1991

E.D. de Goede

On the numerical treatment of the advective terms in 3D shallow water models

Department of Numerical Mathematics Report NM-R9110 April

CWI, nationaal instituut voor onderzoek op het gebied van wiskunde en informatica

CWI is the research institute of the Stichting Mathematisch Centrum, which was founded on February 11, 1946, as a non-profit institution aiming at the promotion of mathematics, computer science, and their applications. It is sponsored by the Dutch Government through the Netherlands organization for scientific research (NWO).

On the numerical treatment of the advective terms in 3D shallow water models

Erik D. de Goede
CWI
P.O. Box 4079, 1009 AB Amsterdam,
The Netherlands

In this paper we present a numerical method for the three-dimensional shallow water equations. These equations describe flows in e.g., shallow seas, rivers and estuaries. Since three-dimensional models require a great computational effort, the method is constructed in such a way that it fully exploits the facilities of vector and parallel computers. This two-stage method, which is unconditionally stable, is applied to a problem involving the development of a circulation in a rectangular basin and to a river problem in which a jetty has been situated.

1980 Mathematics Subject Classification: 65M10, 76D99.
Key Words & Phrases: three-dimensional shallow water equations, method of lines, splitting method, advective terms, vector and parallel computers.
Note: This work was supported by the Ministry of Transport and Public Works (RWS).
Note: This report will be submitted for publication.

1. Introduction

In the past, many numerical methods for the simulation of water flows were based on the so-called two-dimensional shallow water equations. These equations can be obtained from the three-dimensional equations by averaging over the vertical co-ordinate. They only yield the water elevation and the depth-averaged velocities. However, there are many cases in which the vertical structure of the flow is required. For example, when the dispersion of a pollution is desired. In this paper we will develop a numerical method for the three-dimensional shallow water equations. The three-dimensional models are an order of magnitude more expensive than the two-dimensional models and it is therefore important to construct methods that are not only robust and accurate, but also able to fully exploit the facilities of vector and parallel computers.

In [3] we developed an unconditionally stable two-stage method for the three-dimensional shallow water equations in which the advective terms had been omitted and we focussed on the stability conditions imposed by the vertical diffusion and by the terms describing the propagation of the surface waves. In this paper we will incorporate the advective terms into this method.

For the model without advective terms, the unconditionally stable method has been compared with conditionally stable methods [4]. The unconditionally stable method yields the most accurate results. The method is also more efficient, because large time steps can be used. For two-dimensional models, this approach is very similar to the one described in [11], where its feasibility for practical computations has been shown. For three-dimensional models the efficiency of our method is even higher than for two-dimensional models [3].

For the numerical treatment of the advective terms, we will follow the approach developed in [11]. The introduction of the advective terms results in a hardly more complicated system. At the first stage the implicit treatment of the

advective terms yields a large, non-symmetric, linear system. For its solution we will develop a Jacobi-type iteration method. When only one iteration is performed, this corresponds with an explicit treatment of the advective terms. Thus, the method offers the facility of both an explicit and an implicit treatment of the advective terms.

At the second stage the method is comparable with the method developed for the model without the advective terms [3]. Again, a sequence of linear systems has to be solved at this stage. This system is solved by a conjugate gradient method in which a preconditioner based on smoothing is used [3]. It appeared that this iteration method is highly suitable for vector and parallel computers.

In order to facilitate the comparison with existing numerical methods, we will apply our method to test problems from the literature. In the first experiment we will examine the development of a circulation in a rectangular basin with dimensions representative of the North Sea [1,6]. In the second test problem we will study a river flow past a jetty [9].

2. Mathematical model

In this section we will describe a mathematical model for the three-dimensional shallow water equations. The three-dimensional model in sigma co-ordinates is given by [1,10]

$$\frac{\partial u}{\partial t} = - u\frac{\partial u}{\partial x} - v\frac{\partial u}{\partial y} - \omega\frac{\partial u}{\partial \sigma} + fv - g\frac{\partial \zeta}{\partial x} + \lambda\frac{\partial^2 u}{\partial x^2} + \lambda\frac{\partial^2 u}{\partial y^2} + \frac{1}{h^2}\frac{\partial}{\partial \sigma}\left(\mu\frac{\partial u}{\partial \sigma}\right) \tag{2.1}$$

$$\frac{\partial v}{\partial t} = - u\frac{\partial v}{\partial x} - v\frac{\partial v}{\partial y} - \omega\frac{\partial v}{\partial \sigma} - fu - g\frac{\partial \zeta}{\partial y} + \lambda\frac{\partial^2 v}{\partial x^2} + \lambda\frac{\partial^2 v}{\partial y^2} + \frac{1}{h^2}\frac{\partial}{\partial \sigma}\left(\mu\frac{\partial v}{\partial \sigma}\right) \tag{2.2}$$

$$\omega = \frac{1}{h}\left\{ - (1-\sigma)\left(\frac{\partial}{\partial x}\left(h\int_0^1 u\,d\sigma\right) + \frac{\partial}{\partial y}\left(h\int_0^1 v\,d\sigma\right)\right) + \frac{\partial}{\partial x}\left(h\int_\sigma^1 u\,d\sigma\right) + \frac{\partial}{\partial y}\left(h\int_\sigma^1 v\,d\sigma\right)\right\} \tag{2.3}$$

$$\frac{\partial \zeta}{\partial t} = - \frac{\partial}{\partial x}\left(h\int_0^1 u\,d\sigma\right) - \frac{\partial}{\partial y}\left(h\int_0^1 v\,d\sigma\right), \tag{2.4}$$

where the symbols are listed in Appendix A. The equations (2.1)-(2.3) are the momentum equations and (2.4) denotes the continuity equation. In our model the accelerations in the vertical direction have been neglected, because they are very small, particularly when compared with the acceleration due to gravity. This is known as the so-called shallow water approximation.

