JOURNAL OF COMPUTATIONAL PHYSICS 81, 372–405 (1989)

A Three-Dimensional Computational Method for Blood Flow in the Heart I. Immersed Elastic Fibers in a Viscous Incompressible Fluid

CHARLES S. PESKIN AND DAVID M. MCQUEEN

Courant Institute of Mathematical Sciences,
New York University, 251 Mercer Street, New York, New York 10012

Received October 23, 1987; revised April 28, 1988

This paper describes the numerical solution of the 3-dimensional equations of motion of a viscous incompressible fluid that contains an immersed system of elastic fibers. Implementation details such as vectorization and the efficient use of external memory are discussed. The method is applied to the damped vibrations of a fiber-wound toroidal tube, and empirical evidence of convergence is presented. © 1989 Academic Press, Inc.

INTRODUCTION

The heart wall is made up of muscle fibers which are incompressible and neutrally buoyant in blood. These fibers do not run in random directions. On the contrary, at each point of the heart wall there is a well-defined fiber direction which varies smoothly from point to point. The anatomy of the cardiac fibers has been described in detail by Thomas [1] and by Streeter *et al.* [2]. The latter investigators report that a substantial part of the left ventricular wall consists of a nested family of toroidal shells on which the fibers follow geodesic curves. As for mechanical properties, there is considerable evidence that heart muscle can be realistically modeled as a time-dependent elastic material [3].

Motivated by these considerations, we have developed a computational method to solve the equations of motion of a viscous, incompressible fluid containing an immersed system of elastic or contractile fibers. In this paper, we describe the method and apply it to a test problem involving the vibrations of an immersed, fiber-wound, elastic, toroidal tube. Future papers in this series will consider a contractile tube and finally the heart itself.

The work described here is a direct generalization of our previous 2-dimensional work on blood flow in the heart. The development of the 2-dimensional computational method is described in [4, 5]. For applications to heart physiology and the design of prosthetic heart valves, see [6–9]; we anticipate similar applications of the 3-dimensional method once it is ready for use.

0021-9991/89 $3.00

While the methods are similar in principle, there are, of course, vast differences in scale between a 2-dimensional computation and the corresponding 3-dimensional computation. The method described here is implemented on supercomputers, and we shall discuss implementation issues such as vectorization, the potential for parallel computing, and the use of mass storage when the problem is too big to fit into central memory.

Equations of Motion

We consider a viscous, incompressible fluid containing an immersed system of elastic fibers. We think of these fibers as tenuous elements that cut through the fluid without displacing any volume or adding any mass. Their sole effect is to transmit an additional stress which always points in the fiber direction. We also assume that the fluid sticks to the fibers (the no-slip condition of a viscous fluid). In the region of space where the fibers are, the material present is a composite: fluid + fibers; in the rest of space the material is simply fluid. The stress tensor of the composite material takes the form

$$\sigma'_{ij} = -p\,\delta_{ij} + \mu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right) + T'\tau_i\tau_j, \tag{1}$$

where $\mathbf{u}$ is the fluid velocity, p is the fluid pressure, μ is the (constant) fluid viscosity, T' is the fiber tension (force per unit cross-sectional area of composite: the fibers themselves have no cross-sectional area), and $\boldsymbol{\tau}$ is the unit tangent to the fibers.

The assumption that the stress tensor in cardiac muscle has the form given by Eq. (1) (but without the viscous term) has recently been used by Peskin [18] to derive from first principles certain key features of the observed fiber architecture of the left ventricular wall. The same stress tensor has also been used by Chadwick [19] in a model of ventricular mechanics.

According to the assumptions stated above, the composite material is necessarily incompressible (since its volume is made up of incompressible fluid) and its constant density ρ is the same as that of the fluid (since the fibers themselves are massless). This is realistic, since heart muscle is known to be incompressible and neutrally buoyant in blood. According to our assumptions, the viscosity of the composite is also the same as that of the fluid. We do not know whether this assumption is realistic or not, but the principal viscous effect in heart muscle is contained in the velocity dependence of the active force, which points in the fiber direction. Such anisotropic viscosity could be incorporated, if desired, by making the fiber stress T' dependent not only on the fiber strain but also on the rate of fiber strain.

Note that the word "fiber" as used in the foregoing, does not correspond to the use of that word in the expression "muscle fiber." In fact, the muscle fiber is a physical chunk of material with mass and volume. We model such material as a composite with massive incompressible fluid as one component and a massless system of force-bearing fibers as the other.

Since the fiber tension will be computed from the fiber strain, we need a Lagrangian description of the fibers. Accordingly, let

$$\mathbf{x} = \mathbf{X}(q, r, s, t) \tag{2}$$

be a system of curvilinear coordinates aligned with the fibers in such a way that:

(i) fixed q, r, s, designates a material point;

(ii) fixed q, r designates a fiber;

(iii) s measures arc length along a fiber in some reference configuration (not necessarily an unstressed configuration).

Let $T(q, r, s, t)\, dq\, dr$ be the force transmitted by the bundle of fibers $dq\, dr$. (Note the distinction between T and T'; the latter is the force per unit cross-sectional area.) We assume that T is determined by a generalized Hooke's law of the form

$$T = \sigma\left(\left|\frac{\partial \mathbf{X}}{\partial s}\right|; q, r, s, t\right). \tag{3}$$

Consider two nearby points on a given fiber ($q, r =$ constant). The distance between these points is $|d\mathbf{X}| = |(\partial \mathbf{X}/\partial s)\, ds|$. The distance between the same two points in the reference configuration is $|ds|$. Therefore, the first argument of σ measures the ratio of these distances. (If the reference configuration were unstressed, it would be appropriate to call $|\partial \mathbf{X}/\partial s| - 1$ the "strain.") The remaining arguments in σ allow for spatial inhomogeneity and for the elastic properties of the material to be time dependent. The time dependence is needed to model active, contractile muscle.

By hypothesis, the direction associated with the fiber stress T is always that of the fibers themselves. This direction is indicated by the unit tangent

$$\boldsymbol{\tau} = \frac{\partial \mathbf{X}/\partial s}{|\partial \mathbf{X}/\partial s|}. \tag{4}$$

Now consider the bundle of fibers corresponding to some region Ω of the q, r parameter plane. In particular, consider the segment of this bundle lying between $s = a$ and $s = b$. The fiber force acting on this segment from outside the segment is

$$\iint_{\Omega} [(T\boldsymbol{\tau})(q, r, b, t) - (T\boldsymbol{\tau})(q, r, a, t)]\, dq\, dr$$

$$= \iint_{\Omega} \int_a^b \frac{\partial}{\partial s}(T\boldsymbol{\tau})\, ds\, dq\, dr. \tag{5}$$

Since the fibers themselves are massless, the force given by either side of Eq. (5) must be transmitted to the fluid occupying the same region of space as the bundle of fibers in question. Since Ω, a, b are all arbitrary, this shows that the local density of force (with respect to q, r, s) applied by the fibers to the fluid is given by

$$\mathbf{f}=\frac{\partial}{\partial s}(T\boldsymbol{\tau}). \tag{6}$$

Note that $\mathbf{f}$ can be resolved into two orthogonal components, $(\partial T/\partial s)\boldsymbol{\tau}$ and $T(\partial\boldsymbol{\tau}/\partial s)$. The first of these is tangent to the fibers and the second is in the direction of the principal normal. (There is no component of force in the direction of the binormal.)

We shall write the fluid equations in Cartesian coordinates. Therefore, we need a Cartesian force density $\mathbf{F}$ corresponding to $\mathbf{f}$. That is, we seek $\mathbf{F}(\mathbf{x}, t)$ such that

$$\int_{\mathbf{X}(R,t)} \mathbf{F}\,d\mathbf{x}=\int_R \mathbf{f}\,dq\,dr\,ds, \tag{7}$$

where R is an arbitrary region of (q, r, s) space and where $\mathbf{X}(R, t)$ is the image of R under the mapping $(q, r, s)\rightarrow \mathbf{x}=\mathbf{X}(q, r, s, t)$. (Note that $d\mathbf{x}$ stands for the Cartesian volume element $dx_1\,dx_2\,dx_3$. Also note that t is merely a parameter in Eq. (7).)

The standard way to relate $\mathbf{F}$ and $\mathbf{f}$ is to introduce the Jacobian determinant

$$J(q, r, s)=\left(\frac{\partial\mathbf{X}}{\partial q}\times\frac{\partial\mathbf{X}}{\partial r}\right)\cdot\frac{\partial\mathbf{X}}{\partial s}. \tag{8}$$

(J is independent of t because the motion described by $\mathbf{X}(q, r, s, t)$ is incompressible.) Then $d\mathbf{x}=J\,dq\,dr\,ds$, and

$$\mathbf{F}(\mathbf{X}(q, r, s, t), t)\,J(q, r, s)=\mathbf{f}(q, r, s, t) \tag{9}$$

which implicitly defines the function $\mathbf{F}(\mathbf{x}, t)$.

Alternatively, we can avoid all reference to the Jacobian and write an explicit formula for $\mathbf{F}(\mathbf{x}, t)$ in terms of the Dirac δ-function:

$$\mathbf{F}(\mathbf{x}, t)=\int \mathbf{f}(q, r, s, t)\,\delta(\mathbf{x}-\mathbf{X}(q, r, s, t))\,dq\,dr\,ds. \tag{10}$$

Here the integral is over the entire system of fibers and δ stands for the 3-dimensional *delta* function: $\delta(\mathbf{x})=\delta(x_1)\,\delta(x_2)\,\delta(x_3)$.

To verify that Eq. (10) is equivalent to Eq. (7), which defines $\mathbf{F}$, integrate both sides of Eq. (10) over the arbitrary region $\mathbf{X}(R, t)$. On the right-hand side, interchange the order of integration and note that

$$\int_{\mathbf{X}(R,t)} \delta(\mathbf{x}-\mathbf{X}(q, r, s, t))\, d\mathbf{x} = \begin{cases} 1, & \mathbf{X}(q, r, s, t) \in \mathbf{X}(R, t) \\ 0, & \text{otherwise} \end{cases}$$

$$= \begin{cases} 1, & (q, r, s) \in R \\ 0, & \text{otherwise.} \end{cases}$$

The result is precisely Eq. (7). This shows that Eq. (10) implies Eq. (7). The equivalence of the two equations then follows from the arbitrariness of the region of integration.

There are several advantages of Eq. (10) in comparison with Eq. (9). Two of these have already been mentioned above. Equation (10) is an explicit formula for $\mathbf{F}(\mathbf{x}, t)$ and it avoids reference to the Jacobian determinant. A further advantage is that Eq. (10) makes sense even in the case where the fibers are confined to a surface (such as a heart valve leaflet.) In such a case there is one less parameter: the fiber configuration is given by $\mathbf{X}(q, s, t)$, and the formula corresponding to Eq. (10) is

$$\mathbf{F}(\mathbf{x}, t) = \int \mathbf{f}(q, s, t)\, \delta(\mathbf{x}-\mathbf{X}(q, s, t))\, dq\, ds. \tag{11}$$

In this case we have only a double integral, but the δ-function is still 3-dimensional. The result is that $\mathbf{F}$ is not an ordinary function but a distribution (δ-function layer) with support on the surface occupied by the fibers. The final advantage of Eq. (10) is that it leads directly to a numerical procedure for coupling the fibers and the fluid (see below).

Having derived a Cartesian expression for the density of the force applied by the fibers to the fluid, we may now write down equations of motion for the system as a whole:

$$\rho\left(\frac{\partial \mathbf{u}}{\partial t} + \mathbf{u}\cdot\nabla\mathbf{u}\right) + \nabla p = \mu\nabla^2\mathbf{u} + \mathbf{F} \tag{12}$$

$$\nabla\cdot\mathbf{u} = 0 \tag{13}$$

$$\mathbf{F}(\mathbf{x}, t) = \int \mathbf{f}(q, r, s, t)\, \delta(\mathbf{x}-\mathbf{X}(q, r, s, t))\, dq\, dr\, ds \tag{14}$$

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

$$= \int \mathbf{u}(\mathbf{x}, t)\, \delta(\mathbf{x}-\mathbf{X}(q, r, s, t))\, d\mathbf{x} \tag{15}$$

$$\mathbf{f}=\frac{\partial}{\partial s}(T\boldsymbol{\tau}) \tag{16}$$

$$T=\sigma\left(\left|\frac{\partial \mathbf{X}}{\partial s}\right|; q, r, s, t\right) \tag{17}$$

$$\boldsymbol{\tau}=\frac{\partial \mathbf{X}/\partial s}{|\partial \mathbf{X}/\partial s|}. \tag{18}$$

These equations fall into three groups. Equations (12)–(13) are the Navier–Stokes equations of a viscous incompressible fluid with density ρ and viscosity μ. The independent variables in these equations are the position $\mathbf{x}$ and the time t. The fluid velocity and pressure are given by $\mathbf{u}(\mathbf{x}, t)$ and $p(x, t)$. The external force density $\mathbf{F}(\mathbf{x}, t)$ stands for the force per unit volume applied by the fibers to the fluid in which they are immersed.

