Journal of Computational Physics 228 (2009) 1916–1931

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

A fast resurrected core-spreading vortex method with no-slip boundary conditions

Mei-Jiau Huang *, Huan-Xun Su, Li-Chieh Chen

Department of Mechanical Engineering, National Taiwan University, No. 1 Sec. 4 Roosevelt Road, Taipei 106, Taiwan, ROC

ARTICLE INFO

Article history:
Received 10 April 2008
Received in revised form 12 November 2008
Accepted 14 November 2008
Available online 7 December 2008

Keywords:
Core-spreading vortex method
Multipole method
Adaptive domain decomposition
No-slip boundary condition

ABSTRACT

To simulate two-dimensional viscous incompressible flows based on a scheme of blob splitting and merging, we developed a vortex method and employed a fast multipole method to speed the computation of velocities. The diffusion of the vortex sheet induced at a solid wall by the no-slip boundary conditions is first modeled according to the analytical solution of Koumoutsakos and then converted into discrete blobs in the vicinity of the wall. To prevent the vorticity from entering the solid body, we introduce a concept residual circulation in a sense that only a partial circulation of the vortex sheet is diffused into the flow field; the rest remains at the wall. Blobs near the wall are thus avoided. Blobs near the wall that might cause large fluctuations in the strength of the vortex sheet are handled similarly. The solver thus developed requires no grid-based remeshing. We applied this solver to simulate the flow induced with an impulsively initiated circular cylinder; the results agree satisfactorily with those of previous experimental and numerical investigations.

1. Introduction

According to the discrete vortex method [1] developed as a numerical simulator for two-dimensional incompressible flows, the convection of packets of vorticity (vortex elements) is tracked. The compactness of the representation of the vorticity field provides this vortex method with advantages over a representation involving primitive variables, beyond its exactly satisfying boundary conditions at infinity for external flows. Viscous effects are included on taking advantage of grid-based finite-difference methods (such as those described in [2–4]) that make the code no longer grid-free and might result in excessive interpolation errors. Among grid-free viscous solvers, the random-walk approach that Chorin proposed [5] added to the element velocity a pseudo-random velocity like Brownian motion. The particle-strength-exchange scheme [6] approximated parts of the governing equations with integral operators and redistributed the strength (circulation) among vortex elements to account for diffusion. Fishelov [7] proposed to apply the Laplace operator to convolute some truncation function with a Dirac delta distribution and to adjust the circulations of vortex elements to satisfy the vorticity equation. The resulting equations for the circulations of vortex elements of the above two methods are in general coupled, causing the methods to become inefficient. To decouple the circulation equations, the distribution of vortex elements must be highly uniform; these two methods thus encounter difficulties when the flow becomes highly strained, and many remeshing processes are required to maintain a uniform distribution of the vortex elements. This requirement increases the computational effort and prevents schemes from being entirely grid-free. Instead of approximating the Laplace operator with integral operators, the vortex-redistribution method that Shankar and van Dommelen proposed [8,9] distributes fractions of circulation of each vortex to its neighboring vortices within a chosen distance; the fractions are computed on equating the Fourier transforms of

* Corresponding author. Tel.: +886 2 33662696; fax: +886 2 23631755.
E-mail address: mjhuang@ntu.edu.tw (M.-J. Huang).

0021-9991/$ - see front matter © 2008 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2008.11.026

the exactly diffused vorticity and the vorticity resulting after the redistribution. Because a solution of the redistribution equations does not necessarily exist, new vortices with zero circulation must be added somewhere until a solution is found. For more details of these vortex methods, a reader is referred to work of Leonard [10], Anderson and Greengard [11], Gustavson and Sethian [12], Beale et al. [13] and Cottet and Koumoutsakos [14].

Our interest is the core-spreading vortex method that Leonard proposed [1]. The fundamental vortex elements chosen herein are Gaussian blobs. To capture correctly the diffusion phenomenon, the characteristic area of these Gaussian blobs must increase linearly in time. Its well-known convergence problem is remedied with a blob splitting technique first proposed by Rossi [15] and subsequently improved by Huang [16]. According to the latter treatment, splitting is viewed as part of the diffusion. Whenever a vortex blob becomes too fat, it discards some circulation, forming progeny vortices in its surroundings. The strengths and locations of these progeny vortices are determined on preserving the first few moments of vorticity. To control the total number of vortex blobs, one must merge similar and nearby vortices [17,18]. The simplified merging criteria and the cell technique that Huang [18] developed are useful and efficient. This resurrected core-spreading vortex method is attractive because it is deterministic, grid-free, and exact for uniform flow fields. As it does not rely upon operator splitting, it is possible to design methods to constrain the flow to satisfy simultaneously the boundary conditions of no slipping and no penetration, apart from being readily implemented and computationally efficient. Among all core-spreading methods employing axially symmetric Gaussians, Huang's approach [16,18] retains all these advantages and has superior accuracy [16]. This viscous solver is, however, not highly efficient because the velocities are evaluated directly according to the Biot–Savart law and are applicable to only flows without boundaries. Here we apply a fast multipole method to this resurrected core-spreading method, so to extend its capability to simulate flows with solid walls.

Of fast vortex methods of three kinds, the first maps vorticity values between Eulerian and Lagrangian grids and obtains the velocities on solving the Poisson equation [19–21]. A second method integrates numerically the Biot–Savart law with Legendre quadrature and time-changing quadrature nodes and weights [22,23]. A third method groups the influence of nearby vortices. According to the so-called multipole method that Carrier et al. proposed [24], the velocity induced by a group of nearby vortex elements is expanded into a finite multipole series. The associated coefficients are calculated, recorded, and applied whenever the velocity induced on a far-enough vortex element needs computing, according to a group-to-point method. Carrier et al. proposed also a group-to-group method [24] based on Taylor series; the computational work is further decreased to be linearly proportional to the total number of vortex elements. In his group-to-point method, Anderson [25] employed an exact solution of the Poisson equation instead of a multipole series. To match the blob splitting and merging technique and to keep the scheme grid-free, we combined the multipole method with the resurrected core-spreading vortex method.

The group-to-point formula thus developed is used to evaluate the influence of a group of nearby vortex elements on only those vortex elements sufficiently distant; velocities induced among nearby vortex elements are still calculated with the Biot–Savart law (the point-to-point formula). In deciding which method to apply, a domain decomposition is helpful. Because the distribution of vortex elements is scattered, adaptive decomposition of the flow domain is appropriate. For instance, Carrier et al. [24] divided a square domain into four equal parts whenever the number of vortex elements exceeded some limit; cell-to-cell relations in five classes were then identified among these squares. The involved recognition logic was complicated and thereby inefficient. Van Dommelen and Rundensteiner [26] abandoned the group-to-group technique, produced cell-to-cell relations in only two classes, and developed a triad numbering system that rapidly identified the cell-to-cell relations. Draghicescu [27] decomposed a rectangular domain into two equal parts along the longer side. Strain [23] decomposed a rectangular domain into two parts, but the decomposition was performed based on a principle that the numbers of vortex elements in the two subdomains are the same or differ by unity. According to both methods, one must occasionally compute the distance between the rectangular domains when summing the velocities. The outer-ring and inner-ring approximations that Anderson proposed [25] are simple and efficient in summing the velocities, but, as the associated radii of rings are undetermined or empirical, estimating the associated numerical error is difficult. For further information, a reader is referred to [28,29].

In our vortex method, a domain decomposition is required to merge similar and nearby vortex blobs [18]. For consistency, we decided to apply an adaptive domain decomposition and the group-to-point approximation that Carrier et al. proposed [24]. An algorithm similar to the outer-ring and inner-ring approximations was developed for the velocity summation. Specifically, to avoid use of the uncommon 3-based digits [26], we designed a binary numbering system to replace the triad system so as to decrease the memory requirement and for even more rapid identification.

When a solid wall is present in the flow, the potential flow theory [30] is applicable to determine the kind and strength of singularities that can be placed at the wall to satisfy a condition of no penetration. To accommodate the wall effect, Chorin adopted vortex sheets and the theory of the Prandtl boundary layer [5]; the tangential velocity induced by the vortex blobs and any external potential flow at the wall was balanced by that induced by the vortex sheet, and the circulation of the vortex sheet gradually diffused into the flow. The total circulation of the flow remained constant according to the Kelvin theorem [30]. Anderson et al. [31] employed a similar treatment in their finite-element solver, and Park et al. [19] combined it with a vortex-in-cell method. Rossi [32] divided the boundary into finite segments, each imposed with a line source to create the potential flow, and added vortex elements near the boundary segments to approximate a thin boundary layer; on adjusting the strengths of line sources and newly injected vortex elements, the flow conditions of no-slip and no penetration became satisfied concurrently.

We chose to place a vortex sheet at the wall and to solve its strength with a boundary-element method and constant panels [33]; we employed Koumoutsakos' analytical solution [34], properly discretized, to model the diffusion of the vortex sheet. A fast resurrected core-spreading vortex method applicable for flows with solid boundaries was targeted and tested with flow typical of an impulsively initiated circular cylinder.

After reviewing Leonard's original core-spreading vortex method and the blob splitting and merging technique, we propose a fast method and test its related algorithm. The boundary conditions of no-slip and no penetration are handled in Section 4, and the proposed scheme is applied to simulate the flow induced by an impulsively initiated circular cylinder. We then compare the simulation results with the literature and examine the accuracy of the proposed fast vortex method.

