INTERNATIONAL JOURNAL FOR NUMERICAL METHODS IN FLUIDS
Int. J. Numer. Meth. Fluids 2008; **58**:263–286
Published online 7 January 2008 in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/fld.1706

Assessment of regularized delta functions and feedback forcing schemes for an immersed boundary method

Soo Jai Shin, Wei-Xi Huang and Hyung Jin Sung*, †

Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology 373-1, Guseong-dong Yuseong-gu, Daejeon 305-701, Korea

SUMMARY

We present an improved immersed boundary method for simulating incompressible viscous flow around an arbitrarily moving body on a fixed computational grid. To achieve a large Courant–Friedrichs–Lewy number and to transfer quantities between Eulerian and Lagrangian domains effectively, we combined the feedback forcing scheme of the virtual boundary method with Peskin's regularized delta function approach. Stability analysis of the proposed method was carried out for various types of regularized delta functions. The stability regime of the 4-point regularized delta function was much wider than that of the 2-point delta function. An optimum regime of the feedback forcing is suggested on the basis of the analysis of stability limits and feedback forcing gains. The proposed method was implemented in a finite-difference and fractional-step context. The proposed method was tested on several flow problems, including the flow past a stationary cylinder, inline oscillation of a cylinder in a quiescent fluid, and transverse oscillation of a circular cylinder in a free-stream. The findings were in excellent agreement with previous numerical and experimental results. Copyright © 2008 John Wiley & Sons, Ltd.

Received 1 August 2007; Revised 30 October 2007; Accepted 6 November 2007

KEY WORDS: immersed boundary method; regularized delta function; feedback forcing; stability analysis; finite-difference method

1. INTRODUCTION

The immersed boundary (IB) method has received much attention in the field of numerical simulations because it can easily handle viscous flows over or inside complex geometries in a Cartesian grid system. In the IB method, no-slip boundary conditions are imposed at the IBs by introducing a momentum forcing into the Navier–Stokes (N–S) equations. Hence, N–S solvers based on a

*Correspondence to: Hyung Jin Sung, Department of Mechanical Engineering, Korea Advanced Institute of Science and Technology 373-1, Guseong-dong Yuseong-gu, Daejeon 305-701, Korea.
†E-mail: hjsung@kaist.ac.kr

Contract/grant sponsor: Creative Research Initiatives of the Korea Science & Engineering Foundation
Contract/grant sponsor: Brain Korea 21
Contract/grant sponsor: Basic Research Program of the Korea Science & Engineering Foundation; contract/grant number: R01-2004-000-10521-0

Cartesian grid system can be easily extended to complex flow geometries without the need for a boundary-conforming grid. Depending on how the momentum forcing is applied, the IB method is classified as either a discrete or continuous forcing approach [1].

In the discrete forcing approach [2–6], the desired velocities at the IB are obtained by interpolating neighboring points, and the momentum forcing is acquired directly from the discretized equation of motion. Kim *et al.* [3] introduced a mass source/sink to satisfy the local continuity near IB points. Huang and Sung [4] improved the accuracy near the IB by using more accurate mass source/sink. Kim and Choi [5] developed an IB method adopting a non-inertial reference frame that is fixed to the body with an arbitrary motion. However, to simulate flow around a moving body using a fixed Eulerian grid system, more complicated interpolation schemes are needed [6].

In contrast to the discrete forcing approach, the continuous forcing approach can be applied in a straightforward manner when simulating flows with moving boundaries. The continuous forcing approach, which was pioneered by Peskin [7] to simulate blood flow in the human heart, has proved to be an efficient method for simulating fluid–structure interactions. In Peskin's classical IB method [7–11], a singular force distribution at arbitrary Lagrangian positions is determined and applied to the flow equations in the fixed reference frame via a regularized delta function. For immersed elastic boundaries, the singular force is just the elastic force that can be derived by the principle of virtual work or a constitutive law such as Hooke's law. For rigid boundaries, however, the constitutive laws for elastic boundaries are not generally well posed. Goldstein *et al.* [12] developed a virtual boundary method that employs a feedback forcing to enforce the no-slip condition at the surface of a structure embedded in a fluid domain. Saiki and Biringen [13] modified the virtual boundary formulation and proposed the so-called 'area-weighted' virtual boundary method. However, both Goldstein *et al.* and Saiki and Biringen reported that this method suffers from a very severe time-step restriction since the amplitude of the feedback forcing needs to be large for proper operation, resulting in a very stiff system. Lee [14] relieved the time-step restriction by exploiting the stability characteristics of the virtual boundary method. Instead of the regularized delta function, in the virtual boundary method the velocities at the surface of a structure are obtained using a bilinear interpolation, and the momentum forcing is extrapolated back to the surrounding grid points by area-weighted averages. Under this approach, the total amount of forcing is not conserved, in contrast to the conservation of the total forcing achieved using the regularized delta function, as will be analyzed in detail later in this paper. Lai and Peskin [9] considered the boundary to be elastic but extremely stiff for simulation of flow with a rigid body. In their method, IB points move according to the local flow velocity, while in the virtual boundary method the boundary points do not move from the body surface. Accordingly, in Peskin's IB method both the position and the velocity of the IB by feedback forcing have to be compensated. On the contrary, in the virtual boundary method only the velocity of the boundary has to be corrected because the boundary remains on the body surface. As a result, the limitation of the computational time step of the virtual boundary method is different from that of Peskin's IB method due to differences of the forcing scheme. Uhlmann [15] retained the use of regularized delta functions in his direct forcing method, which can be regarded as a special case of the feedback forcing scheme. Huang *et al.* [16] proposed an IB formulation for simulating flexible inextensible filaments in a uniform flow. In their method, the Eulerian fluid motion and the Lagrangian filament motion were solved independently and their interaction force was explicitly calculated using a feedback law.

In this study, we sought to find an efficient IB formulation by accounting for the regularized delta functions and feedback forcing schemes. To this end, we made a detailed comparison of

DOI: 10.1002/fld

Peskin's IB method and the virtual boundary method. Stability analysis of the feedback forcing gains was performed for various types of regularized delta functions to relax the computational time-step restriction and to decrease the l_2-norm error and avoid non-growing oscillations. The analytical solution was compared with numerical results obtained using the proposed IB method. From the stability analysis, an optimum region of the feedback forcing gains was obtained. The present method was validated by simulating three flow problems, namely flow past a stationary cylinder, inline oscillation of a cylinder in a quiescent fluid, and transverse oscillation of a circular cylinder in a free-stream.

2. NUMERICAL APPROACH

The computational configuration considered in the present work is a circular solid object embedded in a fluid, as shown in the schematic diagram in Figure 1. The fluid motion is defined on an Eulerian–Cartesian grid, and the fluid–solid interface is discretized using a Lagrangian grid fixed on the body. The fluid–solid interface ∂S is evenly distributed by N_{L} Lagrangian points, which are denoted by

$$X_l \in \partial S, \quad 1 \leqslant l \leqslant N_{\mathrm{L}} \tag{1}$$

Each Lagrangian point has a discrete volume ΔV_l with thickness equal to the mesh width h. N_{L} is selected such that a discrete volume is comparable with a finite volume of the Eulerian grid, i.e. $\Delta V_l \approx h^m$, where m is the number of space dimensions. Although we illustrate only a circular solid object in Figure 1, the present method can be applied to objects of arbitrary shape.

The governing equations are the N–S equations and the continuity equation, which are discretized as

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}+N\mathbf{u}^{n+1}=-Gp^{n+1/2}+\frac{1}{2Re}(L\mathbf{u}^{n+1}+L\mathbf{u}^n)+\mathbf{f}^n \tag{2}$$

$$D\mathbf{u}^{n+1}=0 \tag{3}$$

where $\mathbf{u}$ is the velocity vector, p is the pressure, Re is the Reynolds number, and $\mathbf{f}$ is the momentum forcing applied to enforce the no-slip boundary condition along the IB. In Equations (2) and (3), N is the linearized discrete convective operator, G is the discrete gradient operator, L is the discrete

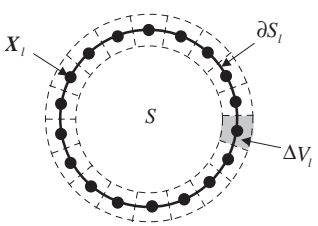

Figure 1. Schematic diagram of a circular solid object S in a fluid domain.

DOI: 10.1002/fld

Laplacian operator, and D is the discrete divergence operator. Here, n denotes the nth time step and Δt denotes the time increment. The discrete spatial operators N, G, L, and D are evaluated using the second-order central finite-difference scheme [17]. The present method is based on the N–S solver adopting the fractional-step method and a staggered Cartesian grid system. A fully implicit time advancement is employed, in which both convection and diffusion terms are advanced using the second-order Crank–Nicholson scheme.

