Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational Physics 223 (2007) 10–49
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

An adaptive, formally second order accurate version of the immersed boundary method

Boyce E. Griffith [a,*], Richard D. Hornung [b], David M. McQueen [a], Charles S. Peskin [a]

[a] Courant Institute of Mathematical Sciences, New York University, 251 Mercer Street, New York, NY 10012, USA
[b] Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, P.O. Box 808, L-561, Livermore, CA 94551, USA

Received 7 November 2005; received in revised form 30 May 2006; accepted 22 August 2006
Available online 12 October 2006

Abstract

Like many problems in biofluid mechanics, cardiac mechanics can be modeled as the dynamic interaction of a viscous incompressible fluid (the blood) and a (visco-)elastic structure (the muscular walls and the valves of the heart). The immersed boundary method is a mathematical formulation and numerical approach to such problems that was originally introduced to study blood flow through heart valves, and extensions of this work have yielded a three-dimensional model of the heart and great vessels. In the present work, we introduce a new adaptive version of the immersed boundary method. This adaptive scheme employs the same hierarchical structured grid approach (but a different numerical scheme) as the two-dimensional adaptive immersed boundary method of Roma et al. [A multilevel self adaptive version of the immersed boundary method, Ph.D. Thesis, Courant Institute of Mathematical Sciences, New York University, 1996; An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534] and is based on a formally second order accurate (i.e., second order accurate for problems with sufficiently smooth solutions) version of the immersed boundary method that we have recently described [B.E. Griffith, C.S. Peskin, On the order of accuracy of the immersed boundary method: higher order convergence rates for sufficiently smooth problems, J. Comput. Phys. 208 (1) (2005) 75–105]. Actual second order convergence rates are obtained for both the uniform and adaptive methods by considering the interaction of a viscous incompressible flow and an anisotropic incompressible viscoelastic shell. We also present initial results from the application of this methodology to the three-dimensional simulation of blood flow in the heart and great vessels. The results obtained by the adaptive method show good qualitative agreement with simulation results obtained by earlier non-adaptive versions of the method, but the flow in the vicinity of the model heart valves indicates that the new methodology provides enhanced boundary layer resolution. Differences are also observed in the flow about the mitral valve leaflets.

MSC: 65M06; 65M50; 74S20; 76D05; 92C10; 92C35

Keywords: Immersed boundary method; Fluid–structure interaction; Adaptive mesh refinement; Projection method; Convergence; Cardiac mechanics; Blood flow; Hemodynamics

* Corresponding author. Tel.: +1 212 998 3212; fax: +1 212 995 4121.
E-mail addresses: griffith@cims.nyu.edu (B.E. Griffith), hornung@llnl.gov (R.D. Hornung), mcqueen@cims.nyu.edu (D.M. McQueen), peskin@cims.nyu.edu (C.S. Peskin).

0021-9991/$ - see front matter
doi:10.1016/j.jcp.2006.08.019

1. Introduction

Many problems in biofluid mechanics can be modeled as the dynamic interaction of a viscous incompressible fluid and a (visco-)elastic structure. One example is cardiac mechanics. In the approach of Peskin and McQueen [4–9], the blood is modeled as a viscous incompressible fluid, whereas the muscular heart wall is modeled as a thick viscoelastic structure with time-dependent elastic parameters, and the flexible heart valve leaflets are modeled as thin elastic boundaries. The immersed boundary method is a mathematical formulation and numerical approach to such problems originally introduced by Peskin to study blood flow through heart valves [10,11]. In the immersed boundary formulation of problems involving the interaction of a viscous incompressible fluid and an incompressible elastic or viscoelastic structure, the configuration of the elastic structure is described by Lagrangian variables (i.e., variables indexed by a coordinate system attached to the elastic structure), whereas the momentum, velocity, and incompressibility of the coupled fluid–structure system are described by Eulerian variables (i.e., in reference to fixed physical coordinates). In the continuous equations of motion, these two descriptions are connected by making use of the Dirac delta function, whereas a smoothed approximation to the delta function is used to link the Lagrangian and Eulerian descriptions when the continuous equations are discretely approximated for computer simulation.

Simulating fluid–structure interaction by the immersed boundary method requires the use of high spatial resolution; however, in many cases, this requirement is somewhat localized to the flow in the neighborhood of the immersed boundaries [1,2]. For the flow away from the immersed boundaries, the need for high spatial resolution is somewhat lessened, although it may be needed in regions of high vorticity, e.g., in the neighborhood of vortices that have been shed from the boundaries and have subsequently moved away from the boundaries into the interior of the flow. If a uniform grid is employed to discretize the (Eulerian) equations of motion for such simulations, the fine grid spacing required to resolve the flow near the immersed boundaries is necessarily employed throughout the entire computational domain, even in regions that may not require such high resolution. By employing an adaptive discretization of the equations of motion, high spatial resolution can be deployed locally where it is most needed, whereas comparatively coarse resolution can be employed where it suffices. In principle, such an adaptive scheme would allow for more efficient utilization of computational resources when compared to non-adaptive strategies, although realizing such gains in practice requires the careful design and implementation of a number of algorithms and data structures.

An adaptive version of the immersed boundary method was first introduced in the Ph.D. thesis of Roma [1] and the subsequent work of Roma et al. [2]. In this earlier work, the hierarchical structured grid approach of Berger and Oliger [12] and Berger and Colella [13] was employed to introduce local spatial refinement in the Eulerian grid in the vicinity of an immersed elastic interface. The simulated dynamics produced by this adaptive version of the immersed boundary method were demonstrated to be virtually identical to those obtained by a non-adaptive method that employed a uniform grid with the same spatial resolution as that of the finest grid level in the adaptive computation. That is to say, despite the fact that the adaptive computation *only* deployed high spatial resolution in a localized region about the elastic interface, the adaptive results were not significantly different from those obtained by a non-adaptive method that employed a uniformly fine Cartesian grid.

In the present work, we describe a *new* adaptive version of the immersed boundary method for problems of fluid–structure interaction, provide empirical convergence results that demonstrate the accuracy of this method for a two-dimensional model problem, and present initial results from the application of this adaptive methodology to the three-dimensional simulation of cardiac fluid mechanics. The present adaptive method is based upon a non-adaptive, formally second order accurate version of the immersed boundary method recently described by Griffith and Peskin [3]. Like the uniform grid algorithm upon which it is based, this new adaptive version of the immersed boundary method is formally second order accurate in the sense that the method is expected to converge at its formal order of accuracy only for problems that possess sufficiently smooth solutions. The present adaptive algorithm employs the same hierarchical structured grid approach (but a different numerical scheme, see below) as that used by Roma, Peskin, and Berger to discretize the Eulerian equations of motion (i.e., the incompressible Navier–Stokes equations). Unlike the method of Roma et al., the present algorithm employs a fully explicit treatment of the Lagrangian equations of motion (i.e., the equations that specify the evolution of the configuration of the elastic structure). In particular, in an

attempt to reduce the occurrence of non-physical oscillations in the computed dynamics, we employ a strong stability-preserving Runge–Kutta method [14] for the time integration of the Lagrangian equations of motion. The present method differs more dramatically from the approach of Roma et al. in the details of its treatment of the Eulerian equations of motion, namely the incompressible Navier–Stokes equations. Although both adaptive schemes employ projection methods to solve the incompressible Navier–Stokes equations, the present work employs a cell-centered projection method that makes use of an implicit L-stable discretization of the viscous terms [15,16] and a second order Godunov method for the explicit treatment of the nonlinear advection terms [17–20]. Generally speaking, projection methods [21–23] are a class of fractional step algorithms for incompressible flow problems that update the velocity by first solving the momentum equation over a time interval without imposing the constraint of incompressibility. Doing so yields an intermediate velocity field that is generally not divergence free. The true updated velocity is then obtained by solving a Poisson problem to enforce the incompressibility constraint. More abstractly, this process *projects* the intermediate velocity onto the space of divergence free vector fields.

In an "exact" projection method, the discrete divergence of the updated velocity is identically zero (or zero to within the tolerance of the linear solver in practice). Even on uniform grids, however, exact cell-centered projections present difficulties. For example, on a periodic grid with an even number of grid cells in each coordinate direction, an exact projection operator possesses a non-trivial nullspace that causes the pressure to decouple into 2^d subfields, where d is the number of spatial dimensions. To date, it appears that exact cell-centered projections have not been successfully implemented for co-located cell-centered velocities defined on hierarchically composed locally refined grids (i.e., such as those used in the present work). Like most recent projection methods for locally refined grids [19,20,24], the present scheme employs a projection method that is not exact but rather is "approximate" in the sense that the discrete divergence of the velocity only *converges* to zero at a second order rate as the *composite* computational grid is refined. (Note that unlike exact projection methods, approximate projection methods typically yield a fully coupled pressure field on both uniform and locally refined grids.) When such methods are used with the immersed boundary method, we have found that it is beneficial to determine the updated velocity and pressure in terms of the solutions to two *different* approximate projection equations at each timestep. This so-called hybrid approach was originally proposed by Almgren et al. for simulating inviscid incompressible flow [25]. Our hybrid projection algorithm, which we first detailed in the non-adaptive context [3], is essentially an extension of their inviscid hybrid method ("version 5") to the viscous case. The adaptive version of our projection method is also similar to earlier adaptive inviscid schemes described by Minion [19] and Martin and Colella [20]; however, note that we employ a somewhat different discretization at interfaces in grid resolution. In particular, for two-dimensional locally refined grids, we employ finite difference discretizations of the gradient and Laplace operators that were introduced by Ewing et al. [26], and in three spatial dimensions, we make use of a straightforward generalization of their approach. Note, however, that Ewing et al. consider only the issue of the accurate discretization of second-order elliptic equations on locally refined grids, not the solution of the incompressible Navier–Stokes equations. To our knowledge, this is the first application of their discretization approach to the simulation of incompressible flows. We believe that this particular treatment is more easily implemented than the approaches of e.g. [19,20,24]. Moreover, this treatment also appears to yield a globally second order accurate projection method [27]. As this approach could be useful in other application areas that require the adaptive simulation of incompressible flows, we include a complete description of the interpolation and finite difference operators that we employ in our cell-centered adaptive projection method for both two- and three-dimensional locally refined grids.

As in our earlier uniform grid study [3], *actual* second order numerical convergence rates are observed when our adaptive immersed boundary method is used to simulate the interaction of a viscous incompressible fluid and a viscoelastic shell (i.e., a body which, although thin, is not infinitely thin). As was first done in [3], we again consider the accuracy of the immersed boundary method for anisotropic incompressible viscoelastic shells with two sets of elastic properties. In the first case, the stiffness of the shell tapers to zero at its edges, so that there is a *continuous* transition in material properties between the fluid and the structure. We also consider the case in which the stiffness of the shell is constant, so that there is a *sharp* discontinuity in the material properties of the coupled system at the fluid–structure interface. At least for the moderate Reynolds number flows considered here, the dynamics generated by the adaptive scheme are virtually identical to those gener-

ated by a non-adaptive method which employs a uniformly fine Cartesian grid. Moreover, for each set of material properties considered, the true solution appears to be sufficiently regular for the adaptive method to converge at or near its formal order of accuracy as the computational grids are refined. (Although the present version of the immersed boundary method has not been validated for standard fluid–structure interaction problems such as flow past a cylinder, note that such a validation study has been performed by Lai and Peskin with a similar formally second order accurate (but non-adaptive) method [28,29]. In particular, note that this earlier version of the immersed boundary method was found to generate the experimentally correct Strouhal number over a range of Reynolds numbers in which St is in fact varying as a function of Re.)

In addition to convergence results in two spatial dimensions, we also present initial results from the application of this adaptive methodology to McQueen and Peskin's three-dimensional model of cardiac mechanics. The results obtained by the adaptive method show good qualitative agreement with results obtained by earlier versions of the immersed boundary method [7–9]. In particular, in the present simulation as in earlier simulations, a prominent vortex is shed from the mitral valve leaflets and migrates to the interior of the left ventricle prior to ventricular systole (i.e., ventricular contraction). Similarly, as was observed in earlier simulations, a prominent right-ventricular vortex is shed from the tricuspid valve leaflets prior to ventricular systole. There are also notable differences, however, between the results obtained by the present version of the immersed boundary method and those obtained by earlier versions of the method. For instance, the flow in the vicinity of the model heart valves indicates that the new methodology provides dramatically enhanced boundary layer resolution. Differences are also observed in the flow in the vicinity of the mitral valve. In the present simulation, there is an additional left-ventricular vortex that swirls about the jet of inflow from the mitral valve prior to ventricular contraction. This particular feature of the flow was not observed in earlier simulations, and its physiological significance, if any, is presently unknown. We also present parallel code timing results that demonstrate that our adaptive strategy substantially reduces the computing resources required to simulate cardiac fluid mechanics.

2. The continuous equations of motion

Consider a system comprised of a viscoelastic structure immersed in a viscous incompressible fluid. We assume that the fluid has uniform density, ρ, and uniform dynamic viscosity, μ. The structure is taken to be incompressible and neutrally buoyant, and the viscous properties of the structure are assumed to be those of the fluid in which it is immersed. Consequently, the momentum, velocity, and incompressibility of the coupled system can be described by the incompressible Navier–Stokes equations, augmented by an appropriately defined body force. (Even in the more complicated case in which the mass density of the structure differs from that of the fluid, the momentum, velocity, and incompressibility of the coupled system can still be described by the incompressible Navier–Stokes equations; see [30–32]. The case in which the viscosity of the structure differs from that of the fluid can also presumably be done by a generalization of the methods proposed here, but this has not yet been attempted.)

The immersed boundary formulation of this problem employs an Eulerian description of the velocity and incompressibility of the fluid–structure system and a Lagrangian description of the configuration of the immersed elastic structure. In particular, the velocity of the entire coupled system is described in terms of an Eulerian velocity field, $\mathbf{u}(\mathbf{x},t)$, where $\mathbf{x}=(x,y,z)$ are fixed physical (Cartesian) coordinates, whereas the configuration of the immersed elastic structure is described in terms of a curvilinear coordinate system. (It is important to emphasize that $\mathbf{u}(\mathbf{x},t)$ refers to the velocity of *whichever* material is physically located at position $\mathbf{x}$ at time t. The same will be true for all of the Eulerian variables, including the pressure and the (Cartesian) elastic force density.) Let (q,r,s) be material curvilinear coordinates attached to the elastic structure so that fixed values of (q,r,s) label a material point for all time t, with $\mathbf{X}(q,r,s,t)$ referring to the Cartesian position of such a material point at time t. The physical domain consists of a region $U \subset \mathbb{R}^3$. For simplicity, we presently take U to be the unit cube and impose periodic boundary conditions. The curvilinear coordinates are restricted to some region of (q,r,s)-space, here denoted $\Omega \subset \mathbb{R}^3$. The configuration of the elastic structure at time t is denoted by $\mathbf{X}(\cdot,\cdot,\cdot,t)$, and the curvilinear force density (i.e., the density with respect to (q,r,s)) generated by the elasticity of the structure is determined by a possibly time-dependent mapping from $\mathbf{X}(\cdot,\cdot,\cdot,t)$, the structure configuration at time t, to the elastic force density at time t, denoted $\mathbf{F}(\cdot,\cdot,\cdot,t)$.

The equations of motion for the system can be written in the following form:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\nabla)\mathbf{u}\right) = -\nabla p + \mu\nabla^2\mathbf{u} + \mathbf{f}, \tag{1}$$

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

$$\mathbf{f}(\mathbf{x},t) = \int_\Omega \mathbf{F}(q,r,s,t)\,\delta(\mathbf{x}-\mathbf{X}(q,r,s,t))\,dq\,dr\,ds, \tag{3}$$

$$\frac{\partial \mathbf{X}}{\partial t}(q,r,s,t) = \mathbf{u}(\mathbf{X}(q,r,s,t),t), \quad t = 0 \;\Rightarrow\; \int_U \mathbf{u}(\mathbf{x},t)\,\delta(\mathbf{x}-\mathbf{X}(q,r,s,t))\,d\mathbf{x}, \tag{4}$$

$$\mathbf{F}(\cdot,\cdot,\cdot,t) = \mathscr{F}_t[\mathbf{X}(\cdot,\cdot,\cdot,t),t]. \tag{5}$$

Eqs. (1) and (2) are the incompressible Navier–Stokes equations written in Eulerian form, where $p(\mathbf{x},t)$ is the pressure and $\mathbf{f}(\mathbf{x},t)$ is the (Cartesian) elastic force density. Eq. (5) formalizes the assumption that the curvilinear elastic force density, $\mathbf{F}(\cdot,\cdot,\cdot,t)$, is determined by a possibly time-dependent mapping of the structure configuration, $\mathbf{X}(\cdot,\cdot,\cdot,t)$. (By permitting the mapping $\mathscr{F}$ to be explicitly time-dependent, we allow for the case in which the (active) structure can do net work on the fluid as it moves through a cycle in configuration space. An example of this is the cardiac cycle, in which the heart does net work on the blood during each heartbeat.)

Eqs. (3) and (4) describe the interaction between the Lagrangian and Eulerian variables. In both equations, the three-dimensional Dirac delta function, $\delta(\mathbf{x}) = \delta(x)\delta(y)\delta(z)$, appears as the kernel of an integral transform that facilitates conversions between Eulerian and Lagrangian quantities. Eq. (3) converts the curvilinear force density into the Cartesian force density. Note that the numerical values of the Cartesian and curvilinear elastic force densities are generally not equal at corresponding points. Nevertheless, $\mathbf{f}$ and $\mathbf{F}$ are equivalent *as densities*. Recalling the defining property of the Dirac delta function,

$$\int_V \delta(\mathbf{x}-\mathbf{X})\,d\mathbf{x} = \begin{cases} 1 & \text{if } \mathbf{X}\in V, \\ 0 & \text{otherwise,} \end{cases} \tag{6}$$

where $V \subset U$ is an arbitrary region of physical space, we see that the densities are indeed equivalent via

$$\begin{aligned}\int_V \mathbf{f}(\mathbf{x},t)\,d\mathbf{x} &= \int_V \int_\Omega \mathbf{F}(q,r,s,t)\,\delta(\mathbf{x}-\mathbf{X}(q,r,s,t))\,dq\,dr\,ds\,d\mathbf{x} \\ &= \int_\Omega \mathbf{F}(q,r,s,t)\left(\int_V \delta(\mathbf{x}-\mathbf{X}(q,r,s,t))\,d\mathbf{x}\right)dq\,dr\,ds = \int_{\mathbf{X}^{-1}(V,t)} \mathbf{F}(q,r,s,t)\,dq\,dr\,ds,\end{aligned}$$

where

$$\mathbf{X}^{-1}(V,t) = \{(q,r,s) \mid \mathbf{X}(q,r,s,t)\in V\}. \tag{7}$$

Note that another way to express $\mathbf{f}$ is by

$$\mathbf{f}(\mathbf{X}(q,r,s,t),t) = J(q,r,s)\,\mathbf{F}(q,r,s,t), \tag{8}$$

where $J(q,r,s)$ denotes the Jacobian determinant of the coordinate transformation $(q,r,s)\mapsto\mathbf{X}(q,r,s,t)$. Thus it is easy to see that although $\mathbf{f}$ and $\mathbf{F}$ are equivalent densities, their pointwise values will not generally be equal, i.e., $\mathbf{f}(\mathbf{X}(q,r,s,t),t) \neq \mathbf{F}(q,r,s,t)$. (Note that $J(q,r,s)$ is time-independent as a consequence of the assumption that the material is incompressible.)

The second of the interaction equations, Eq. (4), relates the material velocity of the elastic structure to the Eulerian velocity field for the coupled system. Since $\mathbf{u}(\mathbf{x},t)$ is the velocity of whichever material is physically located at position $\mathbf{x}$ at time t, for any $(q,r,s)\in\Omega$,

$$\frac{\partial \mathbf{X}}{\partial t}(q,r,s,t) = \mathbf{u}(\mathbf{X}(q,r,s,t),t). \tag{9}$$

As long as $\mathbf{u}$ is continuous, we may evaluate the velocity at $\mathbf{X}(q,r,s,t)$ by making use of the delta function,

$$\mathbf{u}(\mathbf{X}(q,r,s,t),t) = \int_U \mathbf{u}(\mathbf{x},t)\,\delta(\mathbf{x}-\mathbf{X}(q,r,s,t))\,d\mathbf{x}. \tag{10}$$

For the coupled system, continuity of the velocity field follows from the presence of viscosity in both the fluid and the structure.

Before concluding this section, we mention the particular elastic force density mapping that is used in the present work. Suppose that the immersed elastic structure consists of a continuous collection of elastic fibers, where the material coordinates (q,r,s) have been chosen so that a fixed value of the pair (q,r) labels a particular fiber for all time. Let $\boldsymbol{\tau}$ denote the unit tangent vector in the fiber direction,

$$\boldsymbol{\tau} = \frac{\partial \mathbf{X}/\partial s}{|\partial \mathbf{X}/\partial s|}. \tag{11}$$

Since the fibers are elastic, the fiber tension, T, is related to the fiber strain, which is determined by $|\partial \mathbf{X}/\partial s|$. The fiber tension can be expressed by a generalized Hooke's law of the form

$$T = \sigma(|\partial \mathbf{X}/\partial s|; q, r, s). \tag{12}$$

One can show [7,30] that the corresponding curvilinear elastic force density can be put in the form

$$\mathcal{F}[\mathbf{X}](\cdot,\cdot,\cdot,t) = \frac{\partial}{\partial s}(T\boldsymbol{\tau}). \tag{13}$$

Since T and $\boldsymbol{\tau}$ are both defined in terms of $\partial \mathbf{X}/\partial s$, $\mathcal{F}$ is a mapping from the structure configuration to the curvilinear force density, $\mathbf{F}(\cdot,\cdot,\cdot,t)$.

The formulation described above allows for some regions of the physical domain to be occupied only by fluid and for other regions to be occupied by viscoelastic material. At least in the present formulation, it is important to note again that many of the properties of the incompressible viscoelastic material, including its density and viscosity, are identical to those of the surrounding fluid. Thus, another way of viewing the foregoing formulation is that the viscoelastic material is an idealized *composite* material with a viscous incompressible fluid component (the properties of which are described in Eulerian form) and an elastic fiber component (the properties of which are described in Lagrangian form). From this point of view, the sole purpose of the fibers is to provide a Lagrangian description of the additional elastic stresses that are present in the viscoelastic material. These stresses are characterized by the elastic force density, $\mathbf{F}(\cdot,\cdot,\cdot,t)$, as a function of the material configuration, $\mathbf{X}(\cdot,\cdot,\cdot,t)$. Note that when written in Eulerian form, the stress tensor of such a composite material is given by

$$\sigma_{ij}^{f} = -p\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + T^{f}\tau_i\tau_j, \tag{14}$$

where p is the pressure, δ_{ij} is the Kronecker delta, μ is the viscosity, $\mathbf{u} = (u_1,u_2,u_3)$ is the velocity, $\mathbf{x} = (x_1,x_2,x_3)$ are (Cartesian) physical coordinates, T^{f} is the fiber tension, and $\boldsymbol{\tau} = (\tau_1,\tau_2,\tau_3)$ is the unit tangent to the fibers. It is important to note that the Eulerian fiber tension, T^{f}, is different from the Lagrangian fiber tension, T, defined in Eq. (12). In particular, $T^{f}(\mathbf{x},t)$ is the elastic force per unit cross-sectional area of composite material, whereas $T(q,r,s,t)\,dq\,dr$ is the force transmitted by $dq\,dr$, a fiber patch. It can be shown that T^{f} and T are related by

$$T^{f}(\mathbf{X}(q,r,s,t),t) = \frac{|\partial \mathbf{X}/\partial s|}{J(q,r,s)}\,T(q,r,s,t), \tag{15}$$

where, as before, $J(q,r,s)$ denotes the Jacobian determinant of the coordinate transformation $(q,r,s) \mapsto \mathbf{X}(q,r,s,t)$. (See [4] for a derivation of this relationship between T^{f} and T.)