2. Resurrected core-spreading vortex method

The governing equation to be solved is

$$\frac{\partial\omega}{\partial t}+u\frac{\partial\omega}{\partial x}+v\frac{\partial\omega}{\partial y}=v\nabla^2\omega \tag{1}$$

in which appear vorticity ω, velocity $\vec{u}=(u,v)$ and viscosity v. The velocity is divided into two parts, one irrotational $\vec{u}_p$ and another rotational $\vec{u}_\omega$. According to Leonard's core-spreading method, the solution is decomposed into many discrete Gaussian blobs as follows:

$$\omega(\vec{x},t)=\sum_{j=1}^{N}\frac{\Gamma_j}{\pi\sigma_j^2}\exp\left(-\frac{|\vec{x}-\vec{x}_j|^2}{\sigma_j^2}\right) \tag{2}$$

in which appear location $\vec{x}_j$, core size σ_j and strength Γ_j of blob j; they are governed by

$$\frac{d\Gamma_j}{dt}=0 \tag{3a}$$

$$\frac{d\sigma_j^2}{dt}=4v \tag{3b}$$

$$\frac{d\vec{x}_j}{dt}=\vec{u}_p(\vec{x}_j,t)+\vec{u}_\omega(\vec{x}_j,t) \tag{3c}$$

According to the Biot–Savart law, the rotational velocity ($\vec{u}_\omega$) induced by the blobs is,

$$u_\omega(\vec{x}_j,t)=-\sum_{\substack{i=1\\ i\neq j}}^{N}\frac{\Gamma_i}{2\pi}\frac{(y_j-y_i)}{|\vec{x}_i-\vec{x}_j|^2}\left\{1-\exp\left(-\frac{|\vec{x}_i-\vec{x}_j|^2}{\sigma_{ij}^2}\right)\right\} \tag{4a}$$

$$v_\omega(\vec{x}_j,t)=\sum_{\substack{i=1\\ i\neq j}}^{N}\frac{\Gamma_i}{2\pi}\frac{(x_j-x_i)}{|\vec{x}_i-\vec{x}_j|^2}\left\{1-\exp\left(-\frac{|\vec{x}_i-\vec{x}_j|^2}{\sigma_{ij}^2}\right)\right\} \tag{4b}$$

with $\sigma_{ij}^2\equiv(\sigma_i^2+\sigma_j^2)/2$.

The diffusion of Eq. (1) is simulated exactly with Eq. (3a–c) but not convection; the associated error increases with increasing σ_j [10]. As remedy, the core size must be kept small. The idea of blob splitting, suggested by Rossi [15], is to replace a *too fat* blob by thinner ones. Huang's splitting technique [16] that improves the accuracy of Rossi's is described below. A threshold $\sigma_{\max}$ for the core size is set first; whenever a blob has a core size σ_p greater than $\sigma_{\max}$, this blob is replaced immediately by a thinner one surrounded with M others. The core size of these new blobs is chosen to be $\alpha\sigma_p$ and their strengths are determined on preserving the zeroth, second, and fourth moments of vorticity. If Γ_p, Γ'_p and Γ_c are the strengths of the original blob, the thinned blob and the surrounding blobs respectively, the preservation of moments of vorticity produces

$$\Gamma'_p=\Gamma_p/2 \tag{5a}$$

$$\Gamma_c=\Gamma_p/2M \tag{5b}$$

$$r^2/\sigma_p^2=2(1-\alpha^2) \tag{5c}$$

in which r is the distance between the original blob and the surrounding blobs. This splitting method differs from Rossi's mostly in the survival of the original blob. Leaving a blob of half the strength at the original location diminishes significantly the error due to splitting, particularly the error at the original location [16]. Physically, when diffusion is concerned, the vorticity at the original location decreases but always remains maximum. Huang's splitting method approximates this major part of the vorticity with a blob of half the strength and a circumferential vorticity with the surrounding blobs so that it captures the diffusion more accurately. Fig. 1(a) shows the maximum field error induced in one single splitting event against the number (M) of surrounding blobs. The original blob has a circulation $\Gamma_p=\pi$ and $\sigma_p=1$. As seen, the maximum field error associated with Rossi's method does not diminish with increasing M. The error associated with Huang's method decreases with increasing M but rapidly becomes saturated. Fig. 1(b) shows the maximum field error at $t=0.9$ in simulating a decaying

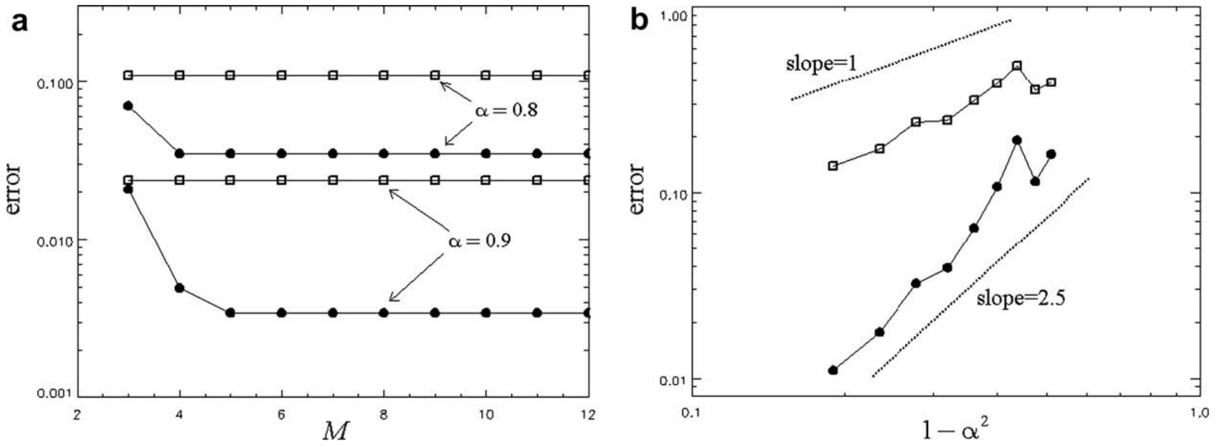


Fig. 1. Maximum field error associated with a single splitting event against M (a) and that in simulating a decaying Burger's vortex (σ_M = 0.4 and M = 6) (b). Solid circles are the results of Huang's method and the open squares are the results of Rossi's.

Burger's vortex. The initial flow consists of a single vortex element of core size 0.4 and circulation 5; the fluid viscosity is v = 0.04. The number of vortex blobs at time t is M^{n+1}, in which n is the integer part of $t/\{\sigma_M^2(1-\alpha^2)/4v\}$. The error associated with Huang's method is clearly about a tenth that associated with Rossi's for α = 0.9. Based on Fig. 1(a,b) and our experience, M = 4 and α = 0.85 are suggested [16,18] when efficiency and accuracy are both considered; these choices are employed in all our simulations presented here.

As the duration of simulation increases, further splitting events occur, and the number of blobs increases rapidly. To contain this situation, similar (with the same sense of rotation) and nearby blobs should become merged into one. If $\{(\Gamma_j,\vec{x}_j,\sigma_j)|j=1,\cdots,S\}$ is the set of similar and nearby blobs, they become replaced with one having $(\Gamma_0,\vec{x}_0,\sigma_0)$, such that

$$\Gamma_0=\sum_{j=1}^{S}\Gamma_j \tag{6a}$$

$$\Gamma_0\vec{x}_0=\sum_{j=1}^{S}\Gamma_j\vec{x}_j \tag{6b}$$

$$\Gamma_0\sigma_0^2=\sum_{j=1}^{S}\Gamma_j\left(\sigma_j^2+|\vec{x}_j-\vec{x}_0|^2\right) \tag{6c}$$

as long as

$$\Gamma_0/\Gamma_{ref}<\pi\varepsilon\alpha^2\sigma_{\max}^2 \tag{7a}$$

$$\sigma_0<\sigma_{\max} \tag{7b}$$

with selected error tolerance ε and reference circulation Γ_{ref}. The maximum field error induced in a single merging event is thus controlled with $\varepsilon\Gamma_{\text{ref}}$.

The sets of similar and nearby blobs to be merged are determined as follows. The entire flow domain is divided into equal square cells of size $\eta\sigma_{\max}$, and the merging criteria are examined over blobs in each cell; at most one merging event is allowed per cell per time step. The distance between neighboring blobs after merging is thus about $\sqrt{2}\eta\sigma_{\max}$ (the diagonal distance of the cell). Because the minimum core size of a blob is α $\sigma_{\max}$, the overlapping ratio (the ratio of the core size to the distance between neighboring blobs) is estimated to be $\alpha/(\sqrt{2}\eta)$.