The present numerical algorithm, including the fluid–solid interaction, is formulated as follows:

$$\mathbf{F}^n = \alpha \int (\mathbf{U}^n(\mathbf{X}_l) - \mathbf{U}_\mathrm{d}(\mathbf{X}_l))\,\mathrm{d}t + \beta(\mathbf{U}^n(\mathbf{X}_l) - \mathbf{U}_\mathrm{d}(\mathbf{X}_l)) \tag{4}$$

$$\mathbf{f}^n = \int \mathbf{F}^n \delta_h(\mathbf{x} - \mathbf{X}_l)\,\mathrm{d}s \tag{5}$$

$$\frac{1}{\Delta t}\mathbf{u}^* + N\mathbf{u}^* - \frac{1}{2Re}L\mathbf{u}^* = \frac{1}{\Delta t}\mathbf{u}^n - Gp^{n-1/2} + \frac{1}{2Re}L\mathbf{u}^n + \mathbf{f}^n \tag{6}$$

$$\Delta t\, DG\delta p = D\mathbf{u}^* \tag{7}$$

$$\mathbf{u}^{n+1} = \mathbf{u}^* - \Delta t\, G\delta p \tag{8}$$

$$p^{n+1/2} = p^{n-1/2} + \delta p \tag{9}$$

$$\mathbf{U}^{n+1} = \int \mathbf{u}^{n+1}\delta_h(\mathbf{x} - \mathbf{X}_l^{n+1})\,\mathrm{d}\mathbf{x} \tag{10}$$

where $\mathbf{U}_\mathrm{d}(\mathbf{X}_l)$ is the desired velocity of the Lagrangian point and $\mathbf{u}^*$ is the intermediate fluid velocity. In the present method, the Lagrangian point is required to move in concert with the rigid-body motion of a solid object. This is achieved by means of the feedback forcing in Equation (4), which is calculated so as to make the velocity at the Lagrangian point equal to the desired velocity. Note that the feedback forcing points are located in a staggered fashion like the velocity components defined on a staggered grid. Next, the feedback forcing in the Lagrangian domain is transferred to the Eulerian domain by using the regularized delta function (Equation (5)). In Equations (6)–(9), we solve $\mathbf{u}^{n+1}$ and $p^{n+1/2}$ using the fractional-step method with the momentum forcing calculated explicitly. Velocity and pressure are decoupled by approximate factorization which is based on block LU decomposition. To calculate the intermediate velocity $\mathbf{u}^*$ in Equation (6), the approximate factorization is further extended to the velocity components. It results in a significant reduction in computation time and memory by avoiding the inversion of a large sparse matrix. The pressure Poisson equation (Equation (7)) is then solved by a direct method using fast Fourier transform or a multigrid method. The details of the present N–S solver can be found elsewhere [17]. The Eulerian velocity at the new time step is interpolated to the Lagrangian points by using the regularized delta functions as kernels in the transfer step (Equation (10)). The Lagrangian velocity obtained is then used to calculate the feedback forcing in the next time step. This procedure is repeated.

DOI: 10.1002/fld

The regularized delta function is employed to transfer quantities between Lagrangian and Eulerian locations in Equations (5) and (10), respectively,

$$\mathbf{U}(\mathbf{X}_l)=\sum \mathbf{u}(\mathbf{x})\delta_h(\mathbf{x}-\mathbf{X}_l)h^3, \quad 1\leqslant l\leqslant N_\mathrm{L} \tag{11}$$

$$\mathbf{f}(\mathbf{x})=\sum_{l=1}^{N_\mathrm{L}} \mathbf{F}(\mathbf{X}_l)\delta_h(\mathbf{x}-\mathbf{X}_l)\Delta V_l \tag{12}$$

In the domain where quantities are transferred between Lagrangian and Eulerian locations, a uniform grid with mesh width h is used. In this study, four types of regularized delta functions are chosen:

$$\delta_h(\mathbf{x})=\frac{1}{h^3}\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right) \tag{13}$$

$$\phi(r)=\begin{cases} 1-|r|, & |r|\leqslant 1 \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

$$\phi(r)=\begin{cases} \frac{1}{3}(1+\sqrt{-3r^2+1}), & |r|\leqslant 0.5 \\ \frac{1}{6}(5-3|r|-\sqrt{-3(1-|r|)^2+1}), & 0.5\leqslant|r|\leqslant 1.5 \\ 0 & \text{otherwise} \end{cases} \tag{15}$$

$$\phi(r)=\begin{cases} \frac{1}{8}(3-2|r|+\sqrt{1+4|r|-4r^2}), & 0\leqslant|r|\leqslant 1 \\ \frac{1}{8}(5-2|r|-\sqrt{-7+12|r|-4r^2}), & 1\leqslant|r|\leqslant 2 \\ 0 & \text{otherwise} \end{cases} \tag{16}$$

$$\phi(r)=\begin{cases} \frac{61}{112}-\frac{11}{42}|r|-\frac{11}{56}|r|^2+\frac{1}{12}|r|^3+\frac{\sqrt{3}}{336}(243+1584|r|-748|r|^2 & \\ \quad -1560|r|^3+500|r|^4+336|r|^5-112|r|^6)^{1/2}, & 0\leqslant|r|\leqslant 1 \\ \frac{21}{16}+\frac{7}{12}|r|-\frac{7}{8}|r|^2+\frac{1}{6}|r|^3-\frac{3}{2}\phi(|r|-1), & 1\leqslant|r|\leqslant 2 \\ \frac{9}{8}-\frac{23}{12}|r|+\frac{3}{4}|r|^2-\frac{1}{12}|r|^3+\frac{1}{2}\phi(|r|-2), & 2\leqslant|r|\leqslant 3 \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

Equation (14) shows the 2-point regularized delta function (linear interpolation in each direction), which is similar to the 'area-weighted' virtual boundary method of Saiki and Biringen [13]. Equation (15) shows the 3-point regularized delta function introduced by Roma *et al.* [8], and Equations (16) and (17) are the 4- and 6-point regularized delta functions introduced by Peskin [10]

DOI: 10.1002/fld

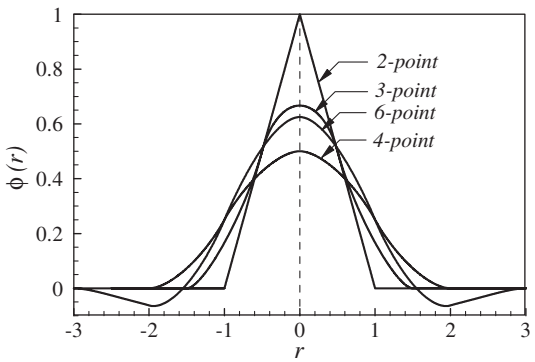

Figure 2. Four types of regularized delta functions.

and Griffith *et al.* [11], respectively. Note that all four types of delta functions have the property

$$\sum \delta_h(\mathbf{x}-\mathbf{X})h^3 = 1 \tag{18}$$

which is the discrete analogue of the basic property of the Dirac delta function. In Figure 2, the value of $\phi(r)$ is maximum at $r=0$ for all functions, and the value of $\phi(0)$ decreases as the points of the delta function increase, except for the 6-point delta function. These properties are kernels of the stability analysis in the following section.

3. STABILITY ANALYSIS

First, we consider the simple case in which the Lagrangian domain is a point located at $x_1+r(x_2-x_1)$ between two Eulerian points x_1 and x_2 in 1-D space, as shown in Figure 3(a). The velocity at the Lagrangian point $U(t)$ is obtained using the 3-point regularized delta function according to Equation (11):

$$U(t)=\phi(-r)u_1(t)+\phi(1-r)u_2(t)+\phi(2-r)u_3(t) \tag{19}$$

where $u_1(t), u_2(t)$, and $u_3(t)$ are velocities at Eulerian points x_1, x_2, and x_3, respectively. The value of r varies between 0.5 and 1.5, depending on the location of the Lagrangian point. The momentum forcing is calculated by the feedback law (Equation (4)) and spreads back to the three nearby Eulerian points using Equation (12) as follows:

$$f_1(t)=\phi(-r)F(t), \quad f_2(t)=\phi(1-r)F(t), \quad f_3(t)=\phi(2-r)F(t) \tag{20}$$

Since only one Lagrangian point is considered, no summation is necessary. Stability analysis for this method should be carried out using the largest Eulerian forcing as this represents the worst case. When the Lagrangian point coincides with the Eulerian point x_2 at $r=1$, the Eulerian forcing reaches its maximum, i.e.

$$f_2(t)\leqslant\phi(0)F(t)=\tfrac{2}{3}F(t) \tag{21}$$

where $\phi(0)=\frac{2}{3}$ for the 3-point regularized delta function.

DOI: 10.1002/fld

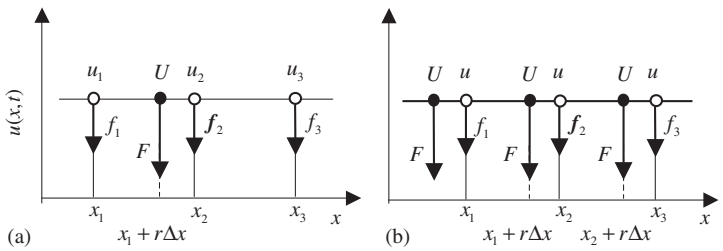

Figure 3. Virtual surface velocity and forcing in the 1-D case: (a) point Lagrangian domain and (b) line Lagrangian domain; •, Lagrangian grid point; ○, Eulerian grid point.

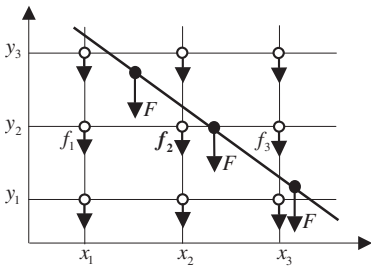

Figure 4. Virtual surface velocity and forcing in the 2-D case: line Lagrangian domain; •, Lagrangian grid point; ○, Eulerian grid point.

