ACADEMIC PRESS

Available online at www.sciencedirect.com

SCIENCE DIRECT®

Journal of Computational Physics 186 (2003) 212–229

JOURNAL OF COMPUTATIONAL PHYSICS

www.elsevier.com/locate/jcp

On the efficiency of a numerical method with periodic time strides for solving incompressible flows

E. Sanmiguel-Rojas, J. Ortega-Casanova, R. Fernandez-Feria *

Universidad de Malaga, E.T.S.Ingenieros Industriales, 29013 Malaga, Spain

Received 28 February 2002; received in revised form 11 December 2002; accepted 14 January 2003

Abstract

An explicit numerical method to solve the unsteady incompressible flow equations consisting on N small time steps Δt between each two much larger time steps $(\Delta t)_1$ is considered. The stability and efficiency of the method is first analyzed using the one-dimensional diffusion equation. It is shown that the use of a time step Δt slightly smaller than the critical one $(\Delta t)_c$ given by numerical stability allows to periodically take a much larger time step (stride) that speeds-up the advance in time in a numerical stable scheme. In particular, the stability analysis shows that for a given value of the stride $(\Delta t)_1$, there is an optimum value of the small time step for which the computational speed is the fastest (N is a minimum), being this speed significantly larger than the corresponding one for an explicit method using $(\Delta t)_c$ only. The efficiency of the method is discussed for different time discretization schemes. The numerical method is then used to solve a particular incompressible flow. It is shown that the method is significantly (about three times) faster than a standard explicit scheme, and yields the same time evolution of the flow (within spatial accuracy). Further, it is shown that a much more higher computational speed and efficiency is reached if one combines an implicit scheme for the periodic strides with the explicit small time steps. With this combination one can speed-up the computations in more than one order of magnitude with the same resolution.

1. Introduction

Explicit numerical methods are sometimes preferred to implicit ones to solve the incompressible Navier–Stokes equations owing to the high computational cost of an implicit method at each time step, particularly for multi-dimensional flows at moderate and high Reynolds numbers. However, explicit methods have severe stability constrains over the time step, which usually has to be several orders of magnitude smaller than the time step one would like to use in accordance with the resolution of the spatial grid. These constraints are particularly severe for low-Reynolds-numbers flows (see, for example [1,2]).

* Corresponding author.

0021-9991/03/$ - see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/S0021-9991(03)00049-4

To overcome these difficulties, several techniques have been used, such as spectral-projection, semi-implicit, and semi-Lagrangian methods (see, for example [3,4], and references given therein). In this paper, we introduce and analyze a conceptually simple approach consisting on an explicit method in which, along with the small time step Δt allowed by the numerical stability, a much longer time step $(\Delta t)_1$ is periodically taken to advance faster in time. These periodic strides, whose magnitude may be selected, for example, in such a way that the temporal and spatial accuracies are of the same order of magnitude, obviously introduce numerical instabilities in the explicit scheme. The method is then based in taking a sufficiently high number N of small time steps between strides that makes the overall method numerically stable in time. Of course, if the small time step Δt is equal to the maximum $(\Delta t)_c$ allowed by the numerical stability of the problem, any single time step $(\Delta t)_1 > \Delta t$ would make unstable the method ($N \to \infty$). However, we shall see that as Δt decreases slightly below $(\Delta t)_c$, the minimum value of small time steps N that stabilizes the method for a given $(\Delta t)_1$ decreases very fast, thus decreasing significantly the computation time to advance in physical time. For a given problem and a given numerical algorithm, there exists an optimum value of $\Delta t < (\Delta t)_c$ which makes the numerical computations the fastest. As we shall see, the speed of the numerical computations can be increased significantly.

To lay the foundations of the method, it is introduced in Section 2 using the one-dimensional diffusion equation. The stability, accuracy, and efficiency of the method is analyzed and checked with numerical stability results. In Section 3 the method is applied to an incompressible flow, the flow through a cylinder with a small hole at its bottom wall. It is shown that both the characteristics of the method and the increasing in the computational speed are similar to those found for the one-dimensional, and linear, case. In addition, it is found that the computational speed and the efficiency of the method can be increased even further by using an implicit scheme for the periodic strides.

2. Numerical method. Stability and efficiency

To introduce the method we use the one-dimensional diffusion equation for some magnitude $u(x,t)$, where x is the spatial coordinate and t the time, which in dimensionless form can be written as

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2}, \tag{1}$$

subjected to the boundary and initial conditions

$$u(0,t) = 0, \quad u(1,t) = 1, \quad u(x,0) = 0. \tag{2}$$

This equation will be solved numerically using an explicit finite difference scheme. In particular, we use in the next section a predictor–corrector scheme to advance in time (hereinafter referred to as PC), combined with centered differences in x. The results are then compared with those obtained by using other different discretization techniques.

2.1. Predictor–corrector scheme

Using a PC method with centered space differences, the discretized form of (1) becomes:
Predictor step

$$\frac{u_j^* - u^n}{\frac{1}{2}\Delta t} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}. \tag{3}$$

Corrector step

$$\frac{u_j^{n+1} - u^n}{\Delta t} = \frac{u_{j+1}^* - 2u_j^* + u_{j-1}^*}{(\Delta x)^2}. \tag{4}$$

As usual, the superscript indicates the time level, and the subscript the spatial grid point; thus, u_j^{n+1} is the discretized value of u at $x = j\Delta x$ and $t = (n+1)\Delta t$, while u_j^* is the intermediate approximation of u at the same grid point and time level. As is well known, this method is second-order accurate both in space and time (e.g. [5]).

In the description of the periodic time stride method that we propose here, the details of the stability analysis are important. To that end we use the von Neumann stability method (see, for example [6,7]): the errors distributed along the grid points x_j at a given time t are expanded as a finite Fourier series

$$E(t, x_j) = \sum_m b_m(t) e^{ik_m x_j}, \tag{5}$$

where k_m is the wave number corresponding to the mth Fourier mode. Since the equation is linear, each mode m satisfies Eqs. (3) and (4) separately. Thus, substituting each mode m of (5) into these equations, one finds that

$$b_m(t + \Delta t) = b_m(t)\left(1 - 4ds_m + 8d^2 s_m^2\right), \tag{6}$$

where

$$d = \frac{\Delta t}{(\Delta x)^2}, \tag{7}$$

and

$$s_m = \sin^2 \frac{k_m \Delta x}{2}. \tag{8}$$

The numerical method is stable if $|b_m(t + \Delta t)/b_m(t)| \leqslant 1$. Since both d and s_m are non-negative, this condition is satisfied if

$$\left(1 - 4ds_m + 8d^2 s_m^2\right) \leqslant 1 \tag{9}$$

(note that the quantity in brackets is always positive). This requirement imposes a maximum time step $(\Delta t)^{(m)}$ for each mode m. The general numerical stability of the explicit method is assured if the time step is less or equal than $(\Delta t)_c = \max_m[(\Delta t)^{(m)}]$. Since $0 \leqslant s_m \leqslant 1$, this maximum value corresponds to the mode with $s_m = 1$, and is given by $d = d_c = 1/2$, providing the well-known stability requirement [6]

$$\Delta t \leqslant (\Delta t)_c = \frac{1}{2}(\Delta x)^2. \tag{10}$$

In order to advance in time as faster as possible, given a spatial mesh size Δx, one usually selects a value Δt as close as possible to the critical one $(\Delta t)_c$. Since the present scheme has second-order accuracy in time, this means that the time accuracy is $O[(\Delta t)_c^2] = O[(\Delta x)^4]$, unnecessarily much less than the spatial accuracy, which is $O[(\Delta x)^2]$. Another possibility within the present explicit scheme, which is the one we analyze here, is to choose a time step Δt smaller than $(\Delta t)_c$, in such a way that one has some margin of numerical stability to periodically take a much larger time step $(\Delta t)_1$ that speeds-up the advance in time. We shall see that, for a given value of $(\Delta t)_1$, there is an optimum value of $\Delta t = (\Delta t)_o < (\Delta t)_c$ for which the overall numerical advance in time is the fastest, and significantly faster than just using $(\Delta t)_c$.