3. Fast vortex method

3.1. Method

In applying the multipole method of Carrier et al. [24] to the solver, for blob j that is far from a group of nearby blobs, namely $\{(\Gamma_n,\vec{x}_n,\sigma_n)|n=1,\cdots,W\}$, we view the latter as a set of point vortices and approximate the induced velocity as

$$V_j^*=u_{\omega,j}-iv_{\omega,j}=\frac{-i}{2\pi}\sum_{n=1}^{W}\frac{\Gamma_n}{Z_j-Z_n} \tag{8}$$

in which appear $i=\sqrt{-1}$, complex position $Z=x+iy$ and complex velocity $V=u_\omega+iv_\omega$; superscript * denotes a complex conjugate. The multipole method expands the velocity, Eq. (8), about position Z_0 within an area occupied by the group of blobs in a Laurent series to term P as follows

$$V_j^* = \sum_{k=1}^{P} \frac{C_k}{(Z_j - Z_0)^k} \tag{9a}$$

$$C_k = \frac{-i}{2\pi} \sum_{n=1}^{W} \Gamma_n (Z_n - Z_0)^{k-1} \tag{9b}$$

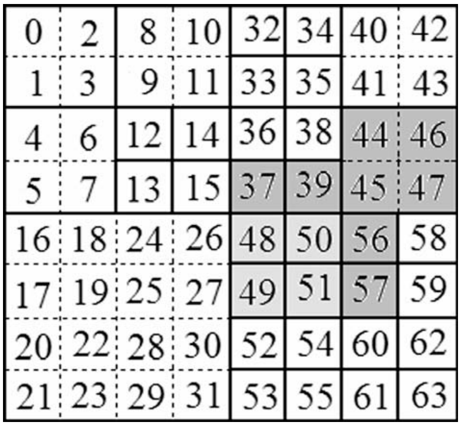
Errors of two kinds are associated with Eqs. (9a) and (9b): the first arises from an approximation of vortex blobs by point sources, and the second is due to truncated terms of higher order; the latter is known to be of order $W\Gamma_n|(Z_n - Z_0)/(Z_j - Z_0)|^{P+1}$, whereas the former error evidently increases with decreasing $|Z_j - Z_0|$ [35]. Rossi [36] derived and analyzed the associated Laurent series for elliptical Gaussian vortex blobs on taking into consideration their finite size; this error is related to the number of terms (P) in the expansion. Rossi suggested a critical value for axially symmetric Gaussian vortex blobs,

$$P_{cr} \approx \frac{2|Z_j - Z_0|^2}{\sigma^2} + 1 \tag{10}$$

which controls the upper bound of the finite-size error. When $P > P_{cr}$, this error dominates over the truncation error. A detailed investigation of all factors contributing to the overall accuracy of the approximation associated with the fast multipole method is found in [37].

The nearby blobs are grouped on taking advantage both of the square cells generated in the previous section for merging similar and nearby blobs and of the concept of adaptive domain decomposition that Carrier et al. proposed [24]: a minimum square domain is identified that must cover the entire flow field (all existing vortex blobs and objectives) and that comprises square cells of size $\eta\sigma_{\max}$. Four neighboring cells become combined into a larger subdomain and four subdomains become combined into an even larger subdomain. This procedure terminates only when one subdomain remains, namely the minimum square domain specified above. The largest subdomains that contain less than n_0 blobs are called childless subdomains. The minimum square domain is labeled as the subdomain of level zero; its four progeny subdomains are called subdomains of first level, and so forth. The maximum level is that of the finest childless subdomains. Fig. 2 illustrates one such adaptive domain decomposition; the grids divided by the dotted lines are the cells used to merge similar and nearby blobs. In total, 64 cells are numbered in a way to be explained later. The squares enclosed with solid lines are adaptive subdomains prepared for the group-to-point computation. The maximum level ($L_{\max}$) of this domain decomposition is three. The finest childless subdomains might be as the same as, or larger than, the cells for merging; in the example they are the same. The childless subdomains might not be the finest.

The neighbors of a subdomain are thus defined as the subdomain itself and those subdomains that are as large as or larger (possible only if the neighboring subdomain is childless) than the subdomain and are connected with it by one side or a point. For childless subdomain 50 in Fig. 2 for instance, its parent is a subdomain consisting of cells 48, 49, 50 and 51 (light grey) and its grandparent is a subdomain comprising cells 48–63. The neighbors of childless subdomain 50 include both light and deep grey cells. The mutually induced velocities among vortex blobs are computed in the following way. The velocities induced by blobs within each childless subdomain on blobs in its neighbors are first calculated via the point-to-point formula, namely Eqs. (4a) and (4b). The velocities induced by blobs within each subdomain on blobs belonging to its parent's neighbors excluding its own neighbors are computed next by the group-to-point formula, Eqs. (9a) and (9b). Beginning with subdomains of the maximum level, the group-to-point calculation is continued level by level until the first level is reached, then terminating the computation of velocities.



0	2	8	10	32	34	40	42
1	3	9	11	33	35	41	43
4	6	12	14	36	38	44	46
5	7	13	15	37	39	45	47
16	18	24	26	48	50	56	58
17	19	25	27	49	51	57	59
20	22	28	30	52	54	60	62
21	23	29	31	53	55	61	63

Fig. 2. Illustration of the employed adaptive domain decomposition and the binary numbering system.

As van Dommelen and Rundensteiner suggested [26], the Laurent coefficients of a parent subdomain are computed with the Laurent coefficients of its four progeny subdomains as follows:

$$C'_k = \sum_{\text{four children}} \left\{ \sum_{n=1}^{k} (m_n^k C_n) \right\} \tag{11a}$$

$$m_k^k = 1 \tag{11b}$$

$$m_{k-l}^k = m_{k-l+1}^k \times dZ \times \frac{k-l}{l} \quad \text{for } l = 1, \ldots, k-1 \tag{11c}$$

in which appear $dZ \equiv Z_0 - Z'_0$ and the center Z'_0 and Laurent coefficients C'_k of the mother subdomain.

3.2. Numbering system

To identify rapidly the neighbors of a subdomain, we number the finest cells as follows. Of C^2 finest cells in total, each cell has two-dimensional coordinates (x,y) for x, y = 0, 1,...,$C-1$. A new binary number is obtained on interlacing the binary digits of x and y; its corresponding decimal number d is exactly the number of the cell. For instance, the cell in the fourth column and the third row in Fig. 2 has coordinates $(x, y) = (3,2)$ or (011,010). The binary number after interlacing the digits is (001110), which equals 14. An array $L(d)$ is then built to memorize the level (L) of the childless subdomain that contains this cell. A subdomain might possess more than one cell, which we recognize by its level L and the smallest number d among its $4^{L_{\max}-L}$ cells. A subdomain that contains cells 48–51 in Fig. 2, for example, is labeled with $(d,L) = (48,2)$. We can hence identify quickly the neighbors of one subdomain (d,L) based on this particular numbering system as follows.

(i) Obtain the binary expression of d and curtail its last $2^{L_{\max}-L}$ digits, yielding a new number d'.
(ii) Separate the odd and even digits of the binary expression of d', and use them to form two individual numbers d_x and d_y; add one to, or subtract one from, either or both d_x and d_y, thus producing numbers in eight couples.
(iii) Generate eight numbers on interlacing the binary digits of the couples of numbers obtained in the preceding step; append $2^{L_{\max}-L}$ zeros to the right and find the corresponding decimal numbers r.
(iv) Test the levels L' of the cells labeled with these eight numbers r. If $L' \geqslant L$, the subdomain labeled with (r,L') is a neighbor desired; otherwise (r',L') is wanted, for which r' is the number obtained by nullifying the latter $2^{L_{\max}-L'}$ digits of the binary expression of r.

In subdomain $(d, L) = (48,2)$ in Fig. 2 for instance, because $L_{\max} = 3$, the binary expression of d' is 1100; the binary expression of d_x is thus 10 and of d_y is 10, and the eight pairs of numbers are (01,10), (11,10), (10,11), (10,01), (11,11), (01,01), (11,01) and (01,11). The eight numbers generated in step (iii) are thus 011000(24), 111000(56), 110100(52), 100100(36), 111100(60), 001100(12), 101100(44) and 011100(28). Except the first and the last, the others have levels equal to, or greater than, 2 (L), and are thus the wanted neighbors of subdomain (48,2). Cells 24 and 28 each have a level one ($L' = 1$); their corresponding values of r' are the same, namely 010000(16). The last neighbor of subdomain (48,2) is consequently subdomain (16,1).

3.3. Tests

To test the accuracy and efficiency of our proposed fast method, we randomly distributed N Gaussian blobs within a circle of diameter 10 and adjusted the core size σ so that the overlap ratio was 1.5; the average relative error induced with the fast

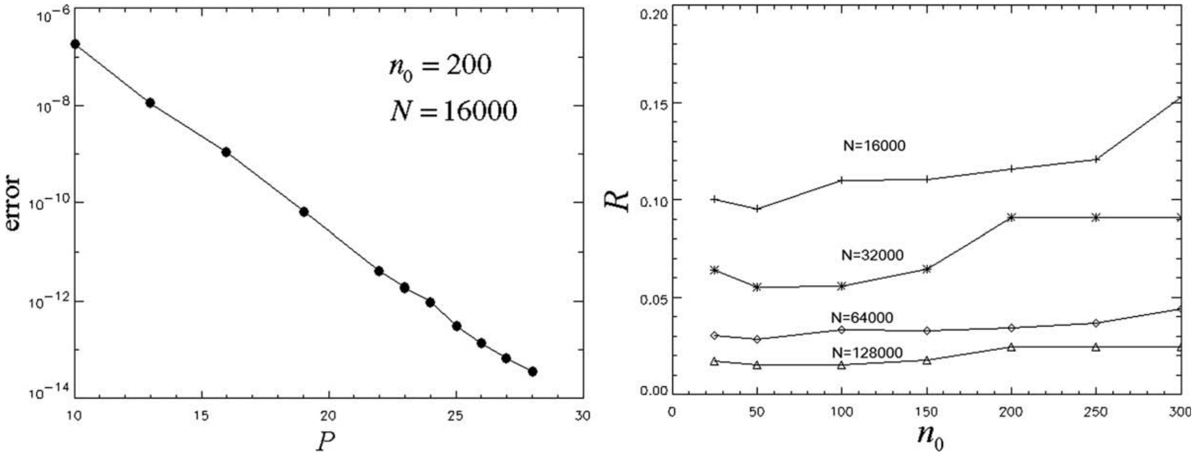


Fig. 3. Average relative error and the CPU time reduction by the fast method.