Next, consider Eqs. (16)–(18). Here the independent variables are the Lagrangian fiber parameters q, r, s, and the time t. Substituting Eqs. (17)–(18) into Eq. (16), we see that these equations amount to a recipe for computing the fiber force density (with respect to q, r, s), $\mathbf{f}(\ ,\ ,\ , t)$, in terms of the fiber configuration $\mathbf{X}(\ ,\ ,\ , t)$.

Finally, there is the middle group, Eqs. (14)–(15), which connect functions of q, r, s and functions of $\mathbf{x}$. The first of these (Eq. (14)) expresses the fiber force as a sum (integral) of δ-function forces localized at the fiber points. The second (Eq. (15)) is the no-slip condition of a viscous fluid. It merely asserts that the fibers move at the local fluid velocity. Note that this is the equation of motion of the fibers and not a constraint on the fluid motion, since the motion of the fibers is here unknown. The second form of Eq. (15), in which the δ-function appears, is written down to emphasize the symmetry between Eqs. (14) and (15). It will also prove useful in the construction of the numerical method, see below.

Note that Eqs. (12)–(18) have the form of a first-order system in the state variables $\mathbf{X}(\ , t)$ and $\mathbf{u}(\ , t)$. To show this, we explain how $(\partial \mathbf{X}/\partial t)(\ , t)$ and $(\partial \mathbf{u}/\partial t)(\ , t)$ are determined by $\mathbf{X}(\ , t)$ and $\mathbf{u}(\ , t)$. First, note that Eq. (15) is an explicit formula for $\partial \mathbf{X}/\partial t$ in terms of $\mathbf{X}(\ , t)$ and $\mathbf{u}(\ , t)$. As for $\partial \mathbf{u}/\partial t$, we proceed as follows: Starting from the fiber configuration $\mathbf{X}(\ , t)$, we use Eqs. (16)–(18) to find T, $\boldsymbol{\tau}$, and then $\mathbf{f}$. With $\mathbf{f}$ known, Eq. (14) determines $\mathbf{F}$. Then with $\mathbf{F}$ and $\mathbf{u}$ known, the next step is to determine p. This can be done by solving the Poisson equation

$$-\nabla^2 p=\rho\nabla\cdot(\mathbf{u}\cdot\nabla\mathbf{u})-\nabla\cdot\mathbf{F} \tag{19}$$

which can be derived from Eqs. (12)–(13). Finally, with $\mathbf{F}$, $\mathbf{u}$, and p known, Eq. (12) becomes a formula for $\partial \mathbf{u}/\partial t$.

Incidentally, the Poisson equation for the pressure (Eq. (19)) explains how it comes about that the local fiber forces produce instantaneous non-local effects. These effects are mediated by the pressure field.

We conclude this section with a derivation of the relation between the two tensions T and T' and a demonstration that the force density $\mathbf{F}$ is, in fact, the divergence of a fiber stress tensor of the form $T'\tau_i\tau_j$ (see Eq. (1)). Let $\mathbf{F}'$ be defined by such a divergence expression:

$$\begin{aligned} F_i' &= \sum_{j=1}^{3} \frac{\partial}{\partial x_j}(T'\tau_i\tau_j) \\ &= \sum_{j=1}^{3} T'\tau_j \frac{\partial}{\partial x_j}(\tau_i) + \tau_i \frac{\partial}{\partial x_j}(T'\tau_j). \end{aligned} \tag{20}$$

Then

$$\begin{aligned} \mathbf{F}' &= T'\boldsymbol{\tau} \bullet \nabla\boldsymbol{\tau} + \boldsymbol{\tau}\nabla \bullet (T'\boldsymbol{\tau}) \\ J\mathbf{F}' &= \frac{JT'}{|\partial\mathbf{X}/\partial s|}\frac{\partial\mathbf{X}}{\partial s} \bullet \nabla\boldsymbol{\tau} + J\boldsymbol{\tau}\nabla \bullet \left(\frac{JT'}{|\partial\mathbf{X}/\partial s|}\frac{1}{J}\frac{\partial\mathbf{X}}{\partial s}\right), \end{aligned} \tag{21}$$

where J is given by Eq. (8). Now let

$$T = \frac{JT'}{|\partial\mathbf{X}/\partial s|}. \tag{22}$$

(This is the desired relation between T and T'.) Then

$$J\mathbf{F}' = T\frac{\partial\boldsymbol{\tau}}{\partial s} + \boldsymbol{\tau}J\nabla \bullet \left(T\frac{1}{J}\frac{\partial\mathbf{X}}{\partial s}\right). \tag{23}$$

At this point, we need the identity

$$\nabla \bullet \left(\frac{1}{J}\frac{\partial\mathbf{X}}{\partial s}\right) = 0 \tag{24}$$

which is easily proved in integral form by considering a volume which is the image of an arbitrary cylinder in q, r, s space: $(q, r) \in \Omega$, $s \in (a, b)$. For such a volume with outward normal $\mathbf{n}$, $(\partial\mathbf{X}/\partial s) \bullet \mathbf{n} = 0$ on the surfaces which are images of the sides of the cylinder. On the images of the ends we have

$$\mathbf{n}\,da = \pm\left(\frac{\partial\mathbf{X}}{\partial q} \times \frac{\partial\mathbf{X}}{\partial r}\right) dq\,dr \tag{25}$$

with the positive sign on the image of $s = b$ and the negative sign on the image of $s = a$. Therefore

$$\frac{1}{J}\frac{\partial\mathbf{X}}{\partial s} \bullet \mathbf{n}\,da = \pm dq\,dr \tag{26}$$

and the total contribution from the two ends of the cylinder is zero. This completes the proof of Eq. (24). Using this result in Eq. (23), we find

$$\begin{aligned} J\mathbf{F}' &= T\frac{\partial \boldsymbol{\tau}}{\partial s} + \boldsymbol{\tau} J \frac{1}{J}\frac{\partial \mathbf{X}}{\partial s} \bullet \nabla T \\ &= T\frac{\partial \boldsymbol{\tau}}{\partial s} + \boldsymbol{\tau}\frac{\partial T}{\partial s} = \frac{\partial}{\partial s}(T\boldsymbol{\tau}) = \mathbf{f}. \end{aligned} \tag{27}$$

It follows by comparision with Eq. (9) that $\mathbf{F}' = \mathbf{F}$ as required.

Numerical Method

In this section we outline the algorithm used to solve the equations of motion. The details of implementation will be discussed in subsequent sections.

As remarked above, the equations of motion may be partitioned into three groups: fluid equations, fiber equations, and interaction equations. This partition is reflected in the computational method. The fluid equations are solved on a regular cubic lattice, which is not distorted or modified in any way by the presence of the fibers. At the outer boundaries of the cube, we impose periodic boundary conditions so that all points are effectively interior points.

The fiber continuum is modeled by a (moving) collection of discrete computational fibers, each of which takes the form of a closed space curve, and each of which is represented in the computation by a finite collection of Lagrangian points that are equally spaced with respect to the fiber parameter s. The fiber points can be anywhere in space: they are not required to coincide with the lattice points of the fluid computation. This makes possible a smooth representation of the fibers and their motion, but it also introduces the difficulty that the computed fluid velocity $\mathbf{u}$ is not defined at the fiber points and that the computed fiber force density $\mathbf{f}$ is not defined at the points of the computational lattice that is used to solve the fluid equations. This difficulty is overcome through the introduction of a sufficiently smooth approximation to the Dirac δ-function, which mediates the interaction between the computational fibers and the computational fluid.

We begin by introducing the notation that will be used to describe the numerical method. Let the fluid domain be a cubic box of side L with periodic boundary conditions. Let N be the number of lattice points in each direction, and let $h = L/N$. Let $\mathbf{e}_s$, $s = 1, 2, 3$, be the unit vectors in each of the three coordinate directions. The components of a vector will be denoted by scalar quantities with subscripts. Thus $\mathbf{u} = \sum_{s=1}^{3} u_s \mathbf{e}_s$. The computational lattice consists of points of the form $\mathbf{x} = \mathbf{j}h$, where $\mathbf{j} = \sum_{s=1}^{3} j_s \mathbf{e}_s$ and where j_s is an integer that satisfies $0 \leqslant j_s \leqslant N-1$. Periodicity is built into the equations by adopting the convention that all functions of $\mathbf{x}$ are periodic with period L in each of the three coordinate directions:

$$\phi(\mathbf{x} + L\mathbf{e}_s) = \phi(\mathbf{x}), \qquad s = 1, 2, 3. \tag{28}$$

This is the same as saying that all arithmetic involving any of the lattice coordinates j_s is understood to be modulo N.

As in Eq. (28), we shall retain the continuous notation for the space variable $\mathbf{x}$; it should be understood throughout that $\mathbf{x}$ is restricted to points of the computational lattice as defined above. Some spatial difference operators that we shall use are the following:

$$(D_s^0\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x}+h\mathbf{e}_s)-\phi(\mathbf{x}-h\mathbf{e}_s)}{2h} \sim \frac{\partial\phi}{\partial x_s} \tag{29}$$

$$(D_s^+\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x}+h\mathbf{e}_s)-\phi(\mathbf{x})}{h} \sim \frac{\partial\phi}{\partial x_s} \tag{30a}$$

$$(D_s^-\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x})-\phi(\mathbf{x}-h\mathbf{e}_s)}{h} \sim \frac{\partial\phi}{\partial x_s} \tag{30b}$$

$$(D_s^+D_s^-\phi)(\mathbf{x}) = \frac{\phi(\mathbf{x}+h\mathbf{e}_s)+\phi(\mathbf{x}-h\mathbf{e}_s)-2\phi(\mathbf{x})}{h^2} \sim \frac{\partial^2\phi}{\partial x_s^2} \tag{31}$$

$$\mathbf{G}\phi = \sum_{s=1}^{3} \mathbf{e}_s D_s^0\phi \sim \operatorname{grad}\phi \tag{32}$$

$$\mathbf{D}\cdot\mathbf{u} = \sum_{s=1}^{3} D_s^0 u_s \sim \operatorname{div}\mathbf{u}. \tag{33}$$

Time will proceed in steps of duration Δt, and we shall use the superscript notation $\mathbf{u}^n(\mathbf{x}) = \mathbf{u}(\mathbf{x}, n\,\Delta t)$.

The fiber notation is as follows. Let N_f be the number of computational fibers and let N_p be the number of computational points on each fiber. (We assume for convenience that N_p is the same for all fibers; this also permits easy vectorization, see below. If it is not practical to have N_p constant, then the fibers should be partitioned into groups, with N_p constant within each group.) Let the fiber domain in parameter space be $(q, r) \in \Omega$, $0 \leqslant s \leqslant L_f$, where L_f is the length of the fibers in the reference configuration. Since the fibers are closed curves, $s = 0$ is equivalent to $s = L_f$. Let a_f be the area of the domain Ω in the q, r plane, and let $\Delta a = a_f/N_f$, $\Delta s = L_f/N_p$.

A reasonable choice for N_p is such that $\Delta s = h/2$, and a reasonable choice for N_f is such that the number of fibers per unit area (in the reference configuration) is $4/h^2$. The latter choice makes $\Delta a = O(h^2)$. With these choices, the density of fiber points will be $8/h^3$ in the region occupied by the fibers, and this volume density will not change appreciably because the fiber points move in an incompressible flow. The fiber resolution proposed here is sufficient to prevent significant leakage of fluid between the fiber points, and little would be gained by refining the fiber resolution further since the overall resolution would be limited in any case by the computational lattice of the fluid.