To that end we define the non-dimensional periodic stride

$$K = \frac{(\Delta t)_1}{(\Delta t)_c} > 1. \tag{11}$$

The value of K can be selected, for instance, in such a way that the order of magnitude of the spatial and temporal numerical accuracies coincide: $(\Delta t)_1 \sim \Delta x$, or $K \simeq 2/\Delta x$. But it can be smaller, or even larger, if one is only interested in finding out the steady-state solution. In the present analysis we shall treat K as a free, usually large, parameter. Between each two strides of magnitude $(\Delta t)_1$, a number N of much smaller time steps of magnitude Δt are taken (see Fig. 1). The number N has to be large enough for the method be numerically stable. According to Eq. (6), the repetition of cycles comprising N small steps and one stride is numerically stable if

$$(1 - 4ds_m + 8d^2 s_m^2)^N (1 - 2Ks_m + 2K^2 s_m^2) \leqslant 1 \tag{12}$$

for every mode m, where d, given by Eq. (7), is associated to the small time step. It is convenient to define the non-dimensional variable η, related to the small time step Δt by

$$\frac{\Delta t}{(\Delta t)_c} = 2d \equiv 1 - \eta \quad (0 \leqslant \eta \leqslant 1). \tag{13}$$

Thus, the minimum value N of small time steps of size η necessary to numerically stabilize a stride of magnitude K is, for each mode m,

$$N^{(m)} = -\frac{\ln(1 - 2Ks_m + 2K^2 s_m^2)}{\ln[1 - 2(1-\eta)s_m + 2(1-\eta)^2 s_m^2]}. \tag{14}$$

This value is plotted as a function of η for $K = 100$ and several values of s_m in Fig. 2. Since all modes are in principle present in the truncating errors of the numerical computations, for each η and K the minimum value of N for numerical stability corresponds to the maximum of $N^{(m)}$ as s_m is varied between zero and unity. As expected, for small η, i.e., for time steps near the critical one $(\Delta t)_c$, N is controlled by the mode corresponding to $s_m = 1$ (left continuous line in Fig. 2). However, the most unstable mode switches to $s_m = s_{mo}$ at $\eta = \eta_o$, with both η_o and s_{mo} depending on K. This mode s_{mo} is easily obtained by maximizing $N^{(m)}$ for η near unity:

$$s_{mo} = \frac{a}{K}, \tag{15}$$

where constant a is the root of

$$\frac{4a^2 - 2a}{1 - 2a + 2a^2} = \ln(1 - 2a + 2a^2), \quad a \simeq 2.7142. \tag{16}$$

$N^{(m)}(\eta)$ corresponding to s_{mo} is also plotted as a continuous line in Fig. 2. The value η_o is then obtained by the crossing of the curves $N^{(m)}(\eta)$ for $s_m = 1$ and s_{mo}

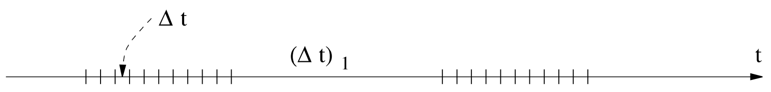

Fig. 1. Small and large time steps.

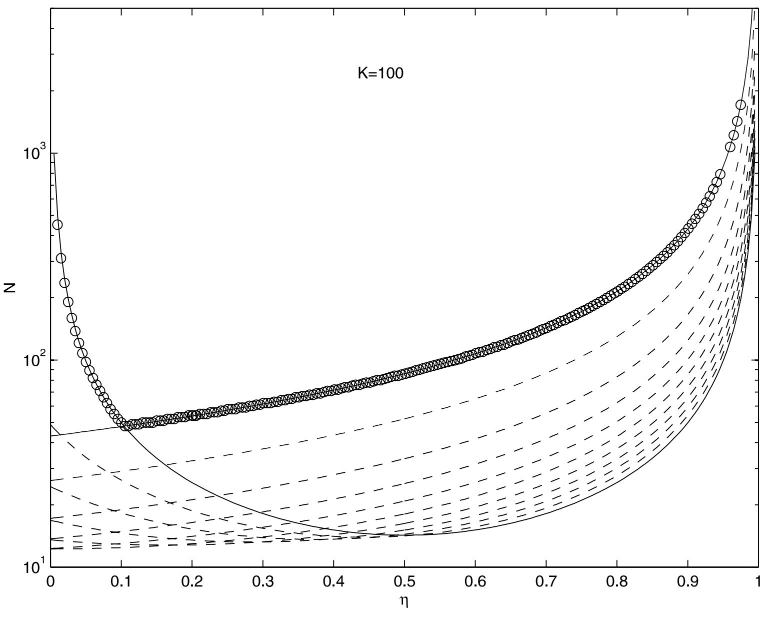

Fig. 2. Minimum value of the number of small time steps N for a predictor–corrector method with $K = 100$ for several values of s_m (1, .9, .8, .7, .6, .5, .4, .3, .2, .1, .0271) as a function of η. The continuous lines correspond to $s_m = 1$ (left) and to $s_m = a/K \simeq 0.0271$. The circles correspond to the minimum values of N for each η obtained numerically with $\Delta x = 0.02$.

$$\frac{\ln(1-2K+2K^2)}{\ln[1-2(1-\eta_o)+2(1-\eta_o)^2]} = \frac{\ln(1-2a+2a^2)}{\ln[1-2(1-\eta_o)a/K+2(1-\eta_o)^2(a/K)^2]}. \tag{17}$$

For large K, η_o can be approximated by

$$\eta_o \simeq \frac{2.3271}{3.3271+\frac{2K}{\ln(2K^2)}}. \tag{18}$$

We see that, for each value of K, there exists an optimum value of the small time step given by $\eta = \eta_o$ for which the number N of these small time steps needed to stabilize the long time step is a minimum. This minimum value is given by

$$N_{\min} = \frac{K\ln(1-2a+2a^2)}{2a(1-\eta_o(K))} \simeq 0.4279\frac{K}{1-\eta_o(K)}. \tag{19}$$

These results have been checked numerically by solving (1) and (2) with the predictor–corrector scheme (3) and (4) and $\Delta x = 2/K = 0.02$ ($K = 100$ in Fig. 2). The minimum value of steps N for the method to be numerically stable is then obtained as the size η of the small time step is varied between zero and unity. The results are plotted as circles in Fig. 2, where it is observed that they coincide with the predicted ones by the above theoretical stability analysis.

To evaluate the efficiency of the method we use the computational speed of each numerical cycle, V, defined as the physical time interval $N\Delta t + (\Delta t)_1$ divided by the computational time. This speed is then normalized with the computational speed V_o corresponding to the standard explicit method that uses only the critical time step $(\Delta t)_c$ to cover the same physical time interval

$$\frac{V}{V_o} = (1-\eta)\left(\frac{N+\frac{K}{1-\eta}}{N+1}\right) \simeq 1-\eta+\frac{K}{N}, \quad N \gg 1. \tag{20}$$

The first factor $(1-\eta)$ corresponds to the decreasing in the computational speed due to the smaller time step Δt used in relation to $(\Delta t)_c$, while the second one (between large brackets) corresponds to the increasing in the computational speed due to the periodic long stride. As observed in Fig. 3, where V/V_o is plotted as a function of η for $K = 100$, and for the values of $N(\eta)$ given in Fig. 2 for $s_m = 1$ and $s_m = s_{mo} = a/K$, the overall effect is to speed-up the computation in relation to the single-step explicit method with $(\Delta t)_c$ (i.e., $V/V_o > 1$), except for very small time steps ($\eta \to 1$). The function $V(\eta)/V_o$ corresponding to $s_{mo} = a/K$ is a straight line given by

$$\frac{V}{V_o} = \left[1+\frac{2a}{\ln(1-2a+2a^2)}\right](1-\eta) \simeq 3.3271(1-\eta), \tag{21}$$

which is independent of K. For $s_m = 1$, V/V_o is, approximately, a parabola. We see in Fig. 3 that the computational speed would have been increased more than seven times (for $K = 100$) using $\eta = 0.5$ if the mode corresponding to $s_m = 1$ would have remain the most unstable for all values of η. However, as we have seen, this is not the case for $\eta > \eta_o$ ($\simeq 0.104$ for $K = 100$), where $s_{mo}(K = 100) \simeq 0.0271$ is the most unstable mode. Thus, the maximum computational speed for $K = 100$ is $V_{\max} \simeq 3V_o$, corresponding to $\eta = \eta_o$. For large K, $\eta_o(K)$ can be easily obtained because it is small, so that it can be approximated by the intersection of two straight lines, that given by (21) (which does not depend on K), and the corresponding one for $s_m = 1$ near $\eta = 0$. The result is given in Eq. (18).