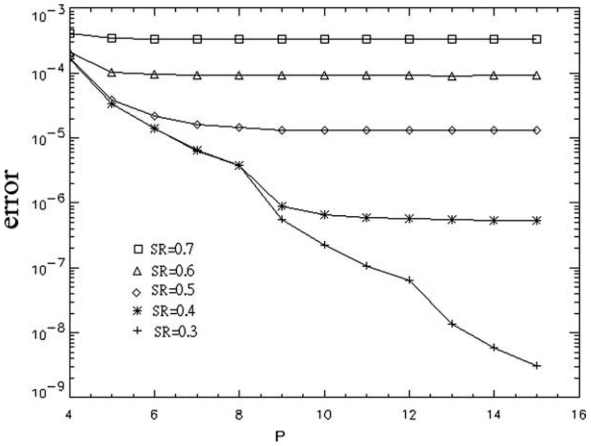


Fig. 4. Relative error vs number of terms P used in the Laurent series under a specified ratio $SR = \sigma/l_{\min}$.

method and the ratio R of CPU time required by the non-fast (the original) and fast methods are shown in Fig. 3. With $N = 16000$ and $n_0 = 200$, the maximum level is 4 (finest subdomain size, $l_{\min}$, is 0.625), and the core size is about 0.105. We estimate the value of $|Z_j - Z_0|$ in Eq. (10) to be 1.5 $l_{\min}$, which is the minimum distance practicable according to the present algorithm. The critical number evaluated with Eq. (10) is about 160. For $P < 30$, the truncation error hence dominates but decreases exponentially with P, as seen in Fig. 3. Moreover, the more numerous are the blobs, the larger is the reduction ratio of CPU times. Because the point-to-point computations increase and the group-to-point calculations decrease with an increasing n_0, an optimal choice $n_0 \approx 50$ seems to exist; this value is larger than in previous investigations [24], likely because present computers are more powerful in vectoral computations, which speeds significantly the point-to-point calculations. In Fig. 4, we attempted to control the ratio of the Gaussian core size to the finest subdomain size ($SR \equiv \sigma/l_{\min}$), and measured the error associated with the fast algorithm. The Gaussian blobs number 16000 in total and $n_0 = 50$. The error ceased perceptibly to decrease beyond some value of P. When $SR = 0.6$, the critical value given by Eq. (10) is 13.5 if $|Z_j - Z_0| = 1.5\ l_{\min}$ is chosen and 6.5 if $|Z_j - Z_0| = l_{\min}$. With $SR = 0.5$, the values are 19 and 9, respectively. When these values are compared with the results in Fig. 4, the optimum number of terms that should be used in the Laurent expansion appears to be

$$P_{opt} = \frac{2l_{\min}^2}{\sigma^2} + 1 = \frac{2}{SR^2} + 1 \tag{12}$$

The latter relations indicate that $P_{opt} \approx 23$ when $SR = 0.3$, consistent with the measured overall error shown in Fig. 4, which decays exponentially up to $P = 15$.

According to another test in Figs. 5 and 6, we simulated the merging of two identical Gaussian vortices, initially having radius 1 and separated a distance $b_0 = 5$. Each vortex has a circulation $\Gamma = 1000$ and Reynolds number $Re = \Gamma/2\pi\nu \approx 160$. In

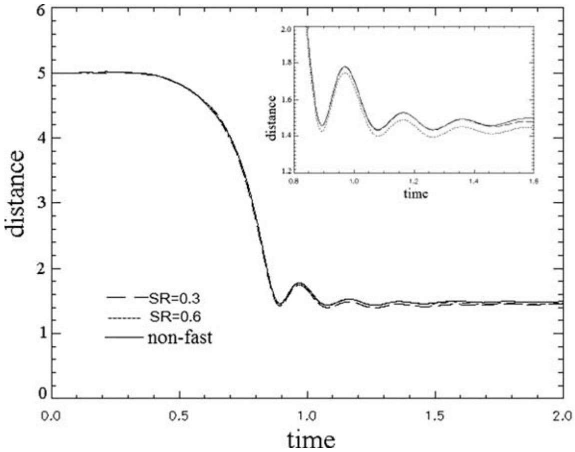


Fig. 5. Temporal evolution of the distance between two vortices merging.

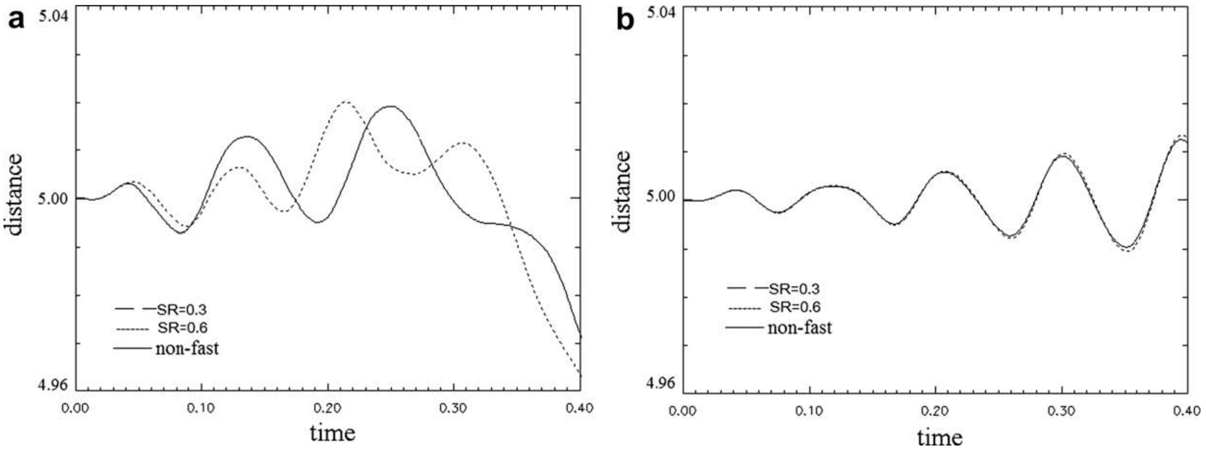


Fig. 6. Early temporal evolution of the distance between two vortices merging: (a) viscous (b) inviscid.

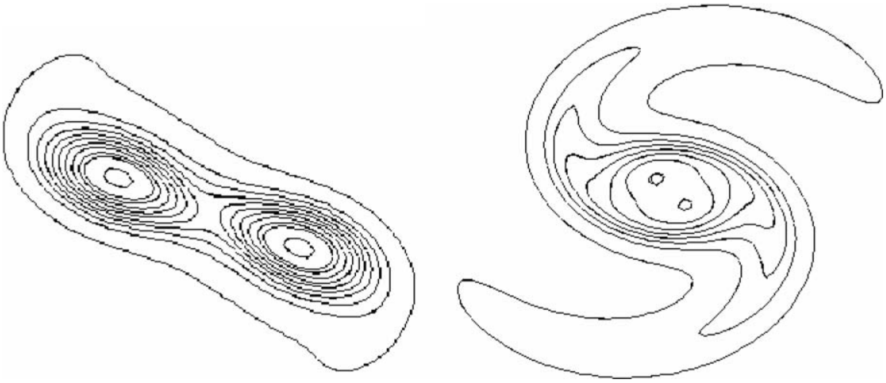


Fig. 7. Instantaneous vorticity contours obtained by the non-fast method (dotted lines) and the fast method with $n_0 = 50$ and $P = 6$ (solid lines) at $t/T = 0.4$ (left) and 0.8 (right).

total 2160 initial blobs have $\sigma = 0.084$; other numerical parameters are $\sigma_{\max} = 0.25$, $\alpha = 0.85$ and $\Delta t = T/100$, with $T = 2\pi^2 b_0^2/\Gamma$. Two fast methods, with $P = 6$ and $SR = 0.6$ or $P = 13$ and $SR = 0.3$, were employed. Fig. 5 shows that the differences in the temporal evolution of the distance between two vortex centers remain small; in particular, the difference between the non-fast result and the fast result for $P = 13$ and $SR = 0.3$ is negligible. On examination on the early-time behaviors we find obvious, but small, differences between the non-fast and fast results for $P = 6$ and SR=0.6, as shown in Fig. 6(a). Because the error associated with the fast algorithm found in Fig. 3 is small, we conjecture that additional error must arise when such an algorithm is combined with the splitting/merging algorithm. Basically, a small error in the velocity due to the fast algorithm produces a slightly different displacement of a vortex blob, which in turn might yield other merging events. Error might therefore accumulate with time. This conjecture was examined on simulating the same flow field except with $\nu = 0$ so that the splitting/merging algorithm is never evoked. Fig. 6(b) shows that the non-fast and fast results of $P = 6$ and $SR = 0.6$ agree satisfactorily. Fig. 7 shows a comparison of the instantaneous vorticity contours of the viscous flows in which a difference between the non-fast and fast results of $P = 6$ and $SR = 0.6$ is scarcely perceptible.

We suggest to determine pertinent numerical parameters as follows. One specifies the desired values of SR and n_0, chooses the optimum value of P determined by Eq. (12), and sets the allowable maximum level for the domain decomposition as $\log_2(L \cdot SR/\sigma_{\max})$, with L as the size of the minimum square that encloses all existing vortex blobs.

