# Development of immersed boundary methods for complex geometries

**By J. Mohd-Yusof**

## 1. Motivation and objectives

For fluid dynamics simulations, the primary issues are accuracy, computational efficiency, and the ability to handle complex geometries. Spectral methods offer the highest accuracy but are limited to relatively simple geometries. In order to accommodate more complex geometries, finite-difference or finite-element methods are generally used. However, these methods suffer from relatively low accuracy, requiring fine meshes to obtain good results. Finite element schemes, while able to handle complex geometries, often require significant computational time for grid generation. Spectral element methods can be used for complex geometries, but the grid stretching inherent in these methods leads to time-step limitations and clustering of grid-points in an inefficient manner.

In general, any computational scheme which requires regridding to accommodate changes in geometry will incur significant penalties in simulating time-varying geometries. For relatively simple motions, it is possible to use grid-stretching techniques (Carlson *et al.*, 1995), but these are still slow. Vortex element methods for moving bodies (Koumoutsakos, 1995) are presently under development but are also rather slow, especially with respect to calculation of spectra.

In Mohd-Yusof (1997) we demonstrated a discrete-time immersed boundary method which allows implementation of complex geometries in existing pseudo-spectral codes. The method does not incur significant additional cost as compared to the base computational scheme and changes in surface geometry simply require modification of the input files without any further modification of the code itself. Although the method appears to work well in this implementation, the actual convergence properties are not well documented. Also, the earlier code did not properly accommodate moving boundaries. Finally there was a term omitted from the derivation of the forcing presented in that report. While that omission does not affect the results from the earlier work, it would invalidate computations in more complex surface geometries.

## 2. Accomplishments

The accomplishments are presented in three sections. First, the correct form of the forcing function is derived and implemented in the B-spline/Fourier code. This accounts for the divergence of the force which was omitted in the previous derivation. Second, the method is extended to accommodate moving boundaries and coupled to a genetic algorithm. In order to minimize the effect of interpolation required by the moving surface geometry, high order interpolation methods are implemented in the B-spline direction only. No interpolation is performed in the Fourier direction:

i.e. the forcing points are constrained to lie on collocation lines. Preliminary results are presented to demonstrate the degree of drag modification which is possible with moving-bump type actuators. Finally, a study is presented which demonstrates the effect of smoothing the forcing profile on the convergence of the results.

### *2.1 Immersed boundary concept*

We begin with an examination of the continuous (in time and space) Navier-Stokes equations to demonstrate the principle of the immersed boundary technique. We consider incompressible flows governed by the Navier–Stokes equations, including the body force term:

$$\frac{\partial \mathbf{u}}{\partial t} = -\mathbf{H} - \nabla P + \frac{1}{Re}\nabla^2 \mathbf{u} + \mathbf{F} \tag{1}$$

and the continuity equation:

$$\nabla \cdot \mathbf{u} = 0 \tag{2}$$

where $Re$ is the Reynolds number, $\mathbf{u} = (u, v, w)$ is the velocity vector, $\mathbf{H} = \mathbf{u} \times \omega = (H_u, H_v, H_w)$ is the convective term, and $\mathbf{F} = (F_u, F_v, F_w)$ is the forcing vector.

The full Navier-Stokes equations allow the inclusion of an external body force. In incompressible flows, this force is generally assumed to derive from some potential field (e.g. gravity) which is constant and therefore may be neglected. However, the NS equations themselves allow the force to be a function of both time and space. In that event, the divergence of the force may be non-zero and, therefore, must be included in the Poisson equation for pressure if that equation is used to solve the system.

The immersed boundary method involves specifying the body force term in such a way as to simulate the presence of a flow boundary within the computational domain without altering the computational grid. The advantage of this is that bodies of almost arbitrary shape can be added without grid restructuring, a procedure which is often time-consuming. Furthermore, multiple bodies may be simulated, and relative motion of those bodies may be accomplished at reasonable computational cost.

The concept of the immersed boundary technique has been used for pseudo-spectral simulations of flows in complex geometries (Goldstein *et al.* 1995). However, the timestep restriction imposed by their derivation severely limits the applicability of the method to turbulent and other strongly time-dependent flows. This restriction can be removed by the use of a *discrete-time* derivation of the forcing value (Mohd-Yusof 1996). When combined with appropriate choice of internal boundary conditions, this scheme leads to a forcing scheme which does not require any filtering of the forcing field.