Next, we consider the case in which the Lagrangian domain is a line immersed in a 1-D Eulerian domain (Figure 3(b)). For simplicity, we assume that the adjacent Eulerian velocities are uniform. Hence, we have $U(t)=u(t)$ according to Equation (18), regardless of the position of the Lagrangian point. The momentum forcing at the Eulerian point x_2 is then obtained:

$$f_2(t)=\phi(2-r)F(t)+\phi(1-r)F(t)+\phi(r)F(t)=F(t) \tag{22}$$

In Figure 4, this analysis is extended to the 2-D case (the thick line indicates the Lagrangian domain). In this case, we again assume that the adjacent Eulerian velocities are uniform. The worst case from the viewpoint of stability occurs when all Lagrangian points coincide with the Eulerian points. In that case, the maximum forcing at the Eulerian point f_2 is acquired:

$$\begin{aligned} f_2(t) &= \phi(-1)\phi(0)F(t)+\phi(0)\phi(0)F(t)+\phi(1)\phi(0)F(t) \\ &= \phi(0)\{\phi(-1)F(t)+\phi(0)F(t)+\phi(1)F(t)\} \\ &= \phi(0)F(t) \end{aligned} \tag{23}$$

Similarly, we can obtain the maximum momentum forcing for other cases; the results are listed in Table I. In its general form, the maximum momentum forcing can be expressed as

$$f_{\max}(t)=C_{\max}F(t)=C_{\max}\left(\alpha\int_0^t u(t')\,\mathrm{d}t'+\beta u(t)\right) \tag{24}$$

DOI: 10.1002/fld

Table I. Maximum Eulerian forcing for 1-D, 2-D, and 3-D cases.

	1-D	2-D	3-D
Point	$\phi(0)F(t)$	$(\phi(0))^2F(t)$	$(\phi(0))^3F(t)$
Line	$F(t)$	$\phi(0)F(t)$	$(\phi(0))^2F(t)$
Area	—	$F(t)$	$\phi(0)F(t)$

where $C_{\max}$ is the coefficient of the maximum Eulerian forcing for each case. Here, we consider a stationary problem for simplicity, i.e. $U_d(t)=0$ in Equation (4). Hence, the N–S equations can be expressed as

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\Delta t}+N\mathbf{u}^{n+1}=-Gp^{n+1/2}+\frac{1}{2Re}(L\mathbf{u}^{n+1}+L\mathbf{u}^n)+C_{\max}\left(\alpha\sum_{i=0}^{n}\mathbf{u}^i\Delta t+\beta\mathbf{u}^n\right) \tag{25}$$

where $\sum_{i=0}^{n}\mathbf{u}^i\Delta t$ is an approximation to $\int_0^t\mathbf{u}(t')\,dt'$. In the present method, since $\alpha(\approx 1/\Delta t^2)$ and $\beta(\approx 1/\Delta t)$ are much larger than the other terms, we can simplify Equation (25) as

$$\mathbf{u}^{n+1}-\mathbf{u}^n\approx C_{\max}\Delta t\left(\alpha\sum_{i=0}^{n}\mathbf{u}^i\Delta t+\beta\mathbf{u}^n\right) \tag{26}$$

To obtain the recurrence formula for the stability analysis, the equation at the previous time step is subtracted from the equation at the present time step, resulting in

$$\mathbf{u}^{n+1}-2\mathbf{u}^n+\mathbf{u}^{n-1}=\alpha'\mathbf{u}^n+\beta'(\mathbf{u}^n-\mathbf{u}^{n-1}) \tag{27}$$

where $\alpha'=C_{\max}\alpha\Delta t^2$ and $\beta'=C_{\max}\beta\Delta t$. Substitution of $\mathbf{u}^n=\mathbf{u}^0r^n$ $(r\equiv\mathbf{u}^{i+1}/\mathbf{u}^i, i=0,1,2,\ldots,n)$ into Equation (27) leads to the equation for r

$$r^2-(2+\alpha'+\beta')r+1+\beta'=0 \tag{28}$$

Thus, the stability region $\text{abs}(r)\leqslant 1$ is

$$-\alpha'-2\beta'\leqslant 4\Rightarrow -C_{\max}\alpha\Delta t^2-2C_{\max}\beta\Delta t\leqslant 4 \tag{29}$$

From the above equation, we can see that the stability region is mainly affected by the coefficient $C_{\max}$, which depends on the type of Lagrangian domain and Eulerian domain, as shown in Table I. For example, for the case in which the Lagrangian domain is a line in a 2-D flow, $C_{\max}=\phi(0)$, which is smaller than $C_{\max}=1$ for the case where the Lagrangian domain is a plane in a 2-D flow. In other words, without applying the Lagrangian forcing to the interior of the solid object (Figure 1), the stability region relaxes for a given feedback forcing gain (α,β), as shown in Equation (29). The stability region can also be affected by the time-advancing scheme. Although only the forward Euler scheme is considered here, the stability regimes for various time-advancing schemes can be obtained in a similar way [14].

DOI: 10.1002/fld

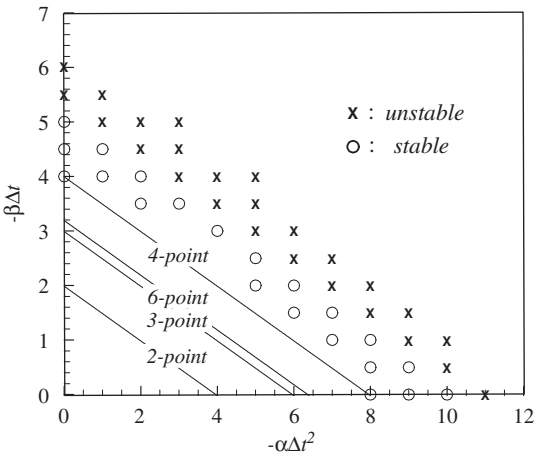

Figure 5. Stability regimes in 2-D flow for several types of delta functions. The dot/cross denotes stable/unstable cases in the simulation of a cylinder moving in a 2-D flow.

Stability regimes of different types of delta functions are displayed in Figure 5 for the case in which the Lagrangian domain is a line immersed in a 2-D flow. The flow is stable in the region below the line and unstable above the line. Stability regimes are wider for the smaller value of $\phi(0)$ in the regularized delta functions. Compared with the 2-point regularized delta function, the stability region of the 4-point regularized delta function is twice as wide in each direction $(-\alpha\Delta t^2, -\beta\Delta t)$. To validate the numerical analysis, simulations of a moving cylinder in a 2-D flow using the 4-point regularized delta function were carried out for different $-\alpha\Delta t^2$'s and $-\beta\Delta t$'s. Stable and unstable cases are denoted by circles and crosses, respectively, in Figure 5. The analytical solution is in good agreement with the numerical results obtained using the present IB method. It is also confirmed that the assumption that the adjacent Eulerian velocities are uniform is reasonable for the above stability analysis. In fact, the difference between adjacent velocities is negligible due to the small mesh width. Under this assumption, the stability analysis process is significantly simplified compared with Lee [14]. Since the worst case is always considered in the theoretical analysis, the actual stability regions of the numerical results will be slightly wider than those of the analytical predictions, as shown in Figure 5.

4. COMPARISON BETWEEN PESKIN'S IB METHOD AND THE VIRTUAL BOUNDARY METHOD

4.1. Feedback forcing scheme

In Peskin's IB method for solving the flow past a circular cylinder, the boundary is considered to be elastic but extremely stiff, as shown in Figure 6(a). A feedback forcing is formulated as follows:

$$\mathbf{F}(s,t) = \kappa(\mathbf{X}_{\mathrm{bs}}(s,t) - \mathbf{X}_{\mathrm{ib}}(s,t)) \tag{30}$$

DOI: 10.1002/fld

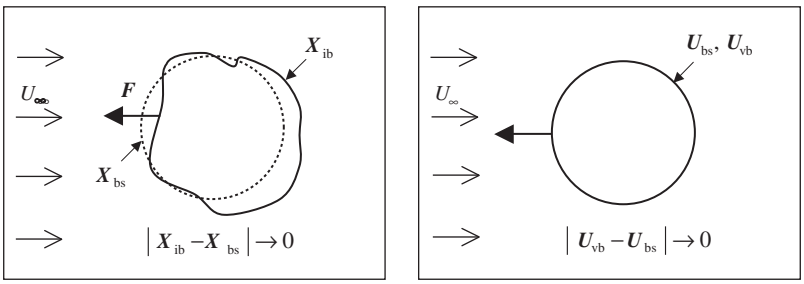

Figure 6. Schematic diagram for comparison of the feedback forcing scheme: (a) Peskin's IB method and (b) the virtual boundary method.

where $\mathbf{X}_{ib}(s,t)$ and $\mathbf{X}_{bs}(s,t)$ denote the IB and its equilibrium location, and the stiffness coefficient $\kappa \gg 1$. The IB $\mathbf{X}_{ib}(s,t)$ of the next time step moves with the local fluid velocity.

On the other hand,in the virtual boundary method the body surface $\mathbf{X}_{bs}(s,t)$ is considered as a virtual boundary where the feedback forcing is applied on the fluid so that the fluid will be at rest on the surface (see Figure 6(b)). Thus, the forcing term is expressed as

$$\mathbf{F}(s,t) = \alpha \int_0^t (\mathbf{U}_{vb}(s,t') - \mathbf{U}_{bs}(s,t'))\,dt' + \beta(\mathbf{U}_{vb}(s,t) - \mathbf{U}_{bs}(s,t)) \tag{31}$$

where $\mathbf{U}_{vb}(s,t)$ denotes the velocity at the virtual boundary interpolated from the fluid velocity field, and $\mathbf{U}_{bs}(s,t)$ denotes the velocity at the surface of the moving body. To apply the feedback forcing defined in Equation (31), α and β should be large negative constants to enforce the no-slip condition. The virtual boundary location $\mathbf{X}_{vb}(s,t)$ is considered as the body surface $\mathbf{X}_{bs}(s,t)$.