4. No-slip boundary conditions

The no-slip boundary condition is fulfilled on placing a vortex sheet of strength γ (circulation per unit length) at the wall. The strength is governed by

$$\gamma(s) - \frac{1}{\pi}\oint_S \frac{\partial}{\partial n}\log|\vec{x}(s) - \vec{x}(s')|\gamma(s')ds' = -2\frac{\partial\psi_{ext}}{\partial n}(\vec{x}(s)) \quad (13a)$$

$$\oint_S \gamma(s,t)ds + \Gamma_{wake}(t) = 0 \quad (13b)$$

in which ψ_{ext} is the stream function associated with the external flow, including free streams and vortex blobs, Γ_{wake} is the total circulation inside the flow, and s and n are tangential and normal coordinates along the solid boundary. We discretized Eq. (13) with the boundary-element method and constant panels (panels of constant strength) [33], and solved with a requirement of least squared error [38]. To avoid fluctuations, we treated the external vortex blobs as point vortices when evaluating the right side of Eq. (13a).

4.1. Diffusion of the vortex sheet

To simulate the diffusion of the vortex sheet, we adopted an approximation solution proposed by Koumoutsakos [34]:

$$\omega_{\gamma,i}(x,y,t+\Delta t)=\sum_{i=1}^{M}\frac{\gamma_i\exp\left(-\frac{y^2}{4\nu\Delta t}\right)}{2\sqrt{\pi\nu\Delta t}}\left[erf\left(\frac{\frac{l_i}{2}-x}{\sqrt{4\nu\Delta t}}\right)+erf\left(\frac{\frac{l_i}{2}+x}{\sqrt{4\nu\Delta t}}\right)\right] \tag{14}$$

in which l_i is the length of panel i (in total M panels are employed), γ_i is the constant strength on the panel at time t, (x,y) are the local tangential and normal coordinates of the panel, and erf is the error function. Eq. (14) assumes a constant vorticity flux ($\partial\ \omega_\gamma/\partial y = -\gamma_i/\nu\Delta t$) at the wall during period (t, $t+\Delta t$). To discretize this, vorticity distribution and to maintain consistency with the resurrected core-spreading vortex method, we generated new blobs, called wall blobs, at fixed locations outside the solid body, as illustrated in Fig. 8; these wall blobs are placed along the central lines normal to the panels. When the length of each panel is much less than the radius of curvature, adjacent blobs might be viewed as collinear, as shown in Fig. 8(b). The size h of the squares that must be adapted according to the growth of the viscous sublayer thickness during one time step is chosen to be $\sqrt{2\nu\Delta t}$. We chose the core size of these new blobs to be $\sigma_\gamma=\sqrt{2}h$.

The strengths of these blobs are then assumed to be the total circulation contributed from the nearby five panels, that is

$$\Delta\Gamma_j=\sum_i\int\int_{h\times h}\omega_{\gamma,i}(x_j,y_j,t+\Delta t)dxdy\equiv\sum_i\Delta\Gamma_{j,i} \tag{15}$$

in which $\Delta\Gamma_j$ is the strength of the jth new blob, and (x_j,y_j) are its coordinates relative to the local coordinate system of panel i. Because we chose for the vortex blobs to employ Gaussian distributions, new blobs nearest the wall are placed distance $r_1=r_0+h/2$ from the wall to decrease the amount of vorticity spread into the interior of the body. The circulation within a distance r_0 from the wall hence becomes replaced back to the wall. This circulation is called the residual circulation. This use of residual circulation might cause a slight delay of diffusion of the vorticity from the wall to the flow but it prevents blobs from remaining too near the solid body, that might otherwise cause large errors in evaluating γ and result in a failure of the entire vortex method. After a series of tests, we found $r_0=0.3h$ to be an effective choice. To conserve the total circulation, we modified the strengths of the wall blobs, Eq. (15), on replacing $\Delta\Gamma_{j,i}$ with

$$\Delta\Gamma'_{j,i}=\Delta\Gamma_{j,i}+\frac{\Delta\Gamma_{j,i}^2}{\sum_i\Delta\Gamma_{j,i}^2}\left(l_j\gamma_j\left[1-erf\left(\frac{r_0}{\sqrt{4\nu\Delta t}}\right)\right]-\sum_i\Delta\Gamma_{j,i}\right) \tag{16}$$

in which the second term in the square brackets is the residual circulation.

4.2. Vortex blobs near the wall

Some external vortex blobs might move, because of convection or splitting and merging, toward the wall and remain so near the wall that their Gaussian tails extend into the interior of the solid body. To avoid this condition, we examined vortex blobs in the neighborhood of the solid body and judged a blob too near the wall according to the following criterion:

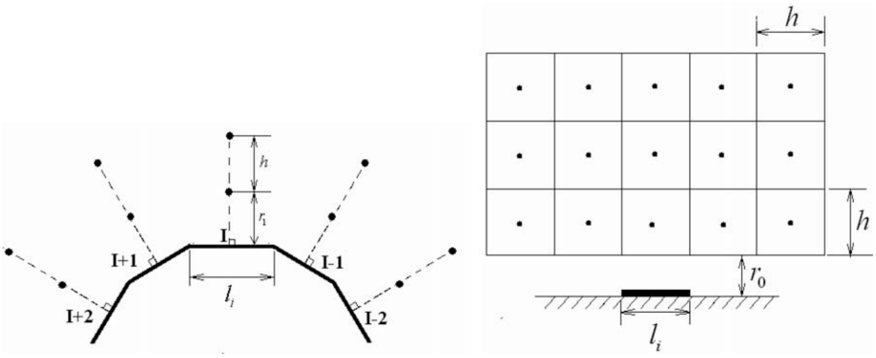


Fig. 8. New blobs generated for a discrete representation of the diffusion of the vortex sheet.

$$\frac{r}{\sigma} < \frac{r_1}{\sigma_\gamma} \equiv r^*$$

in which r is the distance from the wall and σ is the core size of the too-near blob. Too-near blobs then become eliminated and their circulations distributed to newly generated vortex blobs mentioned in Subsection 4.1, the wall blobs. The circulation of one wall blob located at $\vec{x}_j$ thus becomes increased by an amount

$$h^2 \sum_k \frac{\Gamma_k}{\pi \sigma_k^2} \exp\left(-\frac{\left|\vec{x}_j - \vec{x}_k\right|}{\sigma_k^2}\right)$$

in which the sum applies to those too-near blobs. Any deficiency in the circulation becomes treated as part of the residual circulation.

4.3. Time marching

Here we describe a flow chart of the proposed vortex method. Given flow details at time t_n, we convect the vortex blobs with a modified Euler method and make the blob cores grow according to Eq. (3b). The core sizes are then examined, and splitting and merging are performed if necessary. The strength of the vortex sheet at the solid wall is next solved, and denoted γ_n^*. To improve the accuracy, we assumed the constant vorticity flux at the solid wall during $(t_n,\ t_n + \Delta t)$ to be $\partial \omega_\gamma / \partial y = -(\gamma_n + \gamma_n^*)/2\nu\Delta t$. New vortex blobs were then generated to simulate the diffused result of the vortex sheet for time period Δt. Vortex blobs that are too near the wall were handled at the same time. On calculating the residual circulation density γ_{n+1}, this time step terminated.

5. Impulsively initiated circular cylinder

To test the completed fast resurrected core-spreading vortex method, we chose the flow induced with an impulsively initiated circular cylinder, paying special attention to the singularity or discontinuity at time $t = 0$. Because there is no vortex blob at $t = 0$, according to the procedures described in Subsection 4.3, the strength of the vortex sheet remains the same, γ_0, from $t = 0$ to $t = \Delta t$. Fig. 9 shows the distribution of this initial strength and the distribution γ_1 at $t = \Delta t = 0.01$, which Bar-Lev and Yang [39] obtained with a perturbation technique. The difference evidently arises from lack of knowledge of the advection effect during the first time step. To decrease this initial error, we modified the time march of the first time step as follows. A vortex sheet of strength γ_0 was first diffused for a period $\Delta t/2$; the diffused result was approximated with wall blobs according to the description in Subsection 4.1. These blobs were then advected with spreading cores for another period $\Delta t/2$. A new vortex sheet of strength γ_1^* was then calculated and allowed also to diffuse for a period $\Delta t/2$. New wall blobs were again generated to represent this diffused result, and the first time step concluded on calculating the residual circulation density γ_1. The flow proceeded according to the procedures given in Subsection 4.3.

5.1. Re = 550

We first tried to simulate the flow with a Reynolds number $Re = UD/\nu = 550$, in which U is the free stream velocity and D is the diameter of the cylinder. The numerical parameters used in the splitting/merging algorithm are $\sigma_{\max} = \sqrt{6}h$, $\alpha = 0.85$, $\Gamma_{\text{ref}} = UD$, $\varepsilon = 1$, $\eta = 0.5$, and $\Delta t = 0.01D/U$. The overlap among blobs was thus controlled to be about 1.2. Those parameters

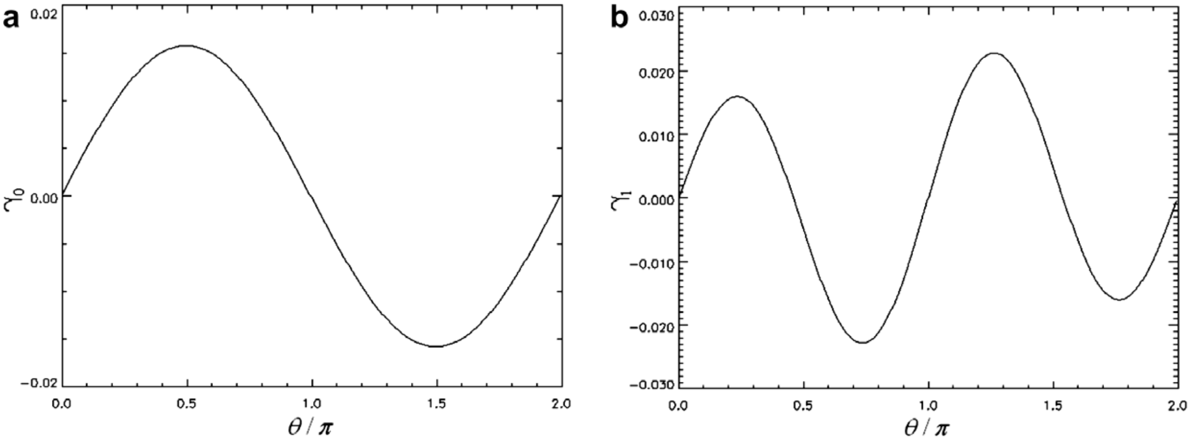


Fig. 9. Distributions of the strength of the vortex sheet induced on the impulsively initiated circular cylinder at $t = 0$ (a) and $t = \Delta t = 0.01$ (b).

associated with the fast method are $P = 13$ and $n_0 = 100$. Both the panel length and h were chosen to be $\sqrt{2\nu\Delta t}$. In total 522 panels ($\approx \pi D/\sqrt{2\nu\Delta t}$) were employed to approximate the circular boundary; the ratio of the panel length to the radius of the cylinder is of order 10^{-2}, $\sigma_\gamma = \sqrt{2}h$ and $r_0 = 0.3h$.

Shown in Fig. 10 are the vorticity contour plots at $tU/D = 1$, 3 and 5. The magnitudes of the contours were chosen to be as the same as those in Fig. 12 of Ploumhans and Winckelmans (PW) [40], who modified the particle-strength-exchange (PSE) method near solid boundaries to avoid a spurious vorticity flux during the convection step. A comparison finds the obtained vorticity structures to be nearly identical. In Fig. 11, we compare the simulated results with experimental measurements [41]. The calculated velocity distributions along the x-axis agree satisfactorily with the measurements. We computed the linear impulse (I) and the drag coefficient (C_D) as follows:

$$I = \sum_{i=1}^{N} y_i \Gamma_i + \sum_{m=1}^{M} y_{c,m}\gamma_m l_m \tag{17a}$$

$$C_D = \frac{I(t-\Delta t) - I(t+\Delta t)}{U^2 D \Delta t} \tag{17b}$$

in which N is the total number of vortex blobs in the flow and M is the number of panels used. The results shown in Fig. 12 are compared with those of Ploumhans and Winckelmans [40], Koumoutsakos and Leonard [42], and the analytic solution derived by Bar-Lev and Yang [39]. Koumoutsakos and Leonard (KL) [42] replaced the differential operators in Eq. (1) with integral operators that were in turn discretized with quadrature having as quadrature points the locations of the blobs. Like the PSE methods, remeshing was necessary to solve the problem of Lagrangian grid distortion. The present results agree bet-

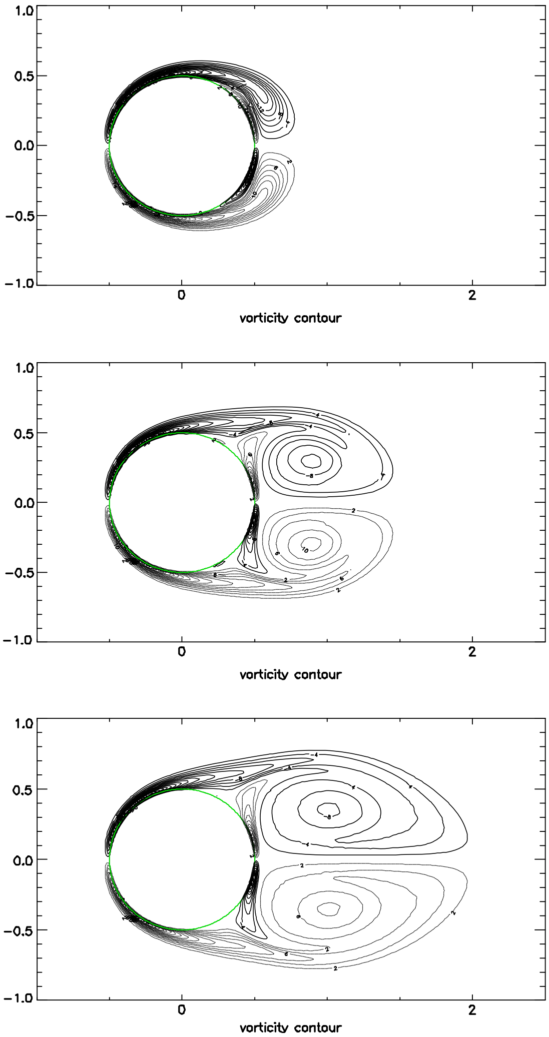


Fig. 10. Vorticity contours (increment $D\Delta\omega/U = 2$) at $tU/D = 1$, 3 and 5 for $Re = 550$.

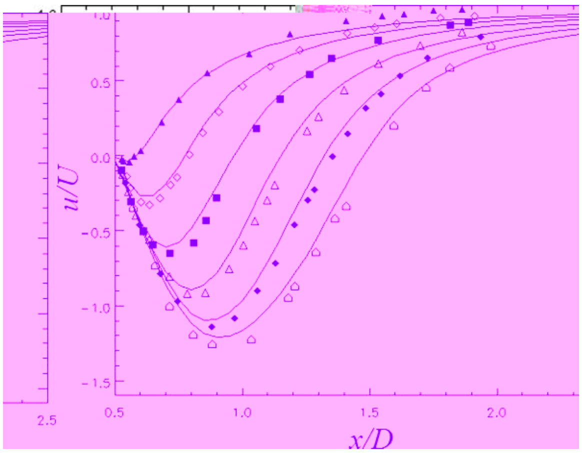


Fig. 11. Velocity distributions along the *x*-axis at several times. Symbols denote experimental measurements [41]. ▲, *tU*/*D* = 0.5; □, *tU*/*D* = 1.0; ■, *tU*/*D* = 1.5; △, *tU*/*D* = 2.0; ♦, *tU*/*D* = 2.5; ⬠, *tU*/*D* = 3.0.

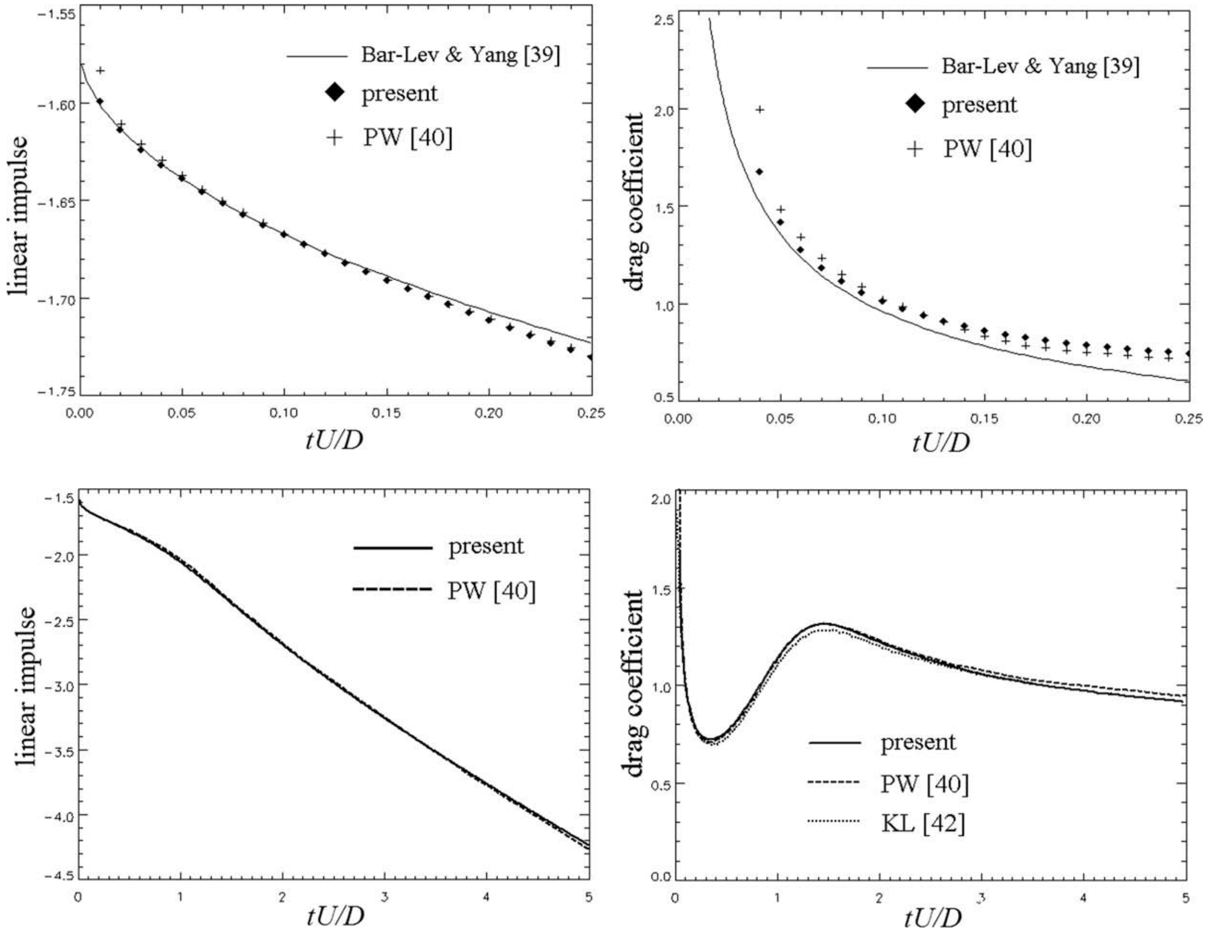


Fig. 12. Linear impulse and drag coefficient of our simulation results compared with those obtained by Ploumhans and Winckelmans [40], Koumoutsakos and Leonard [42] (dotted line), and the analytic solution with a perturbation by Bar-Lev and Yang [39].