3. The discrete equations of motion

In the immersed boundary approach to fluid–structure interaction problems, the solution to the continuous equations of motion, (1)–(5), is approximated by discretizing the Eulerian equations on a Cartesian grid and by discretizing the Lagrangian equations on a discrete lattice in the curvilinear coordinate space. The present work extends the uniform grid method of [3] to provide an adaptive discretization of the Eulerian equations of motion (i.e., the incompressible Navier–Stokes equations). In particular, rather than using a uniform

Cartesian grid to discretize the physical domain, U, we employ a hierarchy of nested Cartesian grids with successively finer mesh spacings. Before describing this locally refined discretization of the Eulerian equations of motion, however, we first summarize the spatial discretization of the Lagrangian equations.

Note that when no local refinement is deployed, the following numerical scheme reduces to the method described in [3]. Rather than providing a summary of the uniform grid scheme, we instead refer the interested reader to [3,27] for a presentation of the non-adaptive methodology. Additionally, note that Sections 3.2–3.4, 3.7, 3.8 provide a rather complete description of the particular adaptive projection method that we have developed. Although this algorithm is described in the context of the immersed boundary method, we expect that it could be useful in other application areas where local refinement is desirable.

3.1. The Lagrangian spatial discretization

The curvilinear coordinate space is discretized on a fixed lattice in (q,r,s)-space with uniform meshwidths $(\Delta q, \Delta r, \Delta s)$. Unless otherwise noted, from now on the curvilinear coordinate indices (q,r,s) will refer to the nodes of the curvilinear computational lattice, so that

$$(q,r,s) = (q_0,r_0,s_0) + (m_q\Delta q, m_r\Delta r, m_s\Delta s) \tag{16}$$

for fixed constants q_0, r_0, and s_0 and integer values of m_q, m_r, and m_s.

For a Lagrangian quantity defined on the curvilinear mesh (i.e., in the present work, $\mathbf{X}$ or $\mathbf{F}$), we employ the notation $\mathbf{X}^n(q,r,s) \equiv \mathbf{X}(q,r,s,t_n)$, where t_n is the time of the nth timestep. The timestep size is implicitly defined by $\Delta t_n = t_{n+1} - t_n$, although we generally employ a fixed uniform timestep Δt. Note that some quantities will be defined at "half-timesteps," $t_{n+\frac{1}{2}} = t_n + \frac{1}{2}\Delta t_n$.

Although the lattice used to discretize the curvilinear coordinate space is fixed throughout a particular simulation, it is important to note that the physical locations of the nodes of the curvilinear mesh, $\mathbf{X}^n(q,r,s) \equiv \mathbf{X}(q,r,s,t_n)$, are free to move throughout the physical domain. In particular, the physical positions of the nodes of the curvilinear mesh are in no way required to conform to the locally refined Cartesian grid described in Section 3.2. By contrast, as we describe below, the Cartesian grid *is* required to adapt to the evolving configuration of the curvilinear mesh since we deploy local refinement in the vicinity of the elastic structure.

Recall that the continuous version of the elastic force density that we employ in the present work is given by Eqs. (11)–(13). To approximate this force density on the curvilinear computational lattice, we introduce a finite difference approximation to differentiation in the s curvilinear coordinate direction, defined by

$$D_s\Psi(q,r,s) = \frac{\Psi\left(q,r,s+\frac{1}{2}\Delta s\right) - \Psi\left(q,r,s-\frac{1}{2}\Delta s\right)}{\Delta s}, \tag{17}$$

where $\Psi(q,r,s)$ is a function defined on the curvilinear computational lattice.

Given a structure configuration, $\mathbf{X}(\cdot,\cdot,\cdot)$, the unit tangent vector, (11), is approximated at "half-integer" multiples of Δs by

$$\boldsymbol{\tau}\left(q,r,s+\frac{1}{2}\Delta s\right) = \frac{D_s\mathbf{X}\left(q,r,s+\frac{1}{2}\Delta s\right)}{\left|D_s\mathbf{X}\left(q,r,s+\frac{1}{2}\Delta s\right)\right|}. \tag{18}$$

Similarly, the fiber tension, (12), is approximated by

$$T\left(q,r,s+\frac{1}{2}\Delta s\right) = \sigma\left(\left|D_s\mathbf{X}\left(q,r,s+\frac{1}{2}\Delta s\right)\right|; q,r,s+\frac{1}{2}\Delta s\right). \tag{19}$$

Finally, Eqs. (18) and (19) may be used to approximate the curvilinear elastic force density, (13), at integer multiples of Δs by

$$\mathbf{F}(q,r,s) = D_s(T\boldsymbol{\tau})(q,r,s). \tag{20}$$

Note that the half-integer multiples of Δs that appear in the foregoing are only intermediate values. In the end, the evaluation of (20) at the nodes of the curvilinear computational lattice requires only the values of $\mathbf{X}(q,r,s)$ at the nodes of the lattice, i.e., for $(q,r,s) = (q_0,r_0,s_0) + (m_q\Delta q, m_r\Delta r, m_s\Delta s)$ for fixed constants q_0, r_0, and s_0 and for integer values of m_q, m_r, and m_s.

3.2. Hierarchical structured Cartesian grids

In the present work, the physical domain, U, is taken to be the periodic unit cube (or, in Section 4, the periodic unit square). The locally refined Cartesian grid that is used to discretize U is composed of the union of rectangular grid *patches* that are organized into a sequence of *patch levels*. We shall frequently refer to the collection of patch levels as the *patch hierarchy*, or simply the hierarchy. The levels are numbered $\ell = 0, \ldots, \ell_{\max}$, where $\ell = 0$ indicates the coarsest level in the hierarchy and $\ell = \ell_{\max}$ indicates the finest level. All of the patches in level ℓ share the same grid spacings, $(\Delta x_\ell, \Delta y_\ell, \Delta z_\ell)$, although for the purposes of the present discussion it suffices to assume that $h_\ell = \Delta x_\ell = \Delta y_\ell = \Delta z_\ell$. The grid spacing on a particular level is not arbitrary; instead, the grid spacing at level $\ell + 1$ is required to be an integer factor $r > 1$ finer than the grid spacing at level ℓ, so that $h_{\ell+1} = h_\ell / r$. Although typical choices for this *refinement ratio* are $r = 2$ or 4, in the present scheme any integer $r > 1$ may be employed as the refinement ratio. (In fact, in the implementation of the adaptive method, the refinement ratio is neither fixed across the entire patch hierarchy nor required to be isotropic. These slight generalizations of the presented method are easily implemented in practice but do not seem sufficiently important to warrant the additional notation required by their description.) The centers of the Cartesian grid cells on level ℓ are the points $\mathbf{x}_{i,j,k} = ((i + \frac{1}{2})h_\ell, (j + \frac{1}{2})h_\ell, (k + \frac{1}{2})h_\ell)$, although it is important to note that in general only patch level 0 completely covers the physical domain. Thus, on a locally refined grid, the level ℓ grid cells are a subset of the cells of a uniform discretization of the physical domain with the same resolution as level ℓ.

The patch levels are required to be properly nested in the sense that the union of the grid patches at level $\ell + 1$ must be strictly contained in the union of the patches at level ℓ. That is, the union of the level ℓ patches must be large enough to provide at least a one cell wide buffer of unrefined level ℓ grid cells around the union of the level $\ell + 1$ patches. Note that this *is not* equivalent to requiring that *each* level $\ell + 1$ patch be contained (strictly or otherwise) within a single level ℓ patch. The nesting requirement is typically relaxed at domain boundaries with prescribed physical boundary conditions (but not at periodic boundaries). Fig. 1 displays a locally refined grid that satisfies the proper nesting condition.

The patch hierarchy is constructed, either at the initial time or at a later point in the computation (i.e., during adaptive regridding), by a simple recursive procedure, as follows. The coarsest level, namely level 0, consists of one or more grid patches whose union completely covers the physical domain, U. Next, having constructed levels $0, \ldots, \ell < \ell_{\max}$, grid cells on level ℓ are tagged for further refinement according to criteria described below, thereby identifying the portion of level ℓ that requires still higher spatial resolution. These tagged cells are grouped together into rectangular grid patches by the Berger–Rigoutsos clustering algorithm [33]. The level ℓ boxes generated by the clustering algorithm are subsequently refined by the refinement ratio, r, to form the new level $\ell + 1$ patches. (Note that a consequence of this construction is that fine level $\ell + 1$ grid *patch* boundaries align with coarse level ℓ grid *cell* boundaries. This property simplifies interlevel data

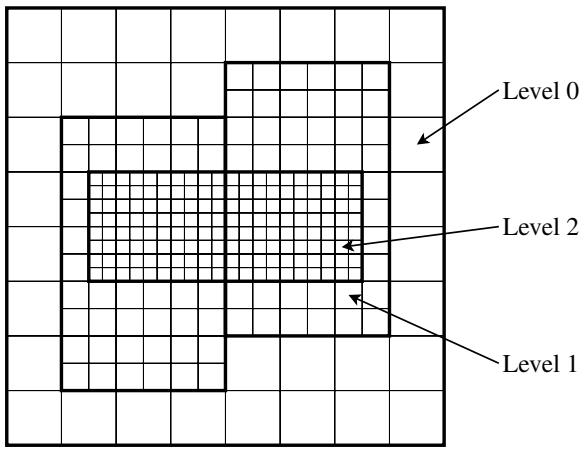

Fig. 1. A properly nested hierarchical structured locally refined Cartesian grid. Patch boundaries are indicated by bold lines. Each level in the patch hierarchy consists of one or more rectangular grid patches, and the levels satisfy the proper nesting condition. Here, the refinement ratio is $r = 2$.

communication as well as the development of numerical methods on the locally refined grid.) This process is repeated until the specified maximum number of levels have been generated. Along the way, care must be taken to ensure that the patch levels satisfy the proper nesting condition. It may also be necessary to further modify the generated boxes to achieve good load balancing and communications efficiency in parallel computational environments, but such details are somewhat beside the focus of the present work (see e.g. [27] for a discussion of one possible route to parallelizing the immersed boundary method).

During the initial construction and subsequent regriddings of the patch hierarchy, grid cells are tagged for refinement when they contain one or more curvilinear mesh nodes. More precisely, cell (i,j,k) on level $\ell < \ell_{\max}$ is tagged for refinement if there exists a curvilinear mesh node (q,r,s) such that $\mathbf{X}^n(q,r,s) \in [ih_\ell,(i+1)h_\ell]\times[jh_\ell,(j+1)h_\ell]\times[kh_\ell,(k+1)h_\ell]$. A consequence of this tagging criteria is that the elastic structure is embedded in the finest level of the patch hierarchy. (A generalization that we do not consider here is to assign portions of the elastic structure to levels other than the finest one.) Additional grid cells are tagged for refinement on level $\ell_{\max} - 1$ both to prevent the structure from escaping the finest level of the patch hierarchy between regridding operations, and to ensure that the structure configuration is sufficiently far from the coarse-fine interface between levels $\ell_{\max} - 1$ and $\ell_{\max}$ to avoid complicating the discretization of the Lagrangian–Eulerian interaction equations. In particular, when velocity interpolation and force spreading are performed via a regularized delta function with a support of d meshwidths in each coordinate direction (see Section 3.5), we ensure that the physical position of each node of the curvilinear mesh is at least $\lceil d/2 \rceil + 1$ grid cells away from the nearest coarse-fine interface on level $\ell_{\max}$. Additional cells may be tagged for refinement according to feature detection criteria (e.g., based on the local magnitude of the vorticity) or other user-defined error estimators.

For a cell-centered quantity $\psi(\mathbf{x},t)$ defined on the composite Cartesian grid, we employ the notation $\psi^n_{i,j,k} \equiv \psi(\mathbf{x}_{i,j,k}, t_n)$, where t_n is the time of the nth timestep. (Recall that the timestep size is implicitly defined by $\Delta t_n = t_{n+1} - t_n$.) Note that some quantities are defined at half-timesteps, $t_{n+\frac{1}{2}} = t_n + \frac{1}{2}\Delta t_n$. In the present algorithm, the velocity, $\mathbf{u}^n_{i,j,k}$, pressure, $p^{n+\frac{1}{2}}_{i,j,k}$, and Cartesian elastic force density, $\mathbf{f}^n_{i,j,k}$, are all cell-centered quantities. The velocity and force density are defined at integer multiples of Δt, whereas the pressure is defined at half-timesteps (Fig. 2).

The Godunov procedure used to approximate the nonlinear advection term that appears in the momentum equation makes use of Eulerian quantities described at the *centers* of the Cartesian grid cells as well as quantities described at the cell *faces*. We shall also make use of both cell-centered and face-centered quantities when defining the various Cartesian grid interpolation operators and finite difference approximations to the spatial differential operators for locally refined grids. If $\psi(\mathbf{x},t)$ is defined on the faces of the Cartesian grid cells, we

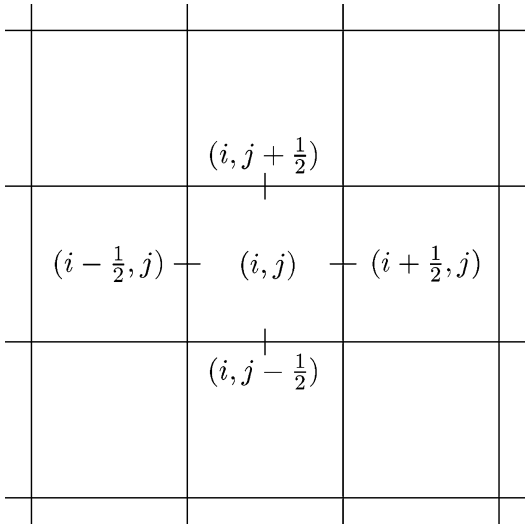

Fig. 2. Locations of cell-centered and face-centered quantities about Cartesian grid cell (i,j) for a two-dimensional grid. Placement on a three-dimensional grid is analogous.

employ the notation $\psi^n_{i\square\frac{1}{2},j,k} \square \psi\eth\mathbf{x}_{i\square\frac{1}{2},j,k}, t_n\flat$ to indicate the evaluation of ψ on the x-faces of the grid, i.e., at the points $\mathbf{x}_{i\square\frac{1}{2},j,k} \frac{1}{4} \eth ih, \eth j \flat \frac{1}{2}\flat h, \eth k \flat \frac{1}{2}\flat h\flat$. Evaluation of ψ on the y- and z-faces is denoted similarly.

By convention, a vector field defined on the Cartesian grid in terms of those vector components that are normal to the faces of the grid cells is called a MAC vector field [34]. That is to say, if $\mathbf{u}^{\mathrm{MAC}} = (u^{\mathrm{MAC}}, v^{\mathrm{MAC}}, w^{\mathrm{MAC}})$ is a MAC vector field, u^{MAC} is defined at the points $\mathbf{x}_{i\square\frac{1}{2},j,k} \frac{1}{4} \eth ih, \eth j \flat \frac{1}{2}\flat h, \eth k \flat \frac{1}{2}\flat h\flat$, whereas v^{MAC} is defined at the points $\mathbf{x}_{i,j\square\frac{1}{2},k} \frac{1}{4} \eth\eth i \flat \frac{1}{2}\flat h, jh, \eth k \flat \frac{1}{2}\flat h\flat$, and w^{MAC} is defined at the points $\mathbf{x}_{i,j,k\square\frac{1}{2}} \frac{1}{4} \eth\eth i \flat \frac{1}{2}\flat h, \eth j \flat \frac{1}{2}\flat h, kh\flat$. In the following discussion, we shall introduce two different MAC vector fields, denoted $\mathbf{u}^{\mathrm{MAC}}$ and $\mathbf{u}^{\mathrm{ADV}}$. Note that these *staggered grid* velocities are distinct from the cell-centered velocity field, which is simply denoted $\mathbf{u}$.

Each grid cell at level $\ell < \ell_{\max}$ either is completely covered by cells on the next finer level or is not refined at all. Analogously, each cell face either is completely covered by cell faces on the next finer level or is not refined at all. Since we assume that the solution on finer levels is more accurate than that on coarser levels, we distinguish between *valid* and *invalid* regions of each level. For a cell-centered quantity, the valid region of level ℓ consists of precisely those level ℓ grid cells that are not covered by any finer grid cells. Similarly, for a face-centered quantity, the valid region of level ℓ consists of those level ℓ grid faces that are not covered by any finer grid faces. Note that face-centered values defined on the coarse-fine interface between levels ℓ and $\ell + 1$ are valid on level $\ell + 1$ but not on level ℓ.

In the present scheme, all quantities defined on the locally refined Cartesian grid are considered *composite grid variables*. For such a variable, the degrees of freedom are precisely those values that are within the *valid* region of each level of the patch hierarchy. The values in the *invalid* region of each level are implicitly defined in terms of the underlying fine grid values. In particular, values of a composite grid variable in the invalid region of level $\ell < \ell_{\max}$ are defined to be the conservative averages of the underlying fine grid values on level $\ell + 1$ (see, e.g., Eqs. (21) and (22)). Note that this definition for the values in the invalid region has a recursive character, since it is possible that an invalid grid cell at level ℓ may be covered not only by cells on level $\ell + 1$ but also by cells from finer levels. We make use of both cell-centered composite grid variables and face-centered composite grid variables (including composite grid MAC vector fields) in the adaptive scheme.

3.3. Cartesian grid interpolation and finite difference operators

We now introduce Cartesian grid interpolation operators and finite difference approximations to the spatial differential operators appearing in the Eulerian equations of motion. For two-dimensional locally refined grids, we follow an approach introduced by Ewing et al. [26] to discretize the gradient and Laplace operators, and in three spatial dimensions, we employ a straightforward generalization of their approach. We also introduce corresponding interpolation operators for locally refined grids which are used to obtain purely cell-centered composite grid discretizations of the gradient and divergence operators. The particular approach that we take is similar to that described by Minion [19] and Martin and Colella [20]; however, the discretizations that we employ at coarse-fine interfaces are somewhat simpler than those presented in [19,20]. As we discuss below, despite its comparative simplicity, our approach still appears to yield a globally second order accurate projection method. (See Section 3.3.4 for a more complete comparison of our present approach to earlier approaches.) Moreover, note that the particular discretizations that we employ are in no way restricted to the context of the immersed boundary method.

We proceed by first defining interpolation and finite difference operators that map cell-centered quantities to face-centered quantities (and vice versa). These "c ! f" and "f ! c" operators are then used to define cell-centered finite difference operators (i.e., operators that map cell-centered quantities to cell-centered quantities) on locally refined Cartesian grids. Note that in the absence of local refinement, these operators all reduce to standard second order accurate approximations.

In an attempt to simplify the discussion, we generally first consider the discretizations at the coarse-fine interface on a two-dimensional grid before considering the more complicated three-dimensional case. In particular, we consider a portion of the interface between levels $\ell \square 1$ and ℓ where a coarse cell, denoted $(I \square 1, J)$, lies directly to the left of $\mathtt{r}$ fine cells, collectively denoted $\eth i, j\flat, \eth i, j \flat 1\flat, \ldots, \eth i, j \flat \mathtt{r} \square 1\flat$. This situation is depicted in Fig. 3. When a careful presentation of the three-dimensional case is required, we shall consider

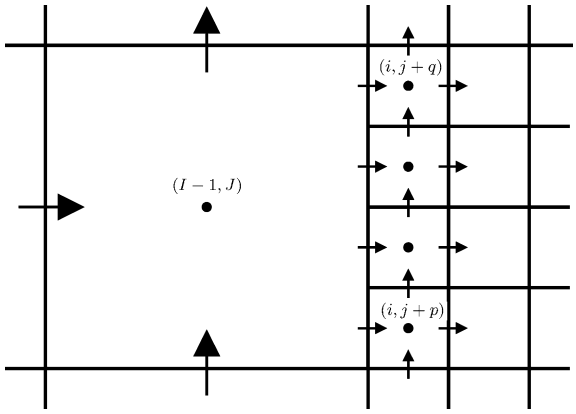

Fig. 3. Locations of cell and face-centered quantities in the vicinity of a coarse-fine interface between levels ℓ and $\ell - 1$ for a two-dimensional locally refined grid. Here, $r = 4$, $p = 0, \ldots, r - 1$, and $q = r - p - 1$. Note that $(i, j) = r - (I, J)$.

an analogous situation for a three-dimensional locally refined grid, where coarse cell $(I - 1, J, K)$ lies directly to the left of r^2 fine cells, $(i, j, k), \ldots, (i, j + r - 1, k), (i, j, k + 1), \ldots, (i, j + r - 1, k + r - 1)$. Note that here, $i = rI$, $j = rJ$, and $k = rK$.

3.3.1. Composite grid definitions for $A^{f \to c}$ and $\mathbf{D}^{f \to c}$.

Recall that values in the invalid regions of each level in the patch hierarchy are defined to be the conservative averages of the underlying fine data. To make this more concrete, consider a scalar function u defined on the x-faces of the composite grid. In reference to the two-dimensional configuration illustrated by Fig. 3, we have that the (invalid) value $u_{I-\frac{1}{2},J}$ is defined as the conservative average of the underlying fine grid values, namely

$$u_{I-\frac{1}{2},J} = \frac{1}{r} \sum_{p=0}^{r-1} u_{i-\frac{1}{2},j+p}. \tag{21}$$

The three-dimensional case is similar, with

$$u_{I-\frac{1}{2},J,K} = \frac{1}{r^2} \sum_{p_j,p_k=0}^{r-1} u_{i-\frac{1}{2},j+p_j,k+p_k}. \tag{22}$$

In either case, the remaining coarse grid values in the invalid region are determined in an analogous fashion.

