Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational and Applied Mathematics 215 (2008) 618–625
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

A surface capturing method for the efficient computation of steady water waves☆

Jeroen Wackers[a,*], Barry Koren[a,b]

[a] *CWI, P.O. Box 94079, 1090 GB Amsterdam, The Netherlands*
[b] *Delft University of Technology, Faculty of Aerospace Engineering, P.O. Box 5058, 2600 GB Delft, The Netherlands*

Received 14 September 2005

Abstract

A surface capturing method is developed for the computation of steady water–air flow with gravity. Fluxes are based on artificial compressibility and the method is solved with a multigrid technique and line Gauss–Seidel smoother. A test on a channel flow with a bottom bump shows the accuracy of the method and the efficiency of the multigrid solver.

MSC: 65N55; 76D05; 76D33

Keywords: Steady water waves; Surface capturing; Multigrid; Artificial compressibility; Osher flux function

1. Introduction

In the design of ships and offshore structures, computations of steady water flow play an important role, e.g., computations of the wave pattern and friction drag of ships can help in optimising ship designs for low drag. To enable efficient design, these computations must be fast and accurate. A well-known technique for computing steady flows is to time-march the unsteady flow equations to steady state. But water flows with free surfaces and gravity effects need a long time to reach a steady state, as they show traveling waves that damp out very slowly. So when large-scale 3D water flows are computed, the size and the complexity of the flows that can be computed is reduced significantly. Therefore, more efficient solution techniques are highly desirable and often used [5,6].

Most existing surface-fitting methods, i.e., methods that model the free surface by deforming the mesh to fit the water surface, are equipped with efficient steady solvers. But if the free surface is modelled with a surface capturing method, like the volume-of-fluid or level set technique, time marching is still the most widely used solution method. A great advantage of surface capturing techniques is that they can handle nonlinear steep waves and near-breaking waves, as well as complex geometries near the water surface. But to fully use this advantage in the computation of steady flow, an efficient solution method is needed.

☆ This research was supported by the Dutch government through the national program BSIK: knowledge and research capacity, in the project BRICKS.
* Corresponding author. Tel.: +31 20 5924176; fax: +31 20 5924199.
E-mail addresses: jeroen.wackers@cwi.nl (J. Wackers), barry.koren@cwi.nl (B. Koren).

0377-0427/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.cam.2006.03.056

We are developing a method for 2D water flow based on steady flow equations. Both the water flow and the air flow above it are modelled. A modified volume-of-fluid capturing technique is used to find the water–air interface and the system is solved with a multigrid method. The flux function is based on artificial compressibility.

2. Flow equations

In order to get a flow discretisation that can be solved easily with multigrid, we base our flow equations on conservation laws only. The water–air interface appears as a mixture zone, with a smooth transition from water to air. The flow of this mixture satisfies the Navier–Stokes equations, just like the pure fluids; therefore, the same equations are valid everywhere in the domain, as long as the bulk density is properly defined. To distinguish between the fluids, we add a mass conservation equation for one of the fluids. Water and air are both considered incompressible: they have constant densities. Defining α as the volume fraction of water, the mixture density is $\rho = \rho_w\alpha + \rho_a(1-\alpha)$. Substituting this relation in the steady compressible (variable-density) 2D Navier–Stokes equations with gravity yields the following, incompressible flow equations:

$$\begin{aligned}
&\frac{\partial}{\partial x}(u) + \frac{\partial}{\partial y}(v) = 0 \quad \text{(tot. mass)},\\
&\frac{\partial}{\partial x}(p+\rho u^2) + \frac{\partial}{\partial y}(\rho uv) = \frac{\partial}{\partial x}(\mu u_x) + \frac{\partial}{\partial y}(\mu u_y) \quad (x\text{-mom.}),\\
&\frac{\partial}{\partial x}(\rho uv) + \frac{\partial}{\partial y}(p+\rho v^2) = \frac{\partial}{\partial x}(\mu v_x) + \frac{\partial}{\partial y}(\mu v_y) - \rho g \quad (y\text{-mom.}),\\
&\frac{\partial}{\partial x}(u\alpha) + \frac{\partial}{\partial y}(v\alpha) = 0 \quad \text{(water mass)}.
\end{aligned} \tag{1}$$

As opposed to single-fluid incompressible flow, the density is not constant, so it cannot be divided out in the momentum equations. However, it still disappears from both continuity equations: total mass conservation has its standard form $u_x + v_y = 0$ and mass conservation for the water reduces to a volume-of-fluid equation. So we have a system of equations that is equivalent to volume-of-fluid, but that is completely in conservation form.