In the vertical, the domain is bounded by the bottom topography and the time-dependent water elevation. To ensure that the three-dimensional domain is also constant in the vertical direction, system (2.1)-(2.4) has been transformed into the constant interval [0,1] by the sigma transformation [8]

$$\sigma = \frac{\zeta - z}{d + \zeta}, \quad \text{where } -d \le z \le \zeta \text{ and } 1 \ge \sigma \ge 0\,. \tag{2.5}$$

The relation between the untransformed (physical) vertical velocity w and the transformed velocity ω is given by [1,10]

$$w = -\omega h + \frac{\partial \zeta}{\partial t} - \sigma \frac{\partial h}{\partial t} + u \left(\frac{\partial \zeta}{\partial x} - \sigma \frac{\partial h}{\partial x} \right) + v \left(\frac{\partial \zeta}{\partial y} - \sigma \frac{\partial h}{\partial y} \right) .$$

The domain is defined by

$$0 \le x \le L , \quad 0 \le y \le B \quad \text{and} \quad 1 \ge \sigma \ge 0 ,$$

i.e., a rectangular basin. Owing to the sigma transformation in the vertical, the domain is constant in time. The boundary conditions at the sea surface ($\sigma = 0$) are given by [1]

$$\omega(x,y,0,t) = 0 , \quad \left(\mu \frac{\partial u}{\partial \sigma} \right)_{\sigma=0} = - \frac{h}{\rho} W_f \cos(\varphi) \quad \text{and} \quad \left(\mu \frac{\partial v}{\partial \sigma} \right)_{\sigma=0} = - \frac{h}{\rho} W_f \sin(\varphi) .$$

Similarly, at the bottom ($\sigma = 1$) we have

$$\omega(x,y,1,t) = 0 , \quad \left(\mu \frac{\partial u}{\partial \sigma} \right)_{\sigma=1} = - h \frac{g\, u_d}{C^2} \sqrt{u_d^2 + v_d^2} \quad \text{and} \quad \left(\mu \frac{\partial v}{\partial \sigma} \right)_{\sigma=1} = - h \frac{g\, v_d}{C^2} \sqrt{u_d^2 + v_d^2} ,$$

where the last two conditions represent a quadratic law of bottom friction.

3. Numerical integration

To discretize system (2.1)-(2.4), we first apply a finite difference space discretization on a spatial grid that is staggered in both the horizontal and the vertical direction (see [2,3]). Figure 1 shows the horizontal grid spacing. The computational domain is covered by an nx·ny·ns rectangular grid. On this grid the spatial derivatives are replaced by second-order finite differences, which results into a semi-discretized system of dimension nx·ny·(3ns+1). Owing to the sigma transformation (2.5), we have a constant number of grid layers in the vertical direction. In what follows, **U**(t) is a grid function whose components $U_{i,j,k}(t)$ approximate the velocity u(t). The components $U_{i,j,k}(t)$ are numbered lexicographically. Likewise, **V**, **Ω**, **Z**, **D** and **H** are grid functions approximating v, ω, ζ, d and h, respectively. Note that **Z**, **D** and **H** are two-dimensional unknowns. The grid sizes in x- and y-direction are denoted by Δx and Δy, respectively. In the vertical direction, we choose a varying grid of thickness $\Delta\sigma_k$, where k refers to the kth grid layer from the surface. Hence, it is possible to increase the resolution near the surface and the bottom.

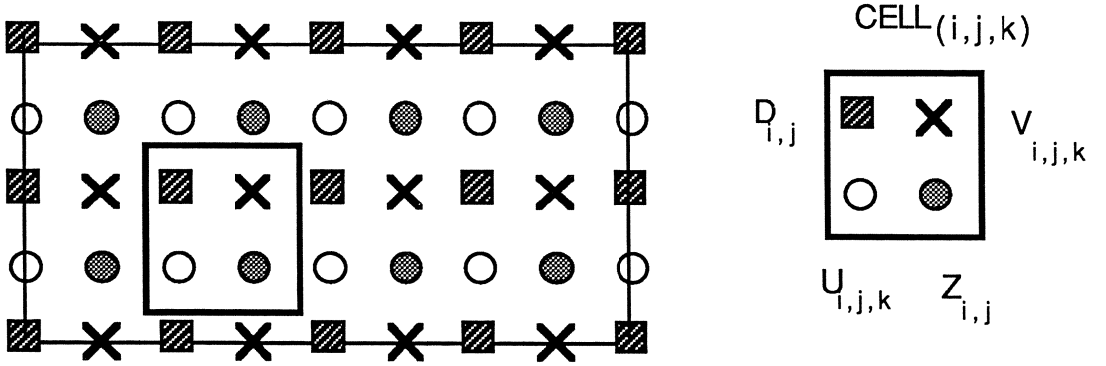

Figure 1. The staggered grid in the horizontal direction

We now describe the time integrator for the semi-discrete system. Our method consists of two stages. At the first stage we compute intermediate approximations, indicated by an asterisk. The upper indices denote the time level, whereas the lower indices refer to the discretization in space. The first stage reads

$$
\begin{aligned}
U^* &= U^n + \tfrac{1}{2}\tau\{-U^n U^*_x - V^n U^*_{1y} - \Omega^n U^*_\sigma + fV^n - gZ^n_{0x} + \lambda U^*_{xx} + \lambda U^*_{yy} + \frac{\mu}{H^2}U^*_{\sigma\sigma}\} \\
V^* &= V^n + \tfrac{1}{2}\tau\{-U^* V^*_{1x} - V^n V^*_y - \Omega^n V^*_\sigma - fU^* - gZ^n_{0y} + \lambda V^*_{xx} + \lambda V^*_{yy} + \frac{\mu}{H^2}V^*_{\sigma\sigma}\} \\
\Omega^* &= \frac{1}{H^*}\{-(1-\sigma)((H^*\int_0^1 U^* d\sigma)_{0x} + (H^*\int_0^1 V^* d\sigma)_{0y}) + (H^*\int_\sigma^1 U^* d\sigma)_{0x} + (H^*\int_\sigma^1 V^* d\sigma)_{0y}\} \\
Z^* &= Z^n + \tfrac{1}{2}\tau\{-(H^n\int_0^1 U^n d\sigma)_{0x} - (H^n\int_0^1 V^n d\sigma)_{0y}\},
\end{aligned}
\tag{3.1a}
$$

where τ denotes the time step. Next, we perform the second stage to arrive at the new time level.

$$
\begin{aligned}
U^{n+1} &= U^* + \tfrac{1}{2}\tau\{-U^{n+1} U^*_x - V^* U^*_y - \Omega^* U^*_\sigma + fV^* - gZ^{n+1}_{0x} + \lambda U^*_{xx} + \lambda U^*_{yy} + \frac{\mu}{H^2}U^*_{\sigma\sigma}\} \\
V^{n+1} &= V^* + \tfrac{1}{2}\tau\{-U^* V^*_x - V^{n+1} V^*_y - \Omega^* V^*_\sigma - fU^* - gZ^{n+1}_{0y} + \lambda V^*_{xx} + \lambda V^*_{yy} + \frac{\mu}{H^2}V^*_{\sigma\sigma}\} \\
\Omega^{n+1} &= \frac{1}{H^{n+1}}\{-(1-\sigma)((H^{n+1}\int_0^1 U^{n+1} d\sigma)_{0x} + (H^{n+1}\int_0^1 V^{n+1} d\sigma)_{0y}) + (H^{n+1}\int_\sigma^1 U^{n+1} d\sigma)_{0x} + (H^{n+1}\int_\sigma^1 V^{n+1} d\sigma)_{0y}\} \\
Z^{n+1} &= Z^* + \tfrac{1}{2}\tau\{-(H^{n+1}\int_0^1 U^{n+1} d\sigma)_{0x} - (H^{n+1}\int_0^1 V^{n+1} d\sigma)_{0y}\}.
\end{aligned}
\tag{3.1b}
$$

In Appendix B a detailed description of the space discretization is given. When applied to two-dimensional problems, this method is very similar to the method developed in [11].

In [3] it has been proved that method (3.1) is unconditionally stable for a model in which the advective terms and been omitted. For a model including the Coriolis term and the advective terms, method (3.1) has the same good stability properties.

Method (3.1) is first-order accurate in time. To obtain second-order accuracy, the Coriolis term, the mixed advective terms and the bottom friction term should be adjusted. However, a second-order treatment of these terms would decrease the computational efficiency of our time splitting method dramatically. This will be explained in the next section. It should be noted that the diffusion terms and the terms describing the propagation of the surface waves are second-order accurate in time.

Almost all spatial derivatives are discretized in a symmetric and therefore non-dissipative way. However, at the first stage, the mixed advective terms $v\partial u/\partial y$ and $u\partial v/\partial x$ are approximated by an upwind discretization. In the numerical experiments, the resulting dissipation is just enough to suppress spurious oscillations. The upwind discretization, which is of fourth-order magnitude, does not lead to an undesired damping of the solution. This approach has been developed in [9].

At the first stage our time splitting method requires the successive solution of two large, non-symmetric, linear systems (first a system for the U-component, next a system for the V-component). At the second stage a nonlinear system has to be solved. These systems and the iteration methods for its solution will be discussed in the next section.