A second issue of importance to the immersed boundary method is the ability of the underlying numerical scheme to place a sufficient number of grid-points near the immersed boundary to adequately resolve the flow scales in that region. While the grid geometry may be considerably simplified as compared to a body-fitted grid, there is still a fundamental need to tailor the grid-point distribution to the underlying flow scales. To this end, we employ a B-spline formulation, which allows flexibility of grid-point distribution, zonal embedded grids, and high accuracy

(Kravchenko *et al.* 1996). Coupled with Fourier-pseudo-spectral methods, this yields a numerical scheme which allows simulation of flows in complex geometries on Cartesian grids with high accuracy.

### *2.1 Numerical method*

We now consider the discrete-time Navier-Stokes equations in general form:

$$\frac{\mathbf{u}^{n+1}-\mathbf{u}^n}{\triangle t} = -\mathbf{H} - \nabla P + \frac{1}{Re}\nabla^2\mathbf{u} + \mathbf{F} \tag{3}$$

We wish to drive the velocity, $\mathbf{u}$, on some surface, $\Omega$, to some desired value, $\mathbf{v}(\Omega)$. Rearrangement of the discrete NS equation gives us the velocity update equation which is of the form:

$$\mathbf{u}^{n+1} = \mathbf{u}^n + \triangle t(-\mathbf{H} - \nabla P + \frac{1}{Re}\nabla^2\mathbf{u} + \mathbf{F}) \tag{4}$$

If we know $\mathbf{H}$, $\nabla P$, and $\nabla^2\mathbf{u}$, then the forcing term is simply:

$$\mathbf{F} = \begin{cases} \mathbf{H} + \nabla P - \dfrac{1}{Re}\nabla^2\mathbf{u} + \dfrac{1}{\triangle t}(\mathbf{v}-\mathbf{u}^n), & \text{on } \Omega; \\ 0 & \text{elsewhere.} \end{cases} \tag{5}$$

#### *2.1.1 Velocity-vorticity formulation*

Following the same approach as in Kim *et al.* (1991), one can reduce Eqs. (1) and (2) to a fourth-order equation for $v$ and a second-order equation for the normal component of vorticity $g$:

$$\frac{\partial}{\partial t}\nabla^2 v = h_v + \frac{1}{Re}\nabla^4 v + f_v \tag{6}$$

$$\frac{\partial}{\partial t} g = h_g + \frac{1}{Re}\nabla^2 g + f_g \tag{7}$$

$$p + \frac{\partial v}{\partial y} = 0 \tag{8}$$

where

$$p = \frac{\partial u}{\partial x} + \frac{\partial w}{\partial z}, \quad g = \frac{\partial u}{\partial z} - \frac{\partial w}{\partial x} \quad f_g = \frac{\partial F_u}{\partial z} - \frac{\partial F_w}{\partial x} \tag{9}$$

$$h_v = -\frac{\partial}{\partial y}\left(\frac{\partial H_u}{\partial x} + \frac{\partial H_w}{\partial z}\right) + \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial z^2}\right)H_v \tag{10}$$

$$f_v = -\frac{\partial}{\partial y}\left(\frac{\partial F_u}{\partial x} + \frac{\partial F_w}{\partial z}\right) + \left(\frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial z^2}\right)F_v \tag{11}$$

$$h_g = \frac{\partial H_u}{\partial z} - \frac{\partial H_w}{\partial x} \tag{12}$$

Note especially that the force in both the equations is transformed in exactly the same manner as the nonlinear terms. In this formulation the velocity field is assumed divergence-free; however, the force is not in general solenoidal, so the projection of the force onto a solenoidal field must be accomplished in the same manner as with the term $h_v$ in Eq. (10). All further numerical details of the code are unchanged from those presented in Mohd-Yusof (1997) and are omitted for the sake of brevity.

### 2.2 Turbulent channel simulation

The immersed boundary method allows a wide range of different surface geometries to be simulated with a single code. This flexibility makes it uniquely suited to coupling with optimization schemes for drag reduction in turbulent flows. Once the parameter space is chosen (which is not necessarily obvious) the code can be used to simulate the flow in question. The results of the simulation can then be fed back into a genetic algorithm (e.g, Koumoutsakos, 1997) to determine the optimal geometry for achieving the desired goal. We envision such a combined simulation/optimization tool as not only a diagnostic tool to explore the performance potential of current surface actuators, but also as a way to design surface actuators which may not yet have been built. To this end, the test case we have chosen for the moving boundary simulation is that of a turbulent channel with moving-bump actuators on one wall.

In this case, the optimization parameters could be the bump height, spacing, streamwise and spanwise extent, and the period of bump motion. For our preliminary tests we fix all but the last parameter and measure the surface drag for varying bump periods.