Comparing Figure 6(a) and (b), we can see that Peskin's IB method uses a boundary whose position moves according to the local flow velocity, whereas the virtual boundary method uses a boundary that always coincides with the body surface. Accordingly, in Peskin's IB method, position compensation must be implemented to correct the position and the velocity of each point on the boundary. In the virtual boundary method, however, velocity compensation is implemented to correct only the velocity of each point on the boundary, because the boundary remains on the body surface. Such differences between the two methods require that $|\kappa|$ in Peskin's IB method be much larger than $|\alpha|$ in the virtual boundary method to ensure accurate body surface conditions. This suggests that the time-step restriction will be stricter for Peskin's IB method than for the virtual boundary method.

4.2. Transfer of quantities between Lagrangian and Eulerian locations

In both methods, a transfer process between Lagrangian and Eulerian locations is necessary since the virtual boundary points do not always coincide with the computational meshes in the discrete grid system. The transfer process of Peskin's IB method is explained in Equations (11) and (12), while the transfer process is a little different in the virtual boundary method. First, the velocity at the virtual boundary point ($\mathbf{X}_{vb}$) is obtained through a linear interpolation using the velocity at

DOI: 10.1002/fld

the nearby mesh points ($\mathbf{x}_i$):

$$\mathbf{U}(\mathbf{X}_{\mathrm{vb}}^l)=\sum \mathbf{u}(\mathbf{x}_i)\delta_{\mathrm{vb}}(\mathbf{x}_i-\mathbf{X}_{\mathrm{vb}}^l),\quad 1\leqslant l\leqslant N_{\mathrm{L}} \tag{32}$$

$$\delta_{\mathrm{vb}}(\mathbf{x})=\phi\left(\frac{x_1}{h}\right)\phi\left(\frac{x_2}{h}\right)\phi\left(\frac{x_3}{h}\right),\quad \phi(\mathbf{r})=\begin{cases}1-|\mathbf{r}|, & |\mathbf{r}|\leqslant 1\\ 0 & \text{otherwise}\end{cases} \tag{33}$$

where $\phi(\mathbf{r})$ is the same as that of the 2-point regularized delta function, and N_{L} is the number of points distributed densely on the virtual boundary. After the feedback forcing is obtained, it is extrapolated back to the nearby mesh points:

$$\mathbf{f}(\mathbf{x}_i)=\frac{1}{N_{\mathrm{V}}}\sum_{i=1}^{N_{\mathrm{V}}}\mathbf{F}(\mathbf{X}_{\mathrm{vb}}^l)\delta_{\mathrm{vb}}(\mathbf{X}_{\mathrm{vb}}^l-\mathbf{x}_i),\quad 1\leqslant l\leqslant N_{\mathrm{L}} \tag{34}$$

where N_{V} is the number of virtual boundary points $\mathbf{X}_{\mathrm{vb}}^l$ that influence the mesh point $\mathbf{x}_i$.

The velocity approximation is almost the same in the two methods, but the forcing approximation is not. There are two significant differences in the forcing approximation. One difference is the number of Lagrangian points. The virtual boundary method uses many points, leading to high computational overheads, and the rule for determining the number of points is ambiguous. By comparison, Peskin's IB method uses fewer points; in this method, it is recommended to use the number of Lagrangian points that makes the volume of each forcing point equivalent to a finite volume of the Eulerian grid ($\Delta V_l\approx h^m$, where m is the number of space dimensions) [15]. The second major difference is the conservation of quantities between the Lagrangian and Eulerian domains. Peskin's IB method uses the class of regularized delta function as kernel in the transfer steps between Lagrangian and Eulerian locations. Accordingly, the total amount of quantities is not changed by the transfer step [10, 15]. The virtual boundary method, by contrast, uses an average forcing and hence changes the total amount of quantities between the transfer steps. Figure 7 displays 2-D cases for the two methods. We assume that all values of the forcing in the Lagrangian domain are F. Figure 7(a) illustrates Peskin's IB method using a 2-point regularized delta function. The Eulerian forcing f can be obtained using

$$\begin{aligned}f&=\phi(1-r_x)\phi(1-r_y)F+\phi(1-r_x)\phi(r_y)F+\phi(r_x)\phi(1-r_y)F+\phi(r_x)\phi(r_y)F\\&=\phi(1-r_x)\{\phi(1-r_y)F+\phi(r_y)F\}+\phi(r_x)\{\phi(1-r_y)F+\phi(r_y)F\}\\&=\phi(1-r_x)F+\phi(1-r_x)F,\quad(\phi(1-r)+\phi(r)=1)\\&=F\end{aligned} \tag{35}$$

Since Peskin's IB method conserves the forcing between the Lagrangian and Eulerian domains, the drag and lift forces of a stationary cylinder can be obtained by integrating the Lagrangian forcings on the boundary ∂S [9]:

$$F_{\mathrm{D}}=-\int_{\partial S}f_x\,\mathrm{d}\mathbf{x}=-\int_{\partial S}F_x\,\mathrm{d}\mathbf{s},\quad F_L=-\int_{\partial S}f_y\,\mathrm{d}\mathbf{x}=-\int_{\partial S}F_y\,\mathrm{d}\mathbf{s} \tag{36}$$

DOI: 10.1002/fld

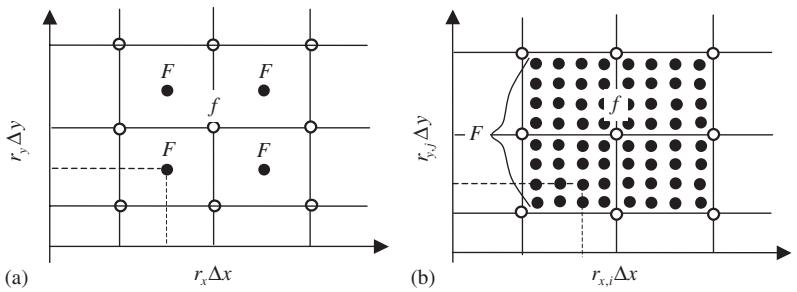

Figure 7. Schematic diagram of the forcing transfer process for area Lagrangian domain in the 2-D case: (a) Peskin's IB method (b) virtual boundary method; •, Lagrangian point; ∘, Eulerian point.

where F_D and F_L are the drag and lift forces, respectively. Figure 7(b) shows that the Eulerian forcing f is not equal to F by the virtual boundary method, i.e.

$$f = \lim_{N_x,N_y\to\infty} \frac{1}{N_x}\frac{1}{N_y}\sum_{i=1}^{N_x}\sum_{j=1}^{N_y} F(1-|r_{x,i}|)(1-|r_{y,j}|)$$

$$= \lim_{N_x\to\infty} \frac{F}{2}\sum_{i=1}^{N_x}(1-|r_{x,i}|)\Delta r_x \lim_{N_y\to\infty}\frac{1}{2}\sum_{j=1}^{N_y}(1-|r_{y,j}|)\Delta r_y$$

$$= \frac{F}{2}\int_{-1}^{1}(1-|r_x|)\,\mathrm{d}r_x \frac{1}{2}\int_{-1}^{1}(1-|r_y|)\,\mathrm{d}r_y = \frac{F}{4} \tag{37}$$

where $\Delta r_x = 2/N_x, \Delta r_y = 2/N_y$, and N_x and N_y are the numbers of Lagrangian points in the x- and y-directions, respectively. The virtual boundary method does not conserve the forcing in the transfer step.

5. RESULTS AND DISCUSSION

5.1. Stationary cylinder

The flow past a stationary cylinder in a free-stream was simulated at two different Reynolds numbers (100 and 185) based on the free-stream velocity u_∞ and the cylinder diameter. The first case was simulated to compare the present method with other methods such as Peskin's IB method. The second case was examined to investigate the effects of the feedback forcing gains (α and β) and types of delta functions.

