6. Fluid-structure interaction: the immersed boundary method

The immersed boundary method (IBM) is presented in this chapter. Its important task is to provide the bidirectional coupling between the fluid motion and the membrane dynamics. The basic idea of this coupling is that the membranes move along with the ambient fluid (no-slip condition at the interface) and that any force acting on the membranes also acts on the fluid and vice versa (Newton's third law). After an overview in section 6.1, the governing equations are motivated in section 6.2. The discretization of the IBM equations is discussed in section 6.3, followed by an alternative motivation of these equations based on statistical physics in section 6.4.

6.1. Overview of the immersed boundary method

The IBM has been introduced by Peskin in the 1970s to simulate the blood flow around heart valves [78, 167, 168]. The purpose of the IBM is the computational modeling of fluid-structure interactions. The mathematical basis consists of two coordinate systems, an Eulerian and a Lagrangian system. The Eulerian variables are defined on a fixed Cartesian mesh while the Lagrangian quantities live on a curvilinear or unstructured mesh which may move on top of the Eulerian mesh. The Eulerian mesh is used to solve the Navier-Stokes equations (NSE) while the Lagrangian system captures the immersed structures (e.g., membranes) in the fluid. In general, the two meshes are not conform (fig. 6.1), which raises the need of interpolations when information is transferred from one mesh to the other. The IBM is a front-tracking coupling method, i.e., the interface location is explicitly known. Unlike the bounce-back scheme (section 5.4), the IBM acts via body forces on the fluid in order to enforce the boundary conditions (BCs) resulting from the presence of the structures immersed in the fluid.

One of the basic assumptions of the IBM is the validity of the no-slip condition, i.e., each immersed structure element moves with the same velocity as the ambient fluid. Conversely, the structure exerts a force on the nearby fluid which enters the NSE as an external forcing term. This force mimics the momentum exchange of the fluid at the structure surface and can also be interpreted as the force obtained from the constitutive model of the elastic immersed material (chap. 7). The forces acting on the fluid are originally computed in the Lagrangian frame of the structure. Thus, the forces have to be spread to the Eulerian mesh in order to solve the NSE. The resulting fluid velocity has to be interpolated back to the Lagrangian mesh for the update of the structure element positions. The underlying interpolation functions (also called interpolation stencils) for the force spreading and the velocity interpolation have to be defined in a consistent way. The algorithm and discretizations are given in sections 6.2 and 6.3.

The IBM offers a number of advantages. First, it can be combined with any Navier-Stokes solver which supports external forcing (e.g., the LBM). Second, the constitutive behavior of the immersed elastic structures is not restricted by the IBM. In that sense, the IBM is a pure coupling method obeying the no-slip condition at the fluid-structure interface. A further advantage is that there are no additional, unphysical parameters in the IBM which have to be tuned or optimized. The implementation of the IBM is comparably simple, and its numerical overhead is small. Particularly with regard to the simulation of suspensions of deformable particles at high

T. Krüger, *Computer Simulation Study of Collective Phenomena in Dense Suspensions of Red Blood Cells under Shear,* DOI 10.1007/978-3-8348-2376-2_6,

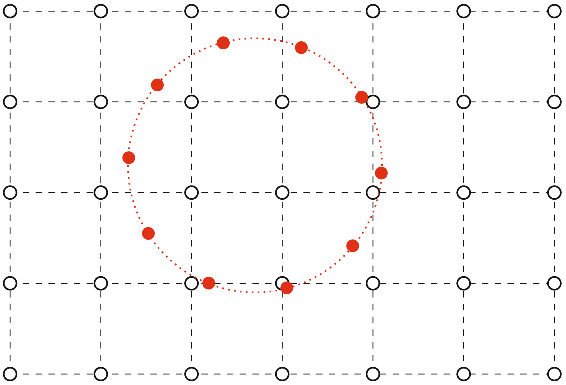


Fig. 6.1.: Eulerian and Lagrangian meshes. A circular object (large dotted circle) of radius $1.4\Delta x$ is described by an ensemble of Lagrangian points (red circles) on the background of the regular Eulerian grid (dashed lines and white circles). Generally, the Lagrangian mesh is not conform with the Eulerian grid.

volume fractions, an important advantage of the IBM is that arbitrarily complex fluid-structure interfaces can be modeled. Numerical problems related to the simulation of dense suspensions of deformable particles are discussed separately in section 8.6.

