ACADEMIC PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 184 (2003) 559–591

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

Stability characteristics of the virtual boundary method in three-dimensional applications

Changhoon Lee *

Division of Mechanical Engineering, School of Mechanical and Electrical Engineering, Yonsei University, Seoul, South Korea

Received 23 January 2002; received in revised form 13 September 2002; accepted 16 October 2002

Abstract

The refined stability analysis of the virtual boundary method proposed by Goldstein et al. (1994) and modified by Saiki and Biringen (1996) is carried out for applications to three-dimensional turbulent flows in complex geometry. The precise stability boundaries in the forcing parameter space for various time-advancing schemes are provided under the assumption that the virtual boundary points are densely distributed. From these and the relevant investigation of frequency of the forced system, the optimum gains of the feedback forcing are suggested. Stability regimes of the Runge–Kutta schemes of various order are much wider than those of the Adams–Bashforth schemes. Specially, the third-order Runge–Kutta scheme allows the use of an order-one CFL number in the integration of the feedback forcing, rendering the method applicable to turbulent flows with complex boundaries. The three-dimensional turbulent flow caused by a surface-mounted box was simulated using a spectral method for evaluation, confirming the stability limit proposed by theoretical estimate. The method was then applied to simulations of the flow around an impulsively starting cylinder and of the rough-wall turbulent boundary layer flow.
© 2002 Elsevier Science B.V. All rights reserved.

Keywords: Virtual boundary method; Stability; Rough-wall turbulent flow

1. Introduction

An immersed boundary method was first proposed by Peskin [16] for the purpose of calculation of flows inside a moving heart. Independently, Goldstein et al. [5,6] developed a virtual boundary method that employs a feedback forcing to impose the no-slip condition at immersed boundaries present in the fluid domain. These two methods are virtually the same when applied to flow with immersed stationary walls. Saiki and Biringen [18] modified the virtual boundary technique and proposed the so-called ‘area-weighted’ virtual boundary method. There have been several applications of the virtual boundary method [4,6,7,11,19] and it has been reported that the method suffers from a very strict time-step restriction since

[*] Fax: +82-2-312-2159.
E-mail address: clee@yonsei.ac.kr (C. Lee).

the amplitude of the feedback forcing needs to be large for proper operation, resulting in a very stiff system. Very small CFL numbers (of order of 10^{-2}) were reported in various cases, making the method unattractive.

For a remedy of this problem, an alternative approach was proposed by Fadlun et al. [4]. Instead of using a feedback forcing with arbitrary gains, the proposed method uses a direct forcing at particular meshes such that in one time step, the method enforces the no-slip condition at the desired immersed boundary points locating between the computational meshes. They claimed that there is no parameter to optimize and the method does not suffer from any time-step restriction. Kim et al. [8] modified this method by adding a mass source to satisfy the local continuity near immersed boundary points. The performance of this method, however, heavily depends on the choice of immersed boundary points where the no-slip condition is imposed and the involved interpolation scheme. Applications of the method in three-dimensional cases become extremely complicated, thus most examples have two-dimensional or axisymmetric boundaries [4,8]. Furthermore, the stability characteristics of the method is not well understood. In some case, the method employing a linear interpolation fails to produce a stable solution [8].

For the sharp representation of immersed boundaries, a Cartesian grid method employing a second-order interpolation was proposed based on finite-volume approach [21,23]. Another approach by Leveque and Li [12] and Li and Lai [13] is to formulate the jump conditions across an immersed surface where the forcing is localized. Applications of these approaches, however, are mainly restricted to two-dimensional cases.

In the present study, we investigate the stability characteristics of the virtual boundary method that was proposed by Goldstein et al. [5] and later modified by Saiki and Biringen [18]. The reason we chose this approach is that an application of this method in three-dimensional cases is straightforward and stable performance is guaranteed as long as the time-step restriction is satisfied. For example, the stability limit for the time step for the second-order Adams–Bashforth scheme was already rigorously provided by Goldstein et al. [5], which is

$$\Delta t < \frac{-\beta - (\beta^2 - 2\alpha k)^{1/2}}{\alpha}, \tag{1}$$

where α and β are the negative gains of a virtual forcing added to the momentum equations,

$$F_i(t) = \alpha \int^t U_i(t')\,\mathrm{d}t' + \beta U_i(t), \tag{2}$$

where $U_i(t)$ is the fluid velocity at virtual boundary points. k in Eq. (1) is a problem-dependent constant of order one. However, in Goldstein et al. [5]'s applications, virtual boundary points coincide with the computational mesh points. In the later application to flow over a cylinder by Saiki and Biringen [18], similar stability characteristics was observed although they employed an interpolation scheme to compute the forcing at virtual boundary points not coinciding with the mesh points. Obviously, the stability limit is dependent on the adopted scheme. Exact stability limit of the method for a particular scheme is very important in three-dimensional applications, especially when turbulent flow is considered. The role of the α-term in the forcing is to add a natural oscillation in the course of enforcing the no-slip condition at a virtual boundary while the β-term adds a damping to the system, thus suppresses the oscillation. Therefore, generally we expect better performance of the virtual forcing with as large values of α and β as allowed by the stability limit. Rapid performance of the forcing becomes critical when the flow under consideration is turbulent since turbulence contains motions of a wide range of timescales down to the order of the Kolmogorov timescale.

As pointed out by Goldstein et al. [5], the stability of the method is determined not only by α and β but also by flow geometry. For this reason, no general rule for determination of the optimum values of α and β has been proposed. It is thus, necessary to derive a guideline for selecting α and β in using various schemes and the corresponding stability limit.

The aim of the present paper is to provide the accurate stability limits and to suggest the optimum values of the forcing gains for various temporal schemes in three-dimensional applications of a virtual boundary method.

2. Virtual boundary method

In this section, we briefly explain the virtual boundary method in three-dimensional applications. Including the virtual forcing term, the Navier–Stokes Equations read

$$\frac{\partial u_i}{\partial t} + u_j \frac{\partial u_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial p}{\partial x_i} + \nu \frac{\partial^2 u_i}{\partial x_j \partial x_j} + F_i, \tag{3}$$

where u_i, ρ, p and ν are the velocity component, density, pressure and viscosity of fluid. F_i is the virtual forcing located along virtual surface points X_j as given by Eq. (2) where the virtual surface velocity $U_i(t)$ is the fluid velocity at the virtual boundary point,

$$U_i(t) = u_i(X_1, X_2, X_3, t). \tag{4}$$

Through this dynamical procedure, it is expected that the velocities at the virtual boundaries quickly decay to zero when α and β are large enough. For an implementation of this method in numerical simulations, two approximation processes are necessary since generally the virtual boundary does not coincide with the computational meshes and the delta-function forcing is not directly realizable in the discrete mesh system. The first approximation occurs when the velocity at the immersed boundary point is extracted through interpolation using the velocities at the nearby mesh points. In the present study, the linear interpolation in three dimensions is adopted. Velocity at an immersed boundary point, therefore, can be written as a linear combination of velocities at nearby 8 mesh points,

$$U_i(t) = \sum_{j=1}^{8} W_j u_{i,j}(t), \tag{5}$$

where numbering of the mesh velocities and weights is shown in Fig. 1. The corresponding weight values are

$$\begin{aligned}
W_1 &= (1-\eta_x)(1-\eta_y)(1-\eta_z),\\
W_2 &= \eta_x(1-\eta_y)(1-\eta_z),\\
W_3 &= (1-\eta_x)(1-\eta_y)\eta_z,\\
W_4 &= \eta_x(1-\eta_y)\eta_z,\\
W_5 &= (1-\eta_x)\eta_y(1-\eta_z),\\
W_6 &= \eta_x\eta_y(1-\eta_z),\\
W_7 &= (1-\eta_x)\eta_y\eta_z,\\
W_8 &= \eta_x\eta_y\eta_z,
\end{aligned} \tag{6}$$

respectively, where η_x, η_y and η_z are defined in the figure. Then the virtual forcing is obtained using (2). For much higher-order approximation, a higher-order interpolation such as cubic spline or the Hermite interpolation can be employed.

The second approximation occurs when this virtual forcing is spread back to the nearby mesh points for numerical simulation. When the same weights (6) are used in this procedure,

$$f_{i,j} = \frac{1}{N_s}\sum_{k=1}^{N_s} W_{j,k} F_i(X_{,k}, t), \tag{7}$$

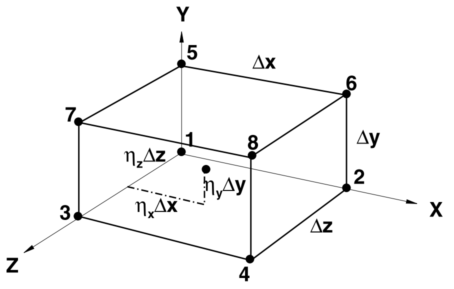


Fig. 1. Numbering of nearby mesh points of a virtual boundary point in three dimensions.

where N_s is the number of the total virtual surface points that influence the mesh point. This is a generalization of the 'area-weighted' virtual boundary method of Saiki and Biringen [18] to three dimensions. This 'volume-weighted' implementation of the virtual surface forcing appears reasonable since the method collects contributions from all the nearby virtual boundary points. Furthermore, it allows us to analyze the stability characteristics as will be shown in the next section. This also makes the method distinct from the alternative approach by Fadlun et al. [4] where they enforces the no-slip condition at few selected points.

3. Stability analysis

In this section, the refined stability analysis of the virtual boundary method introduced in the previous section is presented. First, we consider a one-dimensional case before generalization to multi-dimensional problems to understand how the method works although it is hypothetical. As shown in Fig. 2, an immersed boundary point is located at $x_1 + \eta(x_2 - x_1)$ between two mesh points, x_1 and x_2. Then the velocity at the immersed point, $U(t)$, can be approximated by the linear interpolation,

$$U(t) = (1 - \eta)u_1(t) + \eta u_2(t), \tag{8}$$

where $u_1(t)$ and $u_2(t)$ are velocities at the mesh points. The virtual forcing becomes

$$F(t) = \alpha \int^t U(t')\,\mathrm{d}t' + \beta U(t), \tag{9}$$

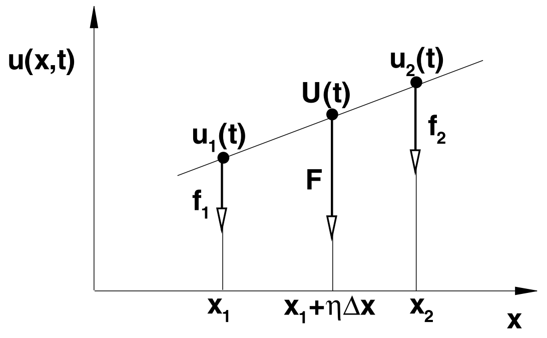


Fig. 2. Schematic showing interpolation and spreading in one dimension.

and this forcing is spread back to the nearby two mesh points using the same weights as used in the interpolation procedure as follows:

$$f_1(t) = (1-\eta)F(t), \tag{10}$$

$$f_2(t) = \eta F(t), \tag{11}$$

where no summation is necessary since only one immersed point is considered. The evolution equations for $u_1(t)$ and $u_2(t)$ are

$$\frac{\mathrm{d}u_1(t)}{\mathrm{d}t} = (1-\eta)^2\left(\alpha\int^t u_1(t')\,\mathrm{d}t' + \beta u_1(t)\right) + \eta(1-\eta)\left(\alpha\int^t u_2(t')\,\mathrm{d}t' + \beta u_2(t)\right), \tag{12}$$

$$\frac{\mathrm{d}u_2(t)}{\mathrm{d}t} = \eta(1-\eta)\left(\alpha\int^t u_1(t')\,\mathrm{d}t' + \beta u_1(t)\right) + \eta^2\left(\alpha\int^t u_2(t')\,\mathrm{d}t' + \beta u_2(t)\right). \tag{13}$$

For these coupled linear integro-differential equations, two eigenvalues exist: One of them is null and the other is $(1-\eta)^2+\eta^2$. The corresponding non-trivial eigenvector turns out to be $U(t)(=(1-\eta)u_1(t)+\eta u_2(t))$ that is the very velocity at the virtual boundary point. Therefore,

$$\frac{\mathrm{d}U(t)}{\mathrm{d}t} = ((1-\eta)^2+\eta^2)\left(\alpha\int^t U(t')\,\mathrm{d}t' + \beta U(t)\right). \tag{14}$$

This dynamical system with negative values of α and β has a solution converging to 0. The rate at which $U(t)$ decays, however, is slower than originally expected due to the prefactor, $(1-\eta)^2+\eta^2$ which ranges between 1/2 and 1 depending on the location of the virtual boundary point. However, stability analysis for this system, if needed, should be carried out with the prefactor equal to 1, since the worst case should be considered. This corresponds to the case where the virtual point approaches to either mesh points, i.e., $\eta \to 1$ or $\eta \to 0$. This implies that the stability of the method is determined by the virtual boundary point closest to the mesh point.