4. Solving the systems

The structure of the systems at both stages determines the efficiency of our unconditionally stable method. At the first stage the Ω- and Z-component can be computed straightforwardly. For the U- and V-component, the method requires the successive solution of two non-symmetric, linear systems of dimension nx·ny·ns. The system for the U-component may be written in the form

$$U^* + \tfrac{1}{2}\tau \{U^n U_x^* + V^n U_y^* + \Omega^n U_\sigma^* - \lambda U_{xx}^* - \lambda U_{yy}^* - \frac{\mu}{H^2} U_{\sigma\sigma}^*\} = B^n , \tag{4.1}$$

where B^n contains the terms at $t=n\tau$. For the V-component, we have a similar system. The matrix at the left-hand side of system (4.1) contains nine non-zero diagonals. Seven diagonals are due to the discretizations in the horizontal direction and three are due to the vertical derivatives, with one overlapping (main) diagonal. Some of these diagonals contain zero elements. System (4.1) may be written as

$$(I + \tfrac{1}{2}\tau D_h + \tfrac{1}{2}\tau D_v) U^* = B^n , \tag{4.1'}$$

where the matrix D_v represents the discretizations in the vertical direction and the contribution on the main diagonal of the discretizations in the horizontal direction. The remaining (six) diagonals resulting from the horizontal derivatives are represented by the matrix D_h.

For the solution of system (4.1'), we apply the preconditioned Jacobi-type method

$$U_{k+1} = U_k + \alpha (I + \tfrac{1}{2}\tau D_v)^{-1} \{ B_u^n - (I + \tfrac{1}{2}\tau D_h + \tfrac{1}{2}\tau D_v) U_k \} , \qquad k=1,2,3,..., \tag{4.2}$$

where α is a relaxation parameter and U_k denotes the kth iterate with $U_0=U^n$. The term $\alpha (I + \tfrac{1}{2}\tau D_v)^{-1}$ represents the preconditioning. Method (4.2) can be written in the more efficient form

$$U_{k+1} = U_k + \alpha \{ (I + \tfrac{1}{2}\tau D_v)^{-1} (B_u^n - \tfrac{1}{2}\tau D_h U_k) - U_k \} . \tag{4.2'}$$

At each iteration step the implicit operator $(I + \frac{1}{2}\tau D_v)^{-1}$ requires the solution of nx·ny tridiagonal systems of dimension ns. For its solution we apply the Gaussian Elimination (double sweep) method, which requires a minimal number of operations. Since this is a recursive method, it is an unattractive method on vector and parallel computers. However, we make use of the fact that a large number of tridiagonal systems of the same dimension has to be solved. Therefore, the systems can be solved efficiently in a vector-parallel mode [2].

For the relaxation parameter α we choose either

$$\alpha = 1 \quad \text{or} \quad \alpha = \frac{1}{\rho(I + \tfrac{1}{2}\tau D_h)} ,$$

where $\rho(\cdot)$ denotes the spectral radius. The Jacobi-type method (4.2') starts with $\alpha=1$. As soon as the residue increases, we switch to the second choice. For this choice the iteration process always converges as opposed to the choice $\alpha=1$. If the method converges for $\alpha=1$, then this convergence is faster than for $\alpha=1/(\rho(I + \frac{1}{2}\tau D_h))$.

At the second stage the equations for the U- and V-component are linear and are not coupled with each other. They are only coupled with the equation for the Ω- and Z-component. If U^{n+1}, V^{n+1} and Z^{n+1} have been computed, then the

values for the Ω-component are computed. Owing to our choice of the time splitting, the components $\mathbf{U}^{n+1}$ and $\mathbf{V}^{n+1}$ can be eliminated and a system merely in the unknown $\mathbf{Z}^{n+1}$ results. In order to accomplish such an elimination, the Coriolis term, the mixed advective terms and the bottom friction term have been treated first-order accurate in time. A second-order treatment of these terms would have resulted in a much more complicated system. In that case, the U- and V-component can not be eliminated easily.

We will now describe the system for each cell (i,j) of component Z. This system reads

$$Z_{i,j}^{n+1} - \frac{\tau^2 g}{4(\Delta x)^2}\left\{ \bar{R}_{i+1,j}^{n+1}\left(Z_{i+1,j}^{n+1} - Z_{i,j}^{n+1}\right) - \bar{R}_{i,j}^{n+1}\left(Z_{i,j}^{n+1} - Z_{i-1,j}^{n+1}\right)\right\} \qquad (4.3)$$

$$- \frac{\tau^2 g}{4(\Delta y)^2}\left\{ \tilde{R}_{i,j}^{n+1}\left(Z_{i,j-1}^{n+1} - Z_{i,j}^{n+1}\right) - \tilde{R}_{i,j-1}^{n+1}\left(Z_{i,j}^{n+1} - Z_{i,j-1}^{n+1}\right)\right\} = B_{i,j}^{n+1/2}, \quad \text{for} \quad \begin{matrix} i=1,\ldots,nx \\ j=1,\ldots,ny \end{matrix},$$

where

$$\bar{R}_{i,j}^{n+1} = \left\{ \tfrac{1}{2}\left(Z_{i,j}^{n+1} + Z_{i-1,j}^{n+1}\right) + \tfrac{1}{2}\left(D_{i,j} + D_{i,j-1}\right)\right\} / \left\{ 1 + \sum_{k=1}^{ns} \Delta\sigma_k \tfrac{1}{2}\tau \left(U_x^{n+1/2}\right)_{i,j,k}\right\},$$

$$\tilde{R}_{i,j}^{n+1} = \left\{ \tfrac{1}{2}\left(Z_{i,j}^{n+1} + Z_{i,j+1}^{n+1}\right) + \tfrac{1}{2}\left(D_{i,j} + D_{i+1,j}\right)\right\} / \left\{ 1 + \sum_{k=1}^{ns} \Delta\sigma_k \tfrac{1}{2}\tau \left(U_y^{n+1/2}\right)_{i,j,k}\right\}.$$

System (4.3) is a nonlinear equation, because $R_{i,j}$ contains the component $Z_{i,j}$. When this system has been solved, the values for the U- and V-component can be computed straightforwardly. System (4.3) can be written in the form

$$A(\mathbf{Z}^{n+1})\,\mathbf{Z}^{n+1} = \mathbf{B}^{n+1/2},$$

where $\mathbf{B}^{n+1/2}$ contains the terms at $t=(n+1/2)\tau$. For its linearization we introduce the iteration process

$$A(\mathbf{Z}^{(q)})\,\mathbf{Z}^{(q+1)} = \mathbf{B}^{n+1/2}, \qquad (4.4)$$

where $\mathbf{Z}^{(0)}=\mathbf{Z}^{n+1/2}$ and the upper index (q) denotes the iteration index. The matrix $A(\mathbf{Z}^{(q)})$ is symmetric. For the solution of system (4.4) we developed a conjugate gradient method in which a preconditioner based on smoothing is used [3].

It should be noted that the water elevation is the only unknown in system (4.4). This system is for both two-dimensional and three-dimensional models of the same (two-dimensional) structure and thus of the same computational complexity. Therefore, the time integration method (3.1) is relatively more efficient for three-dimensional problems than for two-dimensional ones.

5. Numerical experiments

In order to examine its accuracy and computational efficiency, method (3.1) has been applied to a problem with a closed basin and on a river problem in which a jetty has been situated. In the first experiment, we used a closed rectangular basin of 400 km by 800 km with a constant depth of 65 m. The other parameters were f=1.22e-4 s^{-1}, g=9.81 m/s^2, ρ=1025 kg/m^3, φ = 90° , C=70 $m^{1/2}/s$, λ=0.0 m^2/s and μ=0.065 m^2/s. This experiment has also been carried out

in [1,6]. The water was initially at rest and the motion in the closed basin was generated by a constant wind stress of 1.5 kg/ms^2. The computations were performed on a grid with nx=10, ny=18 and ns=11, which implies horizontal mesh sizes of 400/9 km and 800/17 km, respectively. In the vertical direction we chose for every k: $\Delta\sigma_k$=1/ns. We integrated over a period of 100 hours.

At the end of the integration process the numerical solution was compared with a reference solution computed on the same grid with τ=30 seconds. The reference solution may be considered as an almost exact solution of our semi-discretized system. Thus, the accuracy results listed in this section represent the error due to the time integration.

The experiments have been carried out on an Alliant FX/4, which is a mini-supercomputer having four vector processors. In all experiments we have used both the vector optimization and the parallel optimization.

To represent the results we use the following notation:

ERR-· : maximal global error of either u, v or ζ at the end point T = 100 hours
COMP : computation time on the Alliant FX/4.

We required that the residue for the two iteration methods (viz., the Jacobi-type method (4.2') and the CG method) drops below the tolerance 10^{-3}. By choosing this value, we obtain a good compromise between the accuracy and the computational costs. For this test problem the results are listed in Table 5.1.

Table 5.1. Test problem with a rectangular basin.

τ (s)	ERR-u (m/s)	ERR-v (m/s)	ERR-ζ (m)	COMP (s)
360	0.004	0.002	0.005	147.3
1800	0.006	0.005	0.009	30.6
3600	0.009	0.006	0.013	15.5
6000	0.011	0.022	0.035	10.0
7200	0.029	0.031	0.135	8.7

In this experiment, the maximal values for u, v and ζ were about 0.2 m/s, 0.4 m/s and 1.0 m, respectively. In Table 5.1 no accuracy results have been listed for the vertical velocity w, because these velocities were very small. In this experiment the influence of the advective terms was very limited. Therefore, an explicit treatment of these terms (i.e., only one iteration of the Jacobi-type method (4.2')) was sufficient. It should be noted that one iteration of this method is sufficient to obtain an implicit treatment of the vertical diffusion term. Figures 2a-b show the vertical profiles of the U- and V-component at the centre of the rectangular basin at T=100 h. At that time the steady state was reached for moderate values of the time step. Table 5.1 clearly shows that this is not the case for τ=7200 s at T=100 h. If we integrated over a longer period with a time step of 7200 s, then the solution became stationary too. The numerical results are in agreement with the results in [1,6].

For several numerical methods developed for the 3D shallow water equations, the time step is restricted by the CFL condition $\tau < \Delta / \sqrt{2gh}$, where $\Delta=\min(\Delta x,\Delta y)$. Examples of such conditionally stable methods are described in [1, 2, 6 and 7]. In our experiment, this results in a maximally stable time step of about 1245 s. However, in [1,6] the time step was significantly below the CFL condition. In [6] a time step of 360 s was used. The splitting method in [1] used τ=720 s for the advective terms and τ=180 s for the terms describing the propagation of the surface waves. For method (3.1) much larger time steps were possible. Moreover, even for a large time step of 3600 s, the relative errors were still very small.

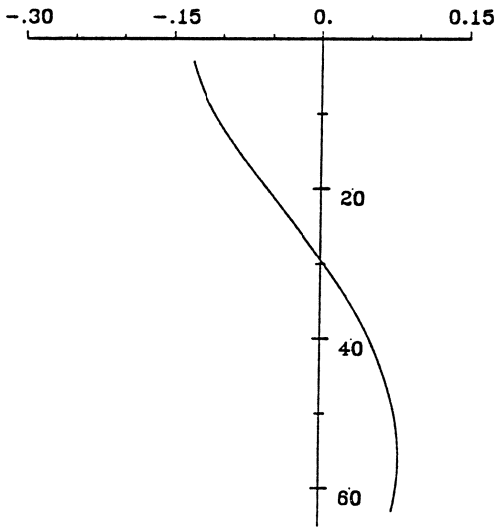

Figure 2a. The vertical U-profile

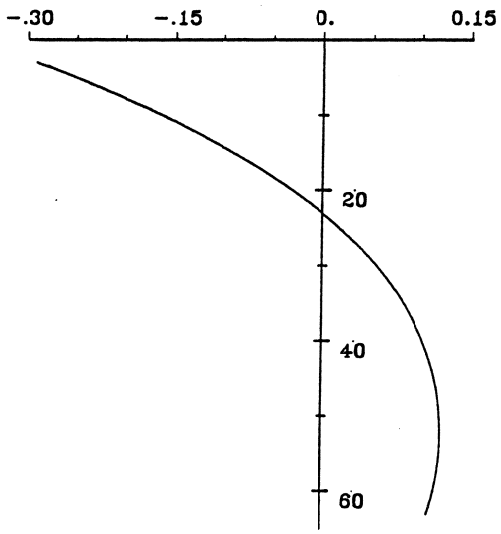

Figure 2b. The vertical V-profile

As mentioned earlier, the advective terms did not play an important role in this experiment. However, in the second experiment these terms play a crucial role. We examined a flow past a jetty [9]. A rectangular domain with a horizontal dimension of 1500 m by 300 m and a constant depth of 25 m was used. At the left open boundary, we prescribed an inflow condition of u=0.5 m/s and at the right boundary a uniform water level ζ=0 m was given. For a detailed description of the initial conditions and boundary conditions we refer to [9]. The horizontal mesh sizes were 25 m.
Near the boundaries, the discretization of the advective terms was chosen in a special way. Especially at inflow, the discretization of the advective terms may cause instabilities. To obtain a stable boundary treatment, we applied the discretization developed in [9]. At the open boundaries, a stabilizing effect is often experienced when Riemann invariants are prescribed. In our experiment, this would have resulted into the inflow condition

$$u + 2\sqrt{gh} = 0.5 + 2\sqrt{gh_o},$$

where h_o denotes the boundary value for h. Since the Riemann invariants (viz., $u \pm 2\sqrt{gh}$) are in general not known, we used the following variant [9]:

$$u + \varepsilon \frac{\partial}{\partial t}(u + 2\sqrt{gh}) = 0.5 , \qquad (5.1)$$

with ε some parameter. The time derivative of the Riemann invariant in (5.1) was discretized in a straightforward manner. For sufficiently small values of ε, the boundary condition (5.1) is only a small perturbation of the original inflow condition u=0.5 m/s. The time-derivative of the Riemann invariant in (5.1) was introduced to obtain a weakly reflective boundary condition for the short wave components. These components mainly originate from the initial values and the eigenfrequencies of the model. Without Riemann invariants, these components may disturb the solution for a long time, because there is, in general, little dissipation in the model.

In this experiment, we varied the number of grid layers in the vertical direction. At first, we only used one vertical grid layer. In this case, a comparison with the results in [9] was possible. Various flow patterns are shown in Figure 3. One clearly sees the development of eddies past the jetty. After three hours, the solution was almost stationary. The numerical results are in agreement with the results in [9].

The discretization of the advective terms, which was developed in [9], is very important in this experiment. Both the special discretization near the boundaries and the introduction of some dissipation by the upwind discretization of some advective terms were necessary in order to obtain stable results when a large number of time steps are performed. For example, a central discretization of the advective terms yielded instabilities.

We also computed three-dimensional velocity profiles (e.g., with ns=5). The results were again in agreement with the reference solution. For realistic time steps the Jacobi-type method (4.2') required less than ten iterations. We observed that the number of iterations hardly depends on the number of grid layers in the vertical direction and also hardly depends on the value of the vertical diffusion coefficient.

In the experiments, we used both the vector and the parallel optimization of the Alliant FX/4. For our integration method (3.1) the computation time was reduced by about a factor of three due to the vectorization, and by an additional factor of three due to the parallel optimization. This shows that this method can be implemented efficiently on vector and parallel computers. In [5] the computational efficiency of this method was demonstrated on the CRAY Y-MP4/464. On four processors we obtained a performance of more than 500 megaflops.

