

Computing non-hydrostatic free-surface flows with TRIWAQ using a pressure correction technique

Project:

NAUTILUS

Working document:

RIKZ/OS/2000.137X



In opdracht van:

Directie Noordzee
Directie Zuid-Holland
Directie Zeeland
Directie Noord-Holland
Gemeentelijk Havenbedrijf Rotterdam
Meetstrategie 2000+

Computing non-hydrostatic free-surface flows with TRIWAQ using a pressure correction technique

Project:	NAUTILUS
Working document:	RIKZ/OS/2000.137x
Date:	Oktober 2000
Author:	M. Zijlema

Contents

1 Introduction	4
2 Formulation and discretization of the water movement including hydrodynamic correction in TRIWAQ	8
2.1 Formulation of the equations for water movement	8
2.2 Discretization in space	11
2.2.1 Layer-averaging	11
2.2.2 Local continuity equation	16
2.2.3 Hydrodynamic pressure gradient in horizontal momentum equations	17
2.2.4 Momentum equation for physical vertical velocity	18
2.3 Time discretization and pressure correction method	19
2.3.1 Time discretization	20
2.3.2 Pressure correction method	21
3 Solution methods for the discretized pressure correction equation	25
3.1 A brief overview of the solution methods	25
3.2 The BiCGSTAB method	27
3.3 The GCR method	28
3.4 Choice of an iterative method	30
3.5 Preconditioning of Krylov subspace methods	30
3.5.1 Row scaling	32
3.5.2 Eisenstat implementation	33
3.6 Stopping criteria and choice of starting vector	33
4 Numerical experiments	35
4.1 Standing short wave in rectangular basin	35
4.2 Exchange flow of stratified fluid	40
4.3 Wave propagation over a bar	41
5 Summary, conclusions and recommendations	46
Literature	49

1 Introduction

Usually, the shallow water models like WAQUA and TRIWAQ are sufficiently accurate for large-scale or far-field flows in coastal seas, estuaries and rivers. Such flows are typically shallow, which means that the horizontal length scale is much larger than the vertical one. As a consequence, the large scale motions are predominantly horizontal, whereas the vertical acceleration is very small, particularly if compared with the gravity acceleration. Hence, the momentum equation for the vertical velocity component is reduced to the hydrostatic pressure law, whereas the *non-hydrostatic* or *hydrodynamic* pressure is neglected. This gives rise to a simplified numerical model, since only two horizontal momentum equations combined with the free-surface condition need to be solved. The outcome of this model consists of two horizontal components of the flow-velocity and the water level representing the hydrostatic pressure. The vertical velocity is computed by means of the equation of continuity.

However, the shallow water model that requires the hydrostatic pressure distribution, prohibits a correct calculation of

- short surface waves,
- flows over a rapidly varying bottom,
- vertical flow recirculations,
- flows around groynes, weirs and sluices,
- exchange flows, gravity and density currents and
- flows near local discharges (e.g. cooling water).

In these so-called near-field flows, the horizontal length scale is not large enough compared with the vertical one. So, we need extend our shallow water model with a hydrodynamic part of the pressure that enable us to predict the abovementioned situations more accurately. As a consequence, the momentum equation for the vertical velocity component has to be solved and also the (local) continuity equation. The flow equations of this extended model contain five unknowns, namely three velocity components, water level (hydrostatic pressure) and hydrodynamic pressure. The local continuity equation is derived from the incompressibility condition and does not contain the pressure. Such a property makes the solution of a non-hydrostatic flow model in general a difficult task, since the continuity equation is strongly related to the pressure computation.

For time-dependent problems the *pressure correction* method is one of the most popular methods to solve the problem concerning the absence of the pressure in the continuity equation. There is a vast amount of literature available on the subject of pressure correction techniques for solving incompressible Navier-Stokes equations. Here, we mention only those publications that bear directly on the type of flow computations that we envisage in this report. The method was originally applied by Harlow and Welch in the MAC (Marker-And-Cell) approach for the computation of free surface incompressible flows [19]. Similar techniques have been developed by Chorin [8] and Van Kan [23]. Pressure correction methods in combination with a time-marching method are very efficient for time-dependent problems, but they can also be applied to stationary cases. A well-known example is the very popular SIMPLE approach (see, for example, [27]). A method that resembles the

pressure correction technique is the *fractional step* method originally developed by Temam [34]. The pressure correction and fractional step techniques belong to the class of so-called *projection methods* [21].

Roughly speaking, the pressure correction approach can be formulated as a prediction step in which the velocity is estimated from the momentum equations using the pressure at the previous time level but without taking into account the continuity equation. Next, the velocity is projected onto the space of divergence-free vector fields, resulting in a Poisson-like equation for the pressure correction (the difference between the new and the old pressure) and the velocity is updated. Furthermore, the fractional step approach consists in removing the pressure term from the momentum equations of which the velocity field is determined. In fact, the fractional step scheme can be regarded as a splitting method. Next, the obtained velocity field, which is not divergence-free, is corrected by means of the solution of the Poisson equation for the pressure at the new time level.

Projection methods can be applied to either the continuous or the discrete Navier-Stokes equations. A disadvantage of the continuous approach is that it is necessary to define boundary conditions for the pressure. This may be difficult for some types of boundaries and is not natural since the Navier-Stokes equations do not require any pressure boundary conditions at all. This problem does not appear in the discrete approach, in which first the space discretization is applied and subsequent the boundary conditions for the momentum equations are incorporated and afterwards the projection method resulting in a discretized Poisson equation for the pressure that already includes its boundary conditions. The issue of boundary conditions for the pressure Poisson equation is discussed extensively in [17].

In contrast with the abovementioned publications, we consider the pressure as a sum of two distributions: the hydrostatic part and a hydrodynamic part that can be seen as the correction to the hydrostatic pressure. This motivates us to apply the projection method, in which the velocity is estimated from the momentum equations using only the hydrostatic pressure and thereafter the Poisson equation for the hydrodynamic part is solved. This idea is proposed in [25] and [7]. In [25], it is shown that this approach leads to a more stable and efficient hydrodynamic flow model than in a case without the splitting of the pressure into hydrostatic and hydrodynamic parts.

The hydrodynamic flow model of Casulli and Stelling [7] is based on the fractional step method in the discrete sense. The first step computes the water level and horizontal velocities based on the hydrostatic approximation, while the second step calculates a non-hydrostatic correction in order to make the velocity field divergence-free. The underlying motivation for this approach is that the existing shallow water solver need not to be adapted, since the correction to the hydrostatic pressure is done *after* the shallow water equations have been solved. As a consequence, this reduced the effort of software maintenance to a minimum. However, the difficulty with this procedure is that during the first step the water level will not be directly affected by the non-hydrostatic correction, because the hydrodynamic pressure is removed from the momentum equations. Also, the fractional step approach introduces a splitting error, since the advection and the pressure gradient do not commute. This means that it reduces the order of temporal accuracy of the scheme by one.

Afterwards, Casulli has made an attempt to improve this model by means of including an *ad hoc* correction for the water level. Due to this correction, as described in his article [6], the new water level is implicitly coupled with the

hydrodynamic pressure. However, in this way, the mass conservation may be lost since, during the hydrostatic step the water level has been determined by the free surface condition and after that no change to the water level should be allowed. Furthermore, the method, that is discretized by the so-called θ -method (a linear combination of the forward and backward Euler schemes), has been further improved by just neglecting the implicit contribution of the hydrodynamic pressure instead of both implicit and explicit parts. This means, for instance, if the Crank-Nicolson scheme is chosen ($\theta = 1/2$), the hydrodynamic pressure has effect on the water level during the hydrostatic step. However, it is still a fractional step method so the splitting error does not reduce, contrary to what is claimed in [6]. This is confirmed by a number of experiments as presented in Subsection 4.1. Moreover, the influence of the hydrodynamic pressure on water level depends on the value of θ . For example, in the fully-implicit case, i.e. $\theta = 1$, the water level is not affected by the hydrodynamic pressure during the first step.

In this report, another method is proposed. Our approach is based on the second order time-accurate pressure correction scheme as described by Van Kan [23]. Firstly, we determine the intermediate velocity field from the momentum equations containing both the hydrostatic and hydrodynamic pressure parts at the new and preceding time levels, respectively. During this step, the water level will be influenced by the hydrodynamic pressure. Secondly, the pressure correction of the hydrodynamic part is computed by means of solving the Poisson equation. Finally, the intermediate velocity is corrected. For the pressure there are no physical boundary conditions and the numerical method followed here will not need them either. However, a constraint is implemented in the discretized Poisson equation such that the continuity of pressure at the free surface is guaranteed. Furthermore, the considered approach can be easily combined with the θ -method. Hence, it is possible to compute the non-hydrostatic correction second order accurately in time. Also, the effect of the hydrodynamic pressure on the water level is independent of the choice of θ . A minor disadvantage of our method is that the kernel of the shallow water solver has to be slightly changed. Section 2 deals with the formulation and discretizations of our hydrodynamic flow model.

The numerical solution of the Poisson equation for the pressure correction is a crucial step of the whole approach, since the overall efficiency of the numerical code as TRIWAQ will depend on its performance. In Cartesian coordinates, fast Poisson solvers based on Fourier transformation or cyclic reduction are in widespread use. On general grids, however, these methods are not applicable. Hence, iterative solution methods with possibly acceleration techniques should be applied. Particularly, preconditioned Krylov subspace and multigrid approaches are strongly recommended for this step of the computation. In this report, we restrict ourselves to the first mentioned type of iterative solvers. Usually, the well-known conjugate gradient (CG) method [20] works fine, if and only if the discretization of the Poisson equation results in linear system of equations with a symmetric positive definite matrix. This is the case, for instance, when a Cartesian fixed layer model is employed (see, for example, [7]). Since, we are dealing with a sigma-transformation, only non-symmetric pressure systems are involved in the pressure correction approach. In our method, we use two well-known Krylov subspace methods, appropriate for solving non-symmetric systems, combined with preconditioners based on incomplete LU decompositions [26], namely BiCGSTAB [39] and GCR [12]. Details on these techniques will be outlined in Section 3.

Finally, Section 4 discusses some test problems which have been solved in order to assess the performance of our method, whereas Section 5 concludes the

report by summarizing the results of the research presented and gives suggestions for future developments.

The research presented in this report is preliminary to the work that will be carried out next year. The non-hydrostatic approach in hand will be employed in the ZEEDELTA model for the determination of the discharge coefficients needed for the application of the 3D barrier formulation to the Haringvliet sluices. Further details can be found in the last paragraph of Section 5. This report was written by M. Zijlema in cooperation with TRI-FLOW COMP, Inc., Dr. J.J. Leendertse, as a part of the Nautilus project “Zoutindringing Haringvliet” by order of the National Institute for Coastal and Marine Management (RIKZ) (order number 22002210, 15/08/2000).

2 Formulation and discretization of the water movement including hydrodynamic correction in TRIWAQ

This section is concerned with the formulation and discretization of the equations for the water movement in which a correction to the hydrostatic pressure is included. The formulation is given in Section 2.1, whereas space discretization is considered in Section 2.2. The basic approach for the accurate calculation of the pressure correction is the pressure correction technique as outlined in Section 2.3.

2.1 Formulation of the equations for water movement

The equations describing the three-dimensional velocity field in incompressible flow are the well-known Navier-Stokes equations and take the following form:

$$\frac{\partial \rho u_\alpha}{\partial t} + \frac{\partial \rho u_\alpha u_\beta}{\partial x_\beta} + \frac{\partial p}{\partial x_\alpha} - \frac{\partial \tau_{\alpha\beta}}{\partial x_\beta} = \rho f_\alpha \quad (2.1)$$

where ρ is the fluid density, u_α is the Cartesian velocity component, p is the pressure, f_α represents an external force per unit volume and $\tau_{\alpha\beta}$ is the deviatoric stress tensor which given by

$$\tau_{\alpha\beta} = \mu \left(\frac{\partial u_\alpha}{\partial x_\beta} + \frac{\partial u_\beta}{\partial x_\alpha} \right) \quad (2.2)$$

where μ is the dynamic viscosity of the fluid. Here, Cartesian tensor notation is employed and also the summation convention in which repeated Greek indices are to be summed over the three coordinate directions, i.e. $\alpha, \beta = 1, 2, 3$. Details can be found in [47].

Due to the assumption of incompressibility, the velocity field is divergence free, i.e.,

$$\frac{\partial u_\alpha}{\partial x_\alpha} = 0 \quad (2.3)$$

Furthermore, we have $D\rho/Dt = 0$, so that

$$\frac{\partial \rho u_\alpha}{\partial t} + \frac{\partial \rho u_\alpha u_\beta}{\partial x_\beta} \equiv \frac{D\rho u_\alpha}{Dt} = u_\alpha \frac{D\rho}{Dt} + \rho \frac{Du_\alpha}{Dt} = \rho \frac{Du_\alpha}{Dt} \quad (2.4)$$

and hence, equation (2.1) is rewritten as

$$\frac{Du_\alpha}{Dt} + \frac{1}{\rho} \frac{\partial p}{\partial x_\alpha} - \frac{1}{\rho} \frac{\partial \tau_{\alpha\beta}}{\partial x_\beta} = f_\alpha \quad (2.5)$$

with $D\bullet/Dt$ the material derivative as defined in (2.4). The external force consists of the gravity and the Coriolis force and is given by

$$f_\alpha = -\delta_{\alpha 3}g + 2\varepsilon_{\alpha\beta\gamma}u_\beta\Omega_\gamma \quad (2.6)$$

where $\delta_{\alpha\beta}$ is the Kronecker delta, $\varepsilon_{\alpha\beta\gamma}$ is the permutation symbol (see also [47]), g is the acceleration of gravity and $\Omega = (\Omega_1, \Omega_2, \Omega_3)$ is the earth rotation vector.

We now decompose the pressure p into the *hydrostatic* part and a *hydrodynamic* or non-hydrostatic part, as follows:

$$p = \underbrace{p_{\text{atm}} + g \int_z^\zeta \rho dz'}_{p_{\text{stat}}} + q \quad (2.7)$$

where p_{atm} is the atmospheric pressure, ζ is the water level and q is the hydrodynamic pressure, which can be considered as the deviation from the hydrostatic pressure p_{stat} . The hydrostatic pressure is determined from the momentum equation for the vertical velocity u_3 by neglecting the advective and the viscous terms and the Coriolis force:

$$\frac{\partial p_{\text{stat}}}{\partial z} = -\rho g \quad (2.8)$$

Usually, the flow is often turbulent and all effects of turbulence on the mean flow may be modelled by means of a so-called k- ε formulation. In expression (2.2), the eddy viscosity denoted as μ_t and computed with the k- ε model is added to the molecular one. Furthermore, it is expected that the variation of the density ρ in the stress term, given in (2.5), is negligible. Let $v_t = \mu_t/\rho_0$ with ρ_0 a constant reference density, substitute (2.2), (2.6) and (2.7) into (2.5) and define $(u_1, u_2, u_3) = (u, v, w)$, then the equations for the mean turbulent flow to be considered read:

$$\begin{cases} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \\ \frac{Du}{Dt} + \frac{1}{\rho} \frac{\partial p_{\text{stat}}}{\partial x} + \frac{1}{\rho} \frac{\partial q}{\partial x} - \nabla \cdot (v_t \nabla u) = 2(\Omega_3 v - \Omega_2 w) \\ \frac{Dv}{Dt} + \frac{1}{\rho} \frac{\partial p_{\text{stat}}}{\partial y} + \frac{1}{\rho} \frac{\partial q}{\partial y} - \nabla \cdot (v_t \nabla v) = 2(\Omega_1 w - \Omega_3 u) \\ \frac{Dw}{Dt} + \frac{1}{\rho} \frac{\partial q}{\partial z} - \nabla \cdot (v_t \nabla w) = 2(\Omega_2 u - \Omega_1 v) \end{cases} \quad (2.9)$$

Since, we are dealing with the free surface flows we need an extra equation for the determination of the water level ζ . In this context, we consider a physical domain that is bounded vertically by the free surface $z = \zeta(x, y, t)$ and the bottom $z = -d(x, y)$. See Figure 2.1. The function $H = \zeta + d$ is called the water depth. Integration of the continuity equation as given in (2.9) over the water depth gives

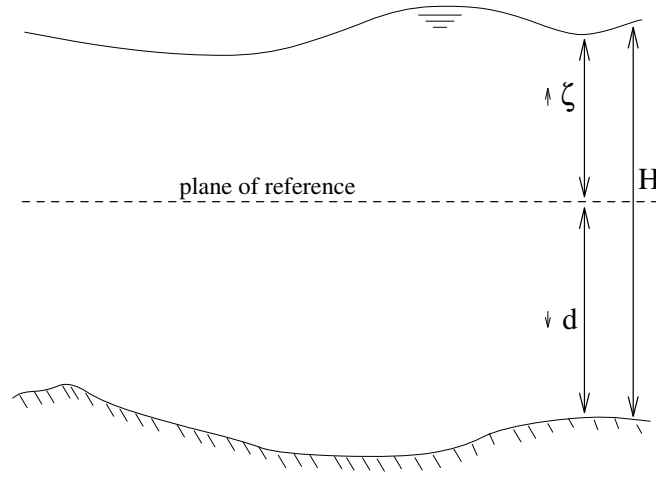


Figure 2.1: Water area with free surface and bottom.

$$\int_{-d}^{\zeta} \frac{\partial u}{\partial x} dz + \int_{-d}^{\zeta} \frac{\partial v}{\partial y} dz + w|_{\zeta} - w|_{-d} = 0 \quad (2.10)$$

By virtue of the Leibniz' rule and inserting the kinematic conditions, which are given by

$$\begin{cases} w|_{\zeta} = \frac{\partial \zeta}{\partial t} + u \frac{\partial \zeta}{\partial x} + v \frac{\partial \zeta}{\partial y} \\ w|_{-d} = -u \frac{\partial d}{\partial x} - v \frac{\partial d}{\partial y} \end{cases} \quad (2.11)$$

into (2.10), we obtain the so-called free surface condition:

$$\frac{\partial \zeta}{\partial t} + \frac{\partial}{\partial x} \int_{-d}^{\zeta} u dz + \frac{\partial}{\partial y} \int_{-d}^{\zeta} v dz = 0 \quad (2.12)$$

Note that (2.12) is sometimes called the *global* continuity equation, whereas the incompressibility constraint given in (2.9) is called the *local* continuity equation.