Next, we extend this analysis to the two-dimensional cases. Virtual points are now distributed along a virtual boundary as shown in Fig. 3. Velocity at the kth virtual boundary point can be approximated as follows when the bilinear interpolation scheme is used.

$$U_k(t) = (1-\eta_{x,k})(1-\eta_{y,k})u_1(t) + \eta_{x,k}(1-\eta_{y,k})u_2(t) + (1-\eta_{x,k})\eta_{y,k}u_3(t) + \eta_{x,k}\eta_{y,k}u_4(t), \tag{15}$$

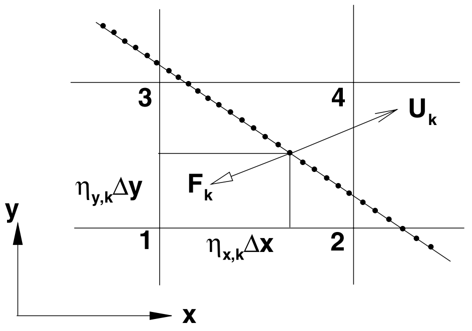


Fig. 3. Virtual boundary velocity and forcing in two dimensions.

where $\eta_{x,k}$ and $\eta_{y,k}$ are defined in the figure. The virtual forcing at the point then becomes

$$F_k(t) = \alpha \int^t U_k(t')\,\mathrm{d}t' + \beta U_k(t). \tag{16}$$

Rearrangement for the mesh velocities with the area-weighted spreading of the forcing yields

$$\begin{pmatrix} \frac{\mathrm{d}u_1(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_2(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_3(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_4(t)}{\mathrm{d}t} \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & p_3 & p_4 \\ p_2 & p_5 & p_4 & p_6 \\ p_3 & p_4 & p_7 & p_8 \\ p_4 & p_6 & p_8 & p_9 \end{pmatrix} \begin{pmatrix} \alpha \int^t u_1(t')\,\mathrm{d}t' + \beta u_1(t) \\ \alpha \int^t u_2(t')\,\mathrm{d}t' + \beta u_2(t) \\ \alpha \int^t u_3(t')\,\mathrm{d}t' + \beta u_3(t) \\ \alpha \int^t u_4(t')\,\mathrm{d}t' + \beta u_4(t) \end{pmatrix}, \tag{17}$$

where

$$p_1 = \frac{1}{N_s} \sum_k^{N_s} (1-\eta_{x,k})^2 (1-\eta_{y,k})^2, \tag{18}$$

$$p_2 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}(1-\eta_{x,k})(1-\eta_{y,k})^2, \tag{19}$$

$$p_3 = \frac{1}{N_s} \sum_k^{N_s} (1-\eta_{x,k})^2 \eta_{y,k}(1-\eta_{y,k}), \tag{20}$$

$$p_4 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}(1-\eta_{x,k})\eta_{y,k}(1-\eta_{y,k}), \tag{21}$$

$$p_5 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 (1-\eta_{y,k})^2, \tag{22}$$

$$p_6 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k}(1-\eta_{y,k}), \tag{23}$$

$$p_7 = \frac{1}{N_s} \sum_k^{N_s} (1-\eta_{x,k})^2 \eta_{y,k}^2, \tag{24}$$

$$p_8 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}(1-\eta_{x,k})\eta_{y,k}^2, \tag{25}$$

$$p_9 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k}^2. \tag{26}$$

The worst case from the stability point of view occurs when $\eta_{x,k} \to 0$ for all k or $\eta_{y,k} \to 0$ for all k, but not simultaneously. This corresponds to the case where the virtual boundary coincides with the mesh line element and the problem degenerates to the one-dimensional case with virtual boundary points distributed between two mesh points. In other words, the maximum eigenvalue among all possible situations can be observed when this case happens. Then the system is simplified. For example, when $\eta_{x,k} = 0$ for all k, $p_2 = p_4 = p_5 = p_6 = p_8 = p_9 = 0$ and p_1, p_3, p_7 can be well approximated by integrals as follows assuming that the virtual boundary points are densely and uniformly distributed.

$$p_1 \simeq \int_0^1 (1-\eta_y)^2 \,\mathrm{d}\eta_y = \frac{1}{3}, \tag{27}$$

$$p_3 \simeq \int_0^1 \eta_y(1-\eta_y) \,\mathrm{d}\eta_y = \frac{1}{6}, \tag{28}$$

$$p_7 \simeq \int_0^1 \eta_y^2 \,\mathrm{d}\eta_y = \frac{1}{3}. \tag{29}$$

The system (Eq. (17)) is then reduced to

$$\begin{pmatrix} \frac{\mathrm{d}u_1(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_3(t)}{\mathrm{d}t} \end{pmatrix} = \begin{pmatrix} \frac{1}{3} & \frac{1}{6} \\ \frac{1}{6} & \frac{1}{3} \end{pmatrix} \begin{pmatrix} \alpha \int^t u_1(t')\,\mathrm{d}t' + \beta u_1(t) \\ \alpha \int^t u_3(t')\,\mathrm{d}t' + \beta u_3(t) \end{pmatrix}. \tag{30}$$

Eigenvalues for this coupled system are 1/2 and 1/6. Then the resulting diagonalized system is

$$\begin{pmatrix} \frac{\mathrm{d}u_1^*(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_3^*(t)}{\mathrm{d}t} \end{pmatrix} = \begin{pmatrix} \frac{1}{2} & 0 \\ 0 & \frac{1}{6} \end{pmatrix} \begin{pmatrix} \alpha \int^t u_1^*(t')\,\mathrm{d}t' + \beta u_1^*(t) \\ \alpha \int^t u_3^*(t')\,\mathrm{d}t' + \beta u_3^*(t) \end{pmatrix}, \tag{31}$$

with $u_1^*(t) = u_1(t) + u_3(t)$ and $u_3^*(t) = u_1(t) - u_3(t)$. The first equation of Eq. (31) should be investigated for stability since it corresponds to the equation for the most dangerous mode with the largest eigenvalue. It is interesting to note that the average of the two mesh velocities approaches zero three times faster than the difference between them although both of them eventually decay to zero. We notice that the prefactor 1/2 of the first mode comes from the combination of the two processes, interpolation and spreading.

For stability analysis of the virtual boundary method in three dimensions, the above analysis is now extended to three-dimensional virtual boundaries. A virtual surface is approximated by distributed virtual points (see Fig. 1 for notations for nearby meshes of one virtual point). The resulting expression for the reduced dynamical system is

$$\begin{pmatrix} \frac{\mathrm{d}u_1(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_3(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_5(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_7(t)}{\mathrm{d}t} \end{pmatrix} = \begin{pmatrix} \frac{1}{9} & \frac{1}{18} & \frac{1}{18} & \frac{1}{36} \\ \frac{1}{18} & \frac{1}{9} & \frac{1}{36} & \frac{1}{18} \\ \frac{1}{18} & \frac{1}{36} & \frac{1}{9} & \frac{1}{18} \\ \frac{1}{36} & \frac{1}{18} & \frac{1}{18} & \frac{1}{9} \end{pmatrix} \begin{pmatrix} \alpha \int^t u_1(t')\,\mathrm{d}t' + \beta u_1(t) \\ \alpha \int^t u_3(t')\,\mathrm{d}t' + \beta u_3(t) \\ \alpha \int^t u_5(t')\,\mathrm{d}t' + \beta u_5(t) \\ \alpha \int^t u_7(t')\,\mathrm{d}t' + \beta u_7(t) \end{pmatrix}, \tag{32}$$

where $u_1(t)$, $u_3(t)$, $u_5(t)$ and $u_7(t)$ are function values at the nearby meshes (see Fig. 1). Detailed derivation of Eq. (32) is provided in the Appendix A. Here, $\eta_{x,k} \to 0$ for all k, corresponding to the case where the virtual surface coincides with a mesh area element. Eigenvalues for this system are $(1/4, 1/12, 1/12, 1/36)$ and the corresponding diagonal system is

$$\begin{pmatrix} \frac{\mathrm{d}u_1^*(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_3^*(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_5^*(t)}{\mathrm{d}t} \\ \frac{\mathrm{d}u_7^*(t)}{\mathrm{d}t} \end{pmatrix} = \begin{pmatrix} \frac{1}{4} & 0 & 0 & 0 \\ 0 & \frac{1}{12} & 0 & 0 \\ 0 & 0 & \frac{1}{12} & 0 \\ 0 & 0 & 0 & \frac{1}{36} \end{pmatrix} \begin{pmatrix} \alpha \int^t u_1^*(t')\,\mathrm{d}t' + \beta u_1^*(t) \\ \alpha \int^t u_3^*(t')\,\mathrm{d}t' + \beta u_3^*(t) \\ \alpha \int^t u_5^*(t')\,\mathrm{d}t' + \beta u_5^*(t) \\ \alpha \int^t u_7^*(t')\,\mathrm{d}t' + \beta u_7^*(t) \end{pmatrix}, \tag{33}$$

with the eigenvector relations,

$$\begin{pmatrix} u_1^*(t) \\ u_3^*(t) \\ u_5^*(t) \\ u_7^*(t) \end{pmatrix} = \begin{pmatrix} 1 & 1 & 1 & 1 \\ 0 & -1 & 1 & 0 \\ -1 & 0 & 0 & 1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \begin{pmatrix} u_1(t) \\ u_3(t) \\ u_5(t) \\ u_7(t) \end{pmatrix}. \tag{34}$$

The first equation of Eq. (33), which has the largest eigenvalue, should be investigated for stability. As in two dimensions, the average of the four mesh velocities decays to zero faster than any other mode.

Therefore, from the relations (Eqs. (14), (31), (33)), the equation that is subject to stability analysis is

$$\frac{\mathrm{d}u^*(t)}{\mathrm{d}t} = \frac{1}{2^{D-1}} \left(\alpha \int^t u^*(t')\,\mathrm{d}t' + \beta u^*(t) \right), \tag{35}$$

where D denotes dimension and $u^*(t)$ is the most dangerous mode. The factor of $1/2^{D-1}$ can cause four-times larger stability regime in the forcing parameter space than when not considered in three dimensions. By considering the worst case in the derivation of the virtual dynamical system with approximations of the involved summation, we could eliminate the problem-dependent property of the system. Here, the major assumption in our analysis is that the virtual boundary points are densely distributed, which replaces the summation by integration. However, it should be reminded that this analysis is possible only when the linear interpolation is used in obtaining the virtual velocity and the same weights are used in the volume-weighted spreading of the delta-function forcing. When a higher-order interpolation scheme is adopted, a larger value of the prefactor is expected since a high-order interpolation usually involves more nearby weights of larger magnitude than the linear interpolation. This implies more severe stability limit. A high-order interpolation, however, does not guarantee improvement of the accuracy of the method since then the accuracy of the virtual boundary method depends dominantly on the approximation scheme used in spreading the delta-function forcing.

We now investigate the stability characteristics of Eq. (35) for various time advancing schemes. When the forward Euler method is adopted for temporal integration, the discretized form of Eq. (35) is

$$u_{n+1} - u_n = \alpha' \int_0^{t_{n+1}} u(t')\,\mathrm{d}t' + \beta' u_n, \tag{36}$$

where $u_n = u(n\Delta t)$ with the time step Δt and $\alpha' = \alpha \Delta t^2 / 2^{D-1}$, $\beta' = \beta \Delta t / 2^{D-1}$, respectively. Note that the integral interval is up to t_{n+1}, not t_n, implying an implicit treatment. The integral is calculated by

$$\int_0^{t_{n+1}} u(t')\,\mathrm{d}t' = (u_0 + u_1 + \cdots + u_n)\Delta t \tag{37}$$

which is just an explicit treatment of the integral since u_{n+1} is not available. The other possible discretization can be obtained by carrying out the integration up to t_n. The stability characteristics for these two choices are basically the same under a transformation of the coefficients, $\alpha'' = \alpha'$ and $\beta'' = \alpha' + \beta'$, where α'' and β'' are coefficients for the second choice, i.e., integration up to t_n. Therefore, we will consider only the first choice in stability analysis. To obtain the recurrence formula for stability analysis, the discretized Eq. (36) at the previous time step is subtracted from Eq. (36) to yield

$$u_{n+1} - 2u_n + u_{n-1} = \alpha' u_n + \beta'(u_n - u_{n-1}). \tag{38}$$

Stability of this scheme can be determined by the behavior of $r(\equiv u_{n+1}/u_n)$. To obtain r, $u_n = u_0 r^n$ is substituted into Eq. (38), resulting in,

$$r^2 - (2 + \alpha' + \beta')r + 1 + \beta' = 0. \tag{39}$$

The stability condition that $\mathrm{abs}(r) \leqslant 1$ yields

$$-\alpha' - 2\beta' \leqslant 4, \tag{40}$$

or,

$$-\frac{\alpha\Delta t^2}{2^{D-1}} - \frac{2\beta\Delta t}{2^{D-1}} \leqslant 4, \tag{41}$$

limiting the magnitudes of α and β together.

For the second-order Adams–Bashforth scheme, the discretized equation becomes

$$u_{n+1} - u_n = \frac{3}{2}(\alpha'(u_0 + u_1 + \cdots + u_n) + \beta' u_n) - \frac{1}{2}(\alpha'(u_0 + u_1 + \cdots + u_{n-1}) + \beta' u_{n-1}), \tag{42}$$

where the same formula for the integral term as in the forward Euler scheme is used. The characteristic equation determining r for this scheme is

$$r^3 - \left(2 + \frac{3}{2}\alpha' + \frac{3}{2}\beta'\right)r^2 + \left(1 + \frac{1}{2}\alpha' + 2\beta'\right)r - \frac{1}{2}\beta' = 0. \tag{43}$$

Stability condition then yields

$$-\frac{\alpha\Delta t^2}{2^{D-1}} - \frac{2\beta\Delta t}{2^{D-1}} \leqslant 2. \tag{44}$$

This is the same relation as Eq. (1) with k equal to 2^{D-1}, thus eliminating problem dependency. It should be noted that stable region for the second-order Adams–Bashforth scheme is narrower than that of the forward Euler scheme, which is a general trend in stability analysis.

Similarly, for the third-order Adams–Bashforth scheme the discretized form of the governing equation is

$$\begin{aligned} u_{n+1} - u_n = {} & \frac{23}{12}(\alpha'(u_0 + u_1 + \cdots + u_n) + \beta' u_n) - \frac{16}{12}(\alpha'(u_0 + u_1 + \cdots + u_{n-1}) + \beta' u_{n-1}) \\ & + \frac{5}{12}(\alpha'(u_0 + u_1 + \cdots + u_{n-2}) + \beta' u_{n-2}), \end{aligned} \tag{45}$$

and the corresponding characteristic equation is

$$r^4 - \left(2 + \frac{23}{12}\alpha' + \frac{23}{12}\beta'\right)r^3 + \left(1 + \frac{16}{12}\alpha' + \frac{39}{12}\beta'\right)r^2 - \left(\frac{5}{12}\alpha' + \frac{21}{12}\beta'\right)r + \frac{5}{12}\beta' = 0, \tag{46}$$

from which the stability condition enforces

$$-\frac{\alpha\Delta t^2}{2^{D-1}} - \frac{2\beta\Delta t}{2^{D-1}} \leqslant \frac{12}{11}, \tag{47}$$

showing much narrow stability range than that of the second-order Adams–Bashforth scheme.