The IBM has been used before in connection with the LBM or other Navier-Stokes solvers in order to simulate suspensions of soft particles [77, 80, 89, 169, 170]. Apart from the application being in the focus of this work (simulation of deformable particles), the IBM is regularly employed to model rigid objects immersed in a fluid. Peskin [78] provides a rich collection of applications and extensions of the IBM such as improving the time-stepping scheme [171], the volume conservation [172], sharpening its interface [173], implementing local grid refinement [174], and parallelizing the IBM [175]. Some applications include Peskin's original work about fluid dynamics of heart valves [167], simulation of particle suspensions [176], platelet aggregation during blood clotting [177], flow in elastic blood vessels [178], simulation of biofilms [179], and flow past a cylinder [180], to name only a few. Additional applications of the IBM are reviewed in a more recent article by Mittal and Iaccarino [181].

6.2. Governing equations of the immersed boundary method

A thorough mathematical derivation of the IBM formalism has been provided by Peskin [78]. It shall not be repeated here. In the following, solely the governing equations and some remarks are collected for a special case of the IBM. It is assumed that

- the immersed structure is a 2D membrane immersed in 3D space and that
- the density of the membrane equals the density of the ambient fluid.

Both assumptions are reasonable when red blood cells (RBCs) are simulated (chap. 4). However, in general, the IBM may also be applied to situations where the above assumptions cannot be made. The corresponding generalized equations can be found in [78].

Let $\boldsymbol{X}$ be the coordinate of a fixed point in the Eulerian frame and $\boldsymbol{x}(r, s, t)$ the position of a marker point comoving with the Lagrangian mesh. (r, s) are two-dimensional curvilinear coordinates for the membrane. The exact form of the curvilinear coordinate system is not important since it will not appear in the discretized equations at the end. Still, it is instructive to use (r, s) in order to better understand the IBM formalism. The governing equations for the fluid-membrane coupling read [78]

$$\boldsymbol{f}(\boldsymbol{X},t) = \int \mathrm{d}r\,\mathrm{d}s\,\tilde{\boldsymbol{f}}(r,s,t)\delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t)), \tag{6.1}$$

$$\dot{\boldsymbol{x}}(r,s,t) = \int \mathrm{d}^3 X\,\boldsymbol{u}(\boldsymbol{X},t)\delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t)). \tag{6.2}$$

Here, $\delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t))$ is the three-dimensional Dirac delta distribution. $\boldsymbol{u}(\boldsymbol{X},t)$ is the velocity of the fluid at coordinate $\boldsymbol{X}$ at time t, and $\dot{\boldsymbol{x}}(r,s,t)$ is the velocity of the Lagrangian marker point $\boldsymbol{x}(r,s,t)$. $\boldsymbol{f}(\boldsymbol{X},t)$ is the force density (force per volume) acting on the fluid at coordinate $\boldsymbol{X}$ and time t. The force density (force per area) in the Lagrangian system at position $\boldsymbol{x}(r,s,t)$ is denoted by $\tilde{\boldsymbol{f}}(r,s,t)$. Eq. (6.2) resembles the no-slip condition at the membrane surface.

It has to be noted that eq. (6.1) and eq. (6.2) behave differently, even though they have the same interaction function $\delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t))$. For a 2D membrane, force densities are area densities. Thus, on the one hand, the force density $\boldsymbol{f}(\boldsymbol{X},t)$ on the left-hand-side of eq. (6.1) is singular like a one-dimensional delta function since the integral is only 2D. On the other hand, the velocities $\dot{\boldsymbol{x}}(r,s,t)$ and $\boldsymbol{u}(\boldsymbol{X},t)$ in eq. (6.2) are both finite. The transformation in eq. (6.1) is called spreading, and the transformation in eq. (6.2) is called interpolation [78].

For the simulations in the present work, eq. (6.1) and eq. (6.2) form the interaction equations between the membranes and the ambient fluid. In the next step, it has to be discussed how the mathematical relations can be discretized in order to use them in numerical simulations.

6.3. Discretization of the immersed boundary method

The discretization of the IBM equations, eq. (6.1) and eq. (6.2), is necessary to implement the model into a numerical scheme. Especially, a reasonable discretized delta function has to be found. In the following, the spatial discretization scheme will be discussed. The time discretization is shortly presented at the end of this section. Omitted intermediate steps and further comments can be found in [78].