5.1.1. Stationary cylinder in a free-stream at $Re = 100$. We used a computational domain of $0 \leqslant x, y \leqslant 8$ and a cylinder with diameter $d = 0.30$ whose center is located at $(1.85, 4.0)$. A Dirichlet boundary condition ($u/u_\infty = 1, v = 0$) was used at the inflow and far-field boundaries, and a convective boundary condition ($\partial u_i/\partial t + c\partial u_i/\partial x = 0$, where c is the space-averaged streamwise

DOI: 10.1002/fld

Table II. Comparison of drag coefficient, lift coefficient, and Strouhal number with those obtained in previous studies.

	h	κ or $-\alpha$	Δt	$\bar{C}_D$	C'_L	St	CFL
Case 1	$\frac{1}{64}$	4.8×10^4	1.2×10^{-2}	1.44	0.35	0.168	1.35
Case 2	$\frac{1}{64}$	4.8×10^4	6.0×10^{-3}	1.44	0.35	0.168	0.7
Case 3	$\frac{1}{64}$	4.8×10^4	6.0×10^{-3}	1.37	0.34	0.163	0.7
Lai and Peskin [9]	$\frac{1}{64}$	4.8×10^4	1.8×10^{-3}	1.52	0.29	0.155	—
Lai and Peskin [9]	$\frac{1}{128}$	9.6×10^4	9.0×10^{-4}	1.45	0.33	0.165	—
Uhlmann [15]	$\frac{1}{128}$	—	3.0×10^{-3}	1.50	0.35	0.172	—
Kim *et al.* [3]	—	—	—	1.33	0.32	0.165	—
Linnick and Fasel [18]	—	—	—	1.34	0.33	0.166	—
Liu *et al.* [19]	—	—	—	1.35	0.34	0.165	—
Huang and Sung [4]	—	—	—	1.36	0.33	0.167	—

velocity) was used at the outflow boundary. Table II shows the drag and lift coefficients, C_D and C_L, obtained using the proposed method, as well as the Strouhal number defined from the oscillation frequency of the lift force. The drag and lift forces were obtained by integrating all the momentum forcing applied on the boundary, as shown in Equation (36). Parameters such as the mesh width h, time step Δt, and feedback forcing gain α were selected to match the conditions of Lai and Peskin [9] and a 4-point regularized delta function was employed [9]. To compare the present method with Peskin's IB method, we used feedback forcing gains of $\alpha=-4.8\times10^4$ and $\beta=0$. Table II indicates that the value of $\alpha=-4.8\times10^4$ used in the present method is large enough to obtain reliable results. By contrast, the results of Lai and Peskin [9] using $\kappa=4.8\times10^4$ deviate somewhat from the other results, especially those obtained in the same study using $\kappa=9.6\times10^4$. These findings are consistent with previous reports showing that compared with the value of $-\alpha$ in the virtual boundary method, a larger value of the stiffness coefficient κ in Peskin's IB method is required to ensure accurate results for a rigid boundary problem [1]. Since we tested the stability region of feedback forcing gains (α,β) with the 4-point regularized delta function (see Figure 5), we used a computational time step of $1.2\times10^{-2}(-\alpha\Delta t^2=6.912)$ to be consistent with $-\alpha\Delta t^2<8$. As a consequence, the present results are in good agreement with those of Lai and Peskin [9], even though the computational time step of the present method ($\Delta t=1.2\times10^{-2}$) is about an order of magnitude larger than that of Lai and Peskin ($\Delta t=9.6\times10^{-4}$). The maximum Courant–Friedrichs–Lewy (CFL) number in the present simulations exceeded 1 due to the adoption of the feedback forcing scheme and the optimization of parameters by stability analysis. Uhlmann [15] also simulated the same problem with the same domain size. Uhlmann's results showed some deviation even though the computational time step and mesh size are smaller than those used in the present work. When the domain size is enlarged to $0\leqslant x, y\leqslant16$ (case 3 in Table II), the present results agree better with those of other studies [3, 18–20].

Table III shows the influences of $\Delta X/h$ on drag and lift coefficients and Strouhal number using the 4-point delta function at $-\alpha\Delta t^2=3.9$ and $-\beta\Delta t=1.9$. The mesh width h was $\frac{1}{128}$, and the computational time step was $\Delta t=0.006$, giving a maximum CFL number of approximately 1.4. Four cases were simulated using evenly distributed Lagrangian points and the last one was tested using non-uniformly distributed Lagrangian points. In the non-uniform case, $\Delta X/h$ is increased

DOI: 10.1002/fld

Table III. Influence of the ratio of Lagrangian point distance to Eulerian grid width on drag and lift coefficients and Strouhal number using the 4-point delta function.

$\Delta X/h$	N_L	$\bar{C}_D$	C'_L	St
2	60	1.45	0.36	0.171
1	120	1.45	0.36	0.171
0.5	241	1.45	0.35	0.171
0.25	482	1.45	0.35	0.171
0.25−2	142	1.45	0.36	0.171

from 0.25 to 2 by the ratio of approximately 1.015. The values resulting from the variations of $\Delta X/h$ are very similar. Actually, the differences of $\bar{C}_D, C'_L$, and St are within 3%, indicating that the number of Lagrangian points has a negligible effect on the results. This also implies that the present method does not require even distribution of Lagrangian points. In this study, however, the Lagrangian points are selected such that a discrete volume is the same as the finite volume of the Eulerian grid, i.e. $\Delta V_l \approx h^m$, for simplicity and computational efficiency of the solution procedure.

5.1.2. Stationary cylinder in a free-stream at $Re=185$. Next we consider the case of a large computational domain $-50d \leqslant x, y \leqslant 50d$, where d denotes the cylinder diameter. The number of grid points in the streamwise (x) and transverse (y) directions was 352×192, respectively. Thirty grid points in each direction were uniformly distributed inside the cylinder and the remaining grid points were stretched outside the cylinder. The cylinder surface was made up of 94 Lagrangian points, and the Reynolds number $Re=u_\infty d/\nu$ was set at 185. The computational time step was $\Delta t=0.01$, leading to a maximum CFL number of approximately 0.6. The boundary conditions at the inflow, far-field and outflow boundaries were the same as those of the former case at $Re=100$.

Four different forcing gains were simulated using a 3-point regularized delta function. For a quantitative comparison, an l_2-norm error is defined as

$$l_2\text{-norm error} \equiv \sqrt{\frac{1}{N_L}\sum_{k=1}^{N_L}(\mathbf{U}(\mathbf{X}_k,t))^2} \tag{38}$$

where N_L is the number of Lagrangian points and $\mathbf{U}(\mathbf{X}_k,t)$ is the velocity at the kth Lagrangian point. l_2-Norm errors of the four cases, displayed in Figure 8, show that the error converges to a smaller value for the larger value of $-\alpha\Delta t^2$, and the error decays more rapidly for the larger value of $-\beta\Delta t$. In a stationary boundary problem, $-\beta\Delta t$ influences only the initial behavior of the error. Two cases, using the same $-\alpha\Delta t^2$ value with different $-\beta\Delta t$ values, show the same level of the error, suggesting that $-\alpha\Delta t^2$ is a more critical parameter for enforcing the no-slip condition [14]. However, $-\beta\Delta t$ is also important as it influences the total computation time. For $-\beta\Delta t=0$, a long computation time is required to converge to a small value. The present findings thus indicate that both $-\alpha\Delta t^2$ and $-\beta\Delta t$ should be considered when optimizing the accuracy and efficiency of the computation.

Four different types of the regularized delta functions were simulated with the largest forcing gains in stability regimes (Figure 9). As expected, the error of the 4-point delta function ($-\alpha\Delta t^2=4, -\beta\Delta t=2$) is smaller than those of the 2-point ($-\alpha\Delta t^2=2, -\beta\Delta t=1$), 3-point

DOI: 10.1002/fld

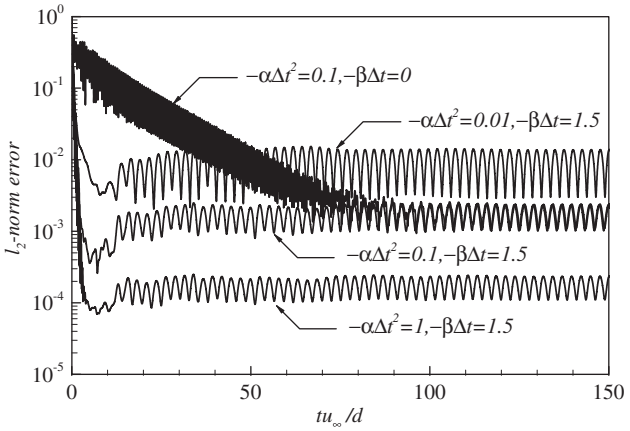

Figure 8. l_2-Norm error of the virtual surface velocity in the streamwise direction normalized by the free-stream velocity u_∞ for different forcing gains.

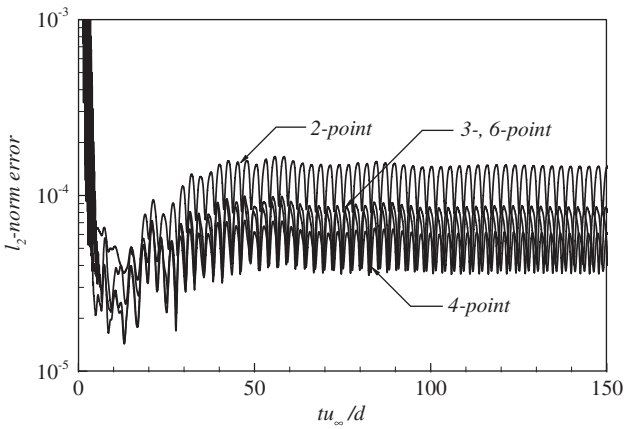

Figure 9. l_2-Norm error of the virtual surface velocity in the streamwise direction normalized by the free-stream velocity u_∞ for several types of delta functions with the largest forcing gains in stability regimes.

$(-\alpha\Delta t^2=3, -\beta\Delta t=1.5)$, and 6-point delta functions $(-\alpha\Delta t^2=3.2, -\beta\Delta t=1.6)$. Table IV also shows that the result obtained using the 4-point delta function is in better agreement with previous results [20, 21] than those of the 2-, 3-, and 6-point delta functions.