The stability characteristics for the family of Runge–Kutta scheme can be investigated in a similar manner. For a second-order Runge–Kutta scheme called the Modified Euler scheme, the equations for the two-step discretization are

$$u_{n+1/2} - u_n = \frac{1}{2}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/2}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{48}$$

$$u_{n+1} - u_{n+1/2} = \left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/2}\right) - \frac{1}{2}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/2}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{49}$$

with $u_{n+1/2}$ denoting the function value at the intermediate step. Subtracting the same equations at the previous time step from the above equations yields

$$u_{n+1/2} - u_{n-1/2} - u_n + u_{n-1} = \frac{1}{2}\Big(\alpha' I_1 + \beta'(u_n - u_{n-1})\Big), \tag{50}$$

$$u_{n+1} - u_n - u_{n+1/2} + u_{n-1/2} = \Big(\alpha' I_2 + \beta'(u_{n+1/2} - u_{n-1/2})\Big) - \frac{1}{2}\Big(\alpha' I_1 + \beta'(u_n - u_{n-1})\Big), \tag{51}$$

where the integrals can be calculated as follows:

$$I_1 = \frac{1}{\Delta t}\int_{t_{n-1/2}}^{t_{n+1/2}} u(t')\,\mathrm{d}t' = \frac{1}{2}(u_{n-1/2} + u_n), \tag{52}$$

$$I_2 = \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} u(t')\,\mathrm{d}t' = \frac{1}{2}(u_n + u_{n+1/2}). \tag{53}$$

Substituting $u_n = r^n$ and $u_{n+1/2} = pr^n$ into the Eqs. (50) and (51) and rearranging yield

$$(p-1)(r-1) = \frac{1}{4}\alpha'(p+r) + \frac{1}{2}\beta'(r-1), \tag{54}$$

$$(r-1)^2 = \frac{1}{2}\alpha'(p+1)r + \beta'(r-1)p. \tag{55}$$

Solving for r and p and using the condition that $\mathrm{abs}(r) \leqslant 1$, we can obtain the following stability condition:

$$-\frac{\alpha\Delta t^2}{2^{D-1}} - \frac{4\beta\Delta t}{2^{D-1}} \leqslant 8. \tag{56}$$

For another version of the second-order Runge–Kutta scheme which is known as the Heun's scheme, a little different stability regime is obtained. The discretized equations for this scheme are

$$u_{n+1/2} - u_n = \left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/2}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{57}$$

$$u_{n+1} - u_{n+1/2} = \frac{1}{2}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/2}\right) - \frac{1}{2}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/2}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{58}$$

and the resulting stability criterion is

$$-\frac{\alpha\Delta t^2}{2^{D-1}} - \frac{\beta\Delta t}{2^{D-1}} + \sqrt{1 - 2\frac{\beta\Delta t}{2^{D-1}} - \left(\frac{\beta\Delta t}{2^{D-1}}\right)^2} \leqslant 6, \tag{59}$$

showing narrower region than that of the modified Euler scheme.

Finally, for the third-order low-storage Runge–Kutta scheme, the discretized equations for the three substeps are

$$u_{n+1/3} - u_n = \frac{8}{15}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/3}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{60}$$

$$u_{n+2/3} - u_{n+1/3} = \frac{5}{12}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+2/3}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/3}\right) - \frac{17}{60}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/3}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{61}$$

$$u_{n+1} - u_{n+2/3} = \frac{3}{4}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1}} u(t')\,\mathrm{d}t' + \beta' u_{n+2/3}\right) - \frac{5}{12}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+2/3}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/3}\right), \tag{62}$$

where $u_{n+1/3}$ and $u_{n+2/3}$ denote the function values at the first and second substeps, respectively. $t_{n+1/3} = t_n + \frac{8}{15}\Delta t$, $t_{n+2/3} = t_n + \frac{2}{3}\Delta t$. Detailed stability analysis can be found in the Appendix B. Since the resulting stability criterion is too complicated to be explicitly expressed, we demonstrate it in Fig. 4 along with stability regimes for other schemes. Compared to the forward Euler or the second-order/third-order

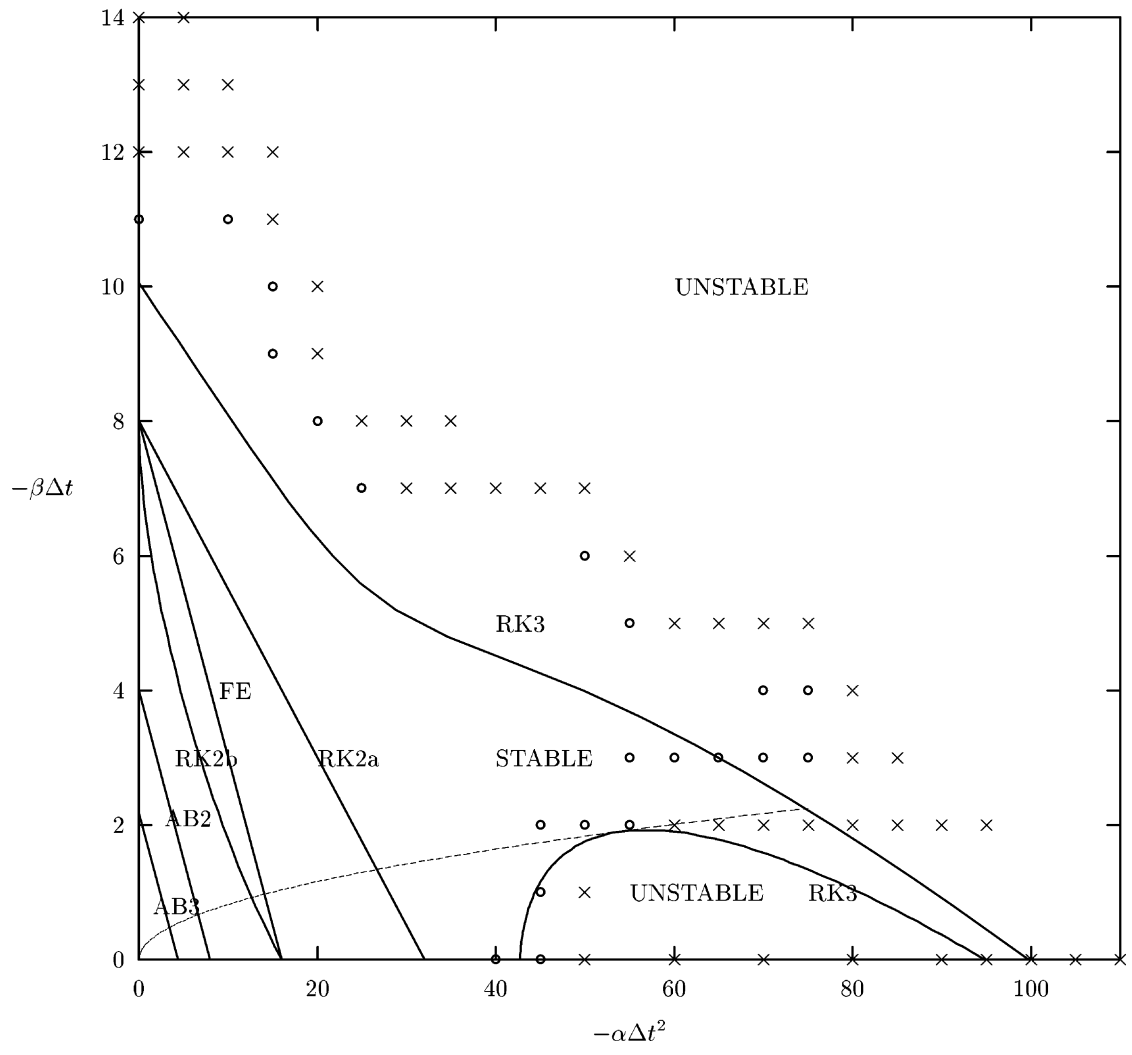


Fig. 4. Stability regimes for several time-advancing schemes: FE denotes the forward Euler scheme; AB2/AB3, the second-order/third-order Adams–Bashforth scheme; RK2a, the second-order Runge–Kutta scheme (modified Euler scheme); RK2b, the second-order Runge–Kutta scheme (Heun's scheme); RK3, the third-order Runge–Kutta scheme. The dot/cross denotes stable/unstable cases obtained in an application of the virtual boundary method to turbulent channel flow containing a box on the bottom.

Adams–Bashforth schemes, the stability regimes of the Runge–Kutta schemes are much wider. Among them, the third-order Runge–Kutta scheme possesses the widest stability range. Specially, the stability range for the third-order Runge–Kutta scheme is about 10 times wider in $-\alpha\Delta t^2$ direction than that of the second-order Adams–Bashforth scheme which is one of the most popular schemes. Since the larger $-\alpha\Delta t^2$, the better the virtual dynamical system performs as will be shown in the next section, the third-order Runge–Kutta scheme is recommended for implementations of the virtual boundary method.

For the third-order Runge–Kutta scheme, it can be observed that there is an unstable region inside the stable regime near the $-\alpha\Delta t^2$ axis. We draw a parabola which is tangential to that unstable region as shown in Fig. 4. The narrow region surrounded by a portion of this parabola, two neutral stability curves and the $-\alpha\Delta t^2$ axis is practically unreachable when the constant CFL condition is used since the time step can decrease in the course of computation, implying moving along a parabola, $-\alpha\Delta t^2 = -\alpha/\beta^2(-\beta\Delta t)^2$, into the unstable region for the given values of α and β. When the constant time-step condition is used, however, this region can be taken advantage of.

The dot/cross symbols denote actual cases that show stable/unstable performances in applications to a direct numerical simulation of turbulent channel flow containing a box on the bottom wall using the third-order Runge–Kutta scheme and the constant CFL condition (see the next section). Since the time step varies during computation, the maximum time step was used. Since our stability analysis is based on the worst case scenario, i.e., the virtual surface coincides with the computational mesh area element, the current case shows a little wider stable regime. Overall, an agreement between the theoretical estimate and numerical result is excellent.

Finally we need to mention about the use of an implicit scheme in the integration of the virtual forcing although a real implementation becomes very complicated due to the interpolation and redistribution processes. For example, when the backward Euler scheme is used, the discretized equation becomes

$$u_{n+1} - u_n = \alpha'(u_0 + u_1 + \cdots + u_n) + \beta' u_{n+1}, \tag{63}$$

where the integral is treated explicitly since an implicit treatment would be equivalent to a modification of β'. The characteristic equation then becomes

$$(1-\beta')r^2 - (1+\alpha'-\beta')r + 1 = 0, \tag{64}$$

from which the condition, $\mathrm{abs}(r) \leqslant 1$, yields

$$-\frac{\alpha\Delta t^2}{2^{D-1}} + \frac{2\beta\Delta t}{2^{D-1}} \leqslant 4, \tag{65}$$

indicating that $-\beta\Delta t$ has no bound while $-\alpha\Delta t^2$ has the same bound as the forward Euler scheme. As will be shown in the next section, however, the performance of the scheme depends more sensitively on α than on β. The bigger $-\alpha$ and the smaller $-\beta$, the better the performance is. Therefore, just an implicit treatment of the β-term does not allow the use of a bigger time step than an explicit treatment. Thus, for the best performance, an implicit scheme has the same time-step restriction as the explicit scheme. This appears opposite to the finding of Fadlun et al. [4] that an implicit treatment of the β-term substantially increases the allowed CFL number. It, however, is not clear whether they select the optimal values of α and β such that the scheme performs the best.

4. Illustrative examples

For evaluation of the virtual boundary method and the corresponding stability, three cases are considered: a direct numerical simulation of the three-dimensional turbulent channel flow with a surface-mounted box; two-dimensional startup flow around a cylinder; a direct numerical simulation of turbulent boundary layer over a rough wall.

4.1. Three-dimensional turbulent channel flow with a surface-mounted box

Stability of the virtual boundary method is investigated by applying the method to a direct numerical simulation of the three-dimensional turbulent flow arising from a surface-mounted box. Before turning on the forcing, the channel flow is fully developed at $Re_\tau(= u_\tau h/\nu) = 100$, where u_τ, h and ν are the wall-shear velocity, the channel half gap and the viscosity of fluid, respectively. A spectral simulation is carried out with Fourier expansions in the streamwise ($x-$) and spanwise ($z-$) directions and a Chebyshev expansion in the wall-normal ($y-$) direction. The size of the channel is $(4\pi, 2, 4\pi/3)h$ and the resolution is $48 \times 65 \times 48$ in $(x-, y-, z-)$ directions, which is an extended grid owing to the 3/2 dealiasing in $(x-, z-)$ directions. Undersirable oscillations have been reported in spectral calculations with an almost delta-function like forcing in the previous studies [5,18]. We observed similar non-growing oscillations specially in the wall-normal direction, but they did not influence the stability or the convergence of the virtual boundary method. The third-order Runge–Kutta scheme and the Crank–Nicolson scheme are adopted for the integrations of the non-linear terms including the virtual boundary forcing and the viscous terms, respectively. Details can be found in Lundbladh et al. [14]. The integral term in the forcing is integrated according to the formulation explained in the previous section. The dimension of the box is $(1.0, 0.2, 0.5)h$ which is mounted in the center of the bottom wall.

Three different number of the virtual surface points were tested to investigated the sensitivity of the result to the configuration. On each surface of the box, 20×20, 40×40 and 80×80 virtual points are uniformly distributed. These correspond to 1961, 7921 and 31 841 points in total, respectively. For quantitative comparison, an l_2-*norm* error defined as follows is monitored,

$$l_2\textit{-norm}\ \text{error} \equiv \sqrt{\frac{1}{N_s}\sum_{k=1}^{N_s} U(X_k, t)^2}, \tag{66}$$

where $U(X_k, t)$ is the surface velocity at the kth surface point. The l_2-*norm* errors for the three cases are illustrated in Fig. 5. It shows that when the number is large enough, the behavior of the error is almost the same whereas too small number of the virtual points leads to instability. Our stability analysis was based on the assumption that the virtual points are densely distributed, thus stability is not guaranteed when the number of the virtual points is not large. Therefore, in all following examples we distributed as many points as possible to take advantage of the precise stability characteristics as long as the computational overhead is not severe. This is one drawback of the current method that potential users should be careful about.

The behavior of the error of the streamwise component of the virtual velocity is shown in Fig. 6 for three different forcing gain values. The time step is determined to satisfy the constant CFL-number condition with the maximum CFL number equal to 1.0. Although these three cases are marginal in the stability diagram (Fig. 4), the errors show a distinct feature: the larger $-\alpha\Delta t_{\max}^2$, the smaller value the error converges to, while $-\beta\Delta t_{\max}$ influences only the initial behavior of the error, i.e., the larger $-\beta\Delta t_{\max}$, the faster the error decays. Therefore, the α-term plays the more important role in enforcing the no-slip condition than the β-term. It should be noted that although an order-one CFL number is maintained in the computation, the error decreases to the level of 1/1000 of the initial value for the best case. The error, however, does not decrease any further due to the dynamic nature of turbulence.