Acknowledgements. The author is grateful to J.G. Blom for providing the figures.

7. References

[1] A.M. Davies, Application of the Galerkin methods to the formulation of a three dimensional hydrodynamic nonlinear hydrodynamic sea model, *Appl. Math. Modelling*, **4**, 245-256 (1980).

[2] E.D. de Goede, A computational model for the three-dimensional shallow water flows on the ALLIANT FX/4, *Supercomputer*, **32**, 43-49 (1988).

[3] E.D. de Goede, A time splitting method for the three-dimensional shallow water equations, *Int. J. Numer. Methods Fluids*, to appear.

[4] E.D. de Goede, *Numerical methods for the 3D shallow water equations on vector and parallel computers*, Report NM-R91xx, CWI, Amsterdam (1991).

[5] E.D. de Goede, *3D shallow water model on the CRAY Y-MP4/464*, Report NM-R9106, CWI, Amsterdam (1991).

[6] R.W. Lardner and H.M. Cekirge, A new algorithm for three-dimensional tidal and storm surge computations, *Appl. Math. Modelling*, **12**, 471-481 (1988).

[7] J.J. Leendertse, Aspects of a computational model for long period water wave propagation, *Memorandum RM-5294-PR*, Rand Corp., Santa Monica, California, 1967.

[8] N.A. Philips, A coordinate system having some special advantages for numerical forecasting, *J. Meteorol.*, **14**, 184-194 (1957).

[9] G.S. Stelling, On the construction of computational methods for shallow water flow problems, *Ph.D. Thesis*, Delft University, 1983.

[10] TRISULA, A multi-dimensional flow and water-quality simulation system, Delft Hydraulics, The Netherlands, 1989.

[11] P. Wilders, Th.L.van Stijn, G.S. Stelling and G.A. Fokkema, A fully implicit splitting method for accurate tidal computations, *Int. J. Numer. Methods Eng.*, **26**, 2707-2721 (1988).

Appendix A. Notation

In this paper the following symbols are used:

C	Chezy coefficient
d	undisturbed depth of water
f	Coriolis term
g	acceleration due to gravity
h	total depth ($= d + \zeta$)
L, B	dimensions of the basin in x- and y-direction, respectively
λ	eddy viscosity coefficient in the horizontal direction
μ	eddy viscosity coefficient in the σ-direction
t	time
u,v	velocity components in x- and y-direction
u_d, v_d	velocity components at some depth near the bottom
w	vertical velocity component in the x-y-z co-ordinate system
W_f	wind stress
x,y	horizontal spatial co-ordinates
ρ	water density
σ	vertical spatial co-ordinate (for a definition see (2.5))
φ	angle between wind direction and the positive x-axis
ω	vertical velocity component in the x-y-σ co-ordinate system
ζ	elevation above undisturbed depth.

Appendix B. Finite differences

The operators used in this paper are of the following form:

$$\{ U\, U_x \}_{i,j,k} = U_{i,j,k} \frac{U_{i+1,j,k} - U_{i-1,j,k}}{2\Delta x}$$

$$\{ Z_{0x} \}_{i,j} = \frac{Z_{i,j} - Z_{i-1,j}}{\Delta x}$$

$$\{ (H \int_0^1 U d\sigma)_{0x} \}_{i,j} = (\tilde{H}_{i+1,j} \sum_{k=1}^{ns} \Delta\sigma_k U_{i+1,j,k} - \tilde{H}_{i,j} \sum_{k=1}^{ns} \Delta\sigma_k U_{i,j,k}) / \Delta x$$

$$\{ U\, V_{1x} \}_{i,j,k} = \begin{cases} \tilde{U}_{i,j,k} (3V_{i,j,k} - 4V_{i-1,j,k} + V_{i-2,j,k}) / (2\Delta x) & \text{if } \tilde{U}_{i,j,k} > 0 \\ \tilde{U}_{i,j,k} (-3V_{i,j,k} + 4V_{i+1,j,k} - V_{i+2,j,k}) / (2\Delta x) & \text{if } \tilde{U}_{i,j,k} \le 0 \end{cases}$$

$$\{ U\, V_x \}_{i,j,k} = \tilde{U}_{i,j,k} \frac{V_{i+1,j,k} - V_{i-1,j,k}}{2\Delta x}$$

$$\{ U_{xx} \}_{i,j,k} = \frac{U_{i+1,j,k} - 2U_{i,j,k} + U_{i-1,j,k}}{(\Delta x)^2},$$

where

$$\tilde{U}_{i,j,k} = 0.25 (U_{i,j,k} + U_{i+1,j,k} + U_{i,j+1,k} + U_{i+1,j+1,k})$$
$$\tilde{H}_{i,j} = 0.5 (H_{i,j} + H_{i-1,j}) .$$

The operators in the y-direction are defined similarly. For the discretizations in the vertical direction we define

$$\{ \Omega\, U_\sigma \}_{i,j,k} = \frac{1}{\Delta\sigma_k} \left\{ \tilde{\Omega}_{i,j,k+1} \frac{\Delta\sigma_k U_{i,j,k+1} + \Delta\sigma_{k+1} U_{i,j,k}}{\Delta\sigma_{k+1} + \Delta\sigma_k} - \tilde{\Omega}_{i,j,k} \frac{\Delta\sigma_{k-1} U_{i,j,k} + \Delta\sigma_k U_{i,j,k-1}}{\Delta\sigma_k + \Delta\sigma_{k-1}} \right\},$$

$$\{ U_{\sigma\sigma} \}_{i,j,k} = \frac{8}{(\Delta\sigma_{k-1} + 2\Delta\sigma_k + \Delta\sigma_{k+1})} \left\{ \frac{\mu_{k+1}(U_{i,j,k+1} - U_{i,j,k})}{\Delta\sigma_{k+1} + \Delta\sigma_k} - \frac{\mu_k (U_{i,j,k} - U_{i,j,k-1})}{\Delta\sigma_k + \Delta\sigma_{k-1}} \right\},$$

where

$$\tilde{\Omega}_{i,j} = 0.5 (\Omega_{i,j} + \Omega_{i-1,j})$$

and μ_k denotes the value of vertical diffusion coefficient at the kth grid layer. Similarly, we define the operators for the V-component.

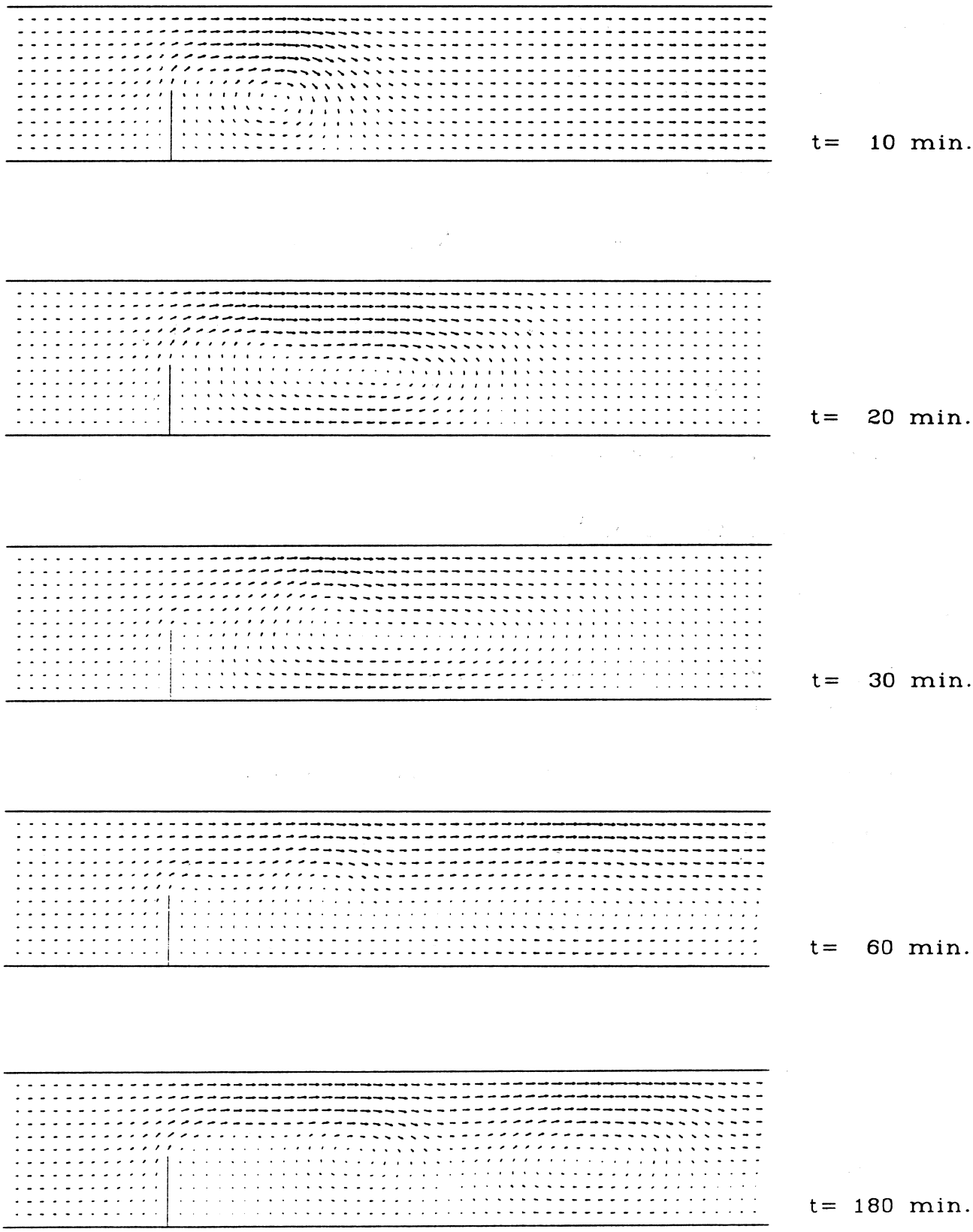

Figure 3. The flow patterns for the river problem with ns=1