Two-Level Methods for Blood Flow Simulation

Andrew T. Barker[1] and Xiao-Chuan Cai[2]

[1] Department of Mathematics, Center for Computation and Technology, Louisiana State University, Baton Rouge, LA 70803-4918, USA, andrewb@math.lsu.edu

[2] Department of Computer Science, University of Colorado, Boulder, CO 80309-0430, USA, cai@cs.colorado.edu

1 Introduction

We consider two-level Newton-Krylov-Schwarz algorithms for blood flow in arteries, which is a computationally difficult and practically important application area [6, 8]. In particular, the similar densities of blood and artery wall make the coupling between fluid and structure strong in both directions, so that partitioned or iterative procedures have difficulties due to the added-mass effect [4]. Instead of a partitioned procedure, we adopt a monolithic computational approach, coupling fluid to structure in one large system that is solved all at once. This tight coupling allows for robustness to parameters and makes our method immune to the added-mass effect. The resulting system is difficult to solve, but we show here that it can be solved efficiently with effective preconditioning strategies specifically designed for parallel computing.

2 Mathematical Model and Discretization

We solve the fully coupled and nonlinear equations for fluid-structure interaction with monolithic coupling of the three components, the fluid, the elastic wall structure, and the moving mesh.

Our visco-elastic model for the artery wall is

$$\rho_s \frac{\partial^2}{\partial t^2}\mathbf{x}_s = \nabla \cdot \sigma_s + \beta \frac{\partial}{\partial t}(\Delta \mathbf{x}_s) - \gamma \mathbf{x}_s \tag{1}$$

where $\mathbf{x}_s$ is the structural displacement, $\sigma_s = -p_s I + (2/3)E_s(\nabla \mathbf{x}_s + \nabla \mathbf{x}_s^T)$ is the Cauchy stress tensor that involves the unknown pressure p_s, ρ_s is the structure density, and β is a visco-elastic parameter. The γ term is included so that we can reproduce a standard fluid-structure test problem with one-dimensional structure as in [1]. To specify the grid displacements $\mathbf{x}_f$, we simply use the Laplace equation $\Delta \mathbf{x}_f = 0$ on the interior of the domain, following the practice in [10].

We model the fluid as a viscous incompressible Newtonian fluid, using the Navier–Stokes equations in the ALE frame

$$\left.\frac{\partial \mathbf{u}_f}{\partial t}\right|_Y + [(\mathbf{u}_f - \omega_g)\cdot\nabla]\mathbf{u}_f + \frac{1}{\rho_f}\nabla p_f = \nu_f \Delta \mathbf{u}_f, \tag{2}$$

$$\nabla\cdot\mathbf{u}_f = 0, \tag{3}$$

Here $\mathbf{u}_f$ is the fluid velocity vector and p_f is the pressure. The given data include the fluid density ρ_f and the kinematic viscosity $\nu_f = \mu_f/\rho_f$. The ALE mesh velocity is $\omega_g = \partial\mathbf{x}_f/\partial t$ and the Y indicates that the time derivative is to be taken in the ALE frame.

Boundary conditions for the fluid equations typically consist of a Dirichlet condition where $\mathbf{u}_f$ takes a given profile at the inlet Γ_i, and a zero traction condition $\sigma_f\cdot\mathbf{n}_f = \mu_f(\nabla\mathbf{u}_f\cdot\mathbf{n}_f) - p_f\mathbf{n}_f = 0$ at the outlet, where $\mathbf{n}_f$ is the unit outward normal. Here we have used $\sigma_f = -p_f I + \mu_f(\nabla\mathbf{u}_f)$.

The physical system, as well as our model and discretization, has strong coupling between the three fields. At the fluid-structure boundary we require that structure velocity match fluid velocity, $\mathbf{u}_f = \partial\mathbf{x}_s/\partial t$, which is a generalization of a no-slip, no penetration condition. We also enforce that the moving mesh must follow the solid displacement, so that the structure can maintain a Lagrangian description. This condition takes the form $\mathbf{x}_f = \mathbf{x}_s$. Again, this reduces to a homogeneous Dirichlet condition in the case of a rigid wall. Finally, we enforce the continuity of traction forces at the boundary. This can be written $\sigma_s\cdot\mathbf{n}_s = -\sigma_f\cdot\mathbf{n}_f$, where $\mathbf{n}_s, \mathbf{n}_f$ are the unit outward normal vectors for the solid and fluid domains, respectively, and σ_s and σ_f are the Cauchy stress tensors. The condition can be thought of as a Neumann-type condition on the structure model. It is important to emphasize that these coupling conditions are enforced implicitly as part of the monolithic system – they are never enforced as boundary conditions with given data from subproblems, as in the iterative coupling approach.