In their discretized forms, the spreading and interpolation equations, eq. (6.1) and eq. (6.2), read

$$\boldsymbol{f}(\boldsymbol{X},t) = \sum_{r,s} \tilde{\boldsymbol{f}}(r,s,t)\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t))\,\Delta r\,\Delta s, \tag{6.3}$$

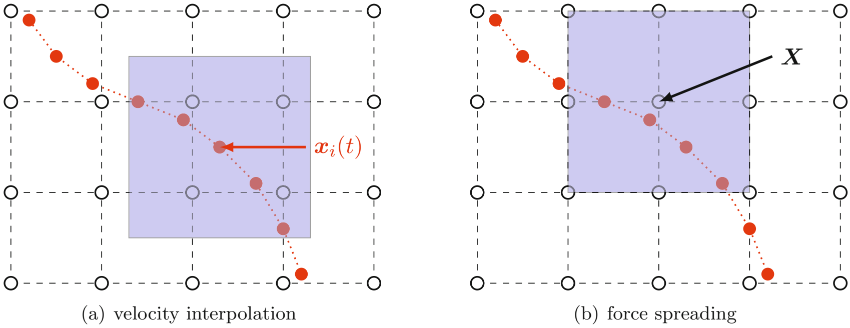
$$\dot{\boldsymbol{x}}(r,s,t+\Delta t) = \sum_{\boldsymbol{X}} \boldsymbol{u}(\boldsymbol{X},t+\Delta t)\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t))\,\Delta x^3 \tag{6.4}$$

where the integration is replaced by a discrete sum and $\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}(r,s,t))$ is the discretized delta function. The time increment is denoted Δt. Δx, Δr, and Δs are the Eulerian lattice constant and the sizes of the Lagrangian membrane elements, respectively. The velocity interpolation and force spreading are illustrated in fig. 6.2.

It has to be stressed again that $\tilde{\boldsymbol{f}}(r,s,t)$ is the force density of the membrane (force per area), whereas $\boldsymbol{F}(r,s,t) = \tilde{\boldsymbol{f}}(r,s,t)\,\Delta r\,\Delta s$ is the force acting on the membrane area defined by $(\Delta r, \Delta s)$. In the discretized formulation, there is a given number of points (nodes) defining the membrane surface. Each of these points can be addressed either by its coordinates (r,s) or by a node index i. A curvilinear coordinate system is not necessarily required, and the mesh can be unstructured (which will be the case throughout this thesis, cf. section 8.3). Replacing (r,s) by the node index i and setting $\Delta x = \Delta t = 1$ in the following, the discretized IBM equations read

$$\boldsymbol{f}(\boldsymbol{X},t) = \sum_i \boldsymbol{F}_i(t)\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}_i(t)), \tag{6.5}$$

$$\dot{\boldsymbol{x}}_i(t+1) = \sum_{\boldsymbol{X}} \boldsymbol{u}(\boldsymbol{X},t+1)\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}_i(t)). \tag{6.6}$$



(a) velocity interpolation

(b) force spreading

Fig. 6.2.: Velocity interpolation and force spreading in the immersed boundary method. A membrane patch is denoted by the curved dotted line. During (a) velocity interpolation, each membrane node (red circles) at position $\boldsymbol{x}_i(t)$ collects velocity information of all lattice nodes within a finite range (square box). During (b) force spreading, each lattice node (white circles) at fixed position $\boldsymbol{X}$ collects force information of all membrane nodes within a finite range (square box). The weights of the interpolation/spreading contributions are given by the value of the discrete delta function, e.g., eq. (6.11).

In this simplified picture, $\boldsymbol{F}_i(t)$ denotes the total force acting on node i which is located at position $\boldsymbol{x}_i(t)$ and has velocity $\dot{\boldsymbol{x}}_i(t)$.

The coupled fluid-membrane system is solved iteratively. For that reason, the positions at the old time step, $\boldsymbol{x}_i(t)$, are used in eq. (6.6) to update the membrane velocity and obtain its value at the next time step, $\dot{\boldsymbol{x}}_i(t + \Delta t)$. These algorithmic details are further elaborated on in section 8.1.

Obviously, a discussion of the discretized delta function $\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}_i(t))$ is still missing in order to complete the discretization. According to Peskin [78], the discretized delta function has to obey a series of restrictions and properties, e.g., the force and the torque should be the same when evaluated in the Eulerian and the Lagrangian systems. Additionally, the discretized delta function should be continuous, which assures that there are neither jumps in the velocity nor in the force when the membrane points move between Eulerian lattice nodes. A complete list of those restrictions and their mathematical significance can be found in [78]. In order to increase computational efficiency, the discretized delta function should have a compact support, i.e., for each Lagrangian mesh point, only the Eulerian fluid points within a finite range should be considered and vice versa. The smallest possible support for realizing all of Peskin's postulates is four Eulerian grid points along each spatial dimension. It can be shown that the same discretized delta function has to be used for spreading and interpolation [78].