To further examine the roles of the parameters, the Eq. (35) is differentiated with respect to time to yield

$$\frac{d^2u}{dt^2} - \frac{\beta}{2^{D-1}}\frac{du}{dt} - \frac{\alpha}{2^{D-1}}u = 0. \tag{67}$$

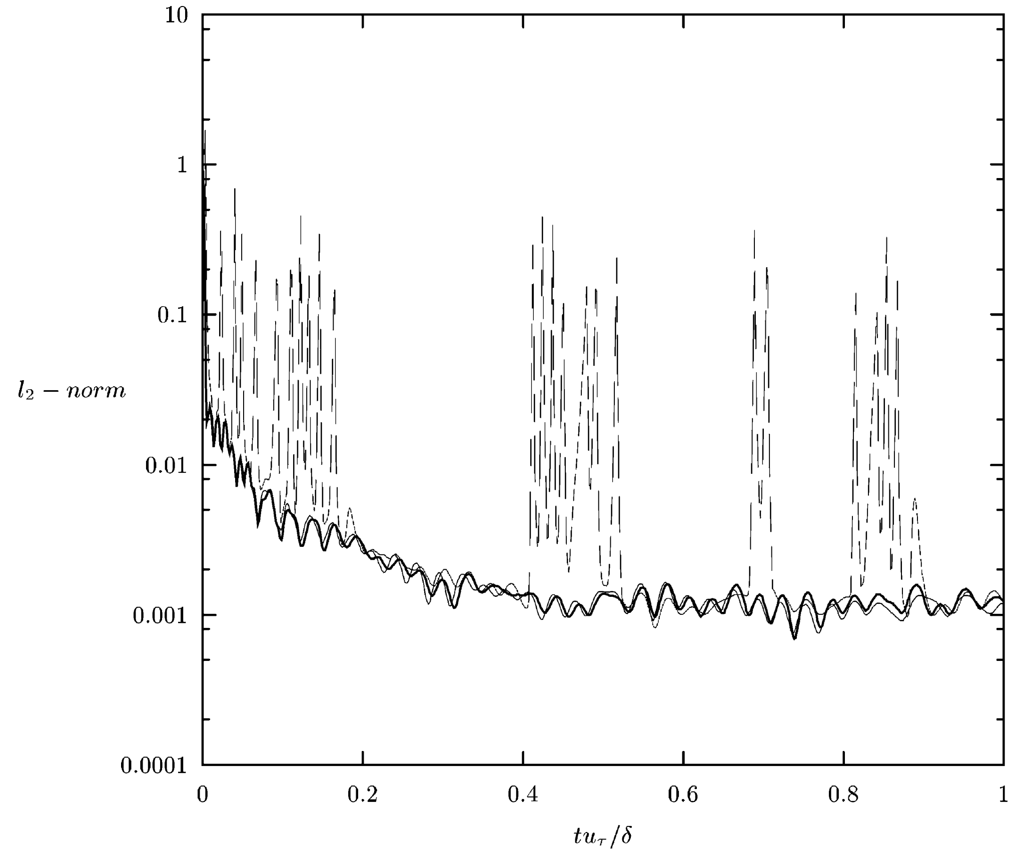


Fig. 5. l_2-*norm* error of the virtual boundary velocity in the streamwise direction normalized by the initial value for three different numbers of virtual points: thick-solid line, $N_p = 31841$; thin-solid line, $N_p = 7921$; dashed line, $N_p = 1961$. $-\alpha\Delta t^2_{\max} = 40$, $-\beta\Delta t_{\max} = 4$ in all cases.

Since α and β are negative, the above equation represents a damped oscillatory system. For this system, the natural frequency normalized by the time step can be easily obtained.

$$\omega_n\Delta t = \frac{1}{2^{D-1}}\sqrt{-\alpha\Delta t^2 - \frac{\beta^2\Delta t^2}{4}}. \tag{68}$$

For a virtual boundary method to perform properly, this frequency should be as large as possible. Otherwise, the virtual dynamical system cannot track the changing flow. A large frequency obviously means large $-\alpha\Delta t^2$ and small $-\beta\Delta t$. Therefore, we conclude that for the best performance, α and β should be selected for given Δt so as to maximize this natural frequency within the stability limit. The recommended values for α and β which maximize the natural frequency for various time-advancing schemes are listed in Table 1 along with the corresponding natural frequencies. The natural frequencies of the third-order Runge–Kutta scheme is of order of 2, which are much larger than those of the Adams–Bashforth schemes. It means that with the optimal gains the virtual forcing works much faster than the flow evolves. Our simulation result in turbulent flow as shown in Fig. 6 supports this analysis. Note that for most schemes, the recommended value of β is zero. In real practice, however, a non-zero but small value of β might be helpful in the initial period for the better initial performance. It should be noted that the

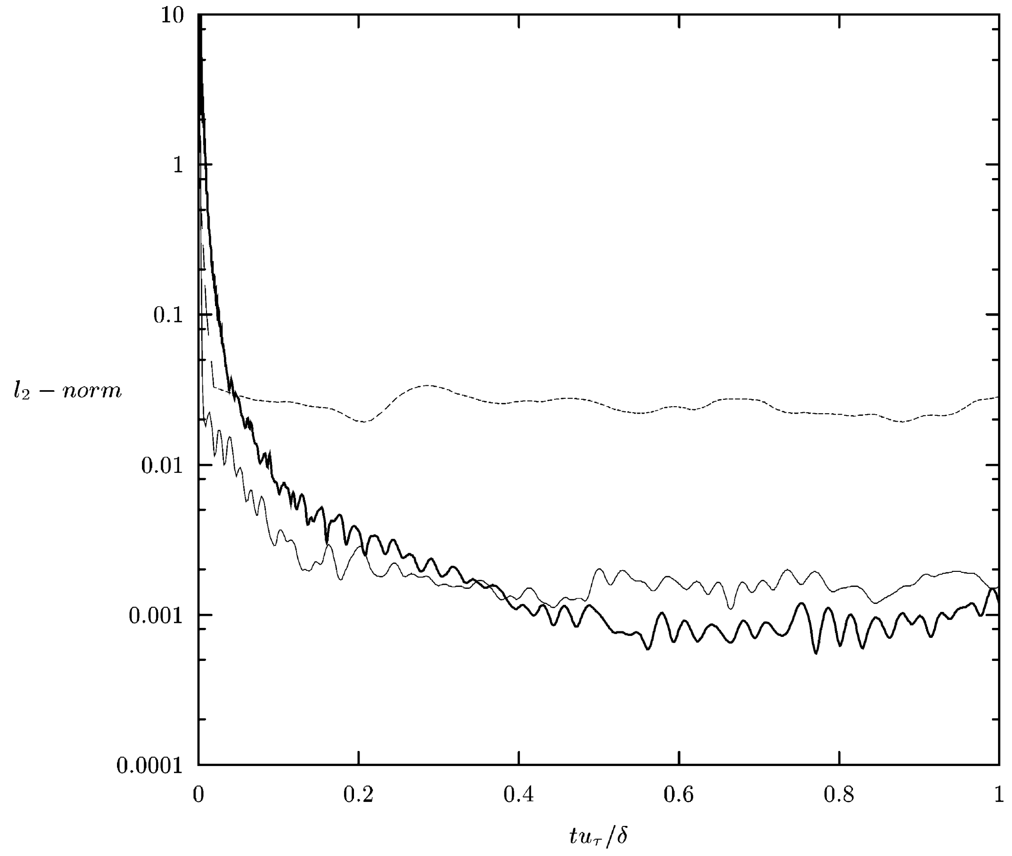


Fig. 6. l_2-*norm* error of the virtual surface velocity in the streamwise direction normalized by the initial value for three different forcing gains: thick-solid line, $-\alpha\Delta t^2_{\max} = 65$, $-\beta\Delta t_{\max} = 3$; thin-solid line, $-\alpha\Delta t^2_{\max} = 20$, $-\beta\Delta t_{\max} = 7$; dashed line, $-\alpha\Delta t^2_{\max} = 0$, $-\beta\Delta t_{\max} = 11$. The CFL number is maintained at 1.0.

Table 1
Recommended forcing parameters

Scheme	FE	AB2	AB3	RK2a	RK2b	RK3[a]	RK3[b]
$-\alpha\Delta t^2$	16	8	4.36	32	16	100	72
$-\beta\Delta t$	0	0	0	0	0	0	2.2
$\omega_n\Delta t$	1.0	0.71	0.15	1.41	1.0	2.5	2.1

[a] Constant time step.
[b] Constant CFL number.

magnitude of the recommended and tested gains ($|\alpha\Delta t^2|, |\beta\Delta t|$) is much larger than those used by the previous calculations [5,6,18,19].

As is shown in Fig. 7, the maximum CFL number also sets the converged error level. Although a smaller CFL number yields smaller error, an order of one CFL number appears enough for the approximation of the no-slip condition. Note that even when the maximum CFL number is 1.0, the error decreases below to 1/100th level of the initial value. This result serves as evidence that the virtual boundary method does not impose an extra time-step restriction other than the order-one maximum CFL number condition. This is

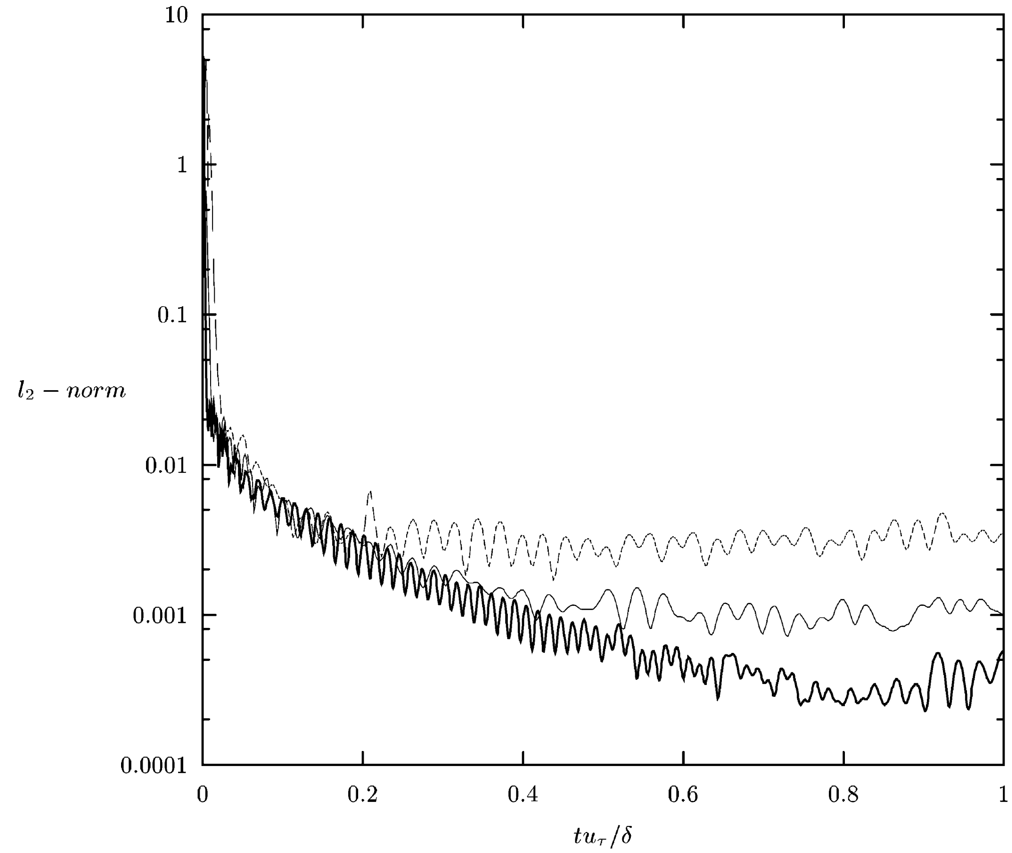


Fig. 7. l_2-*norm* error of the virtual surface velocity in the streamwise direction normalized by the initial value for three CFL numbers: thick-solid line, CFL = 0.25; thin-solid line, CFL = 0.5; dashed line, CFL = 1.0. $-\alpha\Delta t^2_{\max} = 45$, $-\beta\Delta t_{\max} = 1$ in all cases.

true only when the method is applied to three-dimensional flow and the third-order Runge–Kutta scheme is used since the corresponding stability regime is wider than any other scheme as shown in Fig. 4.

Fig. 8 shows how quickly the forcing puts the velocity at the virtual surface point to rest. Even in two time steps, the velocities at the virtual surface decreases below to 10% of the original values. Up to 10 time steps, the influence of the forcing is local. It takes a longer time for the effect of the forcing to propagate throughout the flow field. It is observed that flow naturally evolves inside the box and gets weaker as time increases.

We also tested placing the forcing inside the box too. Saiki and Biringen [18] suggested the use of the interior forcing to obtain a correct solution. We found that the solutions for the case with the boundary forcing only and the case with the boundary and interior forcing are different near the virtual surface since surrounding turbulence is chaotic as shown in Fig. 9. Physically the flow outside the box is independent of the flow inside, but the global nature of the expansion functions used in the simulation causes an interaction between them. However, a few grids away from the virtual boundary, the two cases are almost identical. This suggests that it is not necessary to apply the forcing inside the virtual boundaries in turbulent flows. However, in laminar flows the interior forcing was necessary to get a correct solution since the viscous effect is large. A test for laminar flow is presented in the following section. From the viewpoint of the mass conservation, the case with the interior forcing does not provide an improved result. Investigation of the residual velocity at the surface reveals that the l_2-*norm* error remains at the same level regardless of the use of the interior forcing.