With the values in the invalid regions of each level in the patch hierarchy so defined, when a cell-centered vector field $\mathbf{u}_{i,j,k}$ is defined by interpolating a MAC vector field, $\mathbf{u}^{\mathrm{MAC}}$, from cell faces to cell centers, the individual components of $\mathbf{u}$ are obtained by linear interpolation (averaging) on each level of the composite grid. We employ the notation

$$u_{i,j,k} = \left(A_1^{f \to c} u^{\mathrm{MAC}}\right)_{i,j,k} = \frac{u^{\mathrm{MAC}}_{i+\frac{1}{2},j,k} + u^{\mathrm{MAC}}_{i-\frac{1}{2},j,k}}{2}, \tag{23}$$

$$v_{i,j,k} = \left(A_2^{f \to c} v^{\mathrm{MAC}}\right)_{i,j,k} = \frac{v^{\mathrm{MAC}}_{i,j+\frac{1}{2},k} + v^{\mathrm{MAC}}_{i,j-\frac{1}{2},k}}{2}, \tag{24}$$

$$w_{i,j,k} = \left(A_3^{f \to c} w^{\mathrm{MAC}}\right)_{i,j,k} = \frac{w^{\mathrm{MAC}}_{i,j,k+\frac{1}{2}} + w^{\mathrm{MAC}}_{i,j,k-\frac{1}{2}}}{2}, \tag{25}$$

and we write $\mathbf{u} = A^{f \to c} \mathbf{u}^{\mathrm{MAC}}$. Similarly, the cell-centered divergence of a MAC vector field is approximated on each level of the hierarchy by centered differences, namely

$$\begin{aligned} \left(\mathbf{D}^{f \to c} \cdot \mathbf{u}^{\mathrm{MAC}}\right)_{i,j,k} &= \left(D_x^{f \to c} u^{\mathrm{MAC}}\right)_{i,j,k} + \left(D_y^{f \to c} v^{\mathrm{MAC}}\right)_{i,j,k} + \left(D_z^{f \to c} w^{\mathrm{MAC}}\right)_{i,j,k} \\ &= \frac{u^{\mathrm{MAC}}_{i+\frac{1}{2},j,k} - u^{\mathrm{MAC}}_{i-\frac{1}{2},j,k}}{h_\ell} + \frac{v^{\mathrm{MAC}}_{i,j+\frac{1}{2},k} - v^{\mathrm{MAC}}_{i,j-\frac{1}{2},k}}{h_\ell} + \frac{w^{\mathrm{MAC}}_{i,j,k+\frac{1}{2}} - w^{\mathrm{MAC}}_{i,j,k-\frac{1}{2}}}{h_\ell}. \end{aligned} \tag{26}$$

Away from coarse-fine interfaces, it is clear that $A^{f\to c}$ and $\mathbf{D}^{f\to c\cdot}$ yield standard second-order accurate interpolation and difference operators. It is not difficult to show that $A^{f\to c}$ is in fact second order accurate throughout the composite Cartesian grid. $\mathbf{D}^{f\to c\cdot}$ is second order accurate *away* from coarse-fine interfaces in the composite grid but is only first order accurate at coarse cells *adjacent* to each coarse-fine interface (see [27]).

3.3.2. Composite grid definitions for $A^{c\to f}$ and $\mathbf{G}^{c\to f}$

Away from coarse-fine interfaces, when a MAC vector field, $\mathbf{u}^{\mathrm{MAC}}$, is defined by interpolating $\mathbf{u}_{i,j,k}=(u_{i,j,k}, v_{i,j,k}, w_{i,j,k})$ from cell centers to cell faces, the individual components of $\mathbf{u}^{\mathrm{MAC}}$ are obtained by linear interpolation (averaging). We employ the notation

$$u^{\mathrm{MAC}}_{i-\frac{1}{2},j,k} = \delta A_1^{c\to f} u\big|_{i-\frac{1}{2},j,k} = \frac{u_{i-1,j,k}+u_{i,j,k}}{2}, \tag{27}$$

$$v^{\mathrm{MAC}}_{i,j-\frac{1}{2},k} = \delta A_2^{c\to f} v\big|_{i,j-\frac{1}{2},k} = \frac{v_{i,j-1,k}+v_{i,j,k}}{2}, \tag{28}$$

$$w^{\mathrm{MAC}}_{i,j,k-\frac{1}{2}} = \delta A_3^{c\to f} w\big|_{i,j,k-\frac{1}{2}} = \frac{w_{i,j,k-1}+w_{i,j,k}}{2}, \tag{29}$$

and say in this case that $\mathbf{u}^{\mathrm{MAC}} = A^{c\to f}\mathbf{u}$. Notice that only the normal component of $\mathbf{u}^{\mathrm{MAC}}$ is defined at a given cell face. Similarly, away from coarse-fine interfaces in the composite Cartesian grid, the MAC gradient of a cell-centered scalar quantity, ψ, is approximated at cell faces by

$$\delta G_x^{c\to f}\psi\big|_{i-\frac{1}{2},j,k} = \frac{\psi_{i-1,j,k}-\psi_{i,j,k}}{h_\ell}, \tag{30}$$

$$\delta G_y^{c\to f}\psi\big|_{i,j-\frac{1}{2},k} = \frac{\psi_{i,j-1,k}-\psi_{i,j,k}}{h_\ell}, \tag{31}$$

$$\delta G_z^{c\to f}\psi\big|_{i,j,k-\frac{1}{2}} = \frac{\psi_{i,j,k-1}-\psi_{i,j,k}}{h_\ell}. \tag{32}$$

At coarse-fine interfaces, the situation is more complicated, since the operators must involve cell-centered values taken from both sides of such interfaces. In an attempt to clarify the description of the discretizations employed at coarse-fine interfaces, we first present the two-dimensional case. We then describe the extension of this approach to three spatial dimensions.

Consider a cell-centered scalar function, $u_{i,j,k}$. To determine appropriate definitions for $A^{c\to f}$ and $\mathbf{G}^{c\to f}$ at a coarse-fine interface, we consider the Taylor series expansion of u about points on the interface. In reference to Fig. 3, the relevant Taylor expansions of $u(\mathbf{x})$ evaluated at fine cell faces $\mathbf{x}_{i-\frac{1}{2},j+p}$ are

$$u_{i,j+p} = u_{i-\frac{1}{2},j+p} + \frac{h_\ell}{2}\delta\partial_x u\big|_{i-\frac{1}{2},j+p} + \mathcal{O}(\delta h_\ell^2),$$

$$u_{i,j+q} = u_{i-\frac{1}{2},j+p} + \frac{h_\ell}{2}\delta\partial_x u\big|_{i-\frac{1}{2},j+p} + \delta q - p\,h_\ell\delta\partial_y u\big|_{i-\frac{1}{2},j+p} + \mathcal{O}(\delta h_\ell^2),$$

$$u_{I-1,J} = u_{i-\frac{1}{2},j+p} - \frac{r h_\ell}{2}\delta\partial_x u\big|_{i-\frac{1}{2},j+p} + \left(\frac{r-1}{2}-p\right)h_\ell\delta\partial_y u\big|_{i-\frac{1}{2},j+p} + \mathcal{O}(\delta h_\ell^2),$$

where $p = 0,\ldots,r-1$ and $q = r-p-1$. It is not hard to verify that

$$\delta A_1^{c\to f} u\big|_{i-\frac{1}{2},j+p} - \frac{2}{r+1}\frac{u_{i,j+p}+u_{I-1,J}}{2} + \frac{\delta 2r-1\,u_{i,j+p}-u_{i,j+q}}{2\delta r+1} = u_{i-\frac{1}{2},j+p} + \mathcal{O}(\delta h_\ell^2), \tag{33}$$

and that

$$\delta\mathbf{G}_x^{c\to f} u\big|_{i-\frac{1}{2},j+p} - \frac{2}{r+1}\frac{\delta u_{i,j+p}-u_{I-1,J}}{h_\ell} + \frac{1}{r+1}\frac{u_{i,j+q}-u_{i,j+p}}{h_\ell} = \delta\partial_x u\big|_{i-\frac{1}{2},j+p} + \mathcal{O}(\delta h_\ell). \tag{34}$$

Like the composite grid definition for $\mathbf{D}^{f\to c\cdot}$, the composite grid discrete gradient suffers from a localized reduction in accuracy at coarse-fine interfaces. Note that Eqs. (33) and (34) can be interpreted as "naive" interpolation and difference schemes that have been corrected to achieve second and first order accuracy, respectively.

We next turn our attention to three spatial dimensions. In this case, the relevant Taylor expansions of $u(\mathbf{x})$ evaluated at fine cell faces $\mathbf{x}_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k}$ can be compactly expressed via

$$\begin{pmatrix} u_{i,j\pm p_j,k\pm p_k} \\ u_{i,j\pm q_j,k\pm q_k} \\ u_{I\pm 1,J,K} \end{pmatrix} = \begin{pmatrix} 1 & \frac{h_\ell}{2} & 0 & 0 \\ 1 & \frac{h_\ell}{2} & \delta q_j - p_j\rangle h_\ell & \delta q_k - p_k\rangle h_\ell \\ 1 & -\frac{rh_\ell}{2} & \left(\frac{r+1}{2} - p_j\right)h_\ell & \left(\frac{r+1}{2} - p_k\right)h_\ell \end{pmatrix} \begin{pmatrix} u_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} \\ \delta\partial_x u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} \\ \delta\partial_y u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} \\ \delta\partial_z u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} \end{pmatrix} + \mathcal{O}\delta h_\ell^2\rangle,$$

where $p_j,\ p_k = 0,\ldots,r-1$, $q_j = r - p_j - 1$, and $q_k = r - p_k - 1$. It is not hard to verify that

$$\delta A_1^{c\to f} u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} = \frac{2}{r+1}\,\frac{u_{i,j\pm p_j,k\pm p_k} + u_{I\pm 1,J,K}}{2} + \frac{\delta 2r-1\rangle u_{i,j\pm p_j,k\pm p_k} - u_{i,j\pm q_j,k\pm q_k}}{2\delta r+1\rangle} = u_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} + \mathcal{O}\delta h_\ell^2\rangle, \tag{35}$$

and that

$$\delta \mathbf{G}_x^{c\to f} u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} = \frac{2}{r+1}\,\frac{\delta u_{i,j\pm p_j,k\pm p_k} - u_{I\pm 1,J,K}\rangle}{h_\ell} + \frac{1}{\delta r+1\rangle}\,\frac{u_{i,j\pm q_j,k\pm q_k} - u_{i,j\pm p_j,k\pm p_k}}{h_\ell}.$$
$$= \delta\partial_x u\rangle_{i\pm\frac{1}{2},j\pm p_j,k\pm p_k} + \mathcal{O}\delta h_\ell\rangle. \tag{36}$$

Note the similarity between Eqs. (33) and (35), and between Eqs. (34) and (36).

3.3.3. Cell-centered composite grid operators

With the foregoing MAC operators so defined, we are ready to define their purely cell-centered counterparts. The composite grid cell-centered divergence of a cell-centered vector field, $\mathbf{u} = (u, v, w)$, is approximated at cell centers on each level of the patch hierarchy by

$$\delta \mathbf{D}\cdot\mathbf{u}\rangle_{i,j,k} = \delta \mathbf{D}^{f\to c}\cdot A^{c\to f}\mathbf{u}\rangle_{i,j,k}. \tag{37}$$

Away from coarse-fine interfaces, this definition of the discrete divergence is identical to its standard second order accurate uniform grid counterpart, namely

$$\delta \mathbf{D}\cdot\mathbf{u}\rangle_{i,j,k} = \frac{u_{i+1,j,k} - u_{i-1,j,k}}{2h_\ell} + \frac{v_{i,j+1,k} - v_{i,j-1,k}}{2h_\ell} + \frac{w_{i,j,k+1} - w_{i,j,k-1}}{2h_\ell}. \tag{38}$$

At coarse-fine interfaces, a somewhat more complicated difference stencil is implicitly defined by the composition of the composite grid operators $\mathbf{D}^{f\to c}\cdot$ and $A^{c\to f}$. This approximation is second order accurate away from coarse-fine interfaces but is only first order accurate in the vicinity of a coarse-fine interface.

Likewise, the composite grid gradient of a cell-centered scalar function, ψ, is approximated at cell centers by

$$\delta \mathbf{G}\psi\rangle_{i,j,k} = \delta A^{f\to c}\mathbf{G}^{c\to f}\psi\rangle_{i,j,k}. \tag{39}$$

Again, away from coarse-fine interfaces, this composite grid definition of the discrete gradient is identical to its second order accurate uniform grid counterpart, namely

$$\delta \mathbf{G}\psi\rangle_{i,j,k} = \left(\frac{\psi_{i+1,j,k} - \psi_{i-1,j,k}}{2h_\ell}, \frac{\psi_{i,j+1,k} - \psi_{i,j-1,k}}{2h_\ell}, \frac{\psi_{i,j,k+1} - \psi_{i,j,k-1}}{2h_\ell}\right). \tag{40}$$

Like the composite grid discrete divergence, the cell-centered composite grid gradient is second order accurate away from coarse-fine interfaces but suffers from a localized reduction in accuracy in the vicinity of a coarse-fine interface.

Finally, the Laplacian of a cell-centered scalar function, ψ, is approximated at cell centers via

$$\delta L\psi\rangle_{i,j,k} = \delta \mathbf{D}^{f\to c}\cdot\mathbf{G}^{c\to f}\psi\rangle_{i,j,k}. \tag{41}$$

Note that this composite grid discretization of the Laplace operator is *symmetric* [26]. Away from the coarse-fine interface, the composite grid operator agrees with the standard second order accurate uniform grid discretization, namely

$$\delta L\psi\flat_{i,j,k} \frac{1}{4} \frac{\psi_{i\flat 1,j,k} \flat \psi_{i\square 1,j,k} \square 2\psi_{i,j,k}}{h_\ell^2} \flat \frac{\psi_{i,j\flat 1,k} \flat \psi_{i,j\square 1,k} \square 2\psi_{i,j,k}}{h_\ell^2} \flat \frac{\psi_{i,j,k\flat 1} \flat \psi_{i,j,k\square 1} \square 2\psi_{i,j,k}}{h_\ell^2}. \quad \delta 42\flat$$

Like the composite grid divergence and gradient operators, the cell-centered composite grid approximation to the Laplacian is only second order accurate *away* from coarse-fine interfaces.

3.3.4. Local reductions in the order of accuracy

As we have noted in the foregoing discussion, whereas the composite grid interpolation operators $A^{c\,!\,f}$ and $A^{f\,!\,c}$ are second order accurate throughout the patch hierarchy, each of the composite grid finite difference approximations is only second order accurate *away* from coarse-fine interfaces in the locally refined grid. In the vicinity of such interfaces, each of the composite grid finite difference operators employed in the present work is only first order accurate. Luckily, such localized reductions in the accuracy of the discretizations are acceptable since, at least when solving elliptic problems, it is well known that reducing the order of accuracy on lower dimensional interfaces within the computational domain does not alter the global accuracy of the solution. As we demonstrate below, in practice we observe empirical second order convergence rates for the adaptive immersed boundary method as long as the test problem is sufficiently smooth. Our adaptive projection method has also been verified to yield global second order accuracy when used as a standalone incompressible flow solver in both two and three spatial dimensions [27].

Even though they do not seem to prevent the scheme from attaining global second order accuracy, localized reductions in accuracy could be avoided altogether by employing higher order approximations at the coarse-fine interface. In the case of the discrete gradient operator, this has been done in previous projection methods for locally refined grids [19,20,24], where quadratic interpolation is employed to obtain a more accurate approximation to the gradient at the coarse-fine interface. Implementing the quadratic discretization at coarse-fine interfaces is somewhat more involved than the present approach since additional coarse grid values are used in the discretization. For some coarse-fine interface configurations, it is not even possible to perform the full coarse grid quadratic interpolation [20]. Since this quadratic approximation to the gradient has generally been paired with the same composite grid divergence operator as that used in the present work, the resulting composite grid discretization of ∇^2 still suffers from a localized reduction in accuracy near coarse-fine interfaces. Moreover, this discretization is *non-symmetric*, unlike the discretization employed in the present work.

Since both approaches appear to yield schemes that have essentially the same formal order of accuracy, it would be interesting to compare directly solutions obtained by the present approach with those obtained when the composite grid gradient is discretized by the methods described by [19,20,24]. In particular, it would be useful to determine whether the additional difficulties introduced by employing quadratic interpolation at coarse-fine interfaces (including loss of symmetry in the discretization of ∇^2 and a more complex implementation) are justified by improvements in the quality of the computed solution. At least in the context the immersed boundary method, however, as long as the coarse-fine boundary discretization yields a (globally) second order accurate projection method, we suspect that the choice of coarse-fine discretization will have little impact on the overall quality of the computed dynamics.

3.4. Discrete projection operators

Like all projection-type methods for incompressible flow, our method for the incompressible Navier–Stokes equations makes use of the Hodge decomposition theorem [21–23]. This result says that an arbitrary smooth vector field can be uniquely decomposed as the sum of a divergence free vector field and the gradient of a scalar function. On a *uniform* Cartesian grid, the (cell-centered) discrete analog of this decomposition is

$$\mathbf{w} \frac{1}{4} \mathbf{v} \flat \mathbf{G}\varphi, \quad \delta 43\flat$$

where $\mathbf{w}$ is an arbitrary cell-centered vector field on the Cartesian grid and $\mathbf{v}$ satisfies $(\mathbf{D}\cdot\mathbf{v})_{i,j,k} \equiv 0$ on the grid. Eq. (43) implicitly defines a projection operator, P, given by

$$\mathbf{v} \frac{1}{4} P\mathbf{w} \frac{1}{4} \delta I \square \mathbf{G}\delta\mathbf{D} \square \mathbf{G}\flat^{\square 1}\mathbf{D}\square\flat\mathbf{w}. \quad \delta 44\flat$$

Since $(\mathbf{D}\cdot\mathbf{v})_{i,j,k} \equiv 0$, for any vector field $\mathbf{w}$, $P^2\mathbf{w} = P\mathbf{w}$, so P is idempotent. That is to say, P is a projection.

In practice, the application of the operator defined by Eq. (44) requires the solution of a system of linear equations of the form $\mathbf{D}\cdot\mathbf{G}\varphi = \mathbf{D}\cdot\mathbf{w}$. On a uniform, periodic, three-dimensional Cartesian grid with an even number of grid cells in each coordinate direction, $\mathbf{D}\cdot\mathbf{G}$ has an eight-dimensional nullspace. (In general, on a uniform d-dimensional periodic grid with an even number of grid cells in each coordinate direction, $\mathbf{D}\cdot\mathbf{G}$ has a 2^d-dimensional kernel. Analogous operators are poorly behaved even in the absence of periodic boundary conditions or on grids with odd numbers of grid cells.) This complicates the solution process when iterative methods (such as multigrid) are employed to solve for φ. Moreover, when P is used in the solution of the incompressible Navier–Stokes equations, this non-trivial nullspace results in the decoupling of pressure field on eight sub-grids, leading to a so-called "checkerboard" instability. The difficulties posed by exact cell-centered projections are only compounded in the presence of local mesh refinement.

To avoid these difficulties, it was originally proposed in [35] that the foregoing exact projection be replaced by a carefully chosen approximate projection operator. In the present work, the approximate projection operator, $\tilde{P}$, is defined by

$$\tilde{P}\mathbf{w} = \left(I - \mathbf{G}\left(L^{-1}\mathbf{D}\right)\right)\mathbf{w}. \tag{45}$$

(In the uniform grid case, note that this approximate projection operator is the one first introduced by Lai [36]; see also, e.g. [20].) It is important to note that this operator is *not* a projection, since $L \neq \mathbf{D}\cdot\mathbf{G}$, although for smooth $\mathbf{u}$, $\|\mathbf{D}\cdot\tilde{P}\mathbf{u}\| \to 0$ as the composite grid is refined. On a uniform grid with periodic boundary conditions, it can be demonstrated analytically that $\|\tilde{P}\mathbf{w}\| \leqslant \|\mathbf{w}\|$, so in that case, the cell-centered approximate projection operator is stable [36]. Another important issue with regard to the stability of the overall method is the question of which quantity is to be (approximately) projected [25]; we address this issue below in Section 3.6.

Unlike exact projections for co-located cell-centered vector fields, exact projections of MAC vector fields are easily implemented both on uniform and locally refined grids, and require no computational machinery beyond that required to compute the foregoing approximate cell-centered projection, $\tilde{P}$. To determine the form of the exact MAC projection, first recall that for a cell-centered scalar quantity, ψ,

$$\left(L\psi\right)_{i,j,k} = \left(\mathbf{D}^{\mathrm{f}\to\mathrm{c}}\cdot\mathbf{G}^{\mathrm{c}\to\mathrm{f}}\psi\right)_{i,j,k}. \tag{46}$$

This correspondence allows us to compute easily the exact projection of a MAC vector field. In particular, the MAC projection of $\mathbf{w}^{\mathrm{MAC}}$ is given by

$$\mathbf{v}^{\mathrm{MAC}} = P^{\mathrm{MAC}}\mathbf{w}^{\mathrm{MAC}} = \left(I - \mathbf{G}^{\mathrm{c}\to\mathrm{f}}L^{-1}\mathbf{D}^{\mathrm{f}\to\mathrm{c}}\right)\mathbf{w}^{\mathrm{MAC}}, \tag{47}$$

where P^{MAC} denotes the MAC projection operator. This is an *exact* projection, since $\left(\mathbf{D}^{\mathrm{f}\to\mathrm{c}}\cdot\mathbf{v}^{\mathrm{MAC}}\right)_{i,j,k} \equiv 0$. Moreover, in practice, the application of P^{MAC} requires the solution of the same discrete Poisson problem that must be solved to apply the approximate cell-centered projection, $\tilde{P}$.

The approximate cell-centered projection defined above, $\tilde{P}$, can be reinterpreted in terms of the exact MAC projection, P^{MAC}. Given a cell-centered vector field, $\mathbf{w}$, the approximate projection of $\mathbf{w}$ is determined by first solving a system of linear equations for φ,

$$L\varphi = \mathbf{D}\cdot\mathbf{w}, \tag{48}$$

and then computing

$$\mathbf{v} = \mathbf{w} - \mathbf{G}\varphi. \tag{49}$$

To see the connection between the cell-centered approximate projection and the MAC projection, let $\mathbf{w}^{\mathrm{MAC}}$ be defined by $\mathbf{w}^{\mathrm{MAC}} = A^{\mathrm{c}\to\mathrm{f}}\mathbf{w}$. The (exact) MAC projection of $\mathbf{w}^{\mathrm{MAC}}$ is obtained by first solving a system of linear equations for φ^{f},

$$L\varphi^{\mathrm{f}} = \mathbf{D}^{\mathrm{f}\to\mathrm{c}}\cdot\mathbf{w}^{\mathrm{MAC}}, \tag{50}$$

and then computing

$$\mathbf{v}^{\mathrm{MAC}} = \mathbf{w}^{\mathrm{MAC}} - \mathbf{G}^{\mathrm{c}\to\mathrm{f}}\varphi^{\mathrm{f}}. \tag{51}$$

Since $\mathbf{D}\cdot\mathbf{w} = \mathbf{D}^{\mathrm{f}\,!\,\mathrm{c}}\cdot A^{\mathrm{c}\,!\,\mathrm{f}}\mathbf{w}$, however, it is clear that the solutions to (48) and (50) are equal up to an additive constant, i.e., $\varphi_{i,j,k} - \varphi^{0}_{i,j,k} = C$. Consequently, $(\mathbf{G}\varphi)_{i,j,k} \equiv (A^{\mathrm{f}\,!\,\mathrm{c}}\mathbf{G}^{\mathrm{c}\,!\,\mathrm{f}}\varphi^{0})_{i,j,k}$. Thus, when $\mathbf{w}^{\mathrm{MAC}} = A^{\mathrm{c}\,!\,\mathrm{f}}\mathbf{w}$, the cell-centered discrete gradient that is employed to obtain $\mathbf{v}$ in (49) is precisely the cell-centered interpolation of the MAC gradient that is used to obtain $\mathbf{v}^{\mathrm{MAC}}$ in (51).