3. Flux function

System (1) is discretised with a cell-centred finite-volume technique. The states left and right of the cell faces are taken equal to the state in the cell centre: a first-order reconstruction. Then a flux function is used to compute the flux across the cell face from these two states. The flux function is split in a convective and a diffusive part, to independently control the stability of these parts. And because of the splitting, different boundary conditions can be assigned for convection and diffusion, consistent with the fluxes.

3.1. Linearised Osher convective flux

We discretise the convective part of the flux with the artificial compressibility technique [2,7], that guarantees stability: in the time-dependent flow equations, artificial time derivatives are added to the continuity equations. The resulting hyperbolic system is used to define a Riemann flux function, which is substituted back into the steady flow equations. These are then solved directly with the multigrid technique. Such a technique is also used in [3].

The two-fluid artificial compressibility equations are found by assuming that the densities of the pure fluids in the continuity equations have time derivatives $(\rho_w)_t = p_t/c_w^2$, $(\rho_a)_t = p_t/c_a^2$ (but zero gradients!). The parameters c_w and c_a, with the dimension of velocity, can be chosen freely. We choose $\rho_w c_w^2 = \rho_a c_a^2 = c^2$, with a constant c, to simplify the resulting equations. Substituting this in the time-dependent version of (1) and taking only the

convective part, we get

$$\begin{aligned}
&\frac{1}{c^2}p_t + \frac{\partial}{\partial x}(u) + \frac{\partial}{\partial y}(v) = 0,\\
&\frac{\partial}{\partial t}(\rho u) + \frac{\partial}{\partial x}(p + \rho u^2) + \frac{\partial}{\partial y}(\rho u v) = 0\\
&\frac{\partial}{\partial t}(\rho v) + \frac{\partial}{\partial x}(\rho u v) + \frac{\partial}{\partial y}(p + \rho v^2) = 0,\\
&\alpha_t + \frac{\alpha}{c^2}p_t + \frac{\partial}{\partial x}(u\alpha) + \frac{\partial}{\partial y}(v\alpha) = 0.
\end{aligned} \tag{2}$$

The two p_t's are the artificial time derivatives. Note that the time derivatives in this system cannot be written in conservation form.

System (2) is hyperbolic, so it can be used to construct an Osher type flux function. Writing the system in quasilinear form and regarding only the x-derivatives, we get a system $\mathbf{q}_t + J\mathbf{q}_x = 0$. The eigenvalues of the Jacobian matrix J are

$$\lambda^- = \tfrac{1}{2}u - \sqrt{c^2/\rho + (\tfrac{1}{2}u)^2}, \quad \lambda^{0,1} = u, \quad \lambda^{0,2} = u, \quad \lambda^+ = \tfrac{1}{2}u + \sqrt{c^2/\rho + (\tfrac{1}{2}u)^2}. \tag{3}$$

The four corresponding Riemann invariants are

$$\mathrm{d}J^- = \mathrm{d}p + \rho\lambda^-\mathrm{d}u, \quad J^{0,1} = v, \quad J^{0,2} = \alpha, \quad \mathrm{d}J^+ = \mathrm{d}p + \rho\lambda^+\mathrm{d}u. \tag{4}$$

These values are comparable with the results for the compressible Euler equations: there are two pressure characteristics, running left and right into the flow. However, there is no 'sound speed', the pressure waves have different velocities with respect to the flow. Note that $\lambda^- \leqslant 0$ and $\lambda^+ \geqslant 0$, always. The volume fraction α and the tangential velocity v are convected with the flow.