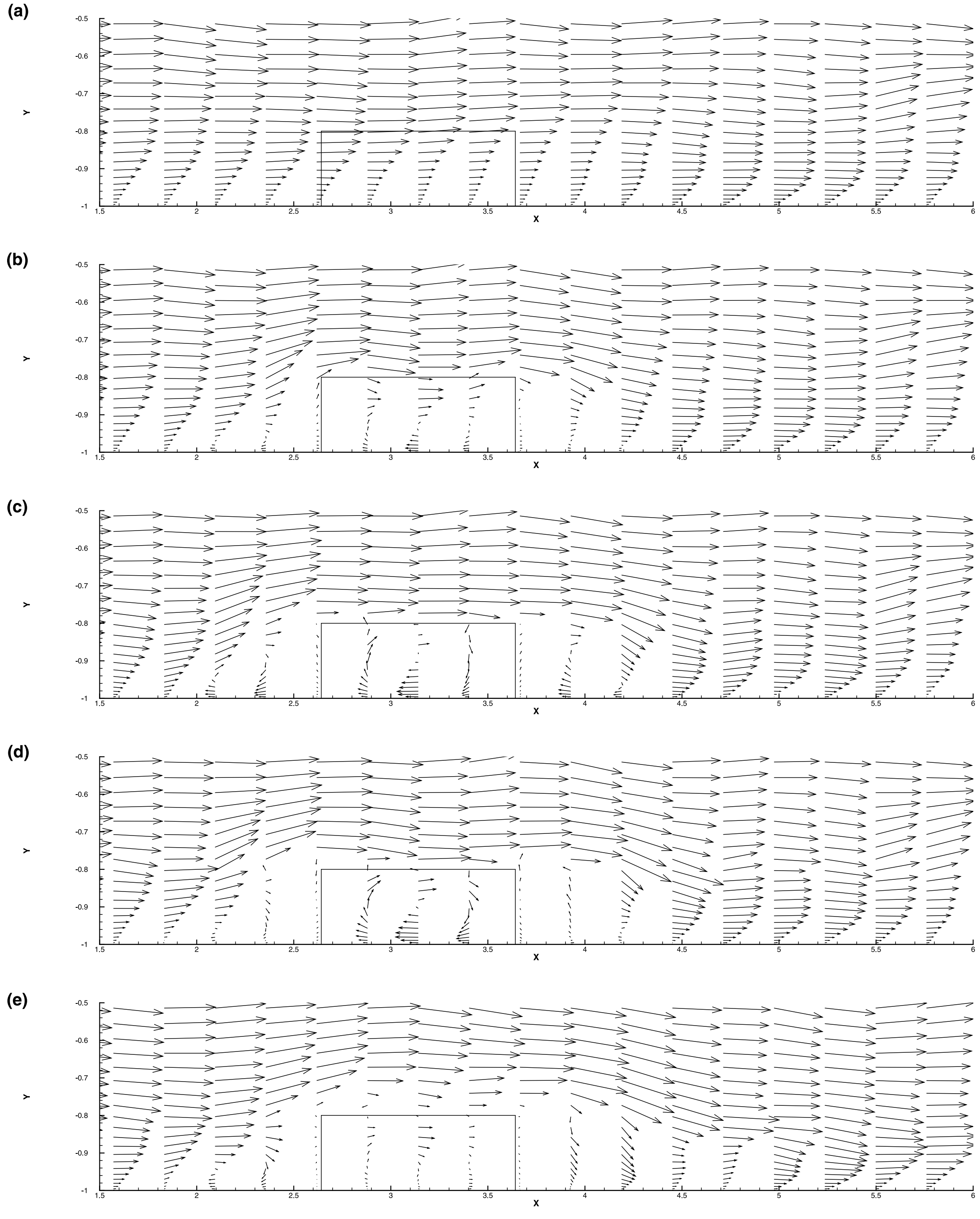
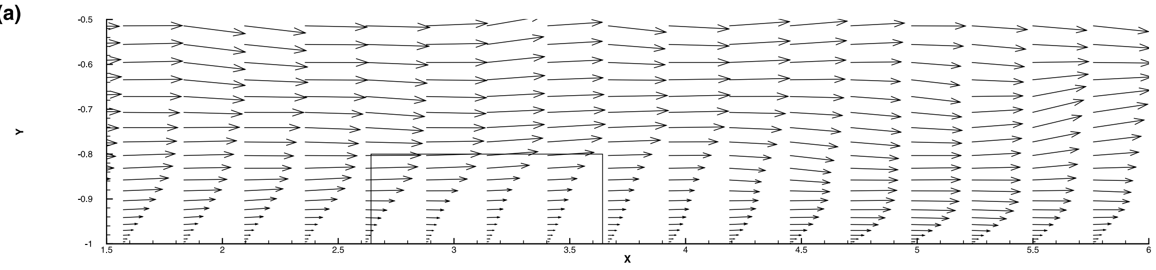
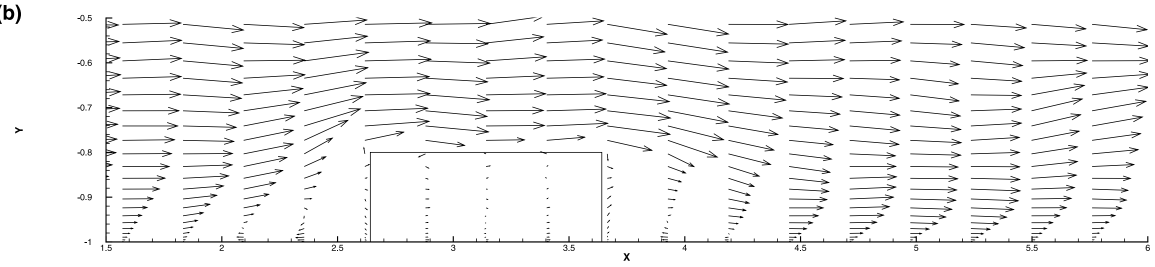
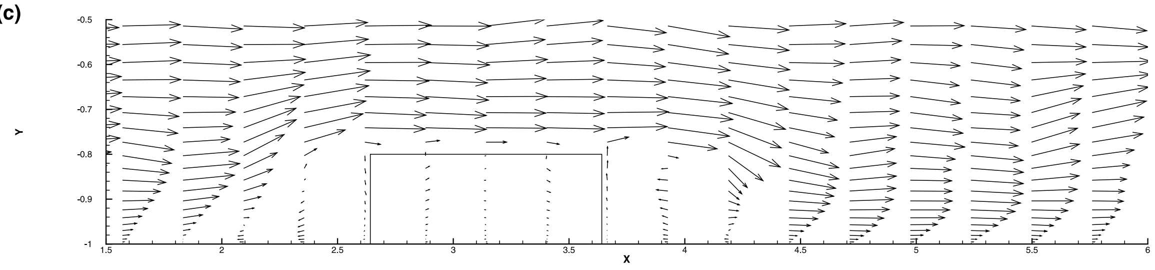
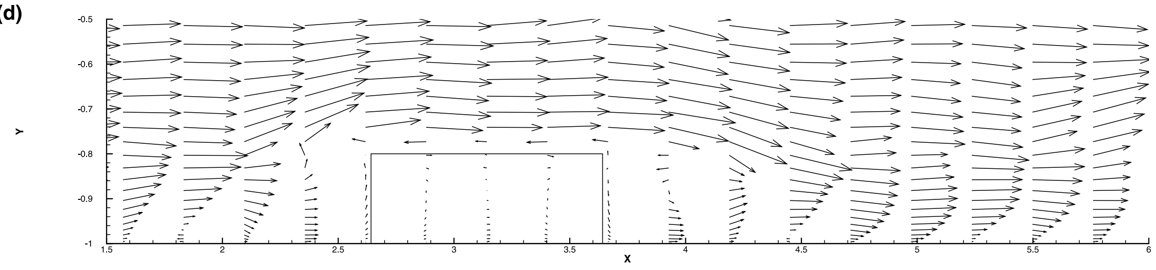


Fig. 8. Velocity vector plots in the x–y plane near the box mounted on the bottom wall at (a) 0, (b) 2, (c) 5, (d) 10, and (e) 100 time steps since the surface forcing is turned on. $-\alpha\Delta t_{\max}^2 = 40$, $-\beta\Delta t_{\max} = 4$.

(a)


(b)


(c)


(d)


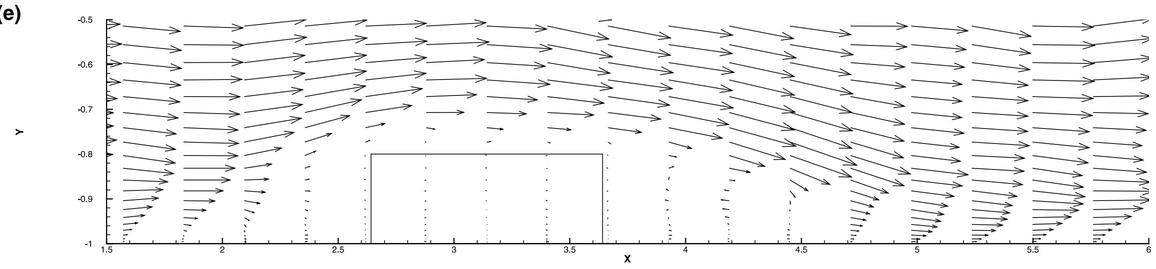
(e)


Fig. 9. Velocity vector plots in the x–y plane near the box mounted on the bottom wall at (a) 0, (b) 2, (c) 5, (d) 10, and (e) 100 time steps since the surface and interior forcing is turned on. $-\alpha\Delta t_{\max}^2 = 40$, $-\beta\Delta t_{\max} = 4$.

4.2. Startup flow around a cylinder

For a further validation of the stability characteristics presented in Section 3 and Fig. 4, we carried out a simulation of the same cylinder startup problem as the one given in Goldstein et al. [5]. For this problem, experimental data was provided by Bouard and Coutanceau [2]. Our computational domain is [2,2] in two dimensions and a cylinder of diameter 0.44 is located at the center. Periodicity is assumed in the streamwise direction and impermeability and constant streamwise velocity condition are imposed at the top and bottom boundaries. Since we are interested in the initial behavior of flow around a cylinder which impulsively starts, such a small computational domain is enough. Our spectral code uses 192 Fourier modes in the streamwise direction and 257 Chebyshev modes in the normal direction. The time stepping algorithm used is the third-order Runge–Kutta method. Ten thousand virtual boundary points are uniformly distributed along the boundary of the cylinder and extra 20 000 virtual points are located in the interior of the cylinder. We also tested the case with boundary forcing alone and found that the interior forcing was necessary to yield a solution closer to the experimental data. In our test in turbulent flow, the difference between the cases with and without the interior forcing was negligible in the region a few grids further than the virtual surface. This suggests that the viscous effect and global nature of the expansion function used in our simulation cause such difference. Main differences between our simulation and Goldstein et al.'s

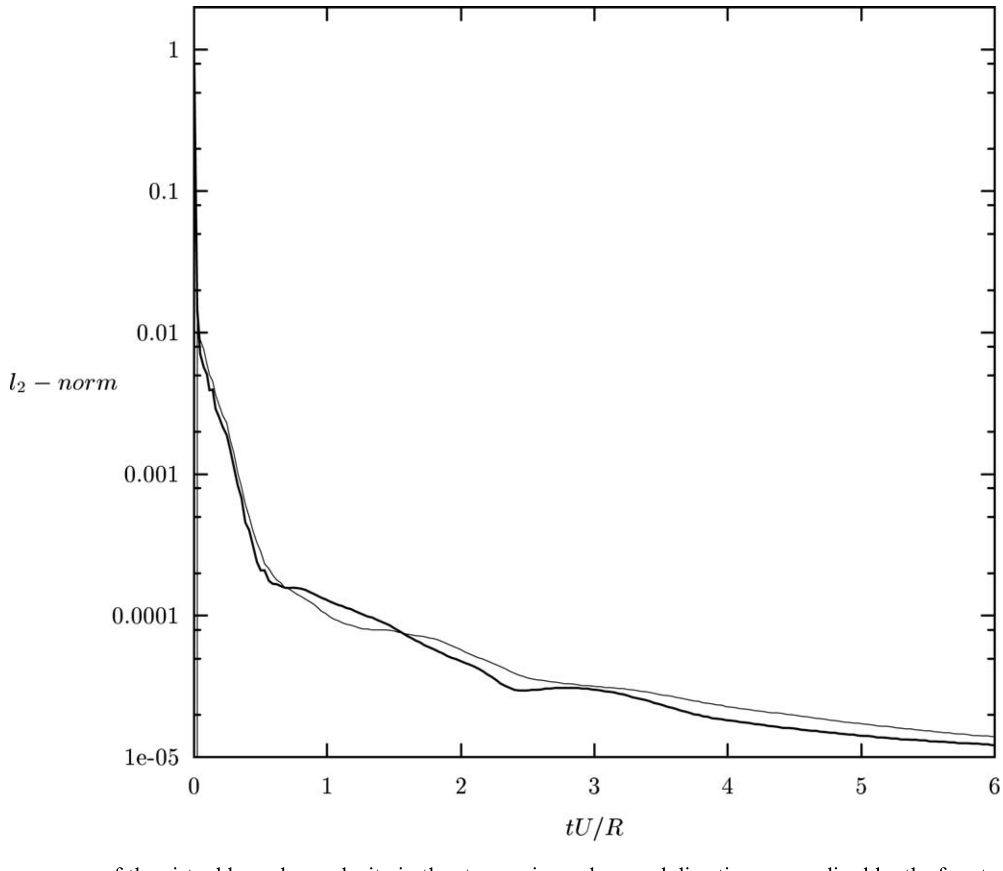


Fig. 10. l_2-*norm* error of the virtual boundary velocity in the streamwise and normal directions normalized by the freestream velocity in the cylinder problem: thick-solid line, streamwise velocity; dashed line, normal velocity. $-\alpha\Delta t_{\max}^2 = 30$, $-\beta\Delta t_{\max} = 2$.

[5] simulation are that many virtual points were located along the surface and interior of the cylinder and area-weighted forcing was used in our simulation while the virtual points were selected as the nearest grid points to the cylinder and smoothing of the forcing was done in their simulation.

To simulate flow around an impulsively starting cylinder, we turned on the forcing at $t = 0$ after velocity was initialized as uniform flow. The Reynolds number (UD/ν) is 550. Here, U and D are the free stream velocity and the cylinder diameter. The maximum CFL number was maintained as 0.1 to quickly enforce the no-slip condition at the surface of the cylinder. Throughout simulation, $\alpha\Delta t^2_{\max} = -30$ and $\beta\Delta t_{\max} = -2$. The l_2-*norm* error of the virtual boundary velocity quickly decays as shown in Fig. 10. In less than $t^*(= tU/R) = 0.05$, the error drops below 1% of the initial value. Here, R is the cylinder radius. Since the flow is laminar and slowly varies in time, the error constantly decreases with time. It should be noted that our forcing parameter values are significantly greater than those used in Goldstein et al. [5] ($\alpha\Delta t^2 = -0.002$ or -0.02 and $\beta\Delta t = -0.3$).