From the foregoing discussion, it might appear that a more natural definition for an approximate cell-centered projection of $\mathbf{w}$ could be obtained by computing $A^{\mathrm{f}\,!\,\mathrm{c}}P^{\mathrm{MAC}}A^{\mathrm{c}\,!\,\mathrm{f}}\mathbf{w}$. That is: First, interpolate the cell-centered vector field to cell faces. Second, compute the exact MAC projection of the face-centered vector field. Third, interpolate the resulting discretely divergence-free, face-centered vector field back onto the cell centers. Although this may seem like a plausible way to define an approximate projection, this approach is not used. As Minion points out [19], the problem with this approach is that when simple averages are used to interpolate from cell centers to cell faces (and vice versa), this method introduces a diffusive term into the discretization that scales like the Cartesian grid spacing. In addition to yielding a method that is at best first order accurate, this diffusive term also dramatically smears out any fine scale features of the flow. Similar diffusive terms occur even if higher order interpolants are used in place of simple averaging. The approximate projection operator employed in the present work avoids these undesirable features.

3.5. A smoothed version of the Dirac delta function

In its treatment of the interaction equations that connect the Lagrangian and Eulerian frames, the immersed boundary method makes use of a smoothed approximation to the Dirac delta function that generally is of the tensor product form

$$\delta_{h_\ell}(\mathbf{x}) = \frac{1}{h_\ell^3}\phi\left(\frac{x}{h_\ell}\right)\phi\left(\frac{y}{h_\ell}\right)\phi\left(\frac{z}{h_\ell}\right), \tag{52}$$

recalling that $\mathbf{x} = (x,y,z)$ and that $h_\ell = \Delta x_\ell = \Delta y_\ell = \Delta z_\ell$.

Following the approach described in [30], it can be shown that the following five postulates uniquely determine one particular choice of $\phi(r)$:

$$\phi(r) \text{ is continuous for all real } r, \tag{53}$$

$$\phi(r) = 0 \quad \text{for } |r| \geqslant 3, \tag{54}$$

$$\sum_{j\ \text{even}} \phi(r-j) = \sum_{j\ \text{odd}} \phi(r-j) = \frac{1}{2} \quad \text{for all real } r, \tag{55}$$

$$\sum_{j} (r-j)^{m}\phi(r-j) = 0 \quad \text{for } m = 1,2,3 \text{ and for all real } r, \tag{56}$$

$$\sum_{j} (\phi(r-j))^2 = C \quad \text{for all real } r, \tag{57}$$

where the constant C is independent of r. (A detailed motivation for these postulates is provided in [7,30].) The unique function $\phi(r)$ that satisfies these requirements is

$$\phi(r) = \begin{cases} \frac{61}{112} - \frac{11}{42}|r| - \frac{11}{56}|r|^2 + \frac{1}{12}|r|^3 + \frac{\sqrt{-3}}{336}(243 + 1584|r| - 748|r|^2 \\ \quad -1560|r|^3 + 500|r|^4 + 336|r|^5 - 112|r|^6)^{\frac{1}{2}}, & 0 \leqslant |r| < 1, \\ \frac{21}{16} + \frac{7}{12}|r| - \frac{7}{8}|r|^2 + \frac{1}{6}|r|^3 - \frac{3}{2}\phi_6^{\mathrm{IB}}(|r|-1), & 1 \leqslant |r| < 2, \\ \frac{9}{8} - \frac{23}{12}|r| + \frac{3}{4}|r|^2 - \frac{1}{12}|r|^3 + \frac{1}{2}\phi_6^{\mathrm{IB}}(|r|-2), & 2 \leqslant |r| < 3, \\ 0, & 3 \leqslant |r|. \end{cases} \tag{58}$$

For this particular $\phi(r)$, the smoothed delta function that is constructed by (52) has a support of six mesh-widths in each coordinate direction (yielding a support of 216 grid cells in three spatial dimensions) and is referred to as the *six-point delta function*, which we denote by $\delta_{6h}^{\mathrm{IB}}(\mathbf{x})$. Many other choices for the regularized delta function, δ_h, are possible. See e.g. [3,37].

3.6. Timestepping

Projection methods for hierarchically composed locally refined grids fall into one of two categories: methods that employ a uniform timestep over the entire range of levels composing the composite grid [19,38], and methods that refine the timestep at the same rate as the spatial grid spacing (i.e., methods that employ *subcycling* in time) [20,24]. Although it has been estimated that projection methods that synchronously advance all levels of the grid hierarchy are less efficient than schemes employing subcycling in time [24], the present method takes the former approach for ease of implementation. We have tried to design our present implementation in such a way that subcycling in time can be introduced without too much additional effort, but the implementation of such a scheme in the immersed boundary context remains future work.

At the beginning of timestep n, we possess approximations to the values of the state variables at time t_n, namely $\mathbf{u}^n$ and $\mathbf{X}^n$. The pressure (which is in principle not a state variable) must be defined at half-timesteps to obtain a consistent second order accurate method. Thus, at the beginning of each timestep $n > 0$, we also possess an approximation to a timestep-lagged pressure, $p^{n-\frac{1}{2}}$. We also maintain an "auxiliary" MAC velocity, $\mathbf{u}^{\mathrm{MAC},n}$, that is used in our treatment of the nonlinear advection term that appears in the momentum equation. (As we describe below, the value of $\mathbf{u}^{\mathrm{MAC}}$ is obtained during the process of updating the cell-centered velocity, $\mathbf{u}$.)

To advance the solution forward in time by the increment Δt, we first compute $\mathbf{X}^{(n+1)}(q,r,s)$, a *preliminary* approximation to the locations of the nodes of the curvilinear mesh at time t_{n+1}. (Note that $\mathbf{X}^{(n+1)}$ is only our first approximation to the structure configuration at time t_{n+1}. The final approximation is denoted $\mathbf{X}^{n+1}$, and generally $\mathbf{X}^{(n+1)} \neq \mathbf{X}^{n+1}$.) To define $\mathbf{X}^{(n+1)}$, Eq. (4) is approximated by

$$\mathbf{X}^{(n+1)}(q,r,s) = \mathbf{X}^n(q,r,s) + \Delta t \sum_{i,j,k \in \text{2level } \ell_{\max}} \mathbf{u}^n_{i,j,k} \delta_{h_{\ell_{\max}}}(\mathbf{x}_{i,j,k} - \mathbf{X}^n(q,r,s)) h^3_{\ell_{\max}}. \tag{59}$$

(Recall that when we employ a regularized delta function with a support of d meshwidths in each coordinate direction, we ensure that the physical position of each node of the curvilinear mesh is at least $\lceil d/2 \rceil + 1$ grid cells away from the nearest coarse-fine interface on level $\ell_{\max}$.)

A discrete approximation to $\mathcal{F}[\mathbf{X}(\cdot,\cdot,\cdot,t)]$ provides the curvilinear elastic force densities corresponding to structure configurations $\mathbf{X}^n$ and $\mathbf{X}^{(n+1)}$, respectively denoted $\mathbf{F}^n$ and $\mathbf{F}^{(n+1)}$. The equivalent Cartesian elastic force densities are obtained by discretizing (3) and are given by

$$\mathbf{f}^n_{i,j,k} = \sum_{q,r,s} \mathbf{F}^n(q,r,s) \delta_{h_{\ell_{\max}}}(\mathbf{x}_{i,j,k} - \mathbf{X}^n(q,r,s)) \Delta q \Delta r \Delta s, \tag{60}$$

$$\mathbf{f}^{(n+1)}_{i,j,k} = \sum_{q,r,s} \mathbf{F}^{(n+1)}(q,r,s) \delta_{h_{\ell_{\max}}}(\mathbf{x}_{i,j,k} - \mathbf{X}^{(n+1)}(q,r,s)) \Delta q \Delta r \Delta s. \tag{61}$$

(Since the elastic structure is embedded within the finest level of the composite Cartesian grid, if $\mathbf{f}_{i,j,k} \neq 0$ for a *valid* cell (i,j,k), that cell must be on level $\ell_{\max}$.) A timestep-centered approximation to the Cartesian elastic force density is defined by

$$\mathbf{f}^{n+\frac{1}{2}} = \frac{1}{2}(\mathbf{f}^n + \mathbf{f}^{(n+1)}). \tag{62}$$

We next determine $\mathbf{u}^{n+1}$, $\mathbf{u}^{\mathrm{MAC},n+1}$, and $p^{n+\frac{1}{2}}$ by integrating the incompressible Navier–Stokes equations in time via a second order projection method similar to the method introduced by Bell et al. [23], a method that in turn is a second order accurate version of Chorin's original projection method [21,22]. Our algorithm, first introduced in [3], extends to the viscous case the hybrid approximate projection method ("version 5") introduced by Almgren et al. for the incompressible Euler equations [25]. In particular, as in [25], we obtain the values of $\mathbf{u}^{n+1}$ and $p^{n+\frac{1}{2}}$ in terms of the solutions to *different* projection equations. The value of $\mathbf{u}^{\mathrm{MAC},n+1}$ is obtained as a byproduct of the computation that yields $\mathbf{u}^{n+1}$.

Given $\mathbf{u}^n$, $\mathbf{u}^{\mathrm{MAC},n}$, $\mathbf{f}^{n+\frac{1}{2}}$, and $p^{n-\frac{1}{2}}$, we first obtain the approximation to the updated velocity, $\mathbf{u}^{n+1}$. We do so by discretizing the momentum equation (1) over the time interval Δt without imposing the constraint of incompressibility on $\mathbf{u}^{n+1}$. Instead, the gradient of the time-lagged pressure provides an approximation to the true pressure gradient. The nonlinear advection term is treated explicitly, and a version of the implicit

L-stable method of Twizell et al. [15] introduced by McCorquodale et al. [16] is used to integrate the viscous terms in time. With $\nu \equiv \mu/\rho$, the discretization of (1) is

$$(I - \eta_2 \nu L)(I - \eta_1 \nu L)\mathbf{u}^{*} = (I + \eta_3 \nu L)\mathbf{u}^{n} + \Delta t (I + \eta_4 \nu L)\left(-\mathbf{N}^{n+\frac{1}{2}} + \frac{1}{\rho}\left(\mathbf{f}^{n+\frac{1}{2}} - \mathbf{G}p^{n-\frac{1}{2}}\right)\right), \tag{63}$$

where $\mathbf{N}^{n+\frac{1}{2}} = \mathbf{N}^{n+\frac{1}{2}}(\mathbf{u}^{n}, \mathbf{u}^{\mathrm{MAC},n}, \mathbf{f}^{n}, \mathbf{G}p^{n-\frac{1}{2}})$ is the explicit approximation to $\frac{1}{2}(\mathbf{u} \cdot \nabla \mathbf{u})^{n+\frac{1}{2}}$ described in the appendix to [3] (note that in three spatial dimensions, we make use of full corner transport coupling [39]; see also [27] for a careful presentation of the particular three-dimensional treatment we employ), and

$$\eta_1 = \frac{a - \sqrt{a^2 - 4a + 2}}{2}\Delta t, \quad \eta_2 = \frac{a + \sqrt{a^2 - 4a + 2}}{2}\Delta t,$$

$$\eta_3 = (1 - a)\Delta t, \quad \eta_4 = \left(\frac{1}{2} - a\right)\Delta t,$$

with $a = 2 - \sqrt{2} - \epsilon$, where ϵ is machine precision.

The solution to Eq. (63) yields an intermediate velocity field, traditionally denoted $\mathbf{u}^{*}$, that is generally not discretely divergence free. In formulating a projection method, one may project either the velocity increment (i.e., $\mathbf{u}^{*} - \mathbf{u}^{n}$) or the intermediate velocity itself. Although either choice yields the same value for $\mathbf{u}^{n+1}$ when exact projections are employed, this is not the case when approximate projection operators are used. Several studies have found that a more stable algorithm is obtained by approximately projecting the intermediate velocity [25,20], and we follow this approach. In particular, $\mathbf{u}^{n+1}$ is obtained by making use of the approximate projection operator, $\tilde{P}$, defined by Eq. (45), yielding

$$\mathbf{u}^{n+1} = \tilde{P}\mathbf{u}^{*}. \tag{64}$$

To compute $\mathbf{u}^{\mathrm{MAC},n+1}$, we first interpolate $\mathbf{u}^{*}$ from cell centers to cell faces, obtaining

$$\mathbf{u}^{\mathrm{MAC},*} = A^{\mathrm{c}\to\mathrm{f}}\mathbf{u}^{*}. \tag{65}$$

In general, $\mathbf{u}^{\mathrm{MAC},*}$ is not discretely divergence free, so $\mathbf{u}^{\mathrm{MAC},n+1}$ is obtained by computing the MAC projection of $\mathbf{u}^{\mathrm{MAC},*}$, i.e.,

$$\mathbf{u}^{\mathrm{MAC},n+1} = P^{\mathrm{MAC}}\mathbf{u}^{\mathrm{MAC},*}. \tag{66}$$

Luckily, this operation does not require the solution of an additional system of linear equations! To see why this is so, recall that computing the approximate projection of $\mathbf{u}^{*}$ requires the solution of a discrete Poisson problem of the form

$$L\varphi = \mathbf{D} \cdot \mathbf{u}^{*}. \tag{67}$$

Since $(\mathbf{D}^{\mathrm{f}\to\mathrm{c}} \cdot \mathbf{u}^{\mathrm{MAC},*})_{i,j,k} \equiv (\mathbf{D}^{\mathrm{f}\to\mathrm{c}} \cdot A^{\mathrm{c}\to\mathrm{f}}\mathbf{u}^{*})_{i,j,k} \equiv (\mathbf{D} \cdot \mathbf{u}^{*})_{i,j,k}$, Eq. 67 is the same linear system that must be solved to project $\mathbf{u}^{\mathrm{MAC},*}$. The solution to 67, φ, may simply be reused to evaluate

$$\mathbf{u}^{\mathrm{MAC},n+1} = \mathbf{u}^{\mathrm{MAC},*} - \mathbf{G}^{\mathrm{c}\to\mathrm{f}}\varphi. \tag{68}$$

Having obtained the values of $\mathbf{u}^{n+1}$ and $\mathbf{u}^{\mathrm{MAC},n+1}$, we now turn our attention to computing the updated pressure. Although it is possible to determine this value in terms of the approximate projection of $\mathbf{u}^{*}$, we have found that it is beneficial to determine $p^{n+\frac{1}{2}}$ by approximately projecting a second intermediate velocity field that is given by a second treatment of the momentum equation. This alternate treatment of (1) is nearly identical to (63) except that it does not include *any* approximation to the pressure gradient, i.e.,

$$(I - \eta_2 \nu L)(I - \eta_1 \nu L)\tilde{\mathbf{u}}^{*} = (I + \eta_3 \nu L)\mathbf{u}^{n} + \Delta t (I + \eta_4 \nu L)\left(-\mathbf{N}^{n+\frac{1}{2}} + \frac{1}{\rho}\mathbf{f}^{n+\frac{1}{2}}\right). \tag{69}$$

The solution to this equation, $\tilde{\mathbf{u}}^{*}$, is the intermediate velocity that we project to obtain $p^{n+\frac{1}{2}}$. We emphasize that $\tilde{\mathbf{u}}^{*}$ is *only* used to compute $p^{n+\frac{1}{2}}$ and is not used in determining our final approximation to the velocity at time t_{n+1}. The approximate projection of $\tilde{\mathbf{u}}^{*}$, however, generates an *alternate* approximation to the velocity at time t_{n+1},

$$\tilde{\mathbf{u}}^{n+1} = \tilde{P}\tilde{\mathbf{u}}^{*} = \tilde{\mathbf{u}}^{*} - \mathbf{G}\tilde{\varphi}, \tag{70}$$

i.e.,

$$\tilde{\mathbf{u}}^{\square} = \tilde{\mathbf{u}}^{n+1} + \mathbf{G}\tilde{\varphi}, \tag{71}$$

where $\tilde{\varphi}$ is defined as the solution to a discrete Poisson problem,

$$L\tilde{\varphi} = \mathbf{D} \cdot \tilde{\mathbf{u}}^{\square}. \tag{72}$$

Since $\tilde{P}$ is an approximate projection operator, in general $\tilde{\mathbf{u}}^{n+1} \neq \mathbf{u}^{n+1}$.

The pressure consistent with (69) and (70) is the scalar function $p^{n+\frac{1}{2}}$ that satisfies

$$(I - \eta_2 vL)(I - \eta_1 vL)\tilde{\mathbf{u}}^{n+1} = (I + \eta_3 vL)\mathbf{u}^n + \Delta t(I + \eta_4 vL)\left(-\mathbf{N}^{n+\frac{1}{2}} + \frac{1}{\rho}\left(\mathbf{f}^{n+\frac{1}{2}} - \mathbf{G}p^{n+\frac{1}{2}}\right)\right). \tag{73}$$

Following [40], a second order accurate approximation to the updated pressure is determined by substituting (70) back into (69) and comparing the result to (73). Doing so, the discrete pressure gradient is seen to satisfy

$$(I + \eta_4 vL)\mathbf{G}p^{n+\frac{1}{2}} = \frac{\rho}{\Delta t}(I - \eta_2 vL)(I - \eta_1 vL)\mathbf{G}\tilde{\varphi}. \tag{74}$$

Consequently, we obtain $p^{n+\frac{1}{2}}$ via

$$p^{n+\frac{1}{2}} = \frac{\rho}{\Delta t}(I + \eta_4 vL)^{-1}(I - \eta_2 vL)(I - \eta_1 vL)\tilde{\varphi}. \tag{75}$$

Since $\eta_4 = -\frac{\Delta t}{2} - \frac{3}{2} + \epsilon\Delta t < 0$, and since L is a non-positive operator, $p^{n+\frac{1}{2}}$ is well-defined by (75). Note that $\tilde{\varphi}$ is proportional to a first order accurate approximation to the time-centered pressure. Full second order accuracy is obtained by solving a system of linear equations in (75). Although $p^{n+\frac{1}{2}}$ has no influence on the value obtained for $\mathbf{u}^{n+1}$, it is used in the next timestep, when computing $\mathbf{u}^{n+2}$.

(Note that in the foregoing, Eqs. (70), (71), (73) and (74), along with the quantity $\tilde{\mathbf{u}}^{n+1}$ that appears in those equations, are used to *derive* the appropriate definition for the updated pressure but not to *compute* the value of $p^{n+\frac{1}{2}}$. In particular, $p^{n+\frac{1}{2}}$ is computed only in terms of $\tilde{\mathbf{u}}^{\square}$ and $\tilde{\varphi}$ via Eqs. (69), (72) and (75).)

With the values $\mathbf{u}^{n+1}$, $\mathbf{u}^{\mathrm{MAC},n+1}$, and $p^{n+\frac{1}{2}}$ in hand, we complete the timestep by computing $\mathbf{X}^{n+1}$. Note that the explicit treatment of the advection terms effectively requires that Δt be sufficiently small to prevent any curvilinear mesh node from moving more than one meshwidth in any coordinate direction during a single timestep. In particular, the time increment must satisfy a CFL condition of the form

$$\Delta t \leqslant C \min_{\ell \in \{0,\ldots,\ell_{\max}\}} h_\ell \Big/ \max_{(i,j,k)\in \text{level } \ell} (|u^n_{i,j,k}|, |v^n_{i,j,k}|, |w^n_{i,j,k}|), \tag{76}$$

where the CFL number, C, is less than one. Since each curvilinear mesh node is at least $\lceil d/2 \rceil + 1$ grid cells away from the nearest coarse-fine interface on level $\ell_{\max}$ at the beginning of the timestep, for each (q,r,s), $\mathbf{X}^{(n+1)}(q,r,s)$ is at least $\lceil d/2 \rceil$ grid cells away from the nearest coarse-fine interface on level $\ell_{\max}$. Consequently, $\mathbf{X}^{n+1}(q,r,s)$ is well-defined by

$$\begin{aligned}\mathbf{X}^{n+1}(q,r,s) = \mathbf{X}^n(q,r,s) + \frac{\Delta t}{2}\Bigg(&\sum_{i,j,k \in \text{level } \ell_{\max}} \mathbf{u}^n_{i,j,k}\delta_{h_{\ell_{\max}}}(\mathbf{x}_{i,j,k} - \mathbf{X}^n(q,r,s))h^3_{\ell_{\max}} \\ &+ \sum_{i,j,k \in \text{level } \ell_{\max}} \mathbf{u}^{n+1}_{i,j,k}\delta_{h_{\ell_{\max}}}(\mathbf{x}_{i,j,k} - \mathbf{X}^{(n+1)}(q,r,s))h^3_{\ell_{\max}}\Bigg).\end{aligned} \tag{77}$$

Note that the evolution of the structure configuration via (59) and (77) takes the form of a second order accurate strong stability-preserving Runge–Kutta method [14]. Eq. (77) is an explicit formula for $\mathbf{X}^{n+1}$, since $\mathbf{X}^{(n+1)}$ is already defined; see Eq. (59).

Finally, we must discuss the initial timestep. The initial state of the system is completely determined by the initial values of $\mathbf{u}$ and $\mathbf{X}$. First, $\mathbf{u}^{\mathrm{MAC}}$ is initialized by interpolating the initial value of $\mathbf{u}$ from cell centers to cell faces,

$$\mathbf{u}^{\mathrm{MAC},0} = A^{\mathrm{c}\to\mathrm{f}}\mathbf{u}^0. \tag{78}$$

To ensure that the initial velocity at least *approximately* satisfies $(\mathbf{D} \cdot \mathbf{u}^0_{i,j,k} = 0$, we then perform an initial approximate projection, replacing $\mathbf{u}^0$ by

$$\mathbf{u}^0 \quad \tilde{P}\mathbf{u}^0. \tag{79}$$

We similarly replace $\mathbf{u}^{\mathrm{MAC},0}$ by its MAC projection,

$$\mathbf{u}^{\mathrm{MAC},0} \quad P^{\mathrm{MAC}}\mathbf{u}^{\mathrm{MAC},0}. \tag{80}$$

Again, having computed the approximate projection of $\mathbf{u}^0$, it is not necessary to solve any additional systems of linear equations to compute this initial MAC projection.

Next, the pressure must be determined from the values of $\mathbf{u}^0$, $\mathbf{u}^{\mathrm{MAC},0}$, and $\mathbf{X}^0$. We obtain the pressure iteratively as follows. First, the pressure is provisionally set to be identically zero. We then perform a preliminary timestep. The computation of this preliminary timestep yields a first approximation to the pressure at time $t = \frac{1}{2}\Delta t_0$. We then iteratively recompute the initial timestep, always using the most recent approximation to the pressure. After a small number of iterations (we use a total of five), we obtain a sufficiently accurate approximation to the pressure at time $t = \frac{1}{2}\Delta t_0$ to achieve overall second order accuracy.

3.7. Adaptive regridding

During the main timestepping loop, the locally refined Cartesian grid is regenerated at periodic intervals according to the refinement criteria described in Section 3.2. During this adaptive regridding process, a new patch hierarchy is generated, and each Eulerian quantity that is maintained on the locally refined Cartesian grid must be transferred from the old patch hierarchy to the new one. Where the old and new level ℓ patches overlap, such Eulerian quantities are copied directly from the old patches to the new ones. The remaining values on the new level ℓ patches are determined by interpolation from coarser levels.

The interpolation process employed during adaptive regridding may be either conservative or non-conservative, depending on the quantity that is being interpolated. For instance, the pressure, p, is not a conserved quantity, and so it suffices to employ simple trilinear interpolation to determine new values on level ℓ from values on the next coarser level (i.e., level $\ell - 1$). In particular, for a non-conserved quantity, those values on the new level ℓ patches that are not supplied by some old level ℓ patch are determined by performing trilinear interpolation using the nearest eight coarse grid values. On the other hand, for uniform density incompressible flows, the velocity, $\mathbf{u}$, is a conserved quantity (because of momentum conservation), and thus maintaining discrete conservation of $\mathbf{u}$ during adaptive regridding prevents spurious changes in the net momentum of the system. In this case, if a value on a new level ℓ patch is not supplied by some old level ℓ patch, it is obtained in terms of a piecewise trilinear reconstruction of $\mathbf{u}$ on the next coarser level via a multidimensional generalization of the monotonized central-difference (MC) limited piecewise linear reconstruction procedure. In the multidimensional MC limited interpolation procedure, the mean value of the reconstruction of $\mathbf{u}$ in a level $\ell - 1$ coarse grid cell (I, J, K) is set to the value $\mathbf{u}_{I,J,K}$, whereas the slope of the piecewise trilinear reconstruction in each coordinate direction is taken to be the MC limited slope in that direction, e.g., the slope of the reconstruction of u in the x-coordinate direction is defined by

$$\mathrm{minmod}\left(\frac{u^n_{I+1,J,K} - u^n_{I-1,J,K}}{2h_{\ell-1}}, 2\frac{u^n_{I,J,K} - u^n_{I-1,J,K}}{h_{\ell-1}}, 2\frac{u^n_{I+1,J,K} - u^n_{I,J,K}}{h_{\ell-1}}\right), \tag{81}$$

where the minmod function of three arguments is

$$\mathrm{minmod}(a,b,c) = \begin{cases} a & \text{if } |a| \leqslant \min(|a|,|b|,|c|) \text{ and } \mathrm{sign}(a) = \mathrm{sign}(b) = \mathrm{sign}(c), \\ b & \text{if } |b| \leqslant \min(|a|,|b|,|c|) \text{ and } \mathrm{sign}(a) = \mathrm{sign}(b) = \mathrm{sign}(c), \\ c & \text{if } |c| \leqslant \min(|a|,|b|,|c|) \text{ and } \mathrm{sign}(a) = \mathrm{sign}(b) = \mathrm{sign}(c), \\ 0 & \text{otherwise.} \end{cases} \tag{82}$$

Thus, if a, b, and c have the same sign, the minmod function evaluates to the one with the smallest modulus; otherwise, it evaluates to zero. By determining the slopes in this manner, the piecewise trilinear reconstruction retains second order accuracy where $\mathbf{u}$ is smooth and avoids creating spurious maxima or minima where $\mathbf{u}$ is not smooth. (See e.g. [41] for a more detailed presentation of slope limiters, including the MC limiter.) When this procedure is used to determine the new fine grid values, the discrete integral of $\mathbf{u}$ (and thus the momentum of the discrete system) is not altered by the adaptive regridding process.

As a result of the foregoing conservative interpolation procedure, the cell-centered velocity, $\mathbf{u}^n$, may no longer be "sufficiently divergence free" with respect to the cell-centered divergence operator, $\mathbf{D}\cdot$, following adaptive regridding. To address this issue, we replace $\mathbf{u}^n$ by its approximate projection after the regridding process is complete. Before doing so, however, we first reinitialize $\mathbf{u}^{\mathrm{MAC},n}$ by interpolating $\mathbf{u}^n$ from cell centers to cell faces,

$$\mathbf{u}^{\mathrm{MAC},n} \quad A^{\mathrm{c}\,!\,\mathrm{f}}\mathbf{u}^n. \tag{83}$$

Next, we replace $\mathbf{u}^n$ by its approximate projection, i.e.,

$$\mathbf{u}^n \quad \tilde{P}\mathbf{u}^n. \tag{84}$$

As usual, we simultaneously replace $\mathbf{u}^{\mathrm{MAC},n}$ by its exact MAC projection,

$$\mathbf{u}^{\mathrm{MAC},n} \quad P^{\mathrm{MAC}}\mathbf{u}^{\mathrm{MAC},n}, \tag{85}$$

without having to solve any additional systems of linear equations.

Note that a consequence of our explicit treatment of the Lagrangian equations of motion (i.e., Eqs. (59) and (77)) is that (76) with $C < 1$ is not the only stability constraint that Δt must satisfy. For many applications of the immersed boundary method, it is often necessary that $\Delta t = \mathcal{O}(h_{\ell_{\max}}^2)$ or even $\Delta t = \mathcal{O}(h_{\ell_{\max}}^4)$ to ensure the stability of the scheme. (Although note that replacing the explicit treatment of the Lagrangian equations of motion in the present scheme with an implicit treatment would presumably free the method from these onerous stability constraints and allow for the stable use of any $C < 1$ [42,43]. Unfortunately, at the present time, the development and implementation of an implicit version of the immersed boundary method that is suitable for large scale simulation remains future work.) Thus, in practice the timestep size is frequently guaranteed to satisfy (76) for $C \approx 1$, so that adaptively regenerating the patch hierarchy once every $n_{\mathrm{regrid}} = \lfloor 1/C \rfloor$ timesteps is sufficiently frequent to prevent the structure from escaping the finest level of the hierarchy. Adaptive regridding could be performed even less frequently by ensuring that each curvilinear mesh node is more than $\lceil d/2 \rceil + 1$ level $\ell_{\max}$ grid cells away from the coarse-fine interface between levels $\ell_{\max} - 1$ and $\ell_{\max}$, where d is the support of δ_h. This could be accomplished, for instance, by employing larger tag buffers when the patch hierarchy is (re-)constructed. In practice, we generally do not make any effort to decrease the regridding frequency since regridding is typically a small fraction of the overall computational cost of the scheme.

3.8. Summary of the adaptive algorithm

The adaptive algorithm is summarized as follows:

1: construct the initial patch hierarchy and initialize all state variables
2: $(\mathbf{u}^0, \mathbf{u}^{\mathrm{MAC},0})$ **project**$(\mathbf{u}^0)$
3: set $p^0 = 0$
4: **for** $n = 0$ to $n_{\max}$ by 1 **do** {the main timestep loop}
5: **if** mod $(n, n_{\mathrm{regrid}}) = 0$ and $n > 0$ **then**
6: regrid the patch hierarchy
7: interpolate Eulerian quantities from the old patch hierarchy to the new one
8: $(\mathbf{u}^n, \mathbf{u}^{\mathrm{MAC},n})$ **project**$(\mathbf{u}^n)$
9: **end if**
10: **for all** (q,r,s) in the curvilinear mesh **do** {see Eq. (59)}
11: interpolate $\mathbf{u}^n$ to $\mathbf{X}^n(q,r,s)$ to determine $\mathbf{U}^n(q,r,s)$
12: compute $\mathbf{X}^{(n+1)}(q,r,s) = \mathbf{X}^n(q,r,s) + \Delta t \mathbf{U}^n(q,r,s)$
13: **end for**
14: **for all** (q,r,s) in the curvilinear mesh **do** {see Eqs. (60) and (61)}
15: compute $\mathbf{F}^n(q,r,s)$ and $\mathbf{F}^{(n+1)}(q,r,s)$ from $\mathbf{X}^n(\cdot,\cdot,\cdot)$ and $\mathbf{X}^{(n+1)}(\cdot,\cdot,\cdot)$
16: determine $\mathbf{f}^n$ and $\mathbf{f}^{(n+1)}$ by spreading $\mathbf{F}^n(q,r,s)$ and $\mathbf{F}^{(n+1)}(q,r,s)$ to the Cartesian grid
17: **end for**
18: set $\mathbf{f}^{n+\frac{1}{2}} = \frac{1}{2}(\mathbf{f}^n + \mathbf{f}^{(n+1)})$

if $n = 0$ **then** {initialize the pressure}
  for $m = 1$ **to** 5 **by** 1 **do**
    compute the explicit approximation to $\frac{1}{2}\delta\mathbf{u} - r\,\mathbf{u}^{\frac{1}{2}}$ from $\mathbf{u}^0$, $\mathbf{u}^{\mathrm{MAC},0}$, $\mathbf{f}^0$, and p^0 {see [3,27]}
    solve Eq. (63) for $\mathbf{u}^*$
    $(\mathbf{u}^1, \mathbf{u}^{\mathrm{MAC},1})$ **project**$(\mathbf{u}^*)$
    solve Eqs. (73) and (75) to determine $p^{\frac{1}{2}}$
    $p^0 \quad p^{\frac{1}{2}}$
  end for
else {use the time-lagged pressure for all timesteps $n > 0$}
  compute the explicit approximation to $\frac{1}{2}\delta\mathbf{u} - r\,\mathbf{u}^{n+\frac{1}{2}}$ from $\mathbf{u}^n$, $\mathbf{u}^{\mathrm{MAC},n}$, $\mathbf{f}^n$, and $p^{n-\frac{1}{2}}$ {see [3,27]}
  solve Eq. (63) for $\mathbf{u}^*$
  $(\mathbf{u}^{n+1}, \mathbf{u}^{\mathrm{MAC},n+1})$ **project**$(\mathbf{u}^*)$
  solve Eqs. (73) and (75) to determine $p^{n+\frac{1}{2}}$
end if
for all (q,r,s) in the curvilinear mesh **do** {see Eq. (77)}
  interpolate $\mathbf{u}^{n+1}$ to $\mathbf{X}^{(n+1)}(q,r,s)$ to determine $\mathbf{U}^{(n+1)}(q,r,s)$
  compute $\mathbf{X}^{n+1}(q,r,s) = \mathbf{X}^n(q,r,s) + \frac{\Delta t}{2}\left(\mathbf{U}^n(q,r,s) + \mathbf{U}^{n+1}(q,r,s)\right)$
end for
end for

In the foregoing pseudocode, note that $\mathbf{U}(\cdot,\cdot,\cdot)$ is taken to be the interpolation of the Cartesian grid velocity, $\mathbf{u}$, to the nodes of the curvilinear mesh, and that the function **project**$(\mathbf{u}^*)$ is defined by the following procedure:

function $(\mathbf{u}, \mathbf{u}^{\mathrm{MAC}})$ **project**$(\mathbf{u}^*)$
  compute $\mathbf{u}^{\mathrm{MAC},*} = A^{c \leftarrow f}\mathbf{u}^*$
  solve $L\varphi = \mathbf{D} \cdot \mathbf{u}^*$ for φ
  compute $\mathbf{u} = \mathbf{u}^* - \mathbf{G}\varphi$
  compute $\mathbf{u}^{\mathrm{MAC}} = \mathbf{u}^{\mathrm{MAC},*} - \mathbf{G}^{c \leftarrow f}\varphi$

4. Computational convergence results in two spatial dimensions

Prior to [3], the convergence of the immersed boundary method typically had been studied computationally for problems which were not sufficiently smooth for the method to attain its formal convergence rate. In particular, most earlier convergence studies focused on the case of a viscous incompressible fluid interacting with an infinitely thin elastic membrane (i.e., an elastic interface or boundary). The true solutions to such problems possess discontinuities at the interface in the pressure and in the normal derivative of the velocity, and these discontinuities are not accurately captured either by the present version of the immersed boundary method or by earlier versions of the method. (An alternative approach is taken by the immersed interface method [44,45], where such discontinuities are explicitly accounted for by the method in a manner that yields higher order accuracy.)

We introduced a different approach to testing the immersed boundary method in [3], where we considered the interaction of a viscous incompressible fluid and an anisotropic incompressible viscoelastic shell of finite thickness. Since the discontinuities which are present in the true interface problem do not arise for the particular viscoelastic shell considered, this approach allowed us to assess the performance of the uniform grid version of the present scheme in a setting where convergence rates that corresponded to the formal order of accuracy of the method were both anticipated and observed. In the present work, we follow the same approach to test the adaptive version of the immersed boundary method. For comparison, we also summarize relevant results first reported in [3]. Although we present in [3] empirical convergence results over a broad range of Reynolds numbers, in the present work we restrict our attention to the moderate Reynolds number case, where $Re = 100$.

In all of our convergence studies, we consider a finite thickness anisotropic viscoelastic shell immersed in a viscous incompressible fluid. The mathematical model of the shell is constructed by superimposing two

components, so that they occupy the same region of space. One of these components is a continuum of massless elastic fibers, and the other is a viscous incompressible fluid that happens to have the same density and viscosity as the fluid in which the shell is immersed. The shell inherits its anisotropic elasticity from its fiber component, and its density, viscosity, and incompressibility from its fluid component. Even though the computational model involves discrete fibers, one should not think of discrete fibers in a continuous fluid, since the fluid equations are also discretized, and the meshwidths of the two discretizations are the same order of magnitude. The material properties of the shell as described above emerge in the common continuum limit of the two discretizations. We consider two particular sets of elastic properties. In Section 4.1, we specify the stiffness of the fibers so that the fiber tension smoothly tends to zero at the edges of the shell. As long as the structure does not become too distorted, the resulting Cartesian elastic force density, $\mathbf{f}$, will be a continuous function of $\mathbf{x}$. In Section 4.2, the fiber tension is taken to be a constant multiple of $|\partial\mathbf{X}/\partial s|$. In this case, the resulting Cartesian elastic force density is only piecewise continuous because of the sharp discontinuity in material properties that occurs at the fluid–structure interface. For the moderate Reynolds number flows considered in the present study, where $Re \square 100$, we observe global second order or nearly second order convergence rates in *both* situations.

Before proceeding to the specification of the two sets of elastic properties in Sections 4.1 and 4.2, and the presentation of the corresponding computational results, we first describe the common aspects for both sets of computations, including the computational meshes used to describe the Cartesian and curvilinear coordinate spaces, the initial conditions, and the choice of timestep.

For the two-dimensional problems considered in the present section, the physical domain, U, is taken to be the periodic unit square and is discretized on either a uniform or an adaptively refined grid. The number of grid cells in each coordinate direction on the coarsest level (i.e., on level 0) is denoted N_0, and the refinement ratio between successive levels of resolution is $\mathfrak{r}$. Thus the effective grid spacing on the finest level of the grid (i.e., on level $\ell_{\max}$) corresponds to that of a uniform grid with $N_{\ell_{\max}} = \mathfrak{r}^{\ell_{\max}} N_0$ grid cells in each coordinate direction. Unlike the discretization of the physical domain, the curvilinear mesh that discretizes the curvilinear coordinate space is fixed throughout each computation. For this two-dimensional computational study, we take the curvilinear coordinate (r,s)-space to be $\Omega = [0,1]\times[0,1]$, and we employ a fixed $M_r \times M_s$ computational lattice in the curvilinear coordinate space, where $M_r = \frac{3}{8}N_{\ell_{\max}}$ and $M_s = \frac{75}{16}N_{\ell_{\max}}$. With $\Delta r = 1/M_r$ and $\Delta s = 1/M_s$, the nodes of the curvilinear mesh are the points

$$(r,s) = (r_0,s_0) + (m_r\Delta r, m_s\Delta s) + \left(\frac{\Delta r}{2},\frac{\Delta s}{2}\right) + (m_r\Delta r, m_s\Delta s), \tag{86}$$

where $m_r \in \{0,1,\ldots,M_r \square 1\}$ and $m_s \in \{0,1,\ldots,M_s \square 1\}$. Here, the shift by $\frac{\Delta r}{2}$ avoids having fibers at the exact edges of the shell, whereas the shift by $\frac{\Delta s}{2}$ is for notational consistency only.

As discussed in Sections 2 and 3.1, the elastic force density generated by the structure configuration is defined in terms of a continuum of elastic fibers, and the curvilinear coordinates, (r,s), are chosen so that a fixed value of r labels a particular fiber for all time t. So that each fiber forms a closed loop, Ω is taken to be periodic in the s-coordinate direction. Note that s is *not* equal to arc length along the fibers. For this particular Lagrangian elastic force density mapping, no boundary conditions are imposed on the other boundaries of the curvilinear coordinate space.

The initial configuration of the viscoelastic body is given by the mapping

$$\mathbf{X}(r,s,0) = \left(\frac{1}{2},\frac{1}{2}\right) + \left(\left(\alpha + \gamma\left(r \square \frac{1}{2}\right)\right)\cos(2\pi s), \left(\beta + \gamma\left(r \square \frac{1}{2}\right)\right)\sin(2\pi s)\right), \tag{87}$$

with $\alpha = 0.2$, $\beta = 0.25$, and $\gamma = 0.0625$. This expression defines the initial configuration of each fiber to be an ellipse, so that the initial configuration of the entire structure is a thick elliptical shell. The value of γ determines the thickness of the shell. Note that γ is a physical parameter and not a numerical parameter, i.e., γ is independent of the grid spacing.

In all computations, the uniform density of the fluid–structure system is taken to be $\rho = 1$, the uniform viscosity is taken to be $\mu = 0.005$, and the initial velocity of the system is taken to be $\mathbf{u}(\mathbf{x},0) \equiv 0$. After being released at $t = 0$, the shell undergoes damped oscillations and tends toward its resting configuration, a circular shell. The computation is halted and convergence is assessed at $t = 0.4$. Using the fiber tensions specified in

either Section 4.1 or Section 4.2, the shell will have approximately completed its first oscillation at this point in the computation, and for both sets of material properties, the corresponding flows have $Re \approx 100$.

In all cases, we employ a uniform timestep that is chosen so that the computed velocity always satisfies the composite grid CFL condition for the CFL number $C = 0.1$, i.e.,

$$\Delta t < 0.1 \min_{\ell \in \{0,\ldots,\ell_{\max}\}} h_\ell / \max_{(i,j) \in \text{level } \ell} (|u^n_{i,j}|, |v^n_{i,j}|). \tag{88}$$

This is a more severe restriction than our explicit treatment of the nonlinear advection term requires; however, since we are treating the elastic force density in an explicit manner, the hyperbolic stability restriction is not the only stability constraint that the timestep size must satisfy. Although Eq. (88) may not be sufficient to ensure stability in the limit as $h_{\ell_{\max}} \to 0$, it appears to be adequate for the uniform and composite grids considered here. To prevent the viscoelastic structure from escaping the finest level of the hierarchical grid in the adaptive computations, we regrid the patch hierarchy every $n_{\text{regrid}} = 8$ timesteps. (For the present adaptive version of the immersed boundary method, adaptively regenerating the patch hierarchy once every $n_{\text{regrid}} = \lfloor 1/C \rfloor$ timesteps is sufficiently frequent to prevent the structure from escaping the finest level of the hierarchy, where C is the maximum CFL number.) For the following computations, Eq. (88) is satisfied for $\Delta t = 0.08/N_{\ell_{\max}}$, and this choice generally appears to result in stable computations for the grid spacings and material parameters we consider.

Below, we present empirical convergence rates for $\mathbf{u}$, p, and $\mathbf{X}$ in appropriately defined discrete L^p norms for $p = 1, 2$, and ∞. For $p = 1$ and 2, the discrete L^p norm of a cell-centered scalar valued function defined on the composite Cartesian grid, ψ, is given by

$$\|\psi\|_p^p = \sum_{\ell=0}^{\ell_{\max}} \sum_{\text{valid } (i,j) \in \text{level } \ell} |\psi_{i,j}|^p h_\ell^3. \tag{89}$$

The discrete L^∞ norm of ψ is defined by

$$\|\psi\|_\infty = \max_{\ell \in \{0,\ldots,\ell_{\max}\}} \max_{\text{valid } (i,j) \in \text{level } \ell} |\psi_{i,j}|. \tag{90}$$

Note that these discrete norms are only defined in terms of those values of ψ in the valid region of each level in patch hierarchy. Analogous definitions are employed for the discrete L^p norms of vector valued functions. For $\mathbf{W}(r,s) = (W_1(r,s), W_2(r,s))$, a vector valued function defined on the curvilinear mesh, the discrete L^p norm for $p = 1$ and 2 is defined by

$$\|\mathbf{W}\|_p = \left(\sum_{r,s} (W_1^2(r,s) + W_2^2(r,s))^{p/2} \Delta r \Delta s \right)^{1/p}. \tag{91}$$

Finally, the L^∞ norm of $\mathbf{W}(r,s)$ is defined by

$$\|\mathbf{W}\|_\infty = \max_{r,s} (W_1^2(r,s) + W_2^2(r,s))^{\frac{1}{2}}. \tag{92}$$

Since analytic solutions are not available for the present test cases, we compute estimates of the convergence rates in a standard manner. In particular, for a computed quantity, ψ, let $e_p(\psi; N_0, L, r)$ denote the discrete L^p norm of the difference in the approximation to ψ obtained when using an L-level hierarchical Cartesian grid with an $N_0 \times N_0$ base grid (and the corresponding curvilinear mesh and timestep) and the approximation to ψ obtained when using an L-level hierarchical Cartesian grid with a $2N_0 \times 2N_0$ base grid (and the corresponding curvilinear mesh and timestep), i.e.,

$$e_p(\psi; N_0, L, r) = \|\psi^{N_0} - \mathcal{I}^{(2N_0, L, r) \to (N_0, L, r)} \psi^{2N_0}\|_p, \tag{93}$$

where $\mathcal{I}^{(2N_0, L, r) \to (N_0, L, r)}$ denotes interpolation from finer to coarser L-level composite grids. An empirical estimate for the convergence rate of ψ in this norm is given by

$$r_p(\psi; N_0, L, r) = \log_2 \left(\frac{e_p(\psi; N_0, L, r)}{e_p(\psi; 2N_0, L, r)} \right). \tag{94}$$

4.1. Tapered elastic stiffness

For the first set of material properties, we set the fiber tension, T, via

$$T = \sigma(\|\partial \mathbf{X}/\partial s\|; r, s) = (1 - \sin(2\pi r - \pi/2))\|\partial \mathbf{X}/\partial s\|. \tag{95}$$

Recalling Eqs. (11)–(13), the resulting Lagrangian elastic force density is given by

$$\mathbf{F} = \frac{\partial}{\partial s}(T\boldsymbol{\tau}) = (1 - \sin(2\pi r - \pi/2))\frac{\partial^2 \mathbf{X}}{\partial s^2}. \tag{96}$$

In the absence of sharp corners in the elastic fibers that comprise the shell, this Lagrangian force density smoothly tapers to zero as r approaches 0 or 1, i.e., there is a *continuous* transition in material properties at the fluid–structure interface. As long as the structure does not become too distorted, the resulting Cartesian elastic force density, $\mathbf{f}$, will remain a continuous function of $\mathbf{x}$.