ter with the results of Ploumhans and Winckelmans [40] before $tU/D \approx 2.5$ (the drag coefficient of Koumoutsakos and Leonard [42] is smaller). The present result agrees with the analytic solution even better at the smallest times, attributed to a

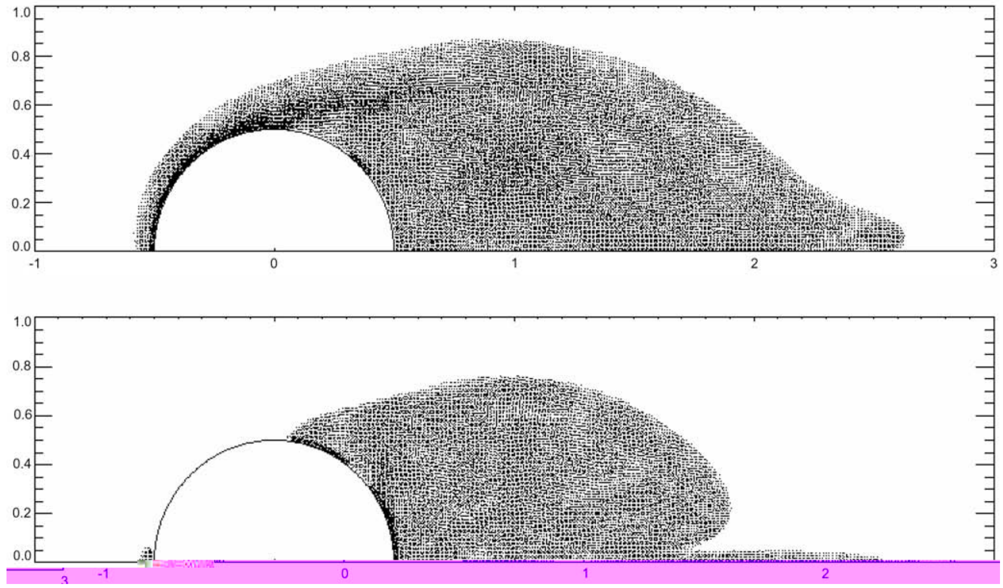


Fig. 13. Instantaneous locations of vortex blobs of counter-clockwise rotation (a) and those of clockwise rotation (b) at $tU/D = 5$ when only blobs of the same sense of rotation are allowed to merge.

special treatment in the first time step. With increasing time, our drag coefficient becomes slightly smaller than that of Ploumhans and Winckelmans [40], matching that of Koumoutsakos and Leonard [42] instead.

Ploumhans and Winckelmans [40] used ∼5000 to 70 000 computational elements and 592 panels. We performed our simulation on a computer (Intel®Core™2 6600@2.4 GHz) in 3.27 h, with 522 panels and vortex blobs numbering 100 000 in total at the final time. Fig. 13 shows the instantaneous distribution of these vortex blobs at the final time step, clearly showing the advantage of compactness associated with the vortex method. Because only vortex blobs of a same sense of rotation are allowed to be merged, there generally exist two blobs within each cell after the merging, one counter-clockwise and the other clockwise. Vortex blobs of opposite senses of rotation can also be merged as long as the associated error is less than the prescribed tolerance [16]. In such a simulation that we performed also, the results were only slightly worse (not shown here),

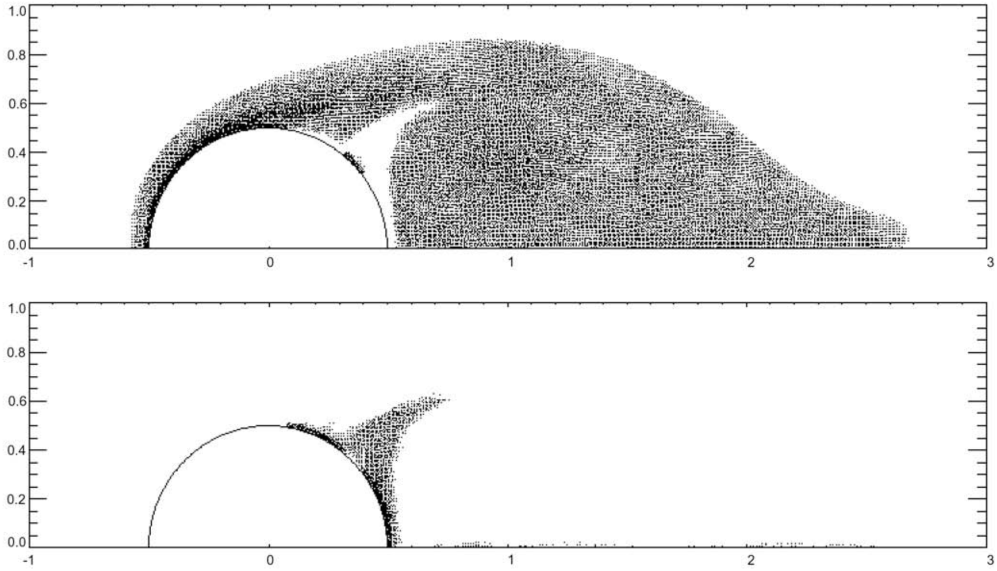


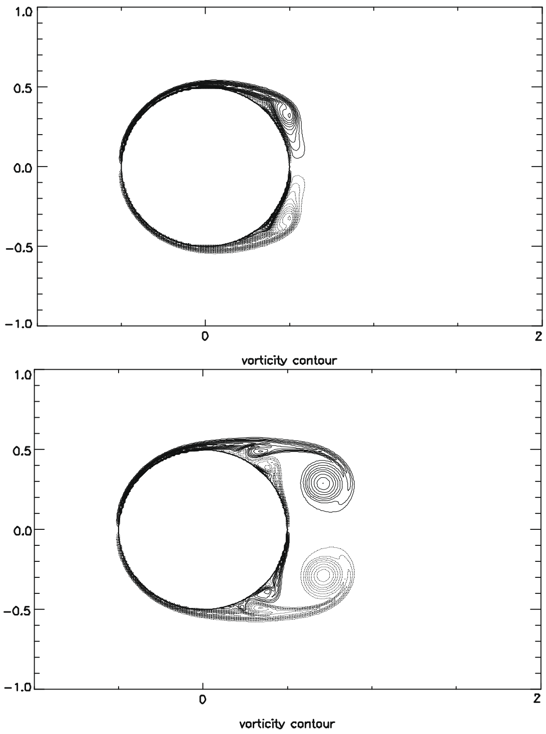
Fig. 14. Instantaneous locations of vortex blobs of counter-clockwise rotation (a) and those of clockwise rotation (b) at $tU/D = 5$ when blobs of opposite senses of rotation are also allowed to merge.

but the total number of vortex blobs decreased to about 62 000 at $tU/D = 5$. The instantaneous locations of these vortex blobs are shown in Fig. 14.

5.2. Re = 3000

We attempted also to simulate a flow having a Reynolds number as large as 3000. To ensure sufficient resolution, we chose $\Delta t = 0.005$ and modified the related numerical parameters accordingly. Numerical parameters unrelated to Δt remained unaltered, except the error tolerance for merging. We selected a value $\varepsilon = 10$ to have an acceptable extent of computation. The vorticity contours at $tU/D = 1$, 2, and 3 are illustrated in Fig. 15 and the computed drag coefficient is shown in Fig. 16. Compared to the results of Ploumhans and Winckelmans [40], our primary vortex is more axially symmetric, likely because of the large error tolerance (i.e. too many merging events allowed), which enhances the diffusion effect. Even so, the second vortices attached to the cylinder agree satisfactorily with theirs [40], whereas the computed drag coefficients coincide at early times but deviate after $tU/D \approx 1$. This derivation is irreducible on decreasing $\sigma_{\max}$ and Δt or varying the numerical parameters related to residual circulation. We thus conjecture that the enhanced diffusion effect from the merging errors is responsible; a smaller ε yields more surviving vortex blobs and thus a great computational amount.

Compared with 1944 panels and 500 000 particles at $tU/D = 5$ used by Ploumhans and Winckelmans [40] and with 300 000 particles at $tU/D = 3$ used by Koumoutsakos and Leonard [42], we employed 1722 panels and 210 000 vortex blobs in total at the final time ($tU/D = 3$) as shown in Fig. 17, requiring 17.3 h on the same computer.



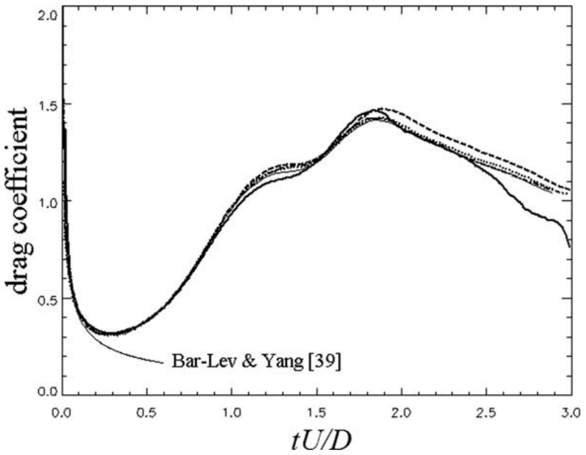


Fig. 16. Drag coefficient obtained with our vortex method (thick solid line), compared to results of Ploumhans and Winckelmans [40] (dashed line), of Koumoutsakos and Leonard [42] (dotted line), and of Shiels et al. (thin solid line), and the spectral-method result of Henderson [28] (dashed-dotted line).