Even though the volume density of the fiber points should remain constant, as described above, there is still the possibility that large deformations will result in distances substantially larger than $h/2$ between adjacent points on a single fiber or between adjacent fibers. Of course, fiber stiffness opposes the first type of deformation. Moreover, in a well-constructed wall such as that of the heart, separation between fibers in any particular layer is opposed by the fibers in nearby layers, since the fiber angle varies from one layer to the next [1, 2, 18, 19]. Nevertheless, large deformations may occur (especially in the cross-fiber direction), and this could result in a leaky wall. To monitor this, we keep track of the volumes of different chambers in the computation. Volume conservation is excellent (see Results).

Let $\mathbf{X}_{lk}^n$ denote the position of the kth point on the lth fiber at $t = n\,\Delta t$. The subscript $l = 1, ..., N_f$ and the subscript $k = 0, ..., (N_p - 1)$. Arithmetic on k is modulo N_p.

The discrete tangent vector is denoted

$$\boldsymbol{\tau}_{l,k+1/2} = \frac{\mathbf{X}_{l,k+1} - \mathbf{X}_{l,k}}{|\mathbf{X}_{l,k+1} - \mathbf{X}_{l,k}|} \tag{34}$$

and the discrete fiber tension is given by

$$T_{l,k+1/2} = \sigma\left(\frac{|\mathbf{X}_{l,k+1} - \mathbf{X}_{l,k}|}{\Delta s}\right). \tag{35}$$

(We assume here a uniform and time-independent stress–strain relation. In the general case, σ would depend on l, k, and n.) Finally, the fiber force density $\mathbf{f}_{lk}$ is given by

$$\mathbf{f}_{lk} = \frac{T_{l,k+1/2}\boldsymbol{\tau}_{l,k+1/2} - T_{l,k-1/2}\boldsymbol{\tau}_{l,k-1/2}}{\Delta s}. \tag{36}$$

Equations (34)–(36) define $\mathbf{f}_{lk}$ as functions of $(\cdots \mathbf{X}_{lk'} \cdots)$. The form of these functions is conveniently summarized as follows. Let

$$\mathbf{g}(\mathbf{r}) = \frac{\mathbf{r}}{|\mathbf{r}|}\,\sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right). \tag{37}$$

Then (since $\mathbf{g}(-\mathbf{r}) = -\mathbf{g}(\mathbf{r})$),

$$\mathbf{f}_{lk}(\cdots \mathbf{X}_{lk'} \cdots) = \frac{\mathbf{g}(\mathbf{X}_{l,k+1} - \mathbf{X}_{lk}) + \mathbf{g}(\mathbf{X}_{l,k-1} - \mathbf{X}_{lk})}{\Delta s}. \tag{38}$$

This formula puts in evidence the fact that $\mathbf{f}_{lk}$ depends only on $\mathbf{X}_{lk}$ and $\mathbf{X}_{l,k\pm 1}$.

We now consider the computational apparatus that connects the fluid lattice and the fiber points. As explained above, this involves an approximation to the Dirac δ-function. Let

$$\delta_h(\mathbf{x}) = \delta_h(x_1)\,\delta_h(x_2)\,\delta_h(x_3), \tag{39}$$

where

$$\delta_h(x)=\begin{cases}\dfrac{1}{4h}\left(1+\cos\dfrac{\pi x}{2h}\right), & |x|\leqslant 2h\\ 0, & |x|\geqslant 2h.\end{cases} \tag{40}$$

The motivation for this particular choice of δ-function has been given elsewhere [4].

We use δ_h to interpolate the fluid velocity to a fiber point:

$$\mathbf{U}_{lk}=\sum_{\mathbf{x}}\mathbf{u}(\mathbf{x})\,\delta_h(\mathbf{x}-\mathbf{X}_{lk})\,h^3 \tag{41}$$

and also to spread the fiber forces out on the computational lattice:

$$\mathbf{F}(\mathbf{x})=\sum_{lk}\mathbf{f}_{lk}\,\delta_h(\mathbf{x}-\mathbf{X}_{lk})\,\Delta a\,\Delta s. \tag{42}$$

The fact that we use the same approximate δ-function in both of these equations makes it possible to derive the power identity:

$$\begin{aligned}\sum_{lk}\mathbf{f}_{lk}\bullet\mathbf{U}_{lk}\,\Delta a\,\Delta s&=\sum_{\mathbf{x}}\mathbf{u}(\mathbf{x})\bullet\sum_{lk}\mathbf{f}_{lk}\,\delta_h(\mathbf{x}-\mathbf{X}_{lk})\,\Delta a\,\Delta s\,h^3\\&=\sum_{\mathbf{x}}\mathbf{u}(\mathbf{x})\bullet\mathbf{F}(\mathbf{x})\,h^3.\end{aligned} \tag{43}$$

This shows that the power applied to the fluid at the fiber points equals the power felt by the fluid on its computational lattice.

We are now in a position to describe the numerical method, which is an algorithm for computing $\mathbf{X}^{n+1}$, $\mathbf{u}^{n+1}$, given $\mathbf{X}^n$, $\mathbf{u}^n$. The first step is to evaluate the fiber forces. This must be done implicitly to avoid numerical instability when the computation is performed with reasonable time steps. The implicit equations that we use to define the fiber forces are as follows. Let $(\cdots\mathbf{X}_{l,k}^{n+1,*}\cdots)$ be defined as the solution of the nonlinear system

$$\mathbf{X}_{lk}^{n+1,*}=\mathbf{X}_{lk}^{n}+\Delta t\,\mathbf{U}_{lk}^{n}+(\Delta t)^2\,\lambda\mathbf{f}_{lk}(\cdots\mathbf{X}_{l'k'}^{n+1,*}\cdots), \tag{44}$$

where

$$\mathbf{U}_{lk}^{n}=\sum_{\mathbf{x}}\mathbf{u}^n(\mathbf{x})\,\delta_h(\mathbf{x}-\mathbf{X}_{lk}^{n})\,h^3 \tag{45}$$

$$\lambda=\frac{1}{\rho}(\Delta a)(\Delta s)\left(\frac{3}{8h}\right)^3\beta \tag{46}$$

and where β is an adjustable parameter the choice of which will be discussed below. Note that the coefficient of β in Eq. (46) is $O(1)$, since $(\Delta a)(\Delta s)=O(h^3)$. Once the system of Eqs. (44) have been solved for the unknowns $\mathbf{X}_{lk}^{n+1,*}$, we evaluate the corresponding fiber forces

$$\mathbf{f}_{lk}^{n+1,*} = \mathbf{f}_{lk}(\cdots \mathbf{X}_{lk'}^{n+1,*} \cdots) \tag{47}$$

and apply them to the fluid. The method used to solve the nonlinear system Eq. (44) will be described under implementation, below.

The foregoing approach to the computation of the fiber forces is motivated by the well-known stability of the backward-Euler method. In this method, the forces would be calculated from the fiber configuration at the end of the time step. This configuration depends, of course, on the fiber forces themselves, and the resulting fixed-point problem is very complicated, since it involves the coupling of fiber points to each other through the fluid. Our approach here is to ignore the fluid-mediated coupling (temporarily) and to compute the fiber forces by applying the backward-Euler method under the approximation that each element of force $\mathbf{f}_{lk}$ influences only the point $\mathbf{X}_{lk}$ at which that element of force is applied. The coefficient $\lambda(\Delta t)^2$ in Eq. (44) estimates the displacement generated by unit $\mathbf{f}_{lk}$ in one time step.

We emphasize the following points with regard to the implicit computation of the fiber forces. First, the approximation of neglecting the fluid-mediated interactions is only made for purposes of computing the fiber forces. Once $\mathbf{f}^*$ has been computed, $\mathbf{X}^*$ is discarded. Then $\mathbf{f}^*$ is applied to the fluid, and the fiber points move at the local fluid velocity. Each element of the fiber force has the opportunity to influence all of the fiber points during this computation. Second, the use of $\mathbf{f}^*$ as opposed to $\mathbf{f}^n$ is for stability, not accuracy. As far as accuracy is concerned, the scheme is consistent because $\mathbf{f}^* \to \mathbf{f}^n$ as $\Delta t \to 0$. Note that this is true for any choice of β such that $\beta(\Delta t)^2 \to 0$ as $\Delta t \to 0$. The actual value of β must be chosen by trial and error. Large β stabilizes the computation by damping out the high spatial frequency components of the fiber force. In the limit $\beta \to 0$, the computation reduces to an explicit scheme in which the fiber forces are computed from the configuration $\mathbf{X}^n + \Delta t\, \mathbf{U}^n$. Finally, we emphasize that nothing has been proved concerning the approximate backward-Euler method outlined above. In practice, the approach that we have described extends the range of stability of the method, but it does not make the method unconditionally stable.

Once the forces $\mathbf{f}_{lk}^{n+1,*}$ have been computed, we use them to define fluid forces $\mathbf{F}^{n+1,*}(\mathbf{x})$ in the manner indicated by Eq. (42). Specifically, we use

$$\mathbf{F}^{n+1,*}(\mathbf{x}) = \sum_{lk} \mathbf{f}_{lk}^{n+1,*}\, \delta_h(\mathbf{x} - \mathbf{X}_{lk}^n)\, \Delta a\, \Delta s. \tag{48}$$

Note that the argument of δ_h involves $\mathbf{X}^n$, not $\mathbf{X}^{n+1}$ or $\mathbf{X}^{n+1,*}$. The reason for this is that $\mathbf{X}^{n+1,*}$ is not yet known and that $\mathbf{X}^{n+1,*}$ involves displacements from $\mathbf{X}^n$ which were not generated by an incompressible flow.

The next step in the algorithm is to solve the Navier–Stokes equations and update the fluid velocity field for one time step under the influence of the applied forces $\mathbf{F}^{n+1,*}$. This is done by Chorin's projection method [10, 11], which is an implicit, fractional-step method. The subroutine that implements this method (see next section for details) makes no explicit reference to the fibers at all: its inputs are the lattice function $\mathbf{u}^n(\mathbf{x})$ and $\mathbf{F}^{n+1,*}(\mathbf{x})$; its outputs are the lattice functions $\mathbf{u}^{n+1}(\mathbf{x})$ and $p^{n+1}(\mathbf{x})$. The procedure is as follows. First set

$$\mathbf{u}^{n+1,0} = \mathbf{u}^n + \frac{\Delta t}{\rho}\mathbf{F}^{n+1,*}. \tag{49}$$

Then solve successively the following linear systems for $s = 1, 2, 3$:

$$\rho\left(\frac{\mathbf{u}^{n+1,s} - \mathbf{u}^{n+1,s-1}}{\Delta t} + u_s^n D_s^0 \mathbf{u}^{n+1,s}\right) = \mu D_s^+ D_s^- \mathbf{u}^{n+1,s}. \tag{50}$$

Finally, solve simultaneously, the following linear system for $\mathbf{u}^{n+1}$, p^{n+1}:

$$\rho\frac{\mathbf{u}^{n+1} - \mathbf{u}^{n+1,3}}{\Delta t} + \mathbf{G}p^{n+1} = 0 \tag{51}$$

$$\mathbf{D} \bullet \mathbf{u}^{n+1} = 0. \tag{52}$$

In the foregoing scheme, various forces are applied separately. In Eq. (49), the fiber forces are applied to the fluid. In Eq. (50), the viscous and convection forces are applied. For each space direction $s = 1, 2, 3$, only the force terms arising from differences (derivatives) in that space direction are considered. (Note that Eq. (50) is a vector equation; the s-differences are applied to all three components of the velocity vector.) Note further that Eq. (50) is an implicit definition of $\mathbf{u}^{n+1,s}$ in terms of $\mathbf{u}^{n+1,s-1}$. The equations for $\mathbf{u}^{n+1,s}$ have the character of periodic tridiagonal systems running in the coordinate direction parallel to $\mathbf{e}_s$. There is no coupling in the other two space directions. The convection velocity in Eq. (50) is $\mathbf{u}^n$ (with components u_s^n), which is known at the beginning of the time step. (Since $\mathbf{u}^n$ is known, Eq. (50) is a linear system.) The reason for using $\mathbf{u}^n$ (as opposed to the most recently computed velocity $\mathbf{u}^{n+1,s-1}$) is that $\mathbf{u}^n$ is divergence-free ($\mathbf{D} \bullet \mathbf{u}^n = 0$) and hence is a legitimate fluid motion. Each of the intermediate velocity fields $\mathbf{u}^{n+1,1}$, $\mathbf{u}^{n+1,2}$, and $\mathbf{u}^{n+1,3}$ has divergence which is $O(\Delta t)$.

In fact, the divergence of $\mathbf{u}^{n+1,3}$ becomes the source of the pressure that is computed by solving the final linear system, Eqs. (51)–(52). To see this, apply $\mathbf{D} \bullet$ to both sides of Eq. (51) and make use of Eq. (52). The result is

$$-\mathbf{D} \bullet \mathbf{G}p^{n+1} = -\rho(\Delta t)^{-1}(\mathbf{D} \bullet \mathbf{u}^{n+1,3}). \tag{53}$$

This is a discrete Poisson equation for the pressure. Once p^{n+1} has been found by solving this Poisson equation, it is substituted into Eq. (51) to find $\mathbf{u}^{n+1}$. This completes the description of the computations performed by the Navier–Stokes solver during one time step.

It remains only to update the fiber position. This is done by interpolating $\mathbf{u}^{n+1}$ to the old fiber positions $\mathbf{X}^n$ and moving the fibers accordingly:

$$\mathbf{X}_{lk}^{n+1} = \mathbf{X}_{lk}^{n} + \Delta t \sum_{\mathbf{x}} \mathbf{u}^{n+1}(\mathbf{x})\, \delta_h(\mathbf{x} - \mathbf{X}_{lk}^{n})\, h^3. \tag{54}$$

Since we have a new velocity $\mathbf{u}^{n+1}$ and a new fiber configuration $\mathbf{X}_{lk}^{n+1}$, the time step is complete.

Implementation: Solution of Implicit Equations

The algorithm of the previous section is incompletely specified, because we have not explained how to solve the nonlinear systems of equations for $(\mathbf{X}^{n+1,*}, \mathbf{f}^{n+1,*})$ or the linear systems for $\mathbf{u}^{n+1,1}$, $\mathbf{u}^{n+1,2}$, $\mathbf{u}^{n+1,3}$, and $(\mathbf{u}^{n+1}, p^{n+1})$.

We begin with the nonlinear systems for the fiber forces, Eqs. (44)–(47). Note that there is a separate system for each fiber. Therefore, we drop the fiber index l in the following. Since we are here considering a single time step, we also drop the time index n. The quantity $(\mathbf{X}^n + \Delta t\, \mathbf{U}^n)$ is known at the beginning of the time step. Here, we refer to this quantity as $\mathbf{X}^0$. Also, we redefine λ here so that it includes the factor $(\Delta t)^2$. With these changes in notation, the nonlinear system takes the form

$$\mathbf{X}_k^* = \mathbf{X}_k^0 + \lambda \mathbf{f}_k(\cdots \mathbf{X}_{k'}^* \cdots), \tag{55}$$

where $k = 0 \cdots N_p - 1$ and where arithmetic on k is understood to be modulo N_p, since the fibers are closed curves. The function f in Eq. (55) is defined as follows:

$$\mathbf{f}_k(\cdots \mathbf{X}_{k'} \cdots) = \frac{\mathbf{g}(\mathbf{X}_{k+1} - \mathbf{X}_k) + \mathbf{g}(\mathbf{X}_{k-1} - \mathbf{X}_k)}{\Delta s}, \tag{56}$$

where

$$\mathbf{g}(\mathbf{r}) = \frac{\mathbf{r}}{|\mathbf{r}|}\, \sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right) \tag{57}$$

and where σ is the fiber stress.

We assume that σ is a smooth function with the properties:

$$\sigma(0) = 0 \tag{58}$$

$$\sigma'(0) = 0 \tag{59}$$

$$\sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right) \geqslant 0 \tag{60}$$

$$\sigma'\left(\frac{|\mathbf{r}|}{\Delta s}\right) \geqslant 0. \tag{61}$$

The first two conditions make the function $\mathbf{g}(\mathbf{r})$ continuous and differentiable (see below) even at $\mathbf{r}=0$. The third condition states that the fibers are either under tension or possibly slack, but never under compression. The last condition states that stretching a fiber cannot decrease its tension.

When the stress function satisfies the conditions (58)–(61), it can be proved that the solution of Eq. (55) exists and is unique. We give the proof partly for the fun of it and partly to set up the apparatus that is used to solve Eq. (55). First, introduce the elastic energy function

$$E(\mathbf{X}_0\cdots\mathbf{X}_{N_p-1})=\sum_k \mathscr{E}\left(\frac{|\mathbf{X}_{k+1}-\mathbf{X}_k|}{\Delta s}\right)\Delta s, \tag{62}$$

where $\mathscr{E}(\theta)=\int_0^\theta \sigma(\theta')\,d\theta'$ so that $\mathscr{E}'=\sigma$. Note that $\mathscr{E}$ (and hence E) is bounded from below by 0. Next evaluate the gradient of E. To do this, recall that $\mathscr{E}'=\sigma$, and use the identity $d|\mathbf{r}|=|\mathbf{r}|^{-1}\mathbf{r}\cdot d\mathbf{r}$. It follows that

$$\begin{aligned} dE&=\sum_k \sigma\left(\frac{|\mathbf{X}_{k+1}-\mathbf{X}_k|}{\Delta s}\right)\frac{\mathbf{X}_{k+1}-\mathbf{X}_k}{|\mathbf{X}_{k+1}-\mathbf{X}_k|}\cdot(d\mathbf{X}_{k+1}-d\mathbf{X}_k)\\ &=\sum_k \mathbf{g}(\mathbf{X}_{k+1}-\mathbf{X}_k)\cdot(d\mathbf{X}_{k+1}-d\mathbf{X}_k). \end{aligned} \tag{63}$$

Shifting the index of the term involving $d\mathbf{X}_{k+1}$, we find

$$\begin{aligned} dE&=-\sum_k \left(g(\mathbf{X}_{k+1}-\mathbf{X}_k)+g(\mathbf{X}_{k-1}-\mathbf{X}_k)\right)\cdot d\mathbf{X}_k\\ &=-\sum_k (\mathbf{f}_k\cdot d\mathbf{X}_k)\,\Delta s. \end{aligned} \tag{64}$$

This shows that $-\mathbf{f}$ is the gradient of E with respect to the inner product

$$(\mathbf{u},\mathbf{v})=\sum_k \mathbf{u}_k\cdot\mathbf{v}_k\,\Delta s. \tag{65}$$

The next step is to consider d^2E, the second variation of E. To do this, we need

$$\begin{aligned} d\mathbf{g}&=d\left(\frac{\mathbf{r}}{|\mathbf{r}|}\right)\sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right)+\frac{\mathbf{r}}{|\mathbf{r}|}\sigma'\left(\frac{|\mathbf{r}|}{\Delta s}\right)\frac{d|\mathbf{r}|}{\Delta s}\\ &=\left(\frac{d\mathbf{r}}{|\mathbf{r}|}-\frac{\mathbf{r}}{|\mathbf{r}|^2}\frac{\mathbf{r}\cdot d\mathbf{r}}{|\mathbf{r}|}\right)\sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right)+\frac{\mathbf{r}}{|\mathbf{r}|}\sigma'\left(\frac{|\mathbf{r}|}{\Delta s}\right)\frac{1}{\Delta s}\frac{\mathbf{r}\cdot d\mathbf{r}}{|\mathbf{r}|}\\ &=G(\mathbf{r})\,d\mathbf{r}, \end{aligned} \tag{66}$$

where $G(\mathbf{r})$ is the 3×3 matrix with elements

$$G_{ij}(\mathbf{r})=\left(\delta_{ij}-\frac{r_ir_j}{|\mathbf{r}|^2}\right)\frac{1}{|\mathbf{r}|}\sigma\left(\frac{|\mathbf{r}|}{\Delta s}\right)+\frac{r_ir_j}{|\mathbf{r}|^2}\frac{1}{\Delta s}\sigma'\left(\frac{|\mathbf{r}|}{\Delta s}\right). \tag{67}$$

To verify the continuity of G at $\mathbf{r}=0$, recall that $\sigma(0)=0$, $\sigma'(0)=0$, and that σ is smooth. Therefore $\sigma=O(|\mathbf{r}|^2)$ and $\sigma'=O(|\mathbf{r}|)$ as $\mathbf{r}\to\mathbf{0}$. It follows that $G_{ij}=O(|\mathbf{r}|)$. Thus we define a continuous function by setting $G_{ij}(\mathbf{0})=0$.

Note that $G_{ij}=G_{ji}$. Moreover, since σ and σ' are both nonnegative by hypothesis, it is easy to show (by the Schwartz inequality) that G is nonnegative definite.

Equipped with the matrix G, we can easily evaluate d^2E. Starting from Eq. (63), we find

$$d^2E=\sum_k \left[G(\mathbf{X}_{k+1}-\mathbf{X}_k)(d\mathbf{X}_{k+1}-d\mathbf{X}_k)\right]\cdot(d\mathbf{X}_{k+1}-d\mathbf{X}_k). \tag{68}$$

It then follows directly from the semi-definiteness of G that $d^2E\geqslant 0$. In other words, E is a convex function.

In summary, we have shown that E is bounded from below by zero, that it is a continuous function with continuous first and seond derivatives, that the gradient of E is $-\mathbf{f}$, and, finally, that E is convex. Now consider the function

$$\phi(\mathbf{X}_0\cdots\mathbf{X}_{N_p-1})=\tfrac{1}{2}\sum_k |\mathbf{X}_k-\mathbf{X}_k^0|^2\,\Delta s+\lambda E(\mathbf{X}_0\cdots\mathbf{X}_{N_p-1}). \tag{69}$$

Clearly ϕ inherits all of the properties of E listed above except that

$$\operatorname{grad}\phi=(\mathbf{X}-\mathbf{X}^0)-\lambda\mathbf{f}(\mathbf{X}). \tag{70}$$

Moreover, ϕ is strictly convex and bounded from below by $\frac{1}{2}\|\mathbf{X}-\mathbf{X}^0\|^2$.

It follows from these properties that ϕ has exactly one stationary point (a global minimum) at some point $\mathbf{X}=\mathbf{X}^*$ and that $\mathbf{X}^*$ satisfies Eq. (55). To show existence of the stationary point consider any number ϕ_0 in the range of ϕ and let $B=\{\mathbf{X}:\phi(\mathbf{X})\leqslant\phi_0\}$. Note that B is a closed, bounded set in R^{3N_p}, since ϕ is continuous and since $\phi\geqslant\frac{1}{2}\|\mathbf{X}-\mathbf{X}^0\|^2$. Therefore, there exists $\mathbf{X}^*\in B$ such that $\phi(\mathbf{X}^*)\leqslant\phi(\mathbf{X})$ for all $\mathbf{X}\in B$. (Of course, it follows that $\phi(\mathbf{X}^*)\leqslant\phi(\mathbf{X})$ for all $\mathbf{X}\in R^{3N_p}$.) Since ϕ has a continuous first derivative, this global minimum $\mathbf{X}^*$ is also a stationary point. That is, $(\operatorname{grad}\phi)(\mathbf{X}^*)=0$. This is identical to Eq. (55) and completes the proof of existence. Uniqueness follows from the fact that a strictly convex function cannot have more than one stationary point. (Suppose there were two: connect them by a straight line segment and consider the restriction of the function to this segment. The first derivative would be zero at both ends of the segment, so the second derivative would have to have zero average over the segment. This is impossible, since the second derivative is strictly positive).

