Supplementary material: Further details of the numerical method

The flow considered here is governed by the three-dimensional, viscous incompressible momentum equation and the continuity equation,

$$\begin{aligned} \frac{\partial u_i}{\partial t} + \frac{\partial u_j u_i}{\partial x_j} &= -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j^2}, \\ \frac{\partial u_i}{\partial x_i} &= 0, \end{aligned} \tag{1}$$

where u_i is the velocity, ρ and ν are the constant density and viscosity, and p is the pressure. The governing equations are discretized on a nonuniform Cartesian grid using a cell-centered, non-staggering arrangement of the primitive variables, u_i and p.

The Cartesian grid covers the entire computational domain, including both the fluid region and the solid body. A standard second-order central difference scheme is used to discretize all the spatial derivatives at the nodes located in the bulk flow region. The incompressible momentum equation is integrated in time using a variation of Chorin's projection method which consists of three sub-steps. In the first sub-step, an advection–diffusion equation is solved in the absence of the pressure, and an intermediate velocity field, u_i^*, is obtained. In this step, both the nonlinear advection terms and the viscous terms are discretized using the Crank–Nicolson scheme to improve the numerical stability. The discrete equation is written as

$$\frac{u_i^* - u_i^n}{\Delta t} + \frac{1}{2}\left[\frac{\delta (U_j u_i)^*}{\delta x_j} + \frac{\delta (U_j u_i)^n}{\delta x_j}\right] = \frac{\nu}{2}\left[\frac{\delta}{\delta x_j}\left(\frac{\delta u_i^*}{\delta x_j}\right) + \frac{\delta}{\delta x_j}\left(\frac{\delta u_i^n}{\delta x_j}\right)\right], \tag{2}$$

where U_j is the velocity discretized at the face center of a computational cell, and $\frac{\delta}{\delta x_j}$ represents a finite-difference approximation of the spatial derivative using a second-order central scheme. The three components of the face-centered velocity, U_j, is obtained by computing the linear average of u_j along the j-direction. The nonlinear algebraic system is solved by a successive substitution approach. That is, the system (2) is first linearized with U_j^* computed from available u_i^* and held constant, and then the entire linear system is iterated for once using the Gauss–Seidel method before U_j^* is updated for the next iteration.

In the second sub-step, a projection function is solved as an approximation of the pressure,

$$\frac{\delta}{\delta x_j}\left(\frac{\delta p^{n+1}}{\delta x_j}\right) = \frac{\rho}{\Delta t}\frac{\delta U_j^*}{\delta x_j}, \tag{3}$$

and an inhomogeneous Neumann boundary condition is imposed at all boundaries. The special treatment at the immersed fluid–solid boundary will be discussed in next sub-section. The Poisson equation (3) is solved with a parallel sparse matrix solver, Aztec. Once the pressure is obtained, the cell-centered velocity is updated as

$$u_i^{n+1} = u_i^* - \frac{\Delta t}{\rho}\frac{\delta p^{n+1}}{\delta x_i}, \tag{4}$$

and the final face-centered velocity, U_i^{n+1}, is updated by averaging u_i^{n+1} along the j-direction.

The fluid-solid interface is represented by a set of Lagrangian marker points and 3-node triangular elements. To implement the boundary conditions at the interface, "ghost nodes" outside

the fluid region are defined at each time step, at which the flow variables are extrapolated [1, 2]. To suppress the numerical oscillations that may happen when solving a moving-boundary problem, "hybrid nodes" are defined inside the fluid region, at which the flow variables are weighted averages between the interpolated solution and the solution to the Navier–Stokes equations.

References

[1] R. Mittal, H. Dong, M. Bozkurttas, F. M. Najjar, A. Vargas, and A. vonLoebbeck. A versatile sharp interface immersed boundary method for incompressible flows with complex boundaries. *J. Comput. Phys.*, 227(10), 2008. 4825-4852.

[2] H. Luo, H. Dai, P. Ferreira de Sousa, and B. Yin. On numerical oscillation of the direct-forcing immersed-boundary method for moving boundaries. *Comput. & Fluids*, 56:61–76, 2012.