#### 2.2.1 Computational mesh

In order to simulate a turbulent channel using the immersed boundary, it is necessary to use computational mesh which is refined near the location of the immersed boundary. In this instance we use a double-cosine stretched grid as shown in Fig. 1. This grid provides the same near-wall resolution as a traditional cosine stretched grid in a conventional calculation. The computational domain extends from $-1$ to 1. The mean location of the immersed boundary wall is fixed at $-0.8$ for all simulations.

For simplicity, the forcing function for the moving bumps is computed on an embedded auxiliary mesh which is refined in the wall normal direction near the location of the bumpy wall, $y = -0.8$. The resulting force is interpolated, using b-splines, to the solution mesh. The spacing of the mesh is identical to the base mesh in the two Fourier directions. In the wall-normal direction, the mesh spacing is approximately equal to the finest spacing on the cosine stretched grid.

#### 2.2.2 Results

Figure 2 shows the drag on the bumpy wall, as a function of time, from the simulation compared with the flat wall case. In all cases the bumpy wall was of the type shown in Fig. 1; four bumps operating in two sets with opposite phase. The bump shape is given by $h(r) = 1 + cos(2\pi r/r_{max})$ for $r < r_{max}$. The height of the bumps varies sinusoidally in time with period $t_{bump}$ (all times are nondimensionalized by the friction velocity and the channel half height). For the cases shown the simulations are started using identical fully developed turbulent channel flow data obtained by running the code with a flat wall immersed boundary. For all the bump frequencies tested, the mean wall drag is slightly reduced by the introduction of the bumps. However, the duration of the simulations performed to date is too short to provide a meaningful result.

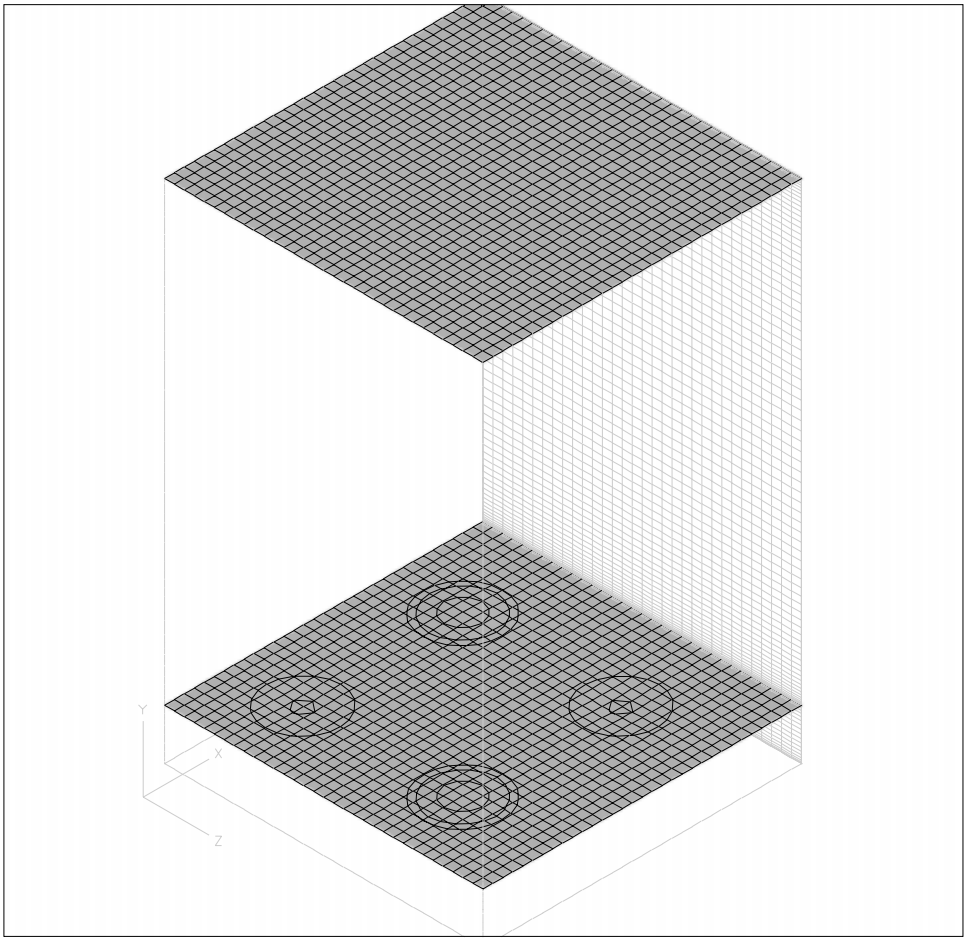

FIGURE 1. Representative sketch of the computational domain. The stretched solution grid is shown, and a representative isosurface showing the location of the immersed boundary.