One of the major assumptions is that the discretized delta function can be factorized,

$$\delta_\Delta(\boldsymbol{x}) = \phi(x)\phi(y)\phi(z). \tag{6.7}$$

This ansatz is not essential, but the computations become simpler, and the cubic lattice structure is taken into account.

It is possible to find various discretized delta functions which have different interpolation ranges. The so-called 4-point stencil reads

$$\phi_4(x) = \begin{cases} \frac{1}{8}\left(3 - 2|x| + \sqrt{1 + 4|x| - 4x^2}\right) & \text{for} \quad 0 \leq |x| \leq 1, \\ \frac{1}{8}\left(5 - 2|x| - \sqrt{-7 + 12|x| - 4x^2}\right) & \text{for} \quad 1 \leq |x| \leq 2, \\ 0 & \text{for} \quad 2 \leq |x|. \end{cases} \tag{6.8}$$

This discretization fulfills all restrictions which are stated by Peskin [78]. The interpolation function

$$\phi_4^c(x) = \begin{cases} \frac{1}{4}(1+\cos(\frac{\pi x}{2})) & \text{for} \quad 0 \leq |x| \leq 2, \\ 0 & \text{for} \quad 2 \leq |x| \end{cases} \tag{6.9}$$

which is an excellent approximation of eq. (6.8) is regularly used in the literature instead.

One can construct an interaction function with a support of three lattice nodes,

$$\phi_3(x) = \begin{cases} \frac{1}{3}(1+\sqrt{1-3x^2}) & \text{for} \quad 0 \leq |x| \leq \frac{1}{2}, \\ \frac{1}{6}(5-3|x|-\sqrt{-2+6|x|-3x^2}) & \text{for} \quad \frac{1}{2} \leq |x| \leq \frac{3}{2}, \\ 0 & \text{for} \quad \frac{3}{2} \leq |x|. \end{cases} \tag{6.10}$$

Similar to $\phi_4(x)$ and $\phi_4^c(x)$, $\phi_3(x)$ is symmetric, $\phi(-x) = \phi(x)$, and it has a continuous first derivative. These two properties have not been claimed, but come in handy. Peskin [78] and Dünweg and Ladd [182] state that Navier-Stokes solvers depending on a central difference scheme cannot use $\phi_3(x)$ since the number of support points is odd. The LBM, however, is not concerned with this restriction. Obviously, there are some advantages of $\phi_3(x)$ over $\phi_4(x)$. First, the envelope volume is decreased from 64 to 27 grid points in 3D reducing the computational overhead. Second, the membrane interface width is decreased. It is reported by Dünweg and Ladd [182] that $\phi_3(x)$ results in hydrodynamics which is nearly as accurate as that for $\phi_4(x)$.

If maximum efficiency is required, it is also possible to use a two-point linear interaction function with a support of two lattice nodes along each axis,

$$\phi_2(x) = \begin{cases} 1-|x| & \text{for} \quad 0 \leq |x| \leq 1, \\ 0 & \text{for} \quad 1 \leq |x|. \end{cases} \tag{6.11}$$

This way, the cubic lattice structure becomes more visible, i.e., the translational symmetry is violated more strongly than for ϕ_3 or ϕ_4 [78, 182]. Obviously, ϕ_2 does not have a continuous derivative, but only eight lattice nodes have to be considered for spreading and interpolation. The shapes of the three discretized delta functions ϕ_2, ϕ_3, and ϕ_4 are shown in fig. 6.3.

In the present work, for reasons of numerical efficiency and for reducing the numerical membrane interface width, usually ϕ_2 is employed (as also in, e.g., [183]). It should be noted that even different discretized delta functions may be used [78, 184, 185].

When the IBM is combined with the LBM, the explicit Euler method is usually employed for the time discretization (e.g., [77]),

$$\boldsymbol{x}_i(t+\Delta t) = \boldsymbol{x}_i(t) + \dot{\boldsymbol{x}}_i(t+\Delta t)\Delta t. \tag{6.12}$$

However, there exist different time integration schemes, [186, 187].