Now we construct a flux function based on an approximate solution of the Riemann problem. Like in Osher's flux function, this approximate solution is found with the characteristic waves. The initial states 0 and 1 of the problem are chosen as the states on the left and right side of a cell face. As mentioned, two pressure waves appear, always running left and right: the output state $\frac{1}{2}$ lies between these waves. The pressure jumps over the cell faces are not large, since incompressible flow is smooth (no shocks); therefore, we can linearise the outer waves. The pressures and the velocities between the waves are equal:

$$p_{1/2} = p_0 - \rho_0\lambda_0^+(u_{1/2} - u_0) = p_1 - \rho_1\lambda_1^-(u_{1/2} - u_1), \tag{5}$$

and thus,

$$\begin{aligned}
u_{1/2} &= u_0 + \frac{p_1 - p_0 + \rho_1\lambda_1^-(u_1 - u_0)}{\rho_1\lambda_1^- - \rho_0\lambda_0^+},\\
p_{1/2} &= p_0 - \rho_0\lambda_0^+\frac{p_1 - p_0 + \rho_1\lambda_1^-(u_1 - u_0)}{\rho_1\lambda_1^- - \rho_0\lambda_0^+}.
\end{aligned} \tag{6}$$

The other two state variables, v and α, do not change over the pressure waves. Therefore, they are chosen purely upwind:

$$\begin{aligned}
&v_{1/2} = v_0, \quad \alpha_{1/2} = \alpha_0 \quad \text{if } u_{1/2} \geqslant 0,\\
&v_{1/2} = v_1, \quad \alpha_{1/2} = \alpha_1 \quad \text{if } u_{1/2} < 0.
\end{aligned} \tag{7}$$

We construct convective fluxes with these state variables.

3.2. Diffusive flux

The diffusive fluxes in the momentum equations are modelled with central differences, e.g.:

$$(\mu u_x)_{1/2} = \mu_{1/2}\frac{u_1 - u_0}{\Delta x},$$
$$(\mu v_x)_{1/2} = \mu_{1/2}\frac{v_1 - v_0}{\Delta x}, \tag{8}$$

with $\mu_{1/2} = \alpha_{1/2}\mu_w + (1 - \alpha_{1/2})\mu_a$. This definition of μ follows when we assume equal strain for the two fluids and average the stress. For non-cartesian grids, (8) is approximated with a control volume approach.

3.3. Gravity

The gravity term in the y-momentum equation is added as a source term $-\rho_i g\Delta x\Delta y$ to the sum of the fluxes in each cell. With that source term present, the cell-face states for the Riemann solver must be adapted, as this solver has a strong coupling between pressure and velocity. And in a uniform horizontal flow with gravity, there is a vertical pressure gradient *without* a velocity gradient. When simple first-order cell face states are fed to the Riemann solver, then there is a pressure jump across each lower–upper cell face. The Riemann solver reacts to this by specifying an (incorrect) vertical velocity at the cell face. We reduce this problem by subtracting the pressure gradient due to gravity from the input states for the Riemann solver:

$$p^{\text{lower}}_{i,j+1/2} = p_{i,j} - \rho_{i,j} g\frac{\Delta y}{2},$$
$$p^{\text{upper}}_{i,j+1/2} = p_{i,j+1} + \rho_{i,j+1} g\frac{\Delta y}{2}. \tag{9}$$

Thus, in a horizontal uniform flow, the Riemann solver sees *no* pressure jumps (in that case, the pressures in (9) are equal) and computes zero vertical velocities. In other flows, the erroneous vertical velocities are reduced an order in the grid size.

4. Smoothing

The multigrid solution technique is based on a local relaxation technique or smoother. Starting from an initial solution, this smoother locally reduces the error in each cell. Repeated application in an iterative process makes the solution converge to the steady flow solution. The smoother must be well adapted to the type of equations to be solved. Here, with the very high Reynolds numbers and density ratios of water–air flow, a robust smoother is needed.