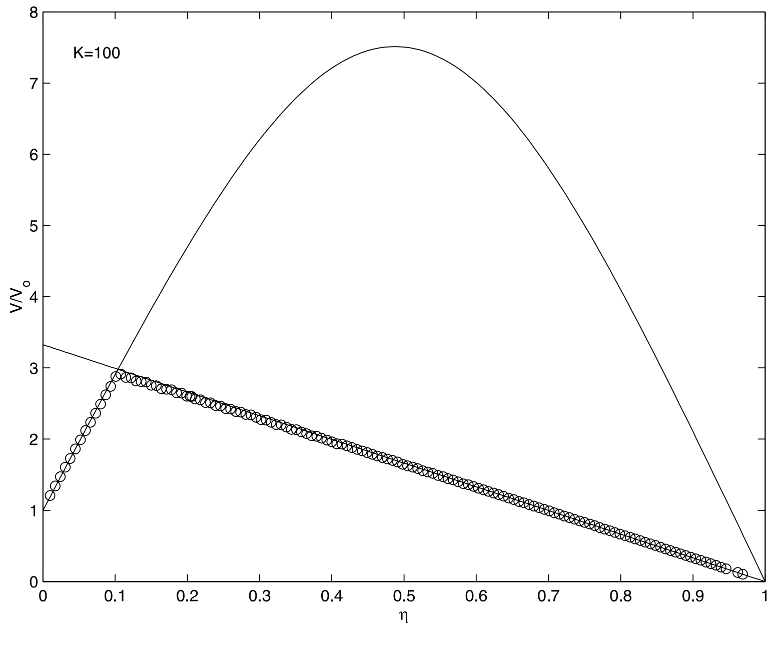

Fig. 3. Normalized computational speed for the same case of Fig. 2 with $s_m = 1$ (parabola) and $s_m = a/K \simeq 0.0271$ (straight line). The same results obtained numerically with $\Delta x = 0.02$ are plotted as circles.

2.2. Efficiency of the method

We have seen that, given a discretization scheme and a non-dimensional long time step K, there exists an optimum value of the non-dimensional small time step η_o for which the computational advance in time is the fastest; that is to say, the number N of small time steps to stabilize the long time step is a minimum. Similar results can be found using other different discretization techniques. Fig. 4 shows these values of $\eta_o(K)$ and $N_{\min}(K)$ for the PC scheme just described, and compare them with the corresponding values obtained with a forward in time and centered in space differences scheme (FTCS for short), which is only first-order accurate in time [5]. For large K, $\eta_o(K)$ can be approximated by (18) for the PC scheme, and by

$$\eta_o \simeq \frac{2a_1 - 1}{2a_1 + \frac{2K}{\ln(2K)}}, \quad a_1 \simeq 2.956, \tag{22}$$

for the FTCS scheme, while the minimum values of N are approximated by $N_{\min} \simeq 0.4297K/(1-\eta_o)$ and $N_{\min} \simeq 0.2785K/(1-\eta_o)$, respectively. Fig. 5 shows the corresponding normalized computational speeds. It is observed that $V_{\max}/V_o$ grows with K and reaches an asymptote as $K \to \infty$ in both cases. Actually, as K increases, V/V_o increases (and η_o decreases) along the straight lines corresponding to $s_m = s_{mo}$ [Eq. (21) for the PC scheme and $V/V_o \simeq 4.5911(1-\eta)$ for the FTCS scheme], so that the highest computational speed is given by the slope of these lines, which are, approximately, 3.3271 and 4.5911 for the PC and FTCS schemes, respectively. Since the reference speed V_o of the FTCS scheme is twice V_o of the PC scheme, this means that the highest computational speed for the FTCS method is almost three times the maximum value for the PC method, i.e., about nine times faster than the standard explicit method.

However, in practice, this is not so because the value of K will usually be selected in such a way that the temporal accuracy is of the same order of magnitude than the spatial one. Since the FTCS method is first-order accurate in time and second-order in space, the corresponding value of $K = (\Delta x)^2/(\Delta t)_c$ is always 2, independently of the value of Δx. In the PC scheme discussed in Section 2.1, which is second-order in time,

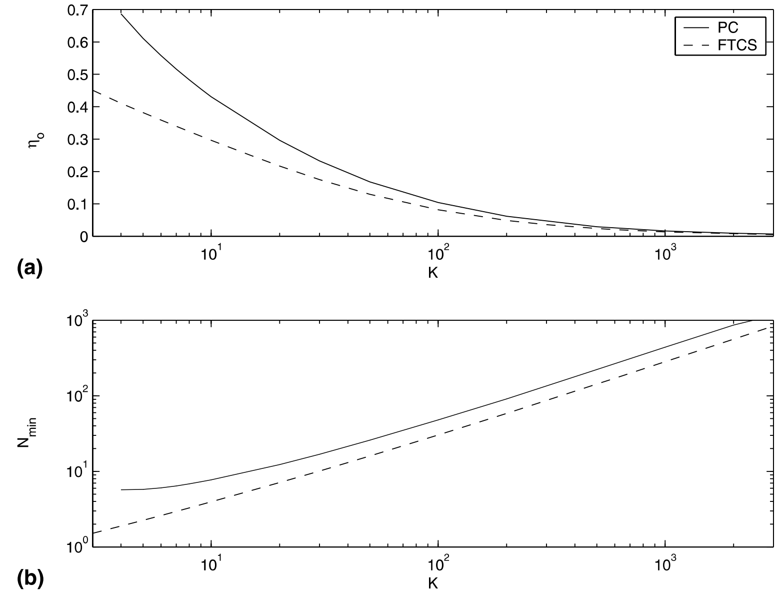

Fig. 4. η_o vs. K (a) and $N_{\min}$ vs. K (b), for the PC discretization scheme (continuous lines) and FTCS scheme (dashed lines).

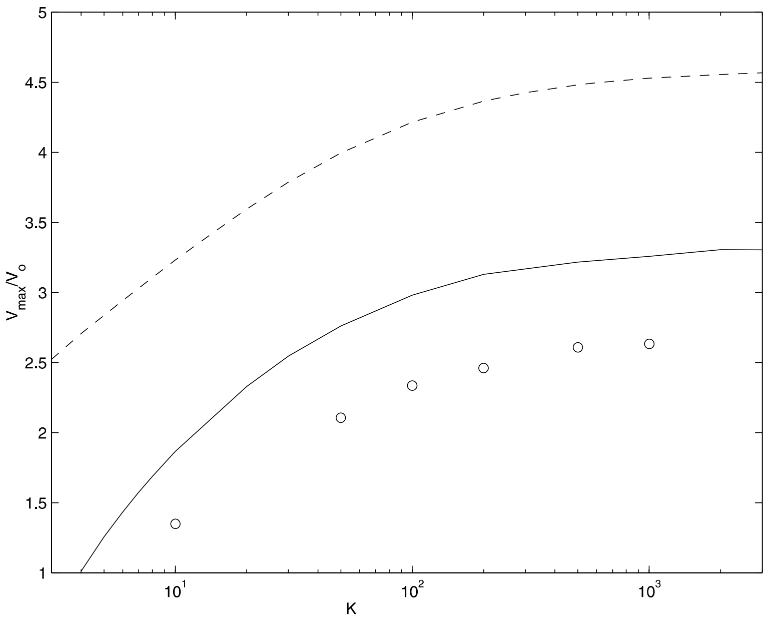

Fig. 5. Maximum of the computational speed as a function of K for the PC discretization scheme (continuous line), FTCS scheme (dashed line), and PC–RK scheme (circles).

the corresponding value of K given by the spatial accuracy is $K = \Delta x/(\Delta t)_c = 2/\Delta x$, so that one can increase K by decreasing Δx. Thus, for the same spatial and temporal resolutions, the use of the PC scheme can be faster than the FTCS if one selects a sufficiently high value of K (small value of Δx).