5.2. Inline oscillation of a circular cylinder

We additionally simulated a periodic inline oscillation of a circular cylinder in a fluid at rest. The Reynolds number is defined as $Re=u_m d/\nu$ based on the maximum velocity u_m and the cylinder diameter d. The Keulegan–Carpenter number is defined as $KC=u_m/fd$ based on the frequency of the oscillation. The parameter set of the present simulation was $Re=100$ and $KC=5$, according to

DOI: 10.1002/fld

Table IV. Comparison of drag coefficients, lift coefficients, and Strouhal numbers obtained using several types of delta functions with the largest forcing gains in stability regions.

	$\bar{C}_{\mathrm{D}}$	$C_{\mathrm{L\,rms}}$	St
2-Point	1.343	0.454	0.190
3-Point	1.312	0.436	0.190
4-Point	1.296	0.430	0.190
6-Point	1.308	0.437	0.190
Experimental results [21]	1.28	—	0.190
Lu and Dalton [20]	1.31	0.422	0.195
Guilmineau and Queutey [21]	1.287	0.443	0.195

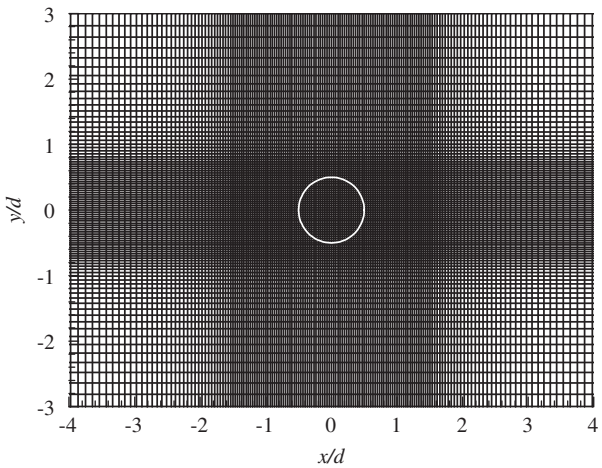

Figure 10. Grid distribution near the cylinder in the simulation of the flow around a cylinder undergoing inline oscillation.

the experimental and numerical results of Dütsch *et al.* [22]. We set the cylinder in time-periodic motion:

$$x_{\mathrm{c}}(t) = -A_{\mathrm{m}} \sin(2\pi f t) \tag{39}$$

where $x_{\mathrm{c}}(t)$ is the position of the cylinder center at time t and A_{m} is the amplitude of the oscillation. The computational domain was $-50d \leqslant x, y \leqslant 50d$, and the number of grid points in the oscillatory (x) and transverse (y) directions was 416×282, respectively. Sixty grid points in each direction were uniformly distributed inside the cylinder and the remaining grid points were stretched outside the cylinder, as shown Figure 10. Because the cylinder oscillates in the x-direction, the uniformly distributed region had a rectangular form with longer length in the x-direction. Neumann boundary conditions were used at all four far-field boundaries. The number of Lagrangian points N_{L} around the cylinder surface was set to 188 and the computational time step was $\Delta t = T/720$ based on the period of the oscillation, leading to a maximum CFL number of approximately 0.6. The 4-point regularized delta function was selected and the feedback forcing gain was chosen as $-\alpha \Delta t^2 = 3.9$

DOI: 10.1002/fld

and $-\beta\Delta t = 1.9$, which is one of the largest cases in the stability region of the 4-point regularized delta function.

In a moving boundary problem, the drag force is obtained by summing over the momentum forcing on the boundary and other terms related to the acceleration of the moving boundary [15], as follows:

$$F_{\mathrm{D}} = -\int_{\partial S} f_x \,\mathrm{d}\mathbf{x} + \frac{\mathrm{d}}{\mathrm{d}t}\int_S u \,\mathrm{d}\mathbf{x} \tag{40}$$

where f_x is the x-component of the force density $\mathbf{f}$. The present case is the rigid-body motion on the volume S:

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_S u \,\mathrm{d}\mathbf{x} = V\frac{\mathrm{d}u_{\mathrm{c}}}{\mathrm{d}t} \tag{41}$$

where V is the volume of the rigid body and u_{c} is the velocity at the cylinder center.

In Figure 11, the time history of the drag coefficient in the oscillatory direction is in excellent agreement with that of Dütsch *et al.* [22]. Figure 12 shows the isolines of pressure and vorticity for different phase angles of the oscillating cylinder motion. During the oscillation of the cylinder, the flow is characterized by two counter-rotating vortices. These results are very similar to those of previous studies [21, 22]. To further compare the present findings with the results of Dütsch *et al.* [22], we examined the velocity profiles at four locations (Figure 13). The results agree very well with the numerical and experimental results of Dütsch *et al.* [22], indicating that the present method accurately describes the velocity field.

5.3. Transverse oscillation of a circular cylinder

Next, we simulated a periodic transverse oscillation of a circular cylinder in a free-stream:

$$y_{\mathrm{c}}(t) = A_{\mathrm{m}}\cos(2\pi f_{\mathrm{e}} t) \tag{42}$$

where y_{c} is the position of the cylinder center, and A_{m} and f_{e} are the amplitude and frequency of the oscillation, respectively. The detailed conditions were $Re = 185$ and $A_{\mathrm{m}}/d = 0.2$. f_{o} is the

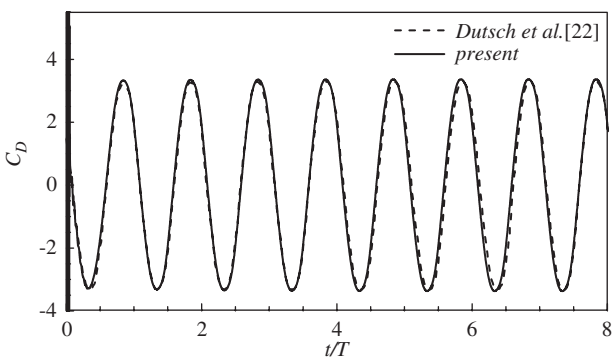

Figure 11. Time history of drag coefficient at $Re = 100$ and $KC = 5$.

DOI: 10.1002/fld

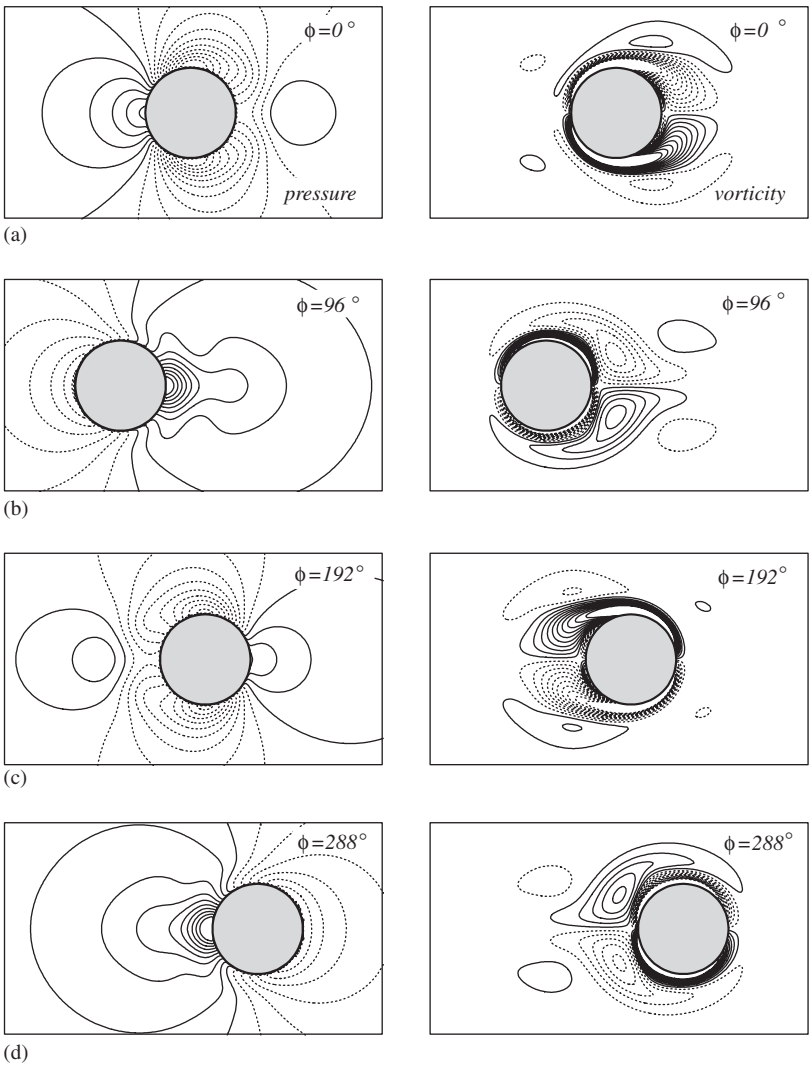

Figure 12. Pressure and vorticity isolines (negative values dashed) at $Re=100$ and $KC=5$ at different phase positions ($\phi=2\pi ft$).

natural shedding frequency from the stationary cylinder. The computational domain was $0\leqslant x, y\leqslant 8$ and 1024×1024 grid points were uniformly distributed in the oscillatory (x) and transverse (y) directions, respectively. The cylinder diameter was set to $d=0.30$, and its center was located at $(1.85, 4.0)$. The cylinder surface was composed by 120 Lagrangian points. A Dirichlet boundary condition was used at the inflow and far-field boundaries, and a convective boundary condition was used at the outflow boundary.

The behavior of the l_2-norm error of the streamwise virtual surface velocity is shown in Figure 14 for three different forcing gains with the 3-point regularized delta function. The error converges

DOI: 10.1002/fld

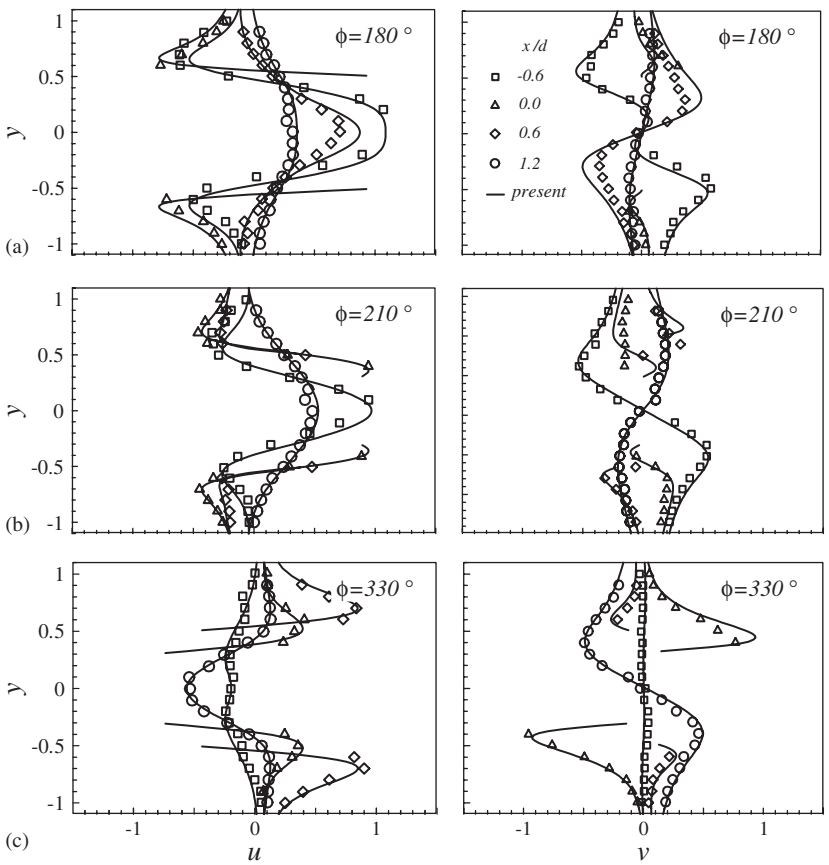

Figure 13. Comparison of the velocity components between the present computation and the experimental results of Dütsch *et al.* [22] at four cross-sections for different phase positions ($\phi = 2\pi ft$).

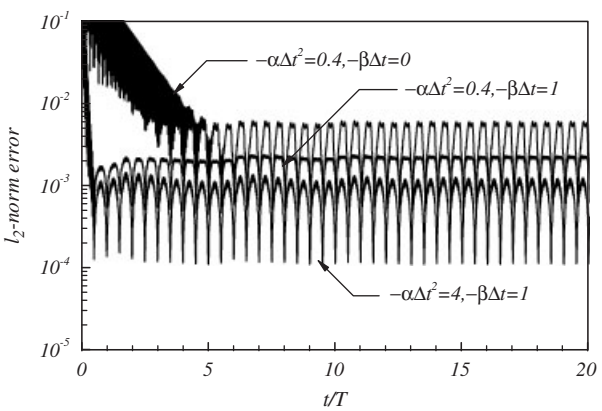

Figure 14. l_2-Norm error of the virtual surface velocity in the streamwise direction normalized by the free-stream velocity u_∞ for three different forcing gains.

DOI: 10.1002/fld

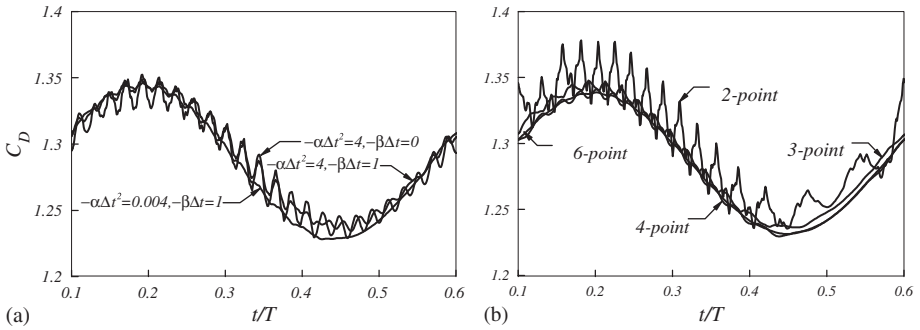

Figure 15. Time history of drag coefficient (a) for different forcing gains using the 3-point delta function and (b) for different types of delta functions with $-\alpha\Delta t^2=0.4$ and $-\beta\Delta t=1$.

to a smaller value for larger $-\alpha\Delta t^2$, as observed above for the stationary problem, and the error decays rapidly for larger $-\beta\Delta t$. Since the initial decays of the error are important in compensation of the boundary condition in moving boundary problems, the error also converges to a smaller value for larger $-\beta\Delta t$. Accordingly, $-\alpha\Delta t^2$ and $-\beta\Delta t$ should be as large as possible to decrease the error.

Figure 15(a) shows the time histories of the drag coefficient for three different forcing gains, determined using the 3-point delta function. Non-growing oscillations become smaller as $-\beta\Delta t$ is increased, but are increased for larger $-\alpha\Delta t^2$. The influence of $-\beta\Delta t$ is greater than that of $-\alpha\Delta t^2$, in that when $-\beta\Delta t$ is large the non-growing oscillations are not significant regardless of the value of $-\alpha\Delta t^2$. The time history of the drag coefficient is illustrated in Figure 15(b) for four types of delta functions with the same forcing gains. As the number of points in the regularized delta function increases, the non-growing oscillations decrease. This suggests that the 4-point regularized delta function with large forcing gains yields better results.

To investigate the effect of delta functions on CPU time, we performed the first 100 time steps of the simulation using 100, 1000, and 10 000 Lagrangian points for different types of delta functions. These computations were performed on Intel 3.4 GHz Pentium 4 Xeon processor. In Table V, CPU time increases for more number of points in the regularized delta function, and this tendency is evidently shown in 10 000 Lagrangian points. In this study, all cases were simulated in 2-D using the number of Lagrangian points of approximately 100; hence, the differences of CPU time among different delta functions are relatively small. In 3-D cases, however, a k-point delta function is supported by k^3 grid cells and much more Lagrangian points are used. Thus, the regularized delta function can be an important factor on computational costs in 3-D cases.

For a further comparison of the present results with previous ones, we performed simulations using a large computational domain of dimensions $-50d \leqslant x, y \leqslant 50d$, with a grid size of 416×282. Sixty grid points were uniformly distributed in each direction inside the cylinder and the grids were stretched outside the cylinder. The cylinder surface was made up of 188 Lagrangian points. Several cases were simulated for different oscillating frequencies in the range $0.8 \leqslant f_e/f_o \leqslant 1.2$. The computational time step was chosen as $\Delta t = T/720$ based on the period of the oscillation, leading to a maximum CFL number in the range of 0.7–1.1. The 4-point regularized delta function was employed and the feedback forcing gains were $-\alpha\Delta t^2=3.9$ and $-\beta\Delta t=1.9$. Time histories of the drag and lift coefficients for $0.8 \leqslant f_e/f_o \leqslant 1.2$ are illustrated in Figure 16. The drag and lift

DOI: 10.1002/fld

Table V. Effect of different types of delta functions and the numbers of Lagrangian points on CPU time (in seconds) required for the first 100 time steps of the simulation.

	$N_L = 100$	$N_L = 1000$	$N_L = 10000$
2-Point	187.25	188.35	193.34
3-Point	187.58	188.65	197.53
4-Point	187.66	188.85	198.08
6-Point	188.20	190.55	213.24

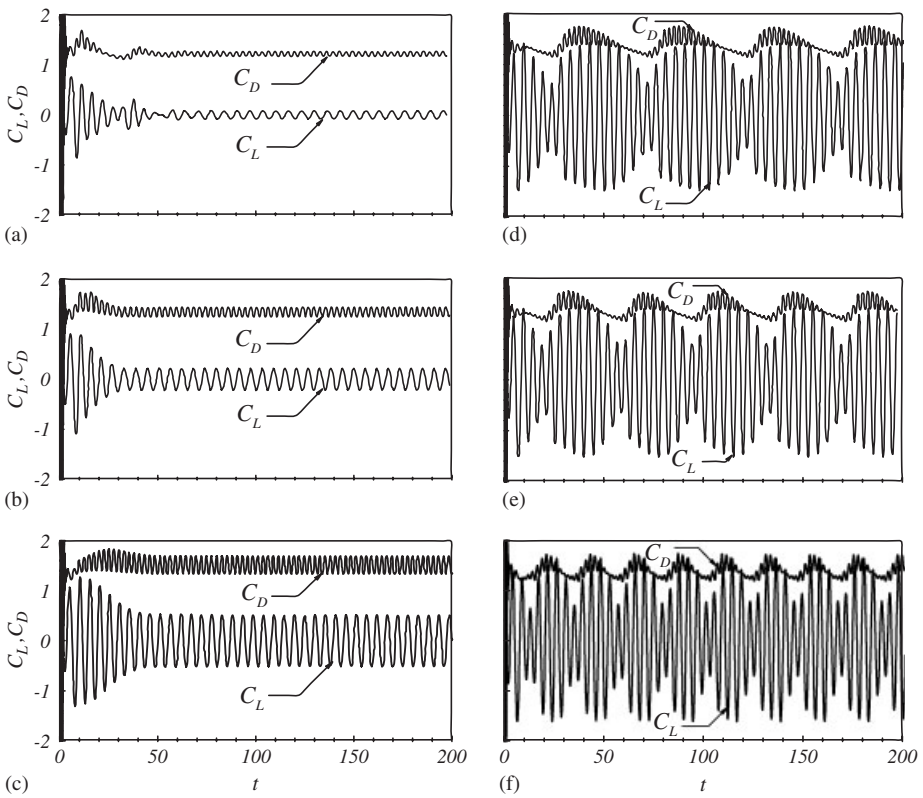

Figure 16. Time history of drag and lift coefficients for $Re = 185$ and $A_e/D = 0.2$ for f_e/f_o values of (a) 0.80, (b) 0.90, (c) 1.00, (d) 1.10, (e) 1.12, and (f) 1.20.

