasive, and more timely treatment [43].

Unfortunately, the blood flow problem is difficult from both a modeling and a computational perspective. Good overviews of the problem can be found in [17,43]. The main challenge we are concerned with here is the effective coupling of the fluid and the structure in a stable and scalable manner. The fluid–structure coupling in arteries is strong, the structure and the fluid domain deform significantly, and the coupling is essential to describing the behavior of the system, which makes the blood flow problem a good application for our method.

☆ This research was supported in part by DOE under DE-FC02-04ER25595, and in part by NSF under Grants CCF-0634894, and CNS-0722023, and DMS-0913089.

* Corresponding author. Tel.: +1 720 255 9191.
E-mail addresses: andrew.barker@colorado.edu (A.T. Barker), cai@cs.colorado.edu (X.-C. Cai).

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.10.001

Because of our focus, certain parts of our model are at this point physically unrealistic. In particular, all our simulations are 2D, we model artery walls as an incompressible linear elastic solid, and our outlet boundary conditions are simplistic. But we demonstrate that our method has attractive convergence properties and good scalability, and even at this stage we can qualitatively reproduce some of the more important physical aspects of blood flow in compliant arteries and quantitatively reproduce results found in the literature.

Blood is a viscous shear-thinning fluid, and it is inhomogeneous, containing a significant proportion of non-fluid particles (blood cells and platelets). At physiological conditions it can be assumed to be incompressible. Most modeling also makes the assumption that blood is a Newtonian fluid, which is a reasonable assumption in large arteries but begins to break down in smaller vessels, where the shear-thinning properties of blood become more apparent [17].

Artery walls are a complex tissue with three layers, each with different non-isotropic material properties. Finding and validating accurate physical models for this tissue has been the focus of a lot of effort [26,50].

A moving artery wall implies a moving fluid domain, which means that the computational mesh for the fluid deforms. The displacement of the fluid mesh forms a third field of solution variables, and the governing equations for this field are another aspect that must be modeled. The goal is to provide smooth, well-conditioned elements as the fluid domain deforms at minimal computational cost. One approach is to simply say the mesh displacements must satisfy the Laplace equation, as in [36], but it is possible to model the mesh as a pseudo-structural system with a time-dependent momentum equation [15,16].

There are three general approaches to fluid–structure coupling. The first approach is explicit coupling, in which the fluid and structure subproblems are solved separately, each one using as boundary conditions results from the other's previous time step. This approach does not guarantee any consistency at the fluid–structure interface, and is unstable unless time steps are very small. The second approach is iterative coupling, where at each time step the fluid and structure subproblems are each solved multiple times, updating each other's boundary conditions, until some desired tolerance is reached, at which point you can advance to the next time step [36,20,30,46,47]. Iterative coupling, however, can also become unstable for large time steps, and in some cases can reduce the order of accuracy of the time-stepping algorithm [33]. Finally, fluid and structure can be coupled monolithically, which means that a single system contains all variables and is solved at once with coupling conditions enforced strongly as part of the algebraic system [6,7,19,23–25,28,48].

One of the primary motivations for the explicit or iterative approaches is the ability to use existing fluid and structure codes from off the shelf, rather than developing a separate fluid–structure interaction code. The advantage of monolithic coupling is better and more robust convergence behavior for a variety of parameters. Our approach uses monolithic coupling.

Little work on the blood flow simulation problem makes explicit reference to parallel performance and scalability, or to the particular solution algorithms that are employed. In particular, the research involving the best physical modeling of the problem says very little about the computational efficiency of the algorithms employed [5,7,19]. In contrast, the work that considers solution methods uses simple geometries and does not consider parallel scalability [24,25], and the work that focuses on parallel performance does not do fluid–structure interaction [22].

This paper is organized as follows. We have just introduced the problem and surveyed some previous results in the literature. Next in Section 2, we analyze the mathematical formulation of the problem, including strong and weak forms and the discretization of the problem for numerical solution, both in space and time. Then, in Section 3, we give a description of the parallel computational algorithms and methods we use to solve our problem. In Section 4 we present numerical results, including comparisons with previous fluid–structure interaction studies in the literature. Finally, we offer some concluding remarks.

2. Coupled fluid–structure problem and its discretization

Our emphasis in this paper is tight, monolithic coupling of fluid and structure. In keeping with that, we present the fluid and structure problems together, going through the mathematical model and its approximation by finite elements briefly. See Fig. 1 for a diagram of the fluid–structure computational domain and for some notation.

In order to model fluid–structure interaction, we take the three-field approach [13,16,36]. That is, we model the structure in the Lagrangian frame, and we would like to model the fluid in the Eulerian frame, but since the fluid domain is moving with time, we are forced instead to consider the arbitrary Lagrangian–Eulerian framework [11,14]. Then our three fields are the structure, the fluid, and the moving mesh for the ALE formulation of the moving fluid domain.

For the fluid in Ω_f we use the incompressible Navier–Stokes equations written in the ALE frames

$$\left.\frac{\partial \mathbf{u}_f}{\partial t}\right|_Y + [(\mathbf{u}_f - \omega_g)\cdot\nabla]\mathbf{u}_f + \frac{1}{\rho_f}\nabla p_f = \nu_f \Delta \mathbf{u}_f + \mathbf{f}_f, \tag{1}$$

$$\nabla \cdot \mathbf{u}_f = 0, \tag{2}$$

where ρ_f is the fluid density, ν_f is the kinematic viscosity, $\omega_g = \partial \mathbf{x}_f / \partial t$ is the velocity of the moving mesh and where the Y indicates that the time derivative is to be taken with respect to the ALE coordinates, not the Eulerian coordinates.

The displacement of the structure in Ω_s is governed by a modified equation of incompressible linear elasticity,

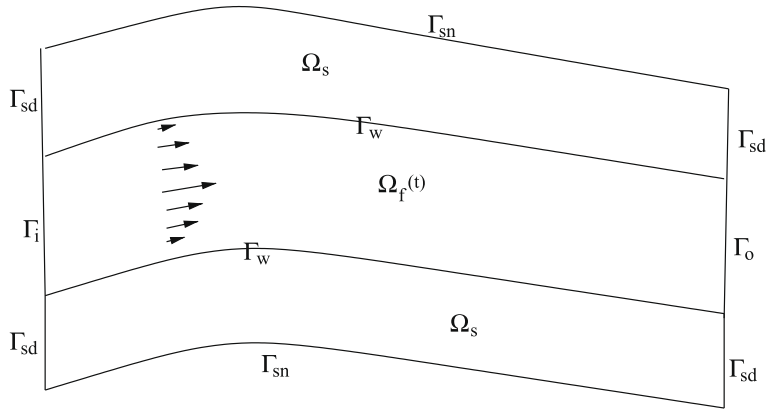

Fig. 1. Ω_0 is the reference configuration of the fluid domain and $\Omega_f(t)$ represents the moving fluid domain at time t; Ω_s is the structure domain in the reference (Lagrangian) configuration. Γ_i and Γ_o are the inlet and outlet boundaries, respectively, to the fluid domain. The dry boundary to the structure Γ_s is further subdivided into a Dirichlet portion Γ_{sd} at inlets and outlets and a Neumann portion Γ_{sn} elsewhere. Γ_w is the wet fluid–structure interaction surface.

$$\rho_s \frac{\partial^2 \mathbf{x}_s}{\partial t^2} + \nabla \cdot \sigma_s - \beta \frac{\partial}{\partial t} \Delta \mathbf{x}_s + \gamma \mathbf{x}_s = \mathbf{f}_s, \tag{3}$$

$$\nabla \cdot \mathbf{x}_s = 0, \tag{4}$$

where $\sigma_s = -p_s I + \mu_s(\nabla \mathbf{x}_s + \nabla \mathbf{x}_s^T)$ is the Cauchy stress tensor for the incompressible structure. In a few cases we use compressible linear elasticity, where (4) is modified to read $\nabla \cdot \mathbf{x}_s + p_s/\lambda_s = 0$. The Lamé parameters λ_s and μ_s are properties of the physical material under consideration, β is a visco-elastic damping parameter, and the γ term is used to represent a radially symmetric artery in two dimensions [2,34]. For the moving mesh we simply make the fluid mesh displacements satisfy a harmonic extension of the moving fluid–structure boundary

$$\Delta \mathbf{x}_f = 0. \tag{5}$$

The assumption that blood is a viscous Newtonian fluid is a good one for large arteries, which is where we focus our analysis. The linear elastic model we use for the artery wall is not such a good model, but it has certainly been used before [27,42,44], and is adequate as long as the wall deformation is not too large. The choice of model for the moving fluid mesh is not physically important, but it does have implications in terms of how much deformation the solver can handle and if the solver becomes ill-conditioned (because of a warped fluid mesh) as the deformation increases. We find that our method, though it is simple, performs quite well, see Fig. 2.

In addition to the equations above, we need boundary conditions, and, much more importantly, coupling conditions at the fluid–structure interface. These coupling conditions are

$$\mathbf{u}_f = \frac{\partial \mathbf{x}_s}{\partial t} \tag{6}$$

which insures continuity of velocity at the wet surface and is the generalization of a no-slip boundary condition for the fluid to the case of a moving boundary,

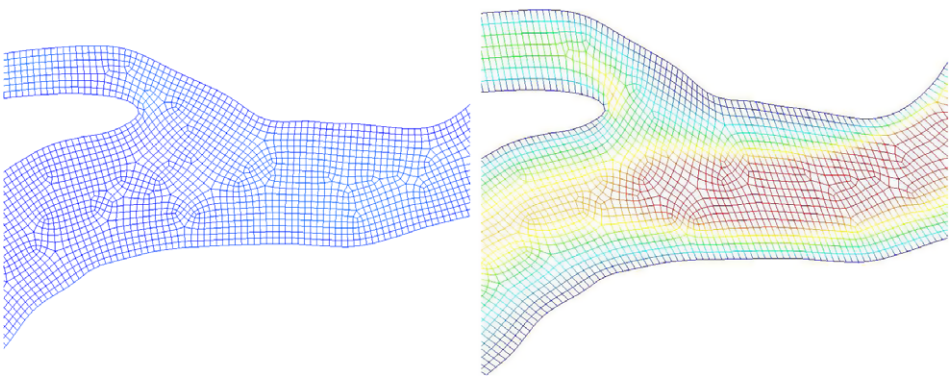

Fig. 2. An undeformed fluid mesh in the reference configuration on the right, compared to the deformed mesh at systole on the right. This simulation is done with a low value of the Young's modulus E and a high inlet velocity to exaggerate the deformation of the fluid domain. The mesh conditioning remains quite acceptable even under large deformation.

$$\sigma_s \cdot \mathbf{n} = -\sigma_f \cdot \mathbf{n} \tag{7}$$

which enforces continuity of traction forces on the boundary, representing in particular the force the fluid imparts on the structure, and

$$\mathbf{x}_f = \mathbf{x}_s \tag{8}$$

which represents the computational mesh for the fluid matching the structure displacement at the boundary, allowing the structure to maintain a Lagrangian description and defining the ALE description for the fluid. Eq. (7) in practice comes in as a (solution-dependent) Neumann-type condition on the right-hand side of the structure equations, while (6) is a (again, solution-dependent) Dirichlet condition on the fluid equations.