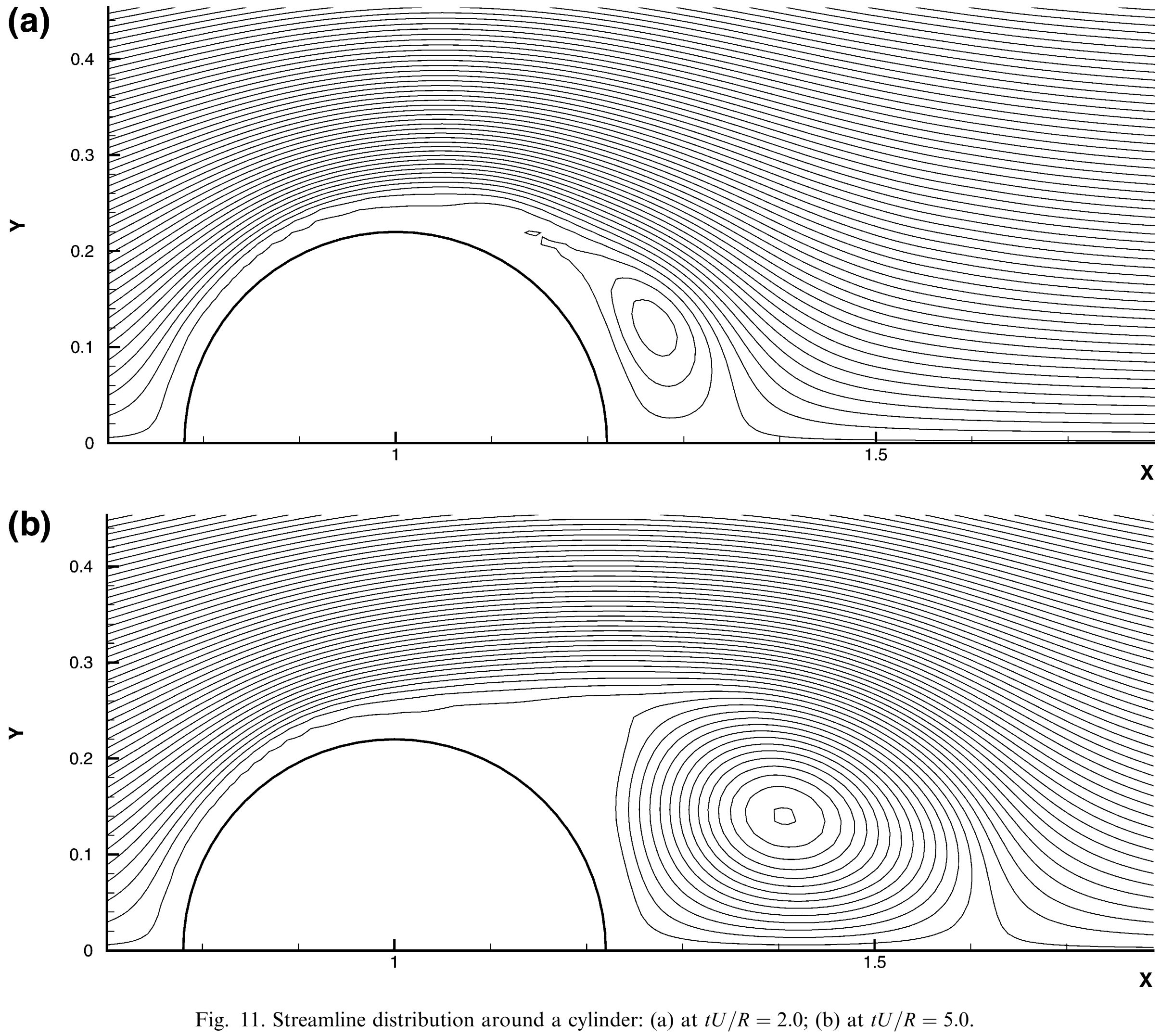


Fig. 11. Streamline distribution around a cylinder: (a) at $tU/R = 2.0$; (b) at $tU/R = 5.0$.

Streamline distributions near the cylinder at $t^* = 2.0$ and 5.0 are illustrated in Fig. 11. Since we have not done any smoothing of the forcing, weak oscillation in streamlines is observed near the surface of the cylinder. The shape of the main wake vortex compares very well with experimental data [2]. The streamwise and normal location of the center of the main vortex a and b and the aft stagnation point location L relative to the rear end point of the cylinder are plotted in Fig. 12 with experimental data [2]. At early times the apt stagnation point location was very accurately predicted by our simulation whereas the location of the vortex core was not well predicted since the wake vortex is too small and too close to the surface of the cylinder. At later times our simulation results deviate from the experimental data mainly due to the limited computation domain used in our simulation. Compared to Goldstein et al.'s [5] simulation, our simulation produced the results showing better agreement at the early times owing to the quick enforcement of the no-slip condition at the surface of the cylinder by using much larger $|\alpha|$ and $|\beta|$.

4.3. Turbulent boundary layer over a rough wall

The rough-wall turbulent boundary flow has long been a research topic due to its consequences in real engineering practice. While there have been many experimental studies, numerical investigation is rare because of difficulty in representing a rough wall in numerical simulations. Typical features of turbulent boundary over a rough wall, the suppression of viscous sublayer and the downward shift of the logarithmic velocity profile, have been observed in many experiments [1,9,10,20]. Despite the recent popular use of direct numerical simulation, the accurate representation of a rough wall in numerical simulation is still

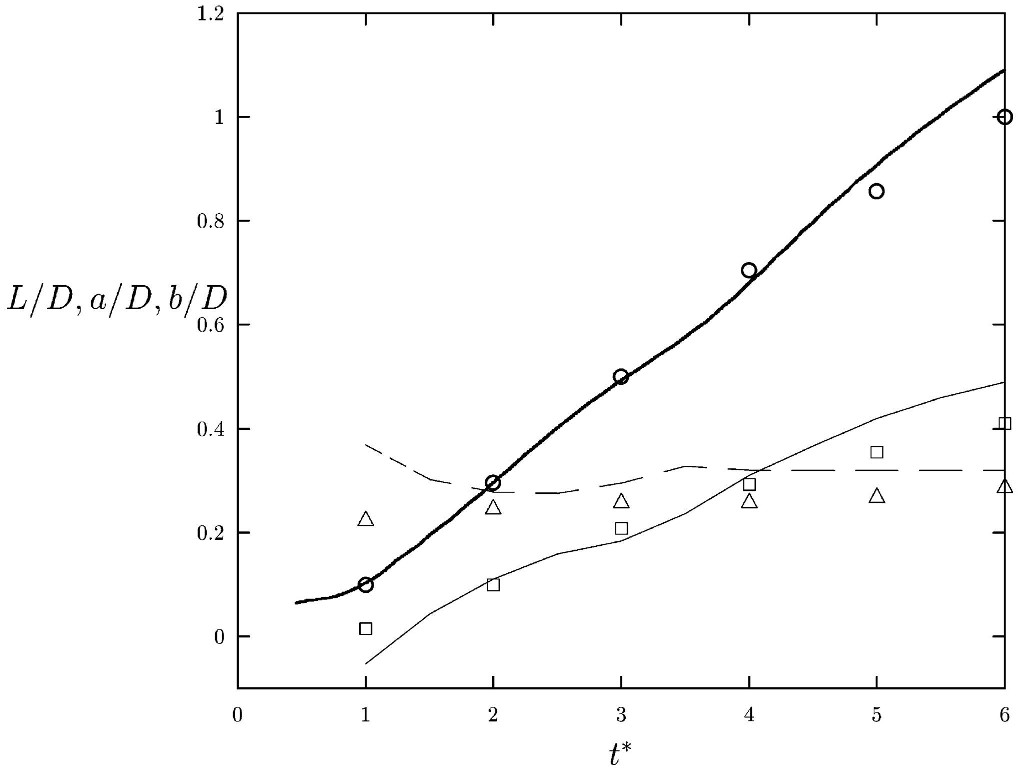
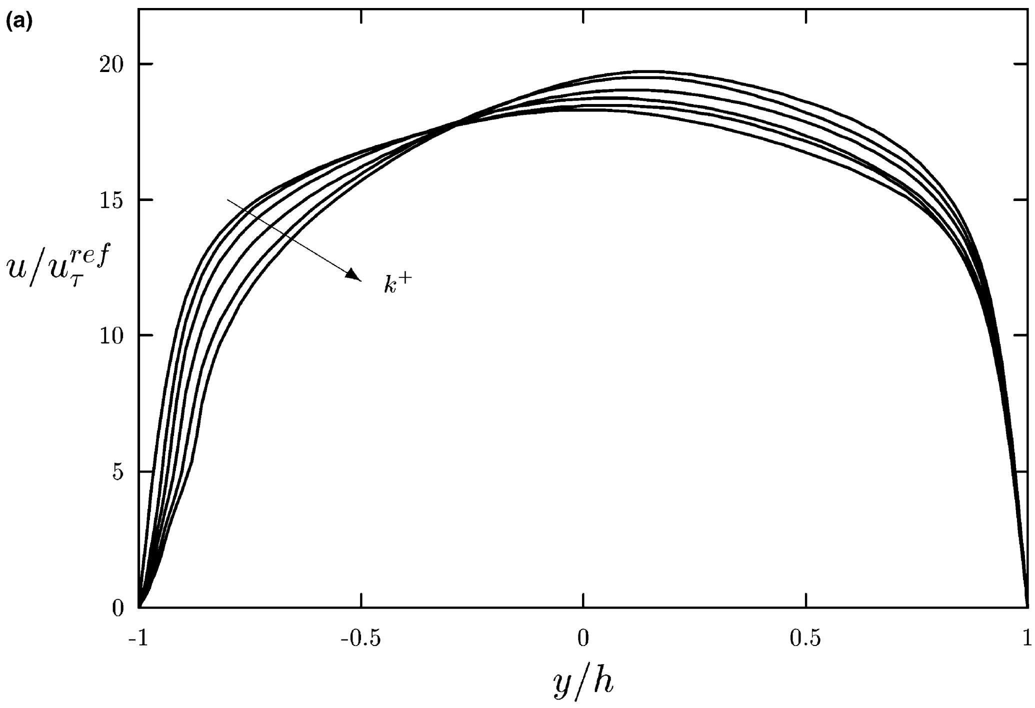


Fig. 12. Time variation of the closed wake length and the position of the main eddy center normalized by the cylinder diameter for $Re = 550$. Present simulation results: thick solid line, L/D; thin solid line, a/D; dashed line, b/D. Experimental results for $Re = 550$, Bouard and Coutanceau (1980): ○, L/D: □, a/D; △, b/D.



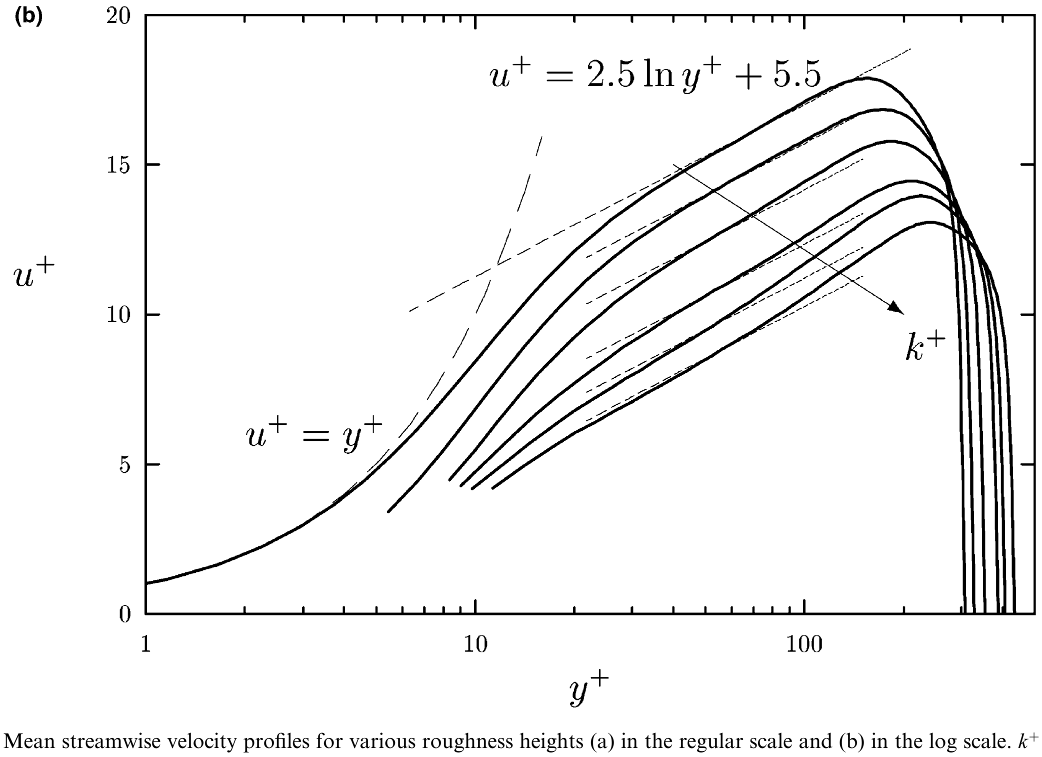


Fig. 14. Mean streamwise velocity profiles for various roughness heights (a) in the regular scale and (b) in the log scale. $k^+ = 0$, 6.58, 10.68, 15.80, 20.95, 27.14. The log profiles, $u^+ = \frac{1}{\kappa}\log y^+ + B - \Delta B^+$, are drawn for comparison.

domain is $(4\pi, 2, 4\pi/3)h$ in the streamwise, the wall-normal and the spanwise directions, respectively. The base flow with smooth walls from which the rough-wall enforcing starts has $Re_\tau = 150$. The computational resolution is maintained at $96 \times 129 \times 96$ on the extended grid although with this resolution the rough-wall surface is not represented in detail. What we are focusing on is not the accuracy in representing the rough wall but the feasibility of the virtual boundary method in such applications. Fully resolved simulations are postponed for now. Cases considered in our study are summarized in Table 2. Re_τ in the table is based on the wall-shear velocity of the rough wall obtained from the balance between the mean pressure gradient and the shear stresses of both walls, therefore, including the form drag due to the roughness elements. The same spectral code was used here as in the case of the previous sections; therefore, the forcing was integrated using the third-order Runge–Kutta scheme. The maximum CFL number is 1.0 and $-\alpha\Delta t^2_{\max} = 50$, $-\beta\Delta t_{\max} = 5$ for all cases considered.

Mean streamwise velocity profiles for various roughness heights are shown in Fig. 14 compared with the smooth-wall case. With the increase of the roughness height, the gradual downward shifting of the logarithmic velocity profile are observed, but the slope tends to increase. This is partly due to a low Reynolds number since the log region is too narrow for the shifted log profile to be pronounced well. For the same reason, we did not increase the roughness height further than $k^+ = 30$. For a fully rough-wall simulation, a simulation of much higher Reynolds number channel flow is necessary. The amount of the downward shift, ΔB^+, defined as follows is listed in Table 2:

$$u^+ = \frac{1}{\kappa}\log y^+ + B - \Delta B^+, \tag{70}$$

with $\kappa = 0.4$ and $B = 5.5$. y^+ is defined as

$$y^+ \equiv \frac{(y - y_0)u_\tau}{\nu}, \tag{71}$$

where y is the distance from the wall and y_0 is the location of the virtual origin. y_0 is determined such that the slope of the velocity in the log region is adjusted to $1/\kappa$ and is listed in Table 2. Since the roughness

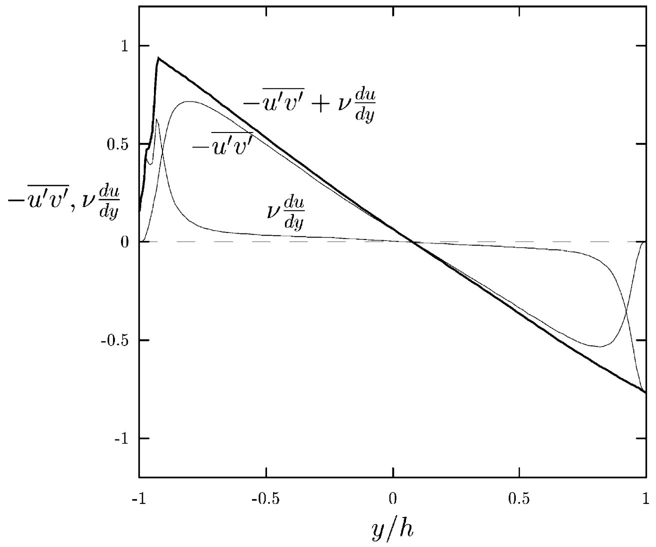
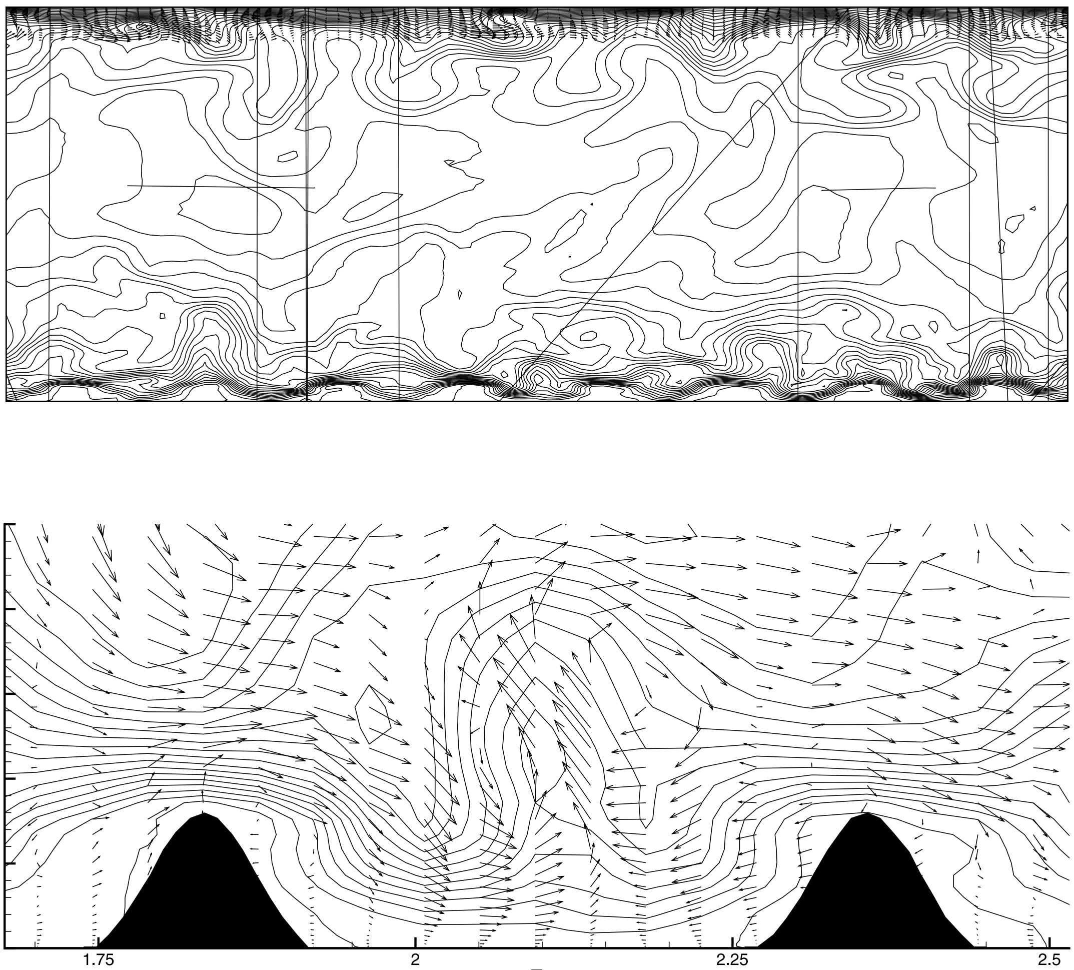


Fig. 15. Shear stress distribution (Case 2), normalized by the friction velocity of the rough wall.

heights considered are in the range of transitional roughness regime, the scaling relation of ΔB^+ for large k^+ was not observed. However, the amount of ΔB^+ is in the correct range for transitional roughness regime suggested by experiments [22].

Shear stress distributions for Case 2 are shown in Fig. 15. The viscous stress and the Reynolds stress are normalized by $|\tau_{\text{wall}}|/\rho$ of the rough wall. The wall-shear stress includes not only the viscous stress but also the form drag due to the roughness elements. Contribution to the total shear stress from this form drag is localized only below the tip of the roughness element as shown in Fig. 15. This contribution is 70–90% to the total wall-shear stress in our cases as listed in the last column of Table 2.

Fig. 16 shows a vector plot of (v, w) with contours of the streamwise velocity in a selected y–z plane. We intentionally chose this plane such that it shows the maximum height of the roughness element distribution. From the distribution of contours near a roughness element, it can be seen that the no-slip condition is not perfectly satisfied at the surface of the roughness element. However, considering the resolution used (Fig. 16 shows the computational mesh points where the vectors start), the representation of the rough wall is relatively good. Although we have not shown here, a very weak flow is formed inside the roughness



elements since we have not applied forcing there. A test with the interior forcing hardly changed the results. It can be observed that near the tips of the roughness elements high-shear regions are newly formed. Detailed statistics and the mechanism of the rough-wall turbulence obtained using fully resolved roughness elements will be reported elsewhere.

5. Conclusion

We have investigated the stability characteristics of the virtual boundary method proposed by Goldstein et al. [5] and later modified by Saiki and Biringen [18]. By detailed analysis in three dimensions, we found that the linear interpolation of the virtual boundary velocity and the subsequent spreading of the virtual forcing relax the time-step limit for stability up to four times. A variety of time-advancing schemes for the integration of the forcing have been examined for stability. Compared to the Adams–Bashforth schemes of the second/third order, the family of Runge–Kutta scheme possesses much wider stable region in the forcing gain space. By investigating the frequency of the resulting dynamical system, we were able to propose the optimum values of the forcing gains for each temporal scheme. Specially, when the third-order Runge–Kutta scheme is adopted, the virtual boundary method performs properly with order-one CFL number. Therefore, it was possible to apply the virtual boundary method to a direct numerical simulation of three-dimensional turbulent flows containing very short-timescale motion. However, it should be reminded that our stability analysis is based on the assumption that the virtual boundary points are densely distributed.

For the evaluation of the virtual boundary method, two kinds of turbulent flows and a two-dimensional laminar flow are considered: three-dimensional flow arising from a surface-mounted box; the flow around an impulsively starting cylinder; the rough-wall turbulent boundary layer flow. In the first example, the stability region in the forcing gain space obtained from stability analysis was confirmed with an excellent agreement. Also, the virtual boundary forcing was shown to work fast enough to response to the evolving turbulence. In the simulation of the startup flow around a cylinder, the virtual boundary method was applied with much larger forcing gains than the previously reported values. The results was in good agreement with the known experimental data. In the third example, for the first time, the method was applied to a simulation of turbulent flow over a rough wall without modeling. Typical characteristics of rough-wall turbulence, the downshift of the log profile of mean velocity was correctly simulated. Although we claim that the virtual boundary method was applied to two turbulent flows, we admit that both flows are weakly turbulent in the region where the forcing is applied, i.e., near the wall. However, we expect that the virtual boundary method will perform equally well in more severe turbulent environments.

Although all the test cases were computed using a spectral method, the virtual boundary method and its stability characteristics are easily extendible without modifications to other kinds of numerical method such as finite-difference method or finite-volume method in calculating unsteady three-dimensional flows.

Finally, we need to mention about the computational overhead. The overhead is definitely linearly proportional to the total number of virtual boundary points used. In our case with the computational mesh, $96 \times 129 \times 96$ and 73792 virtual surface points, the overhead is about 12%, which is not so imposing.

We have not focused on the accuracy of the virtual boundary method. For better accuracy of the interpolation, a higher-order interpolation scheme can be adopted. However, an improvement of the accuracy associated with spreading of the forcing is not so straightforward. It is still a challenging problem to represent a delta-function like forcing in numerical simulations adopting the discrete mesh system. In a finite-volume sense, it might be possible to formulate the jump conditions incorporating the forcing across the virtual boundary [12,13]. Or, an adaptive grid method can be facilitated near the virtual boundary [17]. One drawback of the virtual boundary method is the fact that the mass conservation is not perfectly satisfied in the region surrounded by the virtual surfaces. Placing the forcing in the interior as well as on the boundaries does not fix this problem. Although this problem does not seem to significantly influence the solution, this issue should be resolved. It requires a further study. We are currently investigating on the improvement of the accuracy of the virtual boundary method.

Acknowledgements

We are grateful to the reviewers for very fruitful comments. We acknowledge the support by the Korea Research Foundation through the Grant 2001-041-E00021.

Appendix A. Derivation of the reduced system in three dimensions

We present in this section the derivation of reduced system (Eq. (32)) in three dimensions. Velocity at the kth virtual boundary point is

$$\begin{aligned} U_k(t) = {} & (1-\eta_{x,k})(1-\eta_{y,k})(1-\eta_{z,k})u_1(t) + \eta_{x,k}(1-\eta_{y,k})(1-\eta_z z,k)u_2(t) \\ & + (1-\eta_{x,k})\eta_{y,k}(1-\eta_{z,k})u_3(t) + \eta_{x,k}\eta_{y,k}(1-\eta_{z,k})u_4(t) + (1-\eta \end{aligned}$$

$$f_8(t) = \frac{1}{N_s}\sum_k^{N_s} \eta_{x,k}\eta_{y,k}\eta_{z,k}F_k(t), \tag{A.9}$$

where f_j denotes the forcing at the mesh points. Rearrangement for the mesh velocities yields the following dynamical system,

$$\begin{pmatrix} \frac{du_1(t)}{dt} \\ \frac{du_2(t)}{dt} \\ \frac{du_3(t)}{dt} \\ \frac{du_4(t)}{dt} \\ \frac{du_5(t)}{dt} \\ \frac{du_6(t)}{dt} \\ \frac{du_7(t)}{dt} \\ \frac{du_8(t)}{dt} \end{pmatrix} = \begin{pmatrix} p_1 & p_2 & p_3 & p_4 & p_5 & p_6 & p_7 & p_8 \\ p_2 & p_9 & p_4 & p_{10} & p_6 & p_{11} & p_8 & p_{12} \\ p_3 & p_4 & p_{13} & p_{14} & p_7 & p_8 & p_{15} & p_{16} \\ p_4 & p_{10} & p_{14} & p_{17} & p_8 & p_{12} & p_{16} & p_{18} \\ p_5 & p_6 & p_7 & p_8 & p_{19} & p_{20} & p_{21} & p_{22} \\ p_6 & p_{11} & p_8 & p_{12} & p_{20} & p_{23} & p_{22} & p_{24} \\ p_7 & p_8 & p_{15} & p_{16} & p_{21} & p_{22} & p_{25} & p_{26} \\ p_8 & p_{12} & p_{16} & p_{18} & p_{22} & p_{24} & p_{26} & p_{27} \end{pmatrix} \begin{pmatrix} \alpha\int^t u_1(t')\,dt' + \beta u_1(t) \\ \alpha\int^t u_2(t')\,dt' + \beta u_2(t) \\ \alpha\int^t u_3(t')\,dt' + \beta u_3(t) \\ \alpha\int^t u_4(t')\,dt' + \beta u_4(t) \\ \alpha\int^t u_5(t')\,dt' + \beta u_5(t) \\ \alpha\int^t u_6(t')\,dt' + \beta u_6(t) \\ \alpha\int^t u_7(t')\,dt' + \beta u_7(t) \\ \alpha\int^t u_8(t')\,dt' + \beta u_8(t) \end{pmatrix}, \tag{A.10}$$

where

$$p_1 = \frac{1}{N_s}\sum_k^{N_s}(1-\eta_{x,k})^2(1-\eta_{y,k})^2(1-\eta_{z,k})^2, \tag{A.11}$$

$$p_2 = \frac{1}{N_s}\sum_k^{N_s}\eta_{x,k}(1-\eta_{x,k})(1-\eta_{y,k})^2(1-\eta_{z,k})^2, \tag{A.12}$$

$$p_3 = \frac{1}{N_s}\sum_k^{N_s}(1-\eta_{x,k})^2\eta_{y,k}(1-\eta_{y,k})(1-\eta_{z,k})^2, \tag{A.13}$$

$$p_4 = \frac{1}{N_s}\sum_k^{N_s}\eta_{x,k}(1-\eta_{x,k})\eta_{y,k}(1-\eta_{y,k})(1-\eta_{z,k})^2, \tag{A.14}$$

$$p_5 = \frac{1}{N_s}\sum_k^{N_s}(1-\eta_{x,k})^2(1-\eta_{y,k})^2\eta_{z,k}(1-\eta_{z,k}), \tag{A.15}$$

$$p_6 = \frac{1}{N_s}\sum_k^{N_s}\eta_{x,k}(1-\eta_{x,k})(1-\eta_{y,k})^2\eta_{z,k}(1-\eta_{z,k}), \tag{A.16}$$

$$p_7 = \frac{1}{N_s}\sum_k^{N_s}(1-\eta_{x,k})^2\eta_{y,k}(1-\eta_{y,k})\eta_{z,k}(1-\eta_{z,k}), \tag{A.17}$$

$$p_8 = \frac{1}{N_s}\sum_k^{N_s}\eta_{x,k}(1-\eta_{x,k})\eta_{y,k}(1-\eta_{y,k})\eta_{z,k}(1-\eta_{z,k}), \tag{A.18}$$

$$p_9 = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 (1 - \eta_{y,k})^2 (1 - \eta_{z,k})^2, \tag{A.19}$$

$$p_{10} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k} (1 - \eta_{y,k}) (1 - \eta_{z,k})^2, \tag{A.20}$$

$$p_{11} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 (1 - \eta_{y,k})^2 \eta_{z,k} (1 - \eta_{z,k}), \tag{A.21}$$

$$p_{12} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k} (1 - \eta_{y,k}) \eta_{z,k} (1 - \eta_{z,k}), \tag{A.22}$$

$$p_{13} = \frac{1}{N_s} \sum_k^{N_s} (1 - \eta_{x,k})^2 \eta_{y,k}^2 (1 - \eta_{z,k})^2, \tag{A.23}$$

$$p_{14} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k} (1 - \eta_{x,k}) \eta_{y,k}^2 (1 - \eta_{z,k})^2, \tag{A.24}$$

$$p_{15} = \frac{1}{N_s} \sum_k^{N_s} (1 - \eta_{x,k})^2 \eta_{y,k}^2 \eta_{z,k} (1 - \eta_{z,k}), \tag{A.25}$$

$$p_{16} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k} (1 - \eta_{x,k}) \eta_{y,k}^2 \eta_{z,k} (1 - \eta_{z,k}), \tag{A.26}$$

$$p_{17} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k}^2 (1 - \eta_{z,k})^2, \tag{A.27}$$

$$p_{18} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k}^2 \eta_{y,k}^2 \eta_{z,k} (1 - \eta_{z,k}), \tag{A.28}$$

$$p_{19} = \frac{1}{N_s} \sum_k^{N_s} (1 - \eta_{x,k})^2 (1 - \eta_{y,k})^2 \eta_{z,k}^2, \tag{A.29}$$

$$p_{20} = \frac{1}{N_s} \sum_k^{N_s} \eta_{x,k} (1 - \eta_{x,k}) (1 - \eta_{y,k})^2 \eta_{z,k}^2, \tag{A.30}$$

$$p_{21} = \frac{1}{N_s} \sum_k^{N_s} (1 - \eta_{x,k})^2 \eta_{y,k} (1 - \eta_{y,k}) \eta_{z,k}^2, \tag{A.31}$$

$$p_{22} = \frac{1}{N_s}\sum_{k}^{N_s} \eta_{x,k}(1-\eta_{x,k})\eta_{y,k}(1-\eta_{y,k})\eta_{z,k}^2, \tag{A.32}$$

$$p_{23} = \frac{1}{N_s}\sum_{k}^{N_s} \eta_{x,k}^2(1-\eta_{y,k})^2\eta_{z,k}^2, \tag{A.33}$$

$$p_{24} = \frac{1}{N_s}\sum_{k}^{N_s} \eta_{x,k}^2\eta_{y,k}(1-\eta_{y,k})\eta_{z,k}^2, \tag{A.34}$$

$$p_{25} = \frac{1}{N_s}\sum_{k}^{N_s} (1-\eta_{x,k})^2\eta_{y,k}^2\eta_{z,k}^2, \tag{A.35}$$

$$p_{26} = \frac{1}{N_s}\sum_{k}^{N_s} \eta_{x,k}(1-\eta_{x,k})\eta_{y,k}^2\eta_{z,k}^2, \tag{A.36}$$

$$p_{27} = \frac{1}{N_s}\sum_{k}^{N_s} \eta_{x,k}^2\eta_{y,k}^2\eta_{z,k}^2. \tag{A.37}$$

Now we let $\eta_{x,k} \to 0$ for all k for the consideration of the worst case. Then, $p_2 = p_4 = p_6 = p_8 = p_9 = p_{10} = p_{11} = p_{12} = p_{14} = p_{16} = p_{17} = p_{18} = p_{20} = p_{22} = p_{23} = p_{24} = p_{26} = p_{27} = 0$ and $p_1, p_3, p_5, p_7, p_{13}, p_{15}, p_{19}, p_{21}, p_{25}$ can be well approximated by integrals,

$$p_1 \simeq \int_0^1\int_0^1 (1-\eta_y)^2(1-\eta_z)^2\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{9}, \tag{A.38}$$

$$p_3 \simeq \int_0^1\int_0^1 \eta_y(1-\eta_y)(1-\eta_z)^2\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{18}, \tag{A.39}$$

$$p_5 \simeq \int_0^1\int_0^1 (1-\eta_y)^2\eta_z(1-\eta_z)\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{18}, \tag{A.40}$$

$$p_7 \simeq \int_0^1\int_0^1 \eta_y(1-\eta_y)\eta_z(1-\eta_z)\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{36}, \tag{A.41}$$

$$p_{13} \simeq \int_0^1\int_0^1 \eta_y^2(1-\eta_z)^2\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{9}, \tag{A.42}$$

$$p_{15} \simeq \int_0^1\int_0^1 \eta_y^2\eta_z(1-\eta_z)\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{18}, \tag{A.43}$$

$$p_{19} \simeq \int_0^1\int_0^1 (1-\eta_y)^2\eta_z^2\,\mathrm{d}\eta_y\,\mathrm{d}\eta_z = \frac{1}{9}, \tag{A.44}$$

$$p_{21} \simeq \int_0^1 \int_0^1 \eta_y(1-\eta_y)\eta_z^2 \, \mathrm{d}\eta_y \, \mathrm{d}\eta_z = \frac{1}{18}, \tag{A.45}$$

$$p_{25} \simeq \int_0^1 \int_0^1 \eta_y^2 \eta_z^2 \, \mathrm{d}\eta_y \, \mathrm{d}\eta_z = \frac{1}{9}. \tag{A.46}$$

The system (Eq. (A.10)) then reduces to Eq. (32).

Appendix B. Stability analysis for the third-order Runge–Kutta scheme

For the third-order low-storage Runge–Kutta scheme, the discretized equations for the three substeps are

$$u_{n+1/3} - u_n = \frac{8}{15}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/3}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{B.1}$$

$$u_{n+2/3} - u_{n+1/3} = \frac{5}{12}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+2/3}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/3}\right) - \frac{17}{60}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1/3}} u(t')\,\mathrm{d}t' + \beta' u_n\right), \tag{B.2}$$

$$u_{n+1} - u_{n+2/3} = \frac{3}{4}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+1}} u(t')\,\mathrm{d}t' + \beta' u_{n+2/3}\right) - \frac{5}{12}\left(\frac{\alpha'}{\Delta t}\int_0^{t_{n+2/3}} u(t')\,\mathrm{d}t' + \beta' u_{n+1/3}\right), \tag{B.3}$$

where $u_{n+1/3}$ and $u_{n+2/3}$ denote function values at the first and second substeps, respectively. $t_{n+1/3} = t_n + \frac{8}{15}\Delta t$, $t_{n+2/3} = t_n + \frac{2}{3}\Delta t$. Subtracting the same equations at the previous time step from the above equations yields

$$u_{n+1/3} - u_{n-2/3} - u_n + u_{n-1} = \frac{8}{15}\Big(\alpha' I_1 + \beta'(u_n - u_{n-1})\Big), \tag{B.4}$$

$$u_{n+2/3} - u_{n-1/3} - u_{n+1/3} + u_{n-2/3} = \frac{5}{12}\Big(\alpha' I_2 + \beta'(u_{n+1/3} - u_{n-2/3})\Big) - \frac{17}{60}\Big(\alpha' I_1 + \beta'(u_n - u_{n-1})\Big), \tag{B.5}$$

$$u_{n+1} - u_n - u_{n+2/3} + u_{n-1/3} = \frac{3}{4}\Big(\alpha' I_3 + \beta'(u_{n+2/3} - u_{n-1/3})\Big) - \frac{5}{12}\Big(\alpha' I_2 + \beta'(u_{n+1/3} - u_{n-2/3})\Big), \tag{B.6}$$

Here, integrals can be calculated as follows:

$$I_1 = \frac{1}{\Delta t}\int_{t_{n-2/3}}^{t_{n+1/3}} u(t')\,\mathrm{d}t' = \left(\frac{2}{15}u_{n-2/3} + \frac{1}{3}u_{n-1/3} + \frac{8}{15}u_n\right), \tag{B.7}$$

$$I_2 = \frac{1}{\Delta t}\int_{t_{n-1/3}}^{t_{n+2/3}} u(t')\,\mathrm{d}t' = \left(\frac{1}{3}u_{n-1/3} + \frac{8}{15}u_n + \frac{2}{15}u_{n+1/3}\right), \tag{B.8}$$

$$I_3 = \frac{1}{\Delta t}\int_{t_n}^{t_{n+1}} u(t')\,\mathrm{d}t' = \left(\frac{8}{15}u_n + \frac{2}{15}u_{n+1/3} + \frac{1}{3}u_{n+2/3}\right). \tag{B.9}$$

Substituting $u_n = r^n$, $u_{n+1/3} = pr^n$, $u_{n+2/3} = qr^n$ into Eqs. (B.4)–(B.6) and rearranging yield

$$(p-1)(r-1) = \frac{8}{15}\alpha'\left(\frac{2}{15}p + \frac{1}{3}q + \frac{8}{15}r\right) + \frac{8}{15}\beta'(r-1), \tag{B.10}$$

$$(q-p)(r-1) = \frac{5}{12}\alpha'\left(\frac{1}{3}q + \frac{8}{15}r + \frac{2}{15}pr\right) + \frac{5}{12}\beta' p(r-1) - \frac{17}{60}\alpha'\left(\frac{2}{15}p + \frac{1}{3}q + \frac{8}{15}r\right) - \frac{17}{60}\beta'(r-1), \tag{B.11}$$

$$(r-q)(r-1) = \frac{3}{4}\alpha'\left(\frac{8}{15}r + \frac{2}{15}pr + \frac{1}{3}qr\right) + \frac{3}{4}\beta' q(r-1) - \frac{5}{12}\alpha'\left(\frac{1}{3}q + \frac{8}{15}r + \frac{2}{15}pr\right) - \frac{5}{12}\beta' p(r-1). \tag{B.12}$$

Using the condition that abs$(r) \leqslant 1$, we can obtain the stability condition. We utilized *Mathematica* to get r which is generally complex and enforced $|r| = 1$ to obtain a relation between α and β. The neutral lines drawn in Fig. 4 are combination of the following curves.

$$\alpha' = \frac{3}{160}\left(-917 - 330\beta' \pm (120889 + 245220\beta' - 11100\beta'^2)^{1/2}\right) \tag{B.13}$$

and

$$\alpha' = -\frac{1}{480}(5927 + 1980\beta') + \frac{1}{480c}(8820529 + 9769320\beta' + 680400\beta'^2) + \frac{c}{480} \tag{B.14}$$

with

$$c = \left(-30300346583. - 36710603460\beta' - 14206460400\beta'^2 - 2805192000\beta'^3 + 3600\sqrt{3d}\right)^{1/3} \tag{B.15}$$

and

$$\begin{aligned} d = {} & 5963440614876. - 1428035711796\beta' - 12235153563481\beta'^2 - 1828981759860\beta'^3 \\ & + 5162616348300\beta'^4 + 1701022356000\beta'^5 + 194293080000\beta'^6. \end{aligned} \tag{B.16}$$

References

[1] D.J. Bergstrom, M.F. Tachie, R. Balachandar, Application of power laws to low Reynolds number boundary layers on smooth and rough surfaces, Phys. Fluids 13 (2001) 3277.

[2] R. Bouard, M. Coutanceau, The early stage of development of the wake behind an impulsively started cylinder for $40 < Re < 10^4$, J. Fluid Mech. 101 (1980) 583.

[3] P. Cherukat, Y. Na, T.J. Hanratty, Direct numerical simulation of a fully developed turbulent flow over a wavy wall, Theoret. Comput. Fluid Dyn. 11 (1998) 109.

[4] E.A. Fadlun, R. Verzicco, P. Oralndi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 35, doi:10.1006/jcph.2000.6484.

[5] D. Goldstein, R. Handler, L. Sirovich, Modeling a no-slip flow Boundary with an external force field, J. Comput. Phys. 105 (1993) 354.

[6] D. Goldstein, R. Handler, L. Sirovich, Direct numerical simulation of turbulent flow over a modelled riblet covered surface, J. Fluid Mech. 302 (1995) 333.

[7] D. Goldstein, T.-C. Tuan, Secondary flow induced by riblets, J. Fluid Mech. 363 (1998) 115.
[8] J. Kim, D. Kim, H. Choi, An immersed-boundary finite-volume method for simulations of flow in complex geometries, J. Comput. Phys. 171 (2001) 132, doi:10.1006/jcph.2001.6778.
[9] P.-A. Krogstad, R.A. Antonia, L.W.B. Browne, Comparison between rough- and smooth-wall turbulent boundary layers, J. Fluid Mech. 245 (1992) 599.
[10] P.-A. Krogstad, R.A. Antonia, Surface roughness effects in turbulent boundary layers, Exp. Fluids 27 (1999) 450.
[11] M.-C. Lai, C.S. Peskin, An immersed boundary method with formal second-order accuracy and reduced numerical viscosity, J. Comput. Phys. 160 (2000) 705, doi:10.1006/jcph.2000.6483.
[12] R.J. Leveque, Z. Li, Immersed interface methods for Stokes flow with elastic boundaries or surface tension, SIAM J. Sci. Comput. 18 (3) (1997) 709.
[13] Z. Li, M.-C. Lai, The immersed interface method for the Navier-Stokes Equations with singular forces, J. Comput. Phys. 171 (2001) 822, doi:10.1006/jcph.2001.6813.
[14] A. Lundbladh, S. Berlin, M. Skote, C. Hildings, J. Choi, J. Kim, D.S. Henningson, An efficient spectral method for simulation of incompressible flow over a flat plate, Technical Report 1999:11, Royal Institute of Technology, Sweden, 1999.
[15] Y. Miyake, K. Tsujimoto, M. Nakaji, Direct numerical simulation of rough-wall heat transfer in a turbulent channel flow, Int. J. Heat Fluid Flow 22 (2001) 237.
[16] C.S. Peskin, Numerical analysis of blood flow in the heart, J. Comput. Phys. 25 (1977) 220.
[17] A.M. Roma, C.S. Peskin, M.J. Berger, An adaptive version of the immersed boundary method, J. Comput. Phys. 153 (1999) 509, doi:jcph.1999.6293.
[18] E.M. Saiki, S. Biringen, Numerical simulation of a cylinder in uniform flow: application of a virtual boundary method, J. Comput. Phys. 123 (1996) 450.
[19] E.M. Saiki, S. Biringen, Spatial numerical simulation of boundary layer transition: effects of a spherical particle, J. Fluid Mech. 345 (1997) 133.
[20] H. Schlichting, Boundary Layer Theory 7/e (1979) 616.
[21] H.S. Udaykumar, R. Mittal, P. Rampunggoon, A. Khanna, A sharp interface Cartesian grid method for simulating flow with complex moving boundaries, J. Comput. Phys. 174 (2001) 345, doi:10.1006/jcph.2001.6916.
[22] F.M. White, Viscous Fluid Flow (1974) 490.
[23] T. Ye, R. Mittal, H.S. Udaykumar, W. Shyy, An accurate Cartesian grid method for viscous incompressible flow with complex immersed boundaries, J. Comput. Phys. 156 (1999) 209, doi:jcph.1999.6356.