Available online at www.sciencedirect.com
SCIENCE DIRECT®
Journal of Computational Physics 204 (2005) 302–318
JOURNAL OF COMPUTATIONAL PHYSICS
www.elsevier.com/locate/jcp

A Cartesian grid finite-difference method for 2D incompressible viscous flows in irregular geometries

E. Sanmiguel-Rojas [a], J. Ortega-Casanova [b], C. del Pino [b], R. Fernandez-Feria [b,*]

[a] *Universidad Politécnica de Cartagena, E.T.S. Ingenieros Industriales, 30202 Cartagena, Murcia, Spain*
[b] *Universidad de Málaga, E.T.S. Ingenieros Industriales, 29013 Málaga, Spain*

Received 23 April 2004; received in revised form 11 October 2004; accepted 12 October 2004
Available online 11 November 2004

Abstract

A method for generating a non-uniform Cartesian grid for irregular two-dimensional (2D) geometries such that all the boundary points are regular mesh points is given. The resulting non-uniform grid is used to discretize the Navier–Stokes equations for 2D incompressible viscous flows using finite-difference approximations. To that end, finite-difference approximations of the derivatives on a non-uniform mesh are given. We test the method with two different examples: the shallow water flow on a lake with irregular contour and the pressure driven flow through an irregular array of circular cylinders.

1. Introduction

The numerical simulation of flows with irregular geometries is a problem of increasing interest. In particular, much effort have been dedicated in recent years to the use of Cartesian grids which does not conform to the irregular boundaries [1–6]. In relation to the conventional structured-grid approach with curvilinear grids that conforms to the boundaries, this approach has the main advantage of its simplicity, both in the grid generation and in the governing equations. In addition, the transformation of the governing equations to a curvilinear coordinate system that conforms to very complicated boundaries is not an easy task, and usually affects to the stability, convergence, and accuracy of the numerical solver.

In this paper we present a technique for Cartesian grid generation that conforms to irregular two-dimensional (2D) boundaries. It has the advantage of working with a Cartesian mesh in which all the

* Corresponding author.
E-mail address: ramon.fernandez@uma.es. (R. Fernandez-Feria).

0021-9991/$ - see front matter © 2004 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2004.10.010

boundaries nodes are regular nodes of the grid, thus avoiding the usual complicated interpolations needed for the Cartesian cells cut by immersed boundaries, or the use of artificial body forces, or other artifices such as a distribution of vorticity sources, to impose the boundary conditions (see, for example [1–6]). The price one has to pay is that the Cartesian grid is non-uniform. For this reason we develop second-order accurate finite-difference approximations of the derivatives for non-equidistant grid points that substitutes the usual finite-difference approximations for uniform grids (some of these expressions are already reported in the literature; see, e.g. [7]). Thus, we are led to discrete equations with the same level of complexity than the Cartesian equations discretized on a uniform grid, but conforming to an irregular geometry. One disadvantage in relation to the interpolation techniques given in some of the references cited above is that the present method is not suited for moving interfaces. However, second-order interpolation techniques [4] are not easily applied to Neumann boundary conditions.

The structure of the paper is the following. Section 2 introduces the grid generation technique on a generic, irregular 2D domain. The expression for the finite-difference approximations of the Cartesian derivatives on non-uniform grids are given in Section 3. Sections 4 validates the method with two examples quite different to each other: the 2D shallow water, wind-driven flow on a lake with irregular contour, and the 2D incompressible, pressure-driven flow around an irregular array of circular cylinders. Some conclusions are drawn in the last section.

2. Cartesian grid generation that conforms to an irregular 2D domain

Consider the 2D irregular domain of Fig. 1. Our objective is to generate a Cartesian grid where all the boundary points are regular mesh points. This means that all the interior points have to be collocated in relation to a set of selected boundary points, and that the resulting Cartesian grid will not be uniform. In order to simplify the storage of the grid points location in matrix form, what we propose here is a ray tracing technique. One starts at a given boundary point (marked with a circle in Fig. 1), and generates a set of boundary points by 'Cartesian reflections' of the ray (squares in Fig. 1). In order to avoid an infinite regress, the process ends when the ray reaches a boundary perpendicular to it (i.e., when it reaches a section of the boundary parallel to one of the Cartesian axis), or when a boundary node is generated very close to a previous one (their separation is less than a given tolerance).

It is important to detect first the *main* boundary points or points of intersection between the several sections of the boundary (circles in Fig. 2). The first rays will start from these main points, dividing the domain, and the boundary, in a number sections. Each of these sections is then divided using a number of points on each boundary section which depends on the desired precision. The resulting mesh (see Fig. 2(b)) concentrates the nodes with the desired precision at the different sections of the boundary. Programing this technique is relatively easy and the storing in matrix form of the resulting grid points locations is also straightforward.

The technique is a little more complicated in domains with concave boundary sections like that depicted in Fig. 3. We have traced three rays starting from the three circled points. These rays generate a series of mesh points, both on the boundary and inside the domain, before stoping at a boundary parallel to the x-axis. The last three nodes on that boundary have no corresponding ones on the lower boundary. Thus, in order to facilitate the storage of the nodes in matrix form, i.e., in order to have an structured grid, it is convenient to continue these rays till the lowest boundary (dashed lines in Fig. 3(a)). Though we store all the nodes, we can disregard the three triangled nodes in Fig. 3(b), being the effective node to the right of node j that labelled with $j + 4$. Once the grid is generated, one may create an indirect access with a new index, say jj, such that $jj + 1$ corresponds to $j + 4$. Thus, the final non-structured grid is stored within a structured grid form of $n \times m$ nodes (Fig. 3(c)). The indirect access through the indexes (ii, jj) tell us whether a node of the structured grid is an actual node of the computational non-structured grid.

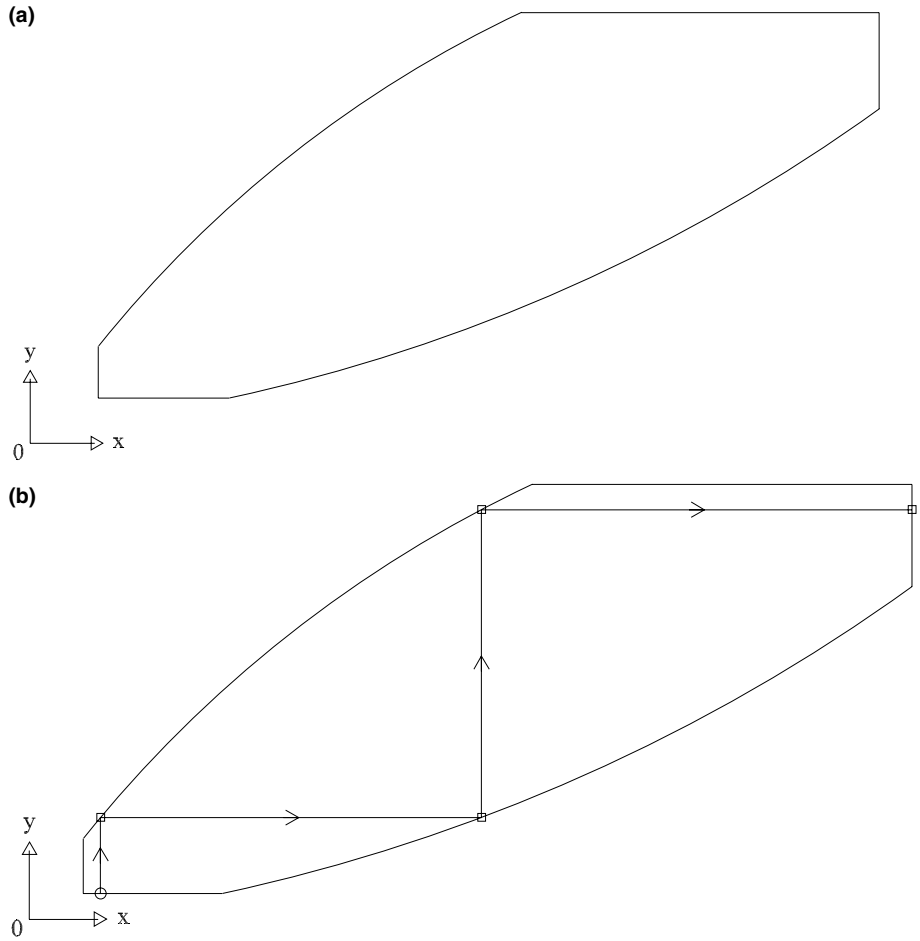


Fig. 1. Generic 2D domain (a) and illustration of grid generation by ray tracing (b).

In very irregular domains it may happen that this ray tracing method may generate very small cells near some boundaries, or even inside the domain. To avoid this we use a lower limit for the cell size, in such a way that nodes that generate cells with size less than this limit are discarded, in a similar way to what is done in concave boundaries, as described above.

3. Finite-difference approximation on non-uniform meshes

In order to discretize the flow equations in the non-uniform grid developed in the above section, one has to use finite-difference approximations on a non-uniform mesh. In this section we develop these finite-difference expressions for all the spatial derivatives appearing in the Navier–Stokes equations. Some of them have been previously reported by Turkel [7]. In particular, Turkel provides the first derivative, and the centered form of the second derivative with first-order truncation error. All the expressions we give below (some of them are given in Appendix A) are second-order accurate, and we include forward and backwards expressions for the second-order derivatives, which are needed at the boundaries. It has been shown [8,9] that second-order accuracy can be obtained (on non-uniform grids) even though local truncation

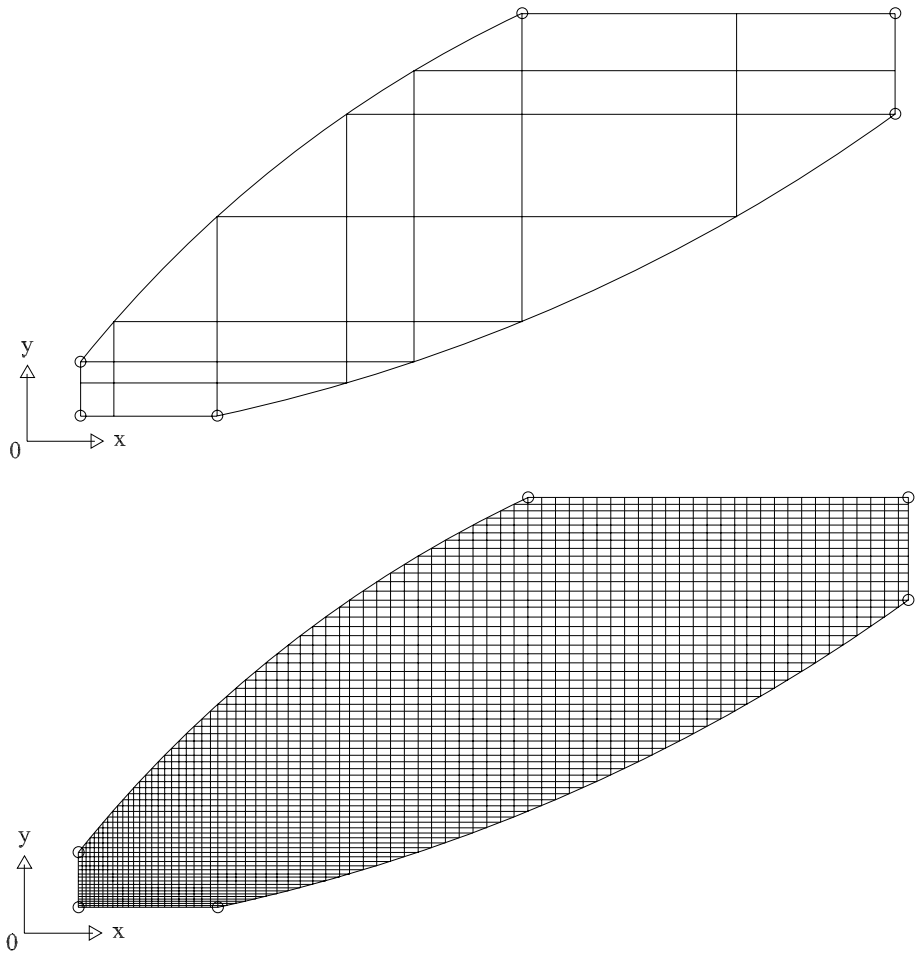


Fig. 2. Main points (circles), and resulting non-uniform grid (b).

errors are of lower order. However, this is valid only for non-uniform grids in which the variation of the mesh size is very small and for linear equations. Since we want to apply the finite-differences method to arbitrary non-uniform meshes, and to the non-linear Navier–Stokes equations, we need second-order truncation errors to reach second-order accuracy. This is important in problems with a long time evolution, such as the examples given below, where second-order accuracy is needed at both inner and boundary nodes.

Consider a 1D non-uniform grid with $nx + 1$ discrete points ($0 \leqslant i \leqslant nx$) located arbitrarily on the unit length (Fig. 4). If the value of a generic function $f(x)$ and its derivatives are known at the point ith, $x_i = i\Delta x$, $\Delta x = 1/nx$, the values of f at the points $i \pm 1$ and $i \pm 2$ can be approximated using Taylor expansions. Indicating with a subscript the grid point, and with primes the derivatives with respect to x, the unknown values $f_{i\pm1}$ and $f_{i\pm2}$ can be written as

$$f_{i+1} = f_i + h_{i+1}f_i' + \frac{h_{i+1}^2}{2}f_i'' + \frac{h_{i+1}^3}{6}f_i''' + \mathrm{O}(h_{i+1}^4), \tag{1}$$

$$f_{i-1} = f_i + h_{i-1}f_i' + \frac{h_{i-1}^2}{2}f_i'' + \frac{h_{i-1}^3}{6}f_i''' + \mathrm{O}(h_{i-1}^4), \tag{2}$$