If the horizontal length scale is much larger than the vertical one, then the water is considered to be shallow. Introducing the dimensionless quantities for the unknowns and the length scales and scaling the continuity and momentum equations with these quantities, one can show that, for shallow water,

- the hydrodynamic pressure q is negligible small compared to the hydrostatic one p_{stat} and
- the vertical velocity w and the vertical acceleration terms in (2.9) can be neglected.

The result of this give rise to simplified flow equations. These so-called shallow water equations are the momentum equations for the horizontal velocities u and v as given in (2.9) with $q \equiv 0$, the hydrostatic pressure equation (2.8) and the free surface condition (2.12). Note that the expressions of the Coriolis force

in (2.9) are also simplified by means of neglecting its horizontal components, i.e. Ω_1 and Ω_2 . The vertical velocity w can be derived from the local continuity equation.

We conclude that the extension of the shallow water equations towards the Navier-Stokes solution for free surface flow consists of

- the hydrodynamic pressure gradient in the horizontal momentum equations, and
- the momentum equation for the vertical velocity.

The numerical solution of the shallow water equations is extensively studied in the Technical documentation [47]. The next subsection deals with the space discretization of the horizontal components of the hydrodynamic pressure gradient and the vertical momentum equation.

2.2 Discretization in space

For an appropriate formulation of the momentum equations to be discretized, we employ curvilinear coordinates in the longitudinal plane and a general prescription of layer interfaces in the transverse direction. However, we derive the momentum equation for the physical vertical velocity w instead of the relative vertical velocity ω that arise from the vertical grid schematization. The reason for this is to keep the formulation and thereby the discretization simple. Discretization of the governing equations is based on layer-averaging, i.e. finite volume discretization in the vertical direction, and finite differencing in the horizontal directions. This discretization is at least second order accurate. We have chosen for the staggered grid arrangement, because then spurious pressure oscillations are absent and boundary conditions for the momentum equations can be implemented easily without the use of interpolations. Further details can be found in the Technical documentation [47].

2.2.1 Layer-averaging

In the vertical direction the physical domain is divided into a fixed number of layers, see Figure 2.2. A flexible scheme is employed for the prescription of the layer interfaces z_k , which move with the water surface. This prescription can be made by defining the layer thickness in a relative way (a constant part of the total water depth, i.e. sigma-transformation) or in an absolute way (a constant layer thickness). For details on this prescription we refer to the Technical documentation [47]. Using this scheme, we shall derive the expression for the hydrodynamic pressure gradient occurring in the horizontal momentum equations and equation for the physical vertical velocity per layer.

Horizontal components of the hydrodynamic pressure gradient

We discuss the derivation of the layer-averaged hydrodynamic pressure term in the u -momentum equation. This derivation is based on the integration of

$$\frac{1}{\rho} \frac{\partial q}{\partial x} \tag{2.13}$$

over a layer k . By means of the Leibniz' rule and assuming that the layer-averaged density, defined as

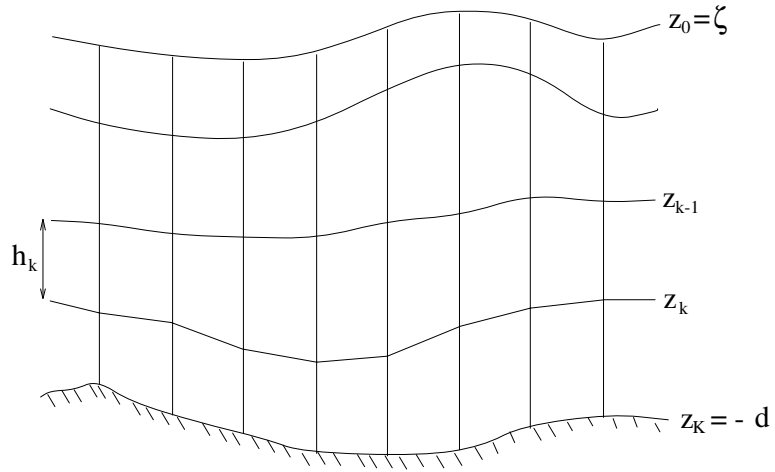


Figure 2.2: Vertical grid definition with layer interfaces.

$$\rho_k = \frac{1}{h_k} \int_{z_k}^{z_{k-1}} \rho dz' \quad (2.14)$$

is constant across layer k , we have

$$\begin{aligned} \int_{z_k}^{z_{k-1}} \frac{1}{\rho} \frac{\partial q}{\partial x} dz' &= \frac{1}{\rho_k} \int_{z_k}^{z_{k-1}} \frac{\partial q}{\partial x} dz' = \\ &= \frac{1}{\rho_k} \left[\frac{\partial h_k q_k}{\partial x} - q(z_{k-1}) \frac{\partial z_{k-1}}{\partial x} + q(z_k) \frac{\partial z_k}{\partial x} \right] \end{aligned} \quad (2.15)$$

with

$$q_k = \frac{1}{h_k} \int_{z_k}^{z_{k-1}} q dz' \quad (2.16)$$

In anticipating of writing the layer-averaged u -momentum equation in the non-conservative form, we write out (2.15) as follows:

$$\frac{\partial h_k q_k}{\partial x} - q(z_{k-1}) \frac{\partial z_{k-1}}{\partial x} + q(z_k) \frac{\partial z_k}{\partial x} = h_k \frac{\partial q_k}{\partial x} + (q(z_k) - q_k) \frac{\partial z_k}{\partial x} - (q(z_{k-1}) - q_k) \frac{\partial z_{k-1}}{\partial x} \quad (2.17)$$

The right-hand side of the layer-averaged u -momentum equation as given in the Technical documentation [47], formulae (2.44), is extended now with the following terms:

$$-\frac{1}{\rho_k} \frac{\partial q_k}{\partial x} - \frac{\partial z_k / \partial x}{\rho_k h_k} (q(z_k) - q_k) + \frac{\partial z_{k-1} / \partial x}{\rho_k h_k} (q(z_{k-1}) - q_k) \quad (2.18)$$

The layer-averaged hydrodynamic pressure term in the v -momentum equation can be derived in a similar way.

We now derive the layer-averaged w -momentum equation. The w -momentum equation is given by

$$\frac{\partial w}{\partial t} + \frac{\partial uw}{\partial x} + \frac{\partial vw}{\partial y} + \frac{\partial w^2}{\partial z} + \frac{1}{\rho} \frac{\partial q}{\partial z} - \nabla \cdot (v_t \nabla w) = 2(\Omega_2 u - \Omega_1 v) \quad (2.19)$$

For the application of layer-averaging of the w -momentum equation we consider Figure 2.3.

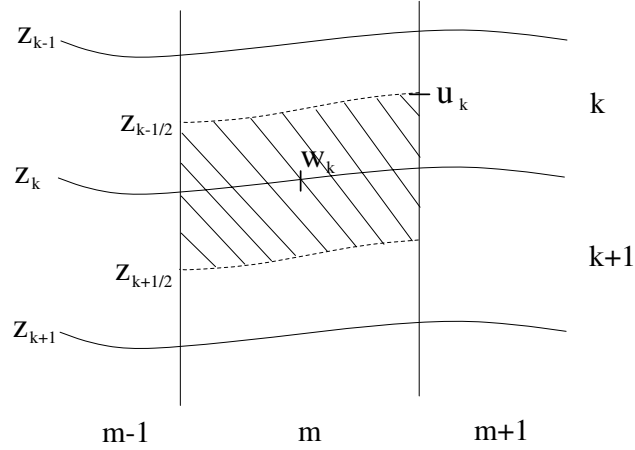


Figure 2.3: Control volume of w_k unknown.

Integration is carried out over a control volume with a w_k unknown located in its centre. We defined the layer-averaged vertical velocity as follows:

$$w_k = \frac{1}{\bar{h}_k} \int_{z_{k+1/2}}^{z_{k-1/2}} w dz' \quad (2.20)$$

with

$$\bar{h}_k = \frac{1}{2} (h_k^\zeta + h_{k+1}^\zeta) \quad (2.21)$$

with h_k^ζ giving the layer thickness of layer k at a water level point.

Using Leibniz' rule and (2.20), integration of the time derivative gives:

$$\int_{z_{k+1/2}}^{z_{k-1/2}} \frac{\partial w}{\partial t} dz' = \frac{\partial \bar{h}_k w_k}{\partial t} - w \frac{\partial z'}{\partial t} \Big|_{z_{k+1/2}}^{z_{k-1/2}} \quad (2.22)$$

and integration of the Coriolis terms gives:

$$2 \int_{z_{k+1/2}}^{z_{k-1/2}} (\Omega_2 u - \Omega_1 v) dz' = 2 \bar{h}_k (\Omega_2 \bar{u}_k - \Omega_1 \bar{v}_k) \quad (2.23)$$

where

$$\begin{aligned}\bar{u}_k &= \frac{1}{h_k} \int_{z_{k+1/2}}^{z_{k-1/2}} u dz' = \frac{1}{h_k} \left(\int_{z_{k+1/2}}^{z_k} u dz' + \int_{z_k}^{z_{k-1/2}} u dz' \right) = \\ &= \frac{1}{4h_k} \left(h_{m-1,k+1}^u u_{m-1,k+1} + h_{m,k+1}^u u_{m,k+1} + h_{m-1,k}^u u_{m-1,k} + h_{m,k}^u u_{m,k} \right)\end{aligned}\quad (2.24)$$

with m the index in x -direction (see Figure 2.3) and h_k^u the layer thickness of layer k at a u -point. The interpolation formulae (2.24) is second order accurate. The expression for \bar{v}_k can be derived in a similar way.

The integral forms of the advective and viscous terms are given by

$$\begin{aligned}\int_{z_{k+1/2}}^{z_{k-1/2}} \left(\frac{\partial u w}{\partial x} + \frac{\partial v w}{\partial y} + \frac{\partial w^2}{\partial z} \right) dz' = \\ \frac{\partial \bar{h}_k \bar{u}_k w_k}{\partial x} + \frac{\partial \bar{h}_k \bar{v}_k w_k}{\partial y} + w(z_{k-1/2}) \left(\omega_{k-1/2} + \frac{\partial z_{k-1/2}}{\partial t} \right) - w(z_{k+1/2}) \left(\omega_{k+1/2} + \frac{\partial z_{k+1/2}}{\partial t} \right)\end{aligned}\quad (2.25)$$

and

$$\begin{aligned}\int_{z_{k+1/2}}^{z_{k-1/2}} \nabla \cdot (v_t \nabla w) dz' &= \int_{z_{k+1/2}}^{z_{k-1/2}} \left(\frac{\partial}{\partial x} \left(v_t \frac{\partial w}{\partial x} \right) + \frac{\partial}{\partial y} \left(v_t \frac{\partial w}{\partial y} \right) + \frac{\partial}{\partial z} \left(v_t \frac{\partial w}{\partial z} \right) \right) dz' = \\ &= v_h \bar{h}_k \frac{\partial^2 w_k}{\partial x^2} + v_h \bar{h}_k \frac{\partial^2 w_k}{\partial y^2} + v_v \left. \frac{\partial w}{\partial z} \right|_{z_{k+1/2}}^{z_{k-1/2}}\end{aligned}\quad (2.26)$$

with

$$\omega_k = w(z_k) - \frac{\partial z_k}{\partial t} - u(z_k) \frac{\partial z_k}{\partial x} - v(z_k) \frac{\partial z_k}{\partial y} \quad (2.27)$$

the vertical velocity relative to layer interface z_k and v_h and v_v the horizontal and vertical eddy viscosities, respectively. For the derivation of (2.25) and (2.26), see [47].

Integration of the vertical hydrodynamic pressure gradient yields

$$\begin{aligned}\int_{z_{k+1/2}}^{z_{k-1/2}} \frac{1}{\rho} \frac{\partial q}{\partial z} dz' &= \int_{z_{k+1/2}}^{z_k} \frac{1}{\rho} \frac{\partial q}{\partial z} dz' + \int_{z_k}^{z_{k-1/2}} \frac{1}{\rho} \frac{\partial q}{\partial z} dz' = \\ &= \frac{1}{\rho_{k+1}} (q(z_k) - q_{k+1}) + \frac{1}{\rho_k} (q_k - q(z_k))\end{aligned}\quad (2.28)$$

Furthermore,

$$q(z_k) \approx \frac{q_{k+1} h_k^\zeta + q_k h_{k+1}^\zeta}{h_k^\zeta + h_{k+1}^\zeta} \quad (2.29)$$

So, we have

$$q(z_k) - q_{k+1} = \frac{h_{k+1}^\zeta (q_k - q_{k+1})}{h_k^\zeta + h_{k+1}^\zeta}, \quad q_k - q(z_k) = \frac{h_k^\zeta (q_k - q_{k+1})}{h_k^\zeta + h_{k+1}^\zeta} \quad (2.30)$$

The integral form of (2.28) then reads

$$\int_{z_{k+1/2}}^{z_{k-1/2}} \frac{1}{\rho} \frac{\partial q}{\partial z} dz' = \bar{\rho}_k^{-1} (q_k - q_{k+1}) \quad (2.31)$$

with

$$\bar{\rho}_k^{-1} \equiv \frac{\rho_k h_{k+1}^\zeta + \rho_{k+1} h_k^\zeta}{(h_k^\zeta + h_{k+1}^\zeta) \rho_k \rho_{k+1}} \quad (2.32)$$

This is recognized as the reciprocal of the "harmonic" average; see also [45]. This special averaging results in second order accuracy, regardless of the steepness of the sigma-layers.

The integration has now been completed. The resulting layer-averaged w-momentum equation reads:

$$\begin{aligned} \frac{\partial \bar{h}_k \bar{w}_k}{\partial t} + \frac{\partial \bar{h}_k \bar{u}_k \bar{w}_k}{\partial x} + \frac{\partial \bar{h}_k \bar{v}_k \bar{w}_k}{\partial y} + w(z_{k-1/2}) \omega_{k-1/2} - w(z_{k+1/2}) \omega_{k+1/2} = \\ - \bar{\rho}_k^{-1} (q_k - q_{k+1}) + \bar{v}_h \bar{h}_k \frac{\partial^2 \bar{w}_k}{\partial x^2} + \bar{v}_h \bar{h}_k \frac{\partial^2 \bar{w}_k}{\partial y^2} + \bar{v}_v \frac{\partial \bar{w}}{\partial z} \Big|_{z_{k+1/2}}^{z_{k-1/2}} + 2 \bar{h}_k (\Omega_2 \bar{u}_k - \Omega_1 \bar{v}_k) \end{aligned} \quad (2.33)$$

For the derivation of the non-conservative form of (2.33) in order to facilitate the approximations with finite differencing, we consider the layer-averaged continuity equation in control volume of w_k :

$$\frac{\partial \bar{h}_k}{\partial t} + \frac{\partial \bar{h}_k \bar{u}_k}{\partial x} + \frac{\partial \bar{h}_k \bar{v}_k}{\partial y} + \omega_{k-1/2} - \omega_{k+1/2} = 0 \quad (2.34)$$

Derivation of (2.34) can be found in [47]. Substitution of (2.34) in (2.33) and division by \bar{h}_k gives the layer-averaged w-momentum equation in the non-conservative form:

$$\begin{aligned}
& \frac{\partial w_k}{\partial t} + \bar{u}_k \frac{\partial w_k}{\partial x} + \bar{v}_k \frac{\partial w_k}{\partial y} + \frac{\omega_{k-1/2}}{h_k} (w(z_{k-1/2}) - w_k) - \frac{\omega_{k+1/2}}{h_k} (w(z_{k+1/2}) - w_k) = \\
& - \frac{\bar{\rho}_k}{h_k} (q_k - q_{k+1}) + v_h \frac{\partial^2 w_k}{\partial x^2} + v_h \frac{\partial^2 w_k}{\partial y^2} + v_v \frac{\partial w}{\partial z} \Big|_{z_{k+1/2}}^{z_{k-1/2}} \frac{1}{h_k} + 2(\Omega_2 \bar{u}_k - \Omega_1 \bar{v}_k)
\end{aligned} \tag{2.35}$$

2.2.2 Local continuity equation

Since, we are dealing with the physical vertical velocity w_k instead of the relative vertical velocity ω_k , the local layer-averaged continuity equation in discretized form is given by:

$$\begin{aligned}
& \left(\sqrt{g_{\eta\eta}} \right)_{m,n} h_{m,n,k}^u u_{m,n,k} - \left(\sqrt{g_{\eta\eta}} \right)_{m-1,n} h_{m-1,n,k}^u u_{m-1,n,k} \\
& - \left(\sqrt{g_{\eta\eta}} \right)_{m,n} (z_{m,n,k-1}^u - z_{m-1,n,k-1}^u) \bar{u}_{m,n,k-1}^{\xi z} + \left(\sqrt{g_{\eta\eta}} \right)_{m,n} (z_{m,n,k}^u - z_{m-1,n,k}^u) \bar{u}_{m,n,k}^{\xi z} + \\
& \left(\sqrt{g_{\xi\xi}} \right)_{m,n} h_{m,n,k}^v v_{m,n,k} - \left(\sqrt{g_{\xi\xi}} \right)_{m,n-1} h_{m,n-1,k}^v v_{m,n-1,k} \\
& - \left(\sqrt{g_{\xi\xi}} \right)_{m,n} (z_{m,n,k-1}^v - z_{m,n-1,k-1}^v) \bar{v}_{m,n,k-1}^{\eta z} + \left(\sqrt{g_{\xi\xi}} \right)_{m,n} (z_{m,n,k}^v - z_{m,n-1,k}^v) \bar{v}_{m,n,k}^{\eta z} + \\
& \sqrt{g_*} (w_{m,n,k-1} - w_{m,n,k}) = 0
\end{aligned} \tag{2.36}$$

where the averaged velocity components follow from (2.24). Furthermore,

$\left(\sqrt{g_{\xi\xi}} \right)_{m,n}$ and $\left(\sqrt{g_{\eta\eta}} \right)_{m,n}$ are geometric quantities calculated at points $(m, n+1/2)$ and $(m+1/2, n)$, respectively, and representing the lengths of gridcell located at (m, n) in ξ - and η -directions, respectively. Details on the curvilinear transformation and discretization of the local layer-averaged continuity equation can be found in [47]. At the surface level and the bottom the kinematic conditions for the vertical velocity w (2.11) are incorporated in (2.36), which yield equations for $k = 1$ and $k = K$, respectively:

$$\begin{aligned}
& \sqrt{g_*} \frac{\partial \zeta}{\partial t} + \left(\sqrt{g_{\eta\eta}} \right)_{m,n} h_{m,n,1}^u u_{m,n,1} - \left(\sqrt{g_{\eta\eta}} \right)_{m-1,n} h_{m-1,n,1}^u u_{m-1,n,1} \\
& + \left(\sqrt{g_{\eta\eta}} \right)_{m,n} (z_{m,n,1}^u - z_{m-1,n,1}^u) \bar{u}_{m,n,1}^{\xi z} + \\
& \left(\sqrt{g_{\xi\xi}} \right)_{m,n} h_{m,n,1}^v v_{m,n,1} - \left(\sqrt{g_{\xi\xi}} \right)_{m,n-1} h_{m,n-1,1}^v v_{m,n-1,1} \\
& + \left(\sqrt{g_{\xi\xi}} \right)_{m,n} (z_{m,n,1}^v - z_{m,n-1,1}^v) \bar{v}_{m,n,1}^{\eta z} - \sqrt{g_*} w_{m,n,1} = 0
\end{aligned} \tag{2.37}$$

and

$$\begin{aligned}
& \left(\sqrt{g_{\eta\eta}} \right)_{m,n} h_{m,n,K}^u u_{m,n,K} - \left(\sqrt{g_{\eta\eta}} \right)_{m-1,n} h_{m-1,n,K}^u u_{m-1,n,K} \\
& - \left(\sqrt{g_{\eta\eta}} \right)_{m,n} \left(z_{m,n,K-1}^u - z_{m-1,n,K-1}^u \right) \bar{u}_{m,n,K-1}^{-\xi z} \\
& \left(\sqrt{g_{\xi\xi}} \right)_{m,n} h_{m,n,K}^v v_{m,n,K} - \left(\sqrt{g_{\xi\xi}} \right)_{m,n-1} h_{m,n-1,K}^v v_{m,n-1,K} \\
& - \left(\sqrt{g_{\xi\xi}} \right)_{m,n} \left(z_{m,n,K-1}^v - z_{m,n-1,K-1}^v \right) \bar{v}_{m,n,K-1}^{-\eta z} + \sqrt{g_*} w_{m,n,K-1} = 0
\end{aligned} \tag{2.38}$$

2.2.3 Hydrodynamic pressure gradient in horizontal momentum equations

The first term of (2.18) is discretized at a u -point (m,n,k) as follows:

$$\frac{1}{\rho_k} \frac{\partial q_k}{\partial x} \approx \frac{1}{\bar{\rho}_{m,n,k}^{-\xi} \left(\sqrt{g_{\xi\xi}} \right)_{m,n}^{\xi\eta}} (q_{m+1,n,k} - q_{m,n,k}) \tag{2.39}$$

with

$$\bar{\rho}_{m,n,k}^{-\xi} = \frac{1}{2} (\rho_{m+1,n,k} + \rho_{m,n,k}) \tag{2.40}$$

Note that the transformation to curvilinear coordinates is already done. The second and third terms of (2.18) will be discretized in the similar way. Therefore, we shall only consider the second term. By means of Taylor expansions we have:

$$\begin{aligned}
q(z_{m,n,k}) \approx & \frac{h_{m,n,k+1}^u dx_{m+1,n} q_{m,n,k} + h_{m,n,k}^u dx_{m+1,n} q_{m,n,k+1} + h_{m,n,k+1}^u dx_{m,n} q_{m+1,n,k} + h_{m,n,k}^u dx_{m,n} q_{m+1,n,k+1}}{(dx_{m,n} + dx_{m+1,n})(h_{m,n,k}^u + h_{m,n,k+1}^u)}
\end{aligned} \tag{2.41}$$

and

$$\bar{q}_{m,n,k}^{-\xi} \approx \frac{dx_{m+1,n} q_{m,n,k} + dx_{m,n} q_{m+1,n,k}}{dx_{m,n} + dx_{m+1,n}} \tag{2.42}$$

where

$$dx_{m,n} = \frac{1}{2} \left(\left(\sqrt{g_{\xi\xi}} \right)_{m,n} + \left(\sqrt{g_{\xi\xi}} \right)_{m,n-1} \right) \tag{2.43}$$

Hence, the discretization of the second term of (2.18) reads:

$$\frac{\partial z_k / \partial x}{\rho_k h_k} (q(z_k) - q_k) \approx \frac{z_{m+1,n,k}^\zeta - z_{m,n,k}^\zeta}{\bar{\rho}_{m,n,k}^\xi \left(\sqrt{g_{\xi\xi}^\zeta} \right)_{m,n}^{\xi\eta}} \frac{dx_{m+1,n} (q_{m,n,k+1} - q_{m,n,k}) + dx_{m,n} (q_{m+1,n,k+1} - q_{m+1,n,k})}{(dx_{m,n} + dx_{m+1,n}) (h_{m,n,k}^u + h_{m,n,k+1}^u)} \quad (2.44)$$

In total, the discretization of (2.18) yields:

$$\begin{aligned} \tilde{G}_{0\xi}[q] = & \frac{1}{\bar{\rho}_{m,n,k}^\xi \left(\sqrt{g_{\xi\xi}^\zeta} \right)_{m,n}^{\xi\eta}} \left\{ q_{m+1,n,k} - q_{m,n,k} + \right. \\ & + (z_{m+1,n,k}^\zeta - z_{m,n,k}^\zeta) \frac{dx_{m+1,n} (q_{m,n,k+1} - q_{m,n,k}) + dx_{m,n} (q_{m+1,n,k+1} - q_{m+1,n,k})}{(dx_{m,n} + dx_{m+1,n}) (h_{m,n,k}^u + h_{m,n,k+1}^u)} + \\ & \left. - (z_{m+1,n,k-1}^\zeta - z_{m,n,k-1}^\zeta) \frac{dx_{m+1,n} (q_{m,n,k-1} - q_{m,n,k}) + dx_{m,n} (q_{m+1,n,k-1} - q_{m+1,n,k})}{(dx_{m,n} + dx_{m+1,n}) (h_{m,n,k-1}^u + h_{m,n,k}^u)} \right\} \quad (2.45) \end{aligned}$$

2.2.4 Momentum equation for physical vertical velocity

The discretization of the layer-averaged w -momentum equation (2.35) is quite straightforward. We discretize, term by term, the contributions to the system of linear equations. However, for the moment, we neglect the horizontal viscous terms.

The time derivative is approximated by

$$\frac{\partial w_{m,n,k}}{\partial t} \quad (2.46)$$

whereas the discretization of the Coriolis force terms is given by

$$2 \left(\bar{\Omega}_2^{\xi z} u_{m,n,k} - \bar{\Omega}_1^{\eta z} v_{m,n,k} \right) \quad (2.47)$$

with the interpolated velocities given by (2.24).

The horizontal advective terms are discretized by means of the second order κ -scheme, as follows:

$$S_{0\xi}^w[u, w] = \frac{\bar{u}_{m,n,k}^{\xi z}}{\left(\sqrt{g_{\xi\xi}^\zeta} \right)_{m,n}} \left(\bar{w}_{m,n,k} - \bar{w}_{m-1,n,k} \right) \quad (2.48)$$

with

$$\bar{w}_{m,n,k} = \begin{cases} w_{m,n,k} + \frac{1}{4} \left[(1+\kappa)(w_{m+1,n,k} - w_{m,n,k}) + (1-\kappa)(w_{m,n,k} - w_{m-1,n,k}) \right], \bar{u}_{m,n,k}^{\xi z} > 0 \\ w_{m,n,k} - \frac{1}{4} \left[(1+\kappa)(w_{m+1,n,k} - w_{m,n,k}) + (1-\kappa)(w_{m+2,n,k} - w_{m+1,n,k}) \right], \bar{u}_{m,n,k}^{\xi z} < 0 \end{cases} \quad (2.49)$$

where the parameter κ is still to be chosen. For all values of $\kappa \in [-1, 1]$, a blended form arises between second order linear upwind differencing and central differencing. The well-known schemes LUDS, QUICK and CUI are obtained by setting $\kappa = -1$, $\frac{1}{2}$ and $\frac{1}{3}$, respectively. The value $\kappa = 0$ gives the Fromm's scheme, while $\kappa = 1$ corresponds to central differencing. For $\kappa \neq \frac{1}{3}$ the local truncation error is of second order; for $\kappa = \frac{1}{3}$ it is of third order. For details, we refer to [21]. Based on several test cases presented in this report, it appears that central differencing did perform well without any problems. If, however, instabilities occur due to the central scheme then one can apply an upwind scheme, which is generally more stable, by simply setting a value of $\kappa \in [-1, 1]$. Similar derivation for $S_{0\eta}^w[v, w]$ can be obtained.

The discretization of the vertical advective term is given by

$$S_{0z}^w[h^\xi, \omega, w] = \frac{(\omega_{m,n,k-1} + \omega_{m,n,k})w_{m,n,k-1} + (\omega_{m,n,k+1} - \omega_{m,n,k-1})w_{m,n,k} - (\omega_{m,n,k} + \omega_{m,n,k+1})w_{m,n,k+1}}{2(h_{m,n,k}^\xi + h_{m,n,k+1}^\xi)} \quad (2.50)$$

and that of vertical diffusion:

$$S_{0zz}^w[v_v, h^\xi, w] = \frac{(\bar{v}_v)_{m,n,k} \frac{w_{m,n,k-1} - w_{m,n,k}}{h_{m,n,k}^\xi} - (\bar{v}_v)_{m,n,k+1} \frac{w_{m,n,k} - w_{m,n,k+1}}{h_{m,n,k+1}^\xi}}{\frac{1}{2}(h_{m,n,k}^\xi + h_{m,n,k+1}^\xi)} \quad (2.51)$$

where the vertical eddy viscosity needs to be interpolated. Note that the relative vertical velocity ω_k in (2.50) is computed with the formulae (2.27).

Finally, the discretization of the hydrodynamic pressure gradient is simply given by

$$\tilde{G}_{0z}^w[q] = \bar{\rho}_{m,n,k}^{-1} \frac{2(q_{m,n,k} - q_{m,n,k+1})}{h_{m,n,k}^\xi + h_{m,n,k+1}^\xi} \quad (2.52)$$

2.3 Time discretization and pressure correction method

The numerical solution of the system given by (2.9) is complicated by the lack of the time derivative for the pressure. The standard approach to do time-stepping is the pressure correction method, already introduced by Harlow and Welch [19], and is developed as a useful way of obtaining an efficient solution algorithm for unsteady incompressible flow. This method decouples the treatment of velocity and pressure terms, which yield a set of easier-to-solve

equations, namely a convection-diffusion equation for the velocity giving an intermediate solution that is not divergence free, and a Poisson equation for the pressure correction meant to make the velocity field divergence free. The latter equation is obtained by taking the divergence of the momentum equations and using the condition of a divergence free velocity field. Pressure correction methods in combination with a time-marching method are very efficient for time-dependent problems. This is the approach we follow. For time discretization of the hydrodynamic pressure we use a linear combination of the forward and backward Euler schemes, the so-called θ -method, and we have chosen the discrete approach of the pressure correction procedure as described by Van Kan [23], which is second order accurate in both space and time. The next two subsections deal with the time discretization and the pressure correction method, respectively.

2.3.1 Time discretization

In this subsection, a brief presentation of the time discretization of the equations for water movement is given. Details can be found in the Technical documentation [47]. The same notations of that reference are employed: the time step is denoted as τ and the variables without any prime as superscript are at (previous) time t , whereas the values of the variables with a single prime and a double prime are taken at the time levels $t+\tau/2$ and $t+\tau$, respectively.

Within the ADI-framework, in the first stage the free-surface condition and the horizontal layer-averaged momentum equations are discretized in time as follows:

$$\begin{aligned} \zeta'_{m,n} + \frac{\frac{1}{2}\tau}{(\sqrt{g_*})_{m,n}} \left(U'_{m,n} (\sqrt{g_{\eta\eta}})_{m,n} - U'_{m-1,n} (\sqrt{g_{\eta\eta}})_{m-1,n} \right) = \\ \zeta_{m,n} + \frac{\frac{1}{2}\tau}{(\sqrt{g_*})_{m,n}} \left(V_{m,n-1} (\sqrt{g_{\xi\xi}})_{m,n-1} - V_{m,n} (\sqrt{g_{\xi\xi}})_{m,n} \right) \end{aligned} \quad (2.53)$$

with $U_{m,n}$ and $V_{m,n}$ the depth-averaged horizontal velocities at (m,n) and $(\sqrt{g_*})_{m,n}$ the volume of gridcell at (m,n) ,

$$\begin{aligned} \frac{u'_{m,n,k} - u_{m,n,k}}{\frac{1}{2}\tau} + S_{0\xi}[u', u] + S_{0\eta}[v', u] + S_{0z}[h'', \omega, u] \\ + S_{0\eta}[u'v', \sqrt{g_{\xi\xi}}] - S_{0\xi}[v'^2, \sqrt{g_{\eta\eta}}] = \\ - G_{0\xi}[\zeta'] - PG_{0\xi}[h'', h', \rho] + f\bar{v}^{\xi\eta}_{m,n,k} \\ + S_{0\xi\xi}[u, v'] - S_{0\eta\eta}[v', u] + S_{0zz}[v_v, h'', u'] \\ - \theta\tilde{G}_{0\xi}[q'] - (1-\theta)\tilde{G}_{0\xi}[q] \end{aligned} \quad (2.54)$$

and

$$\begin{aligned}
& \frac{v'_{m,n,k} - v_{m,n,k}}{\frac{1}{2}\tau} + S_{+\xi}[u, v'] + S_{+\eta}[v, v'] + S_{0z}[h^v, \omega, v'] \\
& + S_{0\xi}[uv', \sqrt{g_{\eta\eta}}] - S_{0\eta}[u^2, \sqrt{g_{\xi\xi}}] = \\
& - G_{0\eta}[\zeta] - PG_{0\eta}[h^v, h^\xi, \rho] + f\bar{u}_{m,n,k}^{\xi\eta} \\
& + S_{0\eta\eta}[u, v] + S_{0\xi\xi}[v, u] + S_{0zz}[v_v, h^v, v'] \\
& - \theta\tilde{G}_{0\eta}[q'] - (1-\theta)\tilde{G}_{0\eta}[q]
\end{aligned} \tag{2.55}$$

For brevity, we have used compact expressions for the spatial derivatives, of which the meaning can be found in [47]. Note the inclusion of the discretized hydrodynamic pressure gradients in (2.54) and (2.55) when comparing with the equations (5.1) and (5.2) presented in [47]. It should be noted also that the vertical part of the Coriolis terms has been neglected. The second stage is similar to the first one, except that the role of ξ - and η -directions and u - and v -velocities is interchanged.

In (2.54) and (2.55), the hydrodynamic pressure gradient terms are discretized in time by means of the θ -method. Depending on the parameter θ , the discretization is explicit ($\theta = 0$) or implicit ($0 < \theta \leq 1$). By virtue of the stability requirements, we always take $\frac{1}{2} \leq \theta \leq 1$. For $\theta = \frac{1}{2}$ we have the second order Crank-Nicolson scheme and for $\theta = 1$ we obtain the first order backward Euler scheme.

Finally, the momentum equation for the vertical velocity component is discretized in time as follows:

$$\begin{aligned}
& \frac{w''_{m,n,k} - w_{m,n,k}}{\tau} + S_{0\xi}^w[u'', w] + S_{0\eta}^w[v'', w] \\
& + \theta S_{0z}^w[(h^\xi)'', \omega'', w''] + (1-\theta)S_{0z}^w[(h^\xi)', \omega'', w] = \\
& - \theta\tilde{G}_{0z}^w[q''] - (1-\theta)\tilde{G}_{0z}^w[q] \\
& + \theta S_{0zz}^w[v_v, (h^\xi)'', w''] + (1-\theta)S_{0zz}^w[v_v, h^\xi, w] \\
& + 2\left(\Omega_2^{\xi z}\bar{u}_{m,n,k} - \Omega_1^{\eta z}\bar{v}_{m,n,k}\right)
\end{aligned} \tag{2.56}$$

where the meaning of the space discretized terms can be found in Subsection 2.2.4. Note that the horizontal terms are discretized explicitly, whereas the vertical terms are integrated implicitly in order to ensure stability.

2.3.2 Pressure correction method

The pressure correction method defines the shift from time level n to level $n+1$ in two steps. In order to explain this method more clearly, the system of algebraic momentum equations (2.54), (2.55), at both ADI-stages, and (2.56) may be summarized as follows:

$$\mathbf{u}'' + A(\mathbf{u}'', \underline{\zeta}'') + \tau\theta G\mathbf{q}'' + \tau(1-\theta)G\mathbf{q} = \mathbf{r} \tag{2.57}$$

where $\underline{\zeta}$, \mathbf{u} and \mathbf{q} denote algebraic vectors containing the water level, velocity and hydrodynamic pressure unknowns and \mathbf{r} is known from previous time step and the boundary conditions. Furthermore, A is a linear algebraic operator

representing the space discretization of the terms occurring in the momentum equations and G is the discretized gradient operator. The discretized local continuity equation given by (2.36) may formally written as:

$$D\mathbf{u}'' = 0 \quad (2.58)$$

where D is the discretized divergence operator.

The essence of the pressure correction algorithm is that the system (2.57) is not solved as it stands, but first a prediction for the intermediate velocity field \mathbf{u}^* is computed from (2.57) with the hydrodynamic pressure at the previous time level:

$$\mathbf{u}^* + A(\mathbf{u}^*, \underline{\zeta}'') + \tau G\mathbf{q} = \mathbf{r} \quad (2.59)$$

This system (2.59) combined with (2.53) is a time discretized form of the shallow water equations and thus can be solved in the similar way as in TRIWAQ. This step is therefore characterized as the hydrostatic step. It should be noted, however, that the gradient of the hydrodynamic pressure need to be included in the existing shallow water solver. When the water level and the horizontal velocity are known, the vertical velocity is computed by means of solving the equation (2.56), which is a part of (2.59), instead of the local continuity equation (2.36). Note that this equation is solved once in a time step. Furthermore, at the end of the first ADI-stage we put $\omega^* = \omega$, whereas it is calculated by means of (2.27). The obtained velocity field is not divergence free and hence, the following step need to be carried out.

In the hydrodynamic step, a pressure correction $\delta\mathbf{q} = \mathbf{q}'' - \mathbf{q}$ is calculated. To find the correction equation, first (2.59) is subtracted from (2.57), neglecting the difference in the discretized spatial terms, $A(\mathbf{u}'', \underline{\zeta}'') - A(\mathbf{u}^*, \underline{\zeta}'')$:

$$\mathbf{u}'' - \mathbf{u}^* = -\tau\theta G\delta\mathbf{q} \quad (2.60)$$

Van Kan [23] has shown that neglecting the difference in the discretized spatial terms does not deteriorate the temporal accuracy, and $\theta = \frac{1}{2}$ gives second order accuracy. Application of the discrete divergence operator D to (2.60) and using (2.58) leads to

$$DG\delta\mathbf{q} = \frac{D\mathbf{u}^*}{\tau\theta} \quad (2.61)$$

which a Poisson equation for the pressure correction $\delta\mathbf{q}$, since DG is a discrete Laplacian. Note that we have implicitly assumed that $\theta \neq 0$. Due to the use of the sigma-transformation the matrix DG contains 25 non-zero diagonals and is non-symmetric. Efficient solution of (2.61) is a subject of the next section. Once \mathbf{q}'' is obtained, we can calculate \mathbf{u}'' by means of (2.60). This velocity field is divergence free.

Since, the matrix DG represents a discretization of the Laplacian operator we should expect to need boundary conditions for the unique solution of (2.61). However, in the formulation of the Navier-Stokes equations, only normal and tangential components of the velocities and/or stresses (wind, bottom friction) need to be described at the boundaries in order to get a unique solution. The pressure is not a thermodynamic variable, as there is no equation of state for an incompressible fluid (see [4]). It is an implicit variable which propagates at

infinite speed in order to keep the flow always and everywhere incompressible. Therefore, no explicit boundary conditions for the pressure are given. Fortunately, the boundary conditions for the momentum equations restrict in some sense the matrices D and G at the boundary, and in this way they implicitly define boundary conditions for the pressure correction equation. For an example, if the velocity is described at a boundary then this results in the matrix DG that may be interpreted as $\partial \delta \mathbf{q} / \partial \mathbf{n}$ given at that boundary (\mathbf{n} is the unit outward normal vector). If the stresses are prescribed, i.e. Neumann conditions for the velocity are given, then this may result in a Laplacian molecule with Dirichlet boundary condition. Hence, the operator DG works exclusively on pressure values in grid points in the interior of the domain. The issue of boundary conditions for the pressure Poisson equation, which does only arise with the continuous approach, is discussed extensively in [17]; see also [23].

However, there are two aspects that need to be elaborated. First, in orthogonal grids it is always possible to eliminate virtual pressure points using the boundary conditions for the momentum equations. However, this is not the case when the sigma-transformation is employed. For the implementation, we therefore adopt the following strategy:

- every virtual pressure that can be eliminated using the boundary conditions will be eliminated, and
- all other virtual pressure unknowns will be expressed in unknowns in the interior by means of second order linear extrapolation.

Second, while in some cases the pressure is well-defined up to an arbitrary additive constant by the Navier-Stokes equations, it is not so for free surface flows. In such flows, the pressure field is determined at a certain level. This follows from the continuity of pressure across the surface. In the air an atmospheric pressure is given. Hence, at the free surface the total pressure must equal the atmospheric pressure. We now show why this causes difficulties for free surface flow computations. Suppose we want to compute the surface elevation of a standing wave in a closed basin (like we do in Section 4.1). On closed boundaries we described zero velocities, both normal and tangential components, whereas at the free surface the normal velocity is given due to the kinematic conditions, and the tangential stress is zero (no wind). As a consequence, application of the pressure correction method leads to a discretized Poisson equation with only Neumann type conditions and thus, the hydrodynamic pressure is determined up to a constant. So, in order to be consistent with the continuity of pressure at the free surface, the following condition must be considered in the computation:

$$q|_{z=\zeta} = 0 \quad (2.62)$$

However, the problem is that no virtual pressure points are presented, so far as the Cartesian case is considered, thanks to the boundary conditions for the momentum equations. Note that the virtual points created by the sigma-transformation will be eliminated through extrapolation. Thus, condition (2.62) is by no means a boundary condition for the momentum equations nor the Poisson equation, since we have implicitly a Neumann condition for the pressure. It is merely a constraint in order to fix the obtained pressure field to a certain level. So, the question is how to incorporate this restriction in the discretized pressure correction equation (2.61) ?

At this moment, we adopt the following approach which seems to work very well: in the upper layer ($k = 1$) we replace the pressure correction equation (2.61) by an other (simpler) equation such that constraint (2.62) holds. Since the vertical velocity w is given at the free surface, there is no spatial variation of the pressure normal to the surface. Hence, the hydrodynamic pressure immediately below the surface equals to that above the surface. Assuming a constant value inside a layer, we take the following equation:

$$\delta q_{m,n,1} = 0 \quad (2.63)$$

Note that an extrapolation like

$$\delta q_{m,n,1} = \frac{\delta q_{m,n,2} h_{m,n,1}^\zeta}{2h_{m,n,1}^\zeta + h_{m,n,2}^\zeta} \quad (2.64)$$

instead of (2.63) will not work, since it is not consistent with $\partial \delta \mathbf{q} / \partial \mathbf{n} = 0$ at the free surface. This is also confirmed by several experiments done in the course of this project, in which (2.64) gives in general poorer results than (2.63).

Two important remarks need to be made here. Assuming a constant value inside a layer is only reasonable if sufficient number of layers is taken. Furthermore, our hydrodynamic model will not work if one layer is specified, because then the hydrodynamic pressure is identically zero due to (2.63). Indeed, there is no possibility to construct a two-dimensional hydrodynamic model without imposing an implicit relation for the hydrodynamic pressure such that (2.62) holds and also $\partial \mathbf{q} / \partial \mathbf{n} = 0$ at the free surface. Such a relation may be based on higher order splines. Also, it is obvious to define the location of the hydrodynamic pressure at the layer interfaces rather than at the cell centres. Details may be found in [32]. An alternative is the application of a Boussinesq-type model. This model is typical depth-averaged and accounts for some influence of the vertical acceleration. Many Boussinesq models incorporating assumptions on the horizontal velocity have been developed for the simulation of wave propagation. The degree of success of a Boussinesq model depends on the nature and accuracy of these assumptions. Ideally, the Boussinesq model should be applicable to a wide range of problems with different frequency-dispersion characteristics, but this is seldom the case. We refer to [10] for further details.

3 Solution methods for the discretized pressure correction equation

3.1 A brief overview of the solution methods

The discretization of the pressure correction equation (2.61) leads to a large sparse linear system of equations that has to be solved. The system of equations is written in matrix notation as $Ax = b$, where A and b are the discretization matrix and vector, respectively, as derived in Section 2. Basically, there are two approaches to solve that system of equations: *direct* and *iterative* solution methods. Direct methods, such as Gaussian elimination, are impractical for the pressure system, since, in general, A is large and sparse. The use of iterative methods, which generate a sequence of approximate solutions $\{x_k\}$, $k = 0, 1, 2, \dots$, is very attractive since, it essentially involve the matrix A only in the context of matrix-vector multiplication. As a consequence, they are relatively cheap in terms of storage and amount of work required in comparison to the direct methods. The evaluation of an iterative method focusses on how quickly the iteratives x_k converge.

In the literature, three classes of iterative methods are considered: (1) *basic* or *splitting*, (2) *multigrid* and (3) *Krylov subspace* methods ([21], [45], [16], [3] and references quoted there). In the first class of methods the matrix A is split as $A = M - N$, with M non-singular, and the iteratives x_k are found in the following process:

$$Mx_{k+1} = Nx_k + b \quad (3.1)$$

Usually, in matrix A the diagonal, strictly lower and strictly upper parts are distinguished and are denoted as D , L and U , respectively. So, $A = L + D + U$. By means of the choice of M as function of L , D and U a specific method is obtained. Two well-known examples of the splitting approach are the Jacobi iteration ($M = D$, $N = -(L+U)$) and the Gauss-Seidel method ($M = D + L$, $N = -U$). For instance, in TRIWAQ, the Jacobi process with the red-black ordering is implemented in order to obtain the solutions of both horizontal momentum and transport equations, whereas the horizontal part of the k - ϵ model equations is solved by means of a Gauss-Seidel variant [47].

Whether or not the iterates obtained by formula (3.1) converge to $x = A^{-1}b$ depends upon the eigenvalues of the so-called amplification matrix $S = M^{-1}N$. The method (3.1) converges if and only if every eigenvalue of S in modulus is less than unity. The Fourier analysis allows a simple estimation of eigenvalues of S , when periodic conditions are assumed (see, for example, [21] and [45]). From this analysis, applied to a simple Poisson equation on a rectangle domain with uniform mesh, it follows that both Jacobi and Gauss-Seidel iterations are slowly converging processes and that their convergence rates deteriorate as the mesh size becomes smaller. Furthermore, this deterioration is found to occur also with other kinds of boundary conditions. In the context of elliptic equations these observations are found to be true of all splitting methods. This can be explained by looking at the frequency distribution of the error and in the way a splitting method treats the different frequencies in the spectrum. Its slow convergence behaviour is due to the poor damping of the low-frequency parts

of the error (representing by long wavelengths); in other words: the asymptotic behaviour of the error is dominated by the eigenvalues of S close to one.

There are many ways to improve the performance of splitting methods. For instance, the rate of convergence of these methods can be improved with multigrid techniques. The underlying principle of these approaches is to approximate the low-frequency errors on coarser grids. The high-frequency components of the error are reduced with a few number of iterations with a splitting method on the fine grid. In fact, this method act as a *smoother* of the error. Theoretically, the multigrid approach is the most efficient of all known iterative methods. However, a disadvantage of the multigrid method is that it is not a black-box strategy; the user should have some knowledge upon the choice of the smoothers and their parameters in order to enhance the overall performance of the multigrid calculation for a certain class of problems. Details on the multigrid technique can be found in [45].

Two other techniques for improving convergence make use of an extrapolation in the direction of the expected change of the updates and of Chebyshev polynomials, respectively. The first one leads to the so-called over-relaxation method. For instance, the successive over-relaxation (SOR) technique is obtained by modifying the Gauss-Seidel iteration as follows: $M = D + \omega L$ and $N = -U - (1-\omega)L$, where ω is the over-relaxation parameter. Its value should be such that it minimizes the in modulus largest eigenvalue of S . However, this value is known for a few special cases such as the Poisson equation on a rectangle domain. A complete survey of SOR theory can be found in [46]. The second approach to accelerate the convergence of an iterative method is the Chebyshev method, in which one approximates the solution x by a vector $y_k \in \text{span}\{x_0, x_1, \dots, x_k\}$ such that $\|y_k - x\|_2$ is minimized in a certain way. The main drawback is that information on the eigenvalues of $I - S$ should be known. More details on this approach, we refer to [36].

A difficulty associated with the over-relaxation and Chebyshev methods is that they depend upon parameters that are sometimes hard to choose properly. An approach without this difficulty is the so-called Conjugate Gradient (CG) method, originally developed by Hestenes and Stiefel [20]. This technique is the prototype of the Krylov subspace methods. In these methods we choose an initial guess x_0 of the system $Ax = b$, and we iteratively compute

$$z_k = x_k - x_0 \in K^k(A, r_0) \equiv \text{span}\{r_0, Ar_0, A^2r_0, \dots, A^{k-1}r_0\} \quad (3.2)$$

with $r_0 = b - Ax_0$, such that x_k is the “best” approximation to x . This means that in each step we minimize the error in some norm. These methods differ in the way the Krylov subspace $K^k(A, r_0)$ is formed and in the way that z_k is computed.

As mentioned before, the first Krylov subspace method is the CG technique, which compute z_k such that the error $x - x_k$ is minimized over the Krylov subspace in the norm that is induced by the inner product $(x, y)_A \equiv x^T A y$, if A is assumed to be symmetric and positive definite. Due to the symmetry, the iteration vectors can be computed by simple three term recursions, so that memory space is needed only for few vectors. This also leads to only two inner products per iteration. Hence, from a computational point of view CG is a very efficient algorithm.

Due to the sigma-transformation, the resulting discretization matrix of the pressure correction equation is not symmetric. For non-symmetric systems a procedure as efficient as CG is not possible. As was proved in [13], for these

systems no Krylov subspace method can satisfy an *optimality* requirement, i.e. the error is minimal measured in a certain norm, and have *short recurrences*, so that each iteration is cheap. If we wish to compute an optimal approximation, then the error or the residual must explicitly minimize over the Krylov subspace in every iteration, resulting in long recurrences. Because of this, a set of basis vectors increasing in each iteration needs to be stored. Also, the amount of work needed for the orthogonalization of the basis vectors grow for increasing number of iterations. This leads to methods that are generally very expensive both in CPU-time and in memory requirements. On the other hand, due to the optimality property, these methods are very robust. The well-known examples of this type of methods are GMRES [29] and GCR [12]. Another approach is to use methods that are based on three term recursions. Although, these cheap methods do not compute an optimal approximation over the whole Krylov subspace, they converge well. The basic method of this class is BiCG [14]. From this method many improvements have been developed: CGS [31], BiCGSTAB and its variants [39], [30], and QMR [15]. For a rather extensive overview of Krylov subspace methods and their properties, we refer to [3].

The TRIWAQ code, in which the pressure correction method is implemented, uses Krylov subspace methods to solve the pressure system: both BiCGSTAB and GCR are employed, so that the user has the flexibility to obtain the non-hydrostatic solution in an efficient and robust manner. The following two subsections outline these methods and their properties. In Subsection 3.4 we give some guidelines for choosing an appropriate solution method for the pressure system.

3.2 The BiCGSTAB method

The BiCGSTAB method is an example of the bi-conjugate gradient approach in which we construct a suitable set of basis vectors for the Krylov subspace by a three term recurrence relation as in the CG method. The basic approach is BiCG [14]. In every iteration, this method requires, apart from a multiplication with the matrix A , an extra multiplication with the transpose of A . Hence, the computational cost is twice as high. Sonneveld was the first to consider whether the extra computations needed for bi-conjugate gradients could not be utilized in a more useful manner. This idea has led to CGS [31]. The rate of convergence of this approach is found to be two times as fast as BiCG. However, the CGS method often has a very irregular convergence behaviour, which may spoil the accuracy of the solution completely [40]. Later, Van der Vorst has proposed a more robust algorithm based on the same idea: BiCGSTAB [39]. This method is implemented. An advantage of this approach is that only a limited amount of memory is required, and that no parameters have to be chosen. A drawback is that it lacks an optimality property. As a consequence, no convergence can be proved. The BiCGSTAB algorithm is presented below.

```

• Select  $x_0, \text{tol}$ 
 $r_0 = b - Ax_0$ 
Choose  $\tilde{r}_0$  such that  $(r_0, \tilde{r}_0) \neq 0$ , e.g.  $\tilde{r}_0 = r_0$ 
 $v_0 = p_0 = 0, \rho_0 = \alpha = \omega_0 = 1$ 
 $k = 0$ 
• while  $\|r_k\|_2 > \text{tol}$  do
     $k = k + 1$ 
     $\rho_k = (\tilde{r}_0, r_{k-1})$ 
     $\beta = (\rho_k / \rho_{k-1})(\alpha / \omega_{k-1})$ 
     $p_k = r_{k-1} + \beta(p_{k-1} - \omega_{k-1}v_{k-1})$ 
     $v_k = Ap_k$ 
     $\alpha = \rho_k / (\tilde{r}_0, v_k)$ 
     $s = r_{k-1} - \alpha v_k$ 
     $t = As$ 
     $\omega_k = (t, s) / (t, t)$ 
     $x_k = x_{k-1} + \alpha p_k + \omega_k s$ 
     $r_k = s - \omega_k t$ 

```

Algorithm 3.1: The BiCGSTAB algorithm.

3.3 The GCR method

In contrast to the bi-conjugate approaches, the minimum residual methods have certain optimality properties, so that the residual is explicitly minimized over the Krylov subspace in every iteration and hence, convergence is assured. Two well-known examples of this kind are the GMRES [29] and GCR [12] algorithms. These methods are mathematically equivalent and also their rate of convergences are comparable. Often, they show a superlinear convergence behaviour [41]. However, they have long recurrences implying the amount of work and storage requirement grow with the number of iterations. In order to restrict the required work and memory, one can use *restarted* or *truncated* versions of these methods. This will be outlined later. In TRIWAQ, only GCR is implemented. The reason for this choice becomes clear at the end of this subsection. For the moment, the GCR algorithm is given below.

```

• Select  $x_0, \text{tol}$ 
 $r_0 = b - Ax_0$ 
 $k = 0$ 
• while  $\|r_k\|_2 > \text{tol}$  do
     $k = k + 1$ 
     $s_k = r_{k-1}$ 
     $v_k = As_k$ 
    for  $i = 1, \dots, k-1$  do    # modified Gram-Schmidt
         $\alpha_i = (v_k, v_i)$ 
         $v_k = v_k - \alpha_i v_i$ 
         $s_k = s_k - \alpha_i s_i$ 
     $s_k = s_k / \|v_k\|_2$ 
     $v_k = v_k / \|v_k\|_2$ 
     $x_k = x_{k-1} + (v_k, r_{k-1}) s_k$ 
     $r_k = r_{k-1} - (v_k, r_{k-1}) v_k$ 

```

Algorithm 3.2: The GCR algorithm.

The GCR method has two drawbacks that are remedied in the GMRES algorithm. The first drawback is the fact that we have to store both the search vectors s_k and their image v_k . The second drawback is that GCR may break down if the residual r_{k-1} is orthogonal to the new basis vector v_k , since we cannot update the residual in that case. On the other hand, GCR has three nice properties that alleviate the main drawbacks of GMRES. The first one is that the approximate solution is always accessible, and thus giving the user the opportunity to employ a stopping criterion that is based on the solution itself instead of the residual. Furthermore, GCR has the benefit of allowing the use of a different preconditioner in each iteration step, which can improve the performance (see Subsection 3.5 on the application of preconditioning of iterative solvers). Finally, the third advantage of GCR has to do with the possibly combination with truncation strategies, which are explained below.

As we have already noted at the start of this subsection, work and memory requirements increase with respect to the number of iterations. This is due to the modified Gram-Schmidt process¹. In practice, one cannot afford to run the full GCR algorithm, and it becomes necessary to use restarts or to truncate vector recursions. In case of restarting the GCR algorithm, one stops GCR after m iterations and use x_m as a starting vector for a following application of GCR. At most $2m$ iteration vectors have to be saved. However, restarting destroys a nice property of full GCR, namely, the optimality property and thereby the superlinear convergence behaviour is lost. Another approach is to employ a truncation strategy. There are many different ways to truncate GCR (see, for example, [44]). In this report, the strategy of Jackson and Robinson [22] is used since, it is robust. The idea is as follows: choose the number ($ntrunc$) of search directions (s_k) that may be kept in memory. If the number of iteration becomes larger than $ntrunc$, the search direction s_j with the smallest absolute value of α_j

¹ In contrast to the classical Gram-Schmidt method, the modified one is stable with respect to rounding errors.

and its image v_i are overwritten by the new search direction s_{k+1} and v_{k+1} . The motivation is that s_i has only a limited influence on the convergence of GCR. Generally, truncated methods have a better convergence behaviour than restarted methods, particularly if superlinear convergence plays an important role [44]. It should be noted, however, that GMRES can only be restarted. Hence, if restarting or truncation is necessary (as is almost the case), truncated GCR is better than restarted GMRES. For efficiency purposes, this is the main reason why we have chosen the implementation of GCR instead of GMRES.

Another attempts to shorten the recursions in the GMRES approach are the so-called GMRESR algorithm, proposed by Van der Vorst and Vuik [42] and FGMRES developed by Saad [28].

3.4 Choice of an iterative method

To obtain the solution of the pressure system, it is not easy to decide which iterative method should be employed. Both BiCGSTAB and GCR methods have their own good and weak properties in terms of efficiency and robustness. Generally, BiCGSTAB is easy to use since, no parameters have to be chosen, and also reasonably fast for large classes of problems. Hence, for efficiency purposes, the user may choose that solver. This is the default in TRIWAQ. However, if break down of bad convergence occur, the most robust choice is the GCR method without truncation. This means that the parameter *ntrunc* should be taken as large as possible. Nevertheless, in many cases it is advised to choose *ntrunc* small, so that work and storage are limited, but not too small since then, GCR may not converge. Fortunately, a thorough sensitivity analysis has shown that GCR is not very sensitive to variations in the value of *ntrunc*. So, the default value of *ntrunc* is 5, which is reasonable for a wide range of problems. Finally, another way to circumvent non-convergence and to save a considerable amount of CPU-time is to enlarge the maximal number of iterations and to lower the required accuracy, respectively. Subsection 3.6 is concerned with stopping criteria, which make the solver to stop at a certain accuracy. The next subsection deals with a technique known as preconditioning to improve the performance of a Krylov subspace method.

3.5 Preconditioning of Krylov subspace methods

It appears that Krylov subspace methods are only fast converging when they are combined with a preconditioner [26]. In this subsection, a description is given of the preconditioners that are implemented in TRIWAQ. They are based on incomplete LU decompositions and are generally very robust and well suited for pressure correction computations. For an overview of preconditioners, we refer to [3].

Preconditioning means that the system $Ax = b$ is multiplied by the inverse of a well chosen matrix, the so-called preconditioner P . The choice of the preconditioner is such that the preconditioned matrix “looks more like” the identity matrix, but at the same time the evaluation of $P^{-1}b$ should be cheap. The reason for preconditioning is that the preconditioned matrix $P^{-1}A$ has a more favourable spectrum, i.e. the eigenvalues are more clustered around unity. Hence, this speed up the convergence of iterative solvers.

In this report, an incomplete $LD^{-1}U$ decomposition of A is used as preconditioner, so that the iterative method is applied to

$$U^{-1}DL^{-1}Ax = U^{-1}DL^{-1}b \quad (3.3)$$

instead of to $Ax = b$. This preconditioner is denoted by ILUD (Incomplete LU-factorization restricted to Diagonal) and is defined by the following rules [37]:

- $\text{diag}(L) = \text{diag}(U) = D$,
- the off-diagonal parts of L and U are equal to the corresponding parts of A ,
- $\text{diag}(LD^{-1}U) = \text{diag}(A)$.

Note that for this preconditioner only the diagonal D has to be kept in memory, whereas the other elements of L and U are taken from the original matrix A . If the last rule is replaced by

$$\bullet \quad \text{rowsom}(LD^{-1}U) = \text{rowsum}(A) \quad (3.4)$$

the so-called Modified ILUD preconditioner of [18] is obtained. This MILUD preconditioner is particularly fast if the hydrodynamic pressure is slowly space varying. In TRIWAQ, we use a convex combination of ILUD and MILUD resulting in RILUD(α) preconditioner (Relax Incomplete LU restricted to Diagonal) [2]:

$$d_i^{\text{RILUD}} = (1 - \alpha)d_i^{\text{ILUD}} + \alpha d_i^{\text{MILUD}} \quad (3.5)$$

where d_i^x is the i -th diagonal element of D computed by the preconditioner $X \in \{\text{ILUD}, \text{MILUD}, \text{RILUD}\}$ and $0 \leq \alpha \leq 1$ is a given parameter. In practice, one chooses α close to 1.0 to optimize convergence rate of the preconditioned iterative solver [2]. In our experiments with short wave over a nearshore bar (for the model description, see Subsection 4.3) we found an optimal value of $\alpha = 0.965$. The measurement of this parameter is based on the averaged reduction factor, defined as:

$$\mu_{\text{nit}} = \left(\frac{\|r_{\text{nit}}\|_2}{\|r_0\|_2} \right)^{\frac{1}{\text{nit}}} \quad (3.6)$$

with nit the number of iterations needed to obtain an accurate solution (see Subsection 3.6 on the determination of the accuracy of the solution). The lower this factor, the higher the convergence rate of the solver. Figure 3.1 shows the effect of α on μ_{nit} of the BiCGSTAB and untruncated GCR algorithms. Note that the BiCGSTAB method performs better than the GCR method. Based on our experiences, the optimal value of $\alpha = 0.965$ leads to a very good rate of convergence on a wide range of problems. Thus, the user does not have to worry about this parameter, although it can be changed in the TRIWAQ-input.

In the rest of this section we discuss the further optimization of RILUD(α) with respect to amount of work. This optimization can be achieved by means of a row scaling and of the use of Eisenstat implementation.

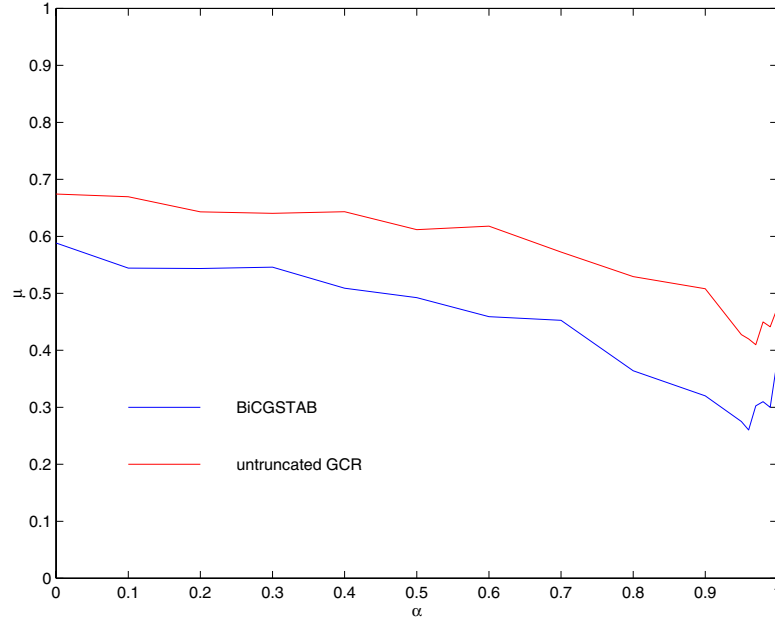


Figure 3.1: Effect of α using the RILUD(α) preconditioner on the averaged reduction factor of BiCGSTAB and untruncated GCR for the experiment with wave over a bar.

3.5.1 Row scaling

In this paragraph it is shown that a row scaling of the preconditioned system leads to less work per iteration [38]. For the RILUD factorization there exists a matrix R such that

$$A = LD^{-1}U - R \quad (3.7)$$

Multiplication by D^{-1} leads to

$$\tilde{A} = D^{-1}A = D^{-1}LD^{-1}U - D^{-1}R = \tilde{L}\tilde{U} - \tilde{R} \quad (3.8)$$

The matrices \tilde{A} , \tilde{L} and \tilde{U} have the following properties:

- $\text{diag}(\tilde{L}) = \text{diag}(\tilde{U}) = I$,
- the off-diagonal parts of \tilde{L} and \tilde{U} are equal to the corresponding parts of \tilde{A} ,
- $\text{diag}(\tilde{L}\tilde{U}) = \text{diag}(\tilde{A})$.

With $\tilde{b} = D^{-1}b$ we apply the iterative method to

$$\tilde{U}^{-1}\tilde{L}^{-1}\tilde{A}x = \tilde{U}^{-1}\tilde{L}^{-1}\tilde{b} \quad (3.9)$$

In this way, the multiplication by D in every iteration is no longer necessary, and hence saving CPU-time. Furthermore, the solution of the triangular systems is cheaper, since the main diagonals of \tilde{L} and \tilde{U} are equal to the identity matrix.

A nice property of this row scaling by D^{-1} is that if L , D and U satisfy the MILUD rule (3.4) then

$$\text{rowsum}(D^{-1}A) = \text{rowsum}(D^{-1}L D^{-1}U)$$

because $\text{rowsum}(R) = 0$ implies $\text{rowsum}(D^{-1}R) = 0$, so that \tilde{L} and \tilde{U} also satisfy the MILUD rule. In the remainder of this section the row-scaled quantities are denoted by skipping the tilde \sim .

3.5.2 Eisenstat implementation

In every iteration step we have to calculate the matrix-vector product $w = U^{-1}L^{-1}Av$. So, the amount of work per iteration is approximately two times as much as for the unpreconditioned system. In [11], it is shown that much of the extra work can be avoided. To achieve this it is necessary to apply the iterative method to

$$L^{-1}AU^{-1}y = L^{-1}b \quad (3.10)$$

where the solution vector x is given by $x = U^{-1}y$. Since, the spectrum of $U^{-1}L^{-1}A$ is equal to the spectrum of $L^{-1}AU^{-1}$ we expect the same convergence rate if we use (3.10) instead of (3.3). During the iterative solution of (3.10) we have to compute $w = L^{-1}AU^{-1}v$. Using the following relations:

$$\begin{aligned} w &= L^{-1}AU^{-1}v = L^{-1}(L + A - L - U + U)U^{-1}v \\ &= U^{-1}v + L^{-1}(v + [\text{diag}(A) - 2I]U^{-1}v) \end{aligned} \quad (3.11)$$

the work to calculate the matrix-vector product is reduced to two vector updates and the solution of an upper and lower triangular system. Hence, one iteration of the preconditioned system costs approximately the same amount of flops as the unpreconditioned system.

3.6 Stopping criteria and choice of starting vector

Because the solution of the pressure correction equation is the most time-consuming part, it is important to have a good stopping criterion. This means that the accuracy of the final iterate is sufficient, whereas the number of required iterations is as small as possible. Generally, the iterative solution method should be stopped if the approximate solution is accurate enough. However, the accuracy criterion should not be too weak since then, the obtained solution may be useless. On the other hand, if the criterion is too severe the solver may never stop or costs too much work. Finally, the choice of a starting vector is also important, because it can influence the number of iterations.

It is common to use the reduction of the residual as a stopping criterion, because Krylov subspace methods require calculation of the residual. When solving the system $Ax = b$, after k iterations we have an approximate solution x_k and the residual $r_k = b - Ax_k$ is related to the convergence error $\epsilon_k = x - x_k$ by $A\epsilon_k = r_k$, so the reduction of the residual results in the reduction of the convergence error.

An iteration process stops at each time step if the ratio of the norm of the final and initial residual is less than a given accuracy: $\|r_k\|_2 / \|r_0\|_2 < \epsilon$. However, in this case, the number of iterations is independent of the initial estimate x_0 . This may

be a drawback, because a good initial vector does not lead to a decrease of the number of iterations. This can be circumvented by taking the following criterion: $\|r_k\|_2/\|b\|_2 < \varepsilon$, so that if the iteration process is started with an accurate estimate the number of iterations is less than using an inaccurate start vector [43].

In the pressure system (2.61), the solution vector consists of the difference of the hydrodynamic pressure q at two consecutive time levels n and $n+1$: $\delta q = q^{n+1} - q^n$. Generally, q is a slowly varying function of time, so that $\|\Delta q\|_2 = O(\Delta t)$, which is relatively small. This motivates us to use the starting vector $x_0 = \Delta q = 0$. With this choice, the above considered stopping criteria are now equivalent.

In the sequel, we take the following values in the above described algorithms of BiCGSTAB and GCR:

$$x_0 = 0, \quad \text{tol} = \varepsilon \|b\|_2 \quad (3.12)$$

The required accuracy depends on the choice of ε , which can be given via the TRIWAQ-input. Because of mass conservation, ε should be chosen sufficiently small, in particular for complex flow problems, so that a relatively high accuracy can be obtain. The default value is $\varepsilon = 10^{-8}$.

It should be noted, however, that due to the preconditioning of the pressure system, the stopping criterion is given by $\|L^{-1}r_k\|_2 < \text{tol}$ instead of $\|r_k\|_2 < \text{tol}$. As a consequence, the accuracy may be affected by the preconditioner in the positive or negative way. Also, the influence of rounding errors can not be neglected, in particular, the BiCGSTAB algorithm is slightly more sensitive to rounding errors than GCR. Hence, it is advised to compare the norm of the final residual to the exact residual $\|b - Ax_k\|_2$. TRIWAQ allows the user to do this comparison and also to check the final convergence.

4 Numerical experiments

In this section the numerical method described in the previous sections is validated by applying it to some relatively simple test cases for which either analytic solutions or experimental data exist. Capability of the method with respect to accuracy, cost and reliability is examined.

4.1 Standing short wave in rectangular basin

Short surface waves are waves of which the wave lengths are of the same order of water depth. For these waves, contrary to the long waves, vertical accelerations can not be neglected. Based on the linear waves theory, the following important *dispersion relation* arises:

$$c = \frac{\omega}{k} = \sqrt{\frac{g}{k} \tanh(kh)} \quad (4.1)$$

where c is the propagation speed of the wave, ω is the frequency of the wave, $k = 2\pi/l$ is the wave number with l the wave length, g is the gravity acceleration and h is the water depth. Note that for shallow water or long waves, i.e.

$h \ll l$, we have $c = \sqrt{gh}$. Since, the shallow water solver TRIWAQ assumes a hydrostatic pressure distribution, it will compute the flow as if it is shallow. Hence, a correction to the hydrostatic pressure is necessary in order to predict the dispersion relation (4.1) correctly. This will be demonstrated by the following numerical example.

We consider a 2DV closed basin with length $l = 20$ m and depth $h = 10$ m. The basin is divided into 20 grid cells of each 1 m, while the depth consists of 10 sigma-layers distributed as follows: 2%, 3%, 4%, 6%, 8%, 10%, 12%, 15%, 20% and 20%. Furthermore, the time step is taken as 0.05 s. It turns out that sufficient accurate solutions have been obtained already with $\varepsilon = 0.01$ given in (3.12). There is no bottom friction and both horizontal and vertical viscosities are set to zero. Initially, the following wave height is taken:

$$\zeta(x) = \zeta_0 \cos\left(\frac{\pi x}{10}\right), \quad 0 \leq x \leq 20 \quad (4.2)$$

where ζ_0 is the small amplitude of the standing wave. In our example, we choose $\zeta_0 = 0.1$ m. Here, the wave length equals the length of the basin. According to the dispersion relation (4.1) we expect that the propagation speed $c = 5.58$ m/s, whereas the wave period $T = 2\pi/\omega = 3.59$ s.

We compare the computed time series of the surface elevation obtained with the hydrostatic and non-hydrostatic approximations in Figure 4.1. We also plot the exact solution of the wave following from potential flow theory. As expected, the wave period corresponding to the hydrostatic computation ($T = l/\sqrt{gh} = 2.02$ s) is totally wrong, whereas the non-hydrostatic case shows almost the same propagation speed of the exact wave. Moreover, according to the linear waves theory, the flow-velocity components have a typical hyperbolic profile in the vertical direction. If, however, shallow water is

assumed, these velocities are constant across the water depth. These observations are clearly demonstrated in Figure 4.2.

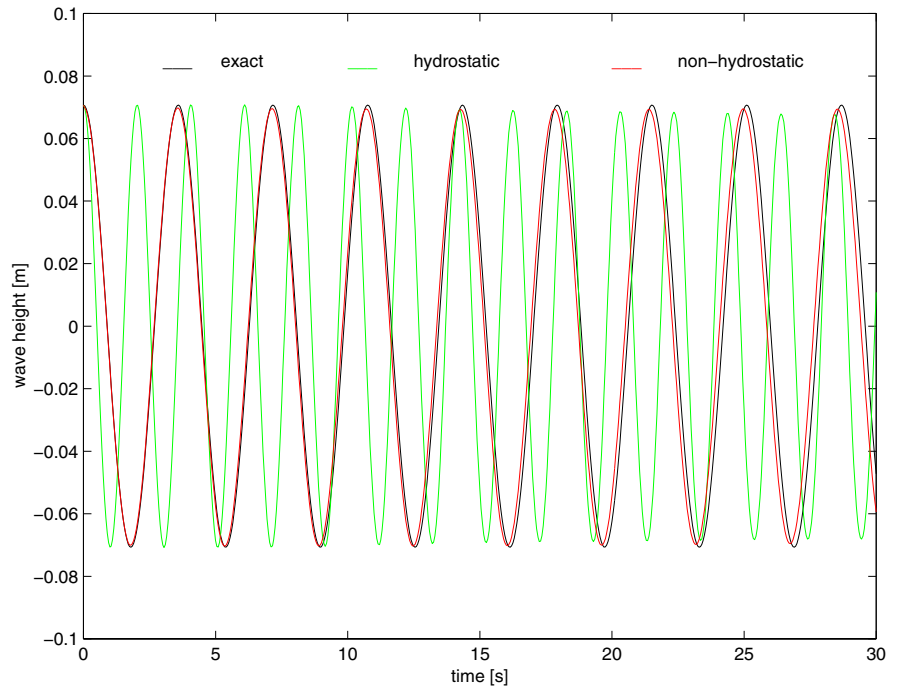


Figure 4.1: Time series of surface elevation at $x = 17.5$ m ($\Delta t = 0.05$ s, $\theta = 1$).

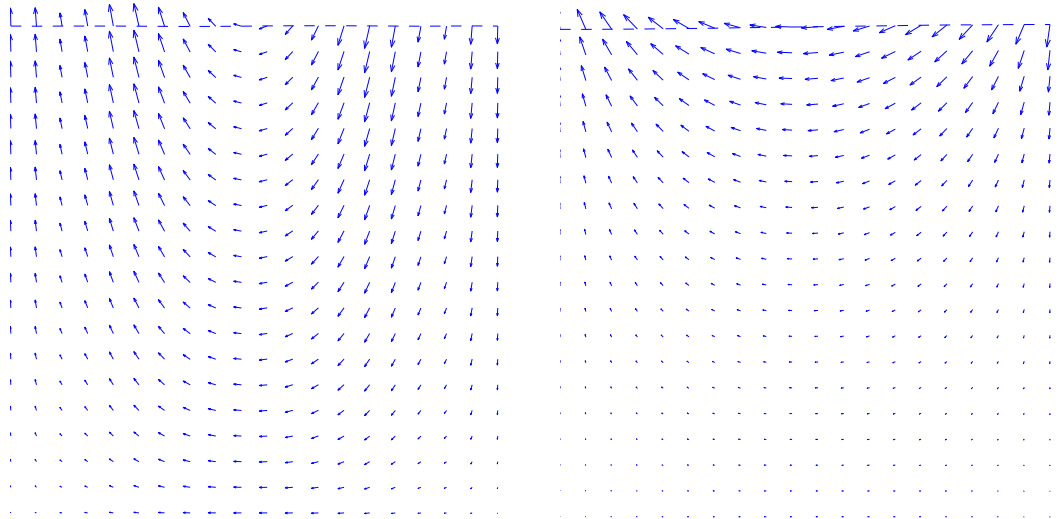


Figure 4.2: Velocity field obtained with the hydrostatic (left) and non-hydrostatic (right) approximation at $t = 0.5$ s.

The reason for taking a non-equidistant layer distribution is that a relatively small upper layer results in a more accurate wave height. This is depicted in Figure 4.3 and is a consequence of imposing the constraint (2.63). Clearly, the frequency of the wave is approximated better by the non-equidistant distribution than by the equidistant one. If one should apply a fixed layer model, a relatively large number of layers should be taken in order to calculate the wave height accurately. Hence, for short surface wave computations, the sigma-transformation is one of the major advantages.

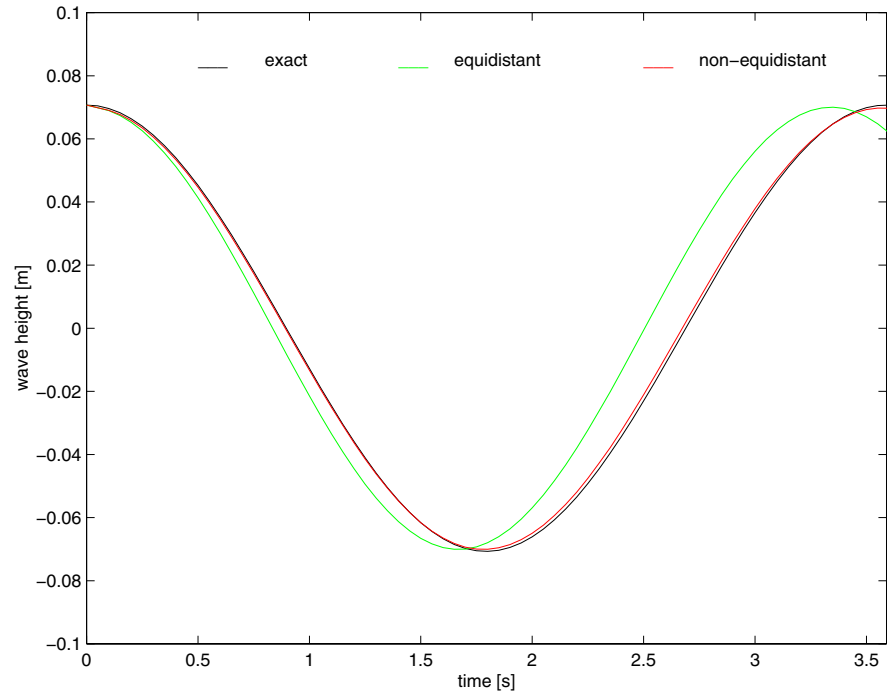


Figure 4.3: Time series of surface elevation, of which computed results are obtained with the non-hydrostatic approximation, at $x = 17.5$ m ($\Delta t = 0.05$ s, $\theta = 1$).

Our results are far more accurate than the results presented in [6]. Moreover, contrary to our results, the computed waves in the calculations of [6] are damped during their propagation. This may be caused by the splitting error introduced in the fractional step framework. To explore this, we have implemented this fractional step approach in our hydrodynamic flow solver, thus replacing the pressure correction method, and repeated the calculation of the abovementioned test case with different temporal discretizations and different time steps. The results are depicted in Figures 4.4 and 4.5. Also, the results of our method, based on the pressure correction approach, for these cases are presented in Figures 4.6 and 4.7. We have found a significant dissipation of the wave height in the fractional step framework, even when the Crank-Nicolson scheme is applied. Nevertheless, the computed propagation speed has not been affected by this approach. These results are more or less the same as given in [6]. Note that, as can be expected, at larger time steps the wiggles occurred in the results obtained with the Crank-Nicolson scheme since, this scheme does not damp high frequencies. The use of the pressure correction method yields much more accurate waves, in which the amplitude is hardly changed. Also, the second order accuracy is achieved with the Crank-Nicolson scheme, although it shows wiggles at larger time steps. Finally, we may also conclude that with the pressure correction approach a larger time step can be applied than with the fractional step technique, without affecting the accuracy too much.

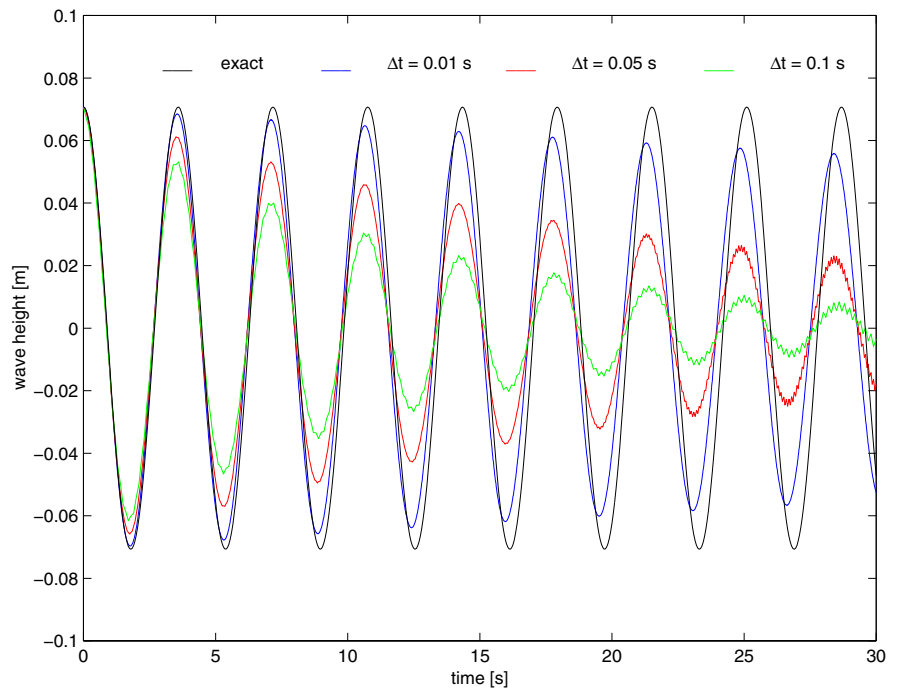


Figure 4.4: Time series of surface elevation at $x = 17.5$ m obtained with the fractional step approach and Crank-Nicolson scheme ($\theta = 0.5$) with different time steps.

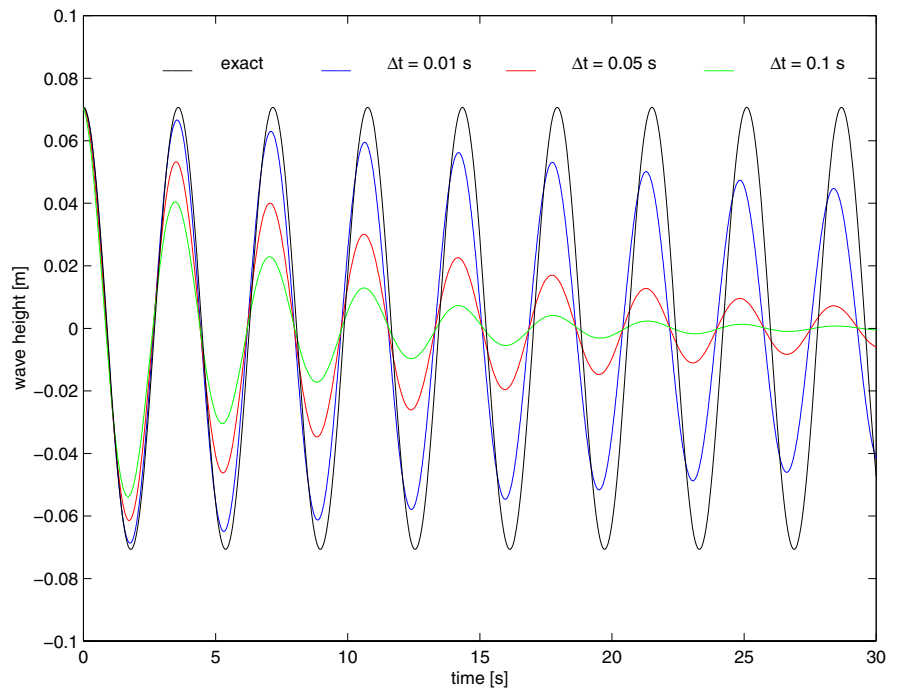


Figure 4.5: Time series of surface elevation at $x = 17.5$ m obtained with the fractional step approach and backward Euler scheme ($\theta = 1$) with different time steps.

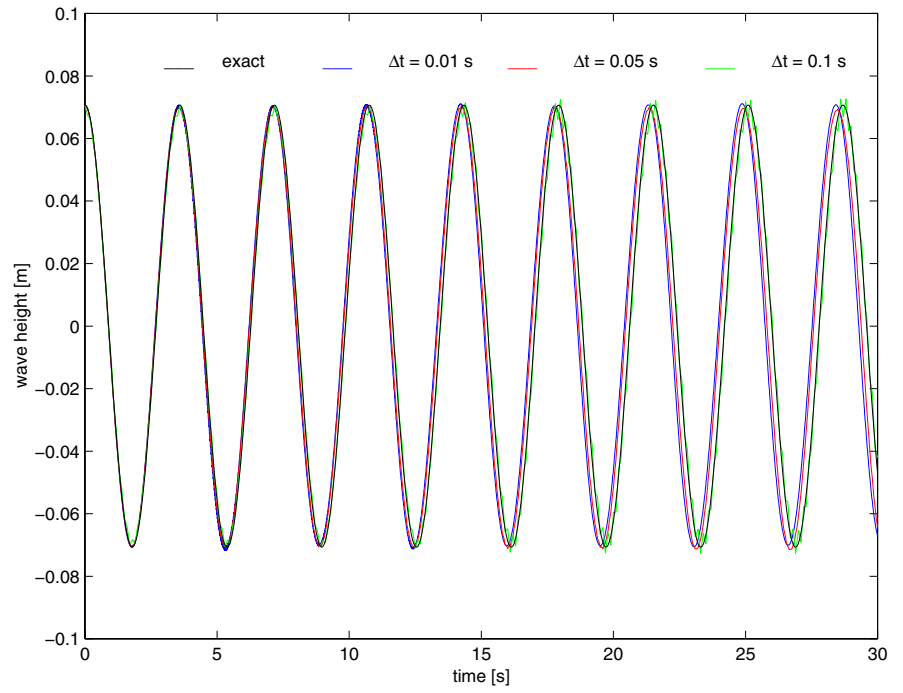


Figure 4.6: Time series of surface elevation at $x = 17.5$ m obtained with the pressure correction approach and Crank-Nicolson scheme ($\theta = 0.5$) with different time steps.

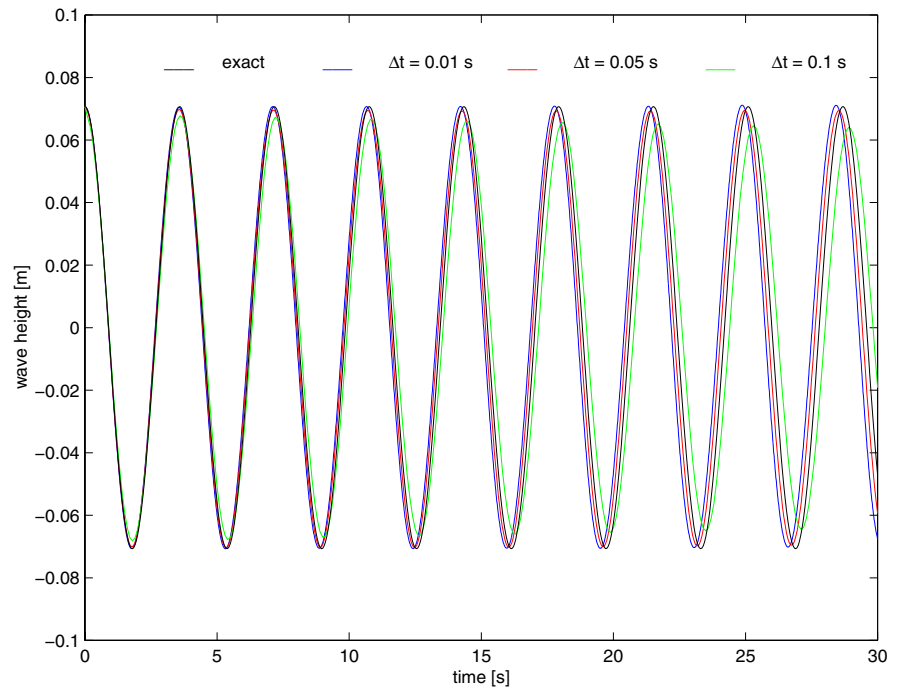


Figure 4.7: Time series of surface elevation at $x = 17.5$ m obtained with the pressure correction approach and backward Euler scheme ($\theta = 1$) with different time steps.

4.2 Exchange flow of stratified fluid

We consider a closed basin with length $l = 2$ m and depth $h = 0.3$ m which is divided into two areas with different concentrations of salt:

$$s = \begin{cases} 5.0 \text{ ppt,} & \text{if } 0 \leq x < 1 \\ 0.2 \text{ ppt,} & \text{if } 1 \leq x \leq 2 \end{cases} \quad (4.3)$$

Due to the gravity, currents develop at the bottom and free surface. The fresh water tends to intrude above the more dense fluid resulting in two fronts moving in opposite directions.

It is well-known that in this model, the vertical accelerations should not be neglected, otherwise vertical sharp fronts will occur in the computation. In reality, the slope of the fronts is smooth and finite. Figures 4.8 and 4.9 show the results of two computations with hydrostatic and non-hydrostatic discretizations, respectively. The following numerical quantities have been

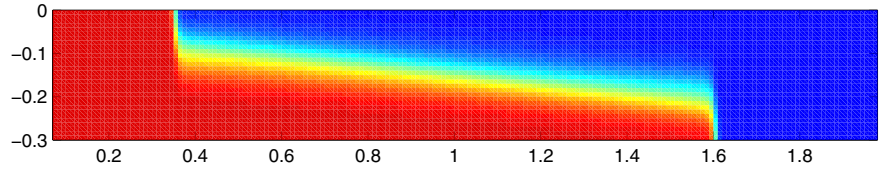


Figure 4.8: Salinity contours at $t = 13.0$ s obtained with the hydrostatic computation.

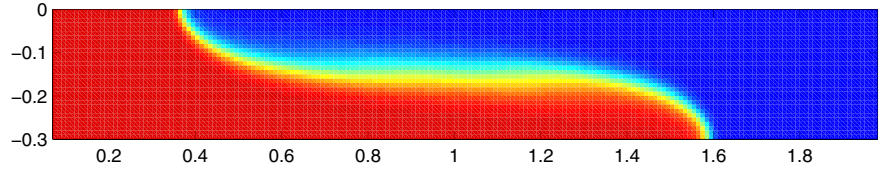


Figure 4.9: Salinity contours at $t = 13.0$ s obtained with the non-hydrostatic computation.

chosen: 30 equidistant layers and $\Delta x = 0.01$ m, $\Delta t = 0.01$ s and $\theta = 0.5$, $\nu_h = \nu_v = 10^{-4}$ m²/s and no bottom friction. For the accurate solution of the Poisson equation, $\varepsilon = 0.01$ in (3.12) has been chosen, which appears to be satisfactory.

Assuming a frictionless flow and no mixing between the layers, the front speed can be derived from the energy budget and equals [35]:

$$v_{\text{front}} = \sqrt{\frac{gh\Delta\rho}{4\rho_0}} \quad (4.4)$$

In our case, we have $\Delta\rho = 0.78\Delta s = 3.74$ kg/m³ and $\rho_0 = 1000$ kg/m³, yielding a theoretical speed of the front of 0.052 m/s. The computed front speed is approximately 0.046 m/s. In Figures 4.8 and 4.9, we also observed that the front speed of fresh water is slightly larger than that of salt wedge. For an explanation of this observation, we refer to [24].

4.3 Wave propagation over a bar

In this subsection we focus on the performance of our hydrodynamic solver in case of a flow over rapidly varying bathymetry. The model to be considered concerns the simulation of the propagation of a surface wave (induced by wind) over a bar as situated in, for example, the shoreface region. A large deformation of the wave occurs due to the interaction with the bar. The wave becomes nonlinear through the generation of higher harmonics above the upward slope of the bar and on the downward slope these harmonics become free, i.e. each component has its own celerity. In order to verify the quality of our method for simulating processes in coastal areas, we have calculated the evolution of a short wave over the same bottom topography as in the experimental tests of Beji and Battjes [5]. The domain is shown in Figure 4.10.

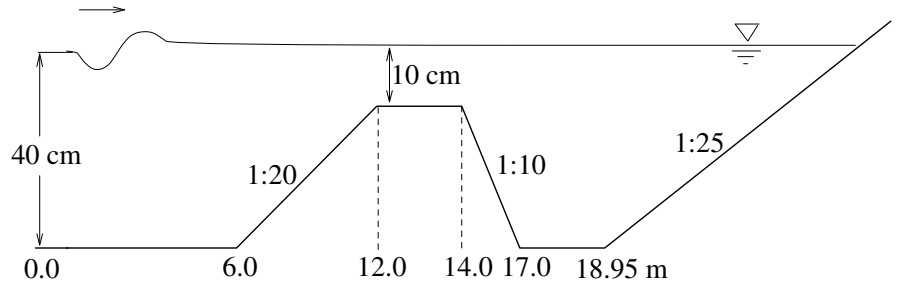


Figure 4.10: The bottom geometry.

The numerical model is 28.95 m long and the depth ranges from 40 cm near the entrance to 10 cm at the bar. In vertical direction the domain is divided into 10 equidistant sigma-layers, whereas in horizontal direction a grid spacing of 0.025 m is used. The boundary condition at the entrance consists of a sinusoidal wave with amplitude of 1 cm and period of 2.0 s. This condition is the same as given in [5]. In the experimental test, at the end of the domain there is a slope beach (1:25) that serves as a wave absorber by means of breaking. Since, our approach can not simulate breaking waves, the domain is cut down at 28.95 m from the inflow boundary and a weakly reflective boundary condition is imposed, as follows:

$$\zeta = 15 \frac{\partial}{\partial t} (2\sqrt{gh} - U) \quad (4.5)$$

where U is the depth-averaged normal velocity at the boundary (see [47] for further details).

During the hydrostatic step a nonlinear system in the unknown water levels has to be solved. This system is linearized and via an iteration process a nonlinear result can be obtained in which mass is conserved. Further details can be found in [47]. The accuracy criterion with which this iteration process has been carried out in our model is as follows:

$$\|\mathbf{u}^q - \mathbf{u}^{q-1}\|_{\infty} < 10^{-6} \quad (4.6)$$

where \mathbf{u} is the layer-averaged velocity vector, $q-1$ and q are the previous and new iteration levels, respectively. It should be noted that criterion (4.6) is much

severe than as usual in the large-scale applications. With respect to the non-hydrostatic step, a number of computations have been carried out with different values of ε as given in (3.12) in order to analyze the accuracy of the pressure system. Based on the comparison between these calculations, the obtained results are accurate enough when $\varepsilon = 0.01$. Furthermore, the Crank-Nicolson scheme ($\theta = 0.5$) has been chosen for an accurate time integration. The time step is taken as $\Delta t = 0.025$ s. Finally, bottom friction and diffusion of momentum are neglected, whereas both horizontal and vertical advective terms are discretized with at least second order accuracy.

The results of both hydrostatic and non-hydrostatic computations are plotted in Figures 4.11 and 4.12, respectively. Furthermore, the measurements of Dingemans [10] that have been obtained in a similar experiment is shown in Figure 4.12 as well. Also, for comparison, the results presented in [5], where

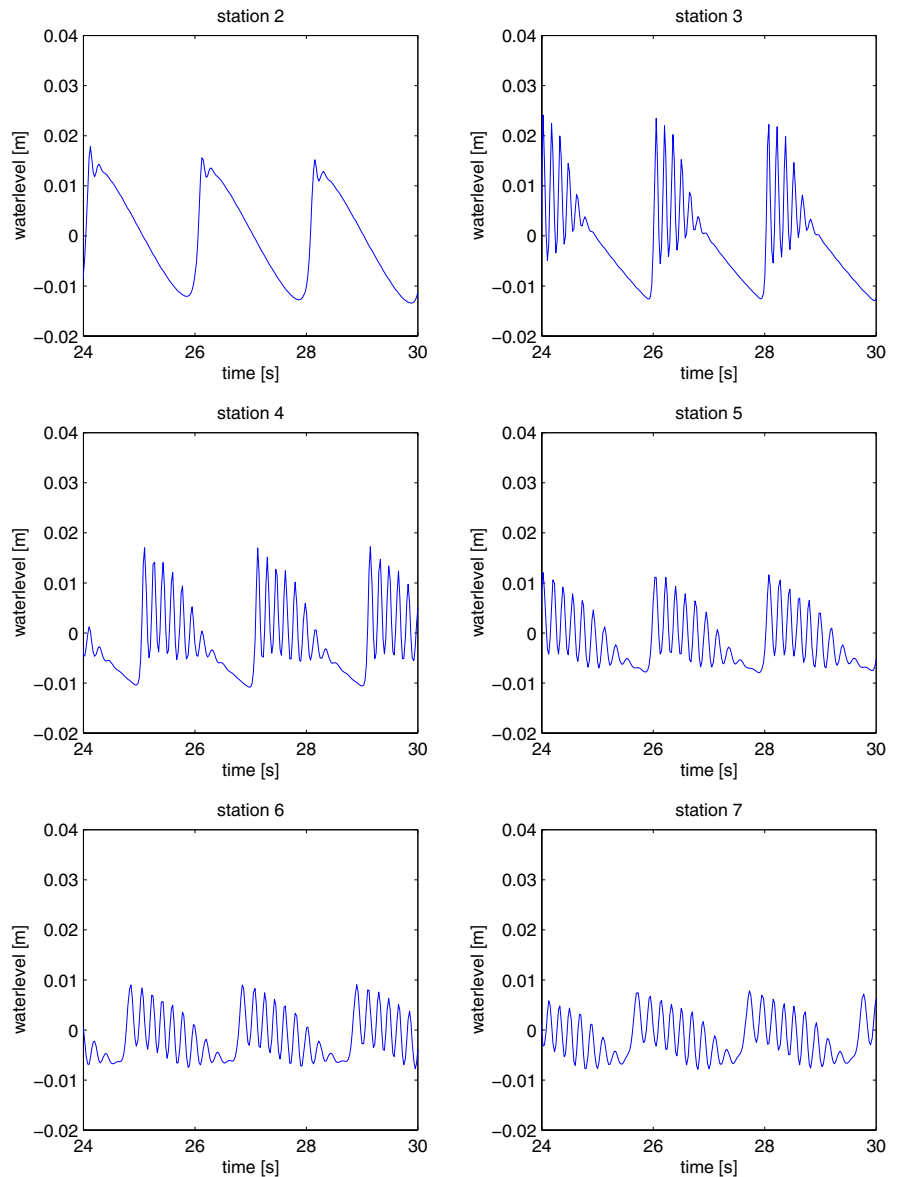


Figure 4.11: Surface elevations at several stations (cf. [5]) obtained with the hydrostatic approximation.

the measurements as well as the computed results obtained with a Boussinesq-like model are given, are depicted in Figure 4.13. Clearly, with the hydrostatic approximation, the wave height fluctuates highly, which is of course not realistic. The results obtained with the non-hydrostatic discretization are very similar to those reported by Dingemans [10] and Beji and Battjes [5], although small differences between the experimental and our results in the last station are observed. This is mainly due to the fact that in the experimental tests some set-up is to be expected due to the slope beach, which is not included in the numerical computation. Also, a weakly reflective boundary is assumed at the end of the basin, whereas the physical model has breaking waves.

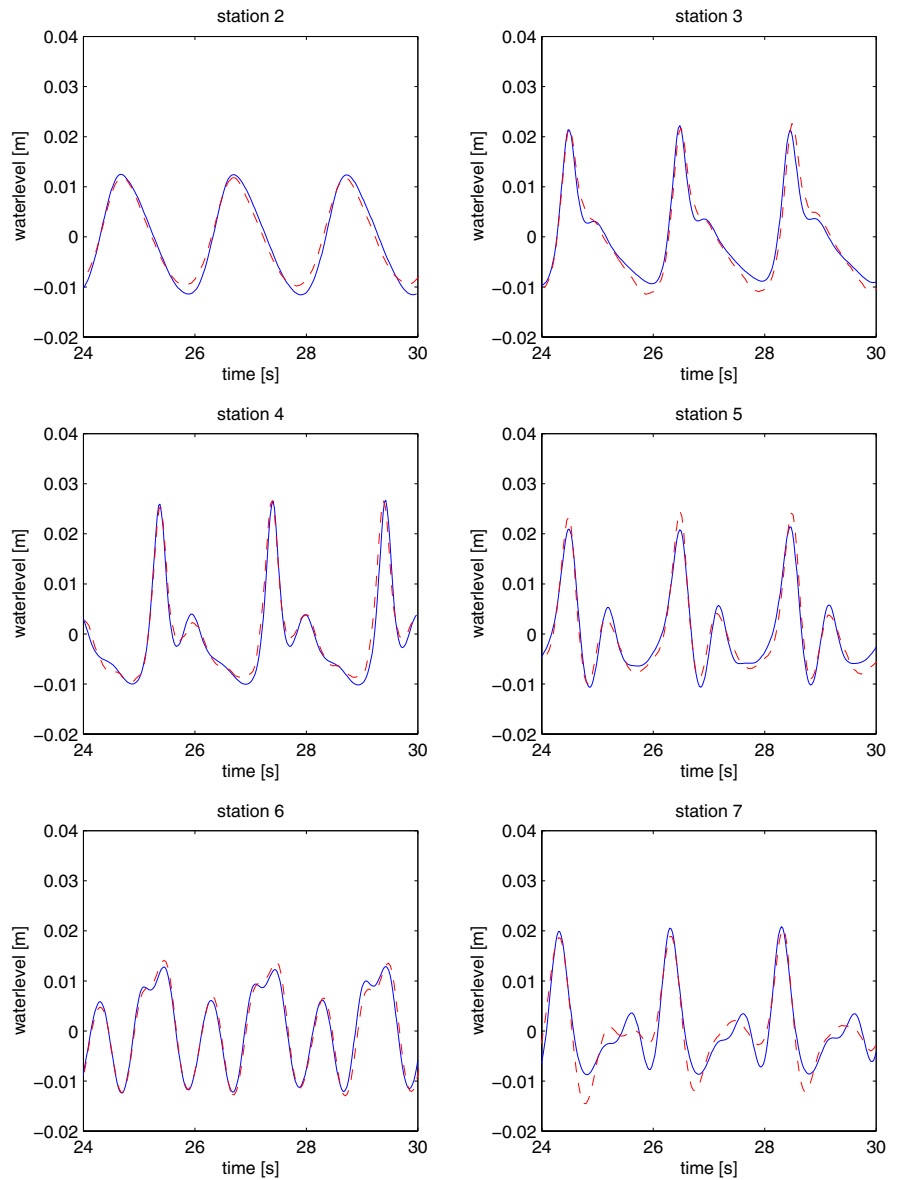


Figure 4.12: Surface elevations at several stations (cf. [5]) obtained with the non-hydrostatic approximation (—) and the measurements of Dingemans [10] (- -).

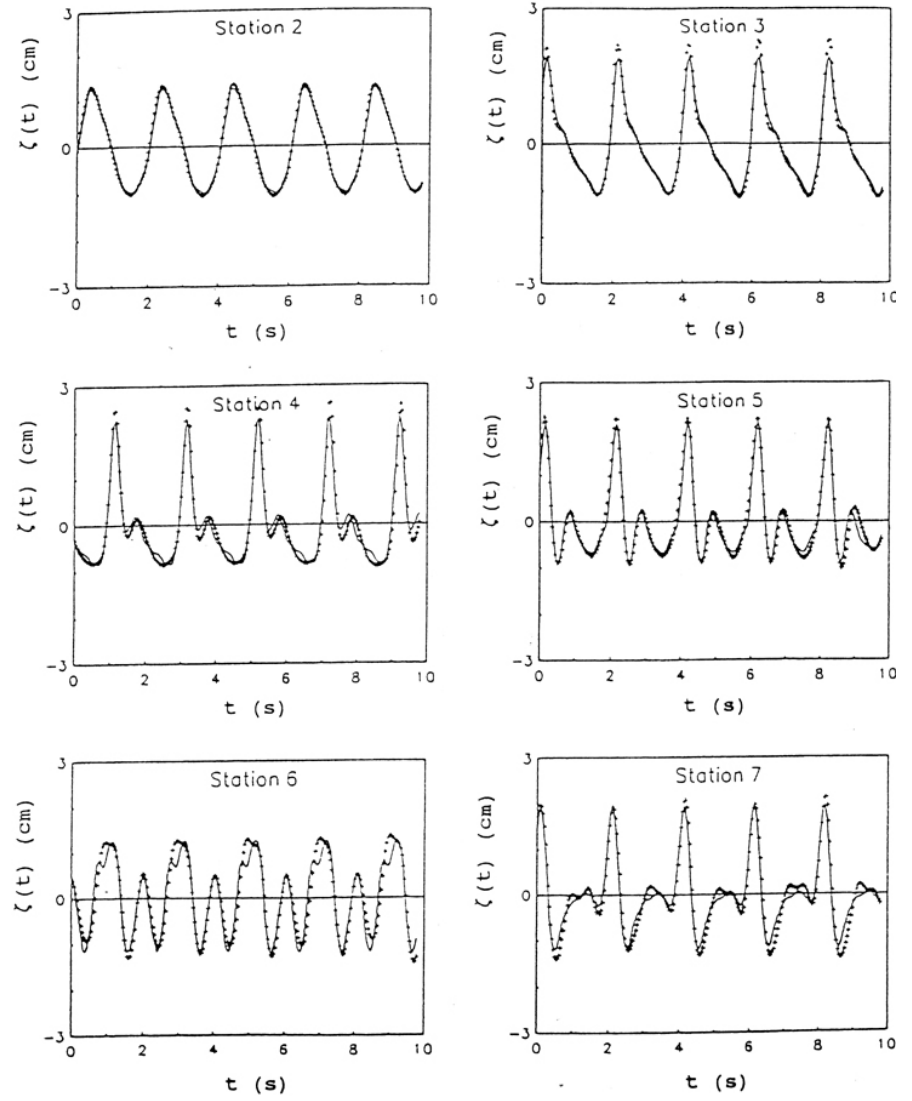


Figure 4.13: Surface elevations at several stations; measurements (—) and numerical results obtained with a Boussinesq model (+++). Taken from [5].

Due to the moving free surface and variations in depth, the sigma-planes vary in the physical space during time. Moreover, contrary to the previous test cases, the resulting pressure matrix is highly non-symmetric. Hence, this motivates us to analyze the convergence rate of the iterative solvers. Figure 4.14 presents the “averaged” l_2 -norm of the residual graphically on the logarithmic scale versus the number of iterations k of BiCGSTAB and untruncated GCR algorithms. The residual norms are averaged over a sufficient number of time steps. From this figure it appears that the convergence behaviour of both methods is linear, which means that they show a very good performance. The averaged number of iterations per time step is 4 for BiCGSTAB and 7 for untruncated GCR. The averaged CPU-time per time step for solving the system is 0.38 s for BiCGSTAB and 1.06 s for untruncated GCR. These CPU-times were measured on a Kayak-PC (450 MHz). We conclude that BiCGSTAB uses less number of iterations and is faster than untruncated GCR. These observations are also found in other test cases. Hence, for practical flow problems the BiCGSTAB algorithm is preferred. The total CPU-times used in the computations with the hydrostatic and non-hydrostatic assumption are 716 s

and 1616 s, respectively. So, the non-hydrostatic calculation costs approximately two times as the hydrostatic one.

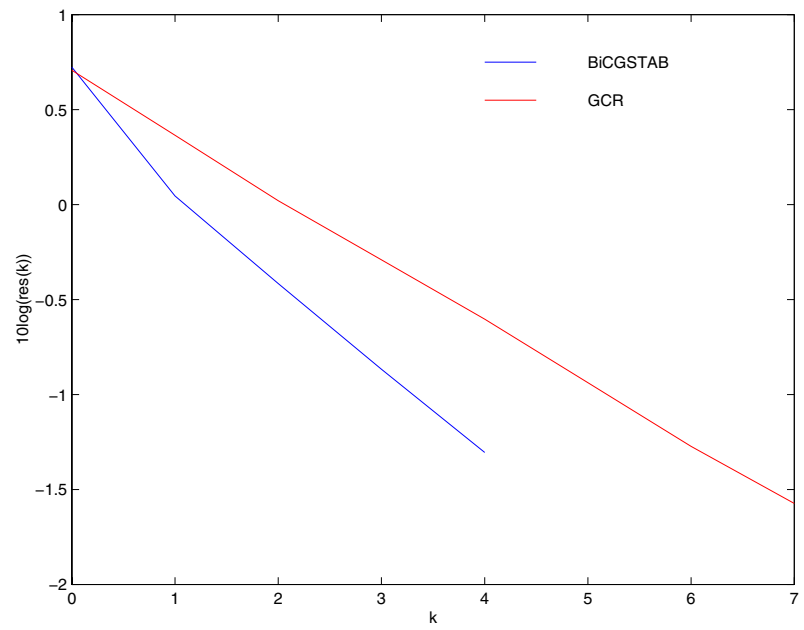


Figure 4.14: Convergence behaviour of BiCGSTAB and untruncated GCR for the wave over a nearshore bar.

5 Summary, conclusions and recommendations

For a number of so-called near-field applications, such as short wave propagations, density driven currents and cooling water discharges, there is no possibility to predict these flows accurately with the standard shallow water solver, e.g. TRIWAQ, unless a correction to the hydrostatic pressure is made.

In Section 2 we have presented the mathematical formulation of the water movement in which the correction to the hydrostatic pressure is included. Next, we have discretize in space and incorporate the boundary conditions. The same discretization techniques used in TRIWAQ have been employed. Finally, we have discretize the resulting equations in time, whereby the following sequence of the computation, based on the pressure correction approach, is employed. The solution method consists of two steps per time step, namely the hydrostatic and hydrodynamic step. In the hydrostatic step, the shallow water equations, which consist of the horizontal momentum equations and the free surface condition, are solved in the same manner as is done TRIWAQ. In these equations the explicit contribution of the hydrodynamic pressure is incorporated, which is the consequence of the application of the pressure correction scheme. After that, the momentum equation for the vertical velocity component is solved. The result of this hydrostatic step is the solution where the continuity of depth-averaged mass is guaranteed but the three-dimensional velocity field does not satisfy the local continuity equation. Hence, the hydrodynamic step must be carried out such that local continuity is satisfied. During the hydrodynamic step, the solution of the Poisson equation for the pressure correction of the hydrodynamic part is found. Next, with this correction the three-dimensional velocity field is updated, so that it becomes divergence free and thus, the continuity of local mass is guaranteed. The described approach has led to some small modifications in the existing kernel of TRIWAQ by means of adding the hydrodynamic pressure gradients to the horizontal momentum equations. Next, a new sub-kernel has been added the software package in which the hydrodynamic step is carried out. The accuracy of the method presented here is $O(\Delta x^2 + \Delta t^2)$. Furthermore, the presented method allows a larger time step than the method of Casulli and Stelling [7], without affecting the accuracy too much.

In the hydrodynamic flow model, the bulk of the computing time goes to solving pressure correction from the Poisson equation, so it pays to do this efficiently. Krylov subspace and multigrid approaches are strongly recommended for this step of the computation. In Section 3, we have restricted ourselves to Krylov subspace techniques, since they can be considered almost as black-box methods, which means that one does not need to have much knowledge of their underlying mechanisms. Ideally, one would like to have an efficient and robust procedure, which minimize the error over the whole Krylov subspace by employing short recurrences. Unfortunately, this only works for symmetric positive definite matrices, whereas we are dealing with a non-symmetric pressure system. This system arises from the discretization of the pressure correction technique, in which the sigma-transformation is used. For this system, two appropriate algorithms have been implemented in TRIWAQ, that either have the optimality property but lack a short recursion formula and vice versa, namely GCR and BiCGSTAB, respectively. Although, GCR can be very expensive in work and storage, it can be making cheap by means of truncation, which has only a small impact on the convergence. Since,

preconditioning techniques can change the spectrum of the system in a way that is favourable for fast convergence, both BiCGSTAB and GCR algorithms are therefore always preconditioned. A very successful class of preconditioning methods consists of incomplete LU decompositions. In Section 3.5, the so-called RILUD(α) preconditioner is discussed. For efficiency, $\alpha = 0.965$ has been chosen. This preconditioner is further optimized by means of a row scaling and by using the Eisenstat implementation. Finally, an appropriate stopping criterion for the pressure system is based on the ratio of the norm of the final residual and the norm of the right-hand side of the system, whereas at the start of the iteration process we always take the zero vector for the solution.

Based on the results discussed in Section 4, we may conclude that the hydrodynamic solver presented in this report can capture the essential phenomena in the small-scale environment, such as density driven currents and flow with wave propagation and deformation near irregular bottom topography. Furthermore, it is expected that the solver can predict the interaction of the tidal flow and short wave propagation as well. It appears that the presented method is very accurate, even if the number of layers is relatively small (let say 5 to 10), thanks to the sigma-transformation. However, it can not cope with one layer, because then it reduces to a fully hydrostatic flow model.

At this moment, our hydrodynamic model is far from complete. First, in the present implementation, horizontal viscous terms are not incorporated in the vertical momentum equation. This equation should be extended with these terms and perhaps also the anti-creepage terms. These latter terms arise from the sigma-transformation of the viscous terms. Secondly, with respect to the turbulence k- ϵ modelling, the production term of turbulent kinetic energy need to be extended with the gradient of the vertical velocity. Thirdly, some typical wave aspects like wave breaking and radiation of different wave modes at open boundaries should also be included. Finally, the present model should be tested for real three-dimensional flow problems as well.

An important remark is made with respect to the efficiency of the hydrodynamic model. The underlying approach for obtaining accurate predictions of near-field flows, namely pressure correction, greatly increases the computational complexity. In spite of the very efficient implementation of the Poisson solvers, such as the preconditioned BiCGSTAB algorithm, the total computing time is approximately doubled when we abandon the hydrostatic pressure distribution. Nevertheless, it is very robust, since no instability nor slow convergence were encountered in the test cases presented in this report. Yet, it is believed that further significant acceleration of the presented method in time can be realised through parallelisation and domain decomposition. In this context, the method can be made more competitive with the shallow water solver TRIWAQ by means of the application of domain decomposition, in which the pressure correction need only to be calculated for subdomains where vertical accelerations are relatively important.

The follow-up of the research presented in this report is the application of the non-hydrostatic approach in the ZEEDELTA model for an accurate prediction of the flow and the salt intrusion in the Haringvliet estuary, where hydrodynamic effects may occur near the Haringvliet sluices. In this context, the presented ZEEDELTA model [1] is not able to resolve the flow with an energy loss through the sluices due to relative large gridsizes. For this reason, a 3D barrier formulation has been developed which is based on a so-called Q-h relation [48]. It relate the flow-rate through the barrier to the fall across the barrier. However, the main disadvantage of this barrier formulation is its dependence on the external information, such as the measurements and sluice structures,

needed to determine the so-called discharge coefficients. Also, the uncertainty of this determination depends strongly on the accuracy and the completeness of the measurements (see [9] and [33]). As a consequence, the predictive capability of the model is reduced. However, as an alternative, the discharge coefficients may be calculated by means of a non-hydrostatic flow computation of the ZEEDELTA model without employing the 3D barrier formulation. Near the Haringvliet sluices a relatively small grid should be employed for accuracy reasons. This can be realised by means of domain decomposition. With the specified discharge coefficients, a fast and more accurate calculation of the ZEEDELTA model including the 3D barrier formulation is enabled and forms a basis for the operational forecast. This follow-up will be carried out next year.

Literature

- [1] Alkyon (ingenieursbureau). Zeedelta model; bouw en eerste afregeling. Report RIKZ/OS-98.111X (A225), National Institute for Coastal and Marine Management, The Hague, The Netherlands, 1998 (in dutch).
- [2] O. Axelsson and G. Lindskog. On the eigenvalue distribution of a class of preconditioning methods. *Numer. Math.*, 48:479-498, 1986.
- [3] R. Barrett, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine and H.A. van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods*. SIAM, Philadelphia, PA, USA, 1994.
- [4] G.K. Batchelor. *An introduction to fluid dynamics*. Cambridge University Press, Cambridge, 1967.
- [5] S. Beji and J.A. Battjes. Numerical simulation of nonlinear wave propagation over a bar. *Coast. Engng.*, 23:1-16, 1994.
- [6] V. Casulli. A semi-implicit finite difference method for non-hydrostatic, free-surface flows. *Int. J. Numer. Meth. Fluids*, 30:425-440, 1999.
- [7] V. Casulli and G.S. Stelling. Numerical simulation of 3D quasi-hydrostatic, free-surface flows. *J. Hydr. Engng.*, 124:678-686, 1998.
- [8] A.J. Chorin. Numerical solution of the Navier-Stokes equations. *Math. Comput.*, 22:745-762, 1968.
- [9] P.F.D. Cohen. *Gevoeligheidsonderzoek met een numeriek getijmodel voor nauwkeurige reproductie van de waterbeweging in het Haringvliet; toetsing met meetgegevens uit 1997 bij gewijzigd beheer van de Haringvlietstuiven*. Master thesis, Delft University of Technology. Report RIKZ/OS-99.126X, National Institute for Coastal and Marine Management, The Hague, The Netherlands, 1999 (in dutch).
- [10] M.W. Dingemans. *Water wave propagation over uneven bottoms*. Ph.D. thesis, Delft University of Technology, 1994.
- [11] S.C. Eisenstat. Efficient implementation of a class of preconditioned conjugate gradient methods. *SIAM J. Sci. Stat. Comput.*, 2:1-4, 1981.
- [12] S.C. Eisenstat, H.C. Elman and M.H. Schultz. Variational iterative methods for nonsymmetric systems of linear equations. *SIAM J. Numer. Anal.*, 20:345-357, 1983.
- [13] V. Faber and T. Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient method. *SIAM J. Numer. Anal.*, 21:352-362, 1984.
- [14] R. Fletcher. Factorizing symmetric indefinite matrices. *Lin. Alg. and its Appl.*, 14:257-277, 1976.

-
- [15] R.W. Freund and N.M. Nachtigal. QMR: a quasi minimal residual method for non-Hermitian linear systems. *Numer. Math.*, 60:315-339, 1991.
 - [16] G.H. Golub and C.F. Van Loan. *Matrix computations*. John Hopkins University Press, Baltimore, USA, 1989.
 - [17] M.P. Gresho and R.L. Sani. On pressure boundary conditions for the incompressible Navier-Stokes equations. *Int. J. Numer. Meth. Fluids*, 7:1111-1145, 1987.
 - [18] I.A. Gustafsson. A class of first order factorization methods. *BIT*, 18:142-156, 1978.
 - [19] F.H. Harlow and J.E. Welch. Numerical calculation of time-dependent viscous incompressible flow of fluid with a free surface. *Phys. of Fluids*, 8:2182-2189, 1965.
 - [20] M.R. Hestenes and E. Stiefel. Methods of conjugate gradients for solving linear systems. *J. Res. Nat. Bur. Standards*, 49:409-436, 1952.
 - [21] C. Hirsch. *Numerical computation of internal and external flows, Vols. 1 and 2*. John Wiley and Sons, Chichester, 1990.
 - [22] C.P. Jackson and P.C. Robinson. A numerical study of various algorithms related to the preconditioned conjugate gradient method. *Int. J. for Numer. Meth. Engng.*, 21:1315-1338, 1985.
 - [23] J.J.I.M. Van Kan. A second-order accurate pressure correction method for viscous incompressible flow. *SIAM J. Sci. Stat. Comput.*, 7:870-891, 1986.
 - [24] C. Kranenburg. *Dichtheidsstromen (handleiding college b81)*. Delft University of Technology, Faculty of Civil Engineering, Delft, The Netherlands, 1996 (in dutch).
 - [25] A. Mahadevan, J. Oliger and R. Street. A nonhydrostatic mesoscale ocean model. Part II: numerical implementation. *J. Phys. Oceanogr.*, 26:1881-1900, 1996.
 - [26] J.A. Meijerink and H.A. van der Vorst. An iterative solution method for linear systems of which the coefficient matrix is a symmetric M-matrix. *Math. Comput.*, 31:148-162, 1977.
 - [27] S.V. Patankar. *Numerical heat transfer and fluid flow*. McGraw-Hill, New York, 1980.
 - [28] Y. Saad. A flexible inner-outer preconditioned GMRES algorithm. *SIAM J. Sci. Stat. Comput.*, 14:461-469, 1993.
 - [29] Y. Saad and M. Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 7:856-869, 1986.
 - [30] G.L.G. Sleijpen, H.A. van der Vorst and D.R. Fokkema. Bi-CGTAB(I) and other hybrid Bi-CG methods. *Numer. Algor.*, 7:75-109, 1994.

-
- [31] P. Sonneveld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 10:36-52, 1989.
- [32] G.S. Stelling. Short note on non-hydrostatic models (very preliminary and very incomplete). Delft University of Technology, Faculty of Civil Engineering, Delft, The Netherlands, May 2000.
- [33] Svašek (ingenieursbureau). 3D-simulatie van de zoutindringing in het Haringvliet tijdens de meetproef maart 1997, Fase 3: Gevoeligheid sluisformulering. Report RIKZ/OS-99.152X, National Institute for Coastal and Marine Management, The Hague, The Netherlands, 1999 (in dutch).
- [34] R. Temam. Sur l'approximation de la solution des equations de Navier-Stokes par la méthode des pas fractionnaires. *Arch. Rational Mech. Anal.*, (I): 32:135-153; (II): 33:377-385, 1969.
- [35] J.S. Turner. *Buoyancy effects in fluids*. Cambridge University Press, 1973.
- [36] R.S. Varga. *Matrix iterative analysis*. Prentice-Hall, Englewood Cliffs, NJ, 1962.
- [37] H.A. van der Vorst. Iterative solution methods for certain sparse linear systems with a non-symmetric matrix arising from pde-problems. *J. Comput. Phys.*, 44:1-19, 1981.
- [38] H.A. van der Vorst. High performance preconditioning. *SIAM J. Sci. Stat. Comput.*, 10:1174-1185, 1989.
- [39] H.A. van der Vorst. BI-CGSTAB: a fast and smoothly converging variant of BICG for the solution of nonsymmetric linear systems. *SIAM J. Sci. Stat. Comput.*, 13:631-644, 1992.
- [40] H.A. van der Vorst. Parallel linear systems solvers: sparse iterative methods. In P. Wesseling, editor, *High performance computing in fluid dynamics*, pages 173-200, Ercoftac Series, vol. 3, Kluwer Academic Publishers, 1996.
- [41] H.A. van der Vorst and C. Vuik. The superlinear convergence behaviour of GMRES. *J. Comput. Appl. Math.*, 48:327-341, 1993.
- [42] H.A. van der Vorst and C. Vuik. GMRESR: a family of nested GMRES methods. *Numer. Lin. Alg. with Appl.*, 1:369-386, 1994.
- [43] C. Vuik. Termination criteria for GMRES-like methods to solve the discretized incompressible Navier-Stokes equations. Report 92-50, Delft University of Technology, Faculty of Technical Mathematics and Informatics, Delft, The Netherlands, 1992.
- [44] C. Vuik. Further experiences with GMRESR. *Supercomputer*, 55:13-27, 1993.
- [45] P. Wesseling. *An introduction to multigrid methods*. John Wiley and Sons, Chichester, 1992.
-

-
- [46] D.M. Young. *Iterative solution of large linear systems*. Academic Press, New York, 1971.
 - [47] M. Zijlema. TRIWAQ - three-dimensional shallow water flow model. Technical documentation, Version 1.1, RKZ-438, National Institute for Coastal and Marine Management, The Hague, The Netherlands, 1998.
 - [48] M. Zijlema. Modelling van de sluis. Implementatie van een vernieuwde 3D sluisformulering in TRIWAQ voor gedeeltelijk geopende sluisen. Report RIKZ/OS/2000.106X, National Institute for Coastal and Marine Management, The Hague, The Netherlands, 2000 (in dutch).