Higher values of K for a given Δx yielding time accuracy in accordance with the spatial resolution may be obtained using a numerical scheme of higher order for the long time steps. In Fig. 5 we have also included the maximum of the computational speed of a predictor–corrector scheme for the small time steps combined with a fourth-order Runge–Kutta method (e.g. [5]) for the strides (PC–RK scheme; V_o is approximately the same than in the PC case because it is mainly controlled by the numerical method used in the small time steps, which is the same in both cases). Given K, the normalized computational speed is lower, but the value of K that equates time and space accuracies is now $2/(\Delta x)^{3/2}$, which allows a larger value of K for a given Δx. For instance, for $\Delta x = 0.02$, the corresponding values for K are 2, 100, and 707 for FCTS, PC, and PC–RK schemes, respectively. The corresponding speed-up factors are, approximately, $V/V_o = 2.8, 3$, and 2.5, respectively.

3. Application to an incompressible flow

With the diffusion equation considered in the previous section one may predict theoretically, and then check numerically, the stability of the numerical method. However, where the method has important advantages is in solving unsteady two-dimensional (2D) and three-dimensional (3D) incompressible flows, where the numerical stability severely constrains the size of the time step in explicit methods, and the reduction of CPU time may be essential for the numerical method to be of practical use. This circumstance is particularly severe in flows with very different length scales, because the time step for numerical stability may be extremely small, even using an implicit or semi-implicit method. For this reason, in this section we

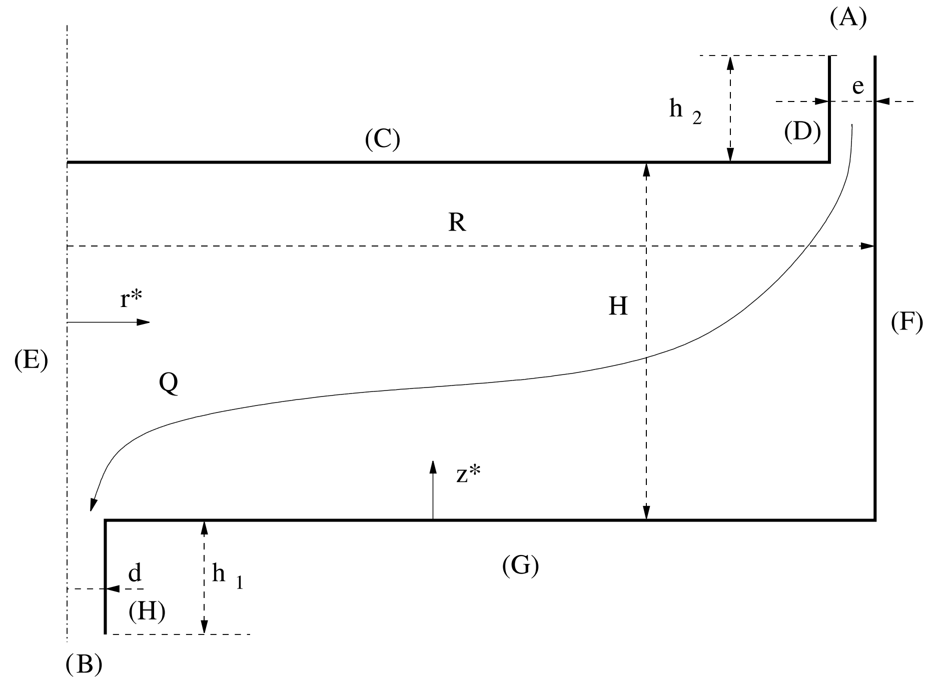

Fig. 6. Sketch of the geometry of a confined sink flow.

apply the stride-method to a confined axisymmetric sink flow in the cylindrical geometry shown in Fig. 6, where the flow exits the cylindrical container through an orifice (sink) at the bottom wall of diameter d much smaller that the radius R of the cylinder (see next section and Appendix A for the details of the flow). For small values of d/R, the maximum value of the time step in an explicit (and even in an implicit or semi-implicit) method is so small that the CPU time to reach a steady state becomes prohibitive. We shall see that the dependence of the computational speed V/V_o upon the non-dimensional small time step has a similar form to that shown in Fig. 3, so that, for each value of K, there exists an optimum value of $\eta = \eta_o(K)$ for which the computational speed is a maximum. In complex flows like this, $(\Delta t)_c$ and $(V_{\max}/V_o)(\eta)$ have to be obtained, in general, numerically. However, the maximum of the computational speed for each K is easily obtained because, as we shall see, the function $V_{\max}(\eta)/V_o$ can be approximated (as in Fig. 3) by two straight lines.

3.1. Flow in a confined sink flow

We consider in this section the flow inside the cylindrical container sketched in Fig. 6. A flow rate Q of an incompressible fluid of kinematic viscosity ν enters the cylindrical tank through the circular ring of width e between the upper endwall and the cylinder. The fluid exits through a small orifice (the sink) of diameter d centred at the bottom endwall.

Cylindrical polar coordinates (r^*, θ, z^*), with velocity field (u^*, v^*, w^*), are used, where the asterisk superscripts denote dimensional quantities. The corresponding dimensionless variables are

$$r = \frac{r^*}{R}, \quad z = \frac{z^*}{H}, \tag{23}$$

$$u=\frac{u^{*}2\pi RH}{Q},\quad v=\frac{v^{*}2\pi RH}{Q},\quad w=\frac{w^{*}2\pi R^{2}}{Q},\tag{24}$$

where R is the radius of the cylinder and H is the height of the cylindrical cavity. To model the incompressible and axisymmetric flow we use the stream function–vorticity–circulation formulation (see, for instance [8]), where stream function Ψ^{*}, vorticity η^{*}, and circulation (actually, angular momentum) Γ^{*} are defined, respectively, through

$$u^{*}=-\frac{1}{r^{*}}\frac{\partial\Psi^{*}}{\partial z^{*}},\quad w^{*}=\frac{1}{r^{*}}\frac{\partial\Psi^{*}}{\partial r^{*}},\tag{25}$$

$$\eta^{*}=[\nabla^{*}\wedge v^{*}]_{\theta}=\frac{\partial u^{*}}{\partial z^{*}}-\frac{\partial w^{*}}{\partial r^{*}},\tag{26}$$

$$\Gamma^{*}=r^{*}v^{*}.\tag{27}$$

With this formulation, the continuity equation is satisfied identically, and the three equations to be solved are the azimuthal components of the momentum and the vorticity equations, together with the definition (26) of η^{*}. These equations, and the boundary conditions, are given in dimensionless form in Appendix A.

Fig. 7(a) shows the streamlines corresponding to the steady state for $Re=250$, which is the Reynolds number selected here to compare the different numerical schemes [Re is defined in Eq. (A.4)]. In fact, we are going to consider this steady state as the initial condition for the unsteady flow with circulation used here to compare the different numerical schemes: After the steady state without rotation has been reached ($t=0$), the upper wall starts rotating with an angular velocity Ω. Thus, for $t\geqslant 0$, the circulation Γ at the upper wall [(C) and (D) in Fig. 6] and at the inlet (A) is not zero, but given by

$$\Gamma=\gamma\frac{8}{\delta^{2}\Delta}r^{2}\quad\text{at } z=1,\ 0\leqslant r\leqslant 1-\epsilon\quad\text{and}\quad 1\leqslant z\leqslant 1+z_{2},\ r=1-\epsilon,\tag{28}$$