We solve the above equations with the finite element method, discretizing in space with the LBB-stable $Q_2 - Q_1$ element. To fully develop the algorithm, we should derive the weak form of the equations, including test functions and function spaces, and from that derive the definitions of all the matrix operators and finite-dimensional approximations that appear. This development, however, is entirely standard, and so here we skip directly to the semi-discrete form which is continuous in time and discrete in space. Those interested in the details should see [4]. The semi-discrete form is

$$M_f \frac{du}{dt} + B(u)u + K_f u - Q_f^T p_f = 0, \tag{9}$$

$$Q_f u = 0, \tag{10}$$

$$\frac{dx_s}{dt} = \dot{x}_s, \tag{11}$$

$$M_s \frac{d\dot{x}_s}{dt} + \beta K_s \dot{x}_s + K_s x_s + \gamma M_s x_s + Q_s^T p_s = A_u u + A_p p_f, \tag{12}$$

$$Q_s x_s = 0, \tag{13}$$

$$K_m x_f = 0, \tag{14}$$

together with the discretized coupling conditions $u_f = \dot{x}_s$ and $x_f = x_s$ on Γ_w, corresponding to (6) and (8). Here, we have two saddle-point problems imbedded in one large, monolithically coupled system. We have reduced the structure equations from second order to first order in time, by doubling the number of degrees of freedom (we keep track of both structure displacement x_s and velocity $\dot{x}_s$), in preparation for the time discretization. Note the right-hand side of this equation, where the terms A_u and A_p are introduced to account for the traction matching in fluid–structure interaction; that is, fluid solution variables are being used to generate a Neumann force on the structure.

Our time discretization is fully implicit, and the preferred method is the second-order trapezoid rule, although our implementation is capable of falling back on a backward Euler method. The trapezoid rule has sometimes been found to be unstable over long time-integrations for nonlinear problems (see for example [21]), but in our case we have not seen instability in any of our tests. We use the same time-stepping scheme for both the fluid and the structure, so we can simply enforce the coupling at each timestep. We can write the time-stepping scheme for the entire monolithically coupled system as

$$M^{n+1}y^{n+1} - M^n y^n - \frac{\Delta t}{2}(K^{n+1}y^{n+1} + K^n y^n) - \frac{\Delta t}{2}(F^{n+1} + F^n) = 0, \tag{15}$$

where

$$y^n = \begin{pmatrix} u_f^n \\ \Delta t p_f^n \\ x_f^n \\ x_s^n \\ \dot{x}_s^n \\ \Delta t p_s^n \end{pmatrix}, \quad M = \begin{pmatrix} M_f & & & & & \\ & & & & & \\ & & I & & & \\ & & & & & \\ & & & & M_s & \\ & & & & & \end{pmatrix}, \tag{16}$$

$$K = \begin{pmatrix} -B - K_f & -(1/\Delta t)Q_f^T & & & & \\ (1/\Delta t)Q_f & & & & & \\ & & K_m & & & \\ & & & & I & \\ -A_u & -A_p & & -K_s & & -(1/\Delta t)Q_s^T \\ & & & (1/\Delta t)Q_s & & \end{pmatrix}. \tag{17}$$

Though written in matrix form, many of the operators above are nonlinear. In particular the B term depends on u_f, and the K_f, M_f and Q terms depend on the moving mesh x_f. This implies that we have a Jacobian of the form

$$J = \begin{pmatrix} J_f & Q_f^T & Z_m & & & \\ -Q_f & & Z_c & & & \\ & & -(\Delta t/2)K_m & & & \\ & & & I & (\Delta t/2)I & \\ (\Delta t/2)A_u & (\Delta t/2)A_p & & -(\Delta t/2)K_s & M_s & Q_s^T \\ & & & -Q_s & & \end{pmatrix}. \tag{18}$$

Here, $J_f = M_f + (\Delta t/2)K_f + (\Delta t/2)\widetilde{J}_f$ and $\widetilde{J}_f$ is the Jacobian derivative matrix of the nonlinear term Bu_f with respect to the fluid velocity variables u_f. Similarly, Z_m and Z_c are the derivatives of the first and second rows, respectively, of (15) with respect to the variables x_f. For more detail on the derivation and form of Z_m, Z_c see Appendix A and [18].

For the sake of space in (18) we have not explicitly shown the coupling conditions (6) and (8). To show the coupled discretization of (6), for example, for only the u_f and $\dot{x}_s$ degrees of freedom, we can write

$$\begin{pmatrix} J_f^{\widetilde{\Omega}_f} & J_f^{\widetilde{\Omega}_f \Gamma_w} & & \\ I & & -I & \\ & M_s^{\widetilde{\Omega}_s} & M_s^{\widetilde{\Omega}_s \Gamma_w} & \\ & M_s^{\Gamma_w \widetilde{\Omega}_s} & M_s^{\Gamma_w} & \end{pmatrix} \begin{pmatrix} u_f^{\widetilde{\Omega}_f} \\ u_f^{\Gamma_w} \\ \dot{x}_s^{\widetilde{\Omega}_s} \\ \dot{x}_s^{\Gamma_w} \end{pmatrix} = \begin{pmatrix} f_f^{\widetilde{\Omega}_f} \\ 0 \\ f_s^{\widetilde{\Omega}_s} \\ f_s^{\Gamma_w} \end{pmatrix}, \tag{19}$$

where the variables u_f and $\dot{x}_s$ have been split into portions on the fluid–structure interface Γ_w and portions on $\widetilde{\Omega}_f = \Omega_f \setminus \Gamma_w$ and $\widetilde{\Omega}_s = \Omega_s \setminus \Gamma_w$, respectively. The coupled discretization of (8) is similar.

This monolithically coupled nonlinear system is solved with a Newton–Krylov–Schwarz algorithm. It is this algorithm that forms the heart of this paper and that we now go on to describe.

3. Scalable domain decomposition algorithms

At each time step, we solve the nonlinear system (15) with an inexact Newton method. At each Newton step we solve a preconditioned linear system of the form $J(y)M^{-1}(Ms) = z$ for the Newton correction s, where M^{-1} is a one-level additive Schwarz preconditioner [38,45]. The nonlinear solver is an inexact Newton method with line search, and the linear system at each Newton step is solved with restarted GMRES. For the nonlinear solver, we provide an exact hand-coded Jacobian that includes all the terms in (18), including the ones related to mesh movement, as in [18]. The nonlinear and linear solvers are standard [38,31], so we focus here on our construction of the preconditioner.

One of the underlying themes of this work is the tight, monolithic coupling of fluid and structure. This monolithic coupling extends all the way down to the preconditioner; even the local solves within the Schwarz preconditioner include both fluid and structure elements, where a subdomain will typically contain both fluid and structure elements. See Fig. 3.

To formally define the restricted additive Schwarz preconditioner M^{-1}, we first partition the entire domain Ω into nonoverlapping subdomains $\Omega_\ell,\ \ell = 1, \ldots, N$. This partition does not take into account the fluid–structure boundary. Then each subdomain Ω_ℓ is extended to overlap its neighbors, with the overlapping domains denoted Ω'_ℓ. The domain decomposition and extension respects element boundaries, so that each Ω_ℓ and Ω'_ℓ consists of an integral number of finite elements. A parameter δ represents the size of the overlap in terms of number of elements. Subdomains on a physical boundary are not extended.

On each subdomain Ω'_ℓ we construct a subdomain preconditioner B_ℓ, which is a restriction of the Jacobian matrix J, that is, it contains entries from J corresponding to degrees of freedom contained in the corresponding subdomain Ω'_ℓ. Homogeneous Dirichlet boundary conditions are used on the subdomain boundaries $\widetilde{\Gamma}$ for all solution variables, including fluid and structure pressure; see Fig. 4. For the Navier–Stokes equations this is not an appropriate boundary condition in theory; however,

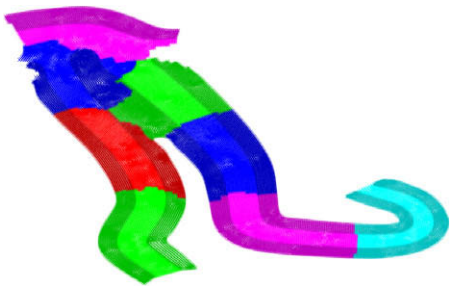

Fig. 3. A portion of the branched geometry, shown with subdomains in different colors and with structure elements darkened compared to fluid elements. Our algorithm does not notice the fluid–structure boundary in the partitions; that is, it sees the colors but not the shading.

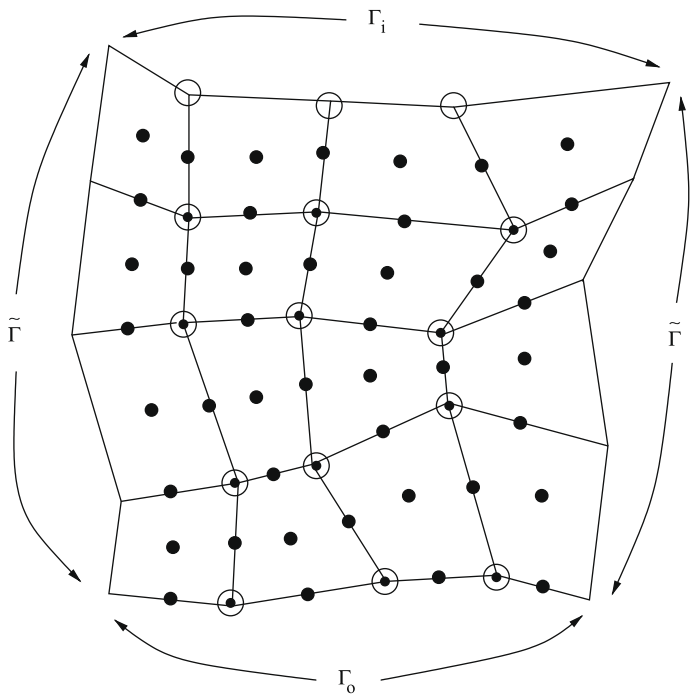

Fig. 4. Boundary conditions for a (fluid-only) subdomain problem. (•) represents a fluid velocity degree of freedom and (○) is fluid pressure. The top and bottom boundaries are physical boundaries, where physical Dirichlet (Γ_i) or Neumann (Γ_o) conditions are applied; the left and right boundaries are subdomain boundaries ($\tilde{\Gamma}$) not corresponding to physical boundaries, where for the subdomain preconditioner non-physical Dirichlet conditions are applied.

in practice it works quite well. We emphasize that this boundary condition choice only affects the preconditioner on local subdomains. For the global solver, appropriate physical boundaries conditions are applied on all boundaries.

The restricted additive Schwarz preconditioner can be written as

$$M^{-1} = (R_1^0)^T B_1^{-1} R_1^\delta + \cdots + (R_N^0)^T B_N^{-1} R_N^\delta. \tag{20}$$

If n is the total number of unknowns in Ω and n'_ℓ is the number of unknowns in Ω'_ℓ, then R_ℓ is an $n'_\ell \times n$ restriction matrix which maps the global vector of unknowns to those belonging to a subdomain. In particular R_ℓ^0 is a restriction that does not include overlap while R_ℓ^δ includes the overlap; if written in matrix form, R_ℓ^0 is the same as R_ℓ^δ with some rows set to zero. The pattern above, where overlap is used to provide information to the subdomain solve, but then the result of that computation in the overlap region is thrown away, is restricted additive Schwarz [9]. Various inexact additive Schwarz preconditioners can be constructed by replacing the matrices B_ℓ above with convenient or inexpensive to compute matrices. In our algorithm we use LU factorization for the subdomain solves.

The ordering of unknowns, though irrelevant mathematically, can have a significant effect on the convergence properties of the solver and on the speed and scalability of our whole algorithm, especially in the parallel setting. In practice, we do not order the systems by field ordering as in (18), but instead element by element. That is, we order the elements so that elements which are nearby geometrically are also as close as possible in the matrix. We order all the unknowns belonging to the first element together, and then all the unknowns that belong to the second but not the first, then all the unknowns that belong to the third element but have not already been placed, and so on. The result is a matrix that, viewed in block structure, has a fairly nice block-banded structure, though the blocks themselves may be messy. At a global level our ordering leads to a matrix that looks mostly like a standard nine-point stencil. See Fig. 5.

The theory for one-level and multilevel Schwarz preconditioners is very well developed for time-independent elliptic problems; see [45,41]. In particular, for elliptic systems the condition number κ of the preconditioned operator satisfies $\kappa \leqslant C(1 + H/\delta)/H^2$ for a one-level preconditioner and $\kappa \leqslant C(1 + H/\delta)$ for the two-level version, where H is the subdomain diameter and C is independent of H, δ, and the discretization size h. The $1/H^2$ term in the one-level preconditioner shows that the number of iterations increases with the number of subdomains. However, for time-dependent parabolic problems, the condition number in the one-level preconditioner remains bounded as long as the time step is not too large [8]. Since our problem is neither elliptic nor parabolic, but instead a complicated system of mixed type, it is not clear how much of this theory applies. We do see increased linear iterations for increasing subdomain counts, but this increase is not as great as the elliptic theory predicts. In the particular case of one-level additive Schwarz preconditioned GMRES, which is our method

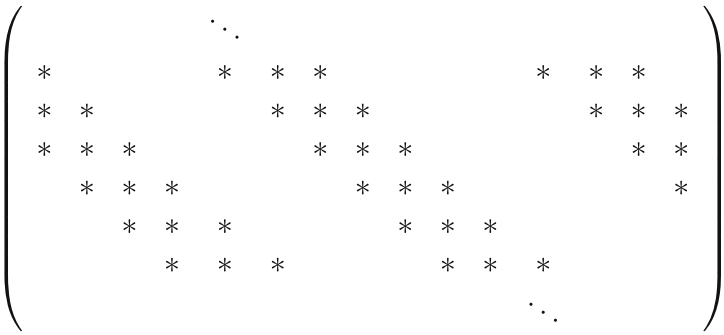

Fig. 5. Nonzero block structure of the Jacobian matrix. The size of the blocks varies, but is no larger than 40 and usually no smaller than 17.

here, recent theory and computations suggest that decreasing the spatial mesh size h should leave the number of GMRES iterations to converge to a prescribed tolerance almost constant [12].

4. Numerical experiments

Our solver is implemented using the Portable Extensible Toolkit for Scientific Computing (PETSc) [3]. All computations are performed on an IBM BlueGene/L supercomputer at the National Center for Atmospheric Research with 1024 compute nodes. Meshes are generated with Cubit from Sandia National Laboratory and partitioned with Parmetis [32,39]. We validate the structure part of the solver based on an analytic problem in [40] and the fluid problem part from [36]. The validation of the fluid–structure problem is discussed below.

4.1. Straight tube

Human artery walls are anisotropic in general, and because of the nature of pulsatile flow we can expect radial displacements to be larger than axial displacements. But there is no reason to believe that axial displacement is always zero. We compare our numerical results with those in [2], where only radial displacement of the artery wall is considered, but our framework also allows for axial displacement. If both are included, the results are noticeably different.

The setup of this test problem is a two-dimensional tube 6 cm by 1 cm, with walls at top and bottom of thickness 0.1 cm. A traction condition is applied at the left boundary to induce a pressure pulse, which then travels to the right, deforming the structure as it goes. In this example, the kinematic viscosity $\nu_f = 0.0035$ kg/m s, the Young's modulus $E = 7.5 \times 10^4$ kg/m s^2, the structure is incompressible, and the inlet pressure pulse takes the form

$$\sigma_f \cdot \mathbf{n}_f = \frac{-P_0}{2}\left[1 - \cos\left(\frac{\pi t}{.0025\text{s}}\right)\right], \tag{21}$$

where $P_0 = 2 \times 10^5$ kg/m s^2. We use $\gamma = 4 \times 10^9$ kg/m^{3}s^2 and $\beta = 0$. The timestep size is $\Delta t = 0.0001$ s, which is small because the timescale of the pressure pulse in (21) is shorter than the pulse produced by a physiological heartbeat. We fix the structure at the inlet and outlet (Γ_{sd}) with a zero displacement condition, and apply zero Neumann conditions to the structure at the dry boundary Γ_{sn}. Neither of these conditions is physically very realistic, but they are commonly used in the literature [19]. A zero traction condition is applied to the fluid at the outlet.

In Fig. 6, we show the pressure averaged along the width of the tube as a function of the axial coordinate for the test problem in [2], with only radial displacements. Similar test problems have been used in [36,1,37]. Our results show very good agreement with the comparison case, as do similar curves for wall displacement and average flow (not shown). The propagation of a pressure pulse is a key test for fluid–structure interaction, as it is completely dependent on the fluid–structure coupling; pressure moves with infinite speed in an incompressible fluid with rigid walls.

The results if we include axial displacement, on the other hand, are noticeably different; see Figs. 7–9. In particular, the pressure pulse is slightly slower, significantly more spread out, and the dip that follows the initial positive pressure pulse is much more pronounced when axial displacement is taken into consideration. Qualitatively similar features can be seen in the pulse profile as viewed in the displacement and average flow plots as compared to those in [2].

4.2. Branching model

Here, we perform simulations for a complex branching geometry derived from clinical data. The model we use comes from biplane angiography data from a pulmonary artery, courtesy of the Children's Hospital, Denver. We project the data to a 2D plane, smooth it, and construct an unstructured quadrilateral mesh with the Cubit package from Sandia National Lab-

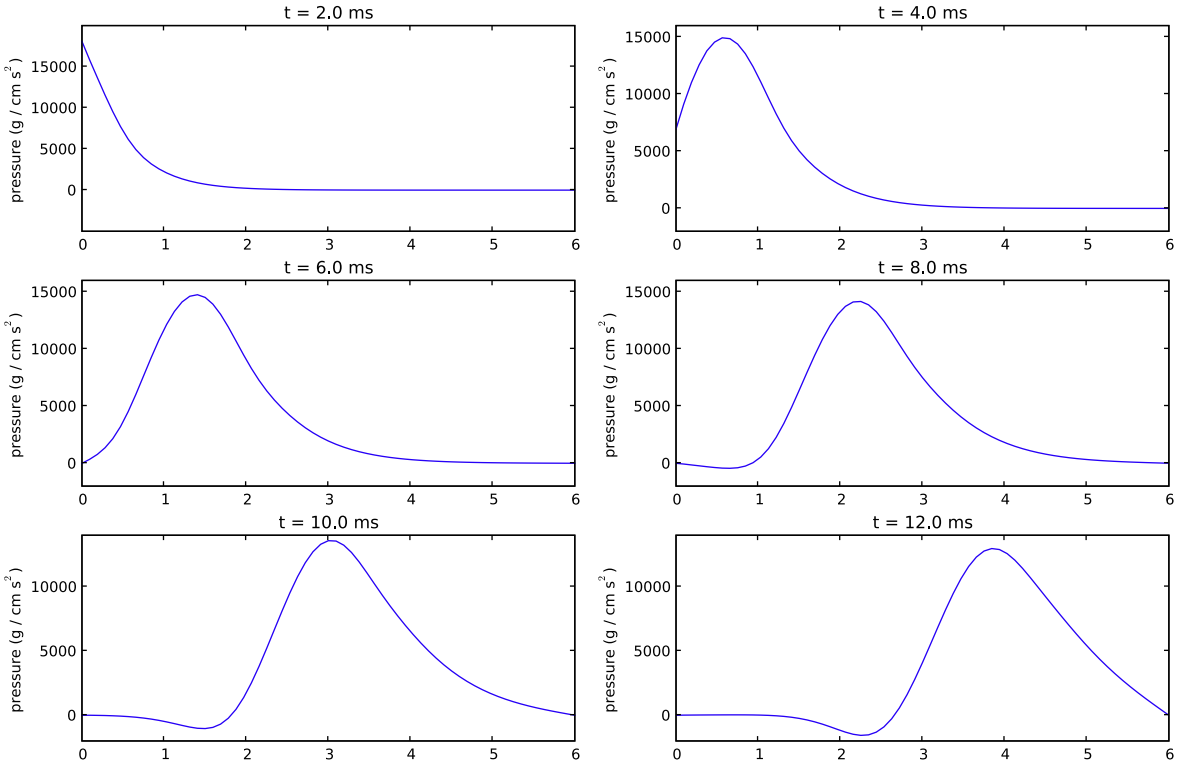

Fig. 6. Average pressure profiles for the test problem in [2] with only radial displacements allowed. The x-axis shows position, in cm, in the axial direction along the model. These results match the test problem very well.

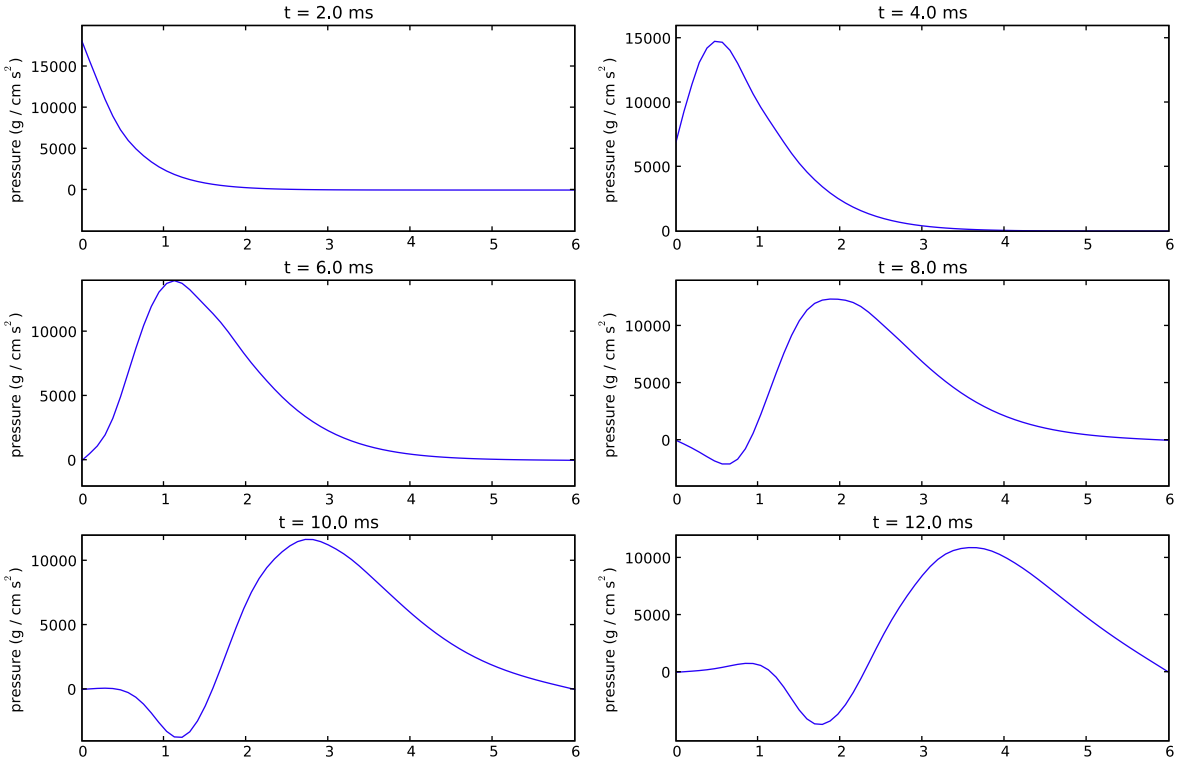

Fig. 7. Average pressure profiles for the test problem in [2] with both radial and axial displacements. The x-axis shows position, in cm, in the axial direction along the model.

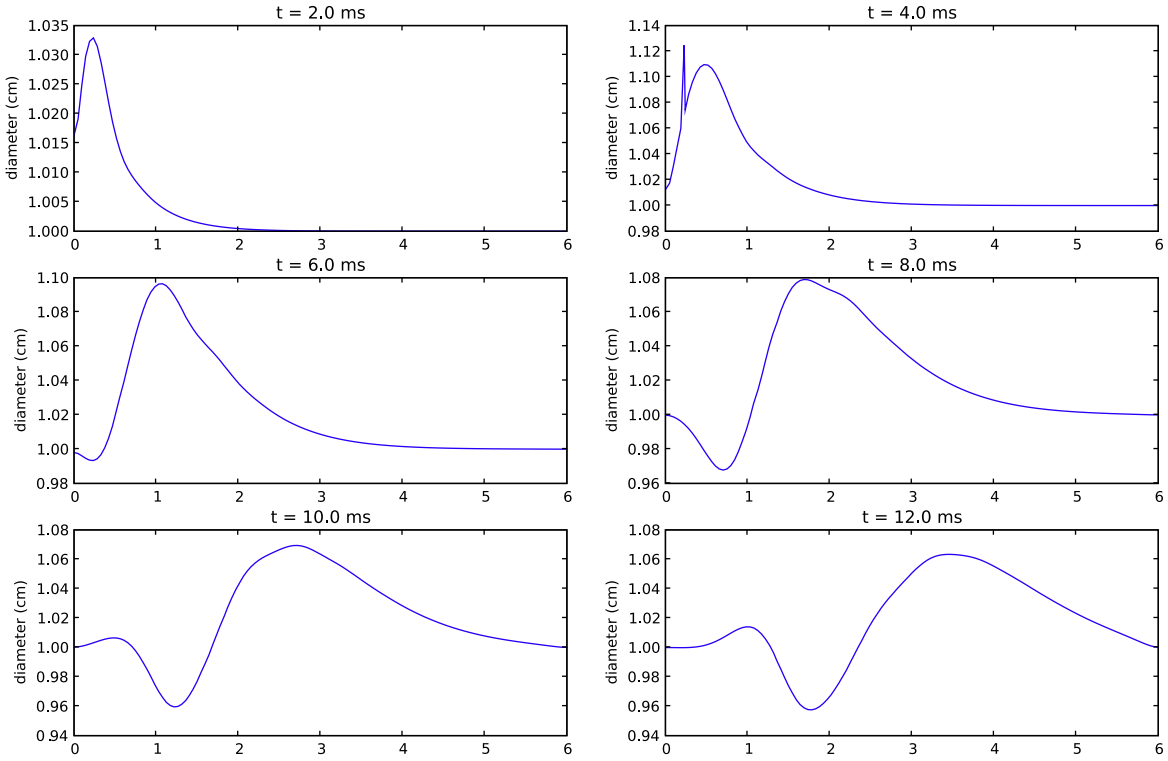

Fig. 8. Diameter of the deforming fluid domain for the test problem in [2], with both radial and axial displacements. The *x*-axis shows position, in cm, in the axial direction along the model.

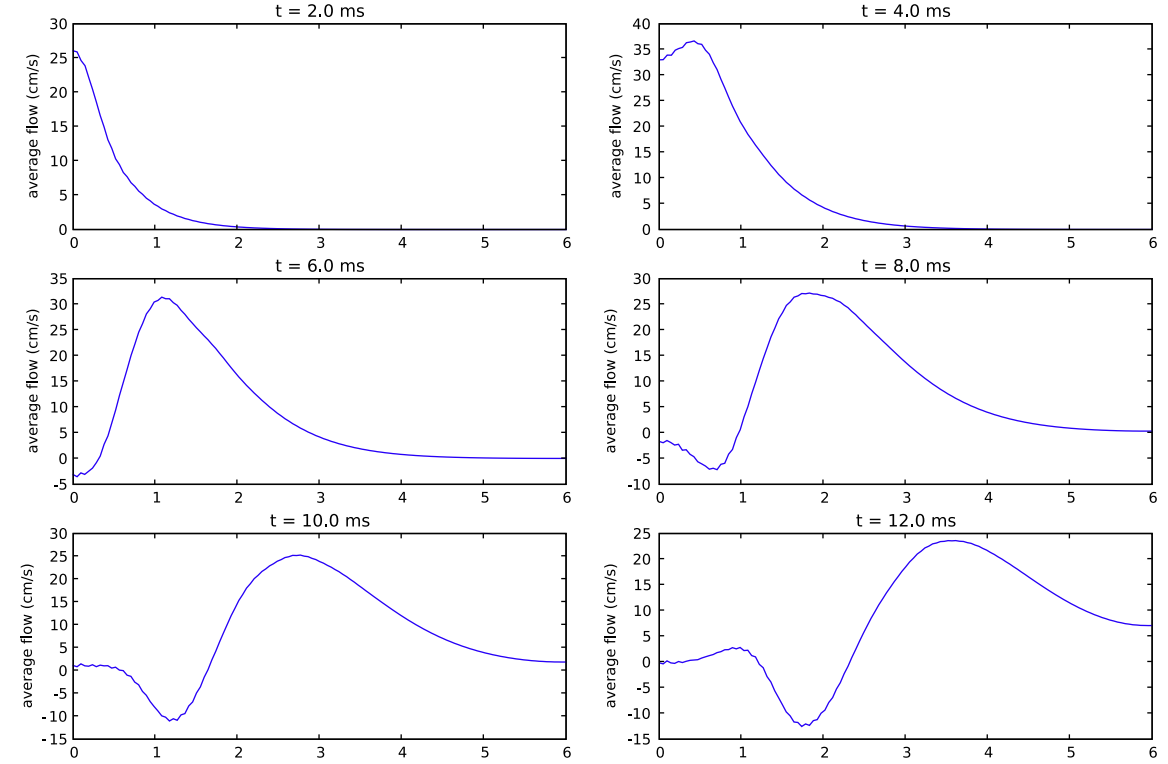

Fig. 9. Average flow profiles for the test problem in [2], with both radial and axial displacements. The *x*-axis shows position, in cm, in the axial direction along the model.

oratory. The fluid–structure interface is specified, and this software guarantees that no elements will cross the interface. The shape of the 2D branching model is visible in Fig. 11, while detail of small pieces of it is visible in Fig. 12. The angiogram data includes blood vessel radius but does not reflect the thickness of the artery wall—we assume the wall thickness is half the artery radius and build the geometry accordingly.

Unless otherwise specified, throughout this section we use a kinematic viscosity $\nu_f = 3.5 \times 10^{-3}$ kg/m s, fluid density and solid density are both 1000 kg/m^3, the Young's modulus of the structure is 7.5×10^3 kg/m s^2, the structure is treated as incompressible, that is, the Poisson ratio is 1/2, and the visco-elastic parameter $\beta = 0.001$ kg/m s. These parameters are chosen from the literature (see [30,7,49]) to be representative of the values in human arteries. The one exception is the Young's modulus E, which is chosen significantly lower than is physiologically reasonable, so that the arteries appear much more flexible in our simulations than in reality—a more realistic value would be 7.5×10^4 kg/m s^2. This choice is made to show more clearly the effects of fluid–structure interaction and to highlight the ability of our solver to handle large deformations of the fluid domain—simulations with varying values of E are shown in Fig. 10, including ones for physiologically realistic value for E used in previous blood flow simulations [2,30]. Though the profiles here are different from each other, they certainly show more similarity with each other than with the rigid-walled case.

In all the numerical results in the remainder of this paper, unless otherwise specified, we use a time step $\Delta t = 0.005$ s—for the parameters of this problem, this time step size resolves essentially the same dynamics as tests with smaller discretization sizes. We stop the linear solver when the preconditioned residual has decreased by a factor of 10^{-4}, we stop the Newton iteration when the nonlinear residual has decreased by a factor of 10^{-6}, and GMRES is set to restart every 60 iterations. For all our simulations we start with zero initial conditions and impulsively apply a velocity boundary condition at the vessel inlet, choosing this case because of the computational difficulty. Tests where the fluid is started from a steady-state rather than impulsively from rest show similar scaling and solver characteristics. For the scaling results we proceed 10 time steps,

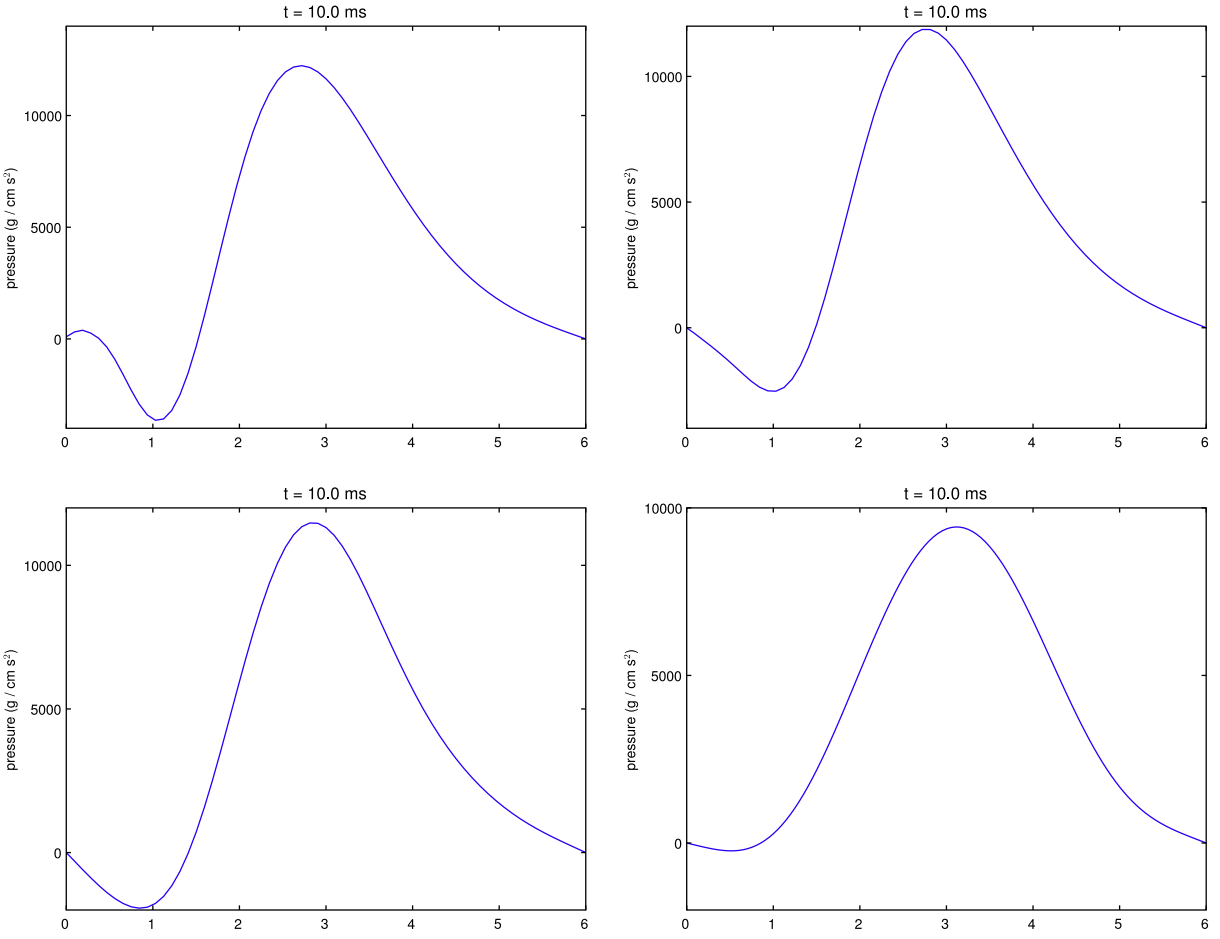

Fig. 10. The pressure pulse in the straight tube problem, at 10 ms, for various values of the Young's modulus E. At top left, $E = 7.5 \times 10^3$ kg/m s^2, at top right $E = 3 \times 10^4$ kg/m s^2, at bottom left $E = 7.5 \times 10^4$ kg/m s^2 (the physiological value used in [2]), and at bottom right $E = 7.5 \times 10^5$ kg/m s^2, which was used in [30]. In all cases $\beta = 0.001$ kg/m s. These plots are to be compared to the bottom left plot in Figs. 6 and 7. The profiles are somewhat different for different E but are qualitatively similar.

reporting average time and nonlinear iteration count per time step, and average GMRES iterations per Newton step. Again, tests of longer duration with more time steps show similar average results. We assign boundary conditions to the structure as in the straight tube example.

Monolithic fluid–structure coupling allows us to stably compute even given large deformations of the fluid wall, and corresponding deformations in the computational domain. Fig. 11 shows an example of this deformation in a simulation that uses realistic physical parameters and a pulsing inlet over two full cardiac cycles. The inlet boundary condition is parabolic in space and time profile like $1 + \sin(\pi t)$ with a maximum of 0.165 m/s.

4.3. Scaling and robustness

The algorithm shows promise of scaling very well; see Fig. 13, which shows several grids scaling almost linearly with number of processors within a certain regime. Unfortunately, as is often the case in one-level Schwarz methods, as the number of subdomains increases, the conditioning of the linear problem gets worse; overlap between adjacent subdomains is not

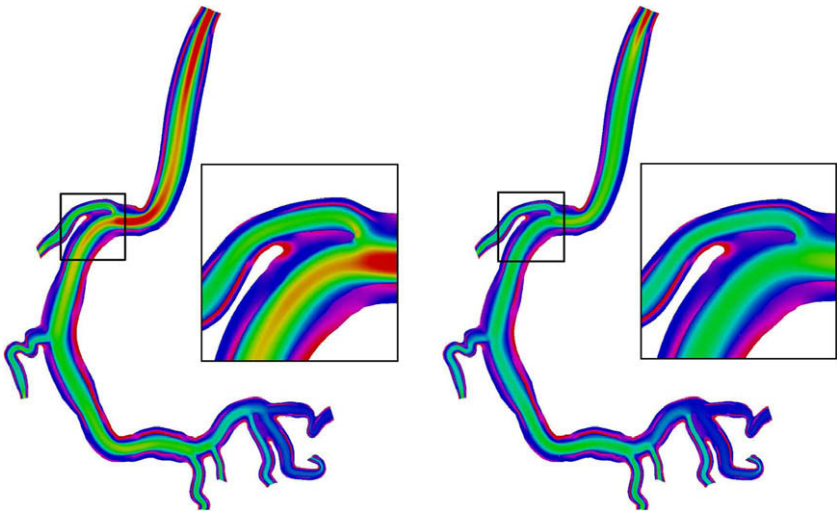

Fig. 11. Deformation of the computational domain at systole (left) and diastole (right). In both pictures the fluid is colored by the norm of velocity (red is high and blue is low) while the structure is colored by pressure (again, red is high and blue is low, but in the structure intermediate values are magenta rather than green). This simulation has Young's modulus $E = 7.5 \times 10^4$ kg/m s^2 and a sinusoidally pulsing inlet boundary condition. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

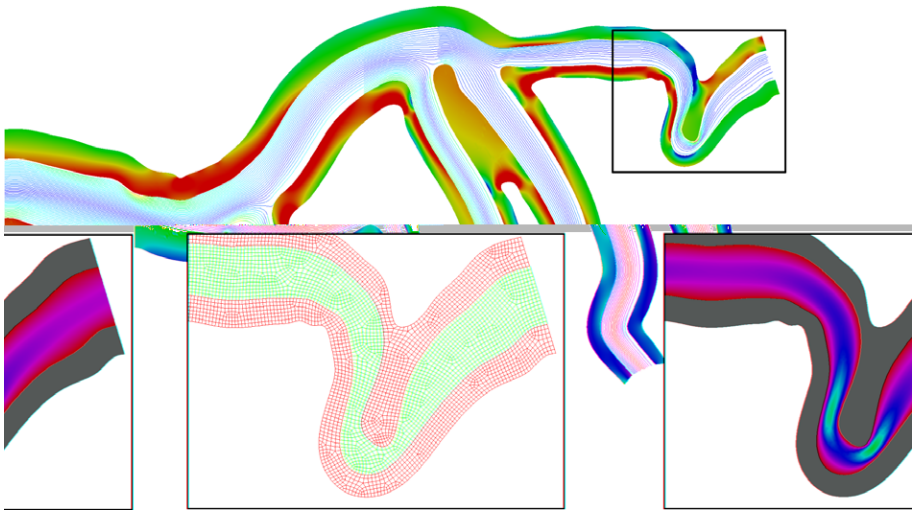

Fig. 12. A portion of our branched computational model. In the main image, the structure is colored by pressure and fluid streamlines are colored by vorticity. In the left inset is a detail of the quadrilateral mesh; in the right inset is a detail of the fluid velocity, both insets representing a magnification of the marked portion.

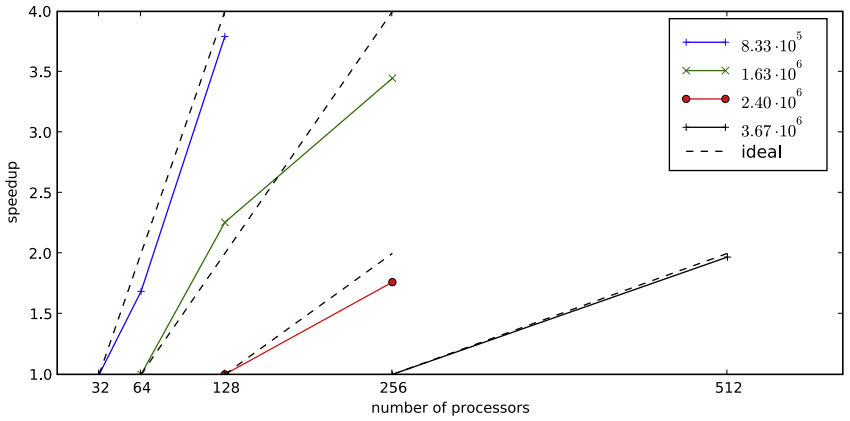

Fig. 13. Speedup versus number of processors for three different size grids, each with the number of unknowns listed in the legend. Speedup is seen to be nearly linear for this range of problem size and processor count.

Table 1
Average number of linear iterations per Newton step and wall time to solution per timestep for increasing number of subdomains, with fixed problem size (1.63 million unknowns) and fixed overlap parameter ($\delta = 4$).

Subdomains	GMRES iterations	Time (s)
64	39.33	273.5
80	43.23	165.5
96	47.97	148.7
112	52.56	117
128	53.54	123.7
160	71.1	85.13
192	77.62	86.26
224	95.87	67.91
256	100.28	63.45

sufficient for global transfer of information. So the number of GMRES iterations increases with number of processors, as you can see in Table 1, where (as in all the tables) "np" is the number of processors, which for our method is always the same as the number of subdomains. This ill-conditioning effect is reduced somewhat if we increase the overlap amount as the processor count increases, but it is still significant. This suggests the need for a two-level Schwarz algorithm with a coarse space.

The solver performs very well for a variety of Reynolds numbers. The compliant structure can absorb some energy from the fluid, so solving the fluid–structure problem for high Reynolds number is noticeably easier than the corresponding rigid-walled problem. Table 2 shows the performance of the solvers for different Reynolds number, as both inlet fluid velocity and fluid viscosity are varied.

The Poisson ratio ν is an important physical parameter for the solid model, with difficulty of simulation typically increasing as ν approaches the incompressible limit $\nu = 1/2$. Biological tissue is nearly incompressible, so it is this limit that is of most interest in the simulation of blood flow. Bazilevs et al. use a Poisson ratio of 0.4 in [7] and use 0.45 more recently in [6], but with our mixed formulation for elasticity we have no difficulty going all the way to incompressibility, with $\nu = 0.5$; see Table 3. Similarly, our solver works well for a large range of Δt; see Table 5.

Table 2
Various Reynolds numbers. All these are grids with 2.40×10^6 degrees of freedom running on 128 processors with overlap parameter $\delta = 4$, and viscous damping $\beta = 0$.

Re	Inlet velocity	Viscosity	Newton	GMRES	Time
5000	5	0.001	3.2	77.97	216.61
4500	45	0.010	4.9	68.76	271.05
3000	30	0.010	4	69.53	224.3
1286	45	0.035	4.9	62.37	242.21
857	30	0.035	3.9	62.46	192.98
500	5	0.010	3	81.93	170.29
429	15	0.035	3.9	68.49	196.26
143	5	0.035	3	72.8	149.32

Table 3
Tests for various values of the Poisson ratio ν. Higher Poisson ratio, even at the incompressible limit, has only a modest effect on the behavior of the solver. For biological tissue, $\nu \approx 0.5$. In both the compressible and incompressible cases we use a mixed formulation for elasticity.

Unknowns	np	ν	Newton	GMRES	Time
8.33×10^5	128	0.4	3.9	37.62	41.72
8.33×10^5	128	0.45	3.9	40.08	42.06
8.33×10^5	128	0.5	3.9	43.97	40.69
1.63×10^6	256	0.4	3.6	57.58	62.13
1.63×10^6	256	0.45	3.6	63.97	64.43
1.63×10^6	256	0.5	3.6	72.22	64.58
2.40×10^6	512	0.4	3.5	93.4	70.88
2.40×10^6	512	0.45	3.4	99.88	70.34
2.40×10^6	512	0.5	3.4	115.29	73.49

Table 4
Here, we show the interplay of various grid sizes, processor counts, and choices of the overlap parameters δ. For larger grids, a larger overlap is needed to keep the number of linear iterations under control. The optimal overlap in each section is highlighted with a *. Here and in Tables 5 and 6 we report the average number of Newton steps per timestep, the average number of GMRES iterations per Newton step, and the average wall clock time (in seconds) to solve one timestep.

Unknowns	np	Overlap	Newton	GMRES	Time
8.33×10^5	64	2	3.9	29.23	99.77
$^*8.33 \times 10^5$	64	3	3.9	21.62	89.54^*
8.33×10^5	64	4	3.9	18.92	120.64
8.33×10^5	64	6	3.9	15.92	102.08
8.33×10^5	64	8	3.9	14.1	116.05
8.33×10^5	128	2	3.9	36.79	41.97
$^*8.33 \times 10^5$	128	3	3.9	28.59	40.75^*
8.33×10^5	128	4	3.9	24.87	43.9
8.33×10^5	128	6	3.9	20.28	65.71
8.33×10^5	128	8	3.9	17.82	65.24
$^*8.33 \times 10^5$	256	2	3.9	72.18	22.68^*
8.33×10^5	256	3	3.9	51.82	26.07
8.33×10^5	256	4	3.9	42.28	24.57
8.33×10^5	256	6	3.9	34.62	38.91
8.33×10^5	256	8	3.9	28.51	57.35
2.40×10^6	128	2	3.9	65.46	185.56
$^*2.40 \times 10^6$	128	3	3.9	44.92	167.96^*
2.40×10^6	128	4	3.9	35.97	177.12
2.40×10^6	128	6	3.9	27.41	218.45
2.40×10^6	256	2	3.9	106.36	88.76
$^*2.40 \times 10^6$	256	3	3.9	75.54	85^*
2.40×10^6	256	4	3.9	60.54	86.56
2.40×10^6	256	6	3.9	47.82	140.42
2.40×10^6	256	8	3.9	41.49	141.81
2.40×10^6	512	2	3.9	384.77	76.99
2.40×10^6	512	3	3.9	311.31	82.24
2.40×10^6	512	4	3.9	201.15	79.88
$^*2.40 \times 10^6$	512	6	3.9	87.67	76.15^*
2.40×10^6	512	8	3.9	69.44	76.65

Another important consideration in the design of fluid–structure interaction algorithms is the added-mass effect, where solution becomes more difficult if the density of the fluid and the structure are close to each other [10]. This problem mostly affects iterative coupling of fluid and structure, and our monolithic coupling seems to be immune to the added-mass effect; see Table 6.

In overlapping domain decomposition preconditioners like the one we use, the overlap parameter δ plays a large role in the solution of the linear system. High δ corresponds to a stronger preconditioner and fewer iterations, but at the cost of more global communication in the parallel algorithm. The effects of various choices of δ can be seen in Table 4.

Table 5
Scaling information for various values of Δt. Normally, problems with smaller time step size are slightly easier to solve, but with our fully implicit method large time steps are handled well.

Unknowns	np	Δt	Newton	GMRES	Time
1.63×10^6	128	1.00×10^{-3}	3.6	14.53	80.81
1.63×10^6	128	2.50×10^{-3}	3.8	36.0	85.33
1.63×10^6	128	5.00×10^{-3}	3.9	37.28	85.8
1.63×10^6	128	1.00×10^{-2}	3.9	139.03	119.51
2.40×10^6	512	1.00×10^{-3}	3.6	32.64	76.54
2.40×10^6	512	2.50×10^{-3}	3.7	93.05	83.44
2.40×10^6	512	5.00×10^{-3}	3.9	87.67	76.15
2.40×10^6	512	1.00×10^{-2}	3.9	477.69	160.74

Table 6
Solver behavior for various combinations of fluid density ρ_f and solid density ρ_s; dynamic viscosity μ_f is kept constant at 3.5×10^{-3} kg/m s. Linear iterations increase for smaller structure density, but in general the solver is robust to various densities, as measured by the time to solution in all cases. In particular, when $\rho_f \approx \rho_s$ which is where the added-mass effect is greatest, solver behavior in the monolithically coupled case is good. These cases are all for a grid with 2.4×10^6 unknowns running on 256 processors.

ρ_f	ρ_s	Newton	GMRES	Time
0.01	1.00	3.9	105.67	262.84
0.10	1.00	3.9	117.21	272.38
1.00	1.00	3.9	113.49	270.96
10.00	1.00	4.1	108.1	268.35
100.00	1.00	5.6	162.8	400.02
1.00	0.01	3.9	275.18	347.07
1.00	0.10	3.9	201.67	308.04
1.00	10.00	3.9	41.03	234.06
1.00	100.00	3.8	52.76	238.06

5. Conclusions and future work

Accurate modeling of blood flow in compliant arteries is a computational challenge. In order to meet this challenge, we need not only to model the physics accurately but also to develop scalable algorithms for parallel computing. In this paper, we have demonstrated a fluid–structure algorithm that monolithically couples fluid to structure, and is therefore, quite robust to changes in physical parameters and large deformation of the fluid domain. In addition, our solver features a Newton–Krylov–Schwarz method that scales well for large grids and large parallel machines, which represents a step forward in blood flow simulation.

In the future, more work is necessary to both improve physical realism and improve scalability. In particular, a coarse space in the Schwarz preconditioner is necessary to insure scalability to large numbers of processors. For physical realism, we need more physically meaningful outlet boundary conditions, a more sophisticated and complete structure model, and eventually a three-dimensional method.

Acknowledgments

Special thanks to Feng-Nan Hwang, Yue Xue, and John Wilson for previous work on this project and to Robin Shandas, Craig Lanning, and Kendall Hunter for help acquiring a patient-specific artery model. Thanks also to Alfio Quarteroni for discussion on the added-mass effect and to Giovanna Guidoboni for discussion about the straight tube model problem.

Appendix A. Jacobian construction

In this appendix, we provide more details for the specific derivations of the terms in the Jacobian matrix J (18) for the nonlinear system (15). Providing an analytic hand-coded Jacobian can be very helpful for the efficiency and accuracy of nonlinear solvers, but the actual construction of the Jacobian is often a practical challenge.

A.1. The nonlinear function

Here, we compute the term J_f from (18), that is, derivatives of the momentum equations with respect to the fluid velocities u_q, v_q. This is the only term of J that is also necessary in the case of an unmoving grid—most of the machinery in this appendix is for the moving grid case. The nonlinear convective term $B(u)u$ in the x direction is

$$\sum_j u_j \int_{\Omega_t} \sum_k \left(\hat{\phi}_k^{(x)} u_k - \frac{\partial x_k}{\partial t} \xi_k^{(x)} \right) \frac{\partial \hat{\phi}_j^{(x)}}{\partial x} \hat{\phi}_i^{(x)} + \sum_j u_j \int_{\Omega_t} \sum_k \left(\hat{\phi}_k^{(y)} v_k - \frac{\partial y_k}{\partial t} \xi_k^{(y)} \right) \frac{\partial \hat{\phi}_j^{(x)}}{\partial y} \hat{\phi}_i^{(x)}, \tag{22}$$

where we have used $\hat{\phi}$ to represent basis functions on the physical domain Ω_t in contrast to the reference domain Ω_0—this distinction will become important later. We have also written out completely the vector-valued basis functions $\hat{\phi}_k^{(x)}$ for clarity—there is a similar term involving the y direction that is omitted above. Since in our implementation we use $\hat{\phi}_k = \xi_k$ and $\partial x_k / \partial t = (x_k^t - x_k^{t-1})/\Delta t$ we can rewrite this as

$$\sum_j u_j \int_{\Omega_t} \sum_k \left(u_k - \frac{x_k^t - x_k^{t-1}}{\Delta t} \right) \hat{\phi}_k^{(x)} \frac{\partial \hat{\phi}_j^{(x)}}{\partial x} \hat{\phi}_i^{(x)} + \sum_j u_j \int_{\Omega_t} \sum_k \left(v_k - \frac{y_k^t - y_k^{t-1}}{\Delta t} \right) \hat{\phi}_k^{(y)} \frac{\partial \hat{\phi}_j^{(x)}}{\partial y} \hat{\phi}_i^{(x)}. \tag{23}$$

In this section of the appendix we are interested in derivatives of the above expressions with respect to u_q and v_q, the velocity degrees of freedom.

Taking the derivative of (23) with respect to u_q, we get

$$\sum_j u_j \int_{\Omega_t} \hat{\phi}_i^{(x)} \frac{\partial \hat{\phi}_j^{(x)}}{\partial x} \hat{\phi}_q^{(x)} + \sum_j \left(u_j - \frac{x_j^t - x_j^{t-1}}{\Delta t} \right) \int_{\Omega_t} \hat{\phi}_i^{(x)} \hat{\phi}_j^{(x)} \frac{\partial \hat{\phi}_q^{(x)}}{\partial x} + \int_{\Omega_t} \sum_k \left(v_k - \frac{y_k^t - y_k^{t-1}}{\Delta t} \right) \hat{\phi}_i^{(x)} \hat{\phi}_k^{(y)} \frac{\partial \hat{\phi}_q^{(x)}}{\partial y}. \tag{24}$$

And now taking the derivative with respect to v_q we have

$$\sum_j u_j \int_{\Omega_t} \hat{\phi}_i^{(x)} \frac{\partial \hat{\phi}_j^{(x)}}{\partial y} \hat{\phi}_q^{(y)}. \tag{25}$$

The y direction equivalent of (22) behaves similarly, and this is how we compute the term J_f in (18).

A.2. Dependence of the nonlinear function on the mesh

Here, we are interested in derivatives of the momentum equations with respect to the moving mesh, so we consider (23) with respect to x_q and y_q. As we will see, there are two dependencies here—one is explicit, because of the appearance of the mesh velocity in the advective term, and the other appears because of the integration over a moving domain. We consider only the explicit nonlinear dependence in this section, and turn to the implicit dependence in Sections A.3 and A.4. This provides one component of the Z_m term in (18).

Taking the derivative of (23) with respect to x_q we get

$$\sum_j u_j \int_{\Omega_t} \frac{-1}{\Delta t} \hat{\phi}_q^{(x)} \frac{\partial \hat{\phi}_j^{(x)}}{\partial x} \hat{\phi}_i^{(x)}. \tag{26}$$

The y direction is similar.

A.3. Continuity equation

The momentum and continuity equations depend nonlinearly on the moving mesh also because the integrals in the weak form of the equations are taken over a moving domain, which depends on the mesh variables. This nonlinear dependence on the moving mesh is quite a bit more complex. To understand it and show its form, we go in detail through the example of the terms in Z_c in (18), the dependence of the continuity equation on the moving mesh. In Section A.4 we will discuss how to generalize this to the terms in Z_m. The continuity equation in its discrete form is $Qu = 0$, and the Jacobian entries in Z_c will take the form

$$J_{ik} = \frac{\partial}{\partial y_k} \sum_j Q_{ij} u_j = \sum_j u_j \frac{\partial}{\partial y_k} Q_{ij} \tag{27}$$

with similar forms for the x_k derivatives. Our finite element discretization gives us

$$Q_{ij} = \int_{\Omega_t} \hat{\psi}_i \frac{\partial \hat{\phi}_j}{\partial x} \tag{28}$$

for some of the entries, or a similar form with a derivative with respect to y. We want to calculate

$$\frac{\partial}{\partial x_k} Q_{ij}, \quad \frac{\partial}{\partial y_k} Q_{ij} \tag{29}$$

so as to compute (27). The first thing to notice is that the domain of integration itself in (28) depends on the quantities x_k, y_k. To deal with this, we do what we do in practical implementations of the finite element method anyway, and map the physical domain Ω_t to a reference domain $\Omega_0 = [-1, 1]^2$.

At this point we need to introduce some notation. The coordinates on the physical domain are x, y and those on Ω_0 are ξ, η. We will denote the finite element basis functions on Ω_t by $\hat{\phi}$ and those on Ω_0 by ϕ. We have the mappings

$$x(\xi,\eta) = \sum_s \phi_s(\xi,\eta)x_s, \quad y(\xi,\eta) = \sum_s \phi_s(\xi,\eta)y_s \tag{30}$$

as interpolations based on the known mesh positions x_s and y_s. We do not have easy access to the inverse mappings $\xi(x,y), \eta(x,y)$, but we can still transform integrals from Ω_t to Ω_0 by

$$\int_{\Omega_t} f(x,y) = \int_{\Omega_s} f(x(\xi,\eta), y(\xi,\eta))j \tag{31}$$

where

$$j(\xi,\eta) = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial x}{\partial \eta}\frac{\partial y}{\partial \xi} \tag{32}$$

is the determinant of the Jacobian of the mapping (30)—we will call it the element Jacobian in contrast to the nonlinear system Jacobian J.

We transform the integral in (28) to get

$$Q_{ij} = \int_{\Omega_t} \hat{\psi}_i \frac{\partial \hat{\phi}_j}{\partial x} = \int_{\Omega_0} \psi_i \frac{\partial \phi_j}{\partial x} j. \tag{33}$$

Because of the finite element mapping (see for example [29]) we have

$$\frac{\partial \phi_j}{\partial x} = \left(\frac{\partial \phi_j}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial \phi_j}{\partial \eta}\frac{\partial y}{\partial \xi}\right)\Big/ j. \tag{34}$$

With this relation we can write

$$Q_{ij} = \int_{\Omega_0} \psi_i \left(\frac{\partial \phi_j}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial \phi_j}{\partial \eta}\frac{\partial y}{\partial \xi}\right), \tag{35}$$

and then using the interpolations (30) we have

$$\frac{\partial y}{\partial \eta} = \sum_s \frac{\partial \phi_s}{\partial \eta} y_s \tag{36}$$

and similarly for $\partial y/\partial \xi$ and similar terms involving the x mapping. In this form we can take the kind of derivative we want, namely

$$\frac{\partial}{\partial y_k}\frac{\partial y}{\partial \eta} = \frac{\partial \phi_k}{\partial \eta}. \tag{37}$$

Applying this to all the terms in (35), we get

$$\frac{\partial}{\partial y_k} Q_{ij} = \int_{\Omega_0} \psi_i \left(\frac{\partial \phi_j}{\partial \xi}\frac{\partial \phi_k}{\partial \eta} - \frac{\partial \phi_j}{\partial \eta}\frac{\partial \phi_k}{\partial \xi}\right) \tag{38}$$

which is an integral consisting only of quantities on the reference element—all dependence on x, y has been suppressed, and so this integral can be computed in the algorithm. Inserting this into (27), we can calculate the entries of the Jacobian corresponding to the nonlinear effects of the moving mesh on the continuity equation—that is, we can fill in the entries of Z_c in (18).

A.4. Other mesh-dependence terms

The other mesh-dependence terms are derived similarly to those for the continuity equation. Each of the other terms forms a part of the term Z_m in (18), but the contributions for the mass matrix M_f, the viscosity matrix K_f, the pressure term Q_f^T, and the nonlinear term B are each calculated separately and then added into the Z_m block of the Jacobian.

The general process is similar to that for the continuity equation—map the integral to the reference domain, and then use (30) and (32) and relations like (34) to calculate derivatives with respect to x_k, y_k. In order to carry this out, we need the corresponding y equation to (34), namely

$$\frac{\partial \phi_j}{\partial y} = \left(-\frac{\partial \phi_j}{\partial \xi}\frac{\partial x}{\partial \eta} + \frac{\partial \phi_j}{\partial \eta}\frac{\partial x}{\partial \xi}\right)\Big/ j. \tag{39}$$

In the case of the continuity equation, two appearances of the element Jacobian j cancelled. In other cases we are not so lucky, so we may need the derivatives

$$\frac{\partial}{\partial x_k} j = \frac{\partial \phi_k}{\partial \xi} \sum_s \frac{\partial \phi_s}{\partial \eta} y_s - \frac{\partial \phi_k}{\partial \eta} \sum_s \frac{\partial \phi_s}{\partial \xi} y_s, \tag{40}$$

$$\frac{\partial}{\partial y_k} j = \frac{\partial \phi_k}{\partial \eta} \sum_s \frac{\partial \phi_s}{\partial \xi} x_s - \frac{\partial \phi_k}{\partial \xi} \sum_s \frac{\partial \phi_s}{\partial \eta} x_s. \tag{41}$$

With these, using the product rule in combination with (34), (39) and (30), we can calculate derivatives with respect to x_k, y_k for any combinations of basis function derivatives. The resulting expressions can be quite complicated to write down but the derivations are not difficult once we have the machinery above.

References

[1] S. Badia, F. Nobile, C. Vergara, Fluid–structure partitioned procedures based on Robin transmission conditions, J. Comput. Phys. 227 (14) (2008) 7027–7051.
[2] S. Badia, A. Quaini, A. Quarteroni, Splitting methods based on algebraic factorization for fluid–structure interaction, SIAM J. Sci. Comput. 30 (4) (2008) 1778–1805.
[3] S. Balay, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, PETSc Users Manual, Technical Report, Argonne National Laboratory, 2008.
[4] A.T. Barker, Monolithic Fluid–structure Interaction Algorithms for Parallel Computing with Application to Blood Flow, Ph.D. Thesis, University of Colorado, Boulder, May 2009.
[5] Y. Bazilevs, V. Calo, J. Cottrell, T. Hughes, A. Reali, G. Scovazzi, Variational multiscale residual-based turbulence modeling for large eddy simulation of incompressible flows, Comput. Methods Appl. Mech. Eng. 197 (2007) 173–201.
[6] Y. Bazilevs, V. Calo, T. Hughes, Y. Zhang, Isogeometric fluid–structure interaction: theory, algorithms and computations, Comput. Mech. (2008) 3–27.
[7] Y. Bazilevs, V. Calo, Y. Zhang, T. Hughes, Isogeometric fluid–structure interaction analysis with applications to arterial blood flow, Comput. Mech. 38 (2006) 310–322.
[8] X.-C. Cai, Additive Schwarz algorithms for parabolic convection–diffusion equations, Numer. Math. 60 (1990) 42–62.
[9] X.-C. Cai, M. Sarkis, A restricted additive Schwarz preconditioner for general sparse linear systems, SIAM J. Sci. Comput. 21 (1999) 792–797.
[10] P. Causin, J.F. Gerbeau, F. Nobile, Added-mass effect in the design of partitioned algorithms for fluid–structure problems, Comput. Methods Appl. Mech. Eng. 194 (42–44) (2005) 4506–4527.
[11] J. Donea, A. Huerta, J.-P. Ponthot, A. Rodríguez-Ferran, Arbitrary Lagrangian-Eulerian methods, in: E. Stein, R. de Borst, T.J. Hughes (Eds.), Encyclopedia of Computational Mechanics, vol. 1, Wiley, 2004, pp. 1–25.
[12] X. Du, D.B. Szyld, A note on the mesh independence of convergence bounds for additive Schwarz preconditioned GMRES, Numer. Linear Algebra Appl. 15 (2008) 547–557.
[13] C. Farhat, CFD on moving grids: from theory to realistic flutter, maneuvering, and multidisciplinary optimization, Int. J. Comput. Fluid Dyn. (2005) 595–603.
[14] C. Farhat, P. Geuzaine, Design and analysis of robust ALE time-integrators for the solution of unsteady flow problems on moving grids, Comput. Methods Appl. Mech. Eng. 193 (39-41) (2004) 4073–4095.
[15] C. Farhat, P. Geuzaine, C. Grandmont, The discrete geometric conservation law and the nonlinear stability of ALE schemes for the solution of flow problems on moving grids, J. Comput. Phys. 174 (2) (2001) 669–694.
[16] C. Farhat, M. Lesoinne, N. Maman, Mixed explicit/implicit time integration of coupled aeroelastic problems: three-field formulation, geometric conservation and distributed solution, Int. J. Numer. Methods Fluids 21 (10) (1995) 807–835 (Finite element methods in large-scale computational fluid dynamics (Tokyo, 1994)).
[17] L. Fatone, P. Gervasio, A. Quarteroni, Multimodels for incompressible flows, J. Math. Fluid Mech. 2 (2) (2000) 126–150.
[18] M.Á. Fernández, M. Moubachir, A Newton method using exact Jacobians for solving fluid–structure coupling, Comput. Struct. 83 (2005) 127–142.
[19] C.A. Figueroa, I.E. Vignon-Clementel, K.E. Jansen, T.J.R. Hughes, C.A. Taylor, A coupled momentum method for modeling blood flow in three-dimensional deformable arteries, Comput. Methods Appl. Mech. Eng. 195 (41-43) (2006) 5685–5706.
[20] L. Formaggia, J.F. Gerbeau, F. Nobile, A. Quarteroni, On the coupling of 3D and 1D Navier–Stokes equations for flow problems in compliant vessels, Comput. Methods Appl. Mech. Eng. 191 (6-7) (2001) 561–582.
[21] B. Fornberg, On the instability of leap-frog and Crank–Nicolson approximations of a nonlinear partial differential equation, Math. Comput. 27 (1973) 45–57.
[22] L. Grinberg, G.E. Karniadakis, A scalable domain decomposition method for ultra-parallel arterial flow simulations, Commun. Comput. Phys. 4 (2008) 1151–1169.
[23] M. Heil, An efficient solver for the fully coupled solution of large-displacement fluid–structure interaction problems, Comput. Methods Appl. Mech. Eng. 193 (1–2) (2004) 1–23.
[24] J.J. Heys, E. Lee, T.A. Manteuffel, S.F. McCormick, On mass-conserving least-squares methods, SIAM J. Sci. Comput. 28 (5) (2006) 1675–1693 (Electronic).
[25] J.J. Heys, T.A. Manteuffel, S.F. McCormick, J.W. Ruge, First-order system least squares (FOSLS) for coupled fluid-elastic problems, J. Comput. Phys. 195 (2) (2004) 560–575.
[26] G.A. Holzapfel, T.C. Gasser, R.W. Ogden, A new constitutive framework for arterial wall mechanics and a comparative study of material models, J. Elasticity 61 (1–3) (2000) 1–48 (Soft tissue mechanics).
[27] J. Hron, M. Mádlík, Fluid–structure interaction with applications in biomechanics, Nonlinear Anal. Real World Appl. 8 (5) (2007) 1431–1458.
[28] B. Hübner, E. Walhron, D. Dinkler, A monolithic approach to fluid–structure interaction using space-time finite elements, Comput. Methods Appl. Mech. Eng. 193 (2004) 2087–2104.
[29] T.J. Hughes, The Finite Element Method: Linear Static and Dynamic Finite Element Analysis, Dover, 2000.
[30] K.S. Hunter, C.J. Lanning, S.-Y.J. Chen, Y. Zhang, R. Garg, D.D. Ivy, R. Shandas, Simulations of congenital septal defect closure and reactivity testing in patient-specific models of the pediatric pulmonary vasculature: a 3D numerical study with fluid–structure interaction, J. Biomech. Eng. 128 (2006) 564–572.
[31] F.-N. Hwang, X.-C. Cai, A parallel nonlinear additive Schwarz preconditioned inexact Newton algorithm for incompressible Navier–Stokes equations, J. Comput. Phys. 204 (2005) 666–691.
[32] G. Karypis, Metis/Parmetis Web Page, University of Minnesota, 2008. <http://glaros.dtc.umn.edu/gkhome/views/metis>.
[33] E. Kuhl, S. Hulshoff, R. de Borst, An arbitrary Lagrangian Eulerian finite-element approach for fluid–structure interaction phenomena, Int. J. Numer. Methods Eng. 57 (2003) 117–142.
[34] X. Ma, G. Lee, S. Wu, Numerical simulation for the propagation of nonlinear pulsatile waves in arteries, Trans. ASME 114 (1992) 490–496.
[35] C.J. Murray, A.D. Lopez, Mortality by cause for eight regions of the world: Global Burden of Disease study, Lancet 349 (1997) 1269–1276.
[36] F. Nobile, Numerical Approximation of Fluid–structure Interaction Problems with Application to Haemodynamics, Ph.D. Thesis, École Polytechnique Fédérale de Lausanne, 2001.

[37] F. Nobile, C. Vergara, An effective fluid–structure interaction formulation for vascular dynamics by generalized Robin conditions, SIAM J. Sci. Comput. 30 (2) (2008) 731–763.
[38] S. Ovtchinnikov, F. Dobrian, X.-C. Cai, D.E. Keyes, Additive Schwarz-based fully coupled implicit methods for resistive Hall magnetohydrodynamic problems, J. Comput. Phys. 225 (2) (2007) 1919–1936.
[39] S.J. Owen, J.F. Shepherd, Cubit Project Web Page, 2008. <http://cubit.sandia.gov/>.
[40] L.F. Pavarino, E. Zampieri, Overlapping Schwarz and spectral element methods for linear elasticity and elastic waves, J. Sci. Comput. 27 (1–3) (2006) 51–73.
[41] B. Smith, P. Bjørstad, W. Gropp, Domain Decomposition, Cambridge University Press, Cambridge, 1996.
[42] D. Tang, C. Yang, H. Walker, S. Kobayashi, D.N. Ku, Simulating cyclic artery compression using a 3D unsteady model with fluid–structure interactions, Comput. Struct. 80 (2002) 1651–1665.
[43] C.A. Taylor, M.T. Draney, Experimental and computational methods in cardiovascular fluid mechanics, Ann. Rev. Fluid Mech. 36 (2004) 197–231.
[44] R. Torii, M. Oshima, T. Kobayashi, K. Takagi, T.E. Tezduyar, Influence of wall elasticity in patient-specific hemodynamic simulations, Comput. Fluids 36 (2007) 160–168.
[45] A. Toselli, O. Widlund, Domain Decomposition Methods—Algorithms and Theory, Springer-Verlag, Berlin, 2005.
[46] F. van de Vosse, J.D. Hart, C.V. Oijen, D. Bessems, T. Gunther, A. Segal, B. Wolters, J. Stijnen, F. Baaijens, Finite-element-based computational methods for cardiovascular fluid–structure interaction, J. Eng. Math. 47 (2003) 335–368.
[47] I.E. Vignon-Clementel, C.A. Figueroa, K.E. Jansen, C.A. Taylor, Outflow boundary conditions for three-dimensional finite element modeling of blood flow and pressure in arteries, Comput. Methods Appl. Mech. Eng. 195 (29-32) (2006) 3776–3796.
[48] X. Yue, F.-N. Hwang, R. Shandas, X.-C. Cai, Simulation of branching blood flows on parallel computers, Biomed. Sci. Instrum. 40 (2004) 325–330.
[49] M. Zamir, The Physics of Pulsatile Flow, Springer, 2000.
[50] Y. Zhang, M.L. Dunn, E. Drexler, C. McCowan, A. Slifka, D. Ivy, R. Shandas, A microstructural hyperelastic model of pulmonary arteries under normo- and hypertensive conditions, Ann. Biomed. Eng. 33 (2005) 1042–1052.