coefficients behave regularly once vortex shedding is established. For $f_e/f_o > 1.0$, both the drag and lift coefficients exhibit beat phenomena, with the beat frequency increasing with excitation frequency. These results are in good agreement with those of Guilmineau and Queutey [21]. The variations of the mean drag, rms drag, and lift fluctuation coefficients as a function of f_e/f_o are presented in Figure 17(a), and the phase angles between the lift coefficient and the vertical position of the cylinder are shown in Figure 17(b). The mean drag coefficient increases with

DOI: 10.1002/fld

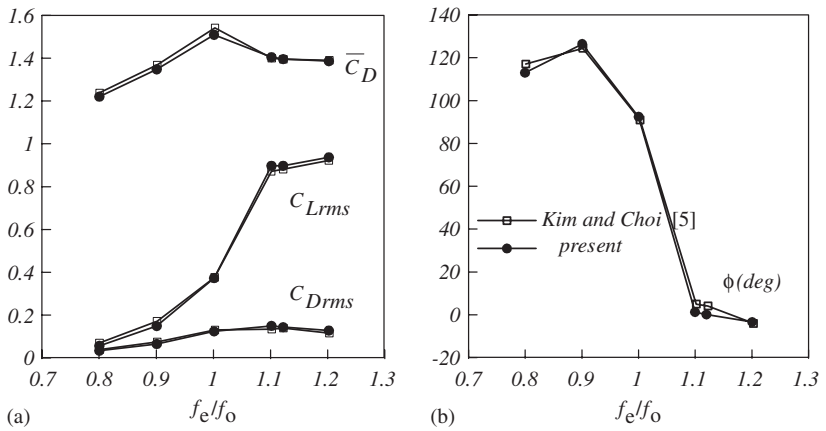

Figure 17. Variations of the force coefficients and phase angle with respect to f_e/f_o: (a) mean drag coefficient and rms drag and lift fluctuation coefficients and (b) phase angle between C_L and the vertical position of the cylinder.

increasing f_e/f_o up to a peak at $f_e/f_o = 1.0$, after which it decreases with further increases of f_e/f_o. The rms lift fluctuation coefficients and the phase angles show a change at $f_e/f_o = 1.1$, when vortex switching occurs [21]. The present results agree well with those of Kim and Choi [5]. The previous methods [5, 21] were based on a non-inertial reference frame that is fixed to the body with an arbitrary motion, while in the present method we use a fixed reference frame and the body is allowed to move across the grid line. Thus, the present method can be extended in a more straightforward manner to multi-body problems or flexible body problems, e.g. the studies of Huang *et al.* [16].

6. CONCLUSIONS

In this study, we compared the virtual boundary method and Peskin's IB method, with a focus on two aspects: the feedback forcing scheme and the transfer process of quantities between Lagrangian and Eulerian locations. The feedback forcing scheme of the virtual boundary method made it possible to use a larger CFL number than was possible in Peskin's IB method, thereby ensuring accurate body surface conditions. However, the average forcing approach of the virtual boundary method was less effective than Peskin's delta function approach for transferring quantities between Lagrangian and Eulerian domains. We therefore combined the feedback forcing scheme of the virtual boundary method with Peskin's regularized delta function approach to improve performance. The resulting numerical method was implemented in a finite-difference and fractional-step context. We analyzed the stability regimes of the feedback forcing gains in the proposed method for several types of delta functions. The stability region of the 4-point regularized delta function was much wider than that of the 2-point delta function. The effects of regularized delta functions and feedback forcing gains (α, β) were also investigated. For the regularized delta function supported by more points, its non-growing oscillations became smaller. On the other hand, the l_2-norm error (a measure of the no-slip condition along the IB) converged to a smaller value for larger $-\alpha\Delta t^2$ and decayed faster

DOI: 10.1002/fld

for larger $-\beta\Delta t$. In the stationary boundary problem, $-\beta\Delta t$ influenced only the initial behavior of the error, whereas in the moving boundary problem the error also converged to a smaller value for larger $-\beta\Delta t$. On the basis of the stability analysis of the present method, we can recommend an optimum region of the feedback forcing gains that enables the use of a large CFL number and decreases the l_2-norm error and non-growing oscillations. The proposed method was applied to the flow past a stationary cylinder, inline oscillation of a cylinder in a quiescent fluid, and transverse oscillation of a circular cylinder in a free-stream at large maximum CFL numbers (0.6–1.4). The present findings are in excellent agreement with previous numerical and experimental results.

ACKNOWLEDGEMENTS

This work was supported by the Creative Research Initiatives of the Korea Science & Engineering Foundation and partially supported by Brain Korea 21 and the Basic Research Program (R01-2004-000-10521-0) of the Korea Science & Engineering Foundation.

REFERENCES

1. Mittal R, Iaccarino G. Immersed boundary methods. *Annual Review of Fluid Mechanics* 2005; **37**:239–261.
2. Fadlun EA, Verzicco R, Orlandi P, Mohd-Yusof J. Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations. *Journal of Computational Physics* 2000; **161**:35–60.
3. Kim J, Kim D, Choi H. An immersed boundary finite-volume method for simulations of flow in complex geometries. *Journal of Computational Physics* 2001; **171**:132–150.
4. Huang W-X, Sung HJ. Improvement of mass source/sink for an immersed boundary method. *International Journal for Numerical Methods in Fluids* 2007; **53**:1659–1671.
5. Kim D, Choi H. Immersed boundary method for flow around an arbitrarily moving body. *Journal of Computational Physics* 2006; **212**:662–680.
6. Yang J, Balaras E. An embedded-boundary formulation for large-eddy simulation of turbulent flows interacting with moving boundaries. *Journal of Computational Physics* 2006; **215**:12–40.
7. Peskin CS. Numerical analysis of blood flow in the heart. *Journal of Computational Physics* 1977; **25**:220–252.
8. Roma A, Peskin C, Berger M. An adaptive version of the immersed boundary method. *Journal of Computational Physics* 1999; **153**:509–534.
9. Lai M-C, Peskin CS. An immersed boundary method with formal second-order accuracy and reduced numerical viscosity. *Journal of Computational Physics* 2000; **160**:705–719.
10. Peskin CS. The immersed boundary method. *Acta Numerica* 2002; **11**:1–39.
11. Griffith BE, Hornung RD, McQueen DM, Peskin CS. An adaptive, formally second order accurate version of the immersed boundary method. *Journal of Computational Physics* 2007; **223**:10–49.
12. Goldstein D, Handler R, Sirovich L. Modeling a no-slip flow boundary with an external force field. *Journal of Computational Physics* 1993; **105**:354–366.
13. Saiki EM, Biringen S. Spatial simulation of a cylinder in uniform flow: application of a virtual boundary method. *Journal of Computational Physics* 1996; **123**:450–465.
14. Lee C. Stability characteristics of the virtual boundary method in three-dimensional applications. *Journal of Computational Physics* 2003; **184**:559–591.
15. Uhlmann M. An immersed boundary method with direct forcing for the simulation of particulate flows. *Journal of Computational Physics* 2005; **209**:448–476.
16. Huang W-X, Shin SJ, Sung HJ. Simulation of flexible filaments in a uniform flow by the immersed boundary method. *Journal of Computational Physics* 2007; **226**:2206–2228.
17. Kim K, Baek S-J, Sung HJ. An implicit velocity decoupling procedure for incompressible Navier–Stokes equations. *International Journal for Numerical Methods in Fluids* 2002; **38**:125–138.
18. Linnick MN, Fasel HF. A high-order immersed interface method for simulating unsteady incompressible flows on irregular domains. *Journal of Computational Physics* 2005; **204**:157–192.
19. Liu C, Zheng X, Sung CH. Preconditioned multigrid methods for unsteady incompressible flows. *Journal of Computational Physics* 1998; **139**:35–57.

DOI: 10.1002/fld

20. Lu X-Y, Dalton C. Calculation of the timing of vortex formation from an oscillating cylinder. *Journal of Fluids and Structures* 1996; **10**:527–541.
21. Guilmineau E, Queutey P. A numerical simulation of vortex shedding from an oscillating circular cylinder. *Journal of Fluids and Structures* 2002; **16**(6):773–794.
22. Dütsch H, Durst F, Becker S, Lienhart H. Low-Reynolds-number flow around an oscillating circular cylinder at low Keulegan–Carpenter numbers. *Journal of Fluid Mechanics* 1998; **360**:249–271.

DOI: 10.1002/fld