We discretize the coupled system with $Q_2 - Q_1$ finite elements for both fluid and structure. We discretize in time with the second order implicit trapezoid rule $y^{n+1} = y^n + (\Delta t/2)(F^{n+1} + F^n)$. For the sake of brevity, we skip the derivation of the weak form (which is standard) and present the fully discrete system. At each time step we solve a nonlinear system of the form

$$(\tilde{M} - (\Delta t/2)\tilde{K})y^{n+1} = (\tilde{M} + (\Delta t/2)\tilde{K})y^n \tag{4}$$

where

$$y^n = \begin{pmatrix} u_f \\ p_f \\ x_s \\ \dot{x}_s \\ p_s \\ x_f \end{pmatrix}^n, \quad \tilde{M} = \begin{pmatrix} M_f & & & & & \\ & & & & & \\ & & I & & & \\ & & & \rho_s M_s & & \\ & & & & & \\ & & & & & \end{pmatrix}, \tag{5}$$

and

$$\tilde{K} = \begin{pmatrix} N(u_f) - \nu_f K_f & -Q_f^T & & & & \\ Q_f & & & & & \\ & & & I & & \\ A_u & A_p & K_s + \gamma M_s & \beta K_s & -Q_s^T & \\ & & Q_s & & & \\ & & & & & K_m \end{pmatrix}. \tag{6}$$

In the above, M_f, M_s are finite-element mass matrices for the fluid and structure, $N(u_f)$ represents the nonlinear term in (2), Q_f, Q_s are discrete divergence operators, and K_f, K_s, K_m are all finite-element stiffness matrices. A_u and A_p are discretizations of the traction-matching condition.

3 Two-Level Newton and Schwarz Methods

At each timestep we have to solve the nonlinear system (4). We solve this nonlinear system with a two-level Newton-Krylov-Schwarz algorithm – it is this algorithm that is the heart of this paper and the key to achieving parallel scalability and performance.

We first obtain an initial guess for the fine nonlinear problem (4) by solving an analogous nonlinear problem on the coarse grid and interpolating to the fine grid. Then we solve (4) with a standard inexact Newton method using that initial guess. At each iteration of Newton's method, we need to solve a linear Jacobian system, for which we use restarted flexible GMRES. The Newton method and the linear solver are standard, so we go on to describe the two-level hybrid preconditioner in detail.

Our hybrid preconditioner has two pieces – a one-level Schwarz domain decomposition component, and a coarse solve component. We describe the domain decomposition part first.

The domain decomposition portion, or fine-grid portion, of the two-level preconditioner is implemented as in [2]. We first partition the finite element mesh on Ω into several meshes on subdomains Ω_ℓ, each one corresponding to a processor of the parallel machine. Then we extend each subdomain Ω_ℓ to overlap its neighbors by a user-specified amount δ, and denote the overlapping domain by $\Omega\,'_\ell$. On each subdomain Ω'_ℓ we construct a subdomain operator B_ℓ, which is a restriction of the Jacobian matrix from the Newton solver. This portion of the preconditioner can be written

$$M_1^{-1} = \sum_{j=1}^{N} (R_j^0)^T B_j^{-1} R_j^\delta \tag{7}$$

where the B_j^{-1} are subdomain solves and the R_j are restriction and interpolation operators for the subdomains. Here R_ℓ^0 is a restriction operator that does not include overlap while R_ℓ^δ includes the overlap, which describes the restricted additive Schwarz variant [3]. The choice of subdomain solves and restriction and interpolation operators leads to different kinds of Schwarz preconditioners with different properties [9]. In our algorithm the $\tilde{B}_\ell^{-1}$ solves are done with LU factorization, which is expensive, but since the subdomain solve is local to a single processor the preconditioner is scalable.

In our implementation, we combine the coarse-level and fine-level preconditioners multiplicatively, while continuing to use additive Schwarz within the fine level. You can write down the application of this hybrid preconditioner M_h^{-1} to a vector x in two steps

$$z = I_H^h B_0^{-1} I_h^H x, \tag{8}$$

$$M_h^{-1} x = z + M_1^{-1}(x - G'_h z) = z + \sum_{j=1}^{N} (R_j^0)^T B_j^{-1} R_j^\delta (x - G'_h z), \tag{9}$$

where I_h^H is a restriction from the fine grid to the coarse grid, $I_H^h = (I_h^H)^T$ is the corresponding interpolation operator from coarse grid to fine grid, and B_0^{-1} is a coarse-grid solve. In this hybrid preconditioner, the additive one-level component (9) means we can do the local subdomain solves in parallel, while we do the coarse and fine levels sequentially.

The coarse solve B_0^{-1} in (8) is normally parallel restarted GMRES, preconditioned with a one-level additive Schwarz method, using the same number of subdomains (and therefore processors) as the fine grid. The matrix that is being used in GMRES here is a Jacobian matrix, constructed independently on the coarse grid. That is, we solve (8) using the one-level algorithm described above. The only difference is that we can solve the coarse problem with a much larger error tolerance than the fine problem, saving computational cost while still being an effective preconditioner.

Using the same basic algorithm for the one-level method on the coarse as on the fine grid has two advantages. First, it is simpler to implement and allows us to reuse some data structures. And second, since we are using the full parallel collective to solve the coarse problem, it allows us to apply the preconditioner multiplicatively, since the coarse solve is done before the fine solve needs any data from it and vice versa. One potential disadvantage is the large number of subdomains of the coarse space, which could lead to the same ill-conditioning problem that drove us to use a two-level method in the first place. In practice, the coarse problem is easy enough to solve and the overlap (which is less costly to increase on the coarse grid) can be made sufficiently large to overcome this difficulty, though for very large simulations we may want to consider additional levels.

The fine and coarse grids in our implementation do not have any necessary connection to each other – they can be generated completely independently by mesh-generating software, and the interpolation and restriction between them is calculated when the program runs. In particular, the fine grid is not a refinement of the coarse grid. The fine grid is partitioned for the domain decomposition and parallel processing by Parmetis [7], and the coarse grid inherits that partition – the elements of the coarse grid are assigned to processors that contain nearby fine-grid elements.

4 Numerical Results

In this section we explore the implications of using a two-level Newton-Krylov-Schwarz method and the interplay of various parameters in that method, comparing to the one-level implementation as we go. We do simulations on a straight tube model, where we can verify results found in the literature and more carefully control the mesh size and number of unknowns, and also consider a more realistic branching artery model derived from clinical data. See [2] for a detailed verification of the same method with a less efficient preconditioner.

In the numerical results in this section, unless otherwise specified, we use an incompressible structure, the fluid density is $1{,}000\ \mathrm{kg/m}^3$, the damping parameter $\beta = 0.01$, and the kinematic viscosity of the fluid is $\nu_f = 0.0035$ kg/m s.

For the solver parameters, we consider the Newton solver on the fine level to have converged if the (absolute) residual is less than 10^{-6}. For fGMRES on the fine level, we have a relative tolerance that changes at each iteration, set by the Eisenstat–Walker method [5]. We restart flexible GMRES every 100 iterations.

We first test the method on a straight tube problem taken from [1]. We have 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 Young's modulus $E_s = 7.5 \cdot 10^4\ \mathrm{kg/m\ s}^2$, the structure is incompressible and has a density of $1{,}100\ \mathrm{kg/m}^3$, and the inlet pressure pulse takes the form $\sigma_f \cdot \mathbf{n}_f = (-P_0/2)\,[1 - \cos((\pi t)/(.0025\mathrm{s}))]$ where $P_0 = 2.0 \cdot 10^5\ \mathrm{kg/m\ s}^2$. The timestep size is $\Delta t = 0.0001$ s.