We use alternating line Gauss–Seidel smoothing. This procedure simultaneously resets the state in a line of cells, such that the net flux into all cells in that line is zero, given the current state in the other cells. This current state is the old state in the cells that have not yet been updated and the new state in all other cells. The direction of the lines is alternately horizontal and vertical. The updating of a line requires the simultaneous solution of the nonlinear flow equations in each cell of the line; this is done with a Newton–Raphson (NR) iterative root-finder. This procedure requires the derivatives of the fluxes with respect to the state. For the Osher solver, these derivatives are known analytically: they are computed together with the fluxes. The NR solution can be computed efficiently, because the linearised system to be solved in the iterations is block-tridiagonal: each cell influences only its two neighbours in the line. Therefore, the total work for a single NR step depends linearly on the total number of cells, just like for point smoothers.

An important aspect of the smoother is its smoothing factor, the highest amplification factor of an error component in the solution, on application of the smoother. When the flow equations are linearised, this factor can be found using Fourier analysis. This smoothing factor, for a range of error frequencies, is shown in Fig. 1. The horizontal and vertical frequency of the error mode are on the coordinate axes. The flow is linearised around the uniform flow $u = U = 1$, $v = 0$, $p = P$, $\alpha = A = 1$ (baseline settings). The high-frequency errors outside the inner box must be damped well, the multigrid procedure takes care of the low-frequency errors. Fig. 1a shows the excellent smoothing of alternating

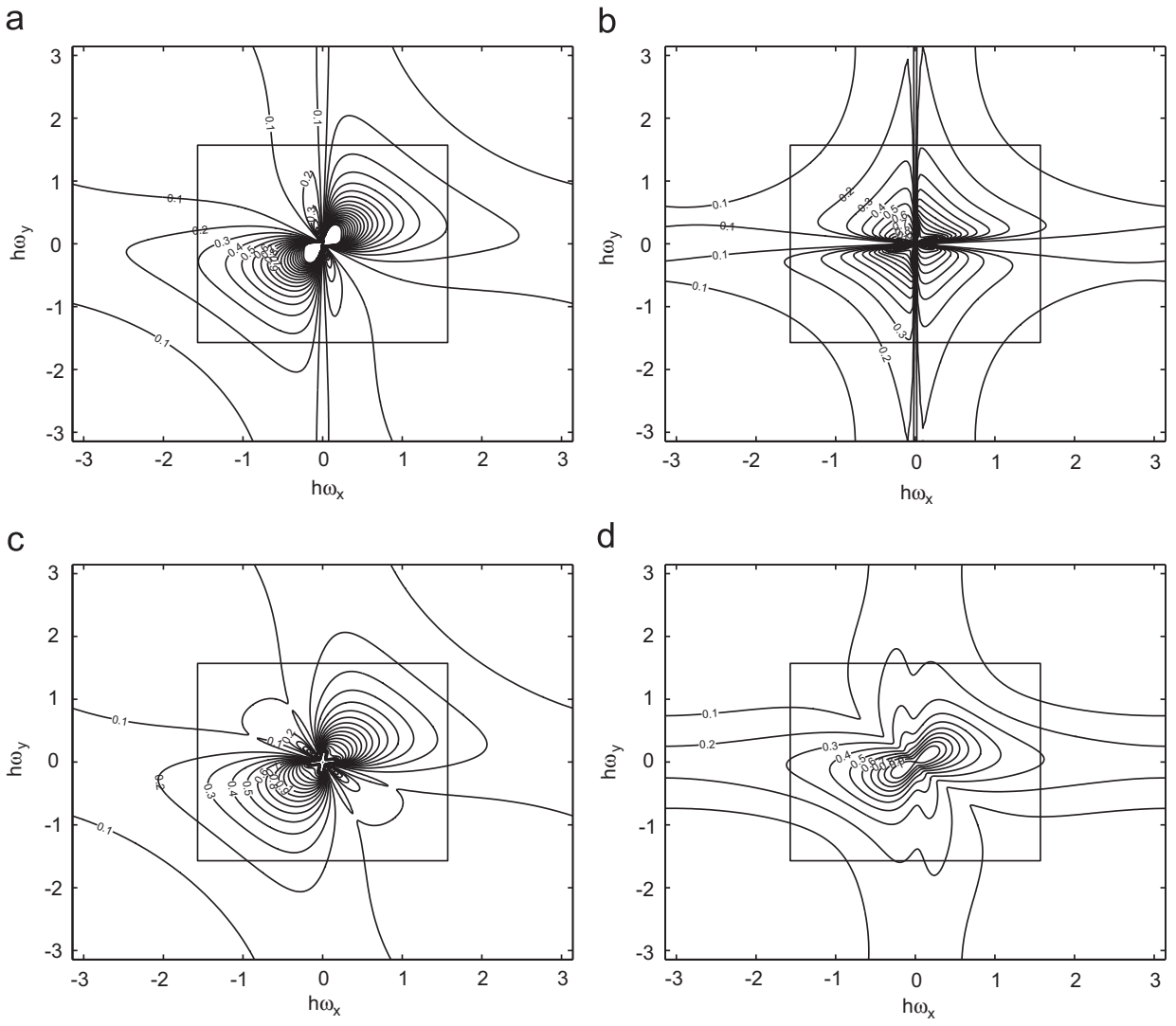

Fig. 1. Smoothing factor for alternating line smoothing with baseline settings (a), with $A = 0$ (b), with diagonal flow ↗ (c) and with underrelaxation (d).

line Gauss–Seidel: all high frequencies are damped with a factor 0.3 or less. In the other three figures, one parameter in the baseline is changed. In Fig. 1b, the density is $A = 0$. Clearly, the flow equations change a lot, but the smoothing is still excellent. And even when the flow direction is changed to $U = 1$, $V = 1$, in Fig. 1c, the damping is still fine. A small problem is the instability of the smoother for very low frequencies (seen in Fig. 1a and c), this can be removed by using a slight underrelaxation (Fig. 1d). Altogether, the line relaxation is an efficient smoother for a wide range of conditions.

5. Multigrid

The multigrid technique that is combined with the Gauss–Seidel smoothing from the last section is well documented [4]. Its principle is to smooth the high-frequency errors on the finest grid (called K) and the lower-frequency errors on a series of underlying coarser grids $0 \cdots K - 1$. The procedure used here is described below.

Denote the Navier–Stokes equations (1) including gravity, discretised on a grid k, by the operator F_k. Then the general problem on each grid is $F_k \mathbf{q}_k = \mathbf{f}_k$, the final problem to be solved is $F_K \mathbf{q}_K = 0$. We call the smoothing operator

M_k and introduce a prolongation operator P_{k-1}^k that copies a correction from one cell on grid $k-1$ to the four cells on grid k that lie in the same location. In the same way, a (four-point average) restriction operator R_k^{k-1} is defined for the defect. Then the nonlinear multigrid (NMG) procedure is defined recursively as follows. It is started on the finest grid, with $\mathbf{f}_K = 0$.

$\mathbf{q}_k^{n+1} =$ **recursive function** NMG$(k, \mathbf{q}_k^n, \mathbf{f}_k^n)$

$$\tilde{\mathbf{q}}_k^n = (M_k)^{q_1}\mathbf{q}_k^n \quad (q_1 \text{ pre-relaxation steps}), \tag{10a}$$

if $k \neq 0$ **then**

$$\mathbf{f}_{k-1}^n = F_{k-1}\mathbf{q}_{k-1}^n - s_{k-1}^n R_k^{k-1}(F_k\tilde{\mathbf{q}}_k^n - \mathbf{f}_k^n) \quad \text{(source term)}, \tag{10b}$$

$$\mathbf{q}_{k-1}^{n+1} = \text{NMG}(k-1, \mathbf{q}_{k-1}^n, \mathbf{f}_{k-1}^n) \quad \text{(MG on coarser grid)}, \tag{10c}$$

$$\tilde{\tilde{\mathbf{q}}}_k^n = \tilde{\mathbf{q}}_k^n + \frac{1}{s_{k-1}^n} P_{k-1}^k(\mathbf{q}_{k-1}^{n+1} - \mathbf{q}_{k-1}^n) \quad \text{(correction)}, \tag{10d}$$

end if

$$\mathbf{q}_k^{n+1} = (M_k)^{q_2}\tilde{\tilde{\mathbf{q}}}_k^n \quad (q_2 \text{ post-relaxation steps}). \tag{10e}$$

This procedure is standard. The only problem is, that the nonlinear operators F_k cannot handle large source terms. For example, a source term could specify a sink for water in a cell whose neighbours contain only air. The only way to get a net inflow of water in that cell is to take a negative amount of water in the cell itself. This may lead to negative densities. Or if a large sink of mass is specified, we may get a cell with inflow on all cell faces. It is impossible to set the net inflow of water into such cells, since the flux of α is determined purely upwind: it does not depend on α in the cells itself.