6.4. Connection between the immersed boundary method and viscous coupling

Eq. (6.1) and eq. (6.2) can also be derived from a more general method. Dünweg and Ladd [182] use a fluctuating LBM in connection with particles dissipatively coupled to the fluid. These particles experience a drag force if their velocity differs from the ambient fluid velocity. Dropping the time from the following equations for simplicity, the drag force acting on particle i is

$$\boldsymbol{F}_i^{\text{d}} = -\Gamma_i \left(\frac{\boldsymbol{p}_i}{m_i} - \boldsymbol{u}(\boldsymbol{x}_i) \right) \tag{6.13}$$

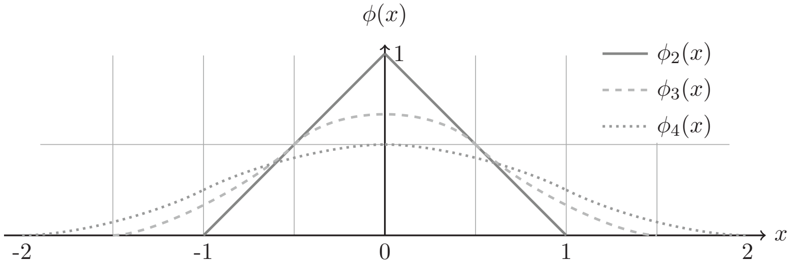


Fig. 6.3.: Discrete delta functions for the immersed boundary method (IBM). The 2-point (solid), 3-point (dashed), and 4-point (dotted) discrete delta functions for the IBM are shown.

where Γ_i is the drag coefficient for the particle, $\boldsymbol{p}_i$ is its momentum, m_i is its mass, and $\boldsymbol{u}(\boldsymbol{x}_i)$ is the fluid velocity at the position of the particle. The equations of motion for the particles are

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{x}_i = \frac{1}{m_i}\boldsymbol{p}_i, \tag{6.14}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{p}_i = \boldsymbol{F}_i^{\mathrm{c}} + \boldsymbol{F}_i^{\mathrm{d}} + \boldsymbol{F}_i^{\mathrm{f}}. \tag{6.15}$$

Here, $\boldsymbol{F}_i^{\mathrm{c}}$ is the conservative force acting on the particle (e.g., due to an external potential or particle interactions), and $\boldsymbol{F}_i^{\mathrm{f}}$ is a Langevin noise for the particle. According to [182], the force density applied to the fluid at Eulerian coordinate $\boldsymbol{X}$ is computed numerically from

$$\boldsymbol{f}(\boldsymbol{X}) = -\sum_i \left(\boldsymbol{F}_i^{\mathrm{d}}(\boldsymbol{x}_i) + \boldsymbol{F}_i^{\mathrm{f}}(\boldsymbol{x}_i)\right)\delta_\Delta(\boldsymbol{X} - \boldsymbol{x}_i) \tag{6.16}$$

with the same discretized delta functions as in section 6.3. It has been shown by Dünweg and Ladd [182] that the fluctuation-dissipation theorem holds for this coupled system. In the following, it will be inferred that the IBM is formally a special case of the viscous coupling. Consequently, the fluctuation-dissipation theorem should also hold for the IBM [188].

Eq. (6.14) and eq. (6.15) can be combined to give

$$m_i\frac{\mathrm{d}^2}{\mathrm{d}t^2}\boldsymbol{x}_i = \boldsymbol{F}_i^{\mathrm{c}} + \boldsymbol{F}_i^{\mathrm{d}} + \boldsymbol{F}_i^{\mathrm{f}} \tag{6.17}$$

which becomes

$$\boldsymbol{F}_i^{\mathrm{d}} + \boldsymbol{F}_i^{\mathrm{f}} = -\boldsymbol{F}_i^{\mathrm{c}} \tag{6.18}$$

in the over-damped, i.e., massless limit ($m_i \to 0$). Combining eq. (6.18) with eq. (6.16) directly results in the IBM force spreading equation, eq. (6.5), if the conservative force is identified as the membrane force. This finding also justifies that, in the present model, the elastic (conservative) membrane force is used in eq. (6.5) to drive the fluid (chap. 7).

In the last step, keeping $m_i \to 0$, eq. (6.13), eq. (6.14), and eq. (6.18) are combined, which yields

$$\dot{\boldsymbol{x}}_i = \boldsymbol{u}(\boldsymbol{x}_i) + \frac{1}{\Gamma_i}(\boldsymbol{F}_i^{\mathrm{c}} + \boldsymbol{F}_i^{\mathrm{f}}). \tag{6.19}$$

In the high friction limit ($\Gamma_i \to \infty$), the no-slip condition is recovered and with it the IBM velocity interpolation, eq. (6.6).

The parameters m_i and Γ_i are purely numerical without any physical significance. In this sense, the IBM is more natural since it does not introduce additional parameters. However, the time steps for the IBM and the fluid solver are required to be identical. This is not the case in the approach followed by Dünweg and Ladd [182] where the molecular dynamics time step for the particles can be chosen much smaller then the hydrodynamic time step.