Flow visualizations of the bumpy-wall results indicates the presence of large low-speed structures which span the entire streamwise extent of the computational domain. These structures seem to be present in all the cases tested, and the structures are centered over the bumps themselves. Note that since the bump pairs are out of phase, the structures extend over one bump and the accompanying dip. The persistence of the structures over long times may also be an artifact of the periodic computational domain.

### *2.3 Convergence properties and smoothing*

There have been, to the author's knowledge, no rigorous proofs of the convergence properties of immersed boundary methods. Evidence suggests that the method of Goldstein *et al.* converges very slowly, if at all. The likely cause of this is the fact that the forcing is applied as a series of point forces at the collocation points. Thus, the force is effectively implemented as a series of $\delta$-functions with commensurately poor convergence properties. Goldstein *et al.* attempt to smooth the forcing by

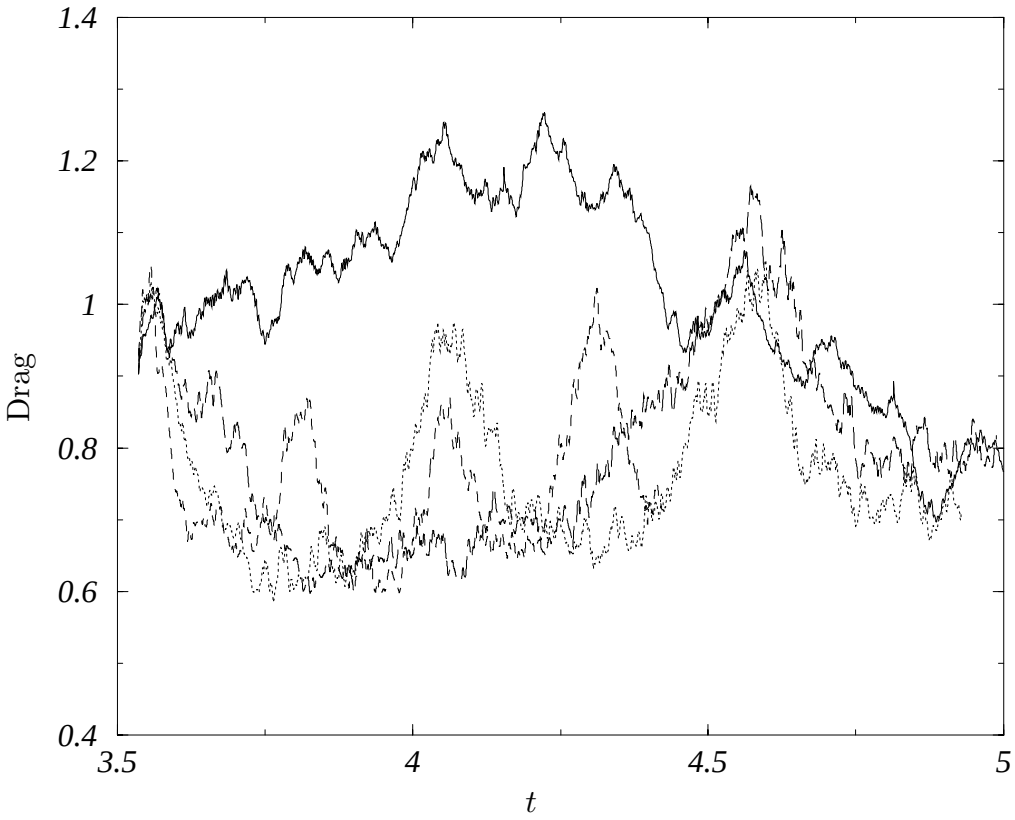


FIGURE 2. Drag traces for varying bump oscillation periods and the flat wall case. ——— flat wall case; ····· $t_{bump} = 1$; — — — $t_{bump} = 2$; - - - - $t_{bump} = 0.5$. All times are nondimensionalized by friction velocity and channel half height.

spreading the force as a Gaussian: this still converges slowly and furthermore allows the force to be non-zero in the solution region of interest.

In contrast, the current method attempts to smooth the force by utilizing an internal boundary layer, confined to the region of the solution which is not part of the flow itself. In this section we will examine the effect of modifying the specification in the force within this internal layer in order to improve the convergence properties of the global solution. Intuitively, the rate at which the solution converges will depend on our ability to enforce the boundary condition with a forcing function which has as narrow a spectrum as possible, confined to the lowest practical wavenumbers. This will be the opposite of the delta function, which has equal energy at all resolved wavenumbers.