Therefore, the source term $\mathbf{f}_{k-1}^n$ must always be sufficiently small. For small source terms, the operator F_k gives sensible output and the NR processes converge. The two terms of $\mathbf{f}_{k-1}^n$ (Eq. (10b)) are made small as follows:

$F_{k-1}\mathbf{q}_{k-1}^n$: Pick the right starting solution $\mathbf{q}_{k-1}^n$. The ideal $\mathbf{q}_{k-1}^n$ is that state, for which $F_{k-1}\mathbf{q}_{k-1}^n \approx 0$. This state is available, as we use an FMG procedure, i.e., we set the first solution $\mathbf{q}_k^0$ on each grid k, from grid 1 upwards, by solving $F_{k-1}\mathbf{q}_{k-1} = 0$ on the next coarsest grid and prolongating this solution to grid k. As a bonus, we get $\mathbf{q}_{k-1}$ for which $F_{k-1}\mathbf{q}_{k-1} \approx 0$. These states are used as the initial solution $\mathbf{q}_{k-1}^n$ for *each* coarse grid correction step.

$s_{k-1}^n R_k^{k-1}(F_k\tilde{\mathbf{q}}_k^n - \mathbf{f}_k^n)$: This term is made small by simply setting the scaling s_{k-1}^n small whenever the other term is large. The coarse grid correction does not depend much on the value of s, so we use a straightforward choice for s_{k-1}^n:

$$s_{k-1}^n = \min\left(1, \frac{1}{D\max|R_k^{k-1}(F_k\tilde{\mathbf{q}}_k^n - \mathbf{f}_k^n)|}\right). \tag{11}$$

$D = 10^2$ works in practice. Also, more elaborate choices for s_{k-1}^n are possible.

6. Cahouet test case

As a test, the flow in a channel with a bottom bump is computed. Experimental results for this test are given by Cahouet [1]. Two cases are computed.

The first has a Froude number of 2.05 (based on inflow water height). The inflow velocity profile is as measured by Cahouet. To keep the flow laminar, the Reynolds number is taken lower than in the experiment: $Re = 1520$. To prevent too thick boundary layers, the top and bottom are modelled as slip walls. The bump has a thickness of 44% of the water height. The curvilinear grid has 128×512 cells. In the velocity plot (Fig. 2a), we see low-velocity regions near the leading and trailing edge of the bump and high-velocity regions above the bump and especially in the air region near the top wall. The volume fraction (Fig. 2b) shows the water surface. At the outflow wall, the interface is spread over five cells. A grid convergence study (Fig. 2c) shows a nearly converged solution and rather good agreement with Cahouet's experiment. The slightly flatter wave may be caused by the absence of a boundary layer.

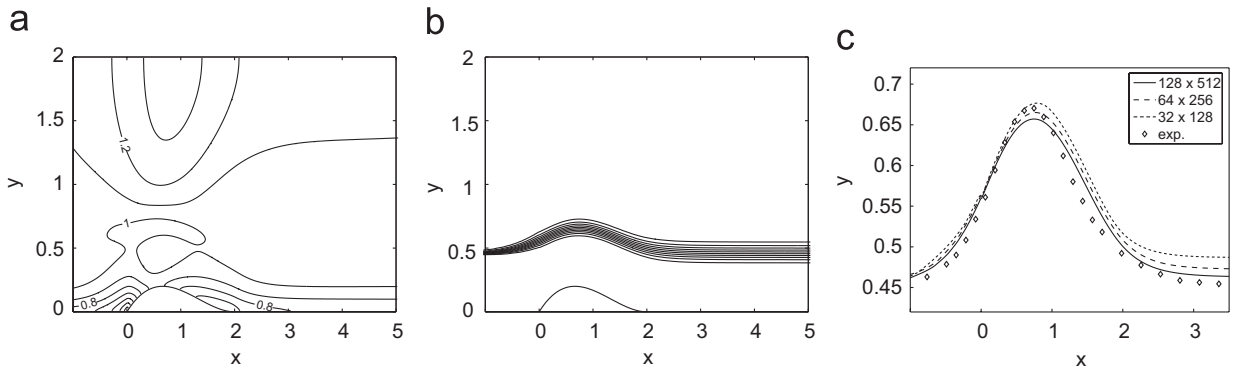

Fig. 2. Cahouet test case, $Fr = 2.05$. Speed (a), volume fraction (b) and a comparison (c) of the $\alpha = 0.5$ isolines with Cahouet's experiment (averaged wave height). The y-coordinate is stretched for clarity, plot (c) is zoomed in.

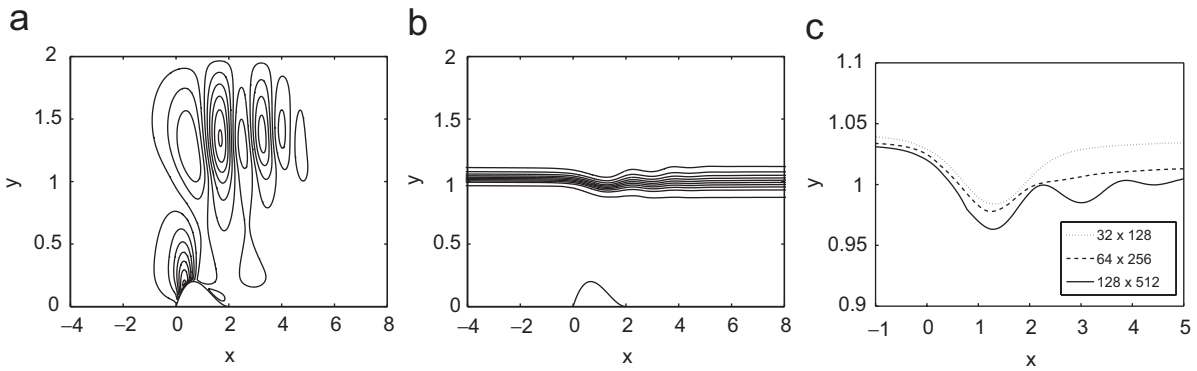

Fig. 3. Cahouet test case, $Fr = 0.43$. Vertical velocity (a), volume fraction (b) and a grid convergence study of the water surface (c).

The second test case is a flow with $Fr = 0.43$, $Re = 3333$, uniform inflow and no-slip lower wall. The grid is stretched on the inflow and outflow side to prevent wave reflection. The stretched cells have a higher aspect ratio (> 10) than the more or less square cells in the centre, but this causes no convergence problems. Due to the stretching, the flow approaching the bump has developed a boundary layer on the bottom. The solution has a wave train developing behind the bump (Fig. 3b). The velocity is continuous over the water surface, the upward— downward motion in the wave train can be seen both in the air and in the water (Fig. 3a). Note that these pictures do not show the entire grid, a part of the stretched grid extends beyond the picture boundaries. A grid refinement study (Fig. 3c) shows that the current solution is far from converged. This is mostly caused by laminar separation from the bottom bump, which is not present in the (turbulent) experiment. So for the accurate solution of this problem, a higher-order method with turbulence modelling is needed.

To assess the efficiency of multigrid, the second test problem is solved both with multigrid on seven grids and with pure Gauss–Seidel smoothing on a single grid. Multigrid convergence improves when the coarsest grid has as few cells as possible; our coarsest grid is the coarsest one that still sees the bottom bump (2×8 cells). Convergence plots are shown in Fig. 4. The NMG strategy (Section 5) gives the 'sawtooth' convergence graph 4a. Only the last 26 are the (expensive) iterations on the finest level. A table of the CPU times on a 1667 MHz PC (Table 1) shows that the use of multigrid greatly reduces the computation time, compared with the single-grid line smoothing. A comparison with single-grid time marching to convergence was not done, but time marching is almost certainly even less efficient than the single-grid line smoother.

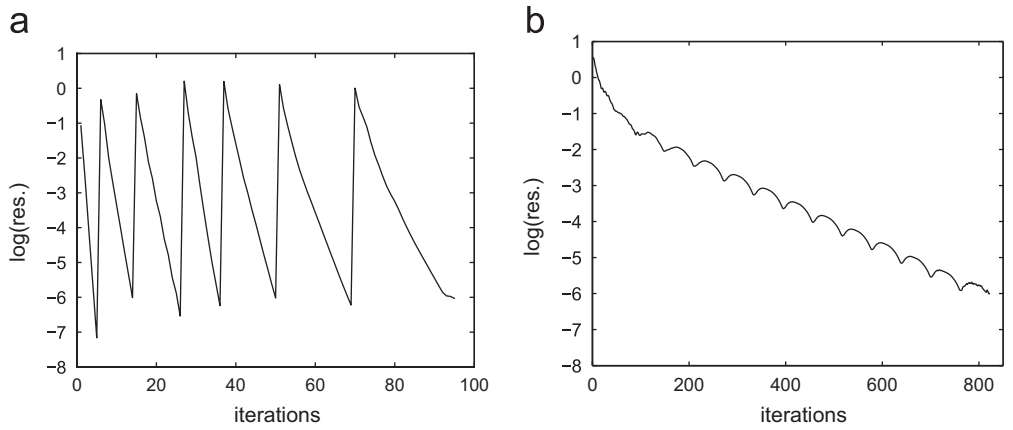

Fig. 4. Convergence histories for multigrid (a) and single grid Gauss–Seidel (b).

Table 1
Cahouet testcase: iterations and CPU time needed for convergence

Method	Iterations	CPU time (s)
Multigrid (seven grids)	95 (26)	690
Single grid line Gauss–Seidel	822	17184

7. Conclusion

A surface-capturing method has been developed for the two-fluid incompressible Navier–Stokes equations with gravity. The steady flow equations are solved efficiently with a combination of multigrid and line Gauss–Seidel smoothing.

The flux discretisation is derived from an artificial compressibility approach. This formulation is used to construct a linearised Osher flux function for the convective fluxes. This flux function has a strong pressure–velocity coupling and is highly nonlinear, yet it gives a stable Gauss–Seidel iteration. A standard multigrid procedure is used. Due to the nonlinear behaviour of the flux function, the coarse grid correction steps cannot handle large source terms. This problem is solved by using smart initial solutions on the coarse grids and by a scaling of the defect from the finer grids.

A test case, the flow in a channel with a bottom bump, shows that the method gives good solutions for different flow regimes and that it is much faster than comparable solution techniques without multigrid. For better accuracy, a higher-order method with turbulence modelling is needed.

References

[1] J. Cahouet, Etude numérique et expérimentale du problème bidimensionnel de la résistance de vagues non-linéaire, Technical Report 185, Ecole Nationale Supérieure de Techniques Avancées, Paris, 1984.
[2] A.J. Chorin, A numerical method for solving incompressible viscous flow problems, J. Comput. Phys. 2 (1976) 12–26.
[3] E. Dick, J. Linden, A multigrid method for steady incompressible Navier–Stokes equations based on flux-difference splitting, Int. J. Numer. Methods Fluids 14 (1992) 1311–1323.
[4] W. Hackbusch, Multi-Grid Methods and Applications, Springer, Berlin, 1985.
[5] T. Hino (Ed.), CFD Workshop Tokyo 2005, National Maritime Research Institute, Tokyo, 2005.
[6] L. Larsson, F. Stern, V. Bertram (Eds.), Gothenburg 2000, A Workshop on Numerical Ship Hydrodynamics, Chalmers University of Technology, Göteborg, 2000.
[7] E. Turkel, Preconditioned methods for solving the incompressible and low speed compressible equations, J. Comput. Phys. 72 (1987) 277–298.