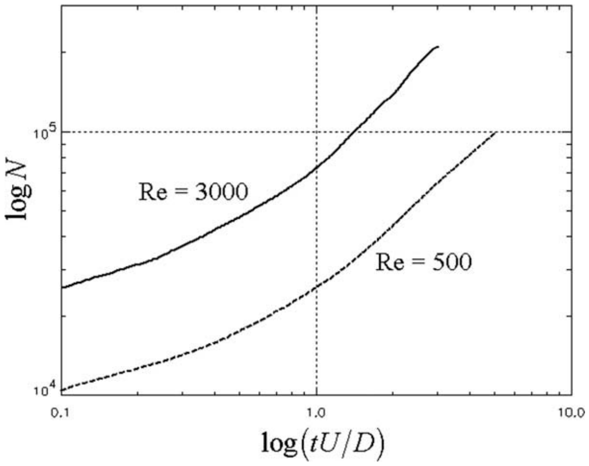


Fig. 17. Number of computational elements vs time for an impulsively initiated cylinder at $Re = 3000$ compared with $Re = 550$.

6. Conclusion

A vortex method suitable for simulating two-dimensional viscous incompressible flows is here proposed such that the vorticity field is composed of Gaussian vortex blobs of which the core areas grow linearly with time. To control the convection error and the computational effort, we applied blob splitting and merging, and a multipole method to speed the computation of velocities assisted by a carefully designed system of binary numbering. The boundary condition of no-slip is satisfied on placing a vortex sheet at the wall and having it diffuse properly into the flow. To prevent the vorticity from entering the body, we introduced a concept residual circulation such that circulation that remains too near the wall is put back to the wall.

The solver so developed that requires no grid-based remeshing has been applied to simulate the flow induced with an impulsively initiated circular cylinder. The results agree satisfactorily with previous experimental and numerical data for $Re = 550$ in both accuracy and efficiency. When the Reynolds number is increased to 3000, the early behaviors of the flow ($tU/D \leqslant 1$) are captured also satisfactorily, but the long-time behaviors deviate from those in preceding numerical investigations; this deviation is attributed to the error arising from merging that is not readily improvable because adjustable parameters are limited. A non-uniform spatial resolution, in both core size and error tolerance associated with the merging, is expected to be helpful and is currently under development.

Acknowledgments

This work was supported by the National Science Council of Taiwan (Grant No. NSC 95-2221-E-002-336-MY2). We are indebted to Prof. Koumoutsakos for useful discussions and a critical reading of the manuscript. Many thanks go to honorable reviewers for very useful comments.

References

[1] A. Leonard, Vortex methods for flow simulations, J. Comput. Phys. 37 (1980) 289–335.

[2] C. Chang, R. Chern, A numerical study of flow around an impulsively started circular cylinder by a deterministic vortex method, J. Fluid Mech. 233 (1991) 243–263.

[3] Z.Y. Lu, S.F. Shen, in: C. Taylor, et al. (Eds.), Numerical Methods in Laminar and Turbulent Flow, Pineridge Press, Swansea, UK 5 (1987) 619.

[4] Z.Y. Lu, T.J. Ross, Diffusing-vortex numerical scheme for solving incompressible Navier–Stokes equations, J. Comput. Phys. 95 (1991) 400–435.

[5] A.J. Chorin, Numerical study of slightly viscous flow, J. Fluid Mech. 57 (1973) 785–796.

[6] P. Degond, S. Mas-Gallic, The weighted particle method for convection–diffusion equations. Part 1: The case of an isotropic viscosity, Math. Comput. 53 (1989) 485–507.

[7] D. Fishelov, A new vortex scheme for viscous flow, J. Comput. Phys. 86 (1990) 211–224.

[8] S. Shankar, L.L. van Dommelen, A new diffusion procedure for vortex methods, J. Comput. Phys. 127 (1996) 88–109.

[9] L.L. Van Dommelen, S. Shankar, Two counter-rotating diffusing vortices, Phys. Fluids A 7 (1995) 808–819.

[10] A. Leonard, Computing three-dimensional incompressible flows with vortex elements, Annu. Rev. Fluid Mech. 17 (1985) 523–559.

[11] C. Anderson, C. Greengard, On vortex methods, SIAM J. Numer. Anal. 22 (1985) 413–440.

[12] K. Gustavson, J. Sethian, Vortex Method and Vortex Motion, SIAM, Philadelphia, 1991.

[13] J.T. Beale, G.H. Cottet, S. Huberson, Vortex Flows and Related Numerical Methods, Kluwer Academic Publisher, 1993.

[14] G.H. Cottet, P. Koumoutsakos, Vortex Methods: Theory and Practice, Cambridge University Press, 2000.

[15] L. Rossi, Resurrecting core-spreading vortex methods: A new scheme that is both deterministic and convergent, SIAM J. Sci. Comput. 17 (1996) 370–397.

[16] M.J. Huang, Diffusion via vortex splitting and regridding via vortex merging in core-spreading vortex methods, Int. J. Numer. Methods Fluids 48 (2005) 521–539.

[17] L. Rossi, Merging computational elements in vortex simulations, SIAM J. Sci. Comput. 18 (1997) 1014–1027.

[18] Mei-Jiau Huang, The physical mechanism of symmetric vortex merger: a new viewpoint, Phys. Fluids 17 (2004) 074105.

[19] J.H. Park, D.J. Lee, Numerical simulation of vortex–wedge interaction, AIAA J. 32 (1994) 1126–1134.

[20] S.B. Baden, E.G. Puckett, A fast vortex method for computing 2D viscous flow, J. Comput. Phys. 91 (1990) 278–297.

[21] A.S. Almgren, T. Buttke, P. Colella, A fast adaptive vortex method in three dimensions, J. Comput. Phys. 113 (1994) 177–200.

[22] G. Russo, J. Strain, Fast triangulated vortex methods for the 2D Euler equations, J.Comput. Phys. 111 (1994) 291–323.

[23] J. Strain, Fast adaptive 2D vortex methods, J. Comput. Phys. 132 (1997) 108–122.

[24] J. Carrier, L. Greengard, V. Rokhlin, A fast adaptive multipole algorithm for particle simulations, SIAM J. Sci. Stat.Comput. 9 (1988) 669–686.

[25] C.R. Anderson, An implementation of the fast multipole method without multipoles, SIAM J. Sci. Stat. Comput. 13 (1992) 923–947.

[26] L. Van Dommelen, E.A. Rundensteiner, Fast, adaptive summation of point forces in the two-dimensional Poisson equation, J. Comput. Phys. 83 (1989) 126–147.

[27] C.I. Draghicescu, An efficient implementation of particle methods for the incompressible Euler equations, SIAM J. Numer. Anal. 31 (1994) 1090–1108.

[28] Doug Shiels, Simulation of Controlled Bluff Body Flow with a Viscous Vortex Method, Ph.D. Thesis, GALCIT, Caltech, USA, 1998.

[29] P.F. Fischer, A.T. Patera, Parallel simulation of viscous incompressible flow, Annu. Rev. Fluid Mech. 26 (1994) 483–527.

[30] J.D. Anderson, Fundamentals of Aerodynamics, fourth ed., McGraw Hill, 2007.

[31] C.R. Anderson, M.B. Reider, Investigation of the use of Prantl/Navier–Stokes equation procedures for two-dimensional incompressible flows, Phys. Fluids 6 (1994) 2380–2389.

[32] L.F. Rossi, Vortex computations of wall jet flows, in: Forum on Vortex Methods for Engineering Applications, Albuquerque, New Mexico, February 22–24, 1995.

[33] Prem K. Kythe, An introduction to boundary element methods, CRC Press, Boca Raton, 1995.

[34] P. Koumoutsakos, A. Leonard, F. Pepin, Boundary condition for viscous vortex methods, J. Comput. Phys. 113 (1994) 52–61.

[35] J.T. Hamilton, G. Majda, On the Rokhlin–Greengard method with vortex blobs for problems posed in all space or periodic in one direction, J. Comput. Phys. 121 (1995) 29–50.

[36] L.F. Rossi, Evaluation of the Biot–Savart integral for deformable elliptical Gaussian vortex elements, SIAM J. Sci. Comput. 28 (2006) 1509–1532.

[37] F.A. Cruz, L.A. Barba, Characterization of the errors of the fast multipole method approximation in particle simulations, Int. J. Numer. Meth. Eng. (2008), Available at <http://arvix.org/abs/0809.1810>.

[38] N.R. Clarke, O.R. Tutty, Construction and validation of a discrete vortex method for the two-dimensional incompressible Navier–Stokes equations, Comput. Fluids 23 (1994) 751–783.

[39] M. Bar-Lev, H.T. Yang, Initial flow over an impulsively started circular cylinder, J. Fluid Mech. 72 (1975) 625–647.

[40] P. Ploumhans, G.S. Winckelmans, Vortex methods for high-resolution simulations of viscous flow past bluff bodies of general geometry, J. Comput. Phys. 165 (2000) 364–406.

[41] R. Buard, 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) 583–607.

[42] P. Koumoutsakos, A. Leonard, High-resolution simulations of the flow around an impulsively started cylinder using vortex methods, J. Fluid Mech. 296 (1995) 1–38.