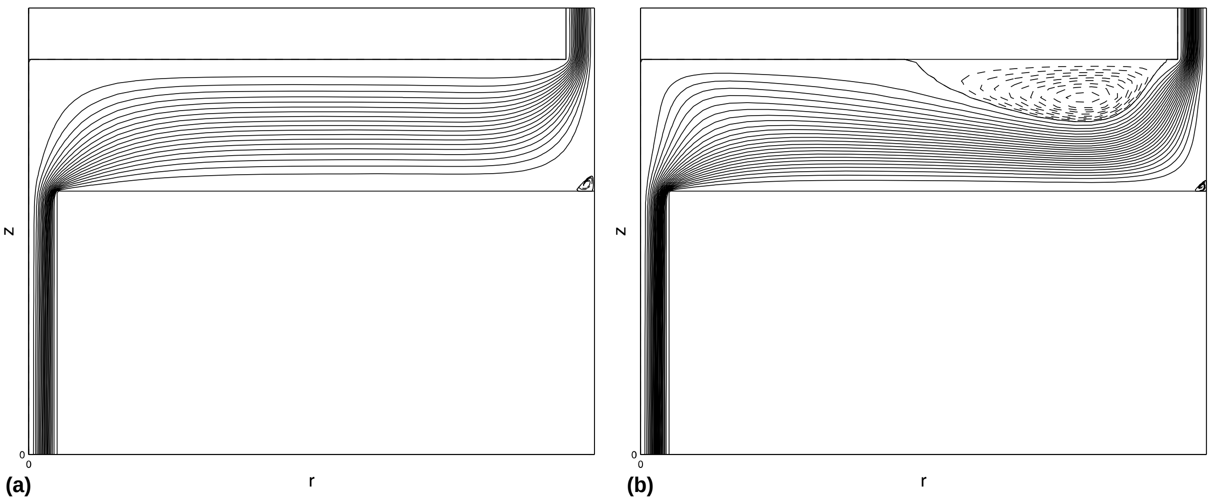

Fig. 7. Steady-state streamlines corresponding to a flow without rotation (a) and with rotation of the upper wall (b), for $Re=250$ and $\gamma=0.1$. 20 equally spaced streamlines between 0 and -1 are plotted with solid lines. Broken lines correspond to regions with flow recirculation, with values of Ψ outside the interval $[-1,0]$.

and

$$\Gamma = \gamma \frac{8}{\delta^2 \varDelta} \frac{(1-\epsilon)^2}{1-(1-\epsilon)^2}(1-r^2) \quad \text{at } z = 1 + z_2, \ 1-\epsilon \leqslant r \leqslant 1, \tag{29}$$

which substitute the boundary conditions (A.10), (A.12), and (A.6) for Γ, respectively. γ is a dimensionless number characterizing the azimuthal velocity of the upper disk,

$$\gamma \equiv \frac{\Omega R}{4Q/\pi d^2}. \tag{30}$$

Fig. 7(b) shows the streamlines corresponding to the new steady state for $Re = 250$ when $\gamma = 0.1$. Both steady states plotted in Fig. 7, the initial condition of Fig. 7(a) and the final condition plotted in Fig. 7(b), have been obtained using an explicit predictor–corrector method with 2070 nodes on a stretched grid that concentrates them at the inflow and at the outflow regions, and a single time step given by its critical value $\varDelta t = (\varDelta t)_c$. The PC explicit scheme can be summarized as follows (see Appendix A):

$$\Gamma^* = \Gamma^n + \frac{\varDelta t}{2} F(\Psi^n, \Gamma^n), \quad \eta^* = \eta^n + \frac{\varDelta t}{2} G(\Psi^n, \Gamma^n, \eta^n), \quad \nabla^2 \Psi^* = -r\eta^*, \tag{31}$$

$$\Gamma^{n+1} = \Gamma^n + \varDelta t F(\Psi^*, \Gamma^*), \quad \eta^{n+1} = \eta^n + \varDelta t G(\Psi^*, \Gamma^*, \eta^*), \quad \nabla^2 \Psi^{n+1} = -r\eta^{n+1}, \tag{32}$$

where second-order spatial discretizations for the spatial derivatives in F and G are used. Thus, the method is second-order accurate both in space and time [8]. In the computational plane $(0 \leqslant x \leqslant 1, 0 \leqslant y \leqslant 1)$, the 2070 nodes are uniformly distributed with $\varDelta x = 0.0125$ and $\varDelta y = 0.0196$, so that the spatial errors are of the order $(\varDelta x)^2 + (\varDelta y)^2 \sim 6 \times 10^{-4}$. It should be noted here that the steady-state streamlines plotted in Fig. 7 are undistinguishable from those obtained using the significantly faster stride-method described in the next sections.

In the explicit computations starting from the initial condition of Fig. 7(a) to reach the new steady state of Fig. 7(b), the critical value of $\varDelta t$ given by numerical stability is $(\varDelta t)_c \simeq 2.3 \times 10^{-6}$, which is almost four orders of magnitude smaller than the spatial grid size. The CPU time to reach the steady-state conditions of Fig. 7(b) was about 57 min in a Silicon Origin 2000. (Note that we have selected a relatively low value of the Reynolds number to reach the steady state in a reasonable CPU time. If, for instance, we had used $Re = 2000$, the number of nodes to reach a similar spatial resolution would have increased to about 10 000, $(\varDelta t)_c$ would be around 10^{-7}, and the CPU time to reach the steady state would rise to more than seven days, making very costly the comparison between the different numerical computations given below.) To have an idea of the azimuthal motion corresponding to the steady state given by the streamlines of Fig. 7(b), we have plotted in Fig. 8 several radial profiles of the azimuthal velocity v at different z-levels inside the cylinder. It is observed that the swirl is concentrated at the sink exit. In particular, the maximum value of v is reached at the point $z = 0$ and $r \simeq 0.025$. Since the temporal and spatial gradients are consequently the largest at this point, we have chosen the temporal evolution of v there to compare the performance of the different numerical schemes given below.

3.2. Explicit scheme for the long time steps

The stride-method discussed in Section 2 has been used to solve this problem for the same Re and spatial grid of Fig. 7, and for several values of K [the PC explicit numerical scheme (31) and (32) is used].

For each value of K, the functions $N(\eta)$ and $(V/V_o)(\eta)$ are very similar to those plotted in Figs. 2 and 3. Fig. 9 shows $(V/V_o)(\eta)$ for $K = 1000$, which is the value that approximately yields the same temporal

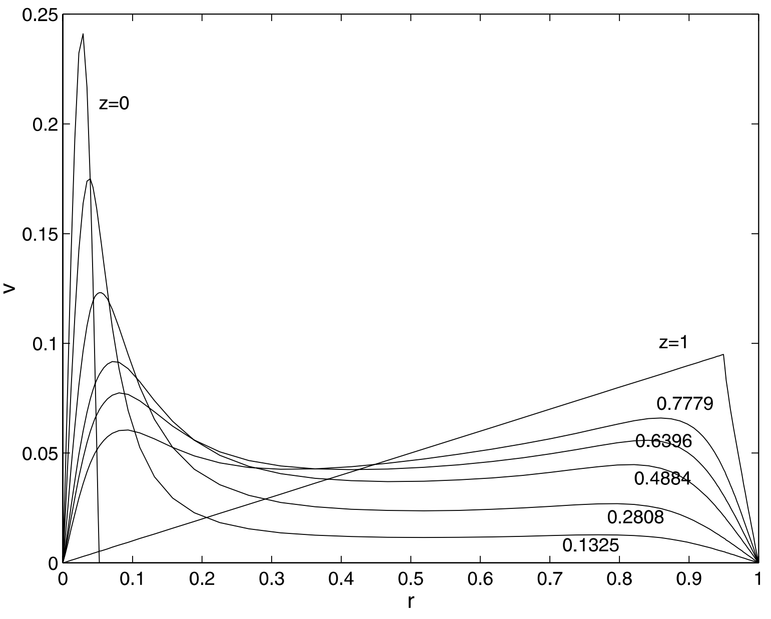

Fig. 8. Radial velocity profiles at different values of z corresponding to the steady state of Fig. 9(b).

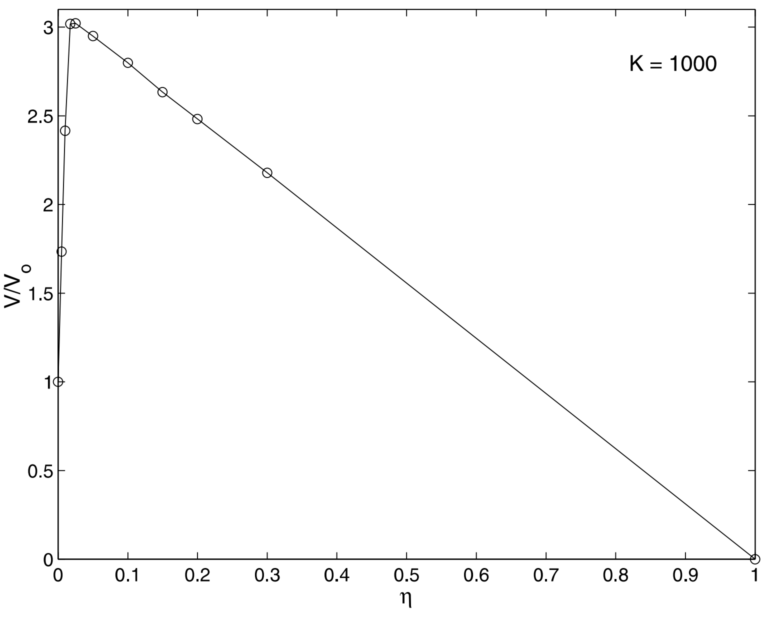

Fig. 9. Normalized values of the computational speed as a function of η for the flow in a confined sink flow using a PC scheme with $K = 1000$. This function is approximated by two straight lines, the crossing of which yields the maximum computational speed.

accuracy than the spatial one. It has a maximum at $\eta_o \simeq 0.025$. Like in the much simpler example of Section 2, this maximum can approximately be obtained by the crossing of two straight lines, one for small η passing through $(V/V_o)(\eta = 0) = 1$, and the other one coming from $(V/V_o)(\eta = 1) = 0$ (remarkably, this

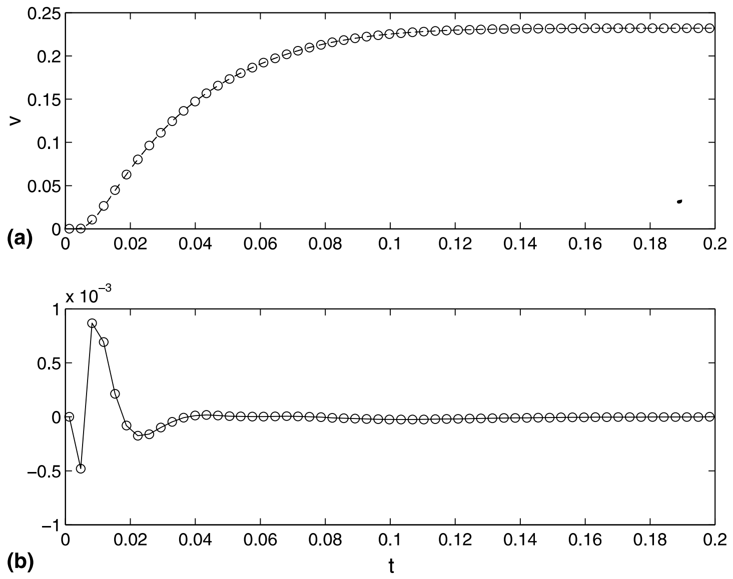

Fig. 10. (a) Time evolution of the azimuthal velocity at $(r = 0.025, z = 0)$ for $K = 1000$. The circles correspond to last points just before the strides. The dashed line is obtained using a single-time step explicit method with $(\Delta t)_c$. (b) Difference between the two numerical solutions.

last one is exactly a straight line in Fig. 9, as in the linear problem of Section 2). Therefore, the optimum value of the small time step for each K, $\eta_o(K)$, can be easily obtained from the computation of just a few points of these two straight lines for each K. It is observed that, for the value of K considered, the computation using η_o is a bit more than three times faster than with the standard explicit scheme.

The time evolution of the azimuthal velocity v at the point $(r = 0.025, z = 0)$ (remember that at this point the time and spatial gradients of v are the largest, see Fig. 8), obtained with $K = 1000$ and the corresponding optimum time step η_o, is shown in Fig. 10(a) together with the results obtained with a single-time step explicit method with $(\Delta t)_c$. It is observed that both time evolutions practically coincide. Actually, the difference between both curves, shown in Fig. 10(b), is always smaller than 10^{-3}, and therefore within the order of magnitude of the second-order spatial errors. The *only* difference is that with the single-step explicit method the steady state (which is assumed at $t = 0.2$) is reached after 57 min of CPU time, while with the stride-method the same conditions are reached just after 19 min of CPU time.

3.3. Implicit scheme for the long time steps

In nonlinear problems such as those governed by the incompressible Navier–Stokes equations, implicit methods are not always unconditionally stable like in the linear diffusion equation (e.g. [1]). Thus, the computational cost of an implicit method is even higher, making prohibitive its use in unsteady complex flows for moderate and high Reynolds numbers. For example, in the present example of a confined sink flow, we find that the maximum time step for $Re = 250$ using an implicit Crank–Nicolson scheme with two iterations (see Appendix A) is $1.453 \times (\Delta t)_c$, where $(\Delta t)_c$ is the maximum time step of the PC explicit method (31) and (32) for the same Re. Since the CPU time for each Δt is about 10 times larger, the semi-implicit method is about seven times slower than the explicit one.

However, since the implicit scheme is more stable numerically, the use of an implicit method for the long time steps may have some advantages over the explicit one because the smaller number N of small explicit

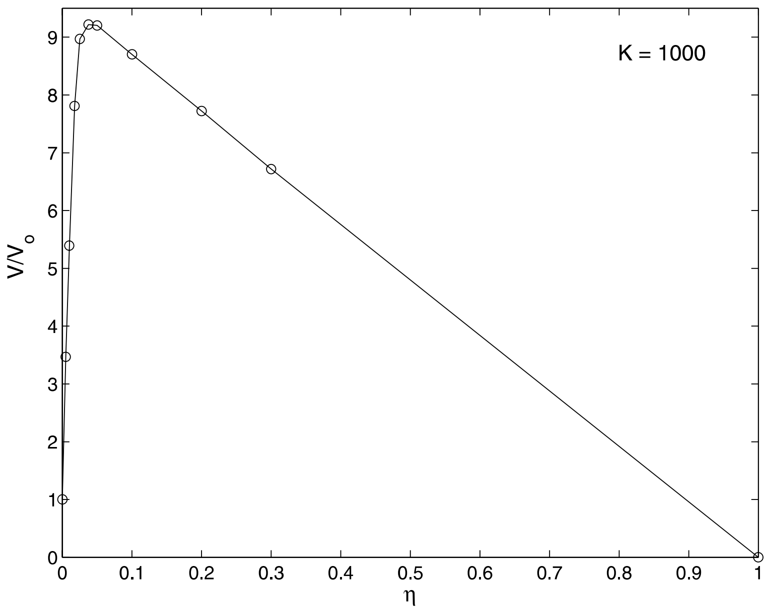

Fig. 11. Normalized values of the computational speed as a function of η for the flow in a confined sink flow using an explicit PC scheme for the small time steps and an implicit Crank–Nicolson scheme for the strides, for $K = 1000$. This function is approximated by two straight lines, the crossing of which yields the maximum speed at $\eta = \eta_o(K)$.

time steps needed to stabilize the method may compensate the larger computational cost of each implicit periodic stride, making the overall computational time smaller. We show in this section that this is the case. Actually, we shall see that the computational speed using a combination of explicit small time steps with implicit long time steps may be about one order of magnitude larger than the computational speed with the single-step explicit method.

For each value of the non-dimensional stride K, the functions $N(\eta)$ and $(V/V_o)(\eta)$, using the implicit scheme (A.15) and (A.16) for the strides, and the explicit scheme (31) and (32) for the small time steps, are qualitatively similar to those plotted in Figs. 3 and 9, existing an optimum value of the small time step $\eta = \eta_o$ for which V/V_o is a maximum for each K. This can be observed in Fig. 11 for $K = 1000$ and $Re = 250$. As before, this maximum can be approximately obtained by the crossing of two straight lines, so that the optimum value of the small time step for each K, $\eta_o(K)$, can be easily determined from the computation of just a few points of these two straight lines. Note that the normalization factor V_o is the same used in the preceding section, i.e., the computational speed of an explicit method with a single time step given by $(\Delta t)_c$. However, to obtain the computational speed [see Eq. (20)] in the present case, we have taken into account that the computational time of an implicit stride is $\alpha \simeq 10$ times larger than the CPU time of the small explicit step:

$$\frac{V}{V_o} = (1-\eta)\left(\frac{N + \frac{K}{1-\eta}}{N+\alpha}\right). \tag{33}$$

Comparing Figs. 9 and 11 it is seen that now η_o ($\simeq$0.0375 for $K = 1000$) is a little larger ($(\Delta t)_o$ is smaller). But, what is more important, the computational speed of the present scheme is about three times larger than that of the preceding section, and almost 10 times faster than the single-step explicit scheme (remember that this is so even after taking into account the correction α for the slower implicit strides). This is due to the much less number of small time steps N needed to stabilize the long time steps when an implicit

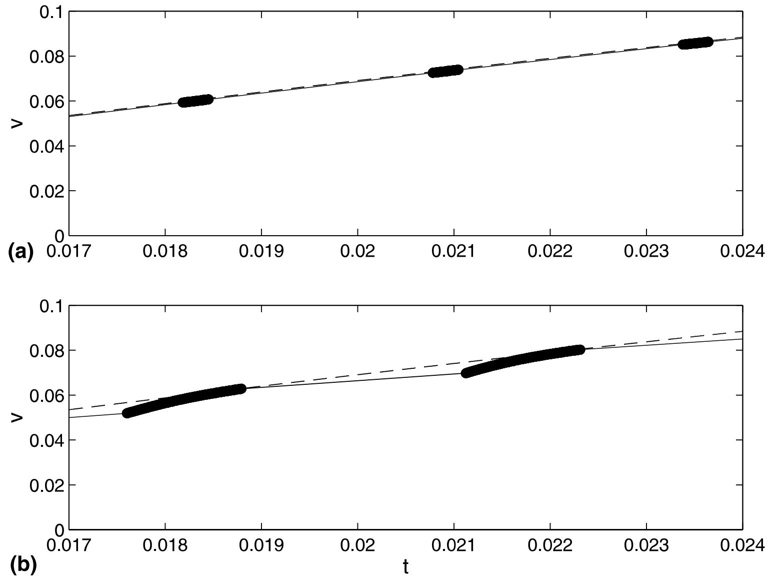

Fig. 12. Comparison between the details of the time evolution of v at the same point of Fig. 10 using the implicit scheme for the strides (a) and the explicit one (b), both with $K = 1000$. The circles correspond to the small time steps and the continuous lines to the strides. The dashed lines show the time evolution obtained using a single-time step explicit method with $\Delta t = (\Delta t)_c$.

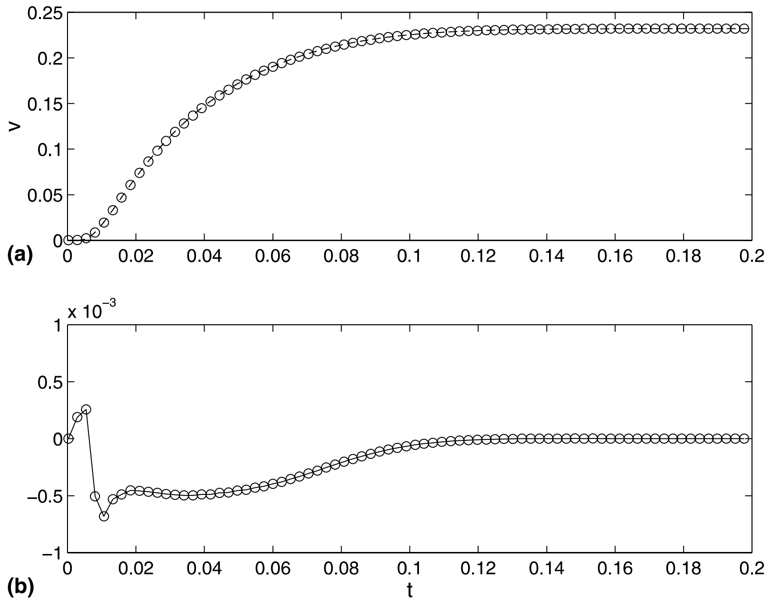

Fig. 13. (a) Time evolution of the azimuthal velocity at $(r = 0.025, z = 0)$ for $K = 1000$ using a Crank–Nicolson method for the strides. The circles correspond to last points just before the strides. The dashed line is obtained using a single-time step explicit method with $(\Delta t)_c$. (b) Difference between the two numerical solutions.

scheme is used for the strides, which more than compensates the larger computational time of the implicit stride. Fig. 12 illustrates this effect for $K = 1000$. It is seen that at least $N = 490$ small (explicit) time steps are needed to numerically stabilize the explicit stride for this value of K (Fig. 12(b)), while just $N = 110$ small (explicit) time steps are enough to stabilize the implicit stride for the same value of K (Fig. 12(a)).

Fig. 13(a) shows the complete time evolution of the azimuthal velocity v using the Crank–Nicolson scheme for the strides, at the same point and conditions of Fig. 10, and compares it with that obtained with the single-step explicit method. Again, both time evolutions practically coincide, but while that obtained with the explicit method takes 57 min of CPU time to reach the steady state (i.e., $t = 0.2$), the one computed with the stride method takes just 6 min of CPU time. Actually, the difference between both computed time evolutions, plotted in Fig. 13(b), is even smaller than that obtained with an explicit scheme for the strides (Fig. 10(b)), and it is within the spatial accuracy of the scheme.

4. Conclusions

With the aim of speeding-up explicit schemes for solving the incompressible Navier–Sokes equations, we present in this paper a method that advances in time with long time steps (strides) intercalated in much smaller time steps. The magnitude of the strides may be selected in accordance with the spatial resolution to match temporal and spatial accuracies. The method is based in the fact that the use of a Δt just slightly smaller than the one allowed by the numerical stability in an explicit scheme, $(\Delta t)_c$, yields enough stability margin to periodically take a much larger time step that speeds-up the computations in a numerically stable scheme. Furthermore, for each value of the relative magnitude of the stride, there is an optimum value of the small time step $(\Delta t)_o$ for which the computational speed is a maximum, which is always significantly larger than the computational speed without strides. These properties have been shown theoretically, and then checked numerically, using a linear diffusion problem. The application of the method to an incompressible flow has shown that the stability properties, and the speeding-up of the numerical method, for the nonlinear Navier–Stokes equations are very similar to those found theoretically for the linear equation. In particular, we give a simple procedure to obtain the optimum value of Δt for which the CPU time is a minimum, given the magnitude of the stride. Finally, we have shown that the use of an implicit scheme for the periodic strides between the explicit small time steps may speed-up the computations by a factor of order 10. In all these cases, the temporal evolutions obtained numerically are practically the same.

We think that where this method will be particularly useful is in the numerical simulation of three-dimensional incompressible flows at moderate and high Reynolds numbers and complex geometries, where the CPU time restrictions are more severe. Actually, in the example given in the preceding section, but for $Re = 2000$, we have been able to reduce the CPU time from more than a week to less than 15 h in a Silicon Origin 2000. It should be noted that the method is compatible with the parallelization of the code, so that the computational speed should be multiplied by the same factor related to the number of processors than in a standard explicit method.

Acknowledgements

This research has been supported by the Ministerio de Ciencia y Tecnologia of Spain (Grant No. BFM00-1323).

Appendix A

The non-dimensional equations governing the flow considered in Section 3 are the following:

$$\frac{\partial \Gamma}{\partial t} = \frac{1}{r}\frac{\partial \Psi}{\partial z}\frac{\partial \Gamma}{\partial r} - \frac{1}{r}\frac{\partial \Psi}{\partial r}\frac{\partial \Gamma}{\partial z} + \frac{8}{Re\,\Delta\delta}\left[\Delta^2\left(\frac{\partial^2 \Gamma}{\partial r^2} - \frac{1}{r}\frac{\partial \Gamma}{\partial r}\right) + \frac{\partial^2 \Gamma}{\partial z^2}\right], \tag{A.1}$$

$$\frac{\partial \eta}{\partial t} = \frac{1}{r}\frac{\partial \Psi}{\partial z}\frac{\partial \eta}{\partial r} - \frac{1}{r}\frac{\partial \Psi}{\partial r}\frac{\partial \eta}{\partial z} - \frac{\eta}{r^2}\frac{\partial \Psi}{\partial z} + \frac{\Gamma}{r^3}\frac{\partial \Gamma}{\partial z} + \frac{8}{Re\,\Delta\delta}\left[\Delta^2\left(\frac{\partial^2 \eta}{\partial r^2} + \frac{1}{r}\frac{\partial \eta}{\partial r} - \frac{\eta}{r^2}\right) + \frac{\partial^2 \eta}{\partial z^2}\right], \tag{A.2}$$

$$\Delta^2\left(\frac{\partial^2 \Psi}{\partial r^2} - \frac{1}{r}\frac{\partial \Psi}{\partial r}\right) + \frac{\partial^2 \Psi}{\partial z^2} = -r\eta, \tag{A.3}$$

where

$$Re = \frac{4Q}{\pi \nu d} \tag{A.4}$$

is the Reynolds number based on the flow properties at the sink exit and

$$\Delta = \frac{H}{R}, \quad \delta = \frac{d}{R} \tag{A.5}$$

are the aspect ratio of the cylinder and the dimensionless radius of the sink, respectively. Note that Ψ, η, and Γ are made dimensionless with $Q/2\pi$, $Q/2\pi RH^2$, and $Q/2\pi H$, respectively. The characteristic time used in the above adimensionalization is $2\pi HR^2/Q$.

These equations are solved with the following boundary conditions (see Fig. 6 for the geometry): at the entrance ((A) in Fig. 6), we assume a Poiseuille axial velocity profile together with $u = v = 0$:

$$\Psi = -\frac{1}{A}\Big(2[r^2 - (1-\epsilon)^2] - [r^4 - (1-\epsilon)^4] - 2K[r^2 \ln r - (1-\epsilon)^2 \ln(1-\epsilon)] + K[r^2 - (1-\epsilon)^2]\Big),$$

$$\eta = -\frac{4\Delta^2}{A}\left[2r + \frac{K}{r}\right], \quad \Gamma = 0 \quad \text{at } z = 1 + z_2,\ 1-\epsilon \leqslant r \leqslant 1, \tag{A.6}$$

where

$$\epsilon = \frac{e}{R}, \quad z_2 = \frac{h_2}{H}, \quad A = 1 - (1-\epsilon)^4 + \frac{[1-(1-\epsilon)^2]^2}{\ln(1-\epsilon)}, \quad K = \frac{1-(1-\epsilon)^2}{\ln(1-\epsilon)}. \tag{A.7}$$

At the sink exit ((B) in Fig. 6), the velocity profiles are assumed to be independent of z:

$$\frac{\partial^2 \Psi}{\partial z^2} = 0, \quad \eta = -\Delta^2\left[\frac{1}{r}\frac{\partial^2 \Psi}{\partial r^2} - \frac{1}{r^2}\frac{\partial \Psi}{\partial r}\right], \quad \frac{\partial \Gamma}{\partial z} = 0 \quad \text{at } z = -z_1 \equiv -\frac{h_1}{H},\ 0 \leqslant r \leqslant \delta/2. \tag{A.8}$$

At the axis of symmetry ((E) in Fig. 6), we have

$$\Psi = \eta = \Gamma = 0 \quad \text{at } -z_1 \leqslant z \leqslant 1,\ r = 0. \tag{A.9}$$

Finally, at the solid walls ((C), (D), (F)–(H) in Fig. 6) the velocity vanishes. Thus,

$$\Psi = 0, \quad \eta = -\frac{1}{r}\frac{\partial^2 \Psi}{\partial z^2}, \quad \Gamma = 0 \quad \text{at } z = 1,\ 0 \leqslant r \leqslant 1-\epsilon, \tag{A.10}$$

$$\Psi = -1, \quad \eta = -\frac{1}{r}\frac{\partial^2 \Psi}{\partial z^2}, \quad \Gamma = 0 \quad \text{at } z = 0,\ \delta/2 \leqslant r \leqslant 1, \tag{A.11}$$

$$\Psi = 0, \quad \eta = -\Delta^2 \frac{1}{r}\frac{\partial^2 \Psi}{\partial r^2}, \quad \Gamma = 0 \quad \text{at } 1 \leqslant z \leqslant 1 + z_2, \ r = 1 - \epsilon, \tag{A.12}$$

$$\Psi = -1, \quad \eta = -\Delta^2 \frac{1}{r}\frac{\partial^2 \Psi}{\partial r^2}, \quad \Gamma = 0 \quad \text{at } 0 \leqslant z \leqslant 1 + z_2, \ r = 1, \tag{A.13}$$

$$\Psi = -1, \quad \eta = -\Delta^2 \frac{1}{r}\frac{\partial^2 \Psi}{\partial r^2}, \quad \Gamma = 0 \quad \text{at } -z_1 \leqslant z \leqslant 0, \ r = \delta/2. \tag{A.14}$$

In the computations given in the main text we have selected $\Delta = 0.25$, $\delta = 0.1$, $\epsilon = 0.05$, $z_1 = 2$, and $z_2 = 0.4$.

The implicit Crank–Nicolson scheme with two iterations used in Section 3.3 for the long time steps may be sketched as follows:

$$\Gamma^* = \Gamma^n - \frac{\Delta t}{2}\left[A(\Psi^n, \Gamma^n) + A(\Psi^n, \Gamma^*) - \frac{1}{Re}(\nabla^2 \Gamma^n + \nabla^2 \Gamma^*)\right],$$

$$\eta^* = \eta^n - \frac{\Delta t}{2}\left[A(\Psi^n, \Gamma^n, \eta^n) + A(\Psi^n, \Gamma^*, \eta^*) - \frac{1}{Re}(\nabla^2 \eta^n + \nabla^2 \eta^*)\right], \quad \nabla^2 \Psi^* = -r\eta^*, \tag{A.15}$$

$$\Gamma^{n+1} = \Gamma^n - \frac{\Delta t}{2}\left[A(\Psi^n, \Gamma^n) + A(\Psi^*, \Gamma^{n+1}) - \frac{1}{Re}(\nabla^2 \Gamma^n + \nabla^2 \Gamma^{n+1})\right],$$

$$\eta^{n+1} = \eta^n - \frac{\Delta t}{2}\left[A(\Psi^*, \Gamma^n, \eta^n) + A(\Psi^n, \Gamma^{n+1}, \eta^{n+1}) - \frac{1}{Re}(\nabla^2 \eta^n + \nabla^2 \eta^{n+1})\right], \quad \nabla^2 \Psi^{n+1} = -r\eta^{n+1}, \tag{A.16}$$

where A represents the convective terms.

References

[1] R. Peyret, T.A. Taylor, Computational Methods for Fluid Flow, Springer, Berlin, 1983.

[2] J. Kim, P. Moin, Application of a fractional-step method to incompressible Navier–Stokes equations, J. Cornput. Phys. 59 (1985) 308–323.

[3] J.M. Lopez, J. Shen, An efficient spectral-projection method for the Navier–Stokes equations in cylindrical geometries, J. Comput. Phys. 139 (1998) 308–326.

[4] C. Xu, R. Pasquetti, On the efficiency of semi-implicit and semi-lagrangian spectral methods for the calculation of incompressible flows, Int. J. Numer. Meth. Fluids 35 (2001) 319–340.

[5] S. Nakamura, Applied Numerical Methods, Prentice-Hall, Englewood Cliffs, NJ, 1991.

[6] G.D. Smith, Numerical Solution of Partial Differential Equations: Finite Difference Methods, Oxford University Press, Oxford, 1985.

[7] C.A.J. Fletcher, Computational Techniques for Fluid Dynamics, Springer, Berlin, 1991.

[8] J.M. Lopez, P.D. Weidman, Stability of stationary endwall boundary layers during spin-down, J. Fluid Mech. 326 (1996) 373–398.