It is instructive to first examine the behavior of the body force in a simple test case, in the absence of any smoothing or interpolation error. The test case used here will be a simple Couette flow, simulated on a periodic domain. That is, we wish to enforce no-slip conditions on two walls, at $x = \pi/2$ and $x = 3\pi/2$, moving with velocity 1 and $-1$, respectively. The 'correct' solution in this case will be a linear velocity profile in the region $\pi/2 < x < 3\pi/2$, and the flow in the remainder of the domain is extraneous.

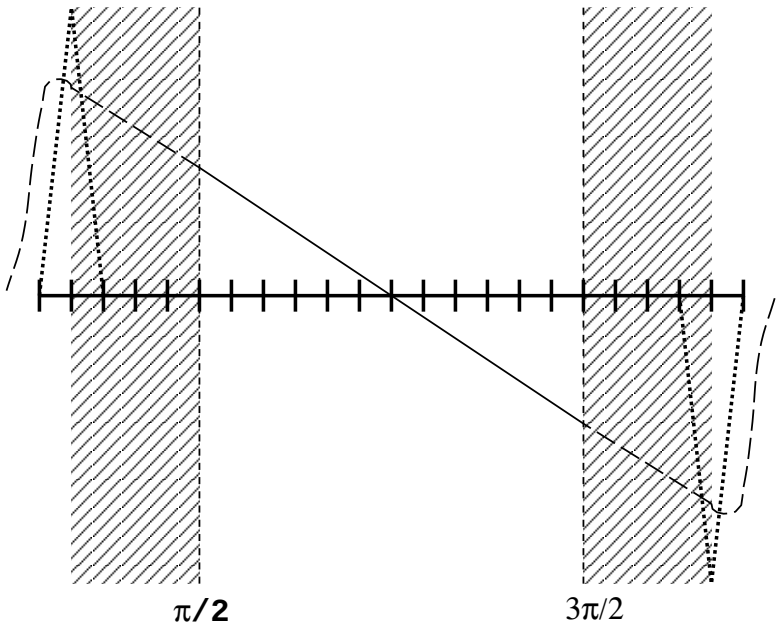


FIGURE 3. Representative sketch of the test case considered showing a sample solution grid. The hatched areas represent the internal boundary layers where the flow reversal is applied. ——— desired solution, – – – – extraneous solution, ····· the required force distribution on this grid, without smoothing.

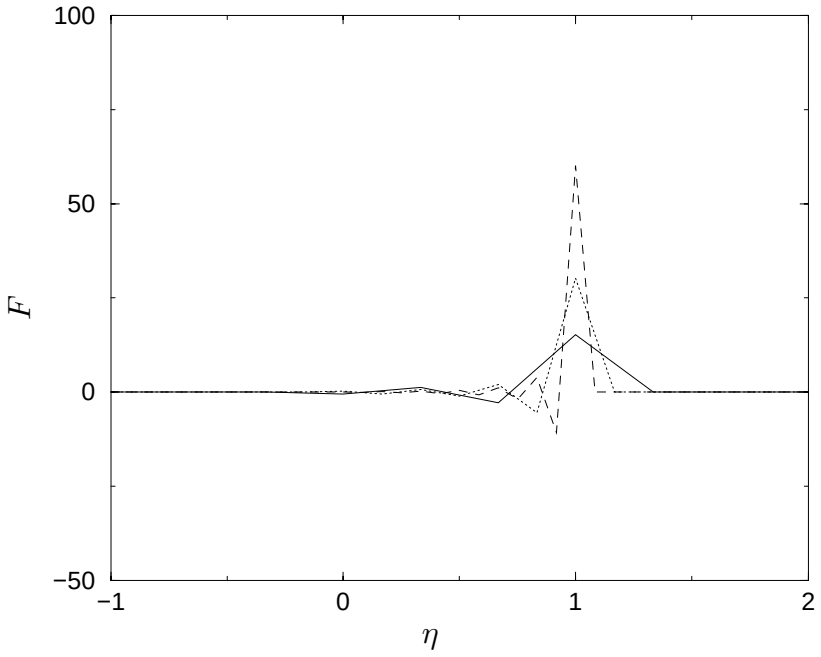


FIGURE 4. Distribution of the force without any smoothing for simulations using ——— 48, ····· 96 and – – – – 192 points.

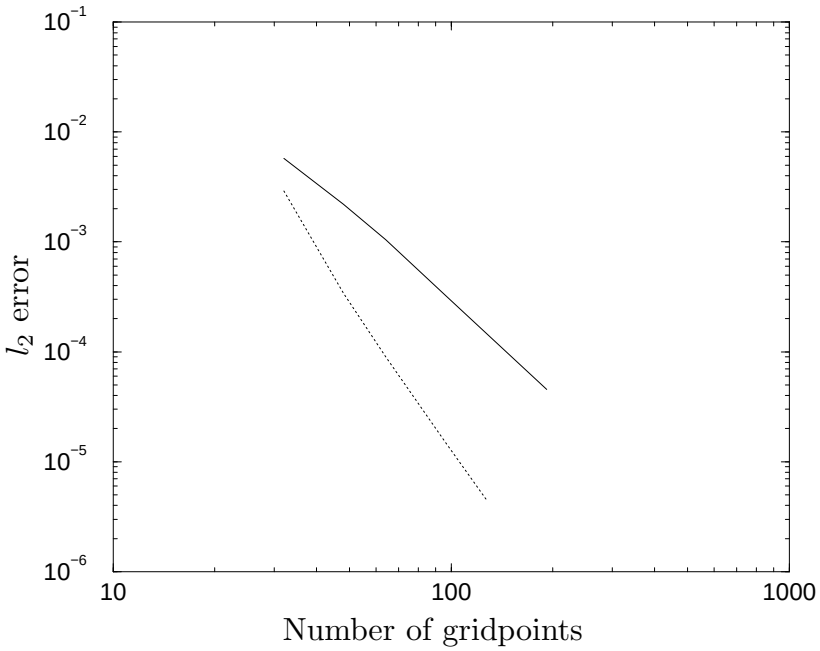


FIGURE 5. Comparison of the convergence rates for the smoothing functions ——— $S(\eta) = 1 - \eta$ and - - - - $S(\eta) = (1 + cos(\eta))/2$, which yield 3rd and 5th order convergence, respectively. Since the choice of error norm does not change the slope, only the $l_2$ data is shown for each case.

Since our forcing scheme requires an internal boundary layer to be formed in the body, we require at a minimum 3 points in this extraneous region to accommodate these two boundary layers (which will have opposite velocities as sketched in Fig. 3). The force distribution required to achieve the desired solution in this case is essentially a pair of $\delta$-functions located at the single collocation point at the edge of the internal BL; the body force applied at the surface is near zero. The vanishing force at the surface is an expected result upon examination of the force specification equation; once the velocity at the surface is small, all the terms vanish except the viscous term, which is driven to zero by the presence of the boundary layer.

If we fix the geometric thickness of this internal boundary layer and perform the simulation on progressively refined grids, then in the absence of any smoothing, the force distribution essentially remains a pair of $\delta$-functions, with increasingly compact support, located at the edge of the internal BL (Fig. 4). As will be shown later, the force is uniformly distributed in spectral space, and thus the convergence of the scheme is poor.

The goal of the smoothing function, it would then appear, is to distribute the force within this internal boundary layer in such a way as to minimize the spectral bandwidth of the final forcing distribution while at the same time not compromising the local character and simplicity of the scheme.

We choose therefore to apply the force in the internal boundary layer in the following way: the force applied to any collocation point within the internal BL, located a distance $\eta$ from the desired wall location, will be the force required to

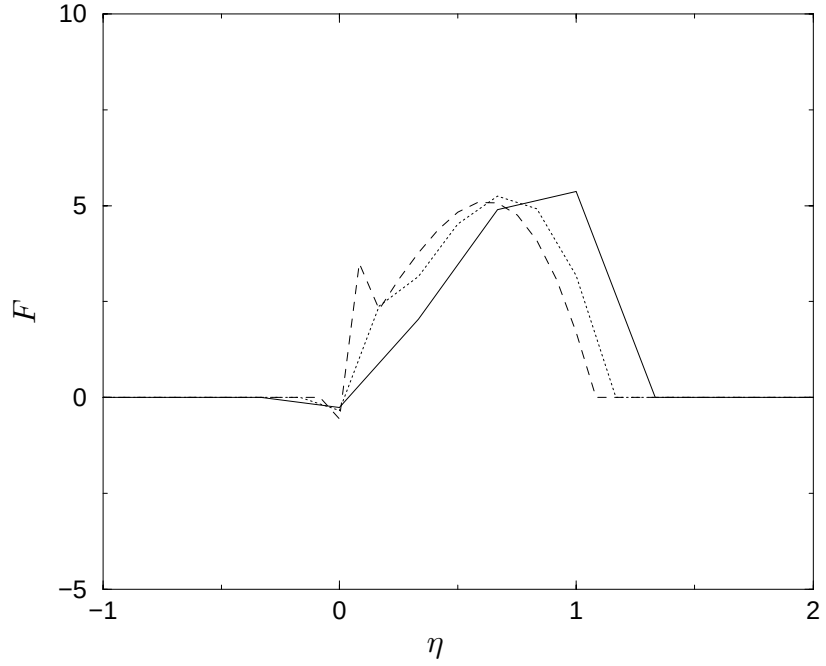


FIGURE 6. Distribution of the force with $S(\eta) = 1 - \eta$, for simulations using ——— 48, ····· 96 and - - - - 192 points.

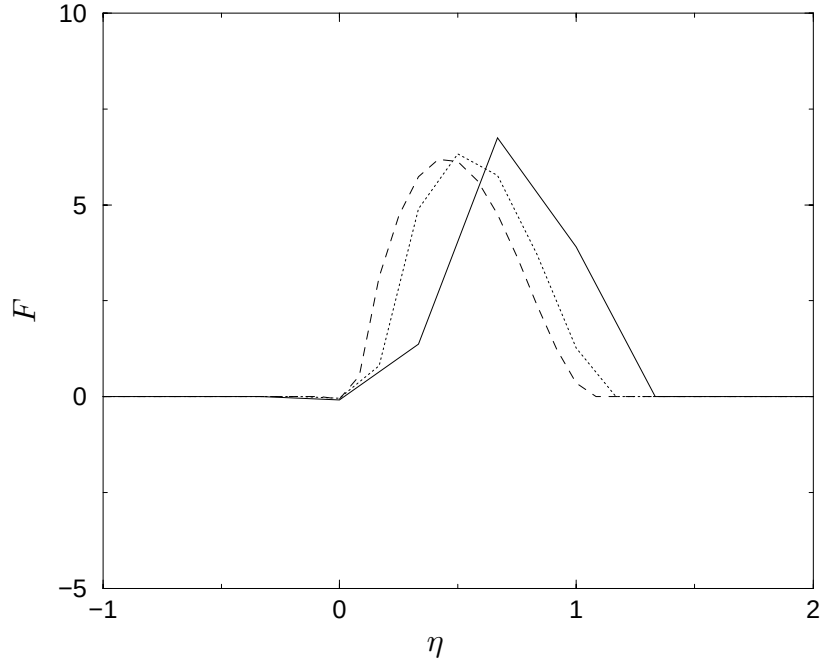


FIGURE 7. Distribution of the force with $S(\eta) = (1 + cos(\eta))/2$ for simulations using ——— 48, ····· 96 and - - - - 192 points.

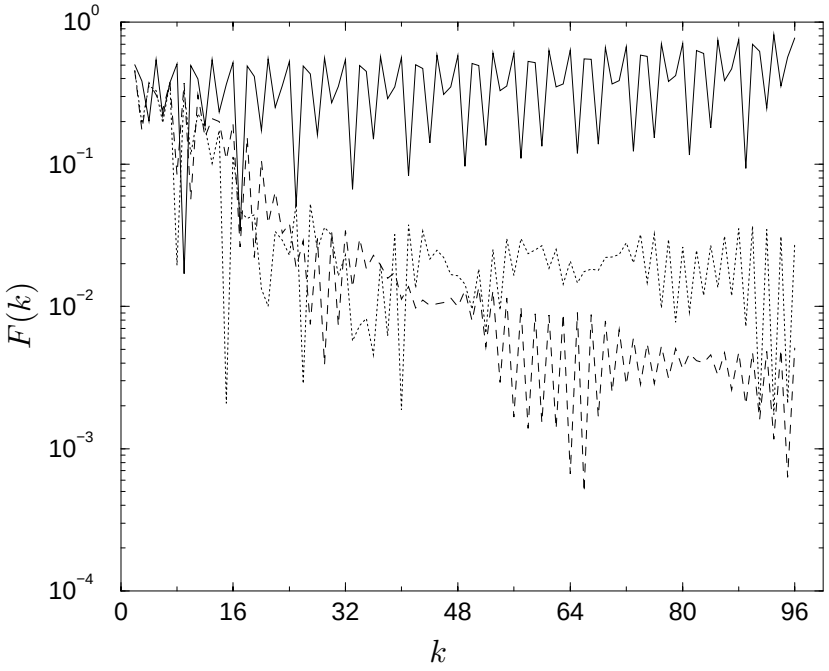


FIGURE 8. Spectra of the force distribution with various smoothing functions used: ——— unsmoothed, ····· , $S(\eta) = 1 - \eta$, - - - - $S(\eta) = (1 + cos(\eta))/2$. For clarity, only the spectra for 192-point simulations are shown.

reverse the velocity at the corresponding mirror-image point in the solution flow, $-\eta$, scaled by some smoothing function $S(\eta)$. In this case we scale $\eta$ such that $\eta = 0$ at the surface and $\eta = 1$ is the thickness of the internal boundary layer. For each case, we track the $l_1$,$l_2$, and $l_{inf}$ norms of the error in the 'true' solution region.