We now consider the numerical solution of Eq. (55). This is done by Newton's method. Let $\mathbf{X}^m$ be the mth guess. The equation for $\mathbf{X}^{m+1}$ is obtained by linearizing Eq. (55) around $\mathbf{X}^m$:

$$\mathbf{X}_k^{m+1}=\mathbf{X}_k^0+\lambda\mathbf{f}_k^m+\lambda\sum_{k'}\left(\frac{\partial\mathbf{f}_k}{\partial\mathbf{X}_{k'}}\right)^m(\mathbf{X}_{k'}^{m+1}-\mathbf{X}_{k'}^m). \tag{71}$$

Subtract $\mathbf{X}_k^m$ from both sides, and let

$$\mathbf{Z}_k^m = \mathbf{X}_k^{m+1} - \mathbf{X}_k^m \tag{72}$$

$$-\mathbf{R}_k^m = \mathbf{X}_k^m - \mathbf{X}_k^0 - \lambda \mathbf{f}_k^m. \tag{73}$$

Then Eq. (71) may be rewritten

$$\mathbf{Z}_k^m - \lambda \sum_{k'} \left(\frac{\partial \mathbf{f}_k}{\partial \mathbf{X}_{k'}} \right)^m \mathbf{Z}_{k'}^m = \mathbf{R}_k^m. \tag{74}$$

To evaluate the 3 × 3 matrices $\partial \mathbf{f}_k / \partial \mathbf{X}_{k'}$, start from Eq. (56). The result is

$$\begin{aligned} \frac{\partial \mathbf{f}_k}{\partial \mathbf{X}_{k'}} &= G(\mathbf{X}_{k+1} - \mathbf{X}_k)(\delta_{k+1,k'} - \delta_{kk'}) + G(\mathbf{X}_{k-1} - \mathbf{X}_k)(\delta_{k-1,k'} - \delta_{kk'}) \\ &= \delta_{k+1,k'} G(\mathbf{X}_{k+1} - \mathbf{X}_k) + \delta_{k-1,k'} G(\mathbf{X}_{k-1} - \mathbf{X}_k) \\ &\quad - \delta_{kk'} [G(\mathbf{X}_{k+1} - \mathbf{X}_k) + G(\mathbf{X}_{k-1} - \mathbf{X}_k)]. \end{aligned} \tag{75}$$

Thus, the equations for Z_k may be written as a periodic block-tridiagonal system,

$$-A_k^m \mathbf{Z}_{k-1}^m + B_k^m \mathbf{Z}_k^m - C_k^m \mathbf{Z}_{k+1}^m = \mathbf{R}_k^m, \tag{76}$$

where A, B, C, are 3 × 3 matrices defined by

$$A_k = \lambda G(\mathbf{X}_{k-1} - \mathbf{X}_k) \tag{77}$$

$$C_k = \lambda G(\mathbf{X}_{k+1} - \mathbf{X}_k) \tag{78}$$

$$B_k = I + A_k + C_k. \tag{79}$$

Note that A, B, C are symmetric (since G is symmetric) and also that $A_{k+1} = C_k$ (since G is an even function.) Therefore Eq. (76) is a symmetric system of equations. The positive definiteness of this system is equivalent to the strict convexity of ϕ, which was proved above.

We now consider the solution of Eq. (76) which has the form of a periodic block-tridiagonal system (with 3 × 3 blocks). In the following discussion we drop the fixed iteration number, m. The periodicity of Eq. (76) is removed as follows. Let M_k be a 3 × 4 matrix (for each k) defined as the solution of

$$-A_k M_{k-1} + B_k M_k - C_k M_{k+1} = D_k, \tag{80}$$

where $k = 1, 2, ..., N_p - 1$ and where

$$D_k = (\mathbf{0} \mid \mathbf{0} \mid \mathbf{0} \mid \mathbf{R}_k)$$

$$M_0 = M_{N_p} = (I \mid \mathbf{0}).$$

That is, D_k is the 3 × 4 matrix whose first three columns are $\mathbf{0}$ and whose last column is $\mathbf{R}_k$, and M_0 and M_{N_p} are 3×4 matrices whose first three columns form the 3×3 identity and whose last column is zero. Note that Eq. (80) amounts to 4 separate non-periodic block tridiagonal systems, one for each column of M. All four systems have the same coefficients. These four systems are solved by block L-U factorization. This is identical to ordinary L-U factorization except that the ordinary arithmetic is replaced by matrix operations involving 3×3 matrices. The routines that handle these matrix operations are written out with no DO loops in order to avoid the overhead of initiating a loop of length 3.

Once the four columns of M_k have been determined (for all k) we seek the solution $\mathbf{Z}_k$ of Eq. (73) in the form of a linear combination of the columns of M_k:

$$\mathbf{Z}_k = M_k \begin{pmatrix} \boldsymbol{\alpha} \\ 1 \end{pmatrix}, \tag{81}$$

where $\boldsymbol{\alpha}$ is a 3-vector (independent of k) which remains to be determined. Before finding $\boldsymbol{\alpha}$, we remark that any $\mathbf{Z}_k$ of the form given by Eq. (81) automatically solves Eq. (76) for $k = 1, 2, ..., N_p - 1$. To show this, post-multiply Eq. (80) by $\left(\begin{smallmatrix} \boldsymbol{\alpha} \\ 1 \end{smallmatrix}\right)$. The result is precisely the restriction of Eq. (76) to $k = 1, 2, ..., N_p - 1$. To determine $\boldsymbol{\alpha}$, use Eq. (76) with $k = 0$:

$$-A_0 \mathbf{Z}_{N_p-1} + B_0 \mathbf{Z}_0 - C_0 \mathbf{Z}_1 = \mathbf{R}_0 \tag{82}$$

$$(-A_0 M_{N_p-1} + B_0 M_0 - C_0 M_1) \begin{pmatrix} \boldsymbol{\alpha} \\ 1 \end{pmatrix} = \mathbf{R}_0. \tag{83}$$

We already have $M_0 = (I \mid \mathbf{0})$. Let M_k be similarly partitioned:

$$M_k = (Q_k \mid \mathbf{Y}_k). \tag{84}$$

Then $\boldsymbol{\alpha}$ satisfies the 3×3 system

$$(B_0 - A_0 Q_{N_p-1} - C_0 Q_1)\boldsymbol{\alpha} = \mathbf{R}_0 + A_0 \mathbf{Y}_{N_p-1} + C_0 \mathbf{Y}_1. \tag{85}$$

This 3×3 system is easily solved for $\boldsymbol{\alpha}$, and then the $\mathbf{Z}_k$ are determined according to Eq. (81). (Note that $\mathbf{Z}_0 \equiv \boldsymbol{\alpha}$, since $M_0 = (I \mid \mathbf{0})$.)

In summary, the nonlinear equations for the fiber forces are solved by Newton's method. The linear system that must be solved at each iteration of Newton's method is a periodic, block-tridiagonal system on each fiber. The periodic problem is reduced to four corresponding non-periodic problems with the same coefficients. These are solved by block L-U factorization, and the periodic solution is constructed as a linear combination of the solutions of the four non-periodic problems.

Next, we consider the linear systems for $\mathbf{u}^{n+1,1}$, $\mathbf{u}^{n+1,2}$, $\mathbf{u}^{n+1,3}$, and $(\mathbf{u}^{n+1}, p^{n+1})$. The methods that we use to solve these linear systems have been described in [12]; we do not repreat the details here. The following brief summary may be useful to the reader, however.

The equations that determine $\mathbf{u}^{n+1,s}$ (Eq. (50)) have the form of periodic tridiagonal systems with coupling only in the direction indicated by $\mathbf{e}_s$. Thus, for each

s, there are $3N^2$, independent systems, three for each line of the lattice parallel to $\mathbf{e}_s$. The three systems on the same line have the same coefficients. The periodicity in these systems is removed in essentially the same manner as described above for the fibers, and the resulting non-periodic systems are solved by L-U factorization.

The equations for $(\mathbf{u}^{n+1}, p^{n+1})$ are reduced to a discrete Poisson equation for p^{n+1}, Eq. (53). This equation is solved by the Fourier–Toeplitz method [13–14]. This method uses the 2-dimensional discrete Fourier transform on all planes $x_3 = \text{constant}$. When this transform is applied to both sides of Eq. (53), the result is a collection of periodic tridiagonal systems, one on each column (parallel to $\mathbf{e}_3$) of the computational lattice. Again, the periodicity is removed by essentially the same method that was described above in the case of the fibers, and the tridiagonal systems for the Fourier coefficients are solved by L-U factorization. Finally, the inverse Fourier transform is applied on all planes $x_3 = \text{constant}$ to recover p^{n+1}, and $\mathbf{u}^{n+1}$ is evaluated by explicit application of Eq. (51).

IMPLEMENTATION: VECTORIZATION

The algorithm described above has extraordinary potential for parallelization. To make this explicit, we list those features of the algorithm that can exploit the capability of a machine to perform parallel computation. (The special case of a vector machine will be discussed in more detail below.)

1. The computation of the fiber forces can be done independently on each fiber.

2. For each $s = 1, 2, 3$, there are N^2 lines of the computational lattice parallel to $\mathbf{e}_s$. We solve 3 periodic tridiagonal systems per line. The systems on one line are independent of those on another.

3. During the pressure computation, we compute the 2-dimensional Fourier transform of the divergence data independently on each of the N planes $x_3 = \text{constant}$. (The same statement holds for the inverse Fourier transform.)

4. Each of the 2-dimensional Fourier transforms (and inverse transforms) is computed by applying the 1-dimensional Fourier transform, first on all lines parallel to $\mathbf{e}_1$ and then on all lines parallel to $\mathbf{e}_2$.

5. The equations that are solved for the Fourier coefficients of the pressure are independent tridiagonal systems on each of the N^2 lines parallel to $\mathbf{e}_3$.

6. The explicit formulae for divergence and gradient can be applied independently at each point of the computational lattice.

7. The coefficients of the computational δ-function (64 numbers per fiber point) can be computed independently for each fiber point.

8. In the operation of spreading the fiber forces out onto the computational lattice, each of the 64 δ-function coefficients at a given fiber point can be independ-

ently multiplied by the corresponding element of force. Note, however, that the different fiber points interact to the extent that two different fiber points may apply force to the same point of the fluid lattice. Whether or not these additive operations can be performed in parallel is a hardware-dependent feature of the machine. (The required parallel operation is known as "fetch-and-add" [15].)

9. In the operation of interpolating the fluid velocity from the fluid lattice to the points of the fibers, the operation required at each fiber point is an inner product between the 64 δ-function coefficients and the corresponding elements of fluid velocity from the $4 \times 4 \times 4$ box of the fluid lattice surrounding the fiber point. Thus, any method of exploiting parallelism in the computation of inner products can be used here. Whether or not the *different* inner products corresponding to different fiber points could be computed in parallel with each other would depend on the capability of different parallel processes to have simultaneous (read-only) access to overlapping segments of the same (fluid velocity) array.

The foregoing survey of parallel aspects of the algorithm has been rather general. Let us now consider the particular form of parallel computation known as vectorization. Vector computers obtain speedup by performing the same operation repetitively on different data sets. If the vector length is n, the n different operations must be independent in the sense that none of the n results are needed as input to any of the n computations.

Moreover, vector computers usually impose the requirement that the data be regularly arranged in memory. The increment in memory location from one item of data to the corresponding item for the next of the n operations is called the *stride.* On Cray computers (which we have used for this work) the stride can be any integer, but best results are obtained if memory-bank conflicts are avoided. On the Cray 2, this means that the stride should be odd.

The issue of stride arises naturally in the context of multidimensional arrays. Given a FORTRAN array with dimensions ND1 × ND2, the stride through memory as the first index is incremented in steps of 1 is 1, while the stride through memory as the second index is incremented in steps of 1 is ND1. This is because array element (I, J) is stored at memory location.

$$(I-1) + ND1 * (J-1) + \text{constant}$$

For applications such as ours, it is therefore critical that vectorization be possible with stride other than 1. (The requirement on the Cray 2 that the stride be odd is easily met by adding extra lines to the arrays as needed.) The requirement of constant stride is naturally met in all of the fluid dynamics routines and the fiber routines as well, but it causes some difficulty in the routines that couple the fibers to the fluid, see below.

Our actual use of vectorization is tempered by another consideration that will be described more fully in the following section. We have structured our code in such a way that only a small part of the data must be present in central memory at any given time. This has the obvious advantage that the problem size is not limited by the central memory of the machine. In the case of the fibers, this simply means that we partition the fibers into groups such that each group is small enough to fit into central memory. In the case of the fluid computation, we partition the fluid lattice into planes $x_3 = \text{constant}$, and we hold only a few adjacent planes in central memory at once. In the interaction routines, we sweep through all lattice planes for each group of fibers.

The strategy outlined above complicates vectorization but does not prevent it. In certain cases it reduces the vector length from N^2 to N. On Cray computers, this causes only a modest drop in efficiency if N is already 64 or greater, since 64 is the length of the vector registers. (Contrary to what one might think, there is some speedup beyond $N = 64$.)

We now come to the specifics of how the code has been vectorized. In almost every case we have consistently applied the following simple strategy: where there are many instances of a computation to be performed, let the innermost loop(s) run over the different instances and let the outer loop(s) control the algorithm itself. For example, consider the computation

```
  DO 1 I=2, N
  B(I)=B(I)-A(I)*C(I-1)
1 C(I)=C(I)/B(I)
```

which is an ingredient of a tridiagonal equation solver. The foregoing loop does not vectorize because of the "dependency" of $C(I)$ on $C(I-1)$. If we have N of these computations to perform, however, we may write

```
  DO 1 I=2, N
  DO2J=1, N
  B(I, J)=B(I, J)-A(I, J)*C(I-1, J)
2 C(I, J)=C(I, J)/B(I, J)
1 CONTINUE
```

In this case the inner loop clearly vectorizes, since there is no interaction between the different values of J. Incidentally, this is a case in which the stride through memory associated with the inner loop is not equal to 1; it is equal to the first dimension of the arrays A, B, and C. As remarked above, this first dimension should therefore be chosen to avoid memory-bank conflicts: on the Cray 2 it should be odd.

The strategy that we have just described is applied in our code in the following circumstances:

(1) On each plane $x_3 = \text{constant}$, we solve tridiagonal systems parallel to $\mathbf{e}_1$ and $\mathbf{e}_2$. In the first case the inner loops march in the $\mathbf{e}_2$ direction while in the second they march in the $\mathbf{e}_1$ direction.

(2) The algorithm also requires the solution of tridiagonal systems on lines parallel to $\mathbf{e}_3$. (This occurs twice: in the computation of $\mathbf{u}^{n+1,3}$ and the solution for the Fourier components of the pressure.) The inner loops run in the $\mathbf{e}_1$ and $\mathbf{e}_2$ direc-

tions. That is, the inner loops sweep over the planes $x_3 = \text{constant}$. Note the happy coincidence that this vectorization strategy is consistent with the partition of the fluid lattice into planes $x_3 = \text{constant}$ and with the idea of holding only a few such planes in central memory at any given time.

(3) The algorithm requires computation of the 2-dimensional discrete Fourier transform on each of the planes $x_3 = \text{constant}$. This is done by applying the fast Fourier transform in the $\mathbf{e}_1$ direction and then in the $\mathbf{e}_2$ direction. Instead of vectorizing the FFT algorithm itself, we simply take a standard (scalar) FFT code and rewrite its key operations as loops that run perpendicular to the direction in which the transform is being computed.

(4) The computation of the fiber forces is vectorized as follows. First, write the code for a single fiber. Then, replace every operation by a loop that runs over a large group (possibly all) of the fibers. Note that all fibers in a group must have the same number of points and that the Newton iteration must proceed on all fibers until the stopping criterion on every fiber has been satisfied. These restrictions illustrate the limitations of a vector (as opposed to a truly parallel) machine. It is important to follow this strategy consistently down to the lowest level of the algorithm. For example, consider the 3×3 matrix operations involved in the solution of the block-tridiagonal systems that are solved during each Newton step. Each arithmetic step within these 3×3 matrix operations is applied to all fibers (in a group) before the next step is initiated.

We now consider the vectorization of the routines that couple the fibers and the fluid. (For related work, see [16, 17].) Recall that each fiber point interacts with a $4 \times 4 \times 4$ box of the fluid lattice. Unfortunately, the 64 points of this box cannot be located by any constant stride through memory. Therefore the GATHER operation must be used to collect relevant data from the fluid lattice and the SCATTER operation must be used to distribute the results. On the Cray 2, these operations vectorize. Fortunately, the number of data items to be GATHERed or SCATTERed is $4^3 = 64$, which exactly fills the vector registers.

A difficulty which arises here, however, is that GATHER and SCATTER each require an index array for indirect addressing. The numbers in this array are most easily computed in 3 nested 4×4 loops, which are too short for efficient vectorization. The same difficulty arises in the computation of the coefficients given by the function δ_h, see Eqs. (39–(40). This difficulty is handled by considering many (up to 64) fiber points at once instead of just one. The structure of the resulting code is as follows:

```
   DO 10 K=0, 3
   DO 10 J=0, 3
   DO 10 I=0, 3
   M=K*16+J*4+I
   DO 10 NPT=1, NPOINTS
   DELTA (M, NPT)=
   INDEX (M, NPT)=
10 CONTINUE
```

Note that this strategy can use a lot of storage if *NPOINTS* is allowed to get large, since the arrays DELTA and INDEX are of size 64 × *NPOINTS*. That is why we do not consider all points at once but put them in groups of 64 (enough to fill the vector registers).

IMPLEMENTATION: SMALL CENTRAL MEMORY

The section describes what to do if the problem is too large to fit into the central memory of the machine. (For the size problems that we currently consider, this will occur on the Cray X-MP but not on the Cray 2.) The techniques outlined here are described in detail in [12]. This reference, however, is limited in scope to the fluid-dynamics subroutine. It is straightforward to apply the same principles to the fibers, but the fiber–fluid interaction causes considerable difficulty which will be considered in this section.

When central memory is too small to hold all of the data required for the computation, the obvious solution is to use disc memory as the principal storage medium and to hold in central memory at any given time only some immediately relevant subset of the data. The limitations on this approach are that disc memory is sequential access (as opposed to random access) and that transfer of information between disc and central memory is slow (especially the initiation of such transfer).

These difficulties are partially overcome by replacing the disc by an SSD (solid-state storage device) which is available with the Cray X-MP. The SSD looks to the user exactly like a disc, but its transfer rate is so high that the fastest way to zero a *large* array on the Cray X-MP is to read in an array of zeros that has been previously stored in the SSD (P. de Forcrand, unpublished remark). The emphasis on *large* in the foregoing should warn the reader that there is still considerable overhead associated with initiation of the transfer operation. This overhead becomes less and less important as the size of the transferred data set increases.

Further improvement in performance can be obtained through the use of asynchronous input–output (i/o), which can be used either with conventional disc or with SSD. In asynchronous mode, an i/o operation can proceed concurrently with computation or with another i/o operation (on a different file). Parallel execution of different i/o operations is limited, however, by the number of available channels. Input–output operations are only initiated by the central processing unit which then proceeds to the next program step (possibly initiation of another i/o operation) while i/o proceeds. WAIT statements are available to force the central processor to wait for completion of i/o on any particular file when the logic of the program requires such synchronization.

We now describe the asynchronous SSD implementation of the fluid dynamics subroutine (for details, see [12]). The main point is the data structure in the SSD and the corresponding data structure in central memory.

All of the 3-dimensional $N \times N \times N$ arrays that are needed for the computation are sorted in the SSD as files, which are conceptually subdivided into records corresponding to the planes $x_3 = \text{constant}$. For those arrays that are repeatedly updated, we maintain two copies: a READ copy that holds old data and a WRITE copy to store new results as they are obtained. Once the file has been completely updated, the READ and WRITE labels (not the data!) are interchanged, and the old data are effectively forgotten. This makes it possible (with asynchronous i/o) to perform READ and WRITE operations simultaneously on the "same" file.

Corresponding to each READ-WRITE pair of N^3 arrays in the SSD, we maintain a much smaller $N \times N \times 4$ array in central memory. This array is used as a circular buffer. At any given time it holds four contiguous planes of data (numbered $0 \ldots 3$). The mapping from the SSD plane index K to the central memory plane index KX is $KX = \text{MOD4}(K)$. In a typical operation, computation proceeds generating new results on plane $KX = \text{MOD4}(K)$ and possibly using (but not altering) the data on planes $KXM1 = \text{MOD4}(K-1)$ and $KXM2 = \text{MOD4}(K-2)$. At the same time as these computations are proceeding, data are read into central memory plane $KXP1 = \text{MOD4}(K+1)$ to prepare for the next pass through the loop (in which K will be greater by 1), and data are simultaneously written out to the SSD from central memory plane $KXM1$. (Note that the data which are written out were updated during the previous pass through the loop during which K was smaller by 1.) For further details concerning the fluid-dynamics subroutine, see [12].

Asynchronous i/o can be applied to the fiber computations in the same manner. The fiber data is stored on the SSD in a file which is conceptually subdivided into records. Each such record contains not one fiber but a large group of fibers. It is convenient if all fibers in a group contain the same numbers of points so that, when these fibers are together in central memory, we can obtain vectorization over the different fibers (see previous section). Alternatively the group may consist of several subgroups, where all fibers in a subgroup have the same number of points. A central memory array large enough to hold three groups of fibers (numbered $0 \ldots 2$) is provided. The mapping from SSD group index K to central memory group index KX is $KX = \text{MOD3}(K)$. While computation is proceeding on fibers in central memory group KX, fiber data are read into central memory group $KXP1 = \text{MOD3}(KX+1)$ and written from central memory group $KXM1 = \text{MOD3}(KX-1)$.

The principal difficulty concerning the use of external memory arises when we consider the transfer of information between the fibers and the fluid. Such interaction arises twice (per timestep) in our algorithm: when the fiber forces act on the fluid velocity field and when the velocity field is interpolated in order to move the fibers. Since the orientation of the fibers is arbitrary, each fiber may interact with many (or even all) planes of the computational lattice even though each individual fiber point only interacts with four planes: the two above it and the two below it.

Therefore the natural strategy is to read in the fibers in groups, as above, and, for each group of fibers, to sweep through all planes of the fluid lattice. This sweep through all planes is conducted in the same style as was described above for the fluid dynamics subroutine, except that more central memory planes are needed, as will be explained below.

Because only a small number of lattice planes will be present in central memory at any given time, we must sort the fiber points in the group that lies in central memory according to their z-coordinate. This is done by constructing a linked list. (This operation could be vectorized by partitioning each group of fibers into 64 subgroups and constructing 64 independent linked lists, but we have not bothered to do so.) The fiber point is assigned the plane index K if it lies above (or on) plane K but below plane $K+1$.

Now a fiber point with plane index K interacts with lattice planes $K-1$, K, $K+1$, and $K+2$. Thus, at least these four planes must be in central memory when the fiber points with plane index K are considered. One additional central-memory plane is needed to hold data from plane $K+3$ that are read in (to prepare for the next value of K) while the computation proceeds. In the interpolation step these five planes would suffice, since the velocity data are unaltered. In the application of the fiber forces, however, we need a sixth central-memory plane to hold updated velocity data while they are written out to SSD plane $K-2$. These data were updated during the previous pass though the loop when K was lower by 1.

In practice, we use a circular buffer consisting of eight planes for the following reason. The computational lattice is periodic with period N, where N is a power of 2. The mapping from SSD plane index K to central memory plane index KX is of the form $KX = \mathrm{MOD}b(K)$, where b is the number of planes in the circular buffer. To enforce periodicity of the computational lattice, this mapping should have the property $\mathrm{MOD}b(K+N) = \mathrm{MOD}b(K)$. In other words, $\mathrm{MOD}b(N) = 0$, or N/b is an integer. This happens if and only if b is also a power of 2 and $b \leqslant N$. The choice $b = 8$ corresponds to the smallest power of 2 that contains the necessary 5 or 6 planes.

Unfortunately the foregoing approach involves an amount of i/o of the fluid variables proportional to the *product* of the number of fiber groups and the number of lattice planes. The only way to keep this product down is to make the fiber groups as large as possible so as to reduce their number as much as possible. The best possible situation is when all fibers can be permanently held in central memory. Then there is no fiber i/o and only two sweeps through the fluid lattice are needed in the interaction routines, one to apply force to the fluid and the other for interpolation. Obviously, the extent to which this can be achieved is both machine and problem dependent.

The following approach, which we have not programmed, avoids the difficulties discussed above. After the fiber forces have been computed, read in the fibers in groups and sort the fiber points (using vectorized linked lists, as discussed above) according to their z-coordinate. Along with the data for each fiber point, record its group number, fiber number, and point number for future reference. Then, instead of immediately considering the fluid variables, write the fiber data onto the SSD in files organized according to the plane index (with a separate file for each plane index!) As the successive fiber groups are considered these files grow in length, until finally each file contains all fiber data for those fiber points in its plane. Once the fiber points are sorted in this way, we can sweep through the fiber and fluid data once and apply all of the fiber forces to the fluid.

Later, when it is time to interpolate the fluid velocity to the fiber points, we already have the fiber points sorted by plane index, so one sweep through the fluid lattice suffices to move all of the fibers points.

Now we must reorganize the fiber points by fiber number and point number in preparation for the next time step. This is done as follows. First sort the data by fiber groups using a separate file for each group. As we sweep through the different planes of fiber data, these groups gradually grow until finally each file holds all fiber points belonging to fibers in its group. These points are still in random order, however. The last step is to read in the groups one by one and sort the data according to fiber number and point number and then write the results back onto the SSD.

This approach involves only two sweeps through the fluid data (one to apply the fiber forces and a second to interpolate the fluid velocity) and an amount of fiber i/o that is linear in the number of fiber points.

What should one do if one has a code written according to the principles outlined in this section and one is offered the use of a computer such as the Cray 2 which lacks an SSD but which has a large enough central memory to contain the entire problem? The answer is very simple: use the code intact but declare that the SSD files should be replaced by internal files. (Alternatively, if internal files are not yet supported, set up 3-dimensional arrays corresponding to the SSD files and replace the i/o routines by routines that simply copy data into or out of these 3-dimensional arrays.) While such internal data transfer may seem rather silly, its cost is quite modest, and this approach has the great advantage of maintaining portability of the code back to a small central memory machine such as the Cray X-MP.

This is what we have actually done. Accordingly, the results reported in the following section were obtained on the Cray 2 with all data residing in central memory. For performance studies on the Cray X-MP/SSD (limited, however, to performance of the fluid-dynamics subroutine), see [12].

Results

In this section we use the foregoing methods to study the vibrations of an immersed, fiber-wound toroidal tube. The tube has two layers of fibers located half a mesh-width apart, so in the limit $h \to 0$ the two layers coincide. Each layer consists of a collection of "helical" fibers which close on themselves after making $m = 3$ short turns and $n = 1$ long turns around the torus. The pitch of the helices is opposite in the two layers, so that the two systems of fibers run across each other and thereby help maintain the integrity of the tube. There is fluid everywhere: inside the tube, outside the tube, and between the layers. The entire system is enclosed in a periodic box. The tube is prestressed in the sense that the volume of fluid within it is sufficiently large that the fibers are always under tension.

Such a tube has an equilibrium configuration in which the fiber tension balances a pressure difference across the walls of the tube. It can be shown that the equilibrium shape of the tube and also the shape of the equilibrium fiber trajectories is completely determined by the ratio of the topological parameters m and n introduced above. When m/n is large, the equilibrium torus is long and thin; when m/n is small, it is short and fat.

Our purpose here is to study the (damped) vibrations of the torus about the equilibrium described above. Accordingly, we set up the tube in a near-equilibrium configuration, and then we introduce a sinusoidal perturbation in tube diameter with period equal to $\frac{1}{3}$ the circumference of the torus. The fluid is initially at rest. These initial conditions establish a damped standing wave which decays away leaving the tube at equilibrium but still under tension with a pressure difference across its walls.

The physical parameters of the problem are as follows. Let L^* be an arbitrary unit of length, T^* an arbitrary unit of time, and M^* an arbitrary unit of mass. Any one computation yields results that are valid for a whole family of physical problems generated by assigning different values to these arbitrary units. In the results that follow, the domain is a cube with edge of length $L = L^*$, the fluid density and viscosity are $\rho = M^*/(L^*)^3$, $\mu = (0.125\ (L^*)^2/T^*)\rho$, and the stiffness of the fibers is such that the period of the damped vibration is about $T = 0.0625\,T^*$. [illegible] the choice of units.

To generate a specific physical example, we choose $L^* = 2\,\text{cm}$, $T^* = 16\,\text{s}$, $M^* = 8\,\text{g}$. Then the physical parameters of the problem are $L = 2\,\text{cm}$, $\rho = 1\,\text{g/cm}^3$, $\mu = (0.03125\,\text{cm}^2/\text{s})\rho$, $T = 1\,\text{s}$. This choice of units and the resulting physical parameters are used for interpretation in the legends of the figures that follow.

Figure 1 shows two perpendicular views of the outer and inner layers of the tube in their initial configurations. The inner layer is only slightly smaller than the outer layer, but the pitch of its fibers is in the opposite direction.

Figure 2 shows the computed velocity fields in the three coordinate planes through the center of the torus. The computation was performed on a 64^3 lattice, but only a 16×16 array of velocity vectors is plotted in each of the coordinate planes. Both layers of the tube are shown (where they intersect the coordinate planes) but the layers are so close together that they mostly appear to coincide. In frame 1, top, the velocity vectors are everywhere outward. This represents an initial adjustment of the torus towards equilibrium that would occur even if the three bumps were not present; it happens because we have used only an approximate formula for the equilibrium shape to initialize the torus. Note that the outward velocities are much larger near the center of the figure than towards the outside. This suggests that the cross section of the tube is becoming smaller and that volume may be conserved; see below for quantitative evidence of volume conservation.

In frame 2 of Fig. 2, top row, we see clearly established the flow pattern of the standing wave with outward flow at 0° and at $\pm 120°$ and with inward flow at $\pm 60°$ and 180° measured from the positive x axis. This flow pattern, which dominates frames 2–3, has reversed itself by frames 6–7 and re-established itself

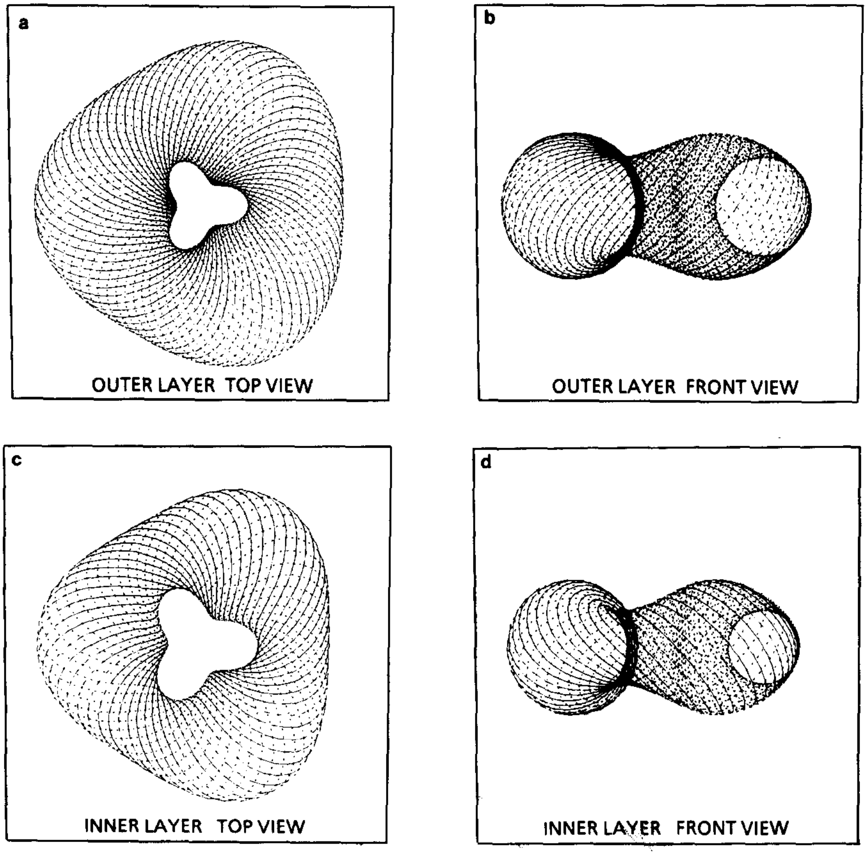

FIG. 1. Two perpendicular views of the outer and inner layers of the fiber-wound toroidal tube in its initial configuration. Solid lines show fibers which can be seen in each view, and broken lines show hidden fibers. Note that the two layers are almost identical in size, but that the pitch of the fibers is opposite so that the two layers will run across each other when the tube is assembled. Outer square shows edges of the periodic box of fluid in which the fiber-wound tube is immersed. This box is a cube of edge 2 cm. The fluid has density $\rho = 1$ gm/cm^3 and viscosity $\mu = (0.03125 \text{ cm}^2/\text{s})\rho$.

again (although with much reduced amplitude) in frames 10–11. Similarly, the three bumps which are clearly at ±60° and 180° in frame 1 have moved to 0° and ±120° in frames 4–5. As the computation proceeds the standing wave is damped out by fluid viscosity until, by frame 12, the fluid is nearly at rest.

Figure 3 shows the computed pressure contours from the same computation as in Fig. 2. In this case, we do not draw the walls of the tube at all, since the pressure contours do that for us. As the tube relaxes, the interior pressure contours gradually disappear, but the pressure contours at the wall persist. This confirms the statement that the equilibrium configuration of the tube is one in which the fibers are under tension and the internal fluid is pressurized.

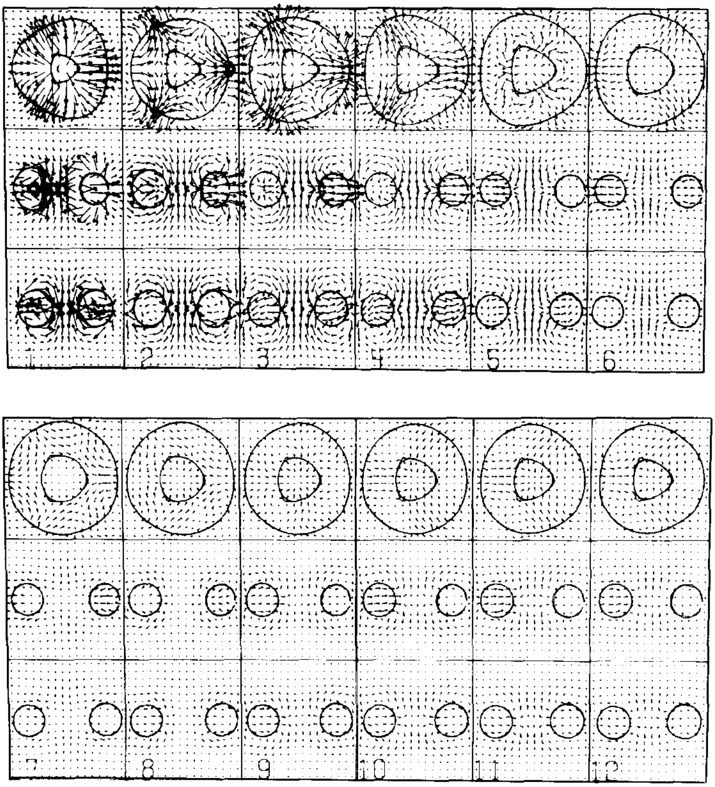

FIG. 2. Velocity vectors in the three coordinate planes through the center of the tube. The interval between frames is 32 time steps, which corresponds to 0.125 s, since each time step is 2^{-8} s. Frame 1 is at time step 16, or $t = 0.0625$ s. Although the computation was done on a 64^3 grid, only 16 points in each direction are shown.

Note the threefold symmetry of the computed results in the top row of Figs. 2–3. There is little indication that this symmetry has been disturbed by the fourfold symmetry of the periodic domain. This suggests that the periodic boundary conditions have only a slight effect on the results.

As stated above, the computation of Figs. 2–3 was performed on a 64^3 lattice. At this resolution, 7 million words of central memory were required (for in-core computation on the Cray 2), and each time step took 16 cpu s. The duration of the run shown in Figs. 2–3 was 368 time steps. These computational requirements are not drastically different from those that we anticipate for the heart computations that are the ultimate goal of this work. In our 2-dimensional work on the left heart [4–9], the resolution 64^2 was adequate, and roughly 1000 time steps per heartbeat were used. The principal increase in cost will be associated with a substantial increase in the number of fibers needed to model the thick ventricular walls. Of course, the costs will increase dramatically if we have to switch to 128^3 resolution.

