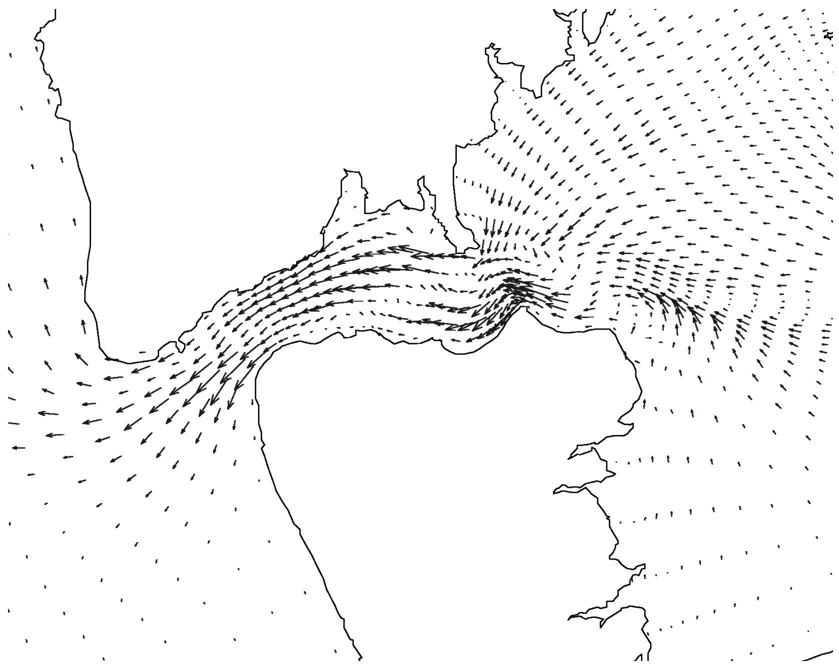
Numerical Accuracy in Solutions of the Shallow-Water Equations

Master Thesis



Peterjan Broomans
Delft, February 2003

Abstract

A few decades ago computational fluid dynamics (CFD) was in its infancy. Limitations on memory and computation speed compelled CFD programmers to use single precision values, which was adequate at that time. Through time the computation speed grew exponentially, and prices of memory banks dropped. So, almost naturally the size and complexity of problems in CFD have increased, but the computation precision has not.

It appears that for large simulations, up to 100.000 grid points or more, the acquired accuracy is questionable. If in the (near) future increased computation speed would allow even larger simulations, measures should be taken to ensure sufficiently accurate results. Increasing the machine precision would achieve this aim. Furthermore, numerical adaptations might also increase the accuracy. Row-scaling will for instance improve the results of the continuity equation at hardly any expense. Stricter and/or different stopping criteria for iterative processes will surely increase the accuracy in several cases.

Several processes, like the turbulence model and the flooding and drying criteria, cause a certain loss of accuracy, which is not always compensated by accuracy improvements. Much could be gained if they were to be improved.

An addition to the complexity of CFD in the course of years is the transition from two-dimensional to three-dimensional models. Unconditional stability of the scheme for the discretised two-dimensional shallow-water equations seems to have become conditional in the three-dimensional case. The numerical stability has become very much dependent on boundary conditions.

Contents

Acknowledgment

This report is the result of my research at WL | Delft Hydraulics. I would like to thank Kees Vuik and Jan Mooiman for their advise and input during this period. Furthermore, I want to express my gratitude towards Prof.dr.ir. P. Wesseling, Arthur Mynett, Jan van Kester, Stef Hummel, Rob Uittenbogaard and Herman Gerritsen for their help and input.

List of symbols

symbol	*description*	*unit*
d	water depth with respect to a reference level	$[m]$
f	Coriolis parameter	$[1/s]$
f_x	Coriolis force component in x-direction (per unit mass)	$[m/s^2]$
f_y	Coriolis force component in y-direction (per unit mass)	$[m/s^2]$
f_z	Coriolis force component in z-direction (per unit mass)	$[m/s^2]$
g	gravitational acceleration	$[m/s^2]$
H	total water depth ($\zeta + d$)	$[m]$
k	turbulent kinetic energy	$[m^2/s^2]$
l_m	mixing length	$[m]$
p	pressure	$[kg/(ms^2)]$
$\bar{p}$	time-averaged pressure	$[kg/(ms^2)]$
p_a	atmospheric pressure	$[kg/(ms^2)]$
r	relative residue	[-]
u	flow velocity component in x-direction	$[m/s]$
$\bar{u}$	time-averaged flow velocity component in x-direction	$[m/s]$
U	depth-averaged flow velocity component in x-direction	$[m/s]$
v	flow velocity component in y-direction	$[m/s]$
$\bar{v}$	time-averaged flow velocity component in y-direction	$[m/s]$
V	depth-averaged flow velocity component in y-direction	$[m/s]$
w	flow velocity component in z-direction	$[m/s]$
$\bar{w}$	time-averaged flow velocity component in z-direction	$[m/s]$
α_ζ	‘reflection coefficient’ (water level boundary)	$[s^2]$
α_U	‘reflection coefficient’ (velocity boundary)	$[s]$
ϵ	dissipation rate of turbulent kinetic energy	$[m/s]$
ϵ_{it}	critical value for original stopping criterion	$[m/s]$
ϵ_r	critical value for residual stopping criterion	[-]
ζ	free surface elevation with respect to a reference level	$[m]$
κ	Von Kármán constant	[-]
κ_∞	condition number in the infinity norm	[-]
κ_2	condition number in the Euclidian norm	[-]
κ_{eff}	effective condition number	[-]
μ	machine precision	[-]
ν	kinematic viscosity	$[m^2/s]$
ν_t	eddy viscosity	$[m^2/s]$
ν_t^{2D}	part of horizontal eddy viscosity due to ”2D turbulence”	$[m^2/s]$
ν_t^H	horizontal eddy viscosity	$[m^2/s]$
ν_t^V	vertical eddy viscosity	$[m^2/s]$

symbol	*description*	*unit*
ρ	density	$[kg/m^3]$
ρ	spectral radius (of a matrix)	[-]
ρ_0	reference density	$[kg/m^3]$
ρ_c	contraction factor	[-]
σ	transformed vertical coordinate	[-]
τ	time step	$[s]$
τ_{bx}	bed shear stress in x-direction	$[1/(ms)]$
τ_{by}	bed shear stress in y-direction	$[1/(ms)]$
$\vec{\tau}_s$	wind stress	$[1/(ms)]$
ϕ	geophysical latitude	[-]
Ω	angular speed of the earth	$[1/s]$
$\underline{\Omega}$	rotation vector of the earth	$[1/s]$
ω	transformed vertical flow velocity component	$[m/s]$

Chapter 1

Introduction

The shallow-water equations describe the flow of water in rivers, lakes and shallow seas, like the North Sea. The computer program Delft3D-FLOW can be used to compute a numerical approximation of the solution of the shallow-water equations. The result of a Delft3D-FLOW simulation can be used to solve the transport equations for some contaminant or to determine whether dikes along a river need to be heightened. They are also much used to predict the flows around and near yet unbuilt structures or possible adjustments to them in seas and rivers, like for instance Flyland, the second 'Maasvlakte' or new harbours.

The solution obtained by Delft3D-FLOW is the product of a modelling process subject to uncertainties and errors, see Figure 1.1. Error and uncertainty can be defined, according to AIAA (1998), as

error: a recognisable deficiency in any phase or activity of modelling and simulation that is not due to lack of knowledge.

uncertainty: a potential deficiency in any phase or activity of modelling the modelling process that is due to lack of knowledge.

These definitions recognise the deterministic nature of errors and the stochastic nature of uncertainty.

The first step in the modelling process is the derivation the physical model, which are in our case the Navier-Stokes equations and the continuity equation. It is generally accepted that they describe natural water flows very well. However, without any simplifications they are very difficult to solve. Therefore, in the next step application of a number of assumptions lead us to a conceptual model: the three-dimensional shallow-water equations. These model assumptions with respect to several physical aspects (viscosity, compressibility, the turbulence model, boundary conditions) are the general source of uncertainty within the modelling process. This uncertainty is assumed to be the main cause of the difference between experimental and computational data (Oberkampf and Blottner, 1998). Uncertainty will not be dealt with in this report.

The complexity of the shallow-water equations impedes us to determine the analytical (i.e. exact) solution, so an approximation will have to be calculated. The next step in the modelling process is the construction of a numerical model for the shallow-water equations. During this construction truncation errors are made. When solving the numerical model small errors, like rounding errors and iteration errors, are unavoidable. To a certain extent it is possible to quantify these errors.

The aim of this research is to track down, explain and quantify numerical errors in Delft3D-FLOW and, if necessary, find solutions to reduce these errors. For this purpose several test

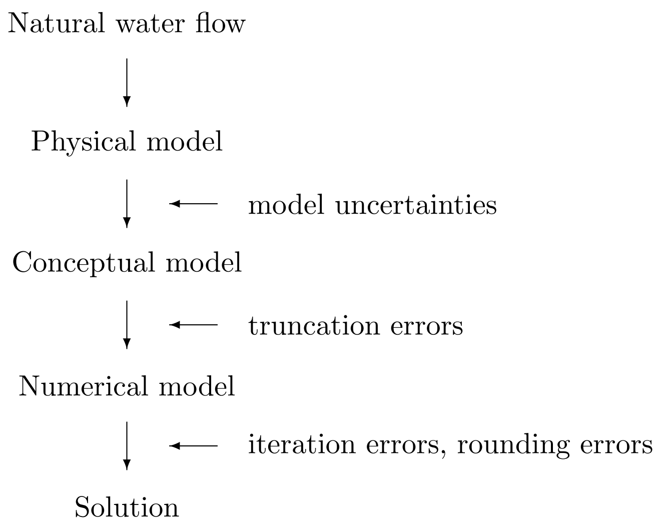


Figure 1.1: Uncertainties and errors in the various steps of the solution process.

cases will be set up, which show erroneous results. Analysis of these results will show or rule out possible causes. If they have been discovered, possible solutions will be tested.

This report treats the complete solution process as shown in Figure 1.1. The derivation of the shallow-water equations from the Navier-Stokes equations and the boundary conditions are dealt with in the second chapter. The third chapter focuses on the staggered grid, the numerical scheme and the procedure for solving the numerical equations. The fourth chapter contains a discourse on analysis of rounding errors and iteration errors of the numerical schemes that are used in Delft3D-FLOW. In the fifth chapter a stability analysis for the three-dimensional shallow-water equations is carried out. The results of the analyses from chapter four and five are performed on several test cases in chapter six, of which the conclusions and recommendations are summarised in the chapters seven and eight.

Chapter 2

Flow Model

The Navier-Stokes equations are a general model which can be used to model water flows in many applications. However, when considering a specific problem, such as shallow-water flows, in which the horizontal scale is much larger than the vertical one the three-dimensional shallow-water equations will suffice.

In this chapter we will derive the Reynolds-averaged Navier-Stokes equations describing turbulent flows for which the length scale of the turbulence is much smaller than that of the problem. From this we will derive the three-dimensional shallow-water equations under the assumption that the pressure is hydrostatically distributed. These equations will be transformed to so-called σ-coordinates, after which the boundary conditions and the turbulence modelling will be discussed.

2.1 Reynolds-Averaged Navier-Stokes Equations

The Navier-Stokes equations are given by

$$\begin{aligned}
\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} &= -\frac{1}{\rho_0}\frac{\partial p}{\partial x} + \nu\triangle u - f_x \\
\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} &= -\frac{1}{\rho_0}\frac{\partial p}{\partial y} + \nu\triangle v - f_y \\
\frac{\partial w}{\partial t} + u\frac{\partial w}{\partial x} + v\frac{\partial w}{\partial y} + w\frac{\partial w}{\partial z} &= -\frac{1}{\rho_0}\frac{\partial p}{\partial z} + \nu\triangle w - f_z - \frac{\rho}{\rho_0}g,
\end{aligned} \tag{2.1}$$

where u, v and w denote the velocity components in the x-, y- and z-direction respectively, ρ the density, ρ_0 the reference density, p the pressure, ν the kinematic viscosity and f_x, f_y and f_z represent the components of the Coriolis forces per unit mass. They are defined by $(f_x, f_y, f_z)^T = -2\underline{\Omega} \times (u, v, w)^T$, where $\underline{\Omega}$ is the earth's rotation vector. The Navier-Stokes equations in (2.1) are valid under the assumption that the density is constant or if the Boussinesq approximation applies.

Assumption 1 (Boussinesq approximation). *The Boussinesq approximation states that if density variations are small the density may be assumed constant in all terms except the gravitational term.*

Due to turbulence eddies small variations occur in the flow velocities and pressure. Usually these variations are too small to be represented in a numerical scheme unless the grid is chosen very fine. To deal with this phenomenon we first decompose the velocities and the pressure as follows

$$u = \bar{u} + u', \quad v = \bar{v} + v', \quad w = \bar{w} + w' \quad \text{and} \quad p = \bar{p} + p', \tag{2.2}$$

where the overbar represents time-averaged quantities. For instance $\bar{u}$ is defined by

$$\bar{u}(t) := \frac{1}{T}\int_t^{t+T} u(\tau)d\tau.$$

The period T should be larger than the turbulence time scale, but smaller than long periodic effects such as the tide. The turbulent fluctuations are given by u', v', w' and p'. Note that the time-averages of these fluctuations are zero, i.e. $\frac{1}{T}\int_t^{t+T} u'\, d\tau = 0$.

When substituting (2.2) into (2.1) and averaging the resulting equations over time the Reynolds-averaged Navier-Stokes equations or simply Reynolds equations arise for turbulent flows. They read

$$\frac{\partial \bar{u}}{\partial t} + \bar{u}\frac{\partial \bar{u}}{\partial x} + \bar{v}\frac{\partial \bar{u}}{\partial y} + \bar{w}\frac{\partial \bar{u}}{\partial z} + \frac{\partial \overline{u'u'}}{\partial x} + \frac{\partial \overline{u'v'}}{\partial y} + \frac{\partial \overline{u'w'}}{\partial z} = -\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial x} - \bar{f}_x \tag{2.3a}$$

$$\frac{\partial \bar{v}}{\partial t} + \bar{u}\frac{\partial \bar{v}}{\partial x} + \bar{v}\frac{\partial \bar{v}}{\partial y} + \bar{w}\frac{\partial \bar{v}}{\partial z} + \frac{\partial \overline{v'u'}}{\partial x} + \frac{\partial \overline{v'v'}}{\partial y} + \frac{\partial \overline{v'w'}}{\partial z} = -\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial y} - \bar{f}_y \tag{2.3b}$$

$$\frac{\partial \bar{w}}{\partial t} + \bar{u}\frac{\partial \bar{w}}{\partial x} + \bar{v}\frac{\partial \bar{w}}{\partial y} + \bar{w}\frac{\partial \bar{w}}{\partial z} + \frac{\partial \overline{w'u'}}{\partial x} + \frac{\partial \overline{w'v'}}{\partial y} + \frac{\partial \overline{w'w'}}{\partial z} = -\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial z} - \bar{f}_z - \frac{\rho}{\rho_0}g. \tag{2.3c}$$

The correlations between fluctuating velocity components ($\overline{u'u'}$, $\overline{u'v'}$, etc.) are unknown. These correlations are responsible for a loss of momentum in the mean flow direction and therefore appear to act as stresses on the fluid. They are called Reynolds stresses. These stresses are much larger than the viscous stresses which have therefore been neglected.

Assumption 2 (Eddy viscosity concept or Boussinesq hypothesis). *Reynolds stresses like viscous stresses depend on the deformation of the mean flow. Thus, the Reynolds stresses are modelled as*

$$\overline{u'v'} = -\nu_t\left(\frac{\partial \bar{v}}{\partial x} + \frac{\partial \bar{u}}{\partial y}\right), \tag{2.4}$$

where ν_t is the so-called eddy viscosity. This eddy viscosity is a priori unknown and a suitable expression has to be constructed (see section 2.5).

2.2 Three-Dimensional Shallow-Water Equations

We speak of shallow water only when a flow satisfies certain characteristic relations.

Assumption 3 (Shallow water). *(i) The characteristic horizontal length scale is much larger than the characteristic vertical length scale. (ii) The characteristic vertical velocity is small in comparison with the characteristic horizontal velocity (Jin, 1993).*

These assumptions allow that the terms $\frac{\partial \bar{w}}{\partial x}$ and $\frac{\partial \bar{w}}{\partial y}$ are neglected. The difference between the horizontal and the vertical length scale justifies a distinction between a horizontal (ν_t^H) and a vertical (ν_t^V) eddy viscosity. But more importantly, the momentum equation in the vertical direction (2.3c) reduces to the hydrostatic pressure distribution

$$\frac{\partial \bar{p}}{\partial z} = -\rho g. \tag{2.5}$$

Integrating this equation results in

$$\bar{p}(x, y, z, t) = g\int_z^\zeta \rho\, dz + p_a, \tag{2.6}$$

where $\zeta = \zeta(x, y, t)$ is the free surface level against the reference plane $z = 0$ and p_a is the atmospheric pressure. Substituting this result in the pressure term of (2.3a) and using Leibnitz' integration rule, yields

$$-\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial x} = -\frac{\rho g}{\rho_0}\frac{\partial \zeta}{\partial x} - \frac{g}{\rho_0}\int_z^\zeta \frac{\partial \rho}{\partial x}\, dz' - \frac{1}{\rho_0}\frac{\partial p_a}{\partial x}.$$

The horizontal pressure gradient is described by differences of the water level ζ through the barotropic term, the first term on the right-hand side, and by density differences in horizontal direction through the baroclinic term, the second term. The last term on the right-hand side describes the contribution of the atmospheric pressure. If we had taken ρ constant, (2.6) would read $\bar{p} = \rho g(\zeta - z) + p_a$ and for the pressure term of (2.3a) we would have

$$-\frac{1}{\rho_0}\frac{\partial \bar{p}}{\partial x} = -g\frac{\partial \zeta}{\partial x} - \frac{1}{\rho_0}\frac{\partial p_a}{\partial x}.$$

If we neglect the atmospheric pressure gradient, the horizontal pressure gradient will reduce to the barotropic term.

When substituting equations (2.4) and (2.6) in equations (2.3a) and (2.3b), taking the density constant, neglecting the atmospheric pressure gradient yields and dropping the overbar, we obtain

$$\begin{aligned}\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + w\frac{\partial u}{\partial z} &= -g\frac{\partial \zeta}{\partial x} + fv \\ &+ 2\frac{\partial}{\partial x}\left(\nu_t^H \frac{\partial u}{\partial x}\right) + \frac{\partial}{\partial y}\left(\nu_t^H\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right) + \frac{\partial}{\partial z}\left(\nu_t^V \frac{\partial u}{\partial z}\right)\end{aligned} \tag{2.7}$$

and

$$\begin{aligned}\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + w\frac{\partial v}{\partial z} &= -g\frac{\partial \zeta}{\partial y} - fu \\ &+ \frac{\partial}{\partial x}\left(\nu_t^H\left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x}\right)\right) + 2\frac{\partial}{\partial y}\left(\nu_t^H \frac{\partial v}{\partial y}\right) + \frac{\partial}{\partial z}\left(\nu_t^V \frac{\partial v}{\partial z}\right),\end{aligned} \tag{2.8}$$

where f, the Coriolis parameter, is defined by

$$f = 2\Omega \sin\phi$$

with Ω the angular speed of the earth and ϕ the latitude. The equations (2.7), (2.8) and the incompressible continuity equation

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \tag{2.9}$$

are called the shallow-water equations. At this point we have a system of three equations (2.7)-(2.9), three unknown velocity components, $u(x, y, z, t)$, $v(x, y, z, t)$ and $w(x, y, z, t)$, and a free boundary, i.e. the unknown free surface $\zeta(x, y, t)$. So, a fourth equation is needed to compute the unknown variables.

To this end we integrate the continuity equation along the vertical axis, what results in

$$w(x, y, \zeta, t) - w(x, y, d, t) = -\int_{-d}^{\zeta} \frac{\partial u}{\partial x}\, dz - \int_{-d}^{\zeta} \frac{\partial v}{\partial y}\, dz, \tag{2.10}$$

where $d = d(x, y)$ is the water depth below the reference plane $z = 0$. Morphological changes of the bed due to the water flow are in general very small and they are therefore neglected. Thus d is not dependent on the time. Equation (2.10) can be rewritten by using substitutions for w at the bottom and the water surface. For $z = \zeta(x, y, t)$ we have

$$w = \frac{d\zeta}{dt} = \frac{\partial \zeta}{\partial t} + u\frac{\partial \zeta}{\partial x} + v\frac{\partial \zeta}{\partial y}. \tag{2.11}$$

A similar expression at the bottom reads

$$w = -u\frac{\partial d}{\partial x} - v\frac{\partial d}{\partial y}. \tag{2.12}$$

Substituting (2.11) and (2.12) into (2.10) and using Leibnitz' integration rule, yields

$$\frac{\partial \zeta}{\partial t} = -\frac{\partial}{\partial x}\int_{-d}^{\zeta} u\, dz - \frac{\partial}{\partial y}\int_{-d}^{\zeta} v\, dz. \tag{2.13}$$

If we define the depth-averaged velocities U and V by $U := \frac{1}{H}\int_{-d}^{\zeta} u\, dz$ and $V := \frac{1}{H}\int_{-d}^{\zeta} v\, dz$, where $H = H(x, y, t) = \zeta + d$ is the water depth, (2.13) reads

$$\frac{\partial \zeta}{\partial t} + \frac{\partial HU}{\partial x} + \frac{\partial HV}{\partial y} = 0. \tag{2.14}$$

2.3 Transformed Equations in σ-coordinates

The bottom and the water surface are usually not parallel to the reference plane ($z = 0$). In order to cope with uneven bottom topographies in numerical applications a transformation is applied to so-called σ-coordinates (Phillips, 1957). This transformation stretches the vertical direction, such that the transformed water depth is constant in space and time. The σ-coordinates are defined by

$$\tilde{x} = x, \quad \tilde{y} = y, \quad \sigma = \frac{z - \zeta}{H}, \quad \tilde{t} = t,$$

where $H = H(x, y, t) = \zeta + d$ is the water depth. The time-derivative in σ-coordinates reads

$$\frac{\partial}{\partial t} = \frac{\partial}{\partial \tilde{t}} + \frac{\partial \sigma}{\partial \tilde{t}}\frac{\partial}{\partial \sigma}.$$

For the spatial derivatives in the horizontal directions similar expressions hold. In the vertical direction we have

$$\frac{\partial}{\partial z} = \frac{1}{H}\frac{\partial}{\partial \sigma}.$$

The hydrostatic pressure relation (2.5) after transformation to σ-coordinates and integration along the vertical axis, reads

$$\tilde{p} = p_a + gH\int_{\sigma}^{0} \rho(\tilde{x}, \tilde{y}, \sigma', \tilde{t})\, d\sigma'. \tag{2.15}$$

The transformed vertical velocity is defined by

$$\begin{aligned}\omega := H\frac{\mathrm{D}\sigma}{\mathrm{D}\tilde{t}} &= H\left[\frac{\partial}{\partial t}\left(\frac{z-\zeta}{H}\right) + u\frac{\partial}{\partial x}\left(\frac{z-\zeta}{H}\right) + v\frac{\partial}{\partial y}\left(\frac{z-\zeta}{H}\right)\right] \\ &= w - \left(\frac{\partial \zeta}{\partial t} + u\frac{\partial \zeta}{\partial x} + v\frac{\partial \zeta}{\partial y}\right) - \sigma\left(\frac{\partial H}{\partial t} + u\frac{\partial H}{\partial x} + v\frac{\partial H}{\partial y}\right).\end{aligned} \tag{2.16}$$

The horizontal velocities u and v remain strictly horizontal after the transformation. Hence $\tilde{u} = u$ and $\tilde{v} = v$. When we substitute equation (2.16) in equation (2.9), noting that H and ζ are not dependent on σ, but $\tilde{u}$ and $\tilde{v}$ are, and assuming that d is not time-dependent, we obtain the continuity equation in transformed coordinates:

$$\frac{\partial \zeta}{\partial \tilde{t}} + \frac{\partial H\tilde{u}}{\partial \tilde{x}} + \frac{\partial H\tilde{v}}{\partial \tilde{y}} + \frac{\partial \omega}{\partial \sigma} = 0. \tag{2.17}$$

Integrating equation (2.17) from the bottom to the surface and using Leibnitz' integration rule, yields

$$\frac{\partial \zeta}{\partial \tilde{t}} + \frac{\partial HU}{\partial \tilde{x}} + \frac{\partial HV}{\partial \tilde{y}} = 0, \tag{2.18}$$

where U and V are depth-averaged velocities defined by $U = \int_{-1}^{0} \tilde{u}\, d\sigma$ and $V = \int_{-1}^{0} \tilde{v}\, d\sigma$. Logically this equation is equal to (2.14) in the previous section, because they are integrated along the vertical axis and therefore invariant under the σ-transformation. Note that the definitions of U and V is equivalent to the definitions in the previous section.

After transformation of (2.7) and (2.8) to σ-coordinates the momentum equations in x- and y-direction of the shallow-water equations arise. They read, dropping the tilde,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\omega}{H}\frac{\partial u}{\partial \sigma} = -\frac{1}{\rho_0}\left(\frac{\partial p}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial p}{\partial \sigma}\right) + fv + F_x + \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_t^V \frac{\partial u}{\partial \sigma}\right) \tag{2.19}$$

and

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\omega}{H}\frac{\partial v}{\partial \sigma} = -\frac{1}{\rho_0}\left(\frac{\partial p}{\partial y} + \frac{\partial \sigma}{\partial y}\frac{\partial p}{\partial \sigma}\right) - fu + F_y + \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_t^V \frac{\partial v}{\partial \sigma}\right). \tag{2.20}$$

Note that the density ρ is not taken constant. The terms F_x and F_y represent the horizontal viscosity terms. They are given by

$$F_x = \left(\frac{\partial}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial}{\partial \sigma}\right)\tau_{xx} + \left(\frac{\partial}{\partial y} + \frac{\partial \sigma}{\partial y}\frac{\partial}{\partial \sigma}\right)\tau_{xy}$$

and

$$F_y = \left(\frac{\partial}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial}{\partial \sigma}\right)\tau_{xy} + \left(\frac{\partial}{\partial y} + \frac{\partial \sigma}{\partial y}\frac{\partial}{\partial \sigma}\right)\tau_{yy},$$

where the Reynolds stresses τ_{xx}, τ_{xy} and τ_{yy} satisfy

$$\begin{aligned}
\tau_{xx} &= 2\nu_t^H \left(\frac{\partial u}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial u}{\partial \sigma}\right) \\
\tau_{xy} &= \nu_t^H \left(\frac{\partial u}{\partial y} + \frac{\partial \sigma}{\partial y}\frac{\partial u}{\partial \sigma} + \frac{\partial v}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial v}{\partial \sigma}\right) \\
\tau_{yy} &= 2\nu_t^H \left(\frac{\partial v}{\partial y} + \frac{\partial \sigma}{\partial y}\frac{\partial v}{\partial \sigma}\right).
\end{aligned}$$

For large scale problems with coarse horizontal grids, i.e. when shear-stresses along the boundaries may be neglected, the forces F_x and F_y may be simplified. The horizontal eddy viscosity ν_t^H is assumed constant and the horizontal eddy viscosity terms are reduced to the Laplace operator:

$$F_x = \nu_t^H \triangle u \quad \text{and} \quad F_y = \nu_t^H \triangle v \tag{2.21}$$

with $\triangle = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. After application of this simplification we allow the horizontal eddy viscosity to vary in space.

Finally we will look at the horizontal pressure gradient in σ-coordinates, which can be written as

$$\frac{\partial p}{\partial x} + \frac{\partial \sigma}{\partial x}\frac{\partial p}{\partial \sigma} = \frac{\partial p}{\partial x} - \frac{1}{H}\left(\frac{\partial \zeta}{\partial x} + \sigma \frac{\partial H}{\partial x}\right)\frac{\partial p}{\partial \sigma}.$$

Substituting the integrated hydrostatic pressure relation (2.15) results in

$$\frac{\partial p_a}{\partial x} + g\frac{\partial}{\partial x}\left(H\int_{\sigma}^{0} \rho \, d\sigma'\right) + g\rho\left(\frac{\partial \zeta}{\partial x} + \sigma\frac{\partial H}{\partial x}\right).$$

This expression has been implemented in Delft3D-FLOW. If we take the density constant and neglect the atmospheric pressure gradient, the horizontal pressure gradient reduces to the barotropic term $g\rho\frac{\partial \zeta}{\partial x}$ as shown before in the previous section.

2.4 Boundary Conditions

Due to the impermeability of the surface and the bottom the following conditions apply:

$$\omega|_{\sigma=-1} = \omega|_{\sigma=0} = 0.$$

By imposing these conditions we neglect evaporation, rainfall and exchange with the ground water. At the sea bed the boundary conditions for the momentum equations are given by

$$\left.\frac{\nu_t^V}{H}\frac{\partial u}{\partial \sigma}\right|_{\sigma=-1} = \frac{1}{\rho}\tau_{bx} \tag{2.22}$$

and

$$\left.\frac{\nu_t^V}{H}\frac{\partial v}{\partial \sigma}\right|_{\sigma=-1} = \frac{1}{\rho}\tau_{by} \tag{2.23}$$

with τ_{bx} and τ_{by} the components of the bed shear stress in x- and y-direction, respectively.

Wind stresses are responsible for non-homogeneous boundary conditions for the momentum equations at the free surface. They read

$$\left.\frac{\nu_t^V}{H}\frac{\partial u}{\partial \sigma}\right|_{\sigma=0} = \frac{1}{\rho}\,|\vec{\tau}_s|\cos\theta \tag{2.24}$$

and

$$\left.\frac{\nu_t^V}{H}\frac{\partial v}{\partial \sigma}\right|_{\sigma=0} = \frac{1}{\rho}\,|\vec{\tau}_s|\sin\theta \tag{2.25}$$

with $|\vec{\tau}_s|$ the wind stress and θ the angle between the wind stress vector and x-axis.

Assumption 4 (Wind stress). *The expression for the wind stress is*

$$|\vec{\tau}_s| = \rho_a C_d U_{10}^2$$

where ρ_a is the air density, U_{10} is the wind speed 10 metres above the water surface and C_d is the wind drag coefficient, which depends on U_{10}.

This dependence is empirical and several implementations are possible, for example a linear relation, a constant value for C_d or a combination of the two.

The conditions at the vertical boundary planes are divided in open and closed boundary conditions. Closed boundaries are situated at the transition between land and water. Open boundaries are virtual "water-water" boundaries. These open boundaries have been introduced in order to limit the computational area.

At the closed boundaries the flow condition prohibiting flow through the boundary reads

$$\underline{n} \cdot \begin{pmatrix} u \\ v \end{pmatrix} \Bigg|_{(x,y)=(x_b,y_b)} = 0 \quad \forall \sigma \in [-1, 0]$$

where (x_b, y_b) is any point on the closed boundary and $\underline{n}$ consists of the first two components of the outward normal of the vertical boundary plane in that point. The slip condition imposed along the closed boundaries depends on the scale of the problem. In large scale problems the tangential shear stresses along closed boundaries can be neglected (free slip condition). For small scale problems this influence is not negligible and is taken into account through a partial slip condition.

At open boundaries the water level, the velocity or the discharge should be prescribed. The velocity at an open boundary is chosen perpendicular to that boundary for computational reasons. For an open boundary one of the following three boundary conditions can be prescribed. They are

$$\begin{aligned} \zeta &= F_\zeta(t) \quad \text{(prescribing the water level)}, \\ U &= F_U(t) \quad \text{(prescribing the normal velocity component)}, \\ Q &= F_Q(t) \quad \text{(prescribing the discharge)}, \end{aligned} \tag{2.26}$$

where ζ is the free surface level (see page 4), U the depth-averaged normal velocity component (see page 7) and Q the discharge through the boundary plane. For simplicity we have assumed that the open boundary is perpendicular to the x-axis, so we only have to specify the velocity component along the x-axis. We will speak of a left and a right open boundary, see Figure 2.1.

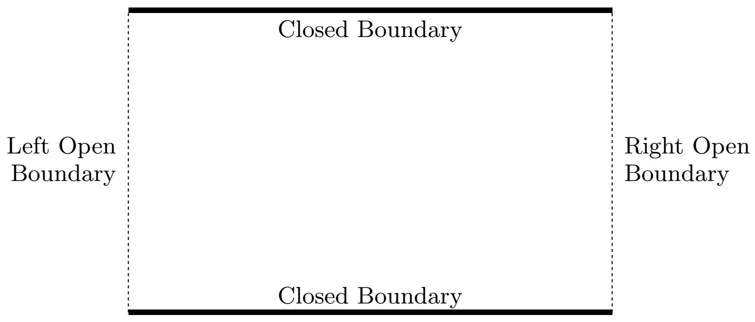


Figure 2.1: An example of a horizontal rectangular geometry.

For the boundary conditions of the velocity and the discharge type a profile has to be prescribed in the vertical direction. This profile can be uniform or logarithmic. Also a complete 3D-profile can be prescribed in which for various depths different conditions are specified.

If an outgoing wave at an open boundary is not prescribed exactly then that wave will (partially) reflect at the boundary and propagate as a disturbance in the area. The boundary conditions in (2.26) will in general cause wave reflections. A weakly-reflective boundary condition is based on the Riemann invariants $U \pm 2\sqrt{gH}$. The boundary condition specified by the user would then be

$$U \pm 2\sqrt{gH} = F_R(t)$$

with "+" for the left and "−" for the right boundary. For numerical purposes it is convenient to have a linearised boundary condition.

Assumption 5. *For weakly-reflecting open boundary conditions the variations of the water level ζ (with respect to the reference plane) are small in comparison with the water depth d.*

Under this assumption the linearised Riemann invariants read

$$\begin{aligned} U \pm 2\sqrt{gH} = U \pm 2\sqrt{g(\zeta + d)} &= U \pm 2\sqrt{gd} \pm \zeta \frac{2g}{2\sqrt{g(\zeta + d)}} + \mathcal{O}(\zeta^2) \\ &\approx U \pm 2\sqrt{gd} \pm \zeta \sqrt{\frac{g}{d}}. \end{aligned}$$

To reduce the reflective properties of the water level and velocity boundary conditions in (2.26) the time-derivative of the Riemann invariant is added. The new boundary conditions are

$$\begin{aligned} \zeta + \alpha_\zeta \frac{\partial}{\partial t}\left(U \pm 2\sqrt{gH}\right) &= F_\zeta(t), \\ U + \alpha_U \frac{\partial}{\partial t}\left(U \pm 2\sqrt{gH}\right) &= F_U(t) \end{aligned}$$

for the water level and the velocity respectively. The reflection coefficients α_ζ and α_U should be chosen sufficiently small to damp short waves introduced by the initial conditions.

2.5 Turbulence Closure Models

In water flows small eddies occur due to turbulence. The grid used in numerical applications is usually too coarse to resolve the turbulent quantities u', v' and w'. Though initially they appear in the Reynolds-averaged Navier-Stokes equations (2.3), due to Assumption 2, the eddy viscosity concept (on page 4), they have vanished. However, an unknown parameter ν_t, the eddy viscosity, has been introduced, for which in shallow waters a horizontal and a vertical version are distinguished. In this section possible expressions or models for ν_t are treated, which are an important part of the eddy viscosity concept.

The horizontal eddy viscosity, ν_t^H, is usually chosen constant. Physically this is unsound, because the eddy viscosity is a flow property, but for large scale problems it has proved to be sufficiently accurate. The vertical eddy viscosity, ν_t^V, is computed according to one of the turbulence closure models. The models in which suitable expressions for ν_t are given or calculated are called turbulence closure models. The following models are available:

1. Constant coefficient.
2. Algebraic turbulence closure model.
3. k–L turbulence closure model.
4. k–ϵ turbulence closure model.

The first model is the simplest but also the most unrealistic as it will lead to a laminar flow. The other models are based on the eddy viscosity concept of Kolmogorov and Prandtl. In this concept the eddy viscosity is related to a characteristic length scale, $\mathcal{L}$, and a characteristic velocity scale, $\mathcal{U}$. This relation reads

$$\nu_t^V \sim \mathcal{L}\mathcal{U}.$$

The explicit form, also known as the Kolmogorov-Prandtl expression, is

$$\nu_t^V = c'_\mu l_m \sqrt{k},$$

where c'_μ is constant derived from the empirical constant c_μ in the k–ϵ model, l_m is the mixing length and k is the turbulent kinetic energy. The last three models differ in their prescription for l_m and k.

In the algebraic model an expression is given for l_m and k. For the mixing length one of the most common choices is given by the Bakhmetev distribution which reads

$$l_m = \kappa(z+d)\sqrt{1-\frac{z+d}{H}}$$

with κ the Von Kármán constant, $\kappa \approx 0.41$. An algebraic expression for k depends on the friction velocities or the velocity gradients.

In the k–L closure model the mixing length is analytically prescribed by $l_m = c_D k^{3/2}/\epsilon$, where c_D is a model coefficient, ϵ is the dissipation rate of turbulent kinetic energy and the turbulent kinetic energy follows from a transport equation.

The k–ϵ closure model uses transport equations for both the turbulent kinetic energy k as well as the dissipation rate of turbulent kinetic energy ϵ. These transport equations read

$$\frac{\partial k}{\partial t} + u\frac{\partial k}{\partial x} + v\frac{\partial k}{\partial y} + \frac{\omega}{H}\frac{\partial k}{\partial \sigma} = \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\frac{\nu_t^V}{\sigma_k}\frac{\partial k}{\partial \sigma}\right) + P_k + B_k - \epsilon,$$
$$\frac{\partial \epsilon}{\partial t} + u\frac{\partial \epsilon}{\partial x} + v\frac{\partial \epsilon}{\partial y} + \frac{\omega}{H}\frac{\partial \epsilon}{\partial \sigma} = \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\frac{\nu_t^V}{\sigma_\epsilon}\frac{\partial \epsilon}{\partial \sigma}\right) + P_\epsilon + B_\epsilon - c_{2\epsilon}\frac{\epsilon^2}{k},$$

where $c_{2\epsilon}$ is a model coefficient, σ_k and σ_ϵ are the Prandtl-Schmidt numbers, P_k and P_ϵ are the production terms and B_k and $B_{k/\epsilon}$ are the buoyancy terms. These equations can be solved with appropriate initial and boundary conditions. Detailed descriptions of these models are beyond the scope of this report. For more information we refer to Launder and Spalding (1972) and Rodi (1985).

Large eddy simulation

Large eddies occur mainly in rivers, harbours and estuaries. Especially in harbours they are unwanted, since they cause sediment deposition, making it necessary to dredge regularly.

The closure models mentioned above recognise the difference between a horizontal and a vertical eddy viscosity. These models prescribe a constant value for the horizontal eddy viscosity, as that is sufficient for large scale problems. However, this approach is physically unsound because the eddy viscosity depends on the local flow characteristics, and it should therefore not be used if large eddies (at least larger than the grid size) play an important role.

The field of large eddy simulation is still highly in development. An 'easy' possibility is to design a new closure model for the horizontal eddy viscosity. Uittenbogaard *et al.* (1992) proposed the following definition for the horizontal eddy viscosity

$$\nu_t^H = \nu_t^{2D} + \nu_t^V,$$

where ν_t^{2D} is the part due to "2D turbulence" and ν_t^V, the vertical eddy viscosity, is the part due to "3D turbulence". Bijvelds (2001) constructed the 'two-length-scale k–ϵ model' to calculate ν_t^{2D} in which is assumed that large eddies are quasi-2D, so a depth-averaged version of the k–ϵ closure model can be used.

Chapter 3

Numerical Scheme for the Shallow-Water Equations

The analytical solution of the shallow-water equations can only be calculated in a very small number of cases, which however do not occur in nature. A numerical approximation is therefore almost inevitable. In Delft3D-FLOW a finite difference scheme on a staggered grid is chosen using an ADI solver in conjunction with a Gauss-Jacobi iteration. In the next sections these numerical methods will be discussed.

3.1 Shallow-Water Equations in Appropriate Form

The equations which are actually going to be approximated numerically are the horizontal momentum equations (2.19) and (2.20), the continuity equation (2.17) and the integrated continuity equation (2.18). The pressure in the momentum equations is given by equation (2.15). Substituting (2.15) in the pressure terms of the momentum equations in x- and y-direction, setting the density constant and neglecting the atmospheric pressure gradient, these terms reduce to the barotropic terms

$$-g\frac{\partial \zeta}{\partial x} \quad \text{and} \quad -g\frac{\partial \zeta}{\partial y},$$

respectively. For the horizontal eddy viscosity terms F_x and F_y we will use the simplified equations in (2.21).

For convenience we repeat the above mentioned equations, taking the remarks into account,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} + \frac{\omega}{H}\frac{\partial u}{\partial \sigma} = -g\frac{\partial \zeta}{\partial x} + fv + \nu_t^H\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) + \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_t^V\frac{\partial u}{\partial \sigma}\right), \tag{3.1a}$$

$$\frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} + \frac{\omega}{H}\frac{\partial v}{\partial \sigma} = -g\frac{\partial \zeta}{\partial y} - fu + \nu_t^H\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) + \frac{1}{H^2}\frac{\partial}{\partial \sigma}\left(\nu_t^V\frac{\partial v}{\partial \sigma}\right), \tag{3.1b}$$

$$\frac{\partial \zeta}{\partial t} + \frac{\partial HU}{\partial x} + \frac{\partial HV}{\partial y} = 0, \tag{3.1c}$$

$$\frac{\partial \zeta}{\partial t} + \frac{\partial Hu}{\partial x} + \frac{\partial Hv}{\partial y} + \frac{\partial \omega}{\partial \sigma} = 0. \tag{3.1d}$$

3.2 Alternating Direction Implicit Schemes in General

When working on a two-dimensional problem using implicit schemes, usually a large banded matrix equation has to be solved. An implicit numerical scheme for the two-dimensional heat equation,

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \tag{3.2}$$

for example, would look like

$$\frac{u_{m,n}^{p+1} - u_{m,n}^{p}}{\Delta t} = \alpha(\delta_x^2 u_{m,n}^{p+1} + \delta_y^2 u_{m,n}^{p+1})$$

with

$$\delta_x^2 u_{m,n}^{p+1} = (u_{m-1,n}^{p+1} - 2u_{m,n}^{p+1} + u_{m+1,n}^{p+1})/\Delta x^2 \quad \text{and}$$
$$\delta_y^2 u_{m,n}^{p+1} = (u_{m,n-1}^{p+1} - 2u_{m,n}^{p+1} + u_{m,n+1}^{p+1})/\Delta y^2.$$

Solving this equation by Gaussian elimination would be too expensive due to fill-in. The alternating direction implicit (ADI) schemes are a good alternative, because only tridiagonal matrix equations have to be solved. The idea is to use implicit numerical approximations in one spatial direction and explicit ones in the other spatial direction.

We will discuss one of the most common ADI schemes, the Peaceman-Rachford scheme, for the two-dimensional heat equation. This ADI scheme uses a time step of $\Delta t/2$ and the direction in which implicit numerical approximations are used alternates. The scheme reads

$$\frac{u_{m,n}^{p+\frac{1}{2}} - u_{m,n}^{p}}{\Delta t/2} = \alpha \delta_x^2 u_{m,n}^{p+\frac{1}{2}} + \alpha \delta_y^2 u_{m,n}^{p}, \tag{3.3a}$$

$$\frac{u_{m,n}^{p+1} - u_{m,n}^{p+\frac{1}{2}}}{\Delta t/2} = \alpha \delta_x^2 u_{m,n}^{p+\frac{1}{2}} + \alpha \delta_y^2 u_{m,n}^{p+1}. \tag{3.3b}$$

In equation (3.3a) the second derivative with respect to x is evaluated implicitly and the second derivative with respect to y explicitly and vice versa in equation (3.3b). Note that both matrix equations are tridiagonal. From now on we will refer to (3.3) as one iteration (in time), divided into two stages.

Stability

For stability analysis we use a Von Neumann stability method. Let

$$u_{m,n}^{p} = G^p e^{i\theta_x m} e^{i\theta_y n},$$

where i is the imaginary unit, $\theta_x, \theta_y \in [0, 2\pi]$ and $G \in \mathbb{R}$, and substitute this expression in the first stage, equation (3.3a). After cancelling several factors this results in

$$G^{p+\frac{1}{2}} = G^p + \frac{1}{2} r_x G^{p+\frac{1}{2}} [e^{-i\theta_x \Delta x} - 2 + e^{i\theta_x \Delta x}] + \frac{1}{2} r_y G^p [e^{-i\theta_y \Delta y} - 2 + e^{i\theta_y \Delta y}]$$

with $r_x = \alpha \Delta t/\Delta x^2$ and $r_y = \alpha \Delta t/\Delta y^2$. Further manipulation leads to the following relation for the amplification factor G:

$$\sqrt{G} = \frac{1 + r_y[\cos \beta_y - 1]}{1 - r_x[\cos \beta_x - 1]} = \frac{1 - 2r_y \sin^2 \frac{1}{2}\beta_y}{1 + 2r_x \sin^2 \frac{1}{2}\beta_x}$$

with $\beta_x = \theta_x \Delta x$ and $\beta_y = \theta_y \Delta y$. For the second stage the relation for the amplification factor reads

$$\sqrt{G} = \frac{1 + r_x[\cos \beta_x - 1]}{1 - r_y[\cos \beta_y - 1]} = \frac{1 - 2r_x \sin^2 \frac{1}{2}\beta_x}{1 + 2r_y \sin^2 \frac{1}{2}\beta_y}.$$

Thus the amplification factor for the whole time step Δt is

$$G = \frac{(1 - 2r_x \sin^2 \frac{1}{2}\beta_x)(1 - 2r_y \sin^2 \frac{1}{2}\beta_y)}{(1 + 2r_x \sin^2 \frac{1}{2}\beta_x)(1 + 2r_y \sin^2 \frac{1}{2}\beta_y)}.$$

Since r_x and r_y are both positive and the sine terms are at most equal to one, the amplification factor is always smaller than one. Hence this ADI scheme is unconditionally stable. Note that each stage separately is conditionally stable with the conditions $r_y \leq 1$ and $r_x \leq 1$, respectively.

Consistency

Next we will discuss the consistency (accuracy) of the scheme. Therefore we rewrite the equations (3.3a) and (3.3b) as

$$(1 - \tfrac{1}{2}\Delta t\, \alpha\, \delta_x^2)u_{m,n}^{p+\frac{1}{2}} = (1 + \tfrac{1}{2}\Delta t\, \alpha\, \delta_y^2)u_{m,n}^{p} \tag{3.4a}$$

$$(1 - \tfrac{1}{2}\Delta t\, \alpha\, \delta_y^2)u_{m,n}^{p+1} = (1 + \tfrac{1}{2}\Delta t\, \alpha\, \delta_x^2)u_{m,n}^{p+\frac{1}{2}}. \tag{3.4b}$$

After multiplication of (3.4a) with $(1 + \frac{1}{2}\Delta t\, \alpha\, \delta_x^2)$ the operators on the left-hand side are commutable. Having done so we substitute (3.4b) in it, resulting in

$$(1 - \tfrac{1}{2}\Delta t\, \alpha\, \delta_x^2)(1 - \tfrac{1}{2}\Delta t\, \alpha\, \delta_y^2)u_{m,n}^{p+1} = (1 + \tfrac{1}{2}\Delta t\, \alpha\, \delta_x^2)(1 + \tfrac{1}{2}\Delta t\, \alpha\, \delta_y^2)u_{m,n}^{p} \tag{3.5}$$

Expansion of the terms in equation (3.5) the ADI scheme is equivalent to

$$\begin{aligned}\frac{u_{m,n}^{p+1} - u_{m,n}^{p}}{\Delta t} =& \frac{\alpha}{2}\frac{\delta_x^2}{\Delta x^2}(u_{m,n}^{p+1} + u_{m,n}^{p}) + \frac{\alpha}{2}\frac{\delta_y^2}{\Delta y^2}(u_{m,n}^{p+1} + u_{m,n}^{p}) \\ & - \frac{\alpha \Delta t}{4}\frac{\delta_x^2}{\Delta x^2}\frac{\delta_y^2}{\Delta y^2}(u_{m,n}^{p+1} - u_{m,n}^{p})\end{aligned} \tag{3.6}$$

Using Taylor series expansion on equation (3.6), we see that

$$\begin{aligned}&\left[\frac{\partial u}{\partial t}\right]_{m,n}^{p} + \frac{1}{2}\Delta t \left[\frac{\partial^2 u}{\partial t^2}\right]_{m,n}^{p} + \mathcal{O}(\Delta t^2) = \\ &\quad \alpha \left[\frac{\partial^2 u}{\partial x^2}\right]_{m,n}^{p} + \tfrac{1}{2}\alpha\Delta t \left[\frac{\partial^3 u}{\partial x^2 \partial t}\right]_{m,n}^{p} + \mathcal{O}(\Delta x^2) + \mathcal{O}(\Delta t \Delta x^2) \\ &\quad + \alpha \left[\frac{\partial^2 v}{\partial x^2}\right]_{m,n}^{p} + \tfrac{1}{2}\alpha\Delta t \left[\frac{\partial^3 v}{\partial y^2 \partial t}\right]_{m,n}^{p} + \mathcal{O}(\Delta y^2) + \mathcal{O}(\Delta t \Delta y^2) \\ &\quad - \tfrac{1}{4}\alpha\Delta t^2 \left[\frac{\partial u}{\partial x^2 \partial y^2 \partial t}\right]_{m,n}^{p} + \mathcal{O}(\Delta t^2 \Delta x^2) + \mathcal{O}(\Delta t^2 \Delta y^2) + \mathcal{O}(\Delta t^3)\end{aligned} \tag{3.7}$$

Using equation (3.2) on (3.7) (twice!) shows that the Peaceman-Rachford scheme is second order accurate in Δt, Δx and Δy. For more details on ADI schemes and the Peaceman-Rachford scheme in particular we refer to Mitchell and Griffiths (1980).

3.3 Grid and Numerical Scheme

This section is largely taken from Stelling (1984). More information on discretisations on staggered grids can be found in Wesseling (2000).

The horizontal grid is the so-called Arakawa C grid, a staggered grid. This means that the variables are arranged on the grid in a special way. The different variables are not calculated at the same physical points in the horizontal plane. Figure 3.1 shows an example of the Arakawa C grid.

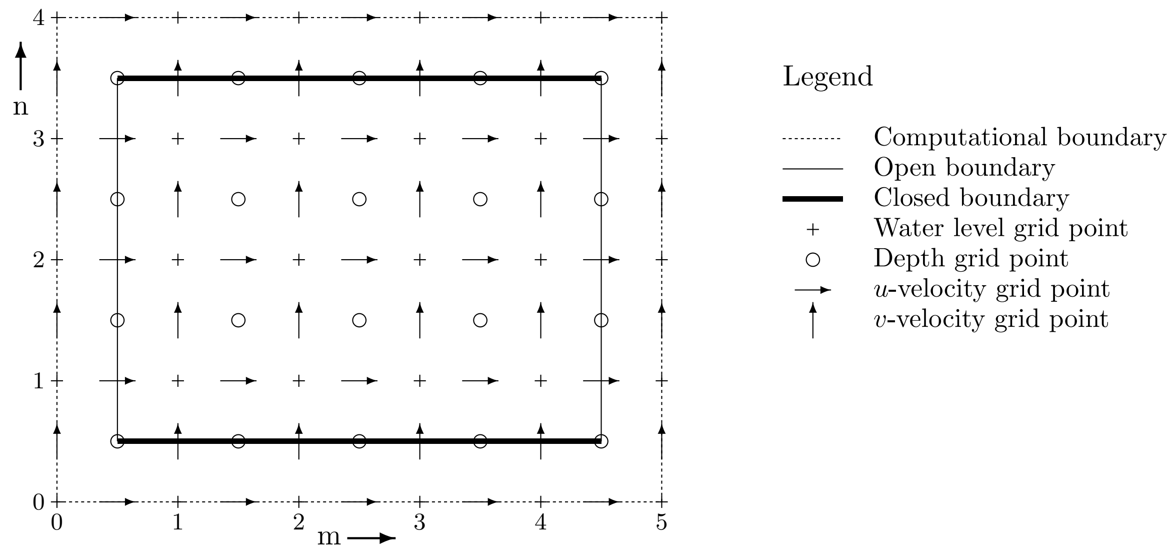


Figure 3.1: An example of the Arakawa C grid.

In general we will refer to the horizontal grid indices with m and n for the x- and y-direction, respectively; and with M and N for their maximum values.

The open and closed boundaries in Figure 3.1 form the model boundary. Only points within this boundary are calculated. For the open boundaries either the normal velocity component at the boundary (e.g. at grid coordinates $(\frac{1}{2}, 2)$) or the water level just outside the boundary (e.g. at $(0, 2)$) should be prescribed.

In the vertical direction we define a number of layers, so-called σ-layers. The number of layers is K. We will use index k to refer to the k^{th} layer with $k = 1$ for the surface layer and $k = K$ for the bottom layer. The discretisation in σ-direction is not necessarily equidistant, so the various layers may differ with respect to their thickness, $\Delta\sigma_k$. Also in the vertical direction a staggered grid is used. Figure 3.2 shows a single vertical grid cell for an arbitrary value of n.

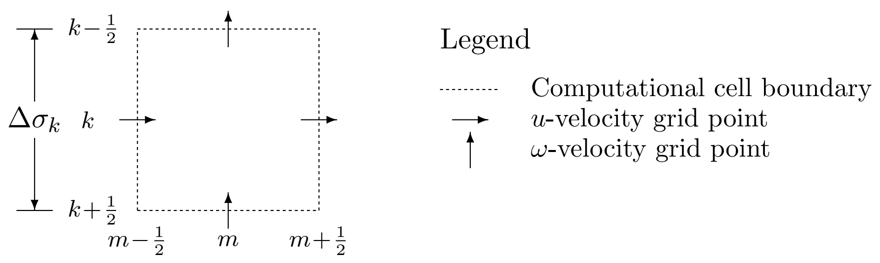


Figure 3.2: A cell from the vertical staggered grid.

The discretisations at the inner points is given term by term in Appendix A. Here we will give the equations (3.1a-d) in approximated discrete form using symbols to represent derivatives.

The first stage reads

$$\begin{aligned}
&\frac{u^{p+\frac{1}{2}}-u^p}{\tau/2}+u^{p+\frac{1}{2}}\,S_{1x}(u^p)+v^{p+\frac{1}{2}}\,S_{1y}(u^p)+\tfrac{\omega^p}{H^p}\,S_\sigma(u^{p+\frac{1}{2}}) \\
&\qquad=-g\,S_{0x}(\zeta^{p+\frac{1}{2}})+\tfrac{1}{(H^p)^2}S_{\sigma\sigma}(u^{p+\frac{1}{2}})+fv^{p+\frac{1}{2}},
\end{aligned} \tag{3.8a}$$

$$\begin{aligned}
&\frac{v^{p+\frac{1}{2}}-v^p}{\tau/2}+u^p\,S_{+x}(v^{p+\frac{1}{2}})+v^p\,S_{+y}(v^{p+\frac{1}{2}})+\tfrac{\omega^p}{H^p}\,S_\sigma(v^{p+\frac{1}{2}}) \\
&=-g\,S_{0y}(\zeta^p)+2\nu_t^H[S_{xx}(v^{p+\frac{1}{2}})+S_{yy}(v^{p+\frac{1}{2}})]+\tfrac{1}{(H^p)^2}S_{\sigma\sigma}(v^{p+\frac{1}{2}})-fu^p,
\end{aligned} \tag{3.8b}$$

$$\frac{\zeta^{p+\frac{1}{2}}-\zeta^p}{\tau/2}+S_{0x}\left(H^{p+\frac{1}{2}}\sum_k \Delta\sigma_k u_k^{p+\frac{1}{2}}\right)+S_{0y}\left(H^p\sum_k \Delta\sigma_k v_k^p\right)=0, \tag{3.8c}$$

$$\frac{\omega_{k-\frac{1}{2}}^{p+\frac{1}{2}}-\omega_{k+\frac{1}{2}}^{p+\frac{1}{2}}}{\Delta\sigma_k}+\frac{\zeta^{p+\frac{1}{2}}-\zeta^p}{\tau/2}+S_{0x}(H^{p+\frac{1}{2}}u_k^{p+\frac{1}{2}})+S_{0y}(H^p v_k^p)=0. \tag{3.8d}$$

The symbols S_{0x} and S_{1x} represent second order accurate central difference schemes of the first derivatives with respect to x over respectively one and two grid cells. Thus

$$S_{0x}(\zeta^p)=(\zeta^p_{m+\frac{1}{2},n}-\zeta^p_{m-\frac{1}{2},n})/\Delta x \quad \text{and} \quad S_{1x}(u^p)=(u^p_{m+1,n}-u^p_{m-1,n})/2\Delta x. \tag{3.9}$$

The second order accurate upwind schemes for the first derivatives are represented by S_{+x} and S_{+y}. For instance

$$S_{+x}(u^p)=\begin{cases}(3u^p_{m,n}-4u^p_{m-1,n}+u^p_{m-2,n})/2\Delta x, & \text{if } u^p_{m,n}>0,\\ (-3u^p_{m,n}+4u^p_{m+1,n}-u^p_{m+2,n})/2\Delta x, & \text{if } u^p_{m,n}\le 0.\end{cases} \tag{3.10}$$

The symbols S_{xx} and S_{yy} are defined at item 12. in the list of approximated terms in Appendix A. And the symbols S_σ and $S_{\sigma\sigma}$ represent the discretisations in items 8. (and 9.) and 14. respectively.

The second stage reads

$$\begin{aligned}
&\frac{u^{p+1}-u^{p+\frac{1}{2}}}{\tau/2}+u^{p+\frac{1}{2}}\,S_{+x}(u^{p+1})+v^{p+\frac{1}{2}}\,S_{+y}(u^{p+1})+\tfrac{\omega^{p+\frac{1}{2}}}{H^{p+\frac{1}{2}}}\,S_\sigma(u^{p+1}) \\
&=-g\,S_{0x}(\zeta^{p+\frac{1}{2}})+2\nu_t^H[S_{xx}(u^{p+1})+S_{yy}(u^{p+1})]+\tfrac{1}{(H^{p+\frac{1}{2}})^2}S_{\sigma\sigma}(u^{p+1})+fv^{p+\frac{1}{2}},
\end{aligned} \tag{3.11a}$$

$$\begin{aligned}
&\frac{v^{p+1}-v^{p+\frac{1}{2}}}{\tau/2}+u^{p+1}\,S_{1x}(v^{p+\frac{1}{2}})+v^{p+1}\,S_{1y}(v^{p+\frac{1}{2}})+\tfrac{\omega^{p+\frac{1}{2}}}{H^{p+\frac{1}{2}}}\,S_\sigma(v^{p+1}) \\
&\qquad=-g\,S_{0y}(\zeta^{p+1})+\tfrac{1}{(H^{p+\frac{1}{2}})^2}S_{\sigma\sigma}(v^{p+1})-fu^{p+1},
\end{aligned} \tag{3.11b}$$

$$\frac{\zeta^{p+1}-\zeta^{p+\frac{1}{2}}}{\tau/2}+S_{0x}\left(H^{p+\frac{1}{2}}\sum_k \Delta\sigma_k u_k^{p+\frac{1}{2}}\right)+S_{0y}\left(H^{p+1}\sum_k \Delta\sigma_k v_k^{p+1}\right)=0, \tag{3.11c}$$

$$\frac{\omega_{k-\frac{1}{2}}^{p+1}-\omega_{k+\frac{1}{2}}^{p+1}}{\Delta\sigma_k}+\frac{\zeta^{p+1}-\zeta^{p+\frac{1}{2}}}{\tau/2}+S_{0x}(H^{p+\frac{1}{2}}u_k^{p+\frac{1}{2}})+S_{0y}(H^{p+1}v_k^{p+1})=0. \tag{3.11d}$$

Note that due to the staggered grid (3.8a) and (3.11a) are calculated at $(m+\frac{1}{2},n,k)$, (3.8b) and (3.11b) at $(m,n+\frac{1}{2},k)$, (3.8c) and (3.11c) at (m,n) and (3.8d) and (3.11d) at (m,n,k).

Further on in this report we will sometimes refer to the equations (3.8a) and (3.8b) as the implicit and explicit momentum equation, respectively, because the water level ζ is taken into account implicitly and explicitly, respectively. Equation (3.8c) will be referred to as the continuity equation. For the equations (3.11a)-(3.11c) the terms are applied in the same manner.

3.4 Boundary Conditions

Careful treatment of the boundary conditions is essential in order to avoid unwanted numerical phenomena, such as artificial boundary layers, unstable discretisations and numerical diffusion, and to approximate the physical flow as well as possible. In this section we will start with the closed boundary conditions, followed by those for the open boundaries and finally we will deal with the boundary conditions at the bottom and the free surface. As an example the grid in Figure 3.1 is used. So, the closed boundaries are parallel to the x-axis and the open boundaries are parallel to the y-axis.

Closed boundary conditions

For calculations on large scale problems, i.e. wide rivers and shallow seas, the impermeability of quays, dykes and dunes at the closed boundaries and the slip condition along these boundaries are supposed to have little effect on the velocities at the inner points. For several terms in the discretised equations the boundary conditions at the closed boundaries are not explicitly used, but the numerical approximations at points close to the boundaries are changed in accordance with them. A more detailed description of these changes is given below. In Figure 3.3 the horizontal grid along a closed boundary is given.

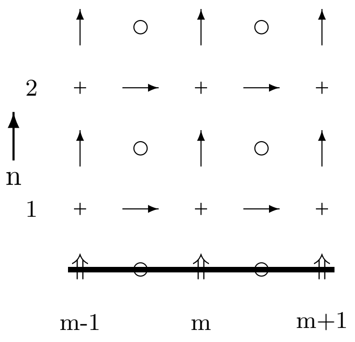


Figure 3.3: Horizontal grid along a closed boundary.

The horizontal advection term $v\frac{\partial v}{\partial y}$ is in practical sense not a large term near a closed boundary. However, when using the numerical discretisation for the derivative (as given in the previous section, item 5. in Appendix A) it is possible that the numerical derivative is much larger. This is due to the fact that the numerical derivative is computed over two grid cells, of which the length, $2\Delta y$, is larger than the scale on which the boundary condition $v = 0$ actually has effect. This insight supports the choice to adjust the numerical approximations near the boundaries rather than explicitly use the boundary conditions at closed boundaries. Stelling (1984) proposed the following approximation for both stages:

$$v\frac{\partial v}{\partial y}\bigg|_{m,1\frac{1}{2}} \approx \begin{cases} 0, & \text{if} \quad v_{m,1\frac{1}{2}} > 0, \\ v_{m,1\frac{1}{2}}(v_{m,2\frac{1}{2}} - v_{m,1\frac{1}{2}})/\Delta y, & \text{if} \quad v_{m,1\frac{1}{2}} \leq 0. \end{cases}$$

Note that in the first case the approximation is zeroth order and in the second case it is first order accurate. The horizontal advection term $v\frac{\partial v}{\partial y}$ at $(m, 2\frac{1}{2})$ is also treated in a special way in the first stage, when $v_{m,2\frac{1}{2}} > 0$. The numerical approximation is

$$v\frac{\partial v}{\partial y}\bigg|_{m,2\frac{1}{2}} \approx \begin{cases} v_{m,2\frac{1}{2}}(v_{m,2\frac{1}{2}} - v_{m,1\frac{1}{2}})/\Delta y, & \text{if} \quad v_{m,2\frac{1}{2}} > 0, \\ v_{m,2\frac{1}{2}}(-3v_{m,2\frac{1}{2}} + 4v_{m,3\frac{1}{2}} - v_{m,4\frac{1}{2}})/2\Delta y, & \text{if} \quad v_{m,2\frac{1}{2}} \leq 0. \end{cases}$$

In large scale problems the use of the free slip condition is allowed. For the horizontal advection term $v\frac{\partial u}{\partial y}$ the free slip condition, $\frac{\partial u}{\partial y} = 0$, and the impermeability condition, $v = 0$, support the following approximations for the first stage:

$$v\frac{\partial u}{\partial y} \approx 0$$

and for the second stage:

$$v\frac{\partial u}{\partial y}\bigg|_{m+\frac{1}{2},1} \approx \begin{cases} 0, & \text{if} \quad \bar{v}_{m+\frac{1}{2},1} > 0, \\ \bar{v}_{m+\frac{1}{2},1}(-3u_{m+\frac{1}{2},1} + 4u_{m+\frac{1}{2},2} - u_{m+\frac{1}{2},3})/2\Delta y, & \text{if} \quad \bar{v}_{m+\frac{1}{2},1} \leq 0. \end{cases}$$

Note that the overbar on v indicates an averaged value as defined in item 6. in Appendix A. In the second stage the numerical approximation at $(m + \frac{1}{2}, 2)$ is replaced with:

$$v\frac{\partial u}{\partial y}\bigg|_{m+\frac{1}{2},2} \approx \begin{cases} \bar{v}_{m+\frac{1}{2},2}(u_{m+\frac{1}{2},2} - u_{m+\frac{1}{2},1})/\Delta y, & \text{if} \quad \bar{v}_{m+\frac{1}{2},2} > 0, \\ \bar{v}_{m+\frac{1}{2},2}(-3u_{m+\frac{1}{2},2} + 4u_{m+\frac{1}{2},3} - u_{m+\frac{1}{2},4})/2\Delta y, & \text{if} \quad \bar{v}_{m+\frac{1}{2},2} \leq 0. \end{cases}$$

The second order derivatives, $\nu^H\frac{\partial^2 u}{\partial y^2}$ and $\nu^H\frac{\partial^2 v}{\partial y^2}$, in the horizontal viscosity terms are approximated as follows near the closed the boundary:

$$\nu^H\frac{\partial^2 u}{\partial y^2}\bigg|_{m+\frac{1}{2},1} \approx \begin{cases} 0 & \text{in the first stage,} \\ \nu^H_{m+\frac{1}{2},1}(u_{m+\frac{1}{2},2} - u_{m+\frac{1}{2},1})/\Delta y^2 & \text{in the second stage,} \end{cases}$$

$$\nu^H\frac{\partial^2 v}{\partial y^2}\bigg|_{m,1\frac{1}{2}} \approx \begin{cases} \nu^H_{m,1\frac{1}{2}}(v_{m,\frac{1}{2}} - 2v_{m,1\frac{1}{2}} + v_{m,2\frac{1}{2}})/\Delta y^2 & \text{in the first stage,} \\ 0 & \text{in the second stage.} \end{cases}$$

Note that in the second expression the velocity component at the closed boundary ($v_{m,\frac{1}{2}}$) is explicitly taken into account, while this is avoided in the first expression. The second order derivatives with respect to x, $\nu^H\frac{\partial^2 u}{\partial x^2}$ and $\nu^H\frac{\partial^2 v}{\partial x^2}$, are approximated with second order accurate central difference schemes.

Boundary conditions on ζ at closed boundaries are only given if necessary. The derivative of ζ with respect to x in equations (3.8a) and (3.11a) does not imply the need for a boundary condition on ζ, because the closed boundaries are parallel to the x-axis. The derivative of ζ with respect to y, however, in the horizontal momentum equation in y-direction requires a boundary condition on ζ, but only in the second stage. It is obvious that the equations (3.11b) and (3.11c) are coupled. To solve these equations (3.11b) is substituted in (3.11c) leading to a tridiagonal system for ζ (see Section 3.5). Points near the closed boundary require data of points just beyond the boundary. The boundary condition $\frac{\partial \zeta}{\partial y} = 0$ satisfies this need.

Open boundary conditions

We recognise two types of open boundaries, namely inflow and outflow boundaries. At outflow boundaries weakly-reflective boundary conditions are preferred. For now, however, we will only use velocity and water level boundary conditions at inflow as well as outflow boundaries. At open boundaries only one type of boundary condition should be given. In Figure 3.4 the horizontal grid along an open boundary is given for both types of boundary conditions.

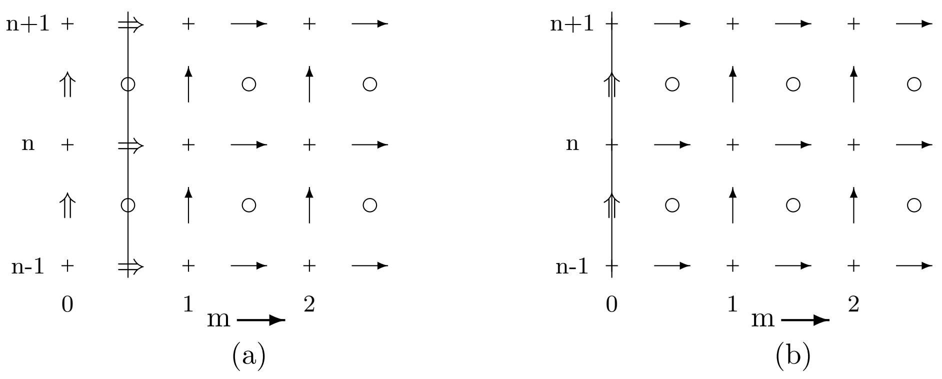


Figure 3.4: The horizontal grid along an open boundary for a velocity boundary condition (a) and for a water level boundary condition (b).

First we will discuss the velocity boundary condition. A velocity boundary condition can be a complete two-dimensional velocity profile or it can consist of a condition on the depth-integrated velocity along the border in combination with a vertical velocity profile. This vertical velocity profile is usually a logarithmic or uniform profile.

At inner points the horizontal advection term $u\frac{\partial u}{\partial x}$ is approximated with a second order central difference scheme in the first stage. Near an open velocity boundary as in Figure 3.4a this approximation does not cause any problems, so it remains unchanged. In the second stage this term is approximated as follows:

$$u\frac{\partial u}{\partial x}\bigg|_{1\frac{1}{2},n} \approx \begin{cases} u_{1\frac{1}{2},n}(u_{1\frac{1}{2},n} - u_{\frac{1}{2},n})/\Delta x, & \text{if} \quad u_{1\frac{1}{2},n} > 0, \\ u_{1\frac{1}{2},n}(-3u_{1\frac{1}{2},n} + 4u_{2\frac{1}{2},n} - u_{3\frac{1}{2},n})/2\Delta x, & \text{if} \quad u_{1\frac{1}{2},n} \leq 0. \end{cases}$$

At open boundaries the tangential velocity component is assumed to be zero. This corresponds with $v_{0,n} = 0 \ \forall n$ for both types of boundary condition in Figure 3.4. The numerical approximation of the horizontal advection term $u\frac{\partial v}{\partial x}$ near open velocity boundaries for the first stage is:

$$u\frac{\partial v}{\partial x}\bigg|_{1,n+\frac{1}{2}} \approx \begin{cases} \bar{u}_{1,n+\frac{1}{2}}(v_{2,n+\frac{1}{2}} - v_{0,n+\frac{1}{2}})/2\Delta x, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} > 0, \\ 0, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} \leq 0 \end{cases}$$

and for the second stage:

$$u\frac{\partial v}{\partial x}\bigg|_{1,n+\frac{1}{2}} \approx \begin{cases} \bar{u}_{1,n+\frac{1}{2}}(v_{1,n+\frac{1}{2}} - v_{0,n+\frac{1}{2}})/\Delta x, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} > 0 \\ \bar{u}_{1,n+\frac{1}{2}}(-3v_{1,n+\frac{1}{2}} + 4v_{2,n+\frac{1}{2}} - v_{3,n+\frac{1}{2}})/2\Delta x, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} \leq 0. \end{cases}$$

The numerical approximations of the horizontal viscosity terms do not need a special treatment near open velocity boundaries and therefore they remain unchanged. For the same reasons as near closed boundaries a boundary condition on ζ is only given when necessary. In the first stage the boundary condition $\frac{\partial \zeta}{\partial x} = 0$ is used.

When a water level boundary condition is prescribed the numerical boundary is set along the ζ grid points (see Figure 3.4b). The numerical approximations near the open boundary of the various terms are different than with a velocity boundary condition. The numerical approximation of the horizontal advection term $u\frac{\partial u}{\partial x}$ at $(\frac{1}{2}, n)$ is replaced with

$$u\frac{\partial u}{\partial x}\bigg|_{\frac{1}{2},n} \approx \begin{cases} 0, & \text{if} \quad u_{\frac{1}{2},n} > 0, \\ u_{\frac{1}{2},n}(u_{1\frac{1}{2},n} - u_{\frac{1}{2},n})/\Delta x, & \text{if} \quad u_{\frac{1}{2},n} \leq 0 \end{cases}$$

for both stages. The horizontal advection term $u\frac{\partial v}{\partial x}$ is discretised near the open boundary according to a second order central scheme in the first stage:

$$u\frac{\partial v}{\partial x}\bigg|_{1,n+\frac{1}{2}} \approx \bar{u}_{1,n+\frac{1}{2}}(v_{2,n+\frac{1}{2}} - v_{0,n+\frac{1}{2}})/2\Delta x.$$

For the second stage either a zeroth order approximation or second order upwind scheme is used, depending on the flow direction of u. The approximation reads

$$u\frac{\partial v}{\partial x}\bigg|_{1,n+\frac{1}{2}} \approx \begin{cases} 0, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} > 0, \\ \bar{u}_{1,n+\frac{1}{2}}(-3v_{1,n+\frac{1}{2}} + 4v_{2,n+\frac{1}{2}} - v_{3,n+\frac{1}{2}})/2\Delta x, & \text{if} \quad \bar{u}_{1,n+\frac{1}{2}} \leq 0. \end{cases}$$

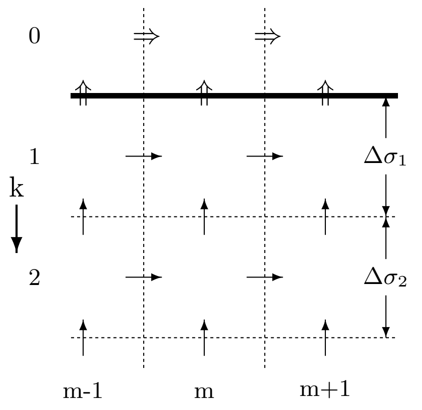
The horizontal viscosity operator for u, $\nu^H \Delta u$, in the second stage is approximated as follows:

$$\nu^H \Delta u\big|_{1\frac{1}{2},n} \approx \nu^H_{1\frac{1}{2},n}(u_{1\frac{1}{2},n-1} - 2u_{1\frac{1}{2},n} + u_{1\frac{1}{2},n+1})/\Delta y^2.$$

The horizontal viscosity terms for v in the first stage are not treated in a special way. In both directions the second order difference schemes are used.

Bottom and surface boundary conditions

At the bottom as well as at the free surface three boundary conditions should be given for each of the velocity components. It is obvious that due to the impermeability of the bottom and the free surface the relative vertical velocity component ω is zero at these boundaries. Bottom friction and wind stresses at the free surface will provide the remaining boundary conditions. In Figure 3.5 the vertical grid along the free surface is given.



The arrow ($\Uparrow$) indicates a Dirichlet boundary condition on the relative vertical velocity component. The other arrow ($\Rightarrow$) indicates a Neumann boundary condition on the horizontal velocity component.

Figure 3.5: Vertical grid along the free surface.

The Neumann boundary conditions for the wind stresses (see eqns. (2.24) and (2.25)) are only applied to the vertical viscosity terms $\frac{1}{H^2}\frac{\partial}{\partial\sigma}(\nu_t^V\frac{\partial u}{\partial\sigma})$ and $\frac{1}{H^2}\frac{\partial}{\partial\sigma}(\nu_t^V\frac{\partial v}{\partial\sigma})$ and not to the vertical advection terms $\frac{\omega}{H}\frac{\partial u}{\partial\sigma}$ and $\frac{\omega}{H}\frac{\partial v}{\partial\sigma}$. For the vertical advection terms at the surface first order upwind schemes are used as numerical approximations. For the velocity component in x-direction, u, at $(m+\frac{1}{2},n)$ this reads

$$\frac{\omega}{H}\frac{\partial u}{\partial\sigma}\bigg|_{k=1} \approx \bar{\omega}_1\frac{u_1 - u_2}{\frac{1}{2}(h_1+h_2)}.$$

In the numerical approximation of the vertical viscosity terms two first order derivatives are subtracted (see list item 14 in Appendix A). Along the surface (when $k = 1$) one of these derivatives equals the expression for wind stress, which accordingly replaces that derivative. The vertical viscosity term for u at the surface becomes

$$\frac{1}{H^2}\frac{\partial}{\partial\sigma}\left(\nu_t^V\frac{\partial u}{\partial\sigma}\right)\bigg|_{k=1} \approx \frac{1}{h_1}\left[\frac{|\vec{\tau}_s|\cos\theta}{\rho} - \nu^V_{1\frac{1}{2}}\frac{u_1-u_2}{\frac{1}{2}(h_1+h_2)}\right].$$

The boundary conditions at the bottom (eqns. (2.22) and (2.23)) are treated equivalently.

3.5 Solution Procedure

In this section we will explain the procedure of solving the discrete three-dimensional shallow-water equations. Only the first stage (3.8) will be treated, since the procedure is equivalent for both stages (3.8) and (3.11). In this section we will adjust the indices for u, v and ω and only use whole numbers, so $v_{m,n}$ corresponds with $v_{m,n+\frac{1}{2}}$ and ω_k with $\omega_{k+\frac{1}{2}}$ for example. As a reminder we note that the number of grid cells in x-, y- and σ-direction are M, N and K, respectively.

Explicit Momentum Equation

First equation (3.8b), the explicit momentum equation in y-direction, will be solved. It is called explicit, because the water level ζ is taken explicit in this equation. However, the horizontal velocity v has been taken implicitly in all the terms of this equation. This results in a large banded matrix equation for $\underline{v}$. The vector $\underline{v}$ is arranged in respectively the vertical and the horizontal (first in y- then in x-direction) directions. The resulting matrix is shown in Appendix B.

The band-width of the matrix is too large to use a direct method. Therefore a Gauss-Jacobi iterative scheme is used in horizontal direction to solve this equation. In the vertical direction the implicit values are maintained (i.e. only the coefficients in T_i remain), resulting in a tridiagonal matrix equation. A single iteration is computed according to a red/black update. The red/black pattern is 'placed' horizontally. If we divide the vector $\underline{v}$ into $\underline{v}_1$, $\underline{v}_2$,..., $\underline{v}_{M\cdot N}$, then the first (red) of the two matrix equations to be solved reads

$$\begin{pmatrix} T_1 & & & & \\ & T_3 & & & \\ & & \ddots & & \\ & & & T_{M\cdot N-3} & \\ & & & & T_{M\cdot N-1} \end{pmatrix} \begin{pmatrix} \underline{v}_1 \\ \underline{v}_3 \\ \vdots \\ \underline{v}_{M\cdot N-3} \\ \underline{v}_{M\cdot N-1} \end{pmatrix} = \underline{f}_1,$$

where $\underline{f}_1$ is the right-hand side including all the variables, which are taken explicitly. The other (black) matrix equation is similar. The right-hand side for this equation, however, is constructed using the latest calculated values for $\underline{v}_1, \underline{v}_3, ..., \underline{v}_{M\cdot N-1}$.

Combining both matrix equations (red and black), while taking in account the last remark, results in a matrix equation with a matrix of the form:

$$\begin{pmatrix} T_1 & \emptyset & \emptyset & \dots & \emptyset & \emptyset & \emptyset & \dots & & & \\ D_{21}^{(-1y)} & T_2 & D_{23}^{(1y)} & \emptyset & \dots & \emptyset & D_{2,N+2}^{(1x)} & \emptyset & \dots & & \\ \emptyset & \emptyset & T_3 & \emptyset & \emptyset & \dots & \emptyset & \emptyset & \emptyset & \dots & \\ \emptyset & D_{42}^{(-2y)} & D_{43}^{(-1y)} & T_4 & D_{45}^{(1y)} & \emptyset & \dots & \emptyset & D_{4,N+4}^{(1x)} & \emptyset & \dots \\ \emptyset & \emptyset & \ddots & \ddots & \ddots & \ddots & \ddots & & & \ddots & \end{pmatrix}$$

If the red/black pattern would be placed differently or if the main flow would be different, then this matrix would also be different. However, it is not expected that the result of the iteration would differ that much.

Implicit Momentum Equation

The equations (3.8a) and (3.8c) are coupled. In order to solve these equations, (3.8a) is solved with respect to $u^{p+\frac{1}{2}}$, after which the results is substituted into (3.8c). In general terms the discrete momentum equation in x-direction (3.8a) can be written as

$$A\underline{u}^{p+\frac{1}{2}} + B\underline{\zeta}^{p+\frac{1}{2}} = \underline{f}^{p}, \tag{3.12}$$

where $A \in \mathbb{R}^{MNK \times MNK}$ is a tridiagonal matrix consisting of MN smaller tridiagonal matrices $A_i \in \mathbb{R}^{K \times K}$. The matrix A_i contains the coefficients for $u_{m,n,k}$ $\forall k$ with $i = (1-m)N + n$. The matrix $B \in \mathbb{R}^{MNK \times (M+1)N}$ can be written as

$$B = \frac{g}{\Delta x}\begin{pmatrix} C & & & & \\ & C & & \emptyset & \\ & & \ddots & & \\ & \emptyset & & C & \\ & & & & C \end{pmatrix} \quad \text{with} \quad C = \begin{pmatrix} -\underline{1} & \underline{1} & & & & \\ & -\underline{1} & \underline{1} & & \emptyset & \\ & & -\underline{1} & \underline{1} & & \\ & \emptyset & & \ddots & \ddots & \\ & & & & -\underline{1} & \underline{1} \end{pmatrix},$$

where $\underline{1} \in \mathbb{R}^{K}$ is the vector with only ones. The vector $\underline{f}^{p}$ contains the remaining terms with only explicit variables and boundary conditions on $\underline{u}$. Due to the staggered grid the matrix C (and hence also B) is not square. Performing a Gaussian elimination process on (3.12) leads to

$$\underline{u}^{p+\frac{1}{2}} + B'\underline{\zeta}^{p+\frac{1}{2}} = \underline{f}'^{p}. \tag{3.13}$$

This procedure does not cause any fill-in in the matrix B, due to the fact that the matrix A consist of smaller tridiagonal matrices A_i. The depth-averaged form of this expression is needed in the continuity equation (3.8c) and can easily be calculated by multiplying every row with the respective relative layer thickness $\Delta\sigma_k$ and adding subsequent rows. This result in

$$\underline{U}^{p+\frac{1}{2}} + B''\underline{\zeta}^{p+\frac{1}{2}} = \underline{f}''^{p} \tag{3.14}$$

with $B'' \in \mathbb{R}^{MN \times (M+1)N}$ and $\underline{U} = \sum_k \Delta\sigma_k u_k$, the depth-averaged velocity component in x-direction.

Substituting this expression for $\underline{U}^{p+\frac{1}{2}}$ and the horizontal velocities in y-direction, $\underline{V}^{p}$, in (3.8c) leads to a tridiagonal system for $\underline{\zeta}^{p+\frac{1}{2}}$ in x-direction. Substituting the result of this equation into (3.13), the expression for $\underline{u}^{p+\frac{1}{2}}$, gives the horizontal velocities in x-direction.

Continuity Equation

The depth-integrated continuity equation (3.8c) describes the level of the water surface ζ in the water column at coordinates (m, n) by calculating the net inflow in that column. This guarantees conservation of mass. It is important for this calculation that the term HU is taken either completely implicit or completely explicit (the same restriction applies to HV). When this is not the case the numerical approximation is not mass conservative. For flow calculations this phenomenon is not so severe. However, when using non-mass-conservative calculated flow data in transport equations for a pollutant, for instance, some of the pollutant may disappear or appear. This is highly unwanted.

Equation (3.8c) is non-linear with respect to ζ, because both factors of HU are implicit. The total water depth H equals $\zeta + d$ and U is given by (3.14). A way to deal with this problem is to use an iterative scheme for $\underline{\zeta}^{p+\frac{1}{2}}$ (Kester and Stelling, 1992). For this scheme the barotropic term $-g\, S_x(\zeta^{p+\frac{1}{2}})$ in equation (3.8a) is multiplied by $H^{p+\frac{1}{2},i}/H^{p+\frac{1}{2},i+1}$ (with $H^{p+\frac{1}{2},0} := H^p$). With this modification the equations (3.8a) and (3.8c) read respectively

$$\begin{aligned}\frac{u^{p+\frac{1}{2}} - u^p}{\tau/2} &+ u^{p+\frac{1}{2}}\, S_x(u^p) + v^{p+\frac{1}{2}}\, S_y(u^p) + \omega^p\, S_\sigma(u^{p+\frac{1}{2}}) \\ &= -g\, \frac{H^{p+\frac{1}{2},i}}{H^{p+\frac{1}{2},i+1}}\, S_x(\zeta^{p+\frac{1}{2},i+1}) + S_{\sigma\sigma}(u^{p+\frac{1}{2}}) + f v^{p+\frac{1}{2}},\end{aligned} \tag{3.15a}$$

$$\frac{\zeta^{p+\frac{1}{2},i+1} - \zeta^p}{\tau/2} + S_x\left(H^{p+\frac{1}{2},i+1} \sum_k \Delta\sigma_k u_k^{p+\frac{1}{2}}\right) + S_y\left(H^p \sum_k \Delta\sigma_k v_k^p\right) = 0, \tag{3.15b}$$

where i denotes the current iterate. After the substitution of (3.15a) in (3.15b) as described above, a linear tridiagonal iterative system for $\underline{\zeta}^{p+\frac{1}{2}}$ arises, of which the result is substituted in (3.15a). By default the number of iterations is set to two, what practically proves to be sufficient.

Finally, the continuity equation (3.8d) can be solved by substitution of the previous computed results. This equation is only used to derive the relative vertical velocity components, $\omega_{m,n,k}$. Note that at a horizontal coordinate (m, n) there are K continuity equations for $K - 1$ unknown vertical velocity components. Therefore one of the equations is used for verification of the calculations.

Chapter 4

Rounding Error Analysis

When using a program like Delft3D-FLOW one would like to acquire the most accurate results possible. However, computers can only calculate with a certain finite accuracy, the machine precision. The cause of this finite accuracy is that after every floating point operation the result is rounded, causing a small rounding error. Due to accumulation of these rounding errors the final result is less accurate than the machine precision. Delft3D-FLOW calculates in single precision, meaning that after a single floating point operation the result is rounded after the seventh digit. To what extent do these rounding errors influence the accuracy of the solution? This depends on the solution algorithm and the relation between the coefficients of the matrices (the condition number). Truncation errors are not considered in this report, although they are usually larger than rounding errors.

In this chapter we will start with the applied solution algorithms, the LU factorisation (Gaussian elimination) and the red-black Gauss-Jacobi block iterative method. Condition numbers will be discussed next after which we arrive at the section on accuracy. Subsequently, we will focus on the Gauss-Jacobi iterative method and possible stopping criteria. We end with a short discourse on possible enhancements regarding the accuracy.

4.1 The LU Factorisation

The tridiagonal matrix equations, (3.8a) with respect to $\underline{u}^{p+\frac{1}{2}}$ and (3.8c) with respect to $\underline{\zeta}^{p+\frac{1}{2}}$, in Delft3D-FLOW are solved using Gaussian elimination. Using an LU factorisation to solve the matrix equations is mathematically the same procedure. Therefore we can use the analysis of the LU factorisation for the solution algorithm in Delft3D-FLOW.

The idea behind Gaussian elimination is to "solve" the lower part of the matrix equation ($A\underline{x} = \underline{b}$) first and then the upper part (note that this might also be done vice versa). Application of the LU factorisation to the matrix A leads to decomposition into two matrices, L and U, of which the product is A. The solution to the matrix equation is now found by solving two triangular matrix equations:

$$L\underline{y} = \underline{b} \quad \text{and} \quad U\underline{x} = \underline{y}.$$

Note that halfway the Gaussian elimination process the remaining matrix equation equals $U\underline{x} = \underline{y}$. So, the main difference with the LU factorisation is that L is not explicitly determined.

In the next section we will analyse the effect of rounding errors during the LU factorisation. In the second section we will treat pivoting and its effect on the error analysis. Finally, we will analyse the effect of rounding errors for the specific case in which tridiagonal matrices are used.

Error Analysis for General Matrices

We will analyse the LU factorisation if applied on a computer with machine precision μ. The following lemma gives an idea of how accurate the LU factorisation can be performed by providing an upper bound for the "residue" of the decomposition: $(\hat{L}\hat{U} - A)$.

Lemma 4.1. *Let A be a real $n \times n$ floating point matrix. If all pivots are nonzero, then the calculated triangular matrices $\hat{L}$ and $\hat{U}$ satisfy*

$$\hat{L}\hat{U} = A + H \quad \text{with} \quad |H| \leq 2(n-1)\mu \left\{ |A| + |\hat{L}||\hat{U}| \right\} + \mathcal{O}(\mu^2),$$

where $H \leq A$ means $h_{ij} \leq a_{ij}\ \forall i, j$.

Proof. See Golub and Van Loan (1989, p. 105-6). □

Solving the matrix equation using L and U is also subject to rounding errors as the next lemma will show.

Lemma 4.2. *Let L be a real lower triangular $n \times n$ floating point matrix. For the computed solution $\hat{\underline{y}}$ of $L\underline{y} = \underline{b}$ we have*

$$(L + F)\hat{\underline{y}} = \underline{b} \quad \text{with} \quad |F| \leq n\mu\, |L| + \mathcal{O}(\mu^2).$$

Proof. See Forsythe and Moler (1967, p. 104-5). □

Thus the computed solution $\hat{\underline{y}}$ satisfies a slightly perturbed system. For $U\underline{x} = \hat{\underline{y}}$ a similar relation holds. Combination of the Lemmas 4.1 and 4.2 provides an idea of the perturbed matrix equation which is actually solved when using an LU factorisation or Gaussian elimination.

Theorem 4.1. *Let $\hat{L}\hat{U}$ be the computed factorisation of the $n \times n$ floating point matrix A, $\hat{\underline{y}}$ the computed solution of $\hat{L}\underline{y} = \underline{b}$ and $\hat{\underline{x}}$ the computed solution of $\hat{U}\underline{x} = \hat{\underline{y}}$. Then $(A + E)\hat{\underline{x}} = \underline{b}$ with*

$$|E| \leq 2n\mu \left\{ |A| + 2|\hat{L}||\hat{U}| \right\} + \mathcal{O}(\mu^2).$$

Pivoting

The previous analysis shows that we must avoid large entries in the computed triangular matrices, $\hat{L}$ and $\hat{U}$. One of the most basic methods to ensure small entries in $\hat{L}$ is pivoting. Before we discuss pivoting we first explain what a pivot is. Therefore we give a more detailed description of the LU factorisation.

During the factorisation of the matrix A to a lower and an upper triangular matrix, A is actually reduced to an upper triangular matrix U through Gaussian elimination. The information of the "inverse" procedure to this reduction is stored in a lower triangular matrix L. After construction of the first $k-1$ columns of L and as many rows of U, the partially constructed L and U read

$$L^{(k-1)} = \begin{pmatrix} 1 & 0 & \dots & 0 & 0 & \dots & 0 & 0 \\ l_{21} & 1 & \ddots & \vdots & \vdots & & & \vdots \\ l_{31} & l_{32} & \ddots & 0 & \vdots & & & \vdots \\ \vdots & & \ddots & 1 & 0 & & & \vdots \\ l_{k,1} & \dots\dots & & l_{k,k-1} & 1 & \ddots & & \vdots \\ \vdots & & & l_{k+1,k-1} & 0 & \ddots & 0 & \vdots \\ \vdots & & & \vdots & \vdots & \ddots & 1 & 0 \\ l_{n1} & \dots\dots & & l_{n,k-1} & 0 & \dots & 0 & 1 \end{pmatrix},$$

$$U^{(k-1)} = \begin{pmatrix} u_{11} & u_{12} & \cdots\cdots & & u_{1,k} & \cdots & u_{1n} \\ 0 & u_{22} & \ddots & & \vdots & & \vdots \\ \vdots & 0 & \ddots & \ddots & \vdots & & \vdots \\ \vdots & & \ddots & u_{k-1,k-1} & u_{k-1,k} & \cdots & u_{k-1,n} \\ \vdots & & & 0 & a^{(k-1)}_{k,k} & \cdots & a^{(k-1)}_{k,n} \\ \vdots & & & \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots\cdots & & 0 & a^{(k-1)}_{n,k} & \cdots & a^{(k-1)}_{n,n} \end{pmatrix}.$$

The entry $a^{(k-1)}_{k,k}$ is called the pivot. If the pivot is larger than all other entries in the same column below the pivot ($a^{(k-1)}_{i,k}$ for $k < i \le n$) in absolute sense then their respective entries in L ($l^{(k-1)}_{i,k}$ for $k < i \le n$) will be at most one.

Pivoting is a technique to ensure small entries in L by interchanging rows (partial pivoting) or rows and columns (complete pivoting). With partial pivoting we change two rows (if necessary) such that the pivot is the largest entry compared to the entries in the same column below the pivot. With complete pivoting we look for the largest entry in the submatrix $U^{(k-1)}_{k:n,k:n}$ and interchange two rows and two columns such that the pivot is the largest entry in the submatrix. Both pivot strategies ensure small entries in L and complete pivoting has some minor advantages with respect to the entries in U.

The main disadvantage of pivoting is that during the LU factorisation we must keep track of all row and column interchanging. Moreover, if we have a tridiagonal matrix (or any other sparse matrix) and we apply a pivoting strategy, we will loose the sparseness during the LU factorisation, causing the need for more memory and computation time. Thus, we would rather avoid pivoting at all and still have small entries in L. The following lemma will show that pivoting is not always necessary.

Lemma 4.3. *Let A be an $n \times n$-matrix. If A^T is diagonally dominant (thus if A is column diagonally dominant) then A has an LU factorisation and $|l_{ij}| \le 1$.*

Proof. See Golub and Van Loan (1989, p. 120). □

Error Analysis for Tridiagonal Matrices

For tridiagonal systems we can improve the results of Lemma 4.2 and Theorem 4.1. Furthermore, we will use the fact that the matrix equations solved in Delft3D-FLOW are practically always column diagonally dominant.

If A is a tridiagonal matrix, then its lower triangular matrix L from the LU factorisation will only have nonzero elements on the main and lower diagonal. In the ensuing lemma this knowledge will be used to improve Lemma 4.2.

Lemma 4.4. *Let L be a real lower triangular $n \times n$ floating point matrix with $l_{ij} = 0$ for $i \ne j, j-1$. For the computed solution $\hat{\underline{y}}$ of $L\underline{y} = \underline{b}$ we have*

$$(L + F)\hat{\underline{y}} = \underline{b} \quad \textit{with} \quad |F| \le 2\mu\,|L| + \mathcal{O}(\mu^2).$$

Proof. The components of the solution $\hat{\underline{y}}$ are computed by

$$\begin{aligned} \hat{y}_1 &:= \mathrm{fl}(b_1/l_{1,1}), \\ \hat{y}_i &:= \mathrm{fl}\left(\frac{\mathrm{fl}(-\mathrm{fl}(l_{i,i-1}\hat{y}_{i-1}) + b_i)}{l_{i,i}}\right) \quad \text{for} \quad i = 2, .., n, \end{aligned} \tag{4.1}$$

where fl denotes that the result of the operation is rounded to the floating point format. For every floating point operation the following relations hold: $\mathrm{fl}(a \otimes b) = (a \otimes b)(1+\delta)$ and $\mathrm{fl}(a \otimes b)(1+\delta) = (a \otimes b)$ where $|\delta| \leq \mu$ (Forsythe and Moler, 1967, p. 90,97). For the equations in (4.1) this leads to

$$\begin{aligned} \hat{y}_1 &= \frac{b_1}{l_{1,1}(1+\delta_{1,1})}, \\ \hat{y}_i &= \frac{-l_{i,i-1}(1+\delta_{i,i-1})\hat{y}_{i-1} + b_i}{l_{i,i}(1+\delta_{i,i})(1+\delta'_{i,i})} \quad \text{for} \quad i = 2,..,n, \end{aligned}$$

where $\delta'_{i,i}$ is the round-off error due to the addition of b_i and the rest of the numerator. The division causes a round-off error which is represented by $\delta_{i,i}$. These equations can be rewritten to the following form

$$\begin{aligned} l_{1,1}(1+\delta_{1,1})\hat{y}_1 &= b_1, \\ l_{i,i-1}(1+\delta_{i,i-1})\hat{y}_{i-1} + l_{i,i}(1+\delta_{i,i})(1+\delta'_{i,i})\hat{y}_i &= b_i \quad \text{for} \quad i = 2,..,n, \end{aligned}$$

or $(L+F)\underline{\hat{y}} = \underline{b}$ with $|F| \leq 2\mu\,|L| + \mathcal{O}(\mu^2)$. □

For $U\underline{x} = \underline{\hat{y}}$ a similar relation holds.

Finally, we will improve Theorem 4.1 for tridiagonal matrices and we add column diagonal dominance as an extra condition to the matrix.

Theorem 4.2. *Let $\hat{L}\hat{U}$ be the computed factorisation of the $n \times n$ tridiagonal floating point matrix A, where A^T is diagonally dominant, $\underline{\hat{y}}$ is the computed solution of $\hat{L}\underline{y} = \underline{b}$ and $\underline{\hat{x}}$ is the computed solution of $\hat{U}\underline{x} = \underline{\hat{y}}$. Then $(A+E)\underline{\hat{x}} = \underline{b}$ with*

$$|E| \leq 2(n-1)\mu\,|A| + (2n+2)\mu|\hat{U}| + \mathcal{O}(\mu^2).$$

Proof. From Lemma 4.4 we have

$$\begin{aligned} (\hat{L}+F)\underline{\hat{y}} &= \underline{b} \qquad & |F| &\leq 2\mu|\hat{L}| + \mathcal{O}(\mu^2), \\ (\hat{U}+G)\underline{\hat{x}} &= \underline{\hat{y}} \qquad & |G| &\leq 2\mu|\hat{U}| + \mathcal{O}(\mu^2) \end{aligned}$$

and thus

$$(\hat{L}+F)(\hat{U}+G)\underline{\hat{x}} = (\hat{L}\hat{U} + F\hat{U} + \hat{L}G + FG)\underline{\hat{x}} = b.$$

From Lemma 4.1 we have

$$\hat{L}\hat{U} = A + H \quad \text{with} \quad |H| \leq 2(n-1)\mu\left\{|A| + |\hat{L}||\hat{U}|\right\} + \mathcal{O}(\mu^2).$$

So, by defining

$$E = H + F\hat{U} + \hat{L}G + FG$$

and taking into account Lemma 4.3 we find $(A+E)\underline{\hat{x}} = \underline{b}$ with

$$|E| \leq 2(n-1)\mu\,|A| + (2n+2)\mu|\hat{U}| + \mathcal{O}(\mu^2).$$ □

This upper bound will be used in determining an upper bound on the relative error for the tridiagonal systems later on in this chapter. It cannot be used for iterative methods like the Gauss-Jacobi scheme discussed in the Section 4.5.

4.2 Residue

The relative residue is defined by the ratio of the infinity norms of the residual vector to the infinity norms of the right-hand side. So, if $A\underline{x} = \underline{b}$ is the matrix equation to be solved and the residual vector is defined by $\underline{r} := A\underline{\hat{x}} - \underline{b}$, then the relative residue would be

$$r = \frac{\|\underline{r}\|_\infty}{\|\underline{b}\|_\infty} = \frac{\|A\underline{\hat{x}} - \underline{b}\|_\infty}{\|\underline{b}\|_\infty},$$

where $\underline{\hat{x}}$ is the computed solution vector. Note that a small residue does not imply a small relative error. However, there is a relation between the residue and the relative error as we will see later on.

4.3 Condition Number

The condition number of the matrix is one of the most important parameters in the accuracy analysis of any matrix equation. For every order of magnitude the condition number increases, the estimate of the upper bound on the relative error also increases an order of magnitude. It should be noted, however, that the condition number represents the worst case scenario for the relative error.

The condition number κ_2 of an $n \times n$-matrix A is defined by the ratio of the largest to the smallest singular value, so we have $\kappa_2(A) = \|A\|_2\|A^{-1}\|_2$. We will use another condition number namely $\kappa_\infty = \|A\|_\infty\|A^{-1}\|_\infty$, because it is more easily computed. These condition numbers are not the same. However, they are equivalent in that it can be shown that

$$\tfrac{1}{n}\kappa_\infty(A) \leq \kappa_2(A) \leq n\kappa_\infty(A).$$

In order to calculate $\kappa_\infty(A)$ we determine $\|A\|_\infty$ and $\|A^{-1}\|_\infty$ separately. The first matrix norm is rather easily computed since we have

$$\|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|,$$

but only if the matrix elements are at our disposal. Within Delft3D-FLOW this is the case with all matrices.

The second matrix norm however is more difficult because we do not know the inverse of the matrix A. Before we proceed we will give a few definitions and lemmas.

Definition 4.1 (K-Matrix). *Let A be an $n \times n$-matrix. A is a K-matrix if A is irreducible, $a_{ii} > 0\ \forall i$, $a_{ij} \leq 0$ for $i \neq j$ and $a_{ii} \geq -\sum_{j \neq i} a_{ij}\ \forall i$, with at least one strict inequality for a certain i.*

Definition 4.2 (M-Matrix). *Let A be an $n \times n$-matrix. A is an M-matrix if A is nonsingular, $a_{ij} \leq 0$ for $i \neq j$ and $A^{-1} \geq 0$.*

Lemma 4.5. *A K-matrix is an M-matrix.*

Proof. See Varga (1962, p. 85). □

Theorem 4.3. *If A is an M-matrix, then $\|A^{-1}\|_\infty$ is equal to the infinity norm of the solution $\underline{y}$ of $A\underline{y} = \underline{d}$, where $\underline{d}$ is the all-one vector.*

Proof. First of all, we note that A^{-1} exists, because A is an M-matrix. Following the definition of matrix norms we have:

$$\|A^{-1}\|_\infty = \max_{\underline{x} \neq \underline{0}} \frac{\|A^{-1}\underline{x}\|_\infty}{\|\underline{x}\|_\infty} = \max_{\|\underline{x}\|_\infty = 1} \|A^{-1}\underline{x}\|_\infty. \tag{4.2}$$

Because A is an M-matrix it follows that $A^{-1} \geq 0$. This implies that the maximal value of the right-hand side of (4.2) is assumed when $\underline{x} = (1, 1, ..., 1)^T$. This corresponds with the infinity norm of the solution $\underline{y}$ of $A\underline{y} = \underline{d}$ where $\underline{d} = (1, 1, ..., 1)^T$. □

In order to calculate the norms of the inverse matrices in Delft3D-FLOW we use the matrix solvers that are already implemented to approach the solution of $A\underline{y} = \underline{d}$ where $\underline{d} = (1, 1, ..., 1)^T$. It seems wrong to use the same matrix solvers to analyse their accuracy. However, we are mainly interested in the order of the condition number rather than the exact figure.

From here on we will drop the ∞-sign in $\|\cdot\|_\infty$.

4.4 Accuracy

We consider the relative error (and to some extent the error itself) of a solution to be the most important quantity in an accuracy analysis. In this section we will construct an upper bound for the relative error of a matrix equation $A\underline{x} = \underline{b}$ using an LU factorisation. Both the implicit momentum equation and the continuity equation are solved using an LU factorisation. Furthermore, we will give a general upper bound which is not dependent on the used matrix solver. We will use this upper bound for all three matrix equations.

When using an LU factorisation we have observed (see Theorem 4.2) that for the computed solution $\hat{\underline{x}}$ we have $(A + E)\hat{\underline{x}} = \underline{b}$ with

$$|E| \leq 2(n-1)\mu\,|A| + (2n+2)\mu|\hat{U}| + \mathcal{O}(\mu^2).$$

Then also $\|E\| \leq 2(n-1)\mu\|A\| + (2n+2)\mu\|\hat{U}\| + \mathcal{O}(\mu^2)$.

Golub and Van Loan (1989, p. 82) present an useful theorem for an upper bound on the relative error.

Theorem 4.4. *Let A and E be real $n \times n$-matrices and $\underline{b}, \underline{f} \in \mathbb{R}^n$ with $\underline{b} \neq 0$ such that*

$$A\underline{x} = \underline{b},$$
$$(A + E)\hat{\underline{x}} = \underline{b} + \underline{f}$$

with $\|E\| \leq \delta\|A\|$ and $\|\underline{f}\| \leq \delta\|\underline{b}\|$. If $\delta\kappa(A) = s < 1$ then $A + E$ is nonsingular and

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \frac{2\delta}{1-s}\kappa(A).$$

Proof. See Golub and Van Loan (1989, p. 82). □

In our case $\underline{f} = \underline{0}$ and we define $\delta := \|E\|/\|A\|$. Applying Theorem 4.4 results in an upper bound for the relative error for the implicit momentum equation and the continuity equation:

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \frac{2\|E\|\kappa(A)}{\|A\| - \|E\|\kappa(A)}. \tag{4.3}$$

Note that we must have $\|E\|\kappa(A) \leq \|A\|$. Practically this seems almost always to be the case.

For the explicit momentum equation we do not have an estimate or upper bound for the perturbation matrix E. Hence, we will derive a more general upper bound for the relative error in the following lemma.

Lemma 4.6. *Let A be a real $n \times n$ matrix, $\underline{b} \in \mathbb{R}^n$ a vector and $\hat{\underline{x}} \in \mathbb{R}^n$ the computed solution of the matrix equation $A\underline{x} = \underline{b}$. Then we have*

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \kappa(A)\frac{\|A\hat{\underline{x}} - \underline{b}\|}{\|\underline{b}\|} = \kappa(A)r,$$

where r is the relative residue as defined in Section 4.2.

Proof. Define the residual vector by $\underline{f} := A\hat{\underline{x}} - \underline{b}$. Then the computed solution $\hat{\underline{x}}$ is the exact solution of the matrix equation $A\hat{\underline{x}} = \underline{b} + \underline{f}$. Subtraction of both matrix equations leads to $A(\hat{\underline{x}} - \underline{x}) = \underline{f}$. Now we can write

$$\begin{cases} A\underline{x} = \underline{b}, \\ A(\hat{\underline{x}} - \underline{x}) = \underline{f} \end{cases} \Rightarrow \begin{cases} \|\underline{b}\| \leq \|A\|\|\underline{x}\|, \\ \hat{\underline{x}} - \underline{x} = A^{-1}\underline{f} \end{cases} \Rightarrow \begin{cases} \frac{1}{\|\underline{x}\|} \leq \|A\|\frac{1}{\|\underline{b}\|}, \\ \|\hat{\underline{x}} - \underline{x}\| \leq \|A^{-1}\|\|\underline{f}\|. \end{cases}$$

Multiplication of the last two relations leads to

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \kappa(A)\frac{\|A\hat{\underline{x}} - \underline{b}\|}{\|\underline{b}\|} = \kappa(A)r.$$

□

We can use this result for every numerical method. From the proof of Lemma 4.6 we can easily derive the following relation.

Corollary 4.1. *Let A be a real $n \times n$ matrix, $\underline{b} \in \mathbb{R}^n$ a vector and $\hat{\underline{x}} \in \mathbb{R}^n$ the computed solution of the matrix equation $A\underline{x} = \underline{b}$. Then we have*

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \|A^{-1}\|\frac{\|\underline{b}\|}{\|\underline{x}\|}\frac{\|A\hat{\underline{x}} - \underline{b}\|}{\|\underline{b}\|} = \|A^{-1}\|\frac{\|\underline{b}\|}{\|\underline{x}\|}r,$$

where r is the relative residue as defined in Section 4.2.

The quantity $\|A^{-1}\|\frac{\|\underline{b}\|}{\|\underline{x}\|}$ is called the *effective* condition number (or condition number of A with respect to the vector $\underline{x}$ as Axelsson (1994, p. 605) calls it) and is always smaller than the condition number $\kappa(A)$.

The relations $\|E\| \leq \mu\|A\|$ and $\|\underline{f}\| \leq \mu\|\underline{b}\|$ hold if the matrix equation $A\underline{x} = \underline{b}$ is solved exactly and if A and $\underline{b}$ are at least subjected to round-off due to machine representation. Thus the "best possible" upper bound for the relative error is

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq 4\mu\kappa(A).$$

4.5 Red-Black Block Gauss-Jacobi Iterative Scheme

The explicit momentum equation (remember that explicit means that the water level ζ is taken explicitly in this equation, but the velocity, either u or v, is taken implicitly) is solved with an iterative method because the band-width of the matrix is too large for practical use of a direct method. The matrix is given in Appendix B. The choice has been made to use a red-black block Gauss-Jacobi iterative scheme as explained in Section 3.5.

Consider the matrix equation $A\underline{x} = \underline{b}$ with

$$A = \begin{pmatrix} D_1 & A_{12} & \dots & A_{1n} \\ A_{21} & D_2 & & A_{2n} \\ \vdots & & \ddots & \vdots \\ A_{n1} & A_{n2} & \dots & D_n \end{pmatrix}.$$

The matrices D_i are not necessarily of the same size, however they are all square. We define $D := diag(D_1, D_2, ..., D_n)$ and $R := D - A$. Now the block Gauss-Jacobi iterative scheme can be written as

$$\begin{aligned} &(1)\ \text{Choose an initial vector } \underline{x}_0, \\ &(2)\ \text{Solve } D\underline{x}_{i+1} = R\underline{x}_i + \underline{b}, \\ &(3)\ \text{Repeat (2) until the stopping criterion is met.} \end{aligned} \tag{4.4}$$

The term red-black refers to the order in which the elements of the vector $\underline{x}_{i+1}$ are computed during step (2) as mentioned in Section 3.5 on page 21.

4.6 Stopping Criteria

Iterative schemes need a stopping criterion to end the iteration process. Such a stopping criterion has to have basically two properties. (i) It has to be satisfied within a finite number of iterations. (ii) It must ensure a sufficient accuracy of the iteration result.

The Gauss-Jacobi iterative scheme implemented in Delft3D-FLOW uses a condition on the difference between consecutive iterands as a stopping criterion. However, analytically it is more sound to use the relative residue. The original stopping criterion reads $\|\underline{x}_n - \underline{x}_{n-1}\| \leq \epsilon_{it}$ (ϵ_{it} is chosen 10^{-6} in Delft3D-FLOW) and when the relative residue is used, it reads

$$\frac{\|A\underline{x}_n - \underline{b}\|}{\|\underline{b}\|} \leq \epsilon_r.$$

The main advantage of the original stopping criterion is that it requires little computation time, especially when compared with the relative residue criterion. It is obvious that we would like to have a reliable stopping criterion as well as little computation time.

In this respect the rate of convergence is an important property of the iteration process. Suppose we want to solve a large-banded matrix equation $A\underline{x} = \underline{b}$ by means of an iterative scheme. This scheme might look like $D\underline{x}_{i+1} = R\underline{x}_i + \underline{b}$ with $A = D - R$. A single iteration satisfies $\underline{x}_{i+1} = D^{-1}R\underline{x}_i + D^{-1}\underline{b}$. Subtracting the same expression for the exact solution $\underline{x}$ leaves us

$$\underline{x}_{i+1} - \underline{x} = D^{-1}R(\underline{x}_i - \underline{x}) = (D^{-1}R)^{i+1}(\underline{x}_0 - \underline{x})$$

with $\underline{x}_0$ as the initial vector. So, the matrix $B := D^{-1}R$ determines the rate of convergence.

Definition 4.3. *Let B be the iteration matrix, $B = D^{-1}R$. Then $\|B^i\|$ is called the convergence factor for i steps and $R_i = \|B^i\|^{1/i}$ is called the average convergence factor (per step for i steps). (See also Axelsson (1994, p. 163))*

Usually B is not explicitly available, which makes it difficult to calculate the convergence factor. It can be shown that the average convergence factor approaches the spectral radius of the iteration matrix. Thus,

$$\|B^i\|^{1/i} \to \rho(B), \quad i \to \infty.$$

However, also the spectral radius is not easily determined. Therefore we analyse a more easily calculated quantity, namely the contraction factor, which is defined with the relation

$$\rho_c(i) = \frac{\|\underline{x}_{i+1} - \underline{x}_i\|}{\|\underline{x}_i - \underline{x}_{i-1}\|}. \tag{4.5}$$

The contraction factor is more or less constant during one iterative process. However, in practice the first and sometimes the last contraction factor are relatively large, see Figure 4.1. The latter might be large, because the result of the iteration process is very close to machine precision.

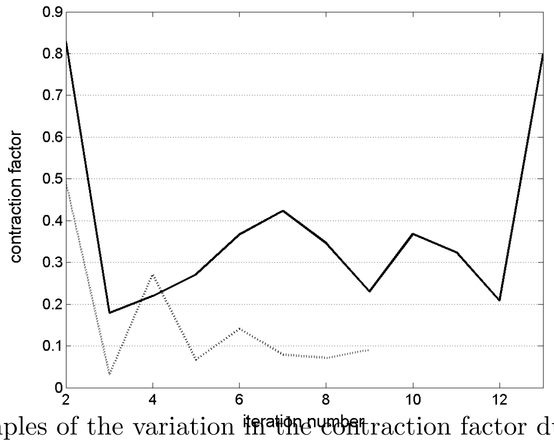


Figure 4.1: Two examples of the variation in the contraction factor during the iterative process.

To formulate an useful stopping criterion we need to know how large the iteration errors will be. The following theorem will help us with that.

Theorem 4.5. *Let $B = D^{-1}R$, $\|B\| < 1$, and $\underline{x}_i$ be defined by (4.4). Then*

$$\|\underline{x} - \underline{x}_i\| \leq \frac{\|B\|}{1 - \|B\|} \|\underline{x}_i - \underline{x}_{i-1}\|, \quad i = 1, 2, ...$$

Proof. See Axelsson (1994, p.166). □

So, the original stopping criterion, $\|\underline{x}_i - \underline{x}_{i-1}\| \leq \epsilon_{it}$, has the following iteration error, if $\|B\| < 1$,

$$\|\underline{x} - \underline{x}_i\| \leq \tfrac{\|B\|}{1-\|B\|} \epsilon_{it}.$$

If we want a relative error $\|\underline{x} - \underline{x}_i\|/\|\underline{x}\| \leq \delta$, then we have to choose

$$\epsilon_{it} \leq \tfrac{1-\|B\|}{\|B\|} \delta \|\underline{x}\|.$$

In practice $\|B\|$ is usually close to 1, which leads to a much smaller value for ϵ_{it} than any given δ. Also, if ϵ_{it} is chosen equal to δ we run the risk of having a large iteration error. The calculation of $\|B\|$ is expensive and the exact solution $\underline{x}$ is unknown. Therefore we will use the approximation ρ_c for the convergence factor. The iterative scheme in Delft3D-FLOW calculates velocities, which are usually $\mathcal{O}(1)$ in the main flow direction. Thus we can estimate $\|\underline{x}\|$ with 1.

For the relative residue stopping criterion, $\|\underline{r}_i\|/\|\underline{b}\| \leq \epsilon_r$, the iteration error is bounded by the relation

$$\|\underline{x} - \underline{x}_i\| = \|A^{-1}\underline{r}_i\| \leq \|A^{-1}\| \cdot \|b\|\, \epsilon_r,$$

where $\underline{r}_i := A\underline{x}_i - \underline{b}$ is the residual vector. If we want a relative error $\|\underline{x} - \underline{x}_i\|/\|\underline{x}\| \leq \delta$, then we have to choose

$$\epsilon_r \leq \frac{\delta\|\underline{x}\|}{\|A^{-1}\|\|\underline{b}\|} = \delta/\kappa_{\mathit{eff}}(A).$$

So, for every order the effective condition number increases ϵ_r must be chosen an order smaller. Hence, a small (effective) condition number is preferred. The computation of $\|A^{-1}\|$ is very expensive, certainly in our case with a large-banded matrix. In a practical sense it is not very useful to implement such a stopping criterion, unless an educated guess for $\|A^{-1}\|$ would be sufficient.

The original stopping criterion is less reliable because if the contraction factor is close to one, the criterion might be met sooner than is tolerable. When the contraction factor is small enough, then the original stopping criterion will suffice. Practically, it shows that for large simulations (over 10^5 grid points) the contraction factor might approach one very close. In these cases a different stopping criterion is desired. We will use the following stopping criterion if the convergence factor exceeds 0.5:

$$\|\underline{x}_{i+1} - \underline{x}_i\| \leq (1 - \rho_c)\epsilon_{it}, \tag{4.6}$$

where ρ_c is the advancing average contraction factor during an iteration cycle. In practical sense we do not want the criterion to be such strict that it will never be met due to machine precision. Therefore we also limit the right-hand side with 10 times the machine precision as a minimum.

In Section 6.8 we will examine the (relative) error, the number of iterations and computation time for these three stopping criteria.

4.7 Enhancing Accuracy

Delft3D-FLOW is programmed with real values in single precision. Single precision is more or less equivalent with eight significant digits. If we consider (4.3) it shows that $\|\hat{U}\|$ and $\kappa(A)$ should not be too large in comparison with the inverse machine precision. If this proves not to be the case, then several possibilities are available to ensure a higher accuracy of the computation.

Of course the first idea is to use double precision (equivalent with sixteen significant digits) for real values instead of single precision. A practical disadvantage is that the program has to be adjusted for this. Furthermore, the computation time will increase and memory storage will be twice as large. Table 4.1 shows that the residues are indeed about a factor 10^8 smaller. The relative error is in case of an *LU* factorisation (continuity equation) proportional to the used machine precision, but in case of an iterative solver (explicit momentum equation) it depends on the stopping criterion.

Another possibility is to decrease the condition number $\kappa(A)$. The test cases show rather large condition numbers (between 10^4 and 10^6). One of the most probable causes for this is the large difference in order between matrix entries on consecutive rows due to the implementation of the open boundary conditions. While main diagonal entries may be as large as 10^5 for internal points of the continuity equation, for open boundary points it is either 1 or -1. Addition of a very small number to a very large number during elimination may cause very low accuracy. It would thus be beneficial to have main diagonal entries which are all of the same order.

Table 4.1: Order of the relative residue computed in single and double precision for the continuity equation (a) and the explicit momentum equation (b).

precision	test case version		
	b01	c05	r11
single	10^{-6}	10^{-6}	10^{-7}
double	10^{-15}	10^{-15}	10^{-16}

(a)

precision	test case version		
	b01	c05	r11
single	10^{-7}	10^{-4}	10^{-7}
double	10^{-9}	10^{-6}	10^{-7}

(b)

The appropriate technique to realise this is called preconditioning, see Golub and Van Loan (1989, p. 124-126). The idea is to multiply the matrix A (and also the vector $\underline{b}$) with a matrix P. The original matrix equation $A\underline{x} = \underline{b}$ then becomes $PA\underline{x} = P\underline{b}$. It is obvious that the exact solution is similar. If the matrix P is chosen close to the inverse of matrix A then the condition number will decrease fairly much. However, the computation of such a matrix P would cause round-off errors itself and cost valuable computation time.

One of the most time-efficient preconditioning methods is called row-scaling. Row-scaling is the technique where every element on the same row including the right-hand side is divided by a certain value. This value should be chosen such that the largest elements on every row are of the same order. Thus the largest element on a row would be a suitable choice. Another option would be to use $\|a_i^T\|$, where a_i^T is the i^{th} row of A and $\|\cdot\|$ is an appropriate norm. All possibilities have in common that the precondition matrix is diagonal.

If the matrix is diagonally dominant, which is the case most of the time for the continuity equation and the explicit momentum equation in Delft3D-FLOW, then division by the main diagonal element seems suitable. The precondition matrix is then defined by $P^{-1} = diag(A)$.

The effect of row-scaling on the condition numbers of the continuity equation is enormous. Some of the test cases show a decrease by factor 10^4, see Table 4.2. The effect on the condition numbers of both momentum equations is not so drastic, but in all cases they have improved.

Table 4.2: Order of the condition numbers with and without row-scaling for the continuity equation (a) and the explicit momentum equation (b).

row-scaling	test case version		
	b01	c05	r11
no	10^4	10^5	10^5
yes	10^1	10^1	10^2

(a)

row-scaling	test case version		
	b01	c05	r11
no	10^1	$10^1 - 10^4$	10^0
yes	10^1	$10^1 - 10^4$	10^0

(b)

Chapter 5

Stability Analysis

Early tests have shown unstable behaviour for some cases in the sense that results might increase dramatically in one time step and diminish in the next (possibly due to variable/dependent matrices). Some test cases react disproportionally to small changes in initial and/or boundary conditions. Stelling (1984) performed a Von Neumann stability analysis on two possible discretisations of the depth-averaged shallow-water equations. We will repeat this analysis for the discretisations as they are presently implemented in Delft3D-FLOW.

We start with a short discussion on the stability of the spatially discretised problem. In the second section the stability analysis for the depth-averaged shallow-water equations is performed and in the third section for the three-dimensional shallow-water equations. It appears that the stability for the three-dimensional shallow-water equations is not unconditional.

5.1 Stability of the Spatially Discretised Problem

The spatially discretised problem is a (set of) differential equation(s) in which the spatial derivatives of the original differential equation(s) have been discretised appropriately. For instance, the two-dimensional heat equation

$$\frac{\partial u}{\partial t} = \alpha \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

would, when spatially discretised, be of the form

$$\frac{d\underline{u}}{dt} = B\underline{u},$$

where B holds all information regarding the chosen spatial discretisations and $\underline{u}$ is the discretised unknown variable. Now, if one of the eigenvalues of B has a positive real part, then the discretisation is unstable regardless of the time discretisation or the time step.

As mentioned in Section 3.5 for only three of the four equations of the shallow-water equations a matrix equation needs to be solved. The fourth equation (continuity equation) is used to calculate the relative vertical velocity ω directly. When written in one matrix equation the three-dimensional shallow-water equations read

$$\frac{d\underline{w}}{dt} = B(\underline{w})\underline{w} + \underline{b} \tag{5.1}$$

with $\underline{w} = (\underline{u}^T, \underline{v}^T, \underline{\zeta}^T)^T$, where $\underline{u}, \underline{v} \in \mathbb{R}^{MNK}$ and $\underline{\zeta} \in \mathbb{R}^{MN}$. The matrix B is dependent on $\underline{w}$ due to the (quadratic) advective terms. The vector $\underline{b}$ contains coefficients regarding the boundary conditions.

The internal array structure of Delft3D-FLOW is not very suitable to compute the matrix $B(\underline{w})$ in that the three equations are computed in three different subroutines in which the explicit terms are not stored. So, all the coefficients would have to be written out to file and read back into memory. This procedure, the construction of the matrix $B(\underline{w})$ and the calculation of the maximal eigenvalue would take too long to carry out for several time steps. Therefore we refrain from doing so.

It is however more easily done in a partial sense by dividing the matrix $B(\underline{w})$ block form,

$$B(\underline{w}) = \begin{pmatrix} B_{11}(\underline{w}) & B_{12}(\underline{w}) & B_{13}(\underline{w}) \\ B_{21}(\underline{w}) & B_{22}(\underline{w}) & B_{23}(\underline{w}) \\ B_{31}(\underline{w}) & B_{32}(\underline{w}) & B_{33}(\underline{w}) \end{pmatrix},$$

and only calculate the maximal eigenvalues for the diagonal blocks $B_{11}(\underline{w})$, $B_{22}(\underline{w})$ and $B_{33}(\underline{w})$. It appears that these maximal eigenvalues are situated around zero. Although this is not adequate to qualify the discretised problem of the three-dimensional shallow-water equations (5.1) as stable or unstable, it is not expected that it is unstable, since the time-integration method is stable provided that the time step is small enough.

5.2 Depth-Averaged Shallow-Water Equations

In this section we will analyse the stability of the depth-averaged shallow-water equations. The equations we will use, are simplified versions of (3.8) and (3.11) in that viscosity, wind stress, bed stress and Coriolis forces are neglected and the advective currents, U and V, and the total water depth H are taken as "frozen" constants. Because we now have one σ-layer, the integrated continuity equation (3.8c) and the continuity equation (3.8d) are the same. Note that in this chapter the frozen constants U and V are *not* the depth-averaged horizontal velocities as in (2.14).

The equations for the point (m, n) read for the first stage

$$\begin{aligned} S_t(u^{p+\frac{1}{2}}) + U\, S_{1x}(u^p) + V\, S_{1y}(u^p) + g\, S_{0x}(\zeta^{p+\frac{1}{2}}) &= 0, \\ S_t(v^{p+\frac{1}{2}}) + U\, S_{+x}(v^{p+\frac{1}{2}}) + V\, S_{+y}(v^{p+\frac{1}{2}}) + g\, S_{0y}(\zeta^p) &= 0, \\ S_t(\zeta^{p+\frac{1}{2}}) + H\, S_{0x}(u^{p+\frac{1}{2}}) + H\, S_{0y}(v^p) + U\, S_{1x}(\zeta^{p+\frac{1}{2}}) + V\, S_{1y}(\zeta^p) &= 0 \end{aligned} \tag{5.2}$$

and for the second stage

$$\begin{aligned} S_t(u^{p+1}) + U\, S_{+x}(u^{p+1}) + V\, S_{+y}(u^{p+1}) + g\, S_{0x}(\zeta^{p+\frac{1}{2}}) &= 0, \\ S_t(v^{p+1}) + U\, S_{1x}(v^{p+\frac{1}{2}}) + V\, S_{1y}(v^{p+\frac{1}{2}}) + g\, S_{0y}(\zeta^{p+1}) &= 0, \\ S_t(\zeta^{p+1}) + H\, S_{0x}(u^{p+\frac{1}{2}}) + H\, S_{0y}(v^{p+1}) + U\, S_{1x}(\zeta^{p+\frac{1}{2}}) + V\, S_{1y}(\zeta^{p+1}) &= 0, \end{aligned} \tag{5.3}$$

where the derivatives are represented by

$$\begin{gathered} S_t(u^{p+\frac{1}{2}}) = (u^{p+\frac{1}{2}} - u^p)/\tfrac{1}{2}\tau, \\ S_{0x}(\zeta^p) = (\zeta^p_{m+\frac{1}{2},n} - \zeta^p_{m-\frac{1}{2},n})/\Delta x, \qquad S_{1x}(u^p) = (u^p_{m+1,n} - u^p_{m-1,n})/2\Delta x, \\ S_{0y}(\zeta^p) = (\zeta^p_{m,n+\frac{1}{2}} - \zeta^p_{m,n-\frac{1}{2}})/\Delta y, \qquad S_{1y}(u^p) = (u^p_{m,n+1} - u^p_{m,n-1})/2\Delta y, \\ S_{+x}(u^p) = \begin{cases} (3u^p_{m,n} - 4u^p_{m-1,n} + u^p_{m-2,n})/2\Delta x, & \text{if } u^p_{m,n} > 0, \\ (-3u^p_{m,n} + 4u^p_{m+1,n} - u^p_{m+2,n})/2\Delta x, & \text{if } u^p_{m,n} \le 0. \end{cases} \end{gathered} \tag{5.4}$$

Note that there are two symbols, S_{0x} and S_{1x}, which represent the second order accurate central difference schemes for the first derivative with respect to x due to the use of a staggered grid. Due to the nonlinearity of the terms $\frac{\partial Hu}{\partial x}$ and $\frac{\partial Hv}{\partial y}$ in the continuity equation ($H = d + \zeta$ is not considered constant at first) they cannot be used as such in the stability analysis. Therefore they have been subjected to the product rule. For $\frac{\partial Hu}{\partial x}$ this results in

$$\frac{\partial Hu}{\partial x} = H\frac{\partial u}{\partial x} + u\frac{\partial H}{\partial x} \approx \bar{H}\frac{\partial u}{\partial x} + U\frac{\partial(\bar{d}+\zeta)}{\partial x} = \bar{H}\frac{\partial u}{\partial x} + U\frac{\partial \zeta}{\partial x},$$

where in the approximation step the total water depth H in the first term, the advective current u and the reference water depth d in the second term are considered constant. We will drop the overbar on H as we have done already in (5.2) and (5.3). Boundary conditions are not taken into account in a Von Neumann stability analysis.

We will prove unconditional stability for (5.2)-(5.3). Therefore we will look at solutions of the form

$$\begin{bmatrix} \tilde{u}^p_{mn} \\ \tilde{v}^p_{mn} \\ \tilde{\zeta}^p_{mn} \end{bmatrix} = \begin{bmatrix} \hat{u}^p \\ \hat{v}^p \\ \hat{\zeta}^p \end{bmatrix} e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)}, \tag{5.5}$$

where $\alpha_1, \alpha_2 \in \mathbb{R}$. The details of the substitution are given for some terms. Substitution of (5.5) into $US_{1x}(u^p)$ gives

$$\begin{aligned} US_{1x}(\tilde{u}^p) &= U\,\frac{\hat{u}^p e^{i\alpha_1(m+1)\Delta x} - \hat{u}^p e^{i\alpha_1(m-1)\Delta x}}{2\Delta x}\, e^{i(\alpha_2 n\Delta y)} \\ &= U\,\frac{e^{i\alpha_1\Delta x} - e^{-i\alpha_1\Delta x}}{2\Delta x}\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)} \\ &= U\,\frac{i\,\sin(\alpha_1\Delta x)}{\Delta x}\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)}. \end{aligned}$$

For $US_{+x}(u^p)$ we have

$$\begin{aligned} US_{+x}(\tilde{u}^p) &= \mathrm{sign}(U)\,\frac{3 - 4e^{-i\,\mathrm{sign}(U)\alpha_1\Delta x} + e^{-2i\,\mathrm{sign}(U)\alpha_1\Delta x}}{2\Delta x}\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)} \\ &= \mathrm{sign}(U)\,\Bigg[\frac{3 - 4\cos(\alpha_1\Delta x) + \cos(2\alpha_1\Delta x)}{2\Delta x} + \\ &\qquad \frac{4i\,\mathrm{sign}(U)\sin(\alpha_1\Delta x) - i\,\mathrm{sign}(U)\sin(2\alpha_1\Delta x)}{2\Delta x}\Bigg]\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)} \\ &= \Bigg[\mathrm{sign}(U)\,\frac{1 - 2\cos(\alpha_1\Delta x) + \cos^2(\alpha_1\Delta x)}{\Delta x} + \\ &\qquad i\frac{2\sin(\alpha_1\Delta x) - \sin(\alpha_1\Delta x)\cos(\alpha_1\Delta x)}{\Delta x}\Bigg]\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)} \\ &= \Bigg[\frac{\mathrm{sign}(U)\,(1 - \cos(\alpha_1\Delta x))^2 + i\sin(\alpha_1\Delta x)(2 - \cos(\alpha_1\Delta x))}{\Delta x}\Bigg]\, \hat{u}^p e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)} \end{aligned}$$

It is left to the reader to work out the substitutions for the other derivatives. Substitution of (5.5) into the first equation of (5.2) and division by $e^{i(\alpha_1 m\Delta x + \alpha_2 n\Delta y)}$ leads to

$$\frac{2}{\tau}\,\hat{u}^{p+\frac{1}{2}} + \frac{i\sin(\frac{1}{2}\alpha_1\Delta x)}{\frac{1}{2}\Delta x}\, g\hat{\zeta}^{p+\frac{1}{2}} = \left(\frac{2}{\tau} - \frac{i\sin(\alpha_1\Delta x)}{\Delta x}\,U - \frac{i\sin(\alpha_2\Delta y)}{\Delta y}V\right)\hat{u}^p.$$

Substitution of (5.5) into all equations of (5.2) and (5.3) and multiplication by $\frac{\tau}{2}$ gives

$$\begin{aligned} A\underline{\hat{w}}^{p+\frac{1}{2}} &= B\underline{\hat{w}}^{p}, \\ C\underline{\hat{w}}^{p+1} &= D\underline{\hat{w}}^{p+\frac{1}{2}} \end{aligned} \tag{5.6}$$

where $\underline{\hat{w}} = [\hat{u}, \hat{v}, \hat{\zeta}]^T$. The matrices A, B, C and D are given by

$$A = \begin{pmatrix} 1 & 0 & \frac{\tau}{2}g\hat{D}_{0x} \\ 0 & 1 + \frac{\tau}{2}U\hat{D}_{+x} + \frac{\tau}{2}V\hat{D}_{+y} & 0 \\ \frac{\tau}{2}H\hat{D}_{0x} & 0 & 1 + \frac{\tau}{2}U\hat{D}_{1x} \end{pmatrix},$$

$$B = \begin{pmatrix} 1 - \frac{\tau}{2}U\hat{D}_{1x} - \frac{\tau}{2}V\hat{D}_{1y} & 0 & 0 \\ 0 & 1 & -\frac{\tau}{2}g\hat{D}_{0y} \\ 0 & -\frac{\tau}{2}H\hat{D}_{0y} & 1 - \frac{\tau}{2}V\hat{D}_{1y} \end{pmatrix},$$

$$C = \begin{pmatrix} 1 + \frac{\tau}{2}U\hat{D}_{+x} + \frac{\tau}{2}V\hat{D}_{+y} & 0 & 0 \\ 0 & 1 & \frac{\tau}{2}g\hat{D}_{0y} \\ 0 & \frac{\tau}{2}H\hat{D}_{0y} & 1 + \frac{\tau}{2}V\hat{D}_{1y} \end{pmatrix},$$

$$D = \begin{pmatrix} 1 & 0 & -\frac{\tau}{2}g\hat{D}_{0x} \\ 0 & 1 - \frac{\tau}{2}U\hat{D}_{1x} - \frac{\tau}{2}V\hat{D}_{1y} & 0 \\ -\frac{\tau}{2}H\hat{D}_{0x} & 0 & 1 - \frac{\tau}{2}U\hat{D}_{1x} \end{pmatrix},$$

where

$$\begin{aligned} \hat{D}_{0x} &= 2\,i\sin(\tfrac{1}{2}\alpha_1\Delta x)/\Delta x, & \hat{D}_{0y} &= 2\,i\sin(\tfrac{1}{2}\alpha_2\Delta y)/\Delta y, \\ \hat{D}_{1x} &= i\sin(\alpha_1\Delta x)/\Delta x, & \hat{D}_{1y} &= i\sin(\alpha_2\Delta y)/\Delta y, \end{aligned}$$
$$\hat{D}_{+x} = [\mathrm{sign}(U)(1-\cos(\alpha_1\Delta x))^2 + i\sin(\alpha_1\Delta x)(2-\cos(\alpha_1\Delta x))]/\Delta x,$$
$$\hat{D}_{+y} = [\mathrm{sign}(V)(1-\cos(\alpha_2\Delta y))^2 + i\sin(\alpha_1\Delta y)(2-\cos(\alpha_2\Delta y))]/\Delta y.$$

The solution $\underline{\hat{w}}^{p+1}$ of (5.6) can now be written as $\underline{\hat{w}}^{p+1} = G\underline{\hat{w}}^{p}$ with $G = C^{-1}DA^{-1}B$, the amplification matrix. Stability is proven if the largest eigenvalue of G in absolute sense is not larger than 1. However, it is sufficient to show that $\|G^p\|_2$ is bounded $\forall p$.

Therefore we rewrite G as:

$$\begin{aligned} G &= \Lambda\Lambda^{-1}C^{-1}\Lambda\Lambda^{-1}D\Lambda\Lambda^{-1}A^{-1}\Lambda\Lambda^{-1}B\Lambda\Lambda^{-1} \\ &= \Lambda(\Lambda^{-1}C\Lambda)^{-1}(\Lambda^{-1}D\Lambda)(\Lambda^{-1}A\Lambda)^{-1}(\Lambda^{-1}B\Lambda)\Lambda^{-1} \\ &= \Lambda\tilde{C}^{-1}\tilde{D}\tilde{A}^{-1}\tilde{B}\Lambda^{-1} \end{aligned}$$

where $\Lambda = diag(1, 1, \sqrt{H/g})$. Note that $\tilde{A}$, $\tilde{B}$, $\tilde{C}$ and $\tilde{D}$ are complex symmetric matrices. This leads to the following relation for $\|G^p\|_2$:

$$\begin{aligned} \|G^p\|_2 &= \|\Lambda(\tilde{C}^{-1}\tilde{D}\tilde{A}^{-1}\tilde{B})^p\Lambda^{-1}\|_2 \\ &= \|\Lambda\tilde{C}^{-1}\cdot\tilde{D}\tilde{A}^{-1}\cdot(\tilde{B}\tilde{C}^{-1}\tilde{D}\tilde{A}^{-1})^{p-1}\cdot\tilde{B}\Lambda^{-1}\|_2 \\ &\le \|\Lambda\tilde{C}^{-1}\|_2\cdot\|\tilde{D}\tilde{A}^{-1}\|_2^p\cdot\|\tilde{B}\tilde{C}^{-1}\|_2^{p-1}\cdot\|\tilde{B}\Lambda^{-1}\|_2. \end{aligned}$$

Stability is ensured if $\|\tilde{D}\tilde{A}^{-1}\|_2$ and $\|\tilde{B}\tilde{C}^{-1}\|_2$ are not larger than 1. Since both pairs of matrices are equivalent we will only show that $\|\tilde{B}\tilde{C}^{-1}\|_2 \le 1$. To prove this we write $\tilde{B}$ and $\tilde{C}$ in the following partitioned form:

$$\tilde{B} = \begin{pmatrix} \tilde{b} & 0 \\ 0 & \tilde{B}_d \end{pmatrix}, \quad \tilde{C} = \begin{pmatrix} \tilde{c} & 0 \\ 0 & \tilde{C}_d \end{pmatrix}$$

with

$$\tilde{b} = 1 - \tfrac{\tau}{2}U\hat{D}_{1x} - \tfrac{\tau}{2}V\hat{D}_{1y}, \qquad \tilde{c} = 1 + \tfrac{\tau}{2}U\hat{D}_{+x} + \tfrac{\tau}{2}V\hat{D}_{+y},$$

$$\tilde{B}_d = \begin{pmatrix} 1 & -\tfrac{\tau}{2}\sqrt{gH}\hat{D}_{0y} \\ -\tfrac{\tau}{2}\sqrt{gH}\hat{D}_{0y} & 1 - \tfrac{\tau}{2}V\hat{D}_{1y} \end{pmatrix}, \quad \tilde{C}_d = \begin{pmatrix} 1 & \tfrac{\tau}{2}\sqrt{gH}\hat{D}_{0y} \\ \tfrac{\tau}{2}\sqrt{gH}\hat{D}_{0y} & 1 + \tfrac{\tau}{2}V\hat{D}_{1y} \end{pmatrix}.$$

It follows that $\tilde{B}_d = \tilde{C}_d^H$ and

$$\tilde{B}\tilde{C}^{-1} = \begin{pmatrix} \tilde{b}/\tilde{c} & 0 \\ 0 & \tilde{C}_d^H \tilde{C}_d^{-1} \end{pmatrix}.$$

Furthermore, it is easily verified that $\tilde{C}_d$ is a normal matrix.

Lemma 5.1. *Let A be a normal matrix (that means $AA^H = A^H A$) then $A^H A^{-1}$ is an unitary matrix.*

Proof.
$$\begin{aligned}(A^H A^{-1})(A^H A^{-1})^H &= A^H A^{-1} A^{-H} A = A^H (A^H A)^{-1} A \\ &= A^H (AA^H)^{-1} A = A^H A^{-H} A^{-1} A = I.\end{aligned}$$ □

From Lemma 5.1 can be derived that $\tilde{C}_d^H \tilde{C}_d^{-1}$ is a unitary matrix. Thus

$$\|\tilde{B}\tilde{C}^{-1}\|_2 = \max(|\tilde{b}/\tilde{c}|, \|\tilde{C}_d^H \tilde{C}_d^{-1}\|_2) = \max(|\tilde{b}/\tilde{c}|, 1).$$

It can be shown that $|\tilde{b}| \le |\tilde{c}|$. So, (5.2)-(5.3) is unconditionally stable.

Figure 5.1 shows the maximal absolute eigenvalue for every pair of (α_1, α_2) for the depth-averaged test case 'c01', see Section 6.3. The axes are bounded by the relations $|\alpha_1 \Delta x| \le \pi$ and $|\alpha_2 \Delta y| \le \pi$. In this test case Δx and Δy are both 1000. The maximal eigenvalue is assumed in the point $(0, 0)$ and it is one. So, this test case is indeed stable.

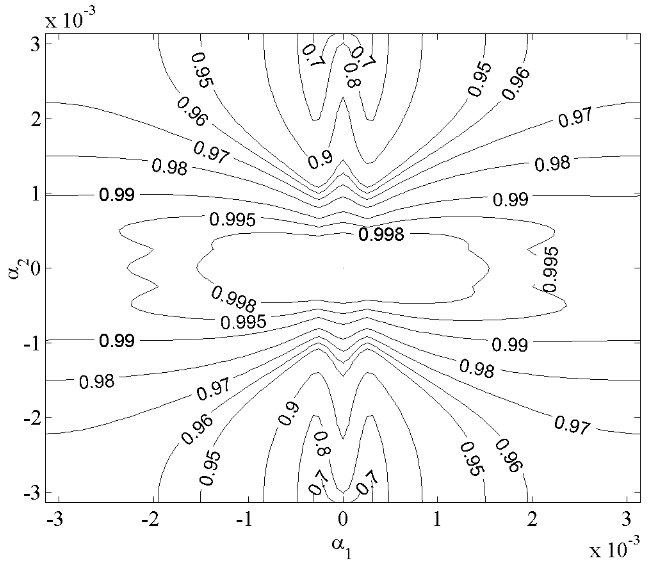


Figure 5.1: Stability field for a stable test case.

5.3 Three-Dimensional Shallow-Water Equations

The stability for the three-dimensional shallow-water equations will be analysed similarly to the depth-averaged equations. The main difference is that now for every σ-layer separate horizontal velocities are distinguished. Through the depth-integrated continuity equation ζ_{mn} depends on u_{mnk} and v_{mnk} $\forall k$, therefore u_{mnk} and v_{mnk} are taken into account for every k separately. Thus the system we will analyse in this section will comprise $2K$ momentum equations (two for every σ-layer), where K is the number of σ-layers, and one depth-integrated continuity equation.

The continuity equations (3.8d) and (3.11d) are not taken into account, because the relative vertical velocity ω is calculated "afterwards". The vertical advective current ω varies in the vertical direction, so we will model it as $\omega(k)$. Furthermore, viscosity will be taken into account, the horizontal eddy viscosity as a frozen constant ν_t^H and the vertical eddy viscosity, dependent on the σ-layer, as $\nu_t^V(k)$. The Coriolis forces are again neglected. Again the product rule has been applied to two terms of the continuity equation.

Then the equations for (m, n) read for the first stage

$$\begin{aligned}
S_t(u_k^{p+\frac{1}{2}}) + U\,S_{1x}(u_k^p) + V\,S_{1y}(u_k^p) + \tfrac{\omega_k}{H}\,S_\sigma(u_k^{p+\frac{1}{2}}) + g\,S_{0x}(\zeta^{p+\frac{1}{2}}) - \tfrac{\nu_t^V(k)}{H^2} S_{\sigma\sigma}(u_k^{p+\frac{1}{2}}) = 0 \quad \forall k,\\
S_t(v_k^{p+\frac{1}{2}}) + U\,S_{+x}(v_k^{p+\frac{1}{2}}) + V\,S_{+y}(v_k^{p+\frac{1}{2}}) + \tfrac{\omega_k}{H}\,S_\sigma(v_k^{p+\frac{1}{2}})\\
+ g\,S_{0y}(\zeta^p) - 2\nu_t^H[S_{xx}(v_k^{p+\frac{1}{2}}) + S_{yy}(v_k^{p+\frac{1}{2}})] - \tfrac{\nu_t^V}{H^2} S_{\sigma\sigma}(v_k^{p+\frac{1}{2}}) = 0 \quad \forall k,\\
S_t(\zeta^{p+\frac{1}{2}}) + H\sum\nolimits_k \Delta\sigma_k S_{0x}(u_k^{p+\frac{1}{2}}) + H\sum\nolimits_k \Delta\sigma_k S_{0y}(v_k^p) + U\,S_{1x}(\zeta^{p+\frac{1}{2}}) + V\,S_{1y}(\zeta^p) = 0
\end{aligned} \tag{5.7}$$

and for the second stage

$$\begin{aligned}
S_t(u_k^{p+1}) + U\,S_{+x}(u_k^{p+1}) + V\,S_{+y}(u_k^{p+1}) + \tfrac{\omega_k}{H}\,S_\sigma(u_k^{p+1})\\
+ g\,S_{0x}(\zeta^{p+\frac{1}{2}}) - 2\nu_t^H[S_{xx}(u_k^{p+1}) + S_{yy}(u_k^{p+1})] - \tfrac{\nu_t^V}{H^2} S_{\sigma\sigma}(u_k^{p+1}) = 0 \quad \forall k,\\
S_t(v_k^{p+1}) + U S_{1x}(v_k^{p+\frac{1}{2}}) + V S_{1y}(v_k^{p+\frac{1}{2}}) + \tfrac{\omega_k}{H}\,S_\sigma(v_k^{p+1}) + g S_{0y}(\zeta^{p+1}) - \tfrac{\nu_t^V(k)}{H^2} S_{\sigma\sigma}(v_k^{p+1}) = 0 \quad \forall k,\\
S_t(\zeta^{p+1}) + H\sum\nolimits_k \Delta\sigma_k S_{0x}(u_k^{p+\frac{1}{2}}) + H\sum\nolimits_k \Delta\sigma_k S_{0y}(v_k^{p+1}) + U\,S_{1x}(\zeta^{p+\frac{1}{2}}) + V\,S_{1y}(\zeta^{p+1}) = 0.
\end{aligned} \tag{5.8}$$

The symbols S_{xx} an S_{yy} represent second order accurate central difference schemes of the second derivative with respect to the x and y respectively. The symbols S_σ and $S_{\sigma\sigma}$ represent the discretisations for the vertical advection term and vertical viscosity term respectively. They are given by

$$\begin{aligned}
S_\sigma(u_k^p) = \frac{2}{\Delta\sigma_{k-1} + 2\Delta\sigma_k + \Delta\sigma_{k+1}} \left[\frac{\Delta\sigma_k + \Delta\sigma_{k+1}}{\Delta\sigma_{k-1} + \Delta\sigma_k}(u_{k-1}^p - u_k^p) + \frac{\Delta\sigma_{k-1} + \Delta\sigma_k}{\Delta\sigma_k + \Delta\sigma_{k+1}}(u_k^p - u_{k+1}^p) \right],\\
S_{\sigma\sigma}(u_k^p) = \frac{u_{k-1}^p - u_k^p}{\frac{1}{2}(\Delta\sigma_{k-1} + \Delta\sigma_k)\Delta\sigma_k} - \frac{u_k^p - u_{k+1}^p}{\frac{1}{2}(\Delta\sigma_k + \Delta\sigma_{k+1})\Delta\sigma_k}.
\end{aligned} \tag{5.9}$$

Compare with the items 8. and 12. in Appendix A.

As done in the previous section we will look at solutions of the form:

$$\begin{bmatrix} \tilde{u}^p_{mn,1} \\ \tilde{u}^p_{mn,2} \\ \vdots \\ \tilde{u}^p_{mn,K} \\ \tilde{v}^p_{mn,1} \\ \tilde{v}^p_{mn,2} \\ \vdots \\ \tilde{v}^p_{mn,K} \\ \tilde{\zeta}^p_{mn} \end{bmatrix} = \begin{bmatrix} \hat{u}^p_1 \\ \hat{u}^p_2 \\ \vdots \\ \hat{u}^p_K \\ \hat{v}^p_1 \\ \hat{v}^p_2 \\ \vdots \\ \hat{v}^p_K \\ \hat{\zeta}^p \end{bmatrix} e^{i(\alpha_1 m\Delta x + \alpha_2 n \Delta y)} = \underline{\hat{w}}^p \, e^{i(\alpha_1 m\Delta x + \alpha_2 n \Delta y)} \tag{5.10}$$

with $\alpha_1, \alpha_2 \in \mathbb{R}$. The size of the vectors $\underline{\hat{w}}^p$ and matrices A, B, C and D depends on the number of σ-layers K. For our convenience we will set K to 3, so $\underline{\hat{w}}^p \in \mathbb{R}^7$ and $A, B, C, D \in \mathbb{R}^{7\times7}$. Substitution of (5.10) into the momentum equation for $k = 2$ of (5.7) leads to

$$\frac{\hat{u}_2^{p+\frac{1}{2}} - \hat{u}_2^p}{\frac{1}{2}\tau} + U\,\hat{D}_{1x}\hat{u}_2^p + V\,\hat{D}_{1y}\hat{u}_2^p + \frac{2\omega_2}{H(\Delta\sigma_1 + 2\Delta\sigma_2 + \Delta\sigma_3)}\left[\frac{\Delta\sigma_2 + \Delta\sigma_3}{\Delta\sigma_1 + \Delta\sigma_2}(\hat{u}_1^{p+\frac{1}{2}} - \hat{u}_2^{p+\frac{1}{2}}) + \right.$$
$$\left.\frac{\Delta\sigma_1 + \Delta\sigma_2}{\Delta\sigma_2 + \Delta\sigma_3}(\hat{u}_2^{p+\frac{1}{2}} - \hat{u}_3^{p+\frac{1}{2}})\right] + g\hat{D}_{0x}\zeta^{p+\frac{1}{2}} - \frac{\nu_t^V(2)}{H^2}\left[\frac{\hat{u}_1^{p+\frac{1}{2}} - \hat{u}_2^{p+\frac{1}{2}}}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)\Delta\sigma_2} - \frac{\hat{u}_2^{p+\frac{1}{2}} - \hat{u}_3^{p+\frac{1}{2}}}{\frac{1}{2}(\Delta\sigma_2 + \Delta\sigma_3)\Delta\sigma_2}\right] = 0.$$

For the top and the bottom layer the vertical viscosity terms and the vertical advective terms are different as mentioned in Section 3.4. The boundary conditions at the free surface and the bottom are used within the vertical viscosity terms and for the vertical advective terms upwind schemes are used instead of central difference schemes. Implementation of these boundary conditions causes a nonzero right-hand side. We will show that this right-hand side can be ignored in a Von Neumann stability analysis.

Consider the one-dimensional linear initial value problem

$$\begin{cases} \frac{dy}{dt} - \lambda y = g(t), \\ y(0) = y_0. \end{cases} \tag{5.11}$$

The computed solution $\hat{y}$ is the exact solution to a slightly perturbed system, namely

$$\begin{cases} \frac{d\hat{y}}{dt} - \lambda \hat{y} = g(t), \\ \hat{y}(0) = y_0 + \epsilon_0. \end{cases} \tag{5.12}$$

When performing a stability analysis we are interested whether the error $\epsilon = \hat{y} - y$ vanishes or explodes. We can now construct an initial value problem for ϵ by subtracting (5.11) from (5.12). This results in

$$\begin{cases} \frac{d\epsilon}{dt} - \lambda \epsilon = 0, \\ \epsilon(0) = \epsilon_0. \end{cases} \tag{5.13}$$

Note that this is only possible for *linear* differential equations. The system in (5.13) is the homogeneous version of (5.11).

Thus substitution of (5.10) into the momentum equation for $k = 1$ of (5.7) gives

$$\frac{\hat{u}_1^{p+\frac{1}{2}} - \hat{u}_1^p}{\frac{1}{2}\tau} + U\,\hat{D}_{1x}\hat{u}_1^p + V\,\hat{D}_{1y}\hat{u}_1^p + \frac{\omega_1}{H}\left[\frac{\hat{u}_1^{p+\frac{1}{2}} - \hat{u}_2^{p+\frac{1}{2}}}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)}\right] + g\,\hat{D}_{0x}\zeta^{p+\frac{1}{2}}$$
$$- \frac{\nu_t^V(1)}{H^2}\left[-\frac{\hat{u}_1^{p+\frac{1}{2}} - \hat{u}_2^{p+\frac{1}{2}}}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)\Delta\sigma_1}\right] = 0.$$

Substitution of (5.10) in all equations of (5.7) and (5.8) gives

$$\begin{aligned} A\underline{\hat{w}}^{p+\frac{1}{2}} &= B\underline{\hat{w}}^p, \\ C\underline{\hat{w}}^{p+1} &= D\underline{\hat{w}}^{p+\frac{1}{2}} \end{aligned} \tag{5.14}$$

where $\underline{\hat{w}} = [\hat{u}_1, \hat{u}_2, \hat{u}_3, \hat{v}_1, \hat{v}_2, \hat{v}_3, \hat{\zeta}]^T$. The matrices A, B, C and D are given in Appendix C.

Due to the asymmetric nature of these matrices the analogy with the stability analysis of the depth-averaged shallow-water equations does not carry on. Therefore we will perform the stability analysis per test case on the computer.

To this end we will set the values for the frozen quantities as best as possible according to the test case and vary α_1 between $-\pi/\Delta x$ and $\pi/\Delta x$ and α_2 between $-\pi/\Delta y$ and $\pi/\Delta y$. For a certain number of pairs (α_1, α_2) we will calculate the maximal absolute eigenvalue of the amplification matrix $G = C^{-1}DA^{-1}B$. If any of these eigenvalues exceeds 1, then the problem is unstable. The result of this analysis for a single problem/test case can be given in a contour plot. Figure 5.2 shows the stability field of an unstable test case. The test case that was used is 'c02', see Section 6.3.

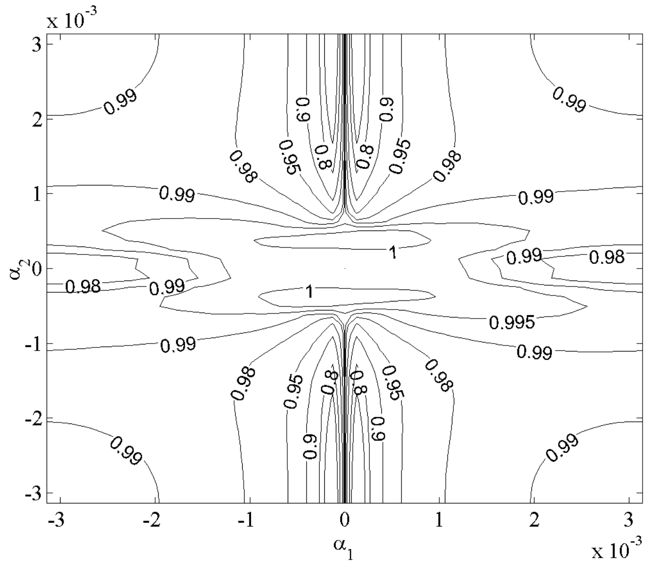


Figure 5.2: Stability field for an unstable test case.

It should be noted that in Delft3D-FLOW the vertical viscosity coefficient ν_t^v cannot be set to a certain value, because it is calculated by means of a turbulence model. So, it remains to be seen if ν_t^v will ever assume such a low value. However, the stability analysis shows that the time integration scheme is not unconditionally stable.

Chapter 6

Test Cases

Several physical and numerical aspects of a shallow water flow problem (may) influence the accuracy and stability of the numerical solution. Some of the physical aspects are bed topography and boundary conditions. Numerical aspects are problem size, vertical σ-layer distribution, time step, turbulence closure model and drying and flooding and most importantly the solution algorithm.

In order to investigate their influence on the numerical accuracy and stability we have set up various test cases. We start with small, simple ones and end with larger, more complex cases. First we will examine a simple case to test the performance of Delft3D-FLOW in double precision. The succeeding test cases will highlight influence of one or two aspects on the accuracy and/or the stability. Finally, we use the eastern part of the Westerschelde as a test case to analyse adjustments to Delft3D-FLOW on a large and actual case.

6.1 Definitions and Settings

In this chapter several computed quantities refer to equations in previous chapters. For the reader's convenience we will summarise them here. Furthermore, we will pay some attention to the settings in Delft3D-FLOW for reproductive purposes.

In Delft3D-FLOW three equations are solved each half time step, viz. the implicit and explicit momentum equation and the depth-integrated continuity equation. The implicit momentum equation and the depth-integrated continuity equation are coupled, i.e. the first is solved with respect to the velocity and substituted into the latter. The (local) continuity equation is used for direct calculation of the vertical velocities. Therefore we are mainly interested in the explicit momentum equation (explicit with respect to ζ) and the depth-integrated continuity equation, which from this point on we will refer to as the continuity equation.

We are mainly interested in the *accuracy* of the result. We analyse the accuracy through calculation of the error and the relative error. The error is defined by $\|\underline{x} - \hat{\underline{x}}\|$ and the relative error is defined by

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|}. \tag{6.1}$$

Furthermore, we distinguish (a) the (relative) error of the solution of a single linear matrix equation (eg. one iteration of the continuity equation or the explicit momentum equation) and (b) the (relative) error of the final result after a whole time step.

(a) The error and the relative error are not computable because we do not have the exact solution $\underline{x}$ at our disposal. General estimates of these errors are not known, but upper bounds on the relative error are. Small upper bounds guarantee high accuracy, but large ones do not automatically signify low accuracy. A general upper bound of the relative error is given in Lemma 4.6:

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \kappa(A) \frac{\|A\hat{\underline{x}} - \underline{b}\|}{\|\underline{b}\|} = \kappa(A)\, r, \tag{6.2}$$

where r is the relative residue. This upper bound can be improved if we use the effective condition number as done in Corollary 4.1 For the *LU* factorisation (applied on the continuity equation) of tridiagonal systems another upper bound can be constructed shown in Equation (4.3):

$$\frac{\|\underline{x} - \hat{\underline{x}}\|}{\|\underline{x}\|} \leq \frac{2\|E\|\kappa(A)}{\|A\| - \|E\|\kappa(A)}. \tag{6.3}$$

For the continuity equation both upper bounds are computed and the smallest is always presented. Averagely the upper bound in (6.2) is smaller than the one in (6.3), but usually not more than a factor 10. Note that the above upper bounds only apply to linear systems.

We remind the reader that the numerical scheme of the continuity equation is in fact non-linear. It is solved with an iterative scheme of which a single iteration is computed by means of a *LU* factorisation (see last part of Section 3.5). The number of iterations is set a priori to two. The upper bounds of the relative error we calculate are for single iterations. Since the residues and upper bounds of the two iterations are more or less the same, we only show results of the first iteration. Due to the non-linearity the matrix of the continuity equation is updated for the second iteration.

(b) Compared to the results computed in single precision the results of the double precision computation can be considered exact. So, we can actually calculate estimates for the absolute and the relative error of the results computed in single precision. We will use the results from the standard output of Delft3D-FLOW. Note that these results do not contain the results of every half time step and every iteration. In this way we can only analyse the accuracy of the complete system and not of single matrix equations.

We want to point out that these computed absolute and relative errors may be very poor estimates, because slightly different matrix equations are solved for single and double precision due to differences in the solutions of the previous time step.

The *relative residue* is defined in Section 4.2 by

$$r = \frac{\|A\hat{\underline{x}} - \underline{b}\|}{\|\underline{b}\|}, \tag{6.4}$$

where $\hat{\underline{x}}$ is the computed solution vector. It is the best quantity to check computational improvement with regard to the accuracy due to adjustments in the numerical scheme. For instance increment of the upper bound of the relative error does theoretically not imply less accuracy, but increment of the relative residue does.

The difference between *single and double precision* is the number of bytes (resp. 8 and 16) used to represent a floating point number. The acquired accuracies for the *representation* of (not a floating point operation on) a real number are given in Table 6.1. They apply to all used computer architectures.

The numerical scheme for explicit momentum equation is computed with an iterative scheme. During computation we keep track of the *number of iterations* and the *contraction factors* of which the latter is defined in Equation 4.5 by

$$\rho_c(i) = \frac{\|\underline{x}_{i+1} - \underline{x}_i\|}{\|\underline{x}_i - \underline{x}_{i-1}\|}. \tag{6.5}$$

Table 6.1: Acquired accuracy

precision	machine precision μ
single	$6.0 \cdot 10^{-8}$
double	$1.1 \cdot 10^{-16}$

In Delft3D-FLOW a hard-coded maximum of 50 iterations is allowed.

We will analyse three *stopping criteria* for the iterative scheme, namely:

original The original stopping criterion: $\|\underline{x}_n - \underline{x}_{n-1}\| \le \epsilon_{it}$, where ϵ_{it} is set to 10^{-6} in Delft3D-FLOW.

residual The residual stopping criterion:

$$\frac{\|A\underline{x}_n - \underline{b}\|}{\|\underline{b}\|} \le \epsilon_r,$$

where ϵ_r is set to 10^{-4}.

original with contraction factor If the advancing average contraction factor ρ_c (see Equation (4.5) and further) is smaller than 0.5, then the original stopping criterion is used. Every time it is larger than 0.5 the following criterion must be met:

$$\|\underline{x}_n - \underline{x}_{n-1}\| \le \max(10\mu, (1-\rho_c)\,\epsilon_{it}). \tag{6.6}$$

Delft3D-FLOW

We used *version* 3.10.10.00 of Delft3D-FLOW. The reason for this is, that it is the latest version written completely in Fortran 77 code. The Fortran 77 compiler on the Sun-Sparc has the option to read declarations of 'reals' as declarations of 'doubles', which is no longer available in the compilers for Fortran 90. Most likely the SGI has a similar compiler option.

The settings we used in Delft3D-FLOW are included in Appendix E.

For the stability analysis we will use the characteristic numbers: CFL and cell-Péclet. The cell-Péclet number is defined as

$$P := \frac{U\,\Delta x}{\nu_t^H}, \tag{6.7}$$

where U is the characteristic advective current. For the CFL number we distinguish two definitions, one for the velocity and one for the surface waves, respectively

$$CFL_U := \frac{U\,\tau}{\Delta x} \quad \text{and} \quad CFL_\zeta := \frac{\sqrt{gH}\,\tau}{\Delta x}. \tag{6.8}$$

6.2 Test Case 1: Square Reservoir with Wind-Driven Flow

This simple test case is used to check whether Delft3D-FLOW produces the same results with different machine precisions. It is a simulation of a square reservoir (10 km by 10 km) with closed boundaries. The wind in southern direction has a constant speed of 10 m/s. The horizontal grid size is 1 km in both directions. Along the vertical axis five equidistant σ-layers are defined. The uniform water depth is 10 m. The time step is chosen equal to 2 minutes.

Table 6.2 shows the difference in order of the relative residue and the relative error. For the continuity equation this difference is about the same as the difference between single and double precision. For the explicit momentum equation, however, the difference is much smaller. This is due to stopping criterion for the Gauss-Jacobi iteration, which is for both machine precisions the same. Note that when computing in double precision this criterion may be set more strict. The contraction factor for this iteration is in both cases in the order of 10^{-3}. With single precision

Table 6.2: Order of the relative residue and the upper bound of the relative error and the number of iterations for the Gauss-Jacobi iteration of test case 1 for different machine precisions.

precision	continuity equation		explicit momentum equation		
	rel. residue	rel. error	rel. residue	rel. error	#iterations
single	10^{-7}	10^{-7}	10^{-7}	10^{-7}	2 – 3
double	10^{-15}	10^{-15}	10^{-8} – 10^{-11}	10^{-8} – 10^{-11}	2 – 3

this leads to a relative error of at most the order 10^{-7} with 2 or 3 iterations, but with double precision the variation in the order of the residue and the relative error is a direct consequence of the variation in the number of iterations.

If we consider the result computed in double precision as the exact solution, then we can compute the (cumulative) error for the single precision result by simple subtraction as done in Figure 6.1. We have repeated this with a smaller grid size (100 m) and a larger grid (6400 grid points). It is clear that both solutions slowly diverge with respect to each other for 64 as well as 6400 grid points. Note that this divergent behaviour only applies to problems with closed

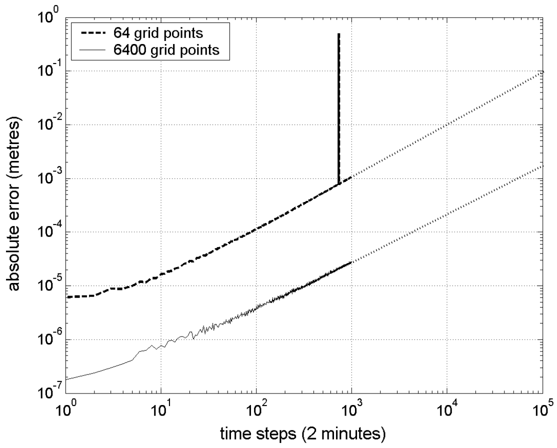


Figure 6.1: Absolute error in ζ for single precision. The dotted lines are linear extrapolations.

boundaries (lakes for instance). From extrapolation of the graph in Figure 6.1 we can conclude that differences in the order of centimetres will occur after 10^4 time steps for 6400 grid points.

When, however, open boundary conditions are imposed (seas, rivers and estuaries), it is most likely that this difference does not grow.

The extra computation time for double precision is about 10 to 15 percent for the Sun Sparc.

6.3 Test Case 2: Reservoir with Island and Dams

This test case will be used to analyse the differences in residue and (relative) error if the grid size is refined several times. Furthermore, refinement of the time step will be analysed. And finally, we will look at the differences between several computer architectures, viz. a Sun-Sparc, an SGI and a PC.

In this test case (WL reference: '01-bakje') a square reservoir (8 km by 8 km) with three closed boundaries and one open boundary is simulated. A square island is situated in the centre of the reservoir from which eight thin dams stretch out toward the boundaries, see Figure 6.2. At the 'lower' boundary a tidal condition is defined, namely $\zeta(t) = 1 + \cos(\pi t/720)$ with t in minutes. The original version of this test case has only open boundaries at which homogeneous boundary conditions ($\zeta = 0$) are imposed except for the lower boundary at which the tidal condition is defined. We will refer to this later on.

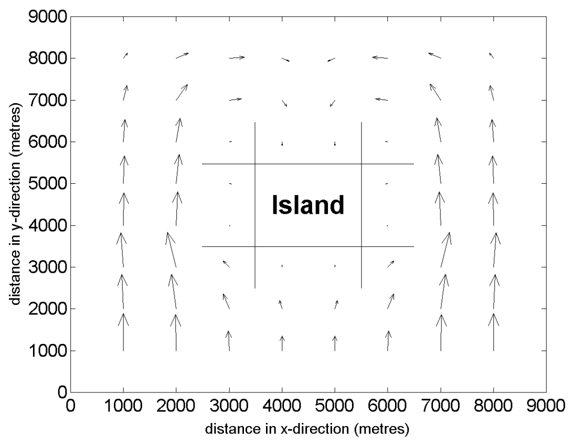


Figure 6.2: Vector field of the depth integrated velocity of test case 2.

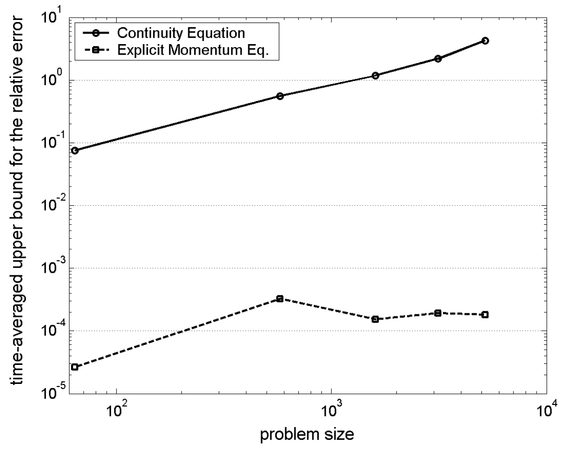
The uniform water depth in this test case is 5 m. Along the vertical axis ten equidistant σ-layers have been defined. The horizontal grid size in both directions is 1 km. Of this test case several different versions have been simulated. The remaining specifications of these versions are given in Table 6.3.

Table 6.3: Specifications of the different versions of test case 2.

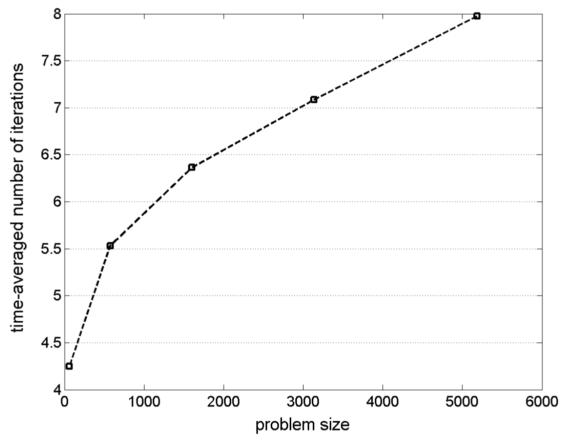
specification	test case version							
	b01	b02	b04	b05	b21	b22	b23	b24
grid size	1 km	1 km	1 km	1 km	$\frac{1}{3}$ km	$\frac{1}{5}$ km	$\frac{1}{7}$ km	$\frac{1}{9}$ km
# grid points	64	64	64	64	576	1600	3136	5184
# σ-layers	10	10	10	1	10	10	10	10
time step	10 min	5 min	2.5 min	10 min	10 min	10 min	10 min	10 min

Grid refinement leads directly to increment of the problem size, numerically speaking. Grid refinement has the advantage that it reduces the truncation error, which might be a reason to carry out the refinement. However, it will have a negative effect on the accuracy of the computed solution. Figure 6.3a shows a double logarithmic graph of the time-average upper bound for the relative error of the continuity equation against the problem size, which is in this case the number of grid points in x-direction multiplied with those in y-direction (thus $nmax \times mmax$ in Delft3D-FLOW).

If assumed that the relative error of the continuity equation is proportional to Kn^α, where K is some constant. Then from Figure 6.3a can be derived that $\alpha \approx 0.91$ in case of the continuity equation, so the upper bound for the relative error depends linearly on the problem size. For the



(a)

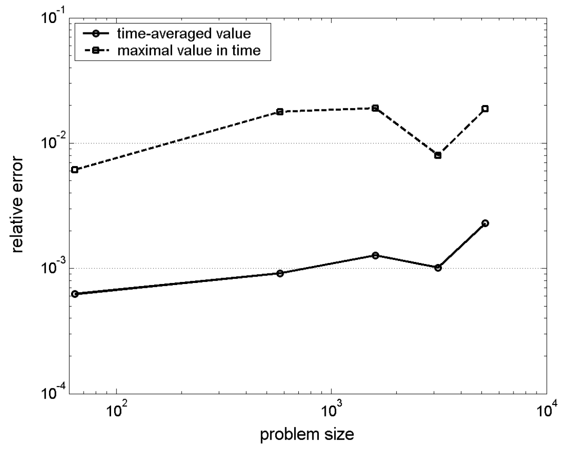


(b)

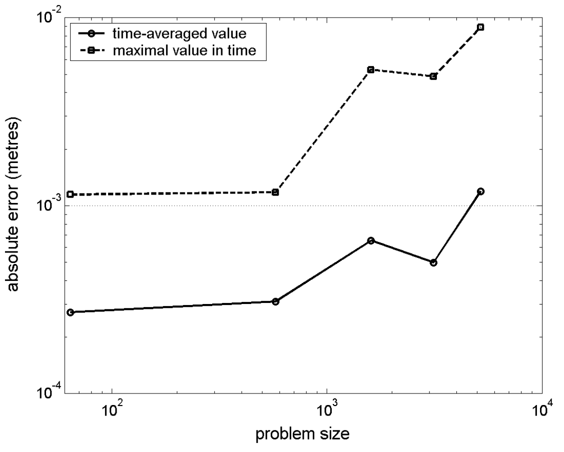
Figure 6.3: The time-averaged upper bound for the relative error (a) and the time-averaged number of iterations (b) when increasing the problem size (test case 2).

explicit momentum equation the accuracy remains rather high due to increment of the number of iterations. For both equations the condition numbers of the matrices are fairly constant for an increasing problem size. For the continuity equation the condition number is of $\mathcal{O}(10^4)$ and for the explicit momentum equation of $\mathcal{O}(10^1)$.

We remind the reader that, besides the determination of an upper bound for the relative error of the continuity equation, as done above and mentioned in Section 6.1 under (a), we can also estimate the relative error in ζ as is explained under (b). In Figure 6.4 the relative error in ζ has been plotted against the problem size.



(a)

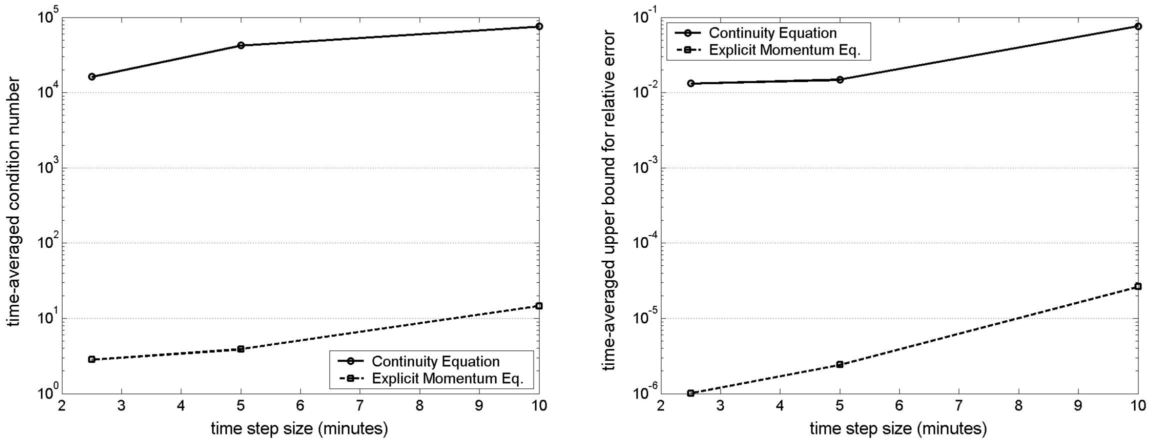


(b)

Figure 6.4: Time-averaged relative (a) and absolute error (b) in ζ when increasing the problem size (test case 2).

The upper bound of the relative error (Figure 6.3a) is so large that it does not guarantee that the calculated result is adequate for practical use. It is wanted that either a stricter upper bound is constructed or the numerical scheme is improved, for instance through application of a preconditioning method. We will use preconditioning later on in Section 6.7.

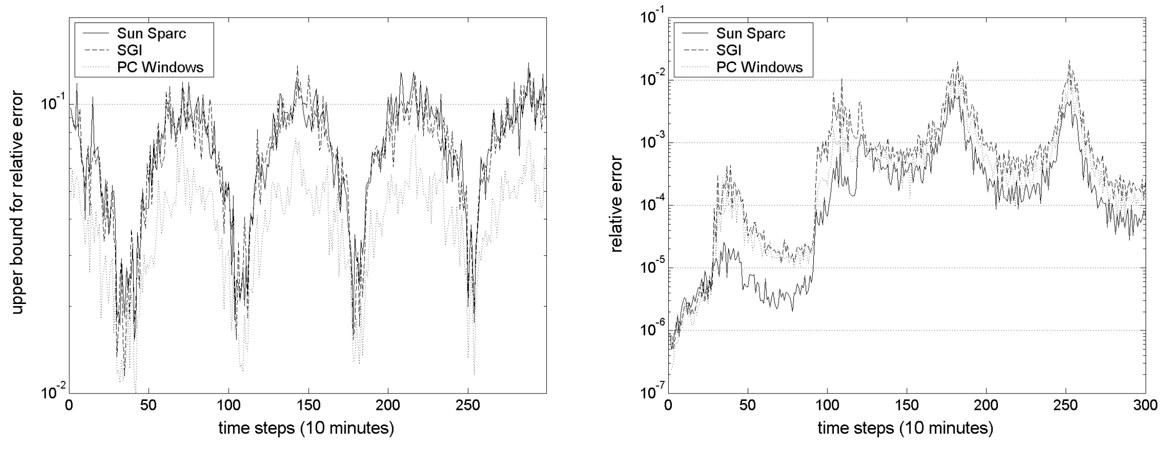
A decrease of the time step leads, just like refinement of the grid, to reduction of the truncation error, but more importantly it increases the elements on the main diagonal of the matrices. Direct consequences are smaller condition numbers and subsequently smaller upper bounds on the relative error, see Figure 6.5.



(a) (b)

Figure 6.5: The time-averaged condition number (a) and the upper bound for the time-averaged relative error (b) when decreasing the size of the time step (test case b01).

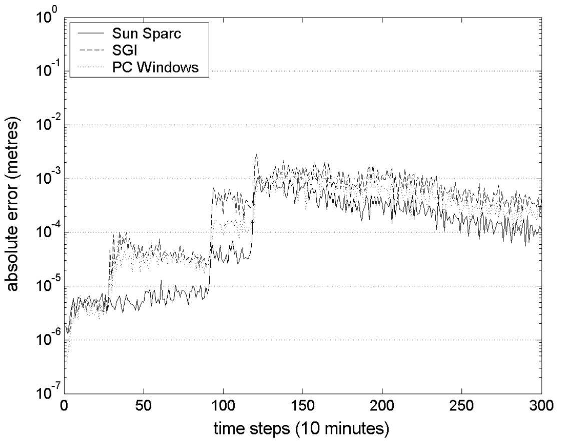
One of the problems is the difference in results when they have been computed on different computer architectures. We have run test case b01 on three different computer machines (Sun Sparc, SGI and PC). Figure 6.6 shows that the upper bounds for the relative error of the continuity equation as well as the relative error in ζ are comparable. We see that the relative error in ζ is only $\mathcal{O}(10^{-2})$ to $\mathcal{O}(10^{-4})$. The absolute error for a small problem (test case b01)



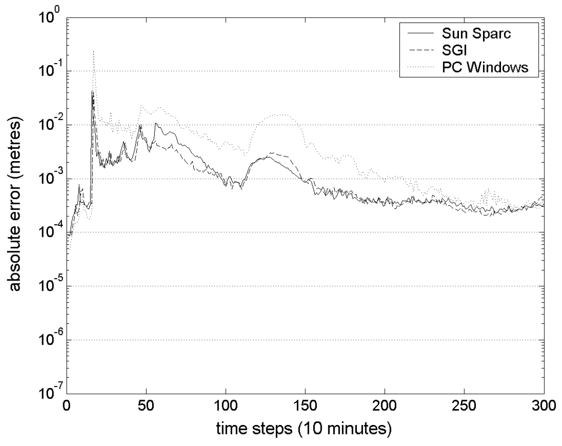
(a) (b)

Figure 6.6: The upper bounds on the relative error of the continuity equation (a) and the relative error in ζ for different architectures (test case b01).

increases to millimetres and then slowly drops, see Figure 6.7a. For a larger problem (test case b24) the absolute error increases rapidly to centimetres, then it slowly decreases to $\mathcal{O}(10^{-4})$ and remains fairly constant for all architectures. Also for larger problems the differences in error between several architectures is negligible.



(a)



(b)

Figure 6.7: The absolute error in ζ for test case b01 (a) and for test case b24 (b) for different architectures.

The original version of this test case has only Dirichlet boundary conditions on the water level ζ and consequently Neumann homogeneous boundary conditions on the perpendicular flow velocity component. The boundary conditions on ζ are homogeneous ($\zeta = 0$) except for the 'lower' boundary which has a tidal boundary condition on ζ. Due to these unnatural imposed boundary conditions high flow velocities occur near the corners where the tidal boundary condition change into homogeneous boundary conditions.

This version shows unstable behaviour (Figure 6.8) when running the simulation with a small difference (1 mm) in the initial water level. The differences in water level between the non-perturbed and the perturbed simulation run up to metres. However, in time the simulation stabilises and the differences drop below the magnitude of the perturbation. It is remarkable to see that, when computing in double precision, this decrease arises sooner than when computing in single precision. Also typical is that with a smaller time step the unstable behaviour still occurs, but stabilises sooner (in simulation time) and quicker than with a larger time step. Rounding errors seem to interfere with the spin-up time of the simulation. So, a preconditioning method might lessen this interference, see Section 6.7.

Note that the absolute error in ζ is influenced by the tide. When the tide is high, the error is a factor two to four higher than for low tide.

6.4 Test Case 3: Reservoir with Tidal Condition

In this second test case (WL reference: '07-chezy') we simulate a rectangular reservoir (20 km by 5 km) with closed boundaries along the long sides and open boundaries on the short sides. On one of the open boundaries the homogeneous water level condition is imposed and on the other open boundary a tidal condition is defined. The vertical grid is chosen counter-intuitively. The relative layer thickness, $\Delta\sigma_k$, is set to

$$0.01, 0.1, 0.09, 0.08, 0.06, 0.05, 0.04, 0.03, 0.02, 0.02, 0.03, 0.04, 0.05, 0.06, 0.08, 0.09, 0.1, 0.01$$

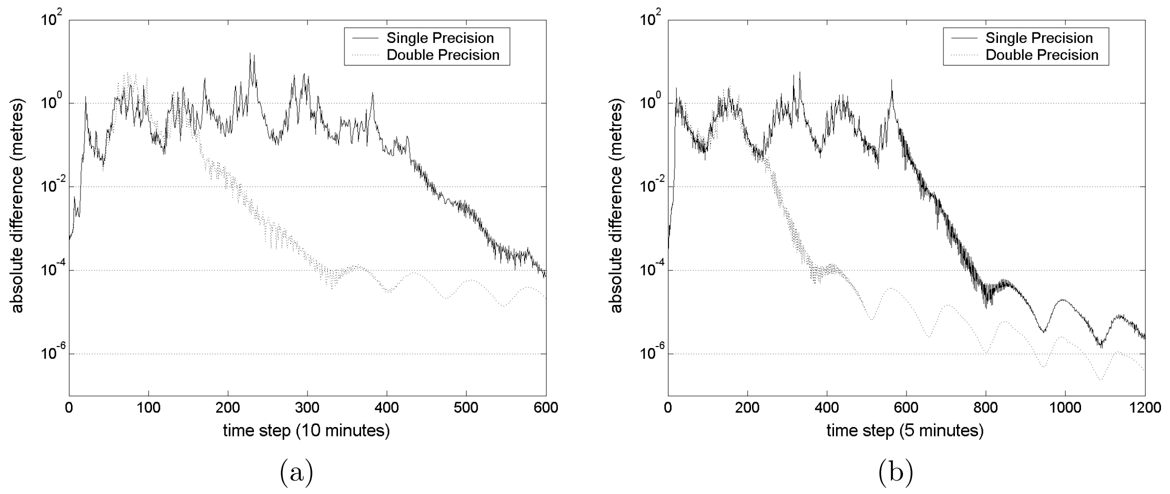


Figure 6.8: Absolute difference in ζ between a perturbed and an unperturbed problem with a time step of 10 (a) and 5 minutes (b). The perturbation is 1 mm on the initial water level ('original' test case 2). Compare with Figure 6.27.

for $k = 1$ to $k = 20$ respectively. This results in rather peculiar flow data at certain points in time, see Figure 6.9. The vertical velocity component in this figure is scaled up by a factor of approximately 1000. As a reference we added Figure 6.10 which shows an example of "normal" flow data.

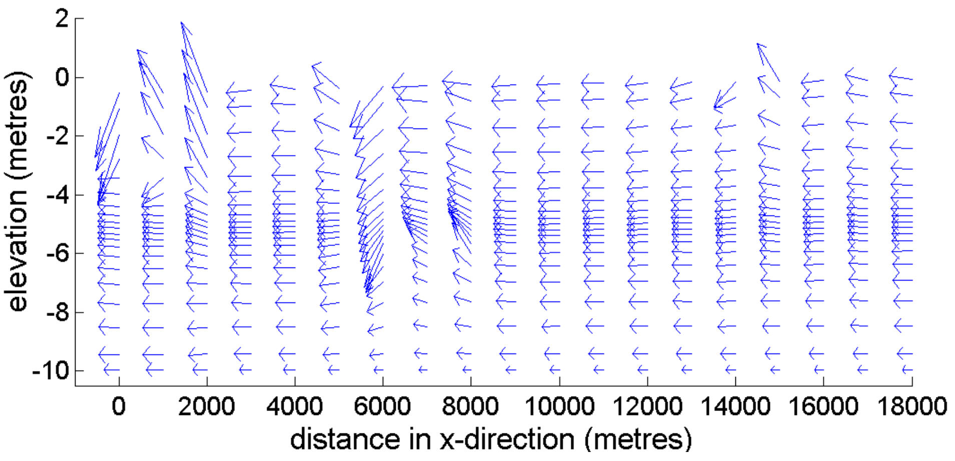


Figure 6.9: Cross-section of a peculiar flow profile at a certain time step. (PC Windows, time step 183)

The horizontal grid size is 1 km. The uniform water depth is 10 m. Wind is simulated in western (left) direction with a constant speed of 10 m/s. Several versions of this test case have been simulated. In Table 6.4 the specifications of the different versions are given. The time step is set to 10 minutes.

This test case is used to test the influence of the σ-layer distribution. Furthermore the results are compared when choosing different turbulence closure models.

We have tested stability for this test case also by disturbing the initial water level with 1 mm. Figure 6.11 shows unstable behaviour for the test case with a non-equidistant σ-layer

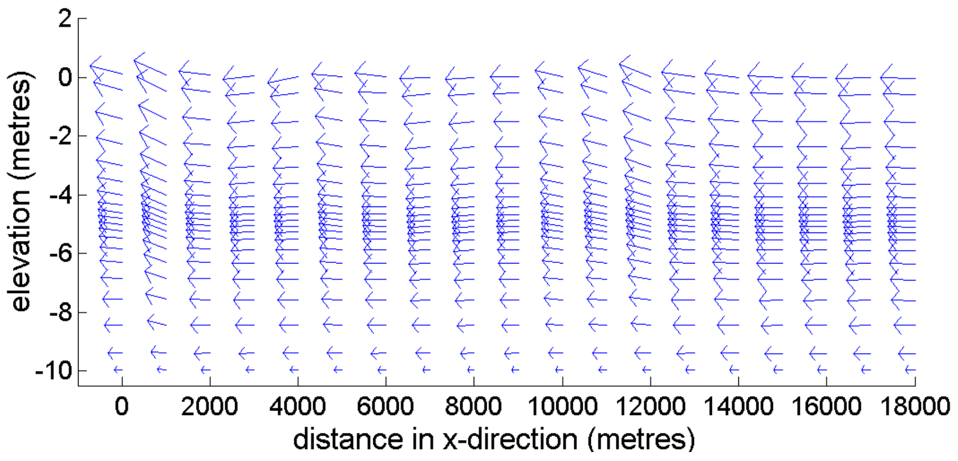


Figure 6.10: Cross-section of a normal flow profile. (PC Windows, time step 203)

Table 6.4: Specifications of the different versions of test case 3.

specification	test case version							
	c01	c02	c05	c07	c08	c09	c10	c12
layer distribution	equid.	equid.	non-equid.	equid.	equid.	equid.	equid.	equid.
# σ-layers	1	20	20	2	3	5	10	50

distribution (c05) for single as well as double precision. For the test case with equidistant σ-layer distribution (c02) a spin-up time of about 3500 minutes (roughly 2 days) is required depending on the desired accuracy. This clearly shows that the kind of σ-layer distribution determines the stability of the simulation.

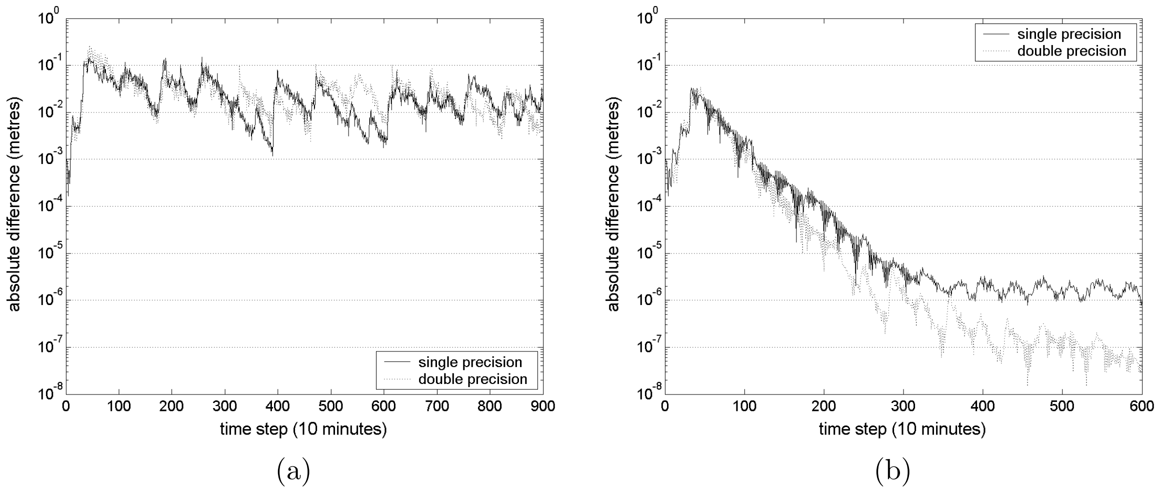


Figure 6.11: Absolute difference in ζ between a perturbed and an unperturbed problem for test case c05 (a) and test case c02 (b). The perturbation is 1 mm on the initial water level.

Number of σ-layers

Another interesting quantity in this respect is the number of σ-layers. An increase of the number of σ-layers would produce larger matrices for the explicit momentum equations, so we would expect larger condition numbers and less accuracy. Remember that the continuity equation is depth-integrated. So, the number of σ-layers is expected to have little effect on the condition numbers of the continuity equation.

The Figures 6.12a through 6.13b are constructed with data from the test cases c01, c09, c10, c02 and c12, all of which have an equidistant layer distribution representing 1, 5, 10, 20 and 50 layers, respectively. From them can be concluded that indeed the residue, the condition number and the upper bound for the relative error for the continuity equation are constant with respect to the number of σ-layers (when no row-scaling is applied). However, Figure 6.14a shows a significant increase in the absolute error in ζ. We will come back to this later.

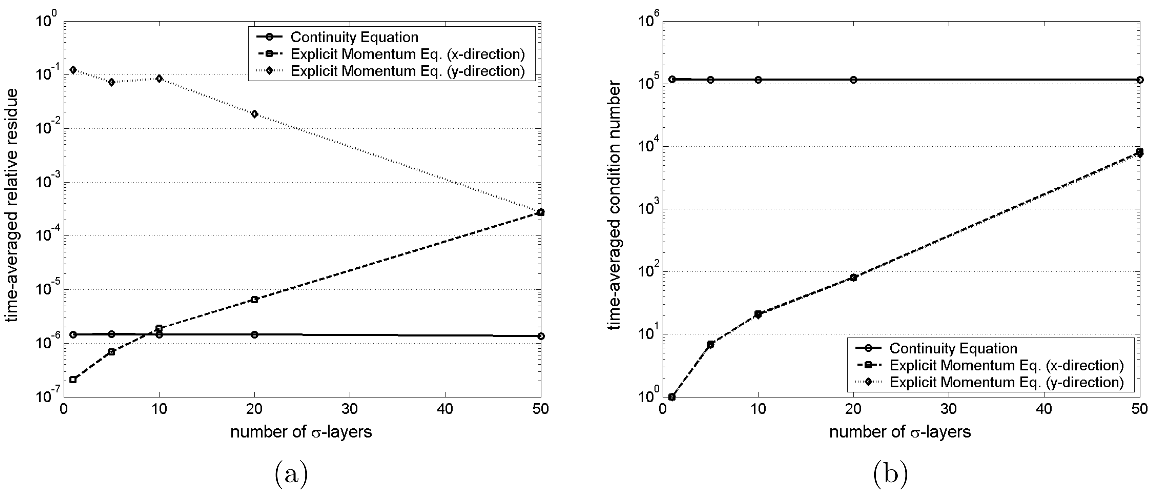


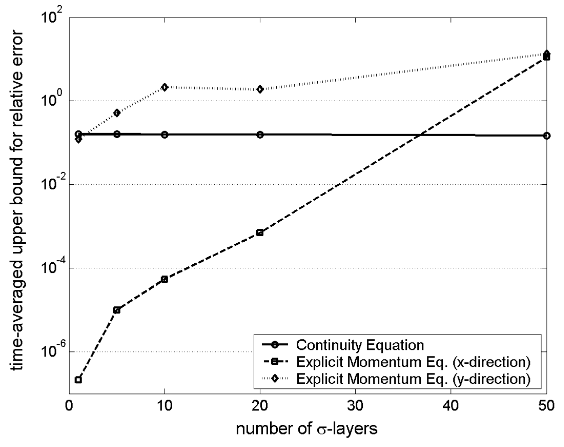
Figure 6.12: The time-average relative residue (a) and the time-averaged condition number (b) for an increasing number of σ-layers (test case 3).

Due to the ADI scheme the explicit momentum equation is being solved alternately in the x- and y-direction. For this test case we have not averaged the data in both directions, because they are rather different. From analysis of the data for the x-direction (dashed lines in Figures 6.12 and 6.13) we conclude that the residue, the condition number and the upper bound for the relative error increase *exponentially* with the number of σ-layers. This coincides with the increase of the absolute error in the flow velocity component v, see Figure 6.14. The number of iterations is however constant.

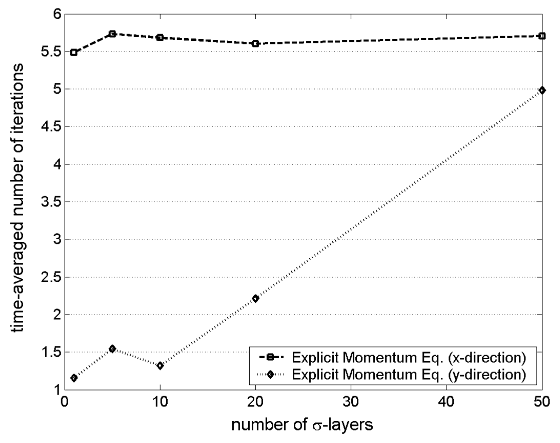
The graphs for the explicit momentum equation in y-direction (dotted lines in Figures 6.12 and 6.13) shows rather peculiar data. The flow in y-direction v is almost zero, as is expected due to the given geometry. The explicit momentum equation is solved by means of an iterative method with the stopping criterion which implies that in absolute sense the result must be accurate in $\mathcal{O}(10^{-6})$. Thus, if v itself should be of $\mathcal{O}(10^{-7})$ for example, then the relative error for v will explode. This also results in a large residue.

Two and three σ-layers

The data for the test cases c07 and c08, respectively two and three σ-layers, have been left out on purpose. The results from and the data on the explicit momentum equation in y-direction

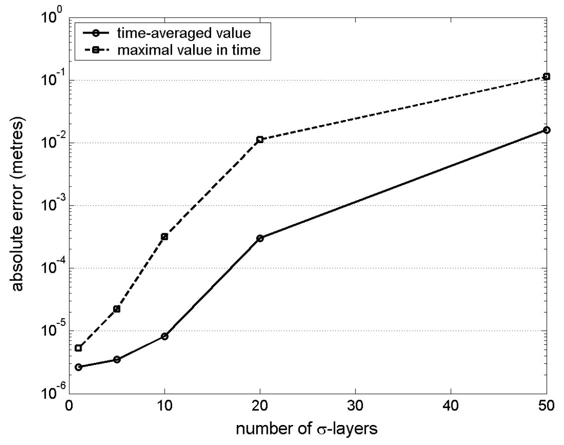


(a)

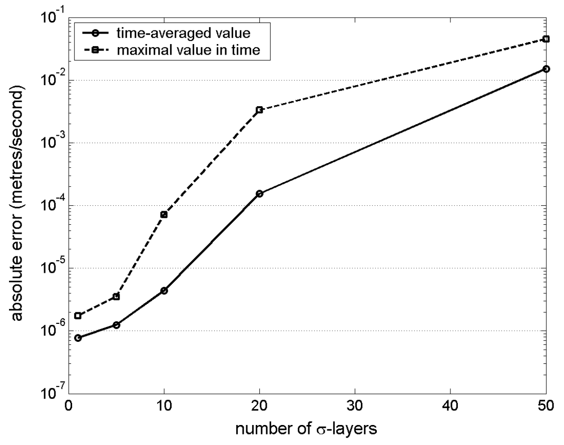


(b)

Figure 6.13: The time-averaged upper bound for the relative error (a) and the time-averaged number of iterations (b) for an increasing number of σ-layers (test case 3).



(a)



(b)

Figure 6.14: Time-averaged absolute error in ζ (a) and in the flow velocity in y-direction v (b) for an increasing number of layers (test case 3).

appear to be in error. Table 6.5 shows the results for these test cases. It is striking to see that,

Table 6.5: Time-averaged data on the explicit momentum equation in y-direction and the absolute error in the water level ζ and in the flow velocity component v for test case c07 and c08.

test case	explicit momentum equation			absolute error	
	relative residue	upper bound rel. error	number of iterations	ζ	v
c07	10^{-3}	10^{-2}	3.3	10^{-4}	10^{-4}
c08	10^{-6}	10^{-5}	5.1	10^{-2}	10^{-2}

for these test cases, with two layers but even more with three layers, more iterations are needed than with one, five or ten layers. Consequently, this results in a lower residue and a lower upper bound for the relative error. Thus we would expect an improvement in the absolute error for v, but it has in fact worsened. The absolute error is of $\mathcal{O}(10^{-2})$ for test case c08, while for test case c01 (one layer) and test case c09 (five layers) it is of $\mathcal{O}(10^{-6})$ (see Figure 6.14b). Moreover, even the absolute error in ζ is larger.

We remind the reader that for the criterion (absolute difference in v must be smaller than 10^{-6}) to be met for three σ-layers more iterations in y-direction are needed to solve the explicit momentum equation in v than for five, ten or twenty σ-layers. This might be induced by a larger value for v. This seems to be the case as we look at the flow profiles in the Figures 6.15 and 6.16. A regular flow profile is shown in Figure 6.17. Furthermore, the profiles are not similar. For five layers and more these differences have not occurred.

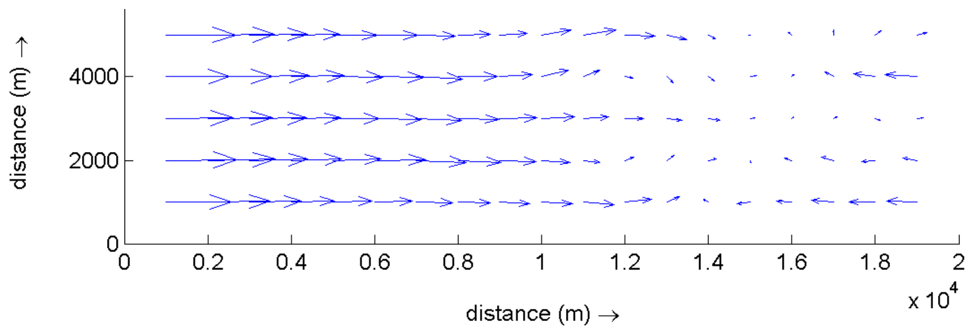


Figure 6.15: Flow profile for the upper layer of test case c08 (three σ-layers) in single precision on time step 568.

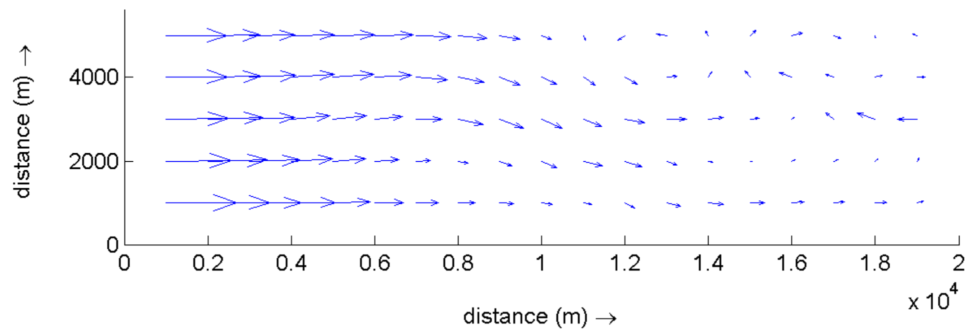


Figure 6.16: Flow profile for the upper layer of test case c08 (three σ-layers) in double precision on time step 568.

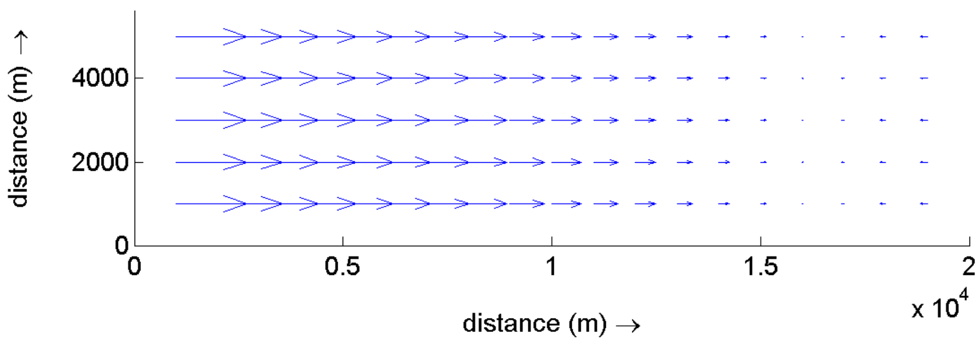


Figure 6.17: Flow profile for the upper layer of test case c09 (five σ-layers) in single precision on time step 568.

Application of row scaling does not show any improvement (no results shown). Thus both machine precision and row scaling have no positive effect on the results for the test cases with two or three σ-layers. This lets us to believe that the discretisations are not adequate or that another process within the flow computation causes these results. If we simulate the test cases with the $k - L$ turbulence model instead of the $k - \epsilon$ turbulence model, the results can be considered "normal" (see Figure 6.18). Also note that absolute errors when using the $k - L$ turbulence model are much smaller than with the $k - \epsilon$ turbulence model. Furthermore, the dependence of the absolute errors on the number of σ-layers is much smaller.

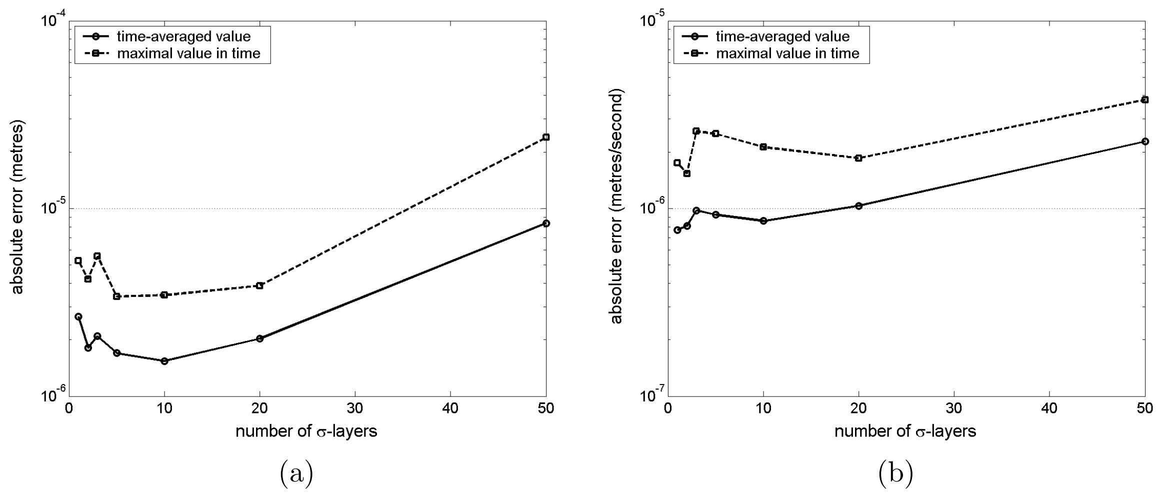


Figure 6.18: Time-averaged absolute error in ζ (a) and in the flow velocity in y-direction v (b) for an increasing number of layers with the $k - L$ turbulence closure model (test case 3). Compare with Figure 6.14.

6.5 Test Case 4: Navigational Channel

The navigational channel in a river or estuary is usually the deepest part of the water flow with the highest flow velocity. The water depth in the navigational channel may be up to 10 times larger than in other parts of the river or estuary. In this test case we will simulate rectangularly shaped reservoirs with deep navigational channels.

For this test case two different version have been simulated. The first one is a schematised reservoir of Keeten-Volkerak. The reservoir is 49 km long and 18 km wide and subjected to tidal influences at an open boundary, situated at one of the short sides. The depth is chosen as a sine function, varying between -15 and -1 m. Due to the symmetry of the problem the actual test case is a reservoir of 9 km wide, see Figure 6.19. (version name: n02).

Figure 6.20 shows an upper bound for the relative error made in the continuity equation of 10%. For the explicit momentum equation this upper bound is highly variable as is its condition number and residue. The absolute error in ζ for single precision is in $\mathcal{O}(10^{-4})$.

Another test case is a WL test case, namely 'Benqué' (test cases n03, n11, n12, n13 and n14). It is a small rectangular shaped reservoir (3.6 km by 7.2 km) with a navigational channel, that meanders in relation to the grid, see Figure 6.21. The white area is the navigational channel (depth: 13 m) and the black area has a water depth of 6 m.

The horizontal grid size is 300 m. It is a two-dimensional problem in the sense that only one σ-layer is defined. The time step is a half minute. With this test case we want to test the

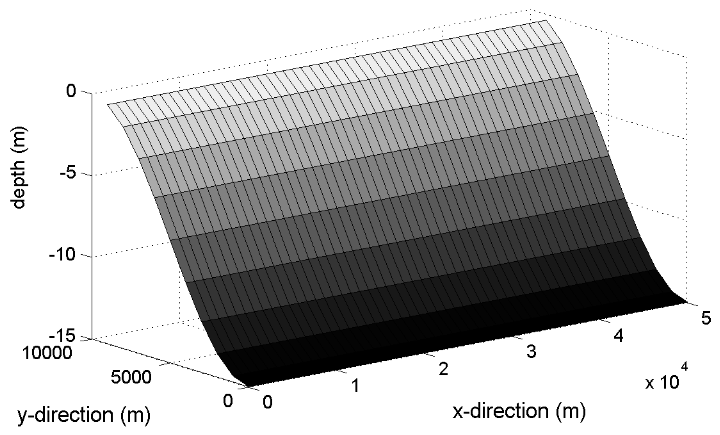


Figure 6.19: The bed topography of test case n02 (schematised Keeten-Volkerak).

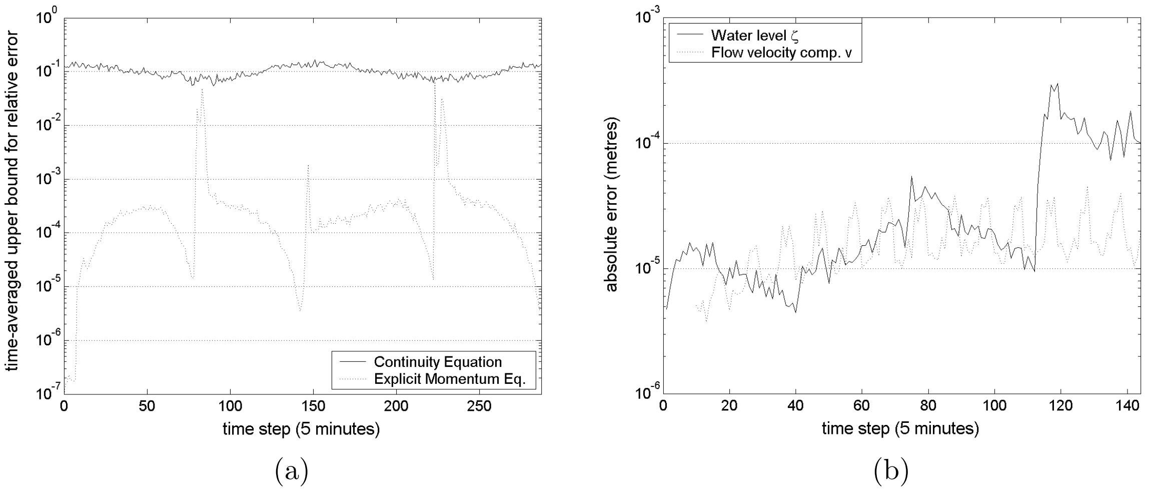


Figure 6.20: Upper bound for the relative error of the continuity equation (a) and the absolute error in ζ for the schematised Keeten-Volkerak (test case n02).

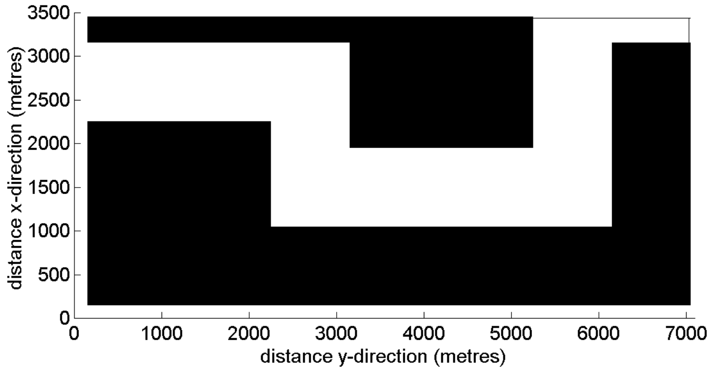


Figure 6.21: The navigational channel meanders in the Benqué test case.

transition between deep and shallow water. Therefore we will look at different versions of this test case, see Table 6.6.

Table 6.6: Specifications of the different versions of the Benqué test case.

specification	test case version				
	n03	n11	n12	n13	n14
navigational depth	13 m	20 m	25 m	30 m	35 m
ratio	1 : 43	1 : 21	1 : 16	1 : 12	1 : 10

A sudden increase in depth between two neighbouring grid points does not influence the residue or the upper bound for the relative error of the continuity equation very much. Very steep slopes (larger than 1:10) may cause a lower accuracy in ζ. For the explicit momentum equation the angle of the slope has little to no effect on the residue, the condition number and the upper bound for the relative error. These quantities are respectively of $\mathcal{O}(10^{-7})$, $\mathcal{O}(10^{0})$ and $\mathcal{O}(10^{-7})$.

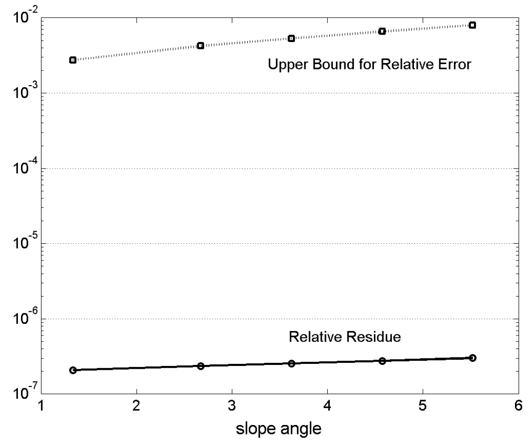


Figure 6.22: The time-averaged relative residue and the time-averaged upper bound for the relative error of the continuity equation (Benqué test case).

6.6 Test Case 5: Long Narrow Reservoir

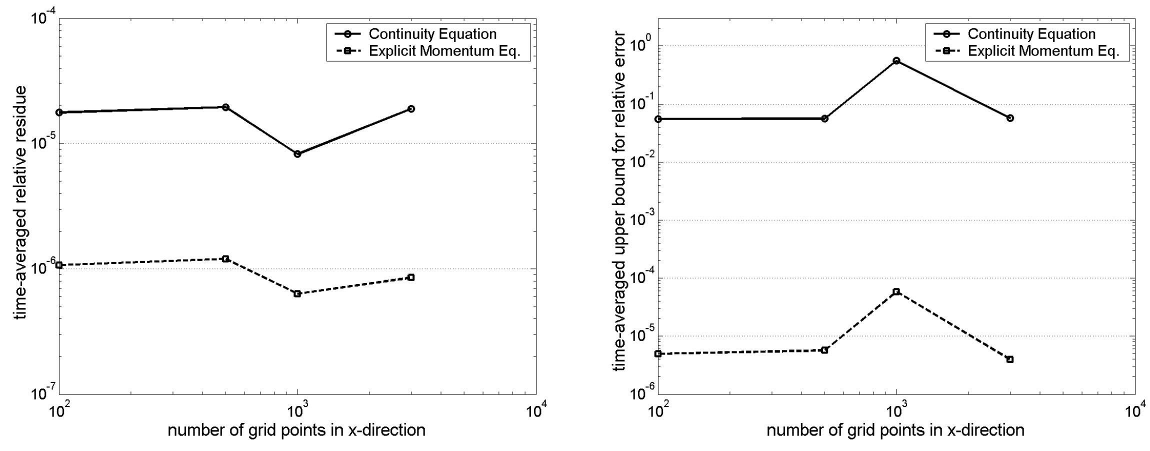
This test case is added in order to demonstrate how Delft3D-FLOW manages large geometries and thus large matrices. The reservoir is narrow but long. On one open boundary a water level boundary condition (tidal, amplitude: 0.5 m) is imposed and on the other side a current boundary condition (velocity: 1.2 m) is imposed. The long boundaries are 'closed'. The problem is two-dimensional, because there is only one σ-layer.

For this test case four versions, differing in size only, have been simulated. In Table 6.7 the specifications of the different versions are given. In all versions only one σ-layer is defined and the horizontal grid size is 100 m. The time step is 5 minutes.

We would expect an increase in residue and for the upper bound of the relative error comparably with the results of test case 2 in Figure 6.3. This is however not the case. A most probable explanation is that the test case is modelled with a single σ-layer ensuring unconditional stability (see Section 5.2) and high accuracy (see test case 3).

Table 6.7: Specifications of the different versions of test case 5.

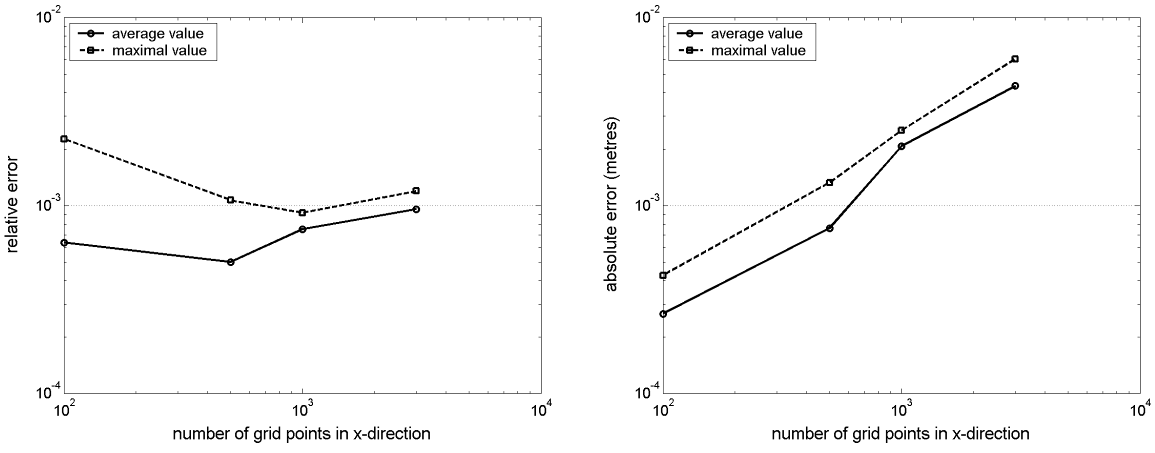
specification	test case version			
	l02	l52	l03	l04
reservoir size	10 $km \times 0.5$ km	50 $km \times 0.5$ km	100 $km \times 0.5$ km	300 $km \times 0.5$ km
grid size	100×5	500×5	1000×5	3000×5



(a) (b)

Figure 6.23: Upper bound for the relative error of the continuity equation (a) and the absolute error in ζ for the schematised Keeten-Volkerak (test case n02).

Figure 6.24b shows that increasing the number of grid points in one direction up to 3000 causes the absolute error in ζ to increase up to millimetres. For large problems (above 400 grid points in both directions) the absolute error is also in order of millimetres.



(a) (b)

Figure 6.24: Relative (a) and absolute error (b) in ζ for the Benqué test cases.

6.7 Row Scaling

As shown in Section 4.7 row scaling leads to a drastic reduction of the condition number of the continuity equation. In this section we will analyse the influence of row scaling on two test cases, viz. the test cases 2 and 3.

Test Case 2

According to Lemma 4.6 the upper bound for the relative error will also reduce (see Figure 6.25a) so a higher accuracy can be guaranteed. The actual proof for the improvement of the solution produced with row scaling is the reduction of the relative residue (see Figure 6.25b). However,

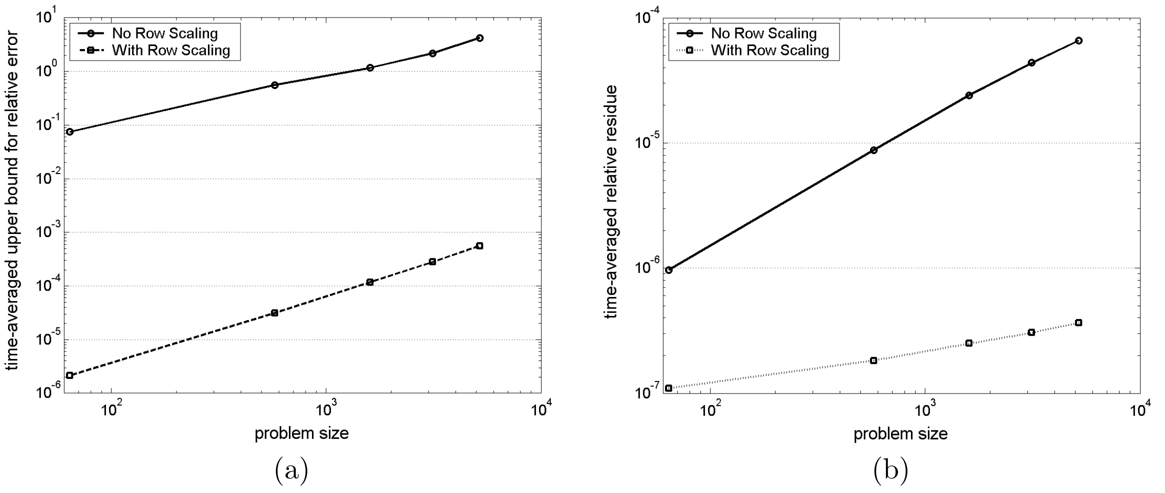


Figure 6.25: The time-averