Journal of Computational Physics 281 (2015) 917–941

Contents lists available at ScienceDirect

Journal of Computational Physics

www.elsevier.com/locate/jcp

An efficient parallel immersed boundary algorithm using a pseudo-compressible fluid solver ☆

Jeffrey K. Wiens *, John M. Stockie

Department of Mathematics, Simon Fraser University, 8888 University Drive, Burnaby, BC, V5A 1S6, Canada

ARTICLE INFO

Article history:
Received 16 May 2013
Received in revised form 26 August 2014
Accepted 29 October 2014
Available online 4 November 2014

MSC:
74F10
76M12
76D27
65Y05

Keywords:
Immersed boundary method
Fluid–structure interaction
Fractional step method
Pseudo-compressibility method
Domain decomposition
Parallel algorithm

ABSTRACT

We propose an efficient algorithm for the immersed boundary method on distributed-memory architectures that has the computational complexity of a completely explicit method and also has excellent parallel scaling. The algorithm utilizes the pseudo-compressibility method recently proposed by Guermond and Minev that uses a directional splitting strategy to discretize the incompressible Navier–Stokes equations, thereby reducing the linear systems to a series of one-dimensional tridiagonal systems. We perform numerical simulations of several fluid–structure interaction problems in two and three dimensions and study the accuracy and convergence rates of the proposed algorithm. We also compare the proposed algorithm with other second-order projection-based fluid solvers. Lastly, the execution time and scaling properties of the proposed algorithm are investigated and compared to alternate approaches.

© 2014 Elsevier Inc. All rights reserved.

1. Introduction

The immersed boundary (IB) method is a mathematical framework for studying fluid–structure interaction that was originally developed by Peskin to simulate the flow of blood through a heart valve [49]. The IB method has been used in a wide variety of biofluids applications including blood flow through heart valves [26,49], aerodynamics of the vocal cords [14], sperm motility [12], insect flight [41], and jellyfish feeding dynamics [32]. The method is also increasingly being applied in non-biological applications [43].

The immersed boundary equations capture the dynamics of both fluid and immersed elastic structure using a mixture of Eulerian and Lagrangian variables: the fluid is represented using Eulerian coordinates that are fixed in space, and the immersed boundary is described by a set of moving Lagrangian coordinates. An essential component of the model is the Dirac delta function that mediates interactions between fluid and IB quantities in two ways. First of all, the immersed boundary exerts an elastic force (possibly singular) on the fluid through an external forcing term in the Navier–Stokes

☆ We acknowledge support from the Natural Sciences and Engineering Research Council of Canada (NSERC) (Grant No. 238776-2011) through a Postgraduate Scholarship (JKW) and a Discovery Grant (JMS). The numerical simulations in this paper were performed using computing resources provided by WestGrid and Compute Canada.

* Corresponding author.
E-mail addresses: jwiens@sfu.ca (J.K. Wiens), jstockie@sfu.ca (J.M. Stockie).
URLs: http://www.jkwiens.com/ (J.K. Wiens), http://www.math.sfu.ca/~stockie (J.M. Stockie).

http://dx.doi.org/10.1016/j.jcp.2014.10.058
0021-9991/© 2014 Elsevier Inc. All rights reserved.

equations that is calculated using the current IB configuration. Secondly, the immersed boundary is constrained to move at the same velocity as the surrounding fluid, which is just the no-slip condition. The greatest advantage of this approach is that when the governing equations are discretized, no boundary-fitted coordinates are required to handle the solid structure and the influence of the immersed boundary on the fluid is captured solely through an external body force.

When devising a numerical method for solving the IB equations, a common approach is to use a fractional-step scheme in which the fluid is decoupled from the immersed boundary, thereby reducing the overall complexity of the method. Typically, these fractional-step schemes employ some permutation of the following steps:

- *Velocity interpolation*: the fluid velocity is interpolated onto the immersed boundary.
- *IB evolution*: the immersed boundary is evolved in time using the interpolated velocity field.
- *Force spreading*: calculate the force exerted by the immersed boundary and spreads it onto the nearby fluid grid points, with the resulting force appearing as an external forcing term in the Navier–Stokes equations.
- *Fluid solve*: evolve the fluid variables in time using the external force calculated in the force spreading step.

Algorithms that fall into this category include Peskin's original method [49] as well as algorithms developed by Lai and Peskin [36], Griffith and Peskin [27], and many others.

A popular recent implementation of fractional-step type is the IBAMR code [35] that supports distributed-memory parallelism and adaptive mesh refinement. This project grew out of Griffith's doctoral thesis [21] and was outlined in the papers [24,27]. In the original IBAMR algorithm, the incompressible Navier–Stokes equations are solved using a second-order accurate projection scheme in which the viscous term is handled with an L-stable discretization [40,61] while an explicit second-order Godunov scheme [11,42] is applied to the nonlinear advection terms. The IB evolution equation is then integrated in time using a strong stability-preserving Runge–Kutta method [20]. Since IBAMR's release, drastic improvements have been made that increase both the accuracy and generality of the software [23,26].

Fractional-step schemes often suffer from a severe time step restriction due to numerical stiffness that arises from an explicit treatment of the immersed boundary in the most commonly used splitting approaches [58]. Because of this limitation, many researchers have proposed new algorithms that couple the fluid and immersed boundary together in an implicit fashion, for example [8,34,37,44,47]. These methods alleviate the severe time step restriction, but do so at the expense of solving large nonlinear systems of algebraic equations in each time step. Although these implicit schemes have been shown in some cases to be competitive with their explicit counterparts [48], there is not yet sufficient evidence to prefer one approach over the other, especially when considering parallel implementations.

Projection methods are a common class of fractional-step schemes for solving the incompressible Navier–Stokes equations, and are divided into two steps. First, the discretized momentum equations are integrated in time to obtain an intermediate velocity field that in general is not divergence-free. In the second step, the intermediate velocity is projected onto the space of divergence-free fields using the Hodge decomposition. The projection step typically requires the solution of large linear systems in each time step that are computationally costly and form a significant bottleneck in CFD codes. This cost is increased even more when a small time step is required for explicit implementations. Note that even though some researchers make use of unsplit discretizations of the Navier–Stokes equations [23,48], there is significant benefit to be had by using a split-step projection method as a preconditioner [22]. Therefore, any improvements made to a multi-step fluid solver can reasonably be incorporated into unsplit schemes as well.

In this paper, we develop a fractional-step IB method that has the computational complexity of a completely explicit method and exhibits excellent parallel scaling on distributed-memory architectures. This is achieved by abandoning the projection method paradigm and instead adopting the pseudo-compressible fluid solver developed by Guermond and Minev [28,29]. Pseudo-compressibility methods relax the incompressibility constraint by perturbing it in an appropriate manner, such as in Temam's penalty method [59], the artificial compressibility method [9], and Chorin's projection method [10,53]. The Guermond–Minev algorithm differentiates itself by employing a directional-splitting strategy, thereby permitting the linear systems of size $N^d \times N^d$ typically arising in projection methods (where $d = 2$ or 3 is the problem dimension) to be replaced with a set of one-dimensional tridiagonal systems of size $N \times N$. These tridiagonal systems can be solved efficiently on distributed-memory computing architectures by combining Thomas's algorithm with a Schur-complement technique. This allows the proposed IB algorithm to efficiently utilize parallel resources [18]. The only serious limitation of the IB algorithm is that it is restricted to simple geometries and boundary conditions due to the directional-splitting strategy adopted by Guermond and Minev. However, since IB practitioners often use a rectangular fluid domain with periodic boundary conditions, this is not a serious limitation. Instead, the IB method provides a natural setting to leverage the strengths of the Guermond–Minev algorithm allowing complex geometries to be incorporated into the domain through an immersed boundary. This is a simple alternative to the fictitious domain procedure proposed by Angot et al. [1].

In Section 2, we begin by stating the governing equations for the immersed boundary method. We continue by describing our proposed numerical scheme in Section 3 where we incorporate the higher-order rotational form of the Guermond–Minev algorithm that discretizes an $\mathcal{O}(\Delta t^2)$ perturbation of the Navier–Stokes equations to yield a formally $\mathcal{O}(\Delta t^{3/2})$ accurate method. As a result, the proposed method has convergence properties similar to a fully second-order projection method, while maintaining the computational complexity of a completely explicit method. In Section 4, we discuss implementation details and highlight the novel aspects of our algorithm. Finally, in Sections 5 and 6, we demonstrate the accuracy, efficiency and parallel performance of our method by means of several test problems in 2D and 3D.

Download English Version:

https://daneshyari.com/en/article/6932112

Download Persian Version:

https://daneshyari.com/article/6932112

Daneshyari.com