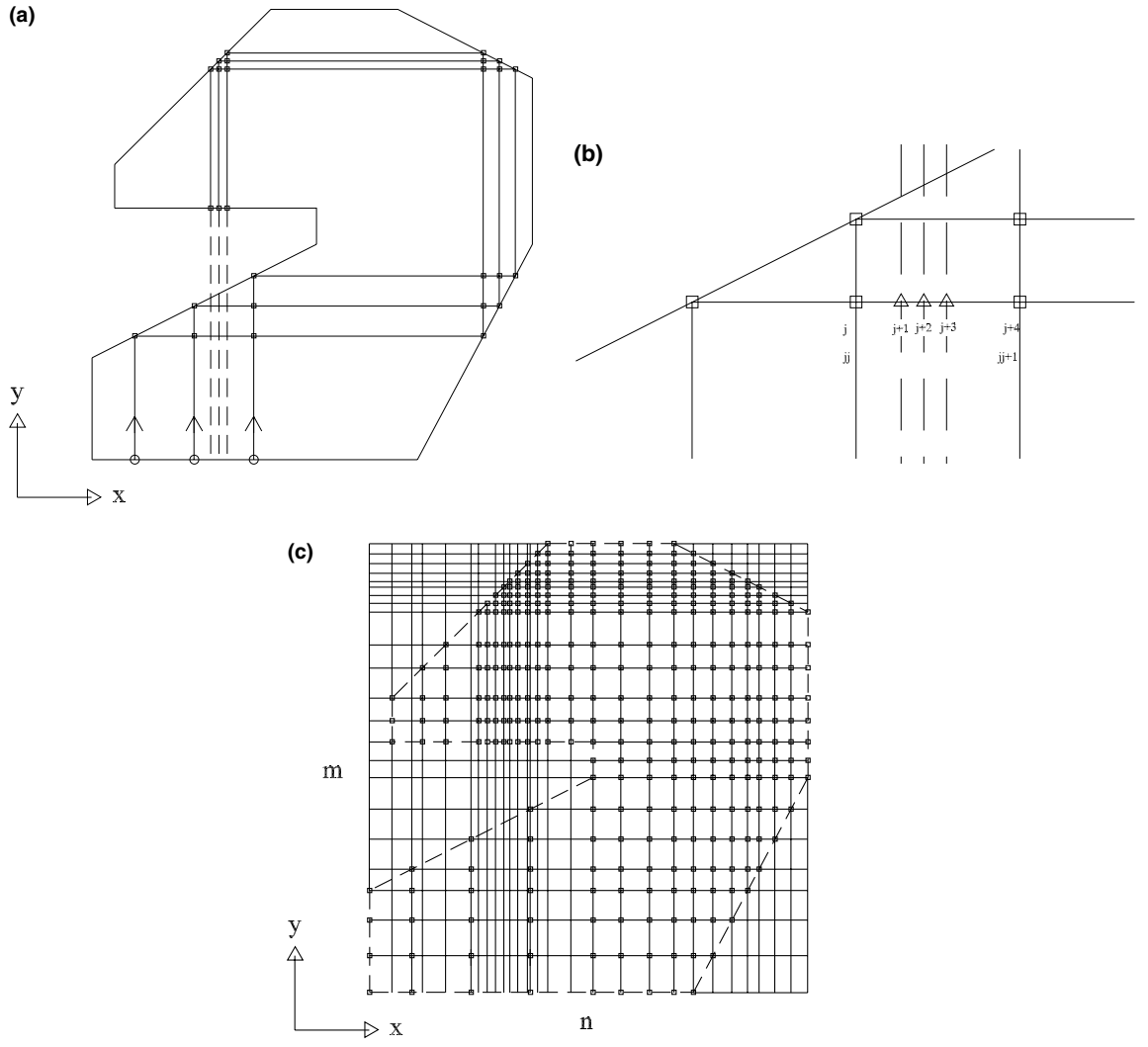


Fig. 3. (a) and (b) Fictitious grid points in a domain with concave boundary sections. (c) Final non-structured grid (squares).

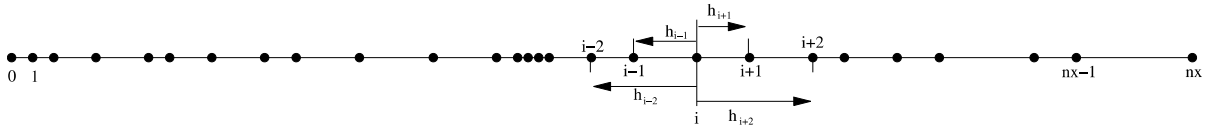


Fig. 4. Nonuniform grid with $nx + 1$ grid points distributed arbitrarily.

$$f_{i+2} = f_i + h_{i+2} f_i' + \frac{h_{i+2}^2}{2} f_i'' + \frac{h_{i+2}^3}{6} f_i''' + \mathrm{O}(h_{i+2}^4), \tag{3}$$

$$f_{i-2} = f_i + h_{i-2} f_i' + \frac{h_{i-2}^2}{2} f_i'' + \frac{h_{i-2}^3}{6} f_i''' + \mathrm{O}(h_{i-2}^4), \tag{4}$$

where the last terms in these expressions indicate the order of the truncation error of the approximation, and

$$\begin{aligned} h_{i+1} &= x_{i+1} - x_i, \\ h_{i-1} &= x_{i-1} - x_i, \\ h_{i+2} &= x_{i+2} - x_i, \\ h_{i-2} &= x_{i-2} - x_i. \end{aligned} \tag{5}$$

These expansions can be used to approximate the nth derivative of f at the point i up to any order of the truncation error, provided that they are conveniently combined. With second-order accuracy, the finite-difference approximation in the centered form for the first and second derivatives of f at the point i are (for higher derivatives, or higher order for the desired truncation error, more points than those considered in (1)–(4) are needed):

$$f_i' = m_i f_{i-1} + s_i f_i + n_i f_{i+1} + E_i, \begin{cases} m_i = \frac{-h_{i+1}}{-h_{i+1}h_{i-1}+h_{i-1}^2}, \\ n_i = \frac{-h_{i-1}}{-h_{i+1}h_{i-1}+h_{i+1}^2}, \\ s_i = -(m_i + n_i), \\ E_i = \frac{1}{6} h_{i+1} h_{i-1} f_i''', \end{cases} \tag{6}$$

$$f_i'' = m_i f_{i-1} + s_i f_i + n_i f_{i+1} + p_i f_{i+2} + E_i, \quad \begin{cases} m_i = \frac{-2(h_{i+1}+h_{i+2})}{(h_{i+1}h_{i+2}-h_{i-1}h_{i+1}+h_{i-1}^2-h_{i-1}h_{i+2})h_{i-1}}, \\ n_i = \frac{-2(h_{i-1}+h_{i+2})}{(-h_{i+1}h_{i-1}+h_{i-1}h_{i+2}+h_{i+1}^2-h_{i+1}h_{i+2})h_{i+1}}, \\ p_i = \frac{2(h_{i+1}+h_{i-1})}{(h_{i+1}h_{i+2}-h_{i-1}h_{i+1}-h_{i+2}^2+h_{i-1}h_{i+2})h_{i+2}}, \\ s_i = -(m_i + n_i + p_i), \\ E_i = \frac{-1}{12}\left(m_i h_{i-1}^4 + n_i h_{i+1}^4 + p_i h_{i+2}^4\right) f_i^{\mathrm{iv}}, \end{cases} \tag{7}$$

$$f_i'' = p_i f_{i-2} + m_i f_{i-1} + s_i f_i + n_i f_{i+1} + E_i, \quad \begin{cases} m_i = \frac{-2(h_{i+1}+h_{i-2})}{(h_{i+1}h_{i-2}-h_{i-1}h_{i+1}+h_{i-1}^2-h_{i-1}h_{i-2})h_{i-1}}, \\ n_i = \frac{-2(h_{i-1}+h_{i-2})}{(-h_{i+1}h_{i-1}+h_{i-1}h_{i-2}+h_{i+1}^2-h_{i+1}h_{i-2})h_{i+1}}, \\ p_i = \frac{2(h_{i+1}+h_{i-1})}{(h_{i+1}h_{i-2}-h_{i-1}h_{i+1}-h_{i-2}^2+h_{i-1}h_{i-2})h_{i-2}}, \\ s_i = -(m_i + n_i + p_i), \\ E_i = \frac{-1}{12}\left(p_i h_{i-2}^4 + m_i h_{i-1}^4 + n_i h_{i+1}^4\right) f_i^{\mathrm{iv}}, \end{cases} \tag{8}$$

where E_i is the truncation error of each approximation. This truncation error is a function of the separation between the *local* points around i. Thus, in a non-uniform grid, expressions (6)–(8) will be more accurate as the grid points becomes closer. Note that two different expressions for the second derivative are given, (7) and (8), neither of them fully centered on the point i. This is because one needs four grid points to have a second-order truncation error in the second derivative, so that two different expressions result depending on whether we use the point $i + 2$, or the point $i - 2$.

If the grid were uniform, $h \equiv h_{i+1}$, $h_{i-1} = -h$, $h_{i+2} = 2h$, $h_{i-2} = -2h$, expressions (6)–(8) are, obviously, the standard centered second-order finite-difference approximation for the first and second derivatives:

$$f_i' = m_i f_{i-1} + s_i f_i + n_i f_{i+1} + E_i, \quad \begin{cases} m_i = \frac{-1}{2h}, \\ n_i = \frac{1}{2h}, \\ s_i = 0, \\ E_i = \frac{-1}{6} h^2 f_i''', \end{cases} \tag{9}$$

$$f_i'' = m_i f_{i-1} + s_i f_i + n_i f_{i+1} + p_i f_{i\pm 2} + E_i, \quad \begin{cases} m_i = \frac{1}{h^2}, \\ n_i = \frac{1}{h^2}, \\ p_i = 0, \\ s_i = \frac{-2}{h^2}. \\ E_i = \frac{-1}{12} h^2 f_i^{\mathrm{iv}}, \end{cases} \tag{10}$$

Now the grid points $i \pm 2$ do not appear in the approximation to the second derivative because $p_i = 0$.

From (1)–(4) one can obtain not only centered approximations, but also forward or backward ones. These expressions, which are needed at the boundaries, are given in Appendix A with second-order accuracy. We also give there finite-difference expressions needed to apply Neumann boundary conditions with second-order accuracy on a non-uniform grid.

4. Results

4.1. Wind-driven flow in a lake

One of the main intended applications of the present technique is the simulations of 2D environmental flows, such as the flows in shallow lakes and reservoirs, which usually have very irregular geometries. For this reason, as a first example of a 2D flow in a complex domain we consider the wind-driven flow in Lake Belau (Northern Germany), for which Podsetchine and Schernewski [10] reported numerical and experimental results which can be used to compare with. The bathymetry of the lake is given in Fig. 5.

Since the lake is not very deep, one may use the vertically integrated equations of continuity and momentum on the horizontal plane (x, y), or shallow water approximation (see, for example, [11]):

$$\frac{\partial \zeta}{\partial t} + \nabla \cdot \mathbf{v} = 0, \tag{11}$$

$$\frac{\partial \mathbf{v}}{\partial t} + \nabla \cdot \left(\frac{\mathbf{v}\mathbf{v}}{H}\right) + gH\nabla\zeta - \nu\left[\nabla^2\mathbf{v} - 2\nabla\zeta \cdot \nabla\left(\frac{\mathbf{v}}{H}\right) - \left(\frac{\mathbf{v}}{H}\right)\nabla^2\zeta\right] - \mathbf{f} \wedge \mathbf{v} - k\mathbf{W} \mid \mathbf{W} \mid + \frac{g\mathbf{v} \mid \mathbf{v} \mid}{C^2H^2} = 0. \tag{12}$$

In these equations, $\mathbf{v} = [u(x, y, t), v(x, y, t)]$ is the depth-averaged horizontal velocity,

$$\mathbf{v} \equiv \int_{-h}^{\zeta} \mathbf{v}_{\mathrm{h}}\, \mathrm{d}z, \tag{13}$$

with $\mathbf{v}_{\mathrm{h}}(x, y, z, t)$ the local horizontal velocity, $H(x, y, t) \equiv h(x, y) + \zeta(x, y, t)$ is the total water depth, with h the depth below the horizontal reference plane ($z = 0$) and ζ the water surface elevation above $z = 0$, $\nabla = \partial/\partial x + \partial/\partial y$, $g \simeq 9.81$ m/s^2 is the acceleration due to gravity, $\mathbf{f} = f\mathbf{e}_z$, with $f \simeq 1.176 \times 10^{-4}$ s^{-1} is the Coriolis parameter, ν is the averaged, horizontal eddy viscosity, and C is the Chezy coefficient.

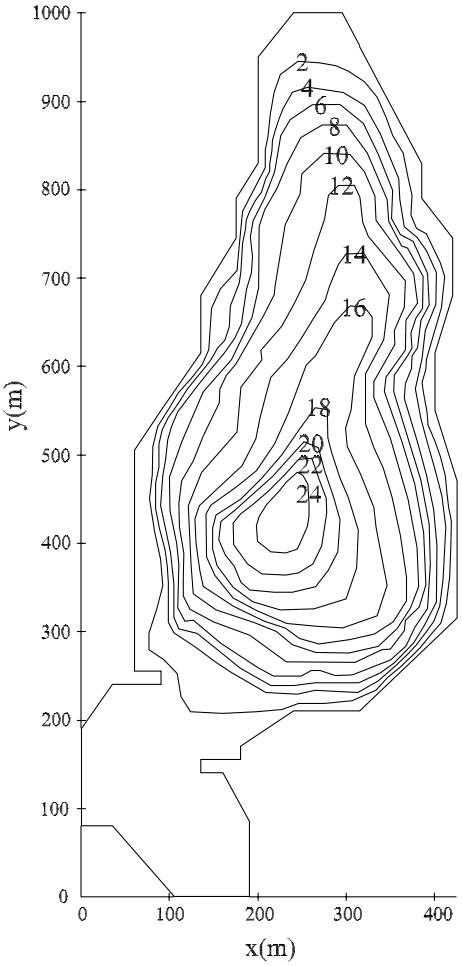


Fig. 5. Digital contour and bathymetry (in meters) of Lake Belau such as they are used in the computations (taken from Fig. 2 in [10]).

The flow will be driven by a wind of velocity $\mathbf{W} = (W_x, W_y)$, through the term $k\mathbf{W}|\mathbf{W}|$, where $k = \rho_a C_W/\rho$, with ρ and ρ_a the densities of water and air, respectively, and C_W the wind drag coefficient. The numerical values of the parameters that will be used to solve these equations are the following [10]: $\nu = 0.01$ m^2/s, $C = 40$ m$^{1/2}$/s, $C_W = 0.002$, $\rho = 10^3$ kg/m^3, $\rho_a = 1.275$ kg/m^3, and a spatially uniform south-westerly wind (a heading of 220°) with a speed $|\mathbf{W}|$ of 6 m/s. The boundary conditions at the contour of the lake are $u = v = 0$.

A mesh of 7517 grid points has been generated using the technique of Section 2 (see Fig. 6). The equations have been discretized in this non-uniform grid using the finite-difference approximations given in Section 3. In particular, we have used Arakawa's grid of the type C [12], where the water elevation ζ is evaluated at the grid points, while the averaged velocity components are evaluated at the mid points of their respective cell sides (see Fig. 7).

An explicit, two-step, second-order accurate, predictor–corrector scheme has been used to advance in time. If one writes Eqs. (11) and (12) schematically as $\partial\zeta/\partial t = A(\mathbf{v})$, $\partial\mathbf{v}/\partial t = \mathbf{B}(\zeta, \mathbf{v})$, these two steps are given by

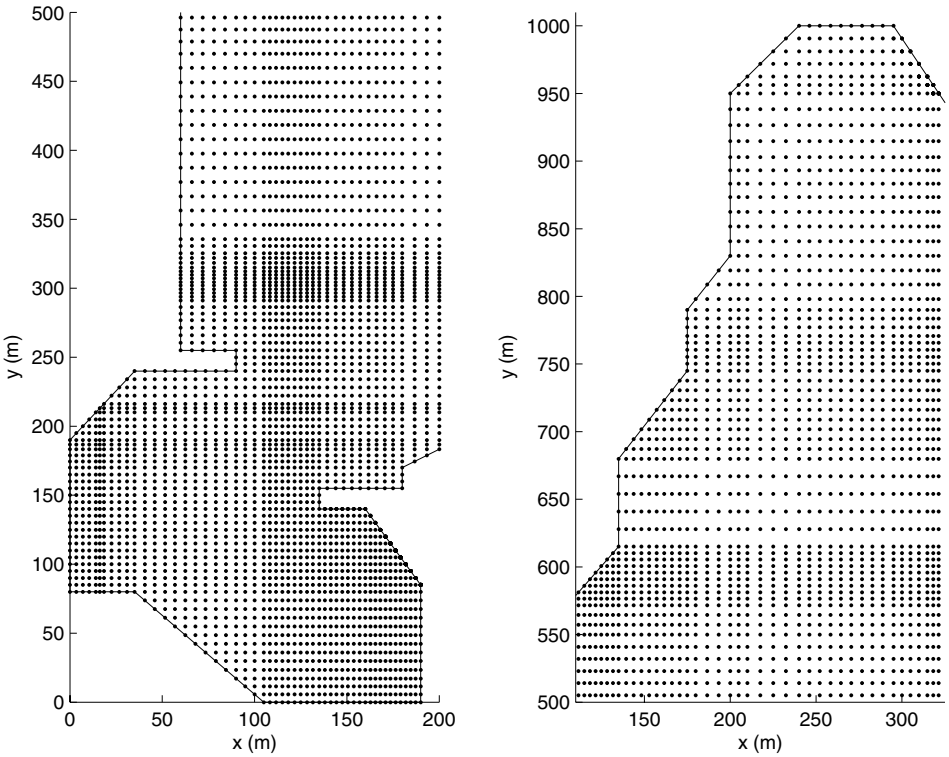


Fig. 6. Two details of the discretized geometry of the lake showing some of the grid points.

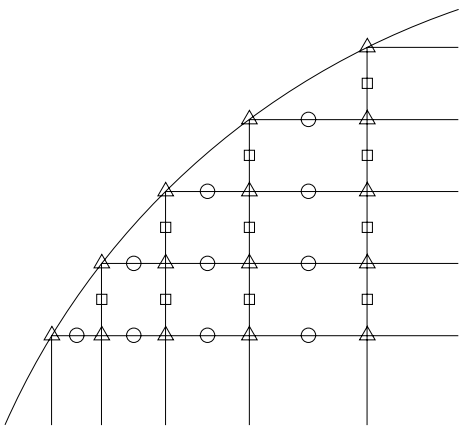


Fig. 7. Arakawa's scheme used in the computations, where ζ is evaluated at the grid points (triangles), u is evaluated at the squares, and v at the circles.

predictor step:

$$\begin{aligned}\mathbf{v}^* &= \mathbf{v}^n + \frac{\Delta t}{2}\mathbf{B}(\zeta^n, \mathbf{v}^n),\\ \zeta^* &= \zeta^n + \frac{\Delta t}{2}A(\mathbf{v}^*),\end{aligned} \tag{14}$$

corrector step:

$$\begin{aligned}\mathbf{v}^{n+1} &= \mathbf{v}^n + \Delta t\mathbf{B}(\zeta^*, \mathbf{v}^*),\\ \zeta^{n+1} &= \zeta^n + \Delta tA(\mathbf{v}^{n+1}),\end{aligned} \tag{15}$$

where the superscripts denote the instant of time, and Δt is the time step. The numerical computations are started at $t = 0$ with the fluid at rest and $\zeta = 0$. We use $\Delta t = 1$ s. The results for the velocity field at $t = 3$ h are plotted in Fig. 8. These results compares very well with those given in Fig. 4(a) of Podsetchine and Schernewski [10], who used a finite-element method on a triangular mesh created with a commercial

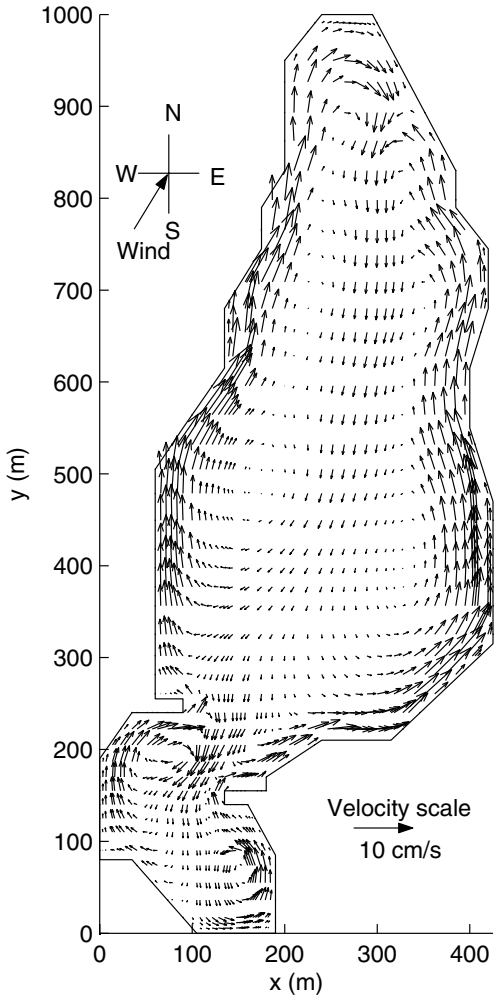


Fig. 8. Averaged velocity field at $t = 3$ h.

software package to solve the shallow water equations. These authors also checked their numerical results with experimental measurements.

4.2. Pressure-driven flow through an array of circular cylinders

As a second example we have selected the 2D incompressible flow around an irregular array of three circular cylinders inside a channel (see Fig. 9). In particular we have considered the pressure driven flow [13] originated by a given pressure difference set between the inlet and the outlet.

The dimensionless equations are

$$\nabla \cdot \mathbf{v} = 0, \tag{16}$$

$$\frac{\partial \mathbf{v}}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} = -\nabla p + \frac{1}{Re} \nabla^2 \mathbf{v}, \tag{17}$$

where $\mathbf{v} = (u, v)$ and p are the dimensionless velocity and pressure, respectively. To non-dimensionalize these equations we have used the diameter D of the cylinders as the length scale, and a characteristic velocity based on the the pressure difference Δp_c between the inlet and the outlet, $V_c = \sqrt{\Delta p_c / \rho}$, where ρ is the fluid density; the Reynolds number is based on this velocity [13]

$$Re = \frac{V_c D}{v} = \sqrt{\frac{\Delta p_c}{\rho}} \frac{D}{v}, \tag{18}$$

where v is the kinematic viscosity of the fluid.

The motion of the fluid is set by the boundary conditions

$$p(x = 0,\ y = 10,\ t) = 1, \quad p(x = 20,\ y = 10,\ t) = 0. \tag{19}$$

The remaining boundary conditions are $\mathbf{v} = 0$ on the cylinders, and at the channel walls, $y = 0$ and $y = 10$. The equations are solved numerically with the second-order (both in space and time) projection method described in [13], using a finite-difference scheme on a non-uniform grid generated with the method

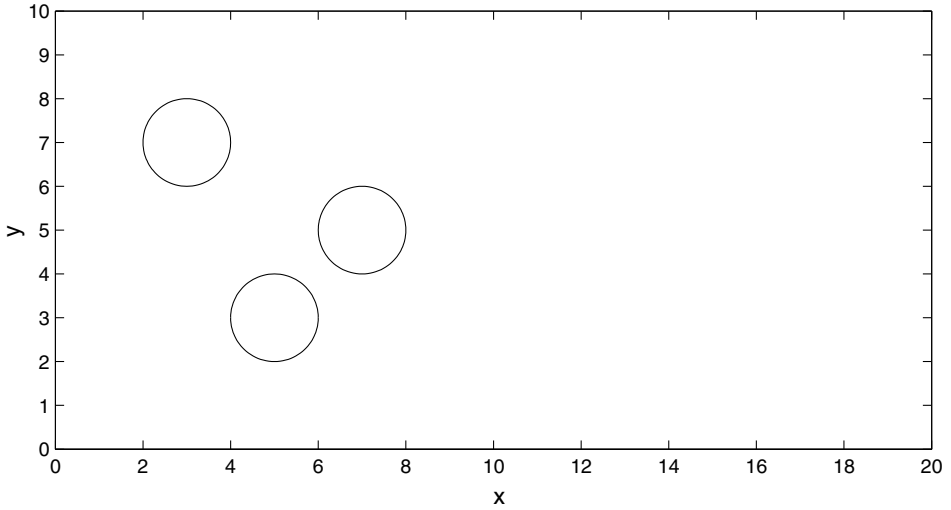
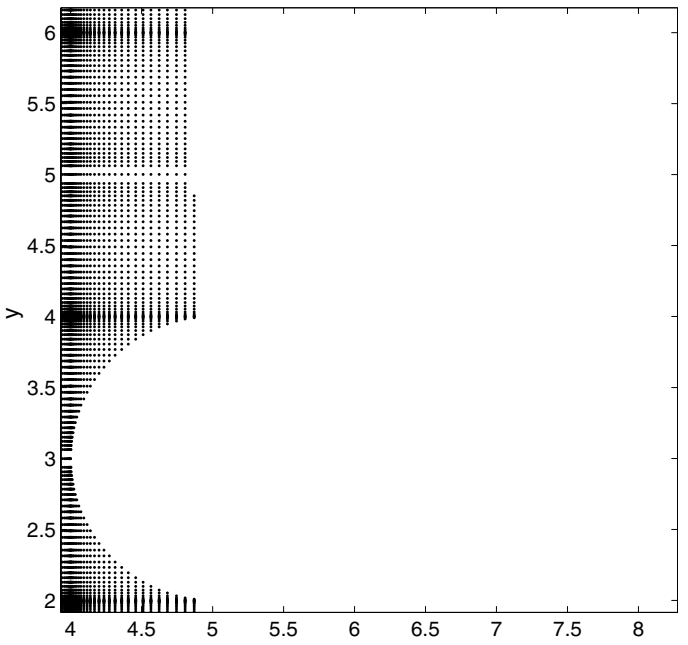


Fig. 9. Geometry of the channel flow through three circular cylinders.


y
6
5.5
5
4.5
4
3.5
3
2.5
2
4
4.5
5
5.5
6
6.5
7
7.5
8

described above. In particular we have used a grid of 438×298 mesh points, and $\Delta t = 2.5 \times 10^{-5}$. A detail of the grid near two of the cylinders is depicted in Fig. 10. To solve numerically the Poisson equation for the pressure we use an ADI based technique, and standard solvers for band matrices with LU factorization from Blas and Lapack packages. The fact that we have now, in general, a non-structured grid does not affect to the efficiency of these Poisson solvers because what is supplied to them are the actual computational nodes and their corresponding discretized equations through the indirect access mentioned in Section 2.

Results for $Re = 50$ are plotted in Figs. 11 and 12. The flow evolves in time from rest until it reaches a quasi-periodic state. This is shown in Fig. 11, where we plot as a function of time the Reynolds number based on the flow rate at the exit of the channel ($x = L = 20$),

$$Re_q \equiv \frac{q}{H} Re, \quad q(t) = \int_0^H u(x = L, y, t) \mathrm{d}y, \tag{20}$$

where $H = 10$ is the width of the channel. Finally, streamlines and isobars at $t = 106$ are plotted in Fig. 12.

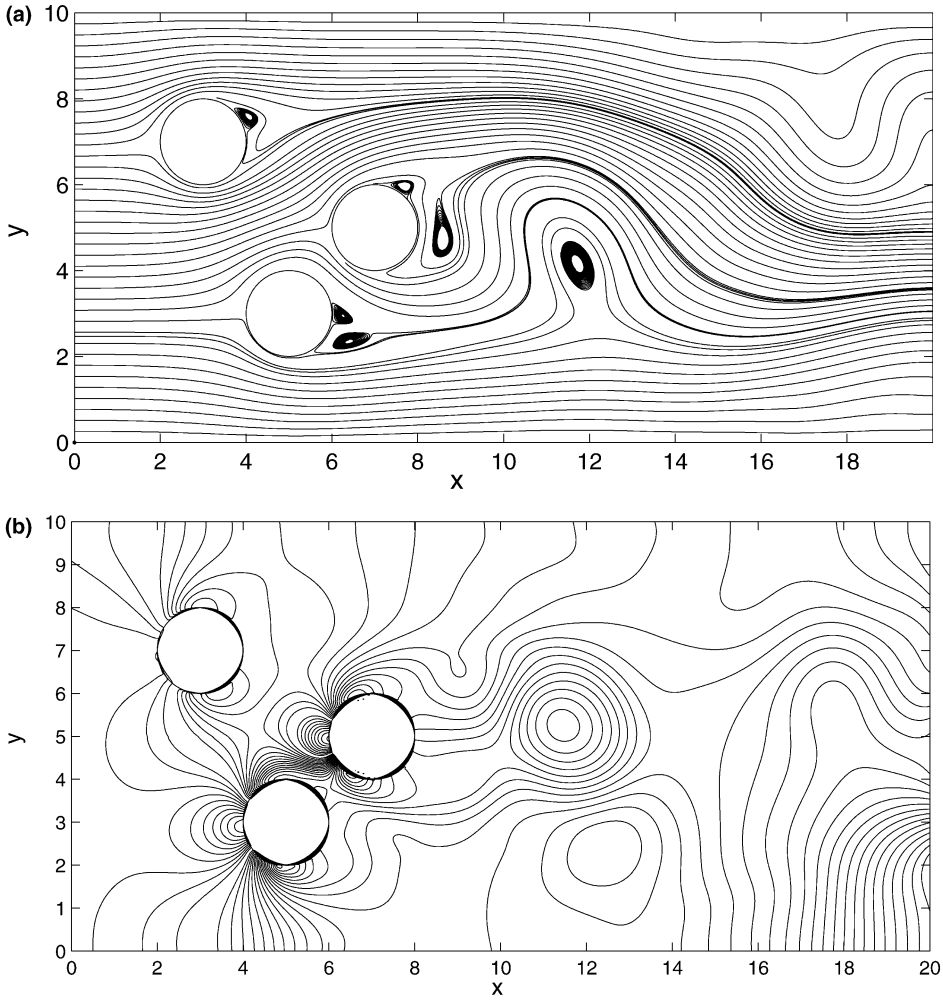


Fig. 12. Streamlines (a), and isobars (b), at $t = 106$.

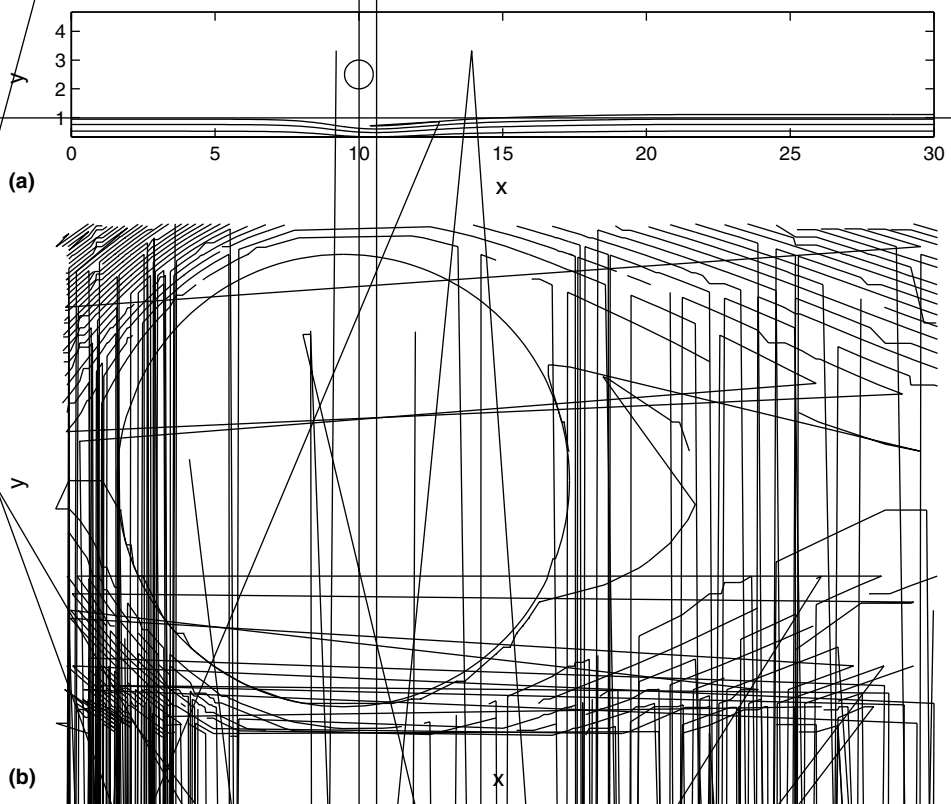


Fig. 13. Flow inside a channel with a single circular cylinder for $Re_q = 10$. (a) Streamlines (all the lengths are scaled with the diameter of the cylinder). (b) Detail of the wake behind the cylinder.

The above results show that the current method has no difficulty in resolving the complex flow pattern and recirculation regions behind the cylinders. However, since no previous results are reported for this particular flow, we consider next the case of a single cylinder inside a channel (see Fig. 13). This flow, with the dimensions given in Fig. 13(a), has been studied experimentally and numerically [14]. It has been reported that the length l of the axisymmetric wake behind the cylinder for moderately small Reynolds number varies linearly with Re_q. Thus, for instance, for $Re_q = 10$ (which in our simulation corresponds to $Re \simeq 17$), the length of the wake behind the cylinder reported is $l = 0.285$ (times de diameter of the cylinder) [14]. Fig. 13(b) shows a detail of the streamlines behind the cylinder obtained numerically by our numerical code, showing that $l \simeq 0.28$, in close agreement with the previous result.

5. Conclusions

We have presented here a finite-difference method in a non-uniform Cartesian grid which allows us to solve 2D unsteady viscous flows in irregular geometries. It has all the advantages of solving numerically the flow equations in a Cartesian mesh (simplicity, accuracy, stability), and has the important peculiarity that all the boundaries are fitted to the regular points of the mesh, so that complicated interpolations at the boundaries are avoided. We have developed a simple method for generating such a non-uniform

Cartesian mesh in complex geometries, and provided second-order finite-difference approximations in non-uniform meshes, which allow us to solve the flow equations with second-order accuracy in all the flow domain, including the boundaries.

To check the validity of the method in flows with both irregular external boundaries and complex immersed boundaries, we have selected two typical examples where we think this method will be of most interest: the 2D (shallow water) flow in a lake with a very complicated external boundary, and a viscous flow with a complex internal boundary such as the flow through an irregular array of cylinders between parallel plates. These flows are solved with second-order time accuracy using two different numerical schemes, and the results show that the non-uniform Cartesian mesh is able to accurately simulate these flows with easy.

Acknowledgement

This research has been supported by the COPT of the Junta de Andalucía (Contract No. 807/31.2116). The numerical simulations have been made in the computer facilities at the SAIT (U.P. Cartagena) and at the SCAI (U. Málaga).

Appendix A

In this appendix we give some additional expressions for finite-difference approximations on a non-uniform grid that we have used in our computations. For example, to apply Dirichlet boundary conditions we need forward or backward expressions for the first and second derivatives. With second-order accuracy, these one sided approximations are:

$$f_i' = s_i f_i + m_i f_{i-1} + p_i f_{i-2} + E_i, \quad \begin{cases} m_i = \frac{-h_{i-2}}{h_{i-1}(h_{i-1}-h_{i-2})}, \\ p_i = \frac{h_{i-1}}{h_{i-2}(h_{i-1}-h_{i-2})}, \\ s_i = -(m_i + p_i), \\ E_i = \frac{1}{6} h_{i-1} h_{i-2} f_i''', \end{cases} \tag{A.1}$$

$$f_i' = s_i f_i + n_i f_{i+1} + p_i f_{i+2} + E_i, \quad \begin{cases} n_i = \frac{-h_{i+2}}{h_{i+1}(h_{i+1}-h_{i+2})}, \\ p_i = \frac{h_{i+1}}{h_{i+2}(h_{i+1}-h_{i+2})}, \\ s_i = -(n_i + p_i), \\ E_i = \frac{1}{6} h_{i+1} h_{i+2} f_i''', \end{cases} \tag{A.2}$$

for the first derivative, and

$$f_i'' = s_i f_i + m_i f_{i-1} + p_i f_{i-2} + q_i f_{i-3} + E_i, \quad \begin{cases} m_i = \frac{-2(h_{i-2}+h_{i-3})}{(-h_{i-2}h_{i-1}+h_{i-2}h_{i-3}+h_{i-1}^2-h_{i-3}h_{i-1})h_{i-1}}, \\ p_i = \frac{2(h_{i-1}+h_{i-3})}{(-h_{i-3}h_{i-1}+h_{i-2}h_{i-1}-h_{i-2}^2+h_{i-2}h_{i-3})h_{i-2}}, \\ q_i = \frac{-2(h_{i-1}+h_{i-2})}{(-h_{i-3}h_{i-1}+h_{i-2}h_{i-1}+h_{i-3}^2-h_{i-2}h_{i-3})h_{i-3}}, \\ s_i = -(m_i + p_i + q_i), \\ E_i = \frac{-1}{12}\left(m_i h_{i-1}^4 + p_i h_{i-2}^4 + q_i h_{i-3}^4\right) f_i^{\mathrm{iv}} \end{cases} \tag{A.3}$$

$$f_i'' = s_i f_i + n_i f_{i+1} + p_i f_{i+2} + q_i f_{i+3} + E_i, \quad \begin{cases} n_i = \frac{-2(h_{i+2}+h_{i+3})}{(-h_{i+2}h_{i+1}+h_{i+2}h_{i+3}+h_{i+1}^2-h_{i+3}h_{i+1})h_{i+1}}, \\ p_i = \frac{2(h_{i+1}+h_{i+3})}{(-h_{i+3}h_{i+1}+h_{i+2}h_{i+1}-h_{i+2}^2+h_{i+2}h_{i+3})h_{i+2}}, \\ q_i = \frac{-2(h_{i+1}+h_{i+2})}{(-h_{i+3}h_{i+1}+h_{i+2}h_{i+1}+h_{i+3}^2-h_{i+2}h_{i+3})h_{i+3}}, \\ s_i = -(n_i + p_i + q_i), \\ E_i = \frac{-1}{12}\left(n_i h_{i+1}^4 + p_i h_{i+2}^4 + q_i h_{i+3}^4\right) f_i^{\mathrm{iv}}, \end{cases} \tag{A.4}$$

for the second derivative ($h_{i\pm3} = x_{i\pm3} - x_i$).

To apply Neumann boundary conditions we need the second-order derivative in terms of the the first-order one. With second-order accuracy, these expressions (forward and backward) are:

$$f_i'' = r_i f_i' + s_i f_i + n_i f_{i+1} + p_i f_{i+2} + E_i, \quad \begin{cases} r_i = \frac{-2(h_{i+1}+h_{i+2})}{h_{i+1}h_{i+2}}, \\ s_i = \frac{-2(h_{i+1}^2+h_{i+1}h_{i+2}+h_{i+2}^2)}{h_{i+1}^2+h_{i+2}^2}, \\ n_i = \frac{-2h_{i+2}}{(h_{i+1}-h_{i+2})h_{i+1}^2}, \\ p_i = \frac{2h_{i+1}}{(h_{i+1}-h_{i+2})h_{i+2}^2}, \\ E_i = \frac{-1}{12} h_{i+1}h_{i+2} f_i^{\mathrm{iv}} \end{cases} \tag{A.5}$$

$$f_i'' = r_i f_i' + s_i f_i + n_i f_{i-1} + p_i f_{i-2} + E_i, \quad \begin{cases} r_i = \frac{-2(h_{i-1}+h_{i-2})}{h_{i-1}h_{i-2}}, \\ s_i = \frac{-2(h_{i-1}^2+h_{i-1}h_{i-2}+h_{i-2}^2)}{h_{i-1}^2+h_{i-2}^2}, \\ n_i = \frac{-2h_{i-2}}{(h_{i-1}-h_{i-2})h_{i-1}^2}, \\ p_i = \frac{2h_{i-1}}{(h_{i-1}-h_{i-2})h_{i-2}^2}, \\ E_i = \frac{-1}{12} h_{i-1}h_{i-2} f_i^{\mathrm{iv}}. \end{cases} \tag{A.6}$$

References

[1] T. Ye, R. Mittal, H.S. Udaykumar, W. Shyy, An accurate cartesian grid method for viscous incompressible flows with complex immersed boundaries, J. Comput. Phys. 156 (1999) 209–240.

[2] D. Calhoun, R.J. LeVeque, An cartesian grid finite-volume method for the advection–diffusion equation in irregular geometries, J. Comput. Phys. 157 (2000) 143–180.

[3] E.A. Fadlun, R. Verzicco, P. Orlandi, J. Mohd-Yusof, Combined immersed-boundary finite-difference methods for three-dimensional complex flow simulations, J. Comput. Phys. 161 (2000) 33–60.

[4] F. Gibou, R. Fedkiw, L.T. Cheng, M. Kang, A second-order-accurate symmetric discretization of the Poisson equation on irregular domains, J. Comput. Phys. 176 (2002) 205–227.

[5] D. Calhoun, A cartesian grid method for solving the two-dimensional streamfunction–vorticity equations in irregular regions, J. Comput. Phys. 176 (2002) 231–275.

[6] A. Gilmanov, F. Sotiropoulos, E. Balaras, A general reconstruction algorithm for simulating flows with complex 3D immersed boundaries on cartesian grids, J. Comput. Phys. 191 (2003) 660–669.

[7] E. Turkel, Accuracy of schemes with nonuniform meshes for compressible fluid-flows, J. Appl. Numer. Math. 2 (1986) 529–550.

[8] T.A. Manteuffel, A.B. White, The numerical solution of second-order boundary value problems on nonuniform meshes, Math. Comput. 47 (1986) 511–535.

[9] H.O. Kreiss, T.A. Manteuffel, B. Swartz, B. Wendroff, A.B. White, Supra-convergent schemes on irregular grids, Math. Comput. 47 (1986) 537–554.
[10] V. Podsetchine, G. Schernewski, The influence of spatial wind inhomogeneity on flow patterns in a small lake, Wat. Res. 33 (1999) 3348–3356.
[11] T. Weijan, Shallow Water Hydrodynamics, Elsevier, Amsterdam, 1992.
[12] A. Arakawa, V. Lamb, Computational design of the basic dynamical processes of the UCLA general circulation model, Methods Comput. Phys. 17 (1977) 173–265.
[13] R. Fernandez-Feria, E. Sanmiguel-Rojas, An explicit projection method for solving incompressible flows driven by a pressure difference, Comput. Fluids 33 (2004) 463–483.
[14] J.H. Chen, W.G. Pritchard, S.J. Tavener, Bifurcation for flow past a cylinder between parallel planes, J. Fluid Mech. 284 (1995) 23–54.