We examine some simple test cases for the force smoothing function $S(\eta)$. Figure 5 shows the convergence rate of the global error (within the domain of interest) for the smoothing functions $S(\eta) = 1 - \eta$, and $S(\eta) = (1 + cos(\eta))/2$. We choose to show only the curves for the $l_2$ error since the choice of norm does not change the slope. The convergence rates for the two smoothing functions are 3rd and 5th order, respectively.

Figures 6 and 7 show the force distributions for the choices of smoothing function $S(\eta) = 1 - \eta$ and $S(\eta) = (1 + cos(\eta))/2$, respectively. Note that in both cases the magnitude of the force has been reduced by an order of magnitude compared to the unsmoothed case.

Our supposition that the reduced spectral bandwidth of the resultant force distribution is responsible for the improved convergence is demonstrated in Fig. 8. The marked reduction in spectral bandwidth of the various smoothed forcing distributions can clearly be seen. Note that the magnitude of the even-numbered wavenumbers is reduced due to the odd symmetry of the problem chosen.

There are several important points to note in these simplified tests. First, the Couette flow case does not require the forcing function to impose no-penetration at the solid surface, only no-slip. The imposition of no-penetration may considerably

alter the convergence properties of the method. Secondly, the convergence properties of the steady state solution may not translate into similar properties for the time-dependent solution. Thirdly, the 'optimal' smoothing function, if one exists, is that which results in the narrowest spectral bandwidth of the forcing field, which is itself a function of the boundary layer profile in the 'external' flow. Thus, the optimal solution for a Couette flow may be different than that for a channel flow, for example. Since we would prefer the forcing scheme to be independent of the flow geometry, further refinement of the smoothing function may be of dubious benefit.

## 3. Future plans

The primary obstacle to obtaining useful results from the combined optimization/simulation scheme is the slow turnaround time for the simulations themselves. The code is being rewritten to improve the computational efficiency, primarily via the implementation of collocation methods for the quadratic terms. An interim solution has been to implement the immersed boundary method in the finite difference code (Lund *et al.*, 1995), which is considerably faster although formally less accurate. The fact that this implementation required less than one day of work underscores the flexibility of the immersed boundary method itself and the ease with which it can be incorporated into existing flow solvers. This method has also been implemented in finite difference LES codes and has proved to be very efficient (Verzicco *et al.*, 1998).

The question of the convergence of the forcing scheme will also be investigated further. There has been, to the author's knowledge, no formal proof of the convergence properties of such schemes in general. For our purposes, it should suffice to continue numerical experiments to attempt to improve on the 5th order convergence of the smoothing demonstrated in this report.

## REFERENCES

Carlson, H. A., Berkooz G. & Lumley J. L. 1995 Direct numerical simulation of flow in a channel with complex, time-dependent wall geometries: A pseudospectral method. *J. Comp. Phys.* **121,** 155-175.

Goldstein, D., Handler R. & Sirovich L. 1993 Modeling a no-slip flow boundary with an external force field. *J. Comp. Phys.* **105,** 354-366.

Koumoutsakos, P. 1995 Fast multipole methods for three-dimensional N-body problems. *Annual Research Briefs 1995*, Center for Turbulence Research, NASA Ames/Stanford Univ., 377-390.

Koumoutsakos, P. 1997 Active Control of Vortex-Wall Interactions. *Phys. of Fluids A.* **9,** 3808-3816.

Kravchenko, A. G., Moin, P. & Moser, R. 1996 Zonal embedded grids for numerical simulations of wall-bounded turbulent flows. *J. Comp. Phys.* **127,** 412-423.

Lund, T.S. & Kaltenbach, H.-J. 1995 Experiments with explicit filtering for LES using a finite difference method. *Annual Research Briefs 1995*, Center for Turbulence Research, NASA Ames/Stanford Univ., 91-105.

Mohd-Yusof, J. 1996 Interaction of massive particles with turbulence. *PhD thesis*, Cornell University.

Mohd-Yusof, J. 1997 Combined immersed-boundary/B-spline methods for simulations of flow in complex geometries. *CTR Annual Research Briefs*, Center for Turbulence Research, NASA Ames/Stanford Univ., 317-327.

Verzicco, R., Mohd-Yusof, J., Orlandi, P. & Haworth, D. 1998 LES in complex geometries using boundary body forces. *Proceedings of the Summer Program 1998*, Center for Turbulence Research, NASA Ames/Stanford Univ., 171-186.