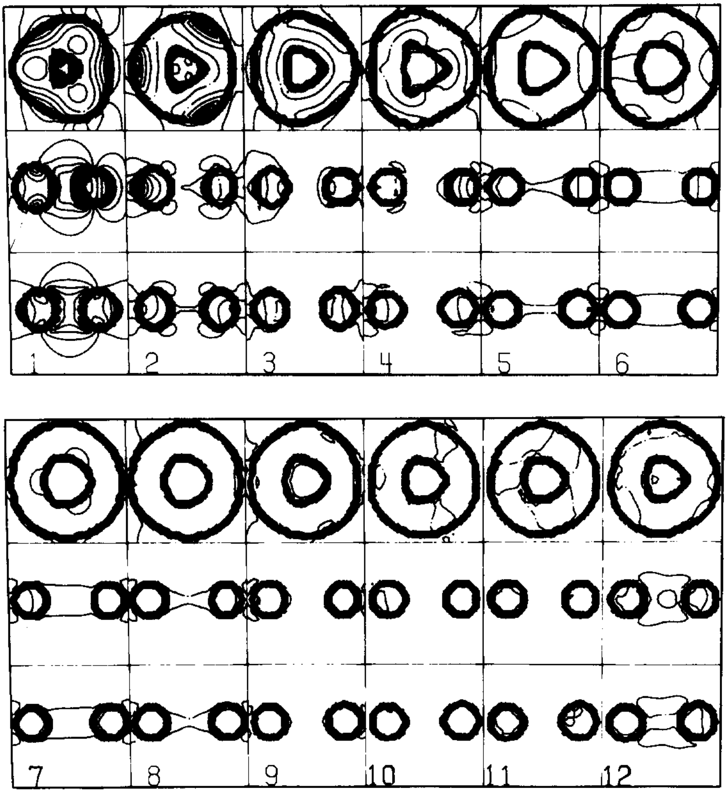

FIG. 3. Pressure contours corresponding to the velocity vectors of Fig. 2. Note the high density of pressure contours (i.e., the steep pressure gradient) near the walls of the tube, which are only indicated by the pressure contours themselves in this figure. The pressure difference across the walls persists as the vibration decays away.

This may prove necessary because we plan to model the entire heart and the nearby great vessels.

One check on the computation of Figs. 2–3 is to see whether the volume of the tube is conserved. This can be done for the inner layer, the outer layer, and also for the small volume of fluid between the two layers. The (relative) losses in volume that are recorded from the beginning to the end of the run are as follows: 0.5% for the volume enclosed within the outer layer, 0.5% for the volume enclosed within the inner layer, and 1% for the volume between the two layers.

A more important check, however, is to perform the same computation on different lattices and to compare the results. The computation shown above was for $NG = 64$. (NG is the number of lattice points in each direction.) To check convergence we use $NG = 32$, 64, and 128. As NG varies, both the number of points on each fiber and also the number of fibers in each layer change in proportion to NG. If we were modeling a thick wall, we would also increase the number of layers in

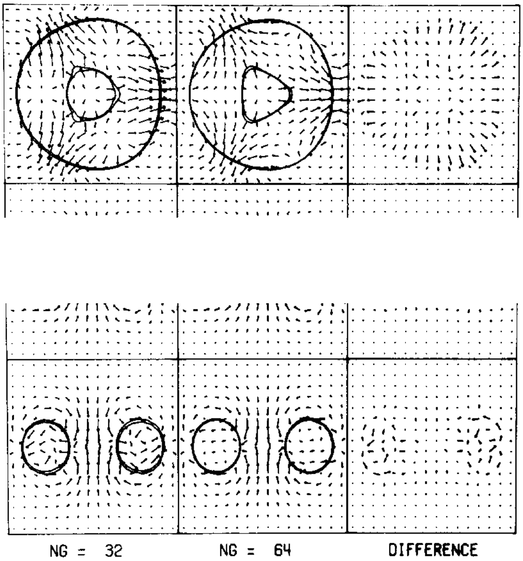

FIG. 4. Comparison of results obtained on a 32^3 lattice with results obtained on a 64^3 lattice. Comparison is made at time $t = 0.172$ s, which is near frame 2 of Fig. 2–3. The difference between the two computations is considerable, especially near the walls of the tube. (Compare Fig. 5.)

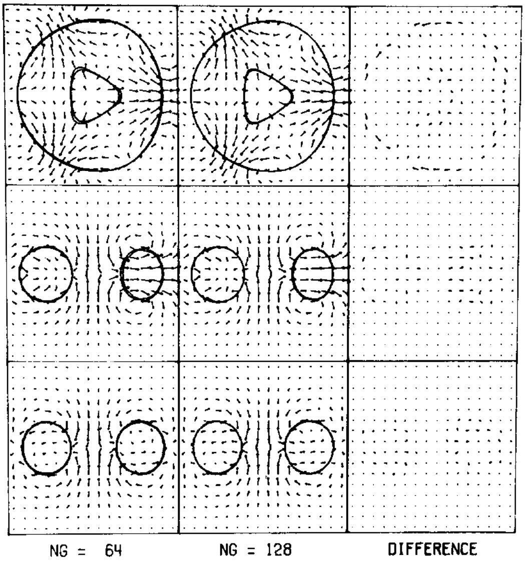

FIG. 5. Comparison of results obtained on a 64^3 lattice with results obtained on a 128^3 lattice. The comparison is made at the same physical time as in Fig. 4, but the differences here are much smaller than in that figure. This is empirical evidence of convergence. For a quantitative version of this evidence, see Tables I and II.

proportion to NG. Here, however, the wall is thin, the number of layers is fixed at two, and the spacing between them is proportional to the lattice spacing. In the convergence study reported here, the time step varies as the square of the lattice spacing. (This may be overly cautious, and other strategies such as time step proportional to lattice spacing should certainly be explored.) The stability parameter β (see Eq. (46)) is adjusted here in proportion to the time step (and hence to the square of the lattice spacing) with $\beta = 2^{-12}$ at the resolution $NG = 64$. Such small values of β mean that little smoothing is done by the implicit computation of the fiber forces. Past experience [4–9] suggests that larger β would be needed if the fiber stiffness were increased.

Results of the convergence study are reported at a particular time corresponding to frame 2 of Figs. 2–3. Note that this time corresponds roughly to $\frac{1}{4}$ period of the standing wave. The convergence results are shown visually in Figs. 4–5. Figure 4 plots the velocity fields (in the coordinate planes, as before) for $NG = 32$ in the first column, then the corresponding velocity fields for $NG = 64$, and finally the difference of these two fields. Figure 5 does the same for $NG = 64$ and 128. Although the difference between the computed results for $NG = 32$ and 64 is substantial (Fig. 4), the difference between the results computed with $NG = 64$ and 128 is encouragingly small. The decreasing difference is empirical evidence of convergence, and the small difference obtained between $NG = 64$ and 128 suggests that $NG = 64$ is a reasonable lattice size for this problem.

The numerical convergence results are summarized in Table I for the entire domain and in Table II for the subdomain obtained by cutting out a shell containing the fibers. The results in Table I are consistent with convergence that is roughly of first order, at least in the L_1 and L_2 norms, and somewhat worse in the *Max* norm. The fact that the results get worse as we go from $L_1 \rightarrow L_2 \rightarrow Max$ norm suggests that the changes from one grid to the next are localized somewhere. Indeed, from Figs. 4–5, it is clear that they are localized near the wall of the tube. This is confirmed in Table II in which we study convergence in a region obtained by cutting out of the domain a shell containing the fibers. In this subdomain, the convergence appears to be of second order in all three norms.

TABLE I

Convergence Study

	Norm		
	L_1	L_2	Max
$\frac{\|\mathbf{u}_{32} - \mathbf{u}_{64}\|}{\|\mathbf{u}_{64}\|}$	0.451	0.607	0.887
$\frac{\|\mathbf{u}_{64} - \mathbf{u}_{128}\|}{\|\mathbf{u}_{128}\|}$	0.192	0.285	0.587
Convergence ratio	0.426	0.470	0.662

TABLE II

Convergence Study away from the Boundary

	Norm		
	L_1	L_2	Max
$\frac{\Vert \mathbf{u}_{32} - \mathbf{u}_{64} \Vert}{\Vert \mathbf{u}_{64} \Vert}$	0.243	0.309	0.696
$\frac{\Vert \mathbf{u}_{64} - \mathbf{u}_{128} \Vert}{\Vert \mathbf{u}_{128} \Vert}$	0.062	0.079	0.155
Convergence ratio	0.255	0.256	0.223

The foregoing convergence results involve only the computed velocity, not the pressure. In the numerical method used here, the pressure is merely an intermediate variable in the velocity computation. Past experience with the fluid dynamics subroutine by itself suggests that the computed pressure has a substantially larger relative error than the computed velocity [12].

Conclusions and Future Work

The principal conclusion is that the methods described in this paper can be used to study the interaction between a viscous, incompressible fluid and an immersed system of massless, elastic fibers. Since our goal is to model the heart, the next step will be to make the fibers contractile. The results of such computations will be presented as part II of this series. Finally (part III), we shall have to assemble an arrangement of contractile fibers that constitutes a realistic model of the heart. Although it is always hard to predict the future, we believe that the methods of this paper will be adequate for these more complicated problems.

Acknowledgments

This work was supported primarily by the National Institutes of Health under research grant HL17859. We are also indebted to the National Science Foundation for partial support under research grant DMS-8312229.

References

1. C. E. Thomas, *Amer. J. Anatomy* **101**, 17 (1957).
2. D. D. Streeter, Jr., W. E. Powers, M. A. Ross, and F. Torrent–Guasp, "Three-Dimensional Fiber Orientation in the Mammalian Left Ventricular Wall," in *Cardiovacular System Dynamics*, edited by J. Baan *et al.* (MIT Press, Cambridge, MA, 1978), p. 73.

3. K. Sagawa, H. Suga, and K. Nakayama, "Instantaneous Pressure–Volume Ratio of the Ventricle versus Instantaneous Force-Length Relation of Papillary Muscle," in *Cardiovascular System Dynamics*, edited by J. Baan *et al.* (MIT Press, Cambridge, MA, 1978), p. 99.
4. C. S. Peskin, *J. Comput. Phys.* **25**, 220 (1977).
5. C. S. Peskin and D. M. McQueen, *J. Comput. Phys.* **37**, 113 (1980).
6. D. M. McQueen, C. S. Peskin, and E. L. Yellin, *Amer. J. Physiol.* **242**, H1095 (1982).
7. D. M. McQueen and C. S. Peskin, *J. Thorac. Cardiovasc. Surg.* **86**, 126 (1983).
8. D. M. McQueen and C. S. Peskin, *Scand. J. Thorac. Cardiovasc. Surg.* **19**, 139 (1985).
9. J. S. Meisner, D. M. McQueen, Y. Ishida, H. O. Vetter, U. Bortolotti, J. A. Strom, R. W. M. Frater, C. S. Peskin, and E. L. Yellin, *Amer. J. Physiol.* **249**, H604 (1985).
10. A. J. Chorin, *Math. Comput.* **22**, 745 (1968).
11. A. J. Chorin, *Math. Comput.* **23**, 341 (1969).
12. S. Greenberg, D. M. McQueen, and C. S. Peskin, "Three-dimensional fluid dynamics in a two-dimensional amount of central memory," in *Wave Motion: Theory, Modeling, and Computation. Proceedings, Conference in Honor of the 60th Birthday of Peter D. Lax* (Springer-Verlag, New York, 1987), p. 185.
13. D. Fischer, G. Golub, O. Hald, C. Leiva, and O. Widlund, *Math. Comput.* **28**, 349 (1974).
14. D. P. O'Leary and O. Widlund, *Math. Comput.* **33**, 849 (1979).
15. A. Gottlieb, "An overview of the NYU Ultracomputer Project," in *Experimental Parallel Computing Architectures*, edited by J. J. Dongarra (North–Holland, Amsterdam, 1987).
16. E. J. Horowitz, *J. Comut. Phys.* **68**, 56 (1987).
17. A. Nishiguchi, S. Orii, and T. Yabe, *J. Comput. Phys.* **61**, 519 (1985).
18. C. S. Peskin, *Commun. Pure Appl. Math.* **42**, 79 (1989).
19. R. S. Chadwick, *Biophys. J.* **39**, 279 (1982).