For these computations, we consider the performance of the uniform and adaptive schemes for the six-point delta function, $\delta_{6h}^{\mathrm{IB}}$, only. (Note that in [3], we consider the performance of the uniform version of the present scheme for several choices of δ_h.) We perform adaptive computations for two different refinement ratios, namely $r = 2$ and $r = 4$. In each case, the hierarchical Cartesian grid consists of two levels (i.e., levels $\ell = 0$ and 1). For comparison, we also perform a sequence of computations using a uniform Cartesian grid. Table 1 summarizes the convergence results in the various discrete norms for $\mathbf{u}$, p, and $\mathbf{X}$ at time $t = 0.4$. These values are obtained via Eqs. (93) and (94). In the uniform grid case, the reported values in the discrete L^1 and L^2 norms are identical to those reported in [3], except that in [3] the empirical convergence rates for the Cartesian grid velocity field were reported separately for each component of $\mathbf{u}$. (Pointwise convergence rates were not reported in our previous uniform grid study [3].)

From the presented data, it is clear that for *both* choices of r, better than second order convergence rates are indicated for nearly all quantities over the range grid spacings considered. In particular, third order convergence rates are observed for the pressure in most cases. Additionally, *pointwise* second order or nearly second order rates are observed in most quantities, and pointwise third order convergence rates are observed for the pressure. (Since we employ finite difference operators that suffer from localized reductions in accuracy at coarse-fine interfaces, note that only first order pointwise convergence rates are expected for sufficiently fine grid spacings.) Moreover, the results obtained by the adaptive scheme for a particular effective fine grid resolution appear to be substantially the same as those obtained by the non-adaptive scheme on an equivalent uniform grid. In particular, at equivalent fine grid resolutions, the normed differences of quantities from successive computations are largely the same, regardless of whether we have employed a uniform grid or an adaptively refined grid. This indirectly suggests that the actual errors are similar in all cases. (Particularly noteworthy is the fact that the norms of the differences in $\mathbf{X}$ obtained from subsequent computations appear to depend mainly on the value of $N_{\ell_{\max}}$, indicating that the computed motion of the viscoelastic structure is essentially the same for equivalent fine grid spacings.) Note, however, that at equivalent fine grid resolutions, the coarse grid in the $r = 4$ case is a factor of two coarser than the base grid in the $r = 2$ case.

Representative adaptive results for $r = 4$ and $N_{\ell_{\max}} = 512$ are displayed in the left-hand columns of Figs. 4 and 5. Similar results were obtained for $r = 2$. (See [3,27] for visualizations of the uniform grid results.)

4.2. Constant elastic stiffness

For the second set of material properties, we set the fiber tension, T, via

$$T = \|\partial \mathbf{X}/\partial s\|, \tag{97}$$

i.e., the stiffness of the fibers does *not* taper to zero at the edge of the shell. Recalling Eqs. (11)–(13), the resulting Lagrangian elastic force density is given by

$$\mathbf{F} = \frac{\partial}{\partial s}(T\boldsymbol{\tau}) = \frac{\partial^2 \mathbf{X}}{\partial s^2}. \tag{98}$$

Table 1
Normed differences of the values of **u**, p, and **X** from successive computations, and the resulting empirical convergence rates, in the discrete L^1, L^2, and L^∞ norms at time $t = 0.4$

$N_{\ell_{\max}}$	Uniform		r ¼ 2		r ¼ 4	
	Difference	Rate	Difference	Rate	Difference	Rate
L^1 difference in **u** at $t = 0.4$						
64	1.0150e □ 02	2.8166	1.0308e □ 02	2.7522	1.0384e □ 02	2.5696
128	1.4407e □ 03	2.2434	1.5299e □ 03	2.2962	1.7492e □ 03	2.0057
256	3.0426e □ 04	–	3.1149e □ 04	–	4.3558e □ 04	–
L^2 difference in **u** at $t = 0.4$						
64	1.2625e □ 02	2.9082	1.2660e □ 02	2.8484	1.2809e □ 02	2.6667
128	1.6819e □ 03	2.2904	1.7578e □ 03	2.3110	2.0172e □ 03	1.9670
256	3.4381e □ 04	–	3.5423e □ 04	–	5.1597e □ 04	–
L^∞ difference in **u** at $t = 0.4$						
64	6.3200e □ 02	2.8288	6.2661e □ 02	2.8076	6.3837e □ 02	2.3286
128	8.8957e □ 03	2.1722	8.9502e □ 03	2.2061	1.2709e □ 02	1.4773
256	1.9737e □ 03	–	1.9396e □ 03	–	4.5644e □ 03	–
L^1 difference in p at $t = 0.4$						
64	4.0813e □ 02	3.6165	4.0199e □ 02	3.5986	3.7191e □ 02	3.4758
128	3.3275e □ 03	2.7233	3.3183e □ 03	2.7886	3.3429e □ 03	2.9206
256	5.0389e □ 04	–	4.8026e □ 04	–	4.4149e □ 04	–
L^2 difference in p at $t = 0.4$						
64	1.0885e □ 01	3.3407	1.0714e □ 01	3.3198	1.0684e □ 01	3.3277
128	1.0745e □ 02	3.0064	1.0729e □ 02	3.0216	1.0642e □ 02	3.0667
256	1.3371e □ 03	–	1.3212e □ 03	–	1.2702e □ 03	–
L^∞ difference in p at $t = 0.4$						
64	9.5518e □ 01	2.7797	9.4336e □ 01	2.7549	9.4621e □ 01	2.8068
128	1.3909e □ 01	3.1454	1.3976e □ 01	3.1546	1.3523e □ 01	3.1496
256	1.5719e □ 02	–	1.5694e □ 02	–	1.5238e □ 02	–
L^1 difference in **X** at $t = 0.4$						
64	2.8595e □ 03	2.0497	2.8284e □ 03	2.0364	2.8138e □ 03	2.0274
128	6.9068e □ 04	2.3244	6.8947e □ 04	2.3234	6.9020e □ 04	2.3266
256	1.3790e □ 04	–	1.3775e □ 04	–	1.3760e □ 04	–
L^2 difference in **X** at $t = 0.4$						
64	3.2570e □ 03	1.7552	3.2196e □ 03	1.7435	3.2054e □ 03	1.7388
128	9.6485e □ 04	2.0251	9.6148e □ 04	2.0230	9.6039e □ 04	2.0334
256	2.3705e □ 04	–	2.3657e □ 04	–	2.3460e □ 04	–
L^∞ difference in **X** at $t = 0.4$						
64	8.4569e □ 03	1.2964	8.3929e □ 03	1.2954	8.3630e □ 03	1.3049
128	3.4432e □ 03	1.7644	3.4195e □ 03	1.7585	3.3848e □ 03	1.8066
256	1.0135e □ 03	–	1.0106e □ 03	–	9.6763e □ 04	–

In these computations, the stiffness of the elastic fibers comprising the shell tapers to zero at the edges of the structure, so that there is a continuous transition in material properties at the fluid–structure interface. The physical domain is described by either a uniform or an adaptively refined Cartesian grid with an effective $N_{\ell_{\max}} \square N_{\ell_{\max}}$ grid spacing on the finest level of the hierarchical grid. All adaptive computations employ a total of two levels, i.e., $\ell_{\max} = 1$, whereas in the uniform grid case, $\ell_{\max} = 0$.

In this case, the Cartesian elastic force density is only a piecewise continuous function of **x** due to a *sharp* transition in material properties at the fluid–structure interface.

As in the previous subsection, for these computations, we consider the performance of the scheme only for the six-point delta function, $\delta_{6h}^{\mathrm{IB}}$. We again perform adaptive computations for refinement ratios r ¼ 2 and r ¼ 4. In each case, the hierarchical Cartesian grid consists of two levels (i.e., levels $\ell = 0$ and 1). We also again

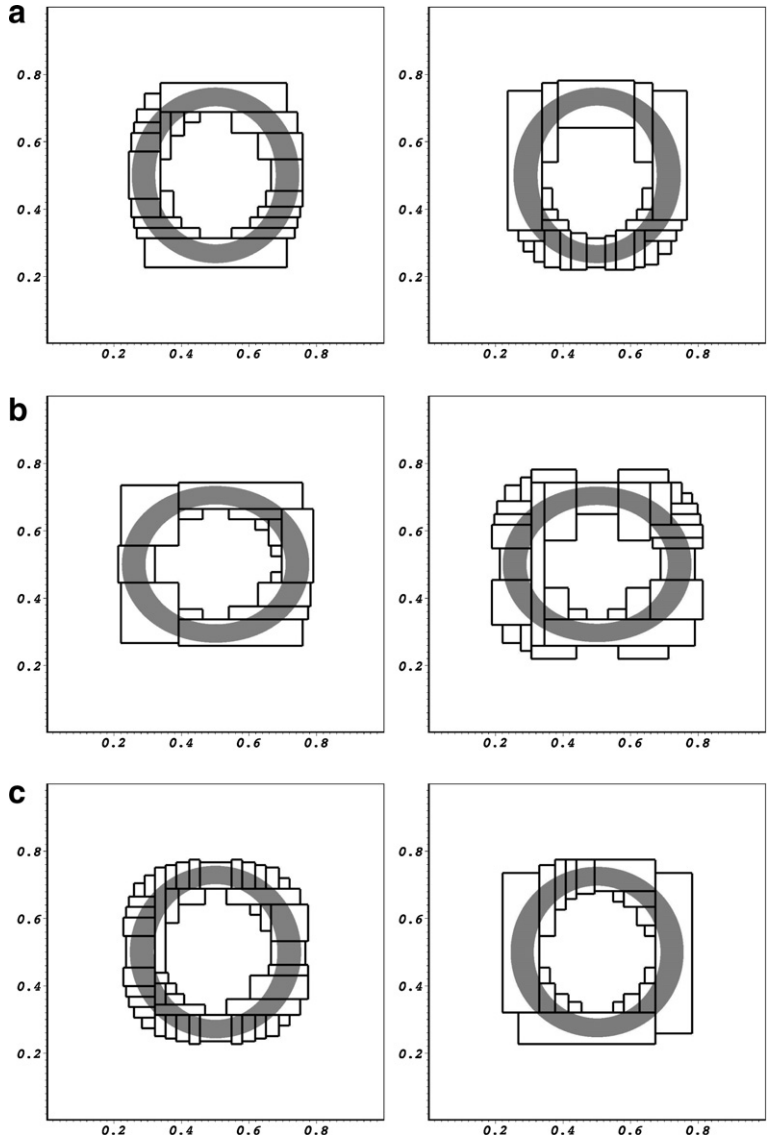

Fig. 4. Location of an anisotropic incompressible viscoelastic shell with tapered (left-hand column) and constant (right-hand column) elastic stiffnesses at (a) $t = 0.08$, (b) $t = 0.20$, and (c) $t = 0.32$ for an adaptive computation using $\delta^{\mathrm{IB}}_{6h}$. In all cases, level 0 consists of a single 128×128 grid patch, and the refinement ratio is r ¼ 4. The volume occupied by the shell is indicated in gray, and fine grid patches are indicated by thick black lines. Note that the refined regions in the Cartesian grid are placed adaptively and cover the elastic structure completely.

perform a sequence of computations on a uniform grid for comparison. Table 2 summarizes the convergence results in the various discrete norms for **u**, p, and **X** at time $t = 0.4$. Again, note that in the uniform grid case, the reported values in the discrete L^1 and L^2 norms are identical to those reported in [3], except that in [3] the empirical convergence rates for the Cartesian grid velocity field were reported separately for each component of **u**. (Pointwise convergence rates were not reported in our previous uniform grid study [3].)

From the presented data, it is clear that for the range of grid spacings considered, the adaptive scheme again achieves convergence rates that are essentially the same as those observed for the uniform discretization, and for *both* choices of r, the results obtained by the adaptive scheme for a particular effective fine grid resolution appear to be substantially the same as those obtained by the non-adaptive scheme on an equivalent

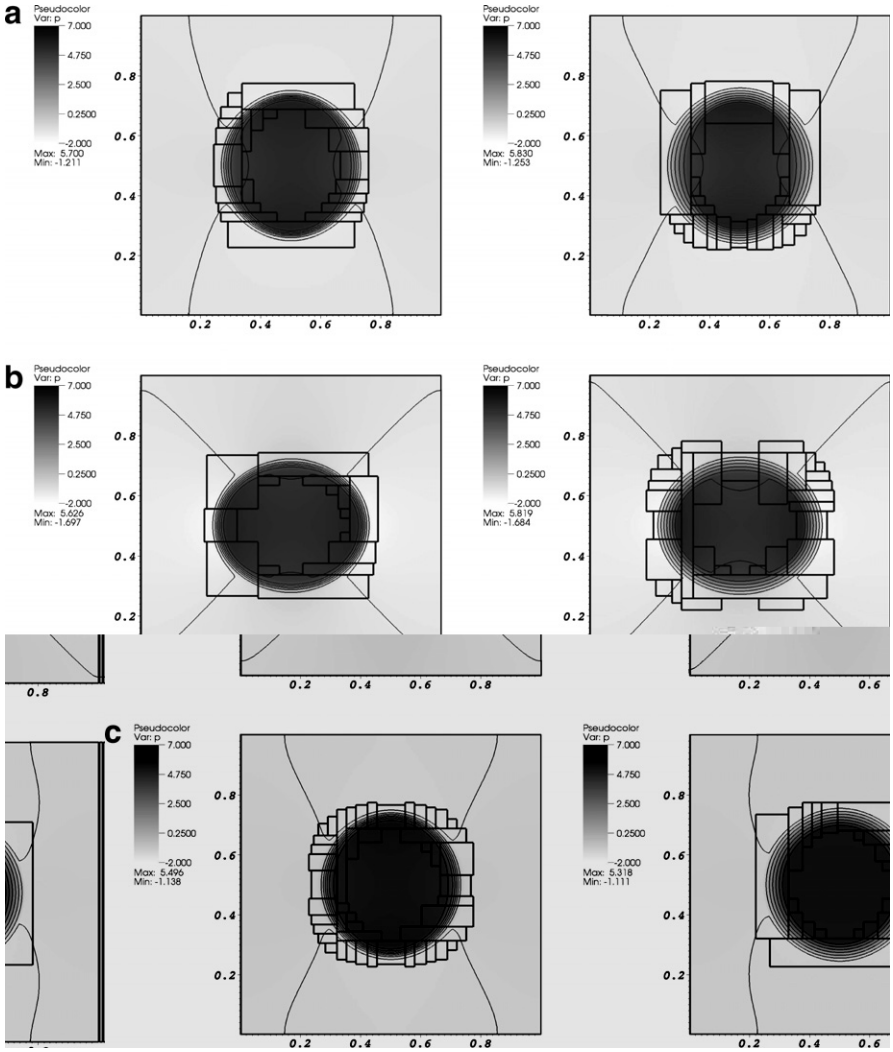

Fig. 5. Similar to Fig. 4, but here displaying computed values of p for a shell with tapered (left-hand column) and constant (right-hand column) elastic stiffnesses at (a) $t = 0.08$, (b) $t = 0.20$, and (c) $t = 0.32$. Pressure contours are indicated by thin black lines, and fine grid patches are indicated by thick black lines. Note that the variation in the pressure is relatively small in the unrefined portions of the hierarchical Cartesian grid. Also note that the essentially radial pressure gradient across the boundary in the case of constant elastic stiffness.

uniform grid. In this case, however, the empirically observed convergence rates are generally not as high as when the stiffness of the elastic fibers comprising the shell tends to zero near the edge of the structure. This is not surprising since, unlike the tapered case, the Cartesian force density in this case is in fact discontinuous. Nonetheless, nearly second order rates are observed in the discrete L^1 and L^2 norms in most cases, and first order or better pointwise convergence rates are observed in all cases.

Representative adaptive results for r ¼ 4 and $N_{\ell_{\max}}$ ¼ 512 are displayed in the right-hand columns of Figs. 4 and 5. Similar results were obtained for r ¼ 2. (See [3,27] for visualizations of the uniform grid results.)

5. Simulating cardiac blood-muscle-valve mechanics

In this section, we briefly present simulation and code performance results obtained from the application of the adaptive methodology described in the present work to the McQueen/Peskin model of cardiac

Table 2
Normed differences of the values of **u**, p, and **X** from successive computations, and the resulting empirical convergence rates, in the discrete L^1, L^2, and L^∞ norms at time $t = 0.4$

$N_{\ell_{\max}}$	Uniform		r ¼ 2		r ¼ 4	
	Difference	Rate	Difference	Rate	Difference	Rate
L^1 difference in **u** at $t = 0.4$						
64	8.8197e □ 03	1.6392	9.0270e □ 03	1.6614	8.9815e □ 03	1.6214
128	2.8315e □ 03	1.6232	2.8537e □ 03	1.6464	2.9191e □ 03	1.6877
256	9.1911e □ 04	–	9.1155e □ 04	–	9.0617e □ 04	–
L^2 difference in **u** at $t = 0.4$						
64	1.0044e □ 02	1.7023	1.0197e □ 02	1.7143	1.0231e □ 02	1.6623
128	3.0863e □ 03	1.6033	3.1077e □ 03	1.6230	3.2325e □ 03	1.6306
256	1.0158e □ 03	–	1.0089e □ 03	–	1.0439e □ 03	–
L^∞ difference in **u** at $t = 0.4$						
64	4.3920e □ 02	1.3837	4.4135e □ 02	1.3942	4.4539e □ 02	1.1489
128	1.6831e □ 02	1.4529	1.6791e □ 02	1.4650	2.0086e □ 02	1.1001
256	6.1481e □ 03	–	6.0825e □ 03	–	9.3700e □ 03	–
L^1 difference in p at $t = 0.4$						
64	1.3798e □ 02	1.7443	1.3690e □ 02	1.7219	1.3359e □ 02	1.6981
128	4.1186e □ 03	1.8255	4.1503e □ 03	1.8989	4.1170e □ 03	1.9865
256	1.1620e □ 03	–	1.1129e □ 03	–	1.0389e □ 03	–
L^2 difference in p at $t = 0.4$						
64	2.8840e □ 02	1.2875	2.8632e □ 02	1.2866	2.9261e □ 02	1.3165
128	1.1815e □ 02	1.5641	1.1737e □ 02	1.5685	1.1748e □ 02	1.5721
256	3.9956e □ 03	–	3.9571e □ 03	–	3.9512e □ 03	–
L^∞ difference in p at $t = 0.4$						
64	2.0990e □ 01	0.9055	2.0998e □ 01	0.9122	2.1377e □ 01	0.9327
128	1.1205e □ 01	1.1194	1.1158e □ 01	1.1063	1.1199e □ 01	1.0483
256	5.1577e □ 02	–	5.1827e □ 02	–	5.4149e □ 02	–
L^1 difference in **X** at $t = 0.4$						
64	1.5062e □ 03	1.5261	1.4935e □ 03	1.5218	1.4981e □ 03	1.5218
128	5.2298e □ 04	2.0493	5.2011e □ 04	2.0681	5.2174e □ 04	2.1183
256	1.2635e □ 04	–	1.2403e □ 04	–	1.2017e □ 04	–
L^2 difference in **X** at $t = 0.4$						
64	1.6878e □ 03	1.3427	1.6749e □ 03	1.3377	1.6808e □ 03	1.3381
128	6.6546e □ 04	1.7700	6.6268e □ 04	1.7743	6.6483e □ 04	1.7804
256	1.9512e □ 04	–	1.9372e □ 04	–	1.9354e □ 04	–
L^∞ difference in **X** at $t = 0.4$						
64	3.8333e □ 03	0.9472	3.8137e □ 03	0.9367	3.8423e □ 03	0.9357
128	1.9881e □ 03	1.1984	1.9924e □ 03	1.1954	2.0087e □ 03	1.1632
256	8.6631e □ 04	–	8.6997e □ 04	–	8.9688e □ 04	–

In these computations, the stiffness of the elastic fibers comprising the shell is constant throughout the structure, so that there is a sharp transition in material properties at the fluid–structure interface. The physical domain is described by either a uniform or an adaptively refined Cartesian grid with an effective $N_{\ell_{\max}} \times N_{\ell_{\max}}$ grid spacing on the finest level of the hierarchical grid. All adaptive computations employ a total of two levels, i.e., $\ell_{\max} = 1$, whereas in the uniform grid case, $\ell_{\max} = 0$.

mechanics [6–9]. This three-dimensional model includes representations of all of the major features of the heart and nearby great vessels, including the four chambers of the heart, i.e., the left and right ventricles and atria; the two atrioventricular (inflow) valves, including the supporting fans of chordae tendineae and papillary muscles; the two arterial (outflow) valves; the veins that return blood to the atria, i.e., the pulmonary veins and the superior and inferior vena cavae; and finally the major arteries that carry the

blood ejected by the ventricles, i.e., the ascending aorta and the main pulmonary artery. It is important to note that the inflow and outflow vessels are not connected to a detailed model of the circulatory system. Instead, the great vessels (including the ascending aorta, the pulmonary artery, the pulmonary veins, and the superior and inferior vena cavae) all have hemispherically capped blind ends. Fluid sources and sinks are supplied in or near these blind ends to establish realistic pressure loads and thereby simulate connections to the rest of the circulation. (See [27] for a complete description of the changes to the continuous and discrete equations of motion that are necessitated by the introduction of fluid sources and sinks that are internal to the physical domain.)

In the continuum limit, many of the structures of the model heart, including the heart valve leaflets and the thin-walled right ventricle, are described as thin elastic boundaries. An important exception is the muscular left ventricular wall, which is described as an anisotropic incompressible viscoelastic shell. Note that a cross-section of the thick left ventricular wall is analogous to a two-dimensional viscoelastic shell like those considered in Section 4. In particular, at the equatorial plane of the model heart, i.e., where the ventricles achieve their maximum diameter, the thickness of the model left ventricular wall varies from approximately 0.59 cm at the interventricular septum to approximately 0.78 cm at the posterior wall. As in the two-dimensional case considered in Section 4, the left ventricular wall has a Poisson ratio of 0.5, and the mass density and viscosity of the left ventricle are presently identical to the surrounding fluid, i.e., the blood. Additional viscous effects within the cardiac musculature are no doubt anisotropic as a result of the well defined muscle

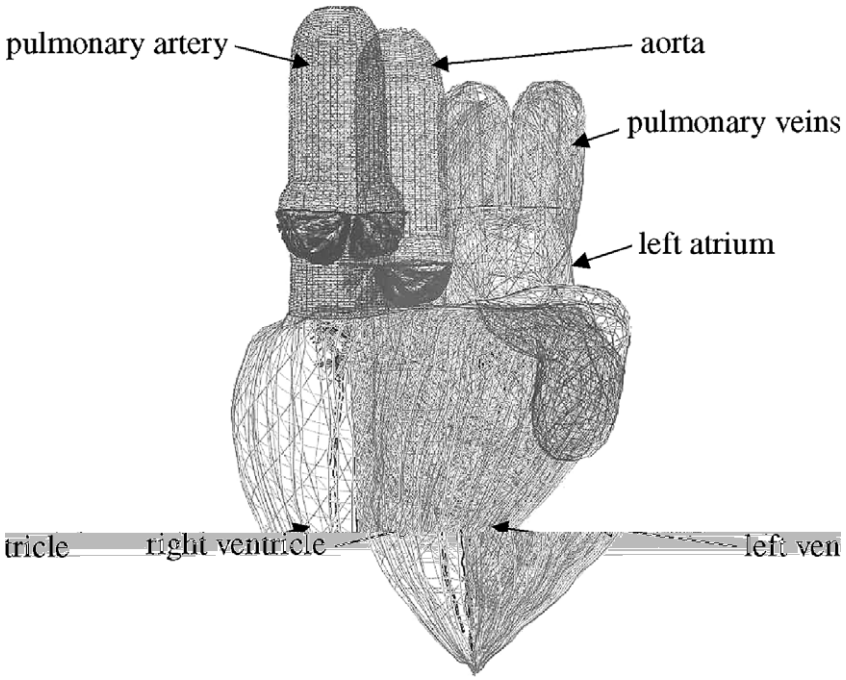

Fig. 6. The fiber structure of the model heart prior to contraction, as viewed from the front of the heart. On the left side of the heart (which appears on the right side of the figure), the four pulmonary veins supply blood to the left atrium, which in turn empties into the left ventricle through the mitral valve (see Figs. 7, 9, and 10). The muscular left ventricle ejects blood through the aortic valve into the ascending aorta. On the right side of the heart (which appears on the left side of the figure), the superior and inferior vena cavae return blood to the right atrium, which in turn empties into the right ventricle through the tricuspid valve (see Fig. 8). The thin-walled right ventricle ejects blood through the pulmonic valve into the main pulmonary artery. Note that in the model, the inflow and outflow vessels all have blind ends, but sources and sinks are provided to establish realistic pressure loads on each side of the heart. (In the present figure and all subsequent figures, only a subset of the model muscle fibers are displayed, whereas all of the model collagen fibers that comprise the heart valve leaflets are shown.) Note that in the simulation results presented in Figs. 6–12, we employ a two-level hierarchically composed Cartesian grid with a refinement ratio of $r = 4$. The global coarse grid (level 0) is a uniform 32^3 grid, and the grid spacing on the finest level is $h_{\ell_{\max}} = 0.20405$ cm, which would correspond to that of a uniform 128^3 grid.

fiber direction that exists at each point of the heart wall. This fiber orientation is already accounted for in our anisotropic model of the elasticity of the heart wall, and it will not be difficult in future work to generalize this anisotropic elasticity model to include fiber-aligned anisotropic viscous effects. At present, however, we only include the isotropic viscosity of the background fluid that is everywhere in an immersed boundary computation.

When discretized, each of the structures of the model is described by a system of one-dimensional elastic fibers: in the case of the valves, the fibers mainly correspond to passive collagen fibers; in the great vessels, they correspond to smooth muscle tissue; and in the myocardium, they correspond to active muscle fibers that possess time-dependent contractile properties. (As the Lagrangian mesh is refined, however, note that this discrete representation approaches the continuous limit described above.) Although a complete description of the elastic properties of these structures is beyond the scope of the present work, most of the forces generated by the elasticity of the model heart are computed in the manner described in Sections 3 and 4, although the fiber tension is determined differently. In particular, the elastic parameters of the fibers, such as the fiber stiffnesses and resting lengths, vary both temporally (to simulate active, contractile muscle) and spatially (to model the delay in contraction between the atria and the ventricles). Nonetheless, implementing the elastic properties specified by the model requires no major changes to the presented numerical scheme because at each timestep the elastic parameters are constant and known. Note that more complete descriptions of the model are avail-

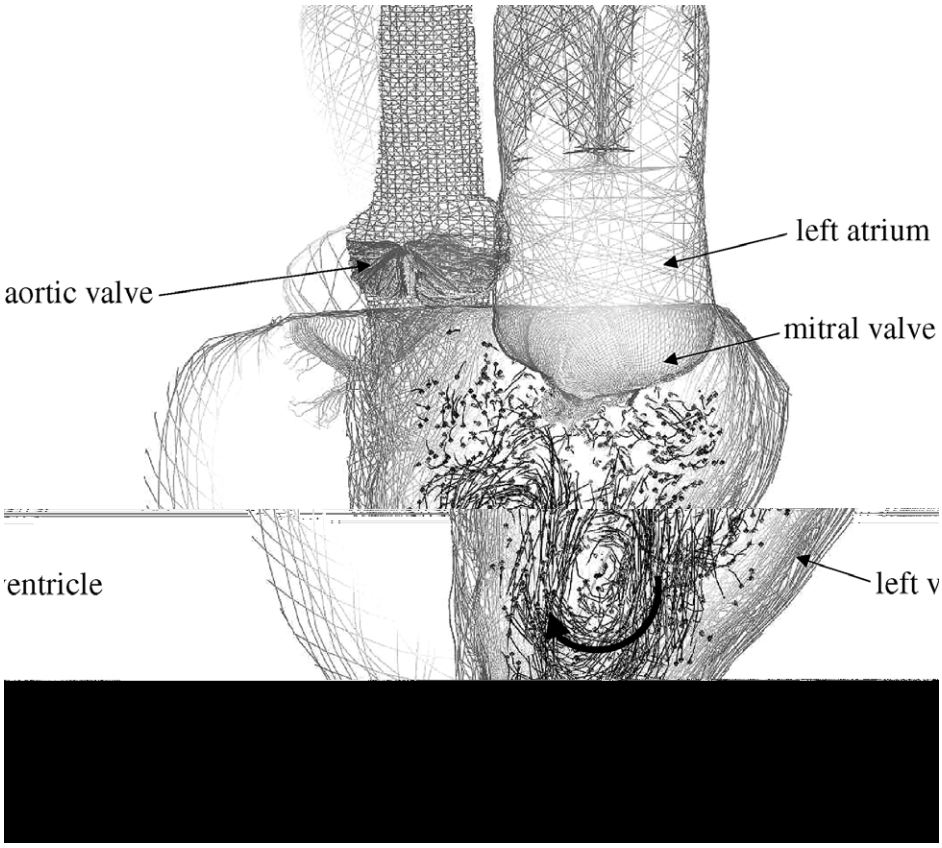

Fig. 7. A prominent vortex is shed from the mitral valve leaflets and migrates to the interior of the left ventricle of the model heart during atrial systole. Note that the present view is from the front of the model heart, so that the left ventricle appears on the right side of the figure. The flow of blood within the heart is indicated by passive fluid markers. The present positions of the fluid markers are shown, and attached to each marker is a dark tail that indicates the recent trajectory of that marker. A superimposed arrow indicates the direction of fluid flow around the vortex. The fibers that comprise the model heart, including the muscular heart wall, the thin valve leaflets, and the great vessels, appear in gray. Again, note that only a subset of the model fibers are displayed in each figure.

able from [6–8,27], and that a full discussion of cardiac physiology can be obtained from any text on medical physiology, e.g., Guyton and Hall [46].

In the present computations, the physical domain is described as a periodic box that has a length of 26.118 cm on each side. (For comparison, note that the circumference of the human mitral valve ring is approximately 10 cm and its diameter is approximately 3.2 cm.) First, we simulate a complete cardiac cycle using a two-level locally refined grid in which the coarse level (level 0) is a uniform grid with 32 grid cells in each coordinate direction, and the fine level (level 1) is adaptively generated with a refinement ratio of r ¼ 4. Results from this simulation are displayed in Figs. 6–12. For timing purposes, we also compare the performance of the adaptive immersed boundary method in three cases: (1) a uniform 128^3 Cartesian grid, (2) a two-level locally refined Cartesian grid with a coarse grid resolution of 64^3 and a refinement ratio of r ¼ 2, and (3) a two-level locally refined Cartesian grid with a coarse grid resolution of 32^3 and a refinement ratio of r ¼ 4. Note that in all cases, the grid spacing for the finest level (i.e., on level 0 in the uniform grid case and on level 1 in the locally refined cases) corresponds to that of a uniform grid with 128 grid cells in each coordinate direction, i.e., the fine grid spacing is $h_{\ell_{\max}} = 0.20405$ cm. As in the two-dimensional computations of Section 4, Cartesian grid cells are tagged for refinement when they contain any curvilinear mesh nodes, and consequently the entire structure of the heart is embedded within the finest level of the hierarchical grid. In an attempt to ensure that any vortices shed from the free edges of the valve leaflets remain within the finest level of the composite grid, cells are also tagged for refinement when the magnitude of the vorticity exceeds a specified relative tolerance. In particular, cell (i,j,k) is tagged for refinement whenever

$$\left(\sqrt{\boldsymbol{\omega}\cdot\boldsymbol{\omega}}\right)_{i,j,k} \geqslant 0.25\left\|\sqrt{\boldsymbol{\omega}\cdot\boldsymbol{\omega}}\right\|_1, \qquad (99)$$

where $\boldsymbol{\omega} = \nabla \times \mathbf{u}$ is the vorticity.

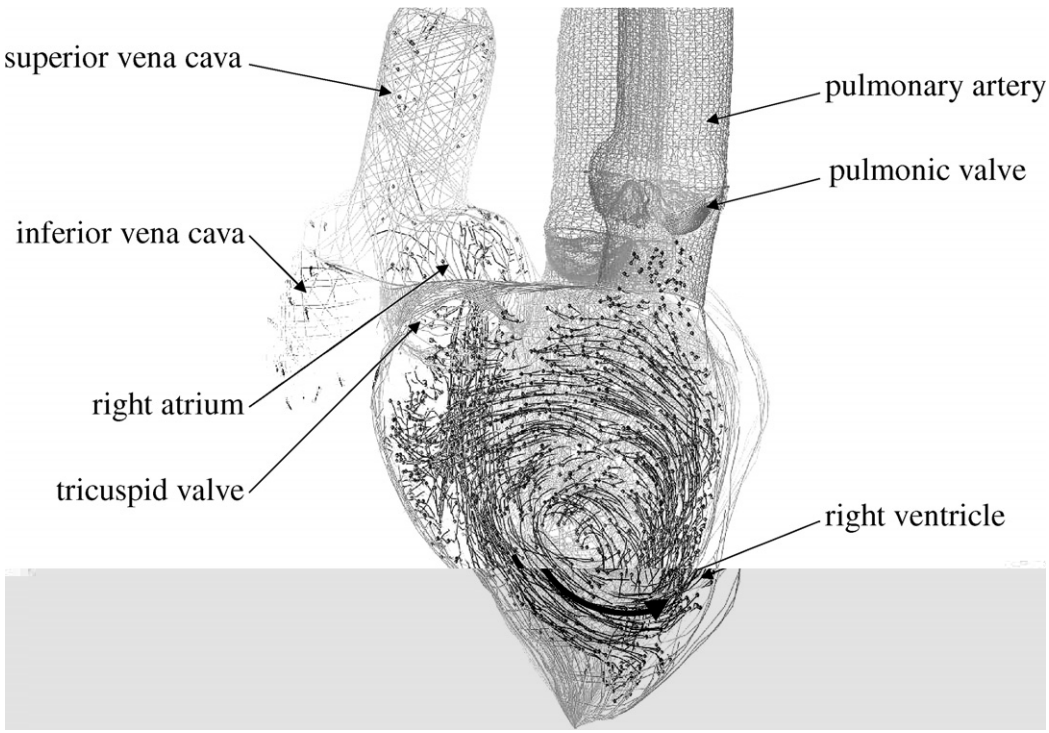

Fig. 8. A prominent vortex is shed from the tricuspid valve leaflets and migrates to the interior of the right ventricle during atrial systole. Note that in the present figure, the heart is viewed from its side, so that the right ventricle appears in the foreground. As before, the flow of blood within the model heart is indicated by passive fluid markers, and a superimposed arrow indicates the direction of fluid flow. The direction of swirl is such that when the right ventricle contracts, fluid is already moving towards the main pulmonary artery.

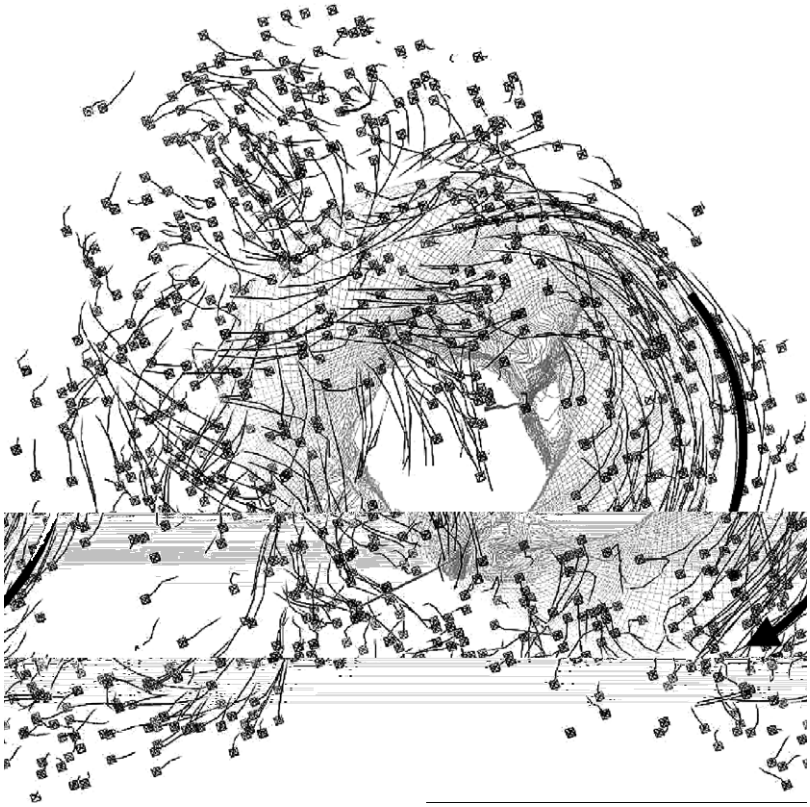

Fig. 9. A prominent clockwise (in the present view) vortex forms in the left ventricle about the jet of inflow through the mitral valve during atrial systole and prior to ventricular systole. The present view looks up from the left ventricle, through the open mitral valve, and into the left atrium. As before, the flow of blood within the model heart is indicated by passive fluid markers, and once again, a superimposed arrow indicates the direction of fluid flow. In the present figure, however, only the fibers that comprise the mitral valve are shown. Note that this particular feature of the flow did not appear in earlier simulations [7–9], and the physiological significance of this particular swirling motion, if any, is presently unknown.

The timestep size is determined to ensure that the CFL number never exceeds 0.1, and hence it suffices to regrid the patch hierarchy every $n_{\text{regrid}} = 10$ timesteps. To allow for more direct comparison with earlier computations performed by McQueen and Peskin, the present simulation employs the four-point delta function, δ_{4h}^{IB}, which is defined in e.g. [30].

Although the simulation results obtained by the present methodology show good qualitative agreement with results obtained by earlier versions of the immersed boundary method [7–9], there are two notable differences between the present computational results and earlier ones. First, the present methodology appears to provide dramatically enhanced boundary layer resolution when compared to earlier versions of the immersed boundary method. This is indicated by the flow through the arterial valves (i.e., the aortic and pulmonic valves). In particular, to prevent failure of the outflow valves during the initial portion of the simulation when the heart is first pressurized, we found that it was necessary to narrow the gaps between the leaflets of both the model aortic and pulmonic valves. The necessity of gaps between the valve leaflets may strike the reader as non-physical; however, as we briefly describe, gaps are necessary in all cases: Note that the interpolated velocity field obtained via the regularized delta function is continuous, so that if the physical positions of two material points happen to coincide at some time $t = T$, they necessarily coincide for all time $t \geqslant T$. Thus, in the absence of gaps between the valve leaflets, it would not be possible for the valves to open. When the valve spacings employed in [7–9] were used with the present version of the immersed boundary method, regurgitrant fluid jets formed near the tips of the valves leaflets, resulting in valve failure. It is important to emphasize that the original wide spacings yielded competent valves

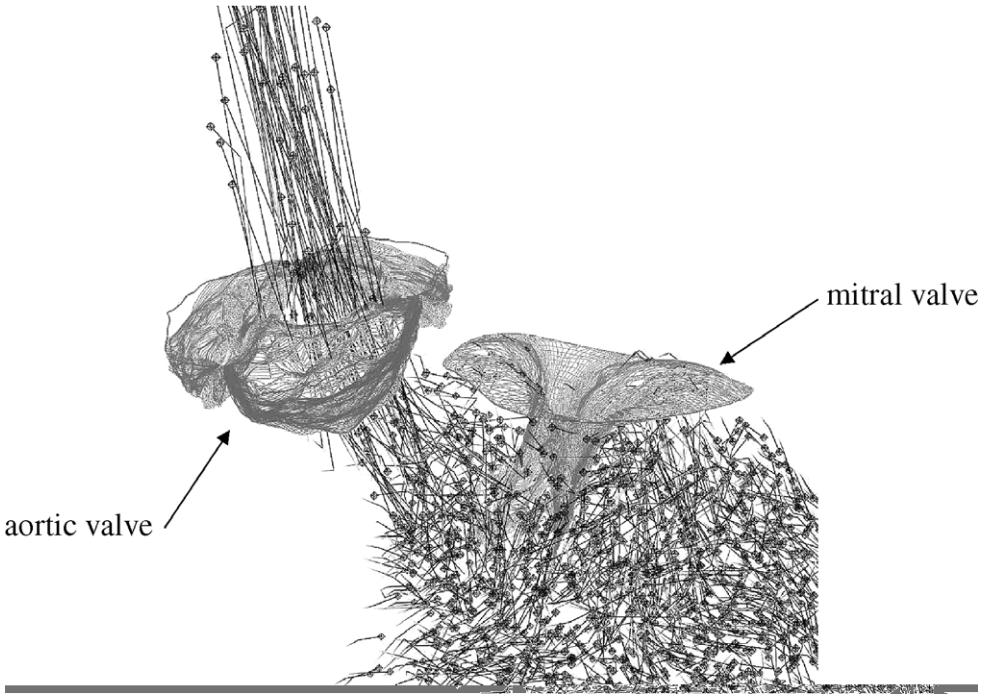

Fig. 10. The aortic and mitral valves during ventricular systole. As before, the flow of blood within the model heart is indicated by passive fluid markers, but here only the fibers that comprise the model aortic and mitral valves are displayed. Notice that the mitral valve prevents back-flow during ventricular ejection.

for earlier versions of the method. The fact that valve spacings that yielded competent valves via earlier versions of the method yield leaky valves for the present numerical scheme indicates that the present methodology provides enhanced boundary layer resolution compared to earlier versions of the immersed boundary method.

The improvement in boundary layer resolution yielded by the present methodology is more striking when we compare the Cartesian grid spacings employed in the present simulation with those used in earlier studies. In the present results, recall that the grid spacing on the finest level of the composite Cartesian grid is $h_{\ell_{\max}} = 0.20405$ cm. In comparison, the earlier uniform grid simulations [7–9] that made use of earlier versions of the immersed boundary method required the use of a *finer* uniform grid spacing in order to obtain *any* flow through the arterial valve *rings*, even in the absence of the arterial valve leaflets! In particular, the earlier uniform grid simulations, the uniform Cartesian grid spacing was $h_{\text{old}} = 0.13603$ cm, i.e., $h_{\text{old}} = \frac{2}{3} h_{\ell_{\max}}$. The value for h_{old} was approximately the coarsest value that allowed for flow through the arterial valves via the older numerical methods. (In fact, with the present methodology, we observe flow the arterial valves for even coarser Cartesian grid spacings than those employed in the presently reported simulation results.) In the present results, the distance from the tips of the aortic valve leaflets to the center of the aortic valve has been reduced by 50%, and the distance from the tips of the pulmonic valve leaflets to the center of the pulmonic valve has been reduced by 87.5%. With the present version of the immersed boundary method, these gaps must be further reduced to obtain competent valves for finer Cartesian grid spacings. Although these differences in the performance of the present and earlier versions of the immersed boundary method are quite dramatic, we believe that they can be attributed in large part to the more sophisticated treatment of the nonlinear advection terms employed in the present version of the method.