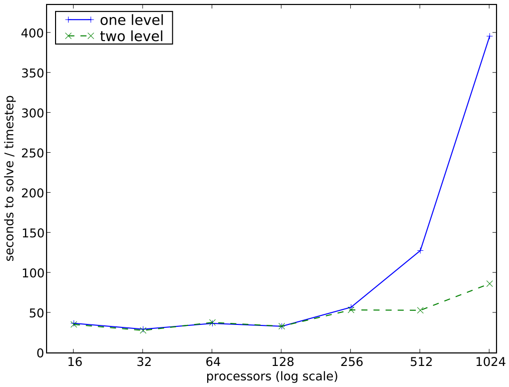

Fig. 1. Weak scaling for a straight tube test problem. The vertical axis shows average walltime in seconds per timestep of the simulation. The number of unknowns is proportional to the number of processors – 1,024 processors is $7.1 \cdot 10^6$ unknowns.

The primary motivation for the two-level preconditioner is to improve scalability for the most physically realistic cases, and we demonstrate that scalability in Fig. 1, which shows weak scaling for the straight tube example in the two-grid case, and where the scalability looks very good out to 1,024 processors. The linear iterations are also kept nearly constant for the two-level case in sharp contrast to the one-level preconditioner (results not shown).

Table 1. Effect of the coarse grid size on the solver behavior for the straight tube case. Coarse size is expressed as a fraction of the number of fine-grid unknowns, and coarse frac represents the proportion of compute time spent on the coarse grid.

Unknowns	np	Coarse size	Levels	fGMRES	Coarse frac	Walltime
$4.51 \cdot 10^5$	64	0.0	One	74.6	0.00	46.21
$4.51 \cdot 10^5$	64	0.03	Two	53.1	0.04	46.33
$4.51 \cdot 10^5$	64	0.12	Two	43.0	0.13	46.84
$7.97 \cdot 10^5$	128	0.0	One	123.2	0.00	41.08
$7.97 \cdot 10^5$	128	0.02	Two	86.9	0.06	42.87
$7.97 \cdot 10^5$	128	0.07	Two	68.7	0.11	43.49
$1.78 \cdot 10^6$	256	0.0	One	313.0	0.00	66.07
$1.78 \cdot 10^6$	256	0.01	Two	205.5	0.06	67.74
$1.78 \cdot 10^6$	256	0.03	Two	209.5	0.12	71.16
$3.16 \cdot 10^6$	512	0.0	One	882.7	0.00	78.27
$3.16 \cdot 10^6$	512	0.004	Two	$1.52 \cdot 10^3$	0.15	143.82
$3.16 \cdot 10^6$	512	0.02	Two	325.6	0.15	66.38
$3.16 \cdot 10^6$	512	0.04	Two	485.8	0.24	83.74
$7.09 \cdot 10^6$	1,024	0.0	One	$5.55 \cdot 10^3$	0.00	426.07
$7.09 \cdot 10^6$	1,024	0.02	Two	522.3	0.15	131.13
$7.09 \cdot 10^6$	1,024	0.03	Two	$4.17 \cdot 10^3$	0.38	548.94

Perhaps the most important implementation detail to consider in designing a two-level method is to choose the size of the coarse grid in order to balance the improvement in conditioning that comes from using a relatively fine coarse grid with the cost of solving the problem on the coarse grid. In Table 1, we present some comparisons of different coarse grid sizes.

In addition to the straight tube problem, we also use a pulmonary artery model taken from clinical data. Here we use a Young's modulus of $E_s = 3.0 \cdot 10^4$ kg/m s^2, and the structure is again incompressible and has a density of $1{,}000$ kg/m^3. We start the simulation from rest, with an impulsive Dirichlet inlet velocity condition of 0.05 m/s. In this more physically realistic and computationally challenging example, the difference in linear iteration counts between one- and two-level methods is even more marked. In Table 2, the two-level method results in a very sharp reduction in linear iterations and a good reduction in compute time for these problems. The two-level method can also be shown to be more robust to a variety of physical parameters.

Table 2. Solver characteristics for increasing number of subdomains, with fixed problem size (1.63 million unknowns) and fixed overlap parameter ($\delta = 0$ for two-level, $\delta = 3$ for one-level).

	fGMRES Iterations		Walltime	
Subdomains	one-level	two-level	one-level	two-level
96	442	237	270	184
112	514	245	277	182
128	487	286	216	163
160	697	282	201	105
192	899	485	168	109
224	1,040	349	152	91.1
256	1,020	382	127	79.9

Table 3. Overlap parameter comparisons for one-level and two-level methods on a branching grid.

Unknowns	np	Levels	δ	Newton	fGMRES	Walltime
$1.63 \cdot 10^6$	128	One	1	3.0	$2.35 \cdot 10^3$	406.32
$1.63 \cdot 10^6$	128	One	2	3.0	820.6	270.19
$1.63 \cdot 10^6$	128	One	3	3.0	487.4	214.43
$1.63 \cdot 10^6$	128	One	4	3.0	356.6	225.61
$1.63 \cdot 10^6$	128	Two	0	3.0	241.2	137.86
$1.63 \cdot 10^6$	128	Two	1	3.0	261.4	186.48
$1.63 \cdot 10^6$	128	Two	2	3.0	225.2	210.11
$1.63 \cdot 10^6$	128	Two	3	3.0	201.4	193.15
$1.63 \cdot 10^6$	128	Two	4	3.0	180.2	210.68
$2.40 \cdot 10^6$	256	One	2	3.0	$3.16 \cdot 10^3$	340.23
$2.40 \cdot 10^6$	256	One	3	3.0	$1.57 \cdot 10^3$	240.06
$2.40 \cdot 10^6$	256	One	4	3.0	$1.02 \cdot 10^3$	207.86
$2.40 \cdot 10^6$	256	Two	0	3.0	423.2	114.98
$2.40 \cdot 10^6$	256	Two	1	3.0	413.4	135.23
$2.40 \cdot 10^6$	256	Two	2	3.0	338.2	148.22
$2.40 \cdot 10^6$	256	Two	3	3.0	435.6	179.23
$2.40 \cdot 10^6$	256	Two	4	3.0	433.2	194.31

The overlap parameter δ in the Schwarz domain decomposition method is one way to adjust the strength of the preconditioner – a higher δ means more information transfer between subdomains, and therefore a faster convergence, but results in larger local problems. Another way to exchange information between subdomains is with a coarse grid, and in Table 3 it is clear that in the two-level method, the need to use overlap is greatly reduced.

5 Conclusion

In this paper we have developed and analyzed two-level Newton-Krylov-Schwarz methods for fluid-structure interaction in the simulation of blood flow. We have demonstrated effective, scalable parallel preconditioners for the fully coupled monolithic problem that allow complicated geometries with realistic parameter values.

References

1. S. Badia, A. Quaini, and A. Quarteroni. Splitting methods based on algebraic factorization for fluid-structure interaction. *SIAM J. Sci. Comput.*, 30(4):1778–1805, 2008.
2. A.T. Barker and X.-C. Cai. Scalable parallel methods for monolithic coupling in fluid-structure interaction with application to blood flow modeling. *J. Comput. Phys.*, 229: 642–659, 2010.
3. X.-C. Cai and M. Sarkis. A restricted additive Schwarz preconditioner for general sparse linear systems. *SIAM J. Sci. Comput.*, 21:792–797, 1999.
4. 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(42–44):4506–4527, 2005.
5. S.C. Eisenstat and H.F. Walker. Choosing the forcing terms in an inexact Newton method. *SIAM J. Sci. Comput.*, 17:16–32, 1996.
6. L. Fatone, P. Gervasio, and A. Quarteroni. Multimodels for incompressible flows. *J. Math. Fluid Mech.*, 2(2):126–150, 2000.
7. G. Karypis. Metis/Parmetis web page, University of Minnesota, 2008. http://glaros.dtc.umn.edu/gkhome/views/metis.
8. C.A. Taylor and M.T. Draney. Experimental and computational methods in cardiovascular fluid mechanics. *Ann. Rev. Fluid Mech.*, 36:197–231, 2004.
9. A. Toselli and O. Widlund. *Domain Decomposition Methods—Algorithms and Theory*. Springer, Berlin, 2005.
10. A.M. Winslow. Adaptive-mesh zoning by the equipotential method. Technical Report, Argonne National Laboratory, 1981.