The second major difference in the dynamics between the present and earlier simulation results is the appearance of a prominent vortex that swirls about the jet of inflow from the mitral valve (see Fig. 10). This vortex forms within the model left ventricle after the onset of atrial systole but before

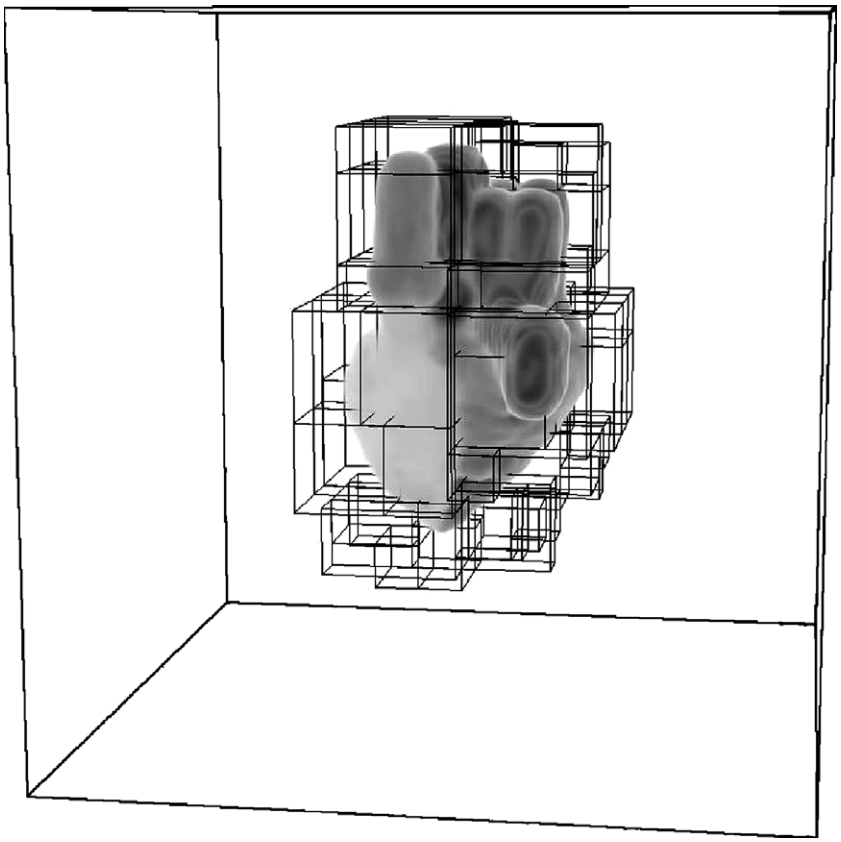

Fig. 11. Volume rendering of the pressure in the model heart during atrial systole and the corresponding locally refined Cartesian grid. Notice that the use of adaptive refinement allows us to employ in an efficient manner a computational domain that is physically larger than that used in earlier uniform grid computations, thereby deceasing the interaction between periodic copies of the model heart. Note that in the simulation results presented in Figs. 6–12, we employ a hierarchically composed Cartesian grid comprised of two levels with a refinement ratio of r ¼ 4. The global coarse grid (level 0) is a 32^3 uniform grid, and the grid spacing on the finest level is $h_{\ell_{\max}} = 0.20405$ cm. In the present figure, borders of the fine level 1 grid patches appear as thick black lines, whereas the borders of the computational domain are indicated by thin black lines. The coarse level 0 grid patches are not shown. To allow the right ventricle to appear clearly in the figure, note that the range of displayed pressure values does not include the full range of computed values.

the onset of ventricular systole. Similar flow patterns were not observed in the earlier simulations. At the present time, the physiological significance of this flow pattern, if any, is unknown. Although it is possible that similar swirling flow patterns occur in real hearts, it is also possible that this flow pattern is simply a consequence of a non-physiological feature of the present model or of under-resolution of the flow. The present simulation results are inadequate to resolve such questions, and clearly this issue warrants further investigation.

To demonstrate the potential of our adaptive scheme to reduce the computational resources required to simulate cardiac fluid mechanics, we also performed the first 100 timesteps of the simulation using 8, 16, 32, and 64 processors for both uniform and locally refined Cartesian grids, although we note that our implementation has not yet been fully optimized for performance. These computations were performed on the Multiprogrammatic and Institutional Computing Capability Resource (MCR) at Lawrence Livermore National Laboratory. (As presently configured, MCR is comprised of 1152 compute nodes, each consisting of two Intel 2.4 GHz Pentium 4 Xeon processors and 4 GB of memory.) Parallel timings are reported in Table 3. Notice that in all cases, the wall clock time required to compute a single timestep is significantly decreased by the use of adaptive mesh refinement. Moreover, the use of local refinement allows us to obtain good performance on

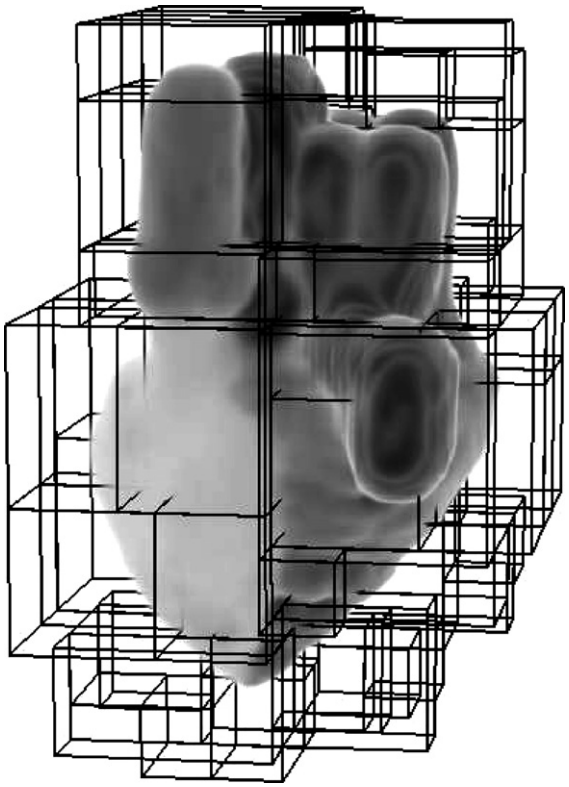

Fig. 12. Similar to Fig. 11, but here showing only the portion of the computational domain that is in the immediate vicinity of the model heart.

Table 3
Wall clock time (in seconds) required to perform a single computed timestep in the three-dimensional simulations of cardiac fluid mechanics

# of Processors	Uniform		r ¼ 2		r ¼ 4	
	Time per timestep	Adaptive speedup	Time per timestep	Adaptive speedup	Time per timestep	Adaptive speedup
8	94.46	–	44.34	2.13	12.53	7.54
16	57.53	–	10.03	5.74	7.61	7.56
32	41.96	–	7.44	5.64	5.81	7.22
64	10.74	–	7.40	1.45	6.74	1.59

Timings are obtained as averages over the first 100 timesteps in the simulation. In these computations, the physical domain is described either by a uniform $128 \times 128 \times 128$ grid, or by an adaptively refined grid in which the grid spacing on the finest level corresponds to that of a 128^3 uniform grid. All adaptive computations employ a total of two levels, so that for r ¼ 2, the global (level 0) coarse grid is a 64^3 uniform grid, whereas for r ¼ 4, the global coarse grid is a 32^3 uniform grid. Adaptive speedup is computed as the wall clock time required by the non-adaptive computation divided by the time required by the adaptive computation. Thus, an adaptive speedup of 2 would indicate that the adaptive computation ran twice as fast as the non-adaptive computation. (Note that the reported speedup numbers account only for the effect of adaptivity. In particular, they do not account for the effect of parallelization, i.e., *adaptive* speedup should not be confused with *parallel* speedup.) It seems likely that the 8, 16, and 32 processor uniform grid timings, as well as the 8 processor adaptive timings for r ¼ 2, are greatly influenced by memory caching effects. Consequently, the adaptive speedup results are likely most representative for the 64 processor case. Nonetheless, notice that with the present implementation and parallel platform, the uniform grid computation essentially *requires* 64 processors, whereas good performance is obtained in the adaptive case for as few as 16 processors.

as few as 16 processors, whereas reasonable performance in the uniform grid case is only obtained with 64 processors (likely as a result of memory cache effects). Again, however, we note that the present implementation has not yet been optimized to maximize parallel efficiency.

6. Conclusions

In the present work, we have introduced a formally second order accurate adaptive version of the immersed boundary method, examined the performance of this scheme for a prototypical fluid–structure interaction problem, and presented results from the application of this adaptive method to the simulation of cardiac blood-muscle-valve mechanics. This new algorithm is an extension of the uniform grid method described in [3], and both the present adaptive method and its uniform grid counterpart are differentiated from most previous versions of the immersed boundary method by their inclusion of several numerical methods intended to reduce the occurrence of non-physical oscillations in the computed dynamics. In particular, we use a strong stability-preserving Runge–Kutta method for the time integration of the structure configuration, an implicit L-stable discretization of the viscous terms in the momentum equation, and a second order Godunov method for the explicit treatment of the nonlinear terms in the momentum equation. We also employ a new hybrid approximate projection method for the incompressible Navier–Stokes equations, a method which has been demonstrated to reduce the occurrence of oscillations in the computed pressure for both uniform and adaptively refined computations [3,27]. To date, we have used this new adaptive hybrid projection method only in the context of the immersed boundary method, but the same algorithm could be used in any application area that would benefit from a locally adaptive projection scheme.

By considering fluid–structure interaction problems which possess sufficiently smooth solutions, *actual* second order convergence rates were demonstrated in our numerical tests of the method for moderate Reynolds number flows. Unlike most previous convergence studies for the immersed boundary method, however, we did not consider the interaction of a true *interface* and an incompressible fluid. When the immersed boundary method is applied to such problems, second order convergence rates are not observed because of the inability of the method to resolve accurately the discontinuities in the pressure and in the normal derivative of the velocity across the interface. We avoided these discontinuities by considering the interaction of an anisotropic incompressible viscoelastic shell of finite thickness and an incompressible fluid, but note that global second order or nearly second order convergence rates were observed not only for the case that there is a smooth transition in material properties at the fluid–structure interface but also for the case in which there is a *sharp* transition in material properties. Such problems are in some sense not as difficult as true interface problems; nonetheless, they are relevant to many application areas where the immersed boundary method is used. A particularly relevant example is the model of the heart and nearby great vessels we employ to simulate cardiac blood-muscle-valve mechanics (see Figs. 6–12 and also [6–9]). Although this model uses thin elastic boundaries to describe the heart valve leaflets, the description of the muscular left ventricular wall is that of an incompressible viscoelastic shell – albeit one with complex, time-dependent, and highly anisotropic elastic properties.

In the present work, we demonstrated for a two-dimensional test problem that our adaptive scheme produces results that are substantially the same as those obtained by the equivalent uniform grid method (i.e., results that are largely identical to those obtained on a uniform grid with resolution that is equal to the highest resolution employed in the adaptive computation). In particular, the adaptive scheme was demonstrated to yield convergence rates that were virtually identical to those produced by the equivalent uniform grid method. Moreover, we found that the adaptive method produced dynamics that were essentially the same as those produced by the equivalent non-adaptive scheme. This is not a surprise, perhaps, since in the test problems considered, the dominant errors in the computed solutions appear to be localized near the fluid–structure interface. In such situations, the least resolved portions of the solution will always be embedded in the finest level of the adaptively refined grid and will generally lie away from coarse-fine interfaces. Nonetheless, this result has important practical implications, since it indicates that for problems with localized fine scale features, it may be possible to obtain well resolved simulation results by adaptively deploying very high spatial resolution in only a limited portion of the computational domain.

We also presented simulation results obtained by applying the present adaptive method to the McQueen/Peskin model of cardiac mechanics. Timing results obtained for these simulations demonstrate that the use of adaptive mesh refinement reduces the computational resources required to perform simulations of cardiac blood-muscle-valve mechanics. These simulation results also indicate that the new methodology provides dramatically enhanced boundary layer resolution compared to earlier versions of the immersed boundary

method. Additionally, new features were observed in the flow patterns in the vicinity of the mitral valve. In both the present and earlier results, a prominent vortex is shed from the mitral valve leaflets and migrates to the interior of the left ventricle. A notable difference between the present simulation results and earlier ones, however, is the appearance in the present results of an additional vortex that swirls about the jet of inflow through the mitral valve prior to ventricular contraction. The physiological significance of this swirling motion is presently unknown, and it may be the product of non-physiological features of the present model heart or of inadequate spatial resolution. Alternatively, this could be our first glimpse of a physiological flow pattern that impacts mitral valve function. In any case, the resolution of this issue is beyond the scope of the present work and clearly merits further investigation.

Finally, we note that in the present work we have not described an important aspect of our methodology, namely the parallel implementation of this new adaptive immersed boundary method. This topic is addressed in [27]. We note here, however, that the parallel implementation relies on the SAMRAI (Structured Adaptive Mesh Refinement Applications Infrastructure) object-oriented C++ framework which is developed at the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory [47–50], the PETSc (Portable, Extensible Toolkit for Scientific Computation) library which is developed at the Mathematics and Computer Science Division at Argonne National Laboratory [51–53], and parallel multigrid solvers that are developed as part of the *hypre* project at the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory [54,55]. Although we provide preliminary timing results in Section 5, a more complete examination of the performance of our parallel implementation remains future work.

Acknowledgments

Portions of this work first appeared in the Ph.D. thesis of BEG [27], research which was supported in part by a New York University Graduate School of Arts and Science Dean's Dissertation Fellowship. BEG was also supported in part by the Department of Energy Computational Science Graduate Fellowship Program of the Office of Scientific Computing and Office of Defense Programs in the United States Department of Energy under contract DE-FG02-97ER25308, and is presently supported in part by National Science Foundation VIGRE Grant DMS-9983190 to the Department of Mathematics at the Courant Institute of Mathematical Sciences at New York University.

Portions of this work were performed under the auspices of the United States Department of Energy by University of California Lawrence Livermore National Laboratory under contract W-7405-Eng-48 UCRL-JRNL-216754.

The authors are indebted to Marsha Berger for her pioneering work on adaptive mesh refinement and for helpful discussions on the use of adaptive mesh refinement in the context of immersed boundary computations. We also thank the Center for Applied Scientific Computing at Lawrence Livermore National Laboratory for providing us with access to the computational resources required to perform the simulations of cardiac blood-muscle-valve mechanics.

References

[1] A.M. Roma, A multilevel self adaptive version of the immersed boundary method, Ph.D. Thesis, Courant Institute of Mathematical Sciences, New York University, 1996.

[2] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (2) (1999) 509–534.

[3] B.E. Griffith, C.S. Peskin, On the order of accuracy of the immersed boundary method: higher order convergence rates for sufficiently smooth problems, J. Comput. Phys. 208 (1) (2005) 75–105.

[4] C.S. Peskin, D.M. McQueen, A three-dimensional computational method for blood flow in the heart. I. Immersed elastic fibers in a viscous incompressible fluid, J. Comput. Phys. 81 (2) (1989) 372–405.

[5] D.M. McQueen, C.S. Peskin, A three-dimensional computational method for blood flow in the heart. II. Contractile fibers, J. Comput. Phys. 82 (2) (1989) 289–297.

[6] C.S. Peskin, D.M. McQueen, Mechanical equilibrium determines the fractal fiber architecture of aortic heart valve leaflets, Am. J. Physiol. Heart Circ. Physiol. 266 (1) (1994) H319–H328.

[7] C.S. Peskin, D.M. McQueen, Fluid dynamics of the heart and its valves, in: H.G. Othmer, F.R. Adler, M.A. Lewis, J.C. Dallon (Eds.), Case Studies in Mathematical Modeling: Ecology, Physiology, and Cell Biology, Prentice-Hall, Englewood Cliffs, NJ, USA, 1996, pp. 309–337.
[8] D.M. McQueen, C.S. Peskin, A three-dimensional computer model of the human heart for studying cardiac fluid dynamics, Comput. Graphics 34 (1) (2000) 56–60.
[9] D.M. McQueen, C.S. Peskin, Heart simulation by an immersed boundary method with formal second-order accuracy and reduced numerical viscosity, in: H. Aref, J.W. Phillips (Eds.), Mechanics for a New Millennium, Proceedings of the 20th International Conference on Theoretical and Applied Mechanics (ICTAM 2000), Kluwer Academic Publishers, Dordrecht, 2001.
[10] C.S. Peskin, Flow patterns around heart valves: a digital computer method for solving the equations of motion, Ph.D. Thesis, Albert Einstein College of Medicine, 1972.
[11] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (3) (1977) 220–252.
[12] M.J. Berger, J. Oliger, Adaptive mesh refinement for hyperbolic partial-differential equations, J. Comput. Phys. 53 (3) (1984) 484–512.
[13] M.J. Berger, P. Colella, Local adaptive mesh refinement for shock hydrodynamics, J. Comput. Phys. 82 (1) (1989) 64–84.
[14] S. Gottlieb, C.-W. Shu, E. Tadmor, Strong stability-preserving high-order time discretization methods, SIAM Rev. 43 (1) (2001) 89–112.
[15] E.H. Twizell, A.B. Gumel, M.A. Arigu, Second-order, L_0-stable methods for the heat equation with time-dependent boundary conditions, Adv. Comput. Math. 6 (3–4) (1996) 333–352.
[16] P. McCorquodale, P. Colella, H. Johansen, A Cartesian grid embedded boundary method for the heat equation on irregular domains, J. Comput. Phys. 173 (2) (2001) 620–635.
[17] P. Colella, Multidimensional upwind methods for hyperbolic conservation laws, J. Comput. Phys. 87 (1) (1989) 171–200.
[18] M.L. Minion, On the stability of Godunov-projection methods for incompressible flow, J. Comput. Phys. 123 (2) (1996) 435–449.
[19] M.L. Minion, A projection method for locally refined grids, J. Comput. Phys. 127 (1) (1996) 158–178.
[20] D.F. Martin, P. Colella, A cell-centered adaptive projection method for the incompressible Euler equations, J. Comput. Phys. 163 (2) (2000) 271–312.
[21] A.J. Chorin, Numerical solution of the Navier–Stokes equations, Math. Comput. 22 (104) (1968) 745–762.
[22] A.J. Chorin, On the convergence of discrete approximations to the Navier–Stokes equations, Math. Comput. 23 (106) (1969) 341–353.
[23] J.B. Bell, P. Colella, H.M. Glaz, A second-order projection method for the incompressible Navier–Stokes equations, J. Comput. Phys. 85 (2) (1989) 257–283.
[24] A.S. Almgren, J.B. Bell, P. Colella, L.H. Howell, M.L. Welcome, A conservative adaptive projection method for the variable density incompressible Navier–Stokes equations, J. Comput. Phys. 142 (1) (1998) 1–46.
[25] A.S. Almgren, J.B. Bell, W.Y. Crutchfield, Approximate projection methods: Part I. Inviscid analysis, SIAM J. Sci. Comput. 22 (4) (2000) 1139–1159.
[26] R.E. Ewing, R.D. Lazarov, P.S. Vassilevski, Local refinement techniques for elliptic problems on cell-centered grids I. Error analysis, Math. Comput. 56 (194) (1991) 437–461.
[27] B.E. Griffith, Simulating the blood-muscle-valve mechanics of the heart by an adaptive and parallel version of the immersed boundary method, Ph.D. Thesis, Courant Institute of Mathematical Sciences, New York University, 2005 also available from <http://www.math.nyu.edu/~griffith>.
[28] M.-C. Lai, Simulations of the flow past an array of circular cylinders as a test of the immersed boundary method, Ph.D. Thesis, Courant Institute of Mathematical Sciences, New York University, 1998.
[29] M.-C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2) (2000) 705–719.
[30] C.S. Peskin, The immersed boundary method, Acta Numer. 11 (2002) 479–517.
[31] L. Zhu, C.S. Peskin, Simulation of a flapping flexible filament in a flowing soap film by the immersed boundary method, J. Comput. Phys. 179 (2) (2002) 452–468.
[32] L. Zhu, C.S. Peskin, Interaction of two flapping filaments in a flowing soap film, Phys. Fluids 15 (7) (2003) 1954–1960.
[33] M.J. Berger, I. Rigoutsos, An algorithm for point clustering and grid generation, IEEE Trans. Syst. Man Cybernet. 21 (5) (1991) 1278–1286.
[34] F.H. Harlow, J.E. Welch, Numerical calculation of time-dependent viscous incompressible flow of fluid with free surface, Phys. Fluids 8 (12) (1965) 2182–2189.
[35] A.S. Almgren, J.B. Bell, W.G. Szymczak, A numerical method for the incompressible Navier–Stokes equations based on an approximate projection, SIAM J. Sci. Comput. 17 (2) (1996) 358–369.
[36] M.F. Lai, A projection method for reacting flow in the zero mach number limit, Ph.D. Thesis, University of California at Berkeley, 1993.
[37] A.-K. Tornberg, B. Engquist, Numerical approximations of singular source terms in differential equations, J. Comput. Phys. 200 (2) (2004) 462–488.
[38] L.H. Howell, J.B. Bell, An adaptive mesh projection method for viscous incompressible flow, SIAM J. Sci. Comput. 18 (4) (1997) 996–1013.
[39] J. Saltzman, An unsplit 3D upwind method for hyperbolic conservation laws, J. Comput. Phys. 115 (1) (1994) 153–168.
[40] D.L. Brown, R. Cortez, M.L. Minion, Accurate projection methods for the incompressible Navier–Stokes equations, J. Comput. Phys. 168 (2) (2001) 464–499.
[41] R.J. LeVeque, Finite Volume Methods for Hyperbolic Problems, Cambridge University Press, Cambridge, 2002.

[42] C. Tu, C.S. Peskin, Stability and instability in the computation of flows with moving immersed boundaries: a comparison of three methods, SIAM J. Sci. Stat. Comput. 13 (6) (1992) 1361–1376.
[43] A.A. Mayo, C.S. Peskin, An implicit numerical method for fluid dynamics problems with immersed elastic boundaries, in: A.Y. Cheer, C.P. van Dam (Eds.), Fluid Dynamics in Biology: Proceedings of an AMS-IMS-SIAM Joint Summer Research Conference, Contemporary Mathematics, vol. 140, American Mathematical Society, Providence, RI, USA, 1993, pp. 261–277.
[44] R.J. LeVeque, Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (3) (1997) 709–735.
[45] L. Lee, R.J. LeVeque, An immersed interface method for incompressible Navier–Stokes equations, SIAM J. Sci. Comput. 25 (3) (2003) 832–856.
[46] A.C. Guyton, J.E. Hall, Textbook of Medical Physiology, tenth ed., Saunders, Philadelphia, PA, USA, 2000.
[47] SAMRAI: structured adaptive mesh refinement application infrastructure. Available from: <http://www.llnl.gov/CASC/SAMRAI>.
[48] A.M. Wissink, R.D. Hornung, S.R. Kohn, S.G.S.N. Elliott, Large scale parallel structured AMR calculations using the SAMRAI framework, in: Proceedings of the SC01 Conference on High Performance Networking and Computing, Denver, CO, USA, 2001, also available as LLNL Technical Report UCRL-JC-144755.
[49] R.D. Hornung, S.R. Kohn, Managing application complexity in the SAMRAI object-oriented framework, Concurr. Comp.: Pract. E. 14 (5) (2002) 347–368.
[50] A.M. Wissink, D. Hysom, R.D. Hornung, Enhancing scalability of parallel structured AMR calculationsProceedings of the 17th ACM International Conference on Supercomputing (ICS03), ACM Press, New York, NY, USA, 2003, pp. 336–347, also available as LLNL Technical Report UCRL-JC-151791.
[51] S. Balay, K. Buschelman, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith, H. Zhang, PETSc Web page, http://www.mcs.anl.gov/petsc.
[52] 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 ANL-95/11 – Revision 2.1.5, Argonne National Laboratory, 2004.
[53] S. Balay, V. Eijkhout, W.D. Gropp, L.C. McInnes, B.F. Smith, Efficient management of parallelism in object oriented numerical software libraries, in: E. Arge, A.M. Bruaset, H.P. Langtangen (Eds.), Modern Software Tools in Scientific Computing, Birkhäuser, Basel, 1997, pp. 163–202.
[54] *hypre*: High performance preconditioners. Available from: <http://www.llnl.gov/CASC/hypre>.
[55] R.D. Falgout, U.M. Yang, *hypre*: a library of high performance preconditioners, in: P.M.A. Sloot, C.J.K. Tan, J.J. Dongarra, A.G. Hoekstra (Eds.), Computational Science – ICCS 2002 Part III, Lecture Notes in Computer Science, vol. 2331, Springer, Berlin, 2002, pp. 632–641, also available as LLNL Technical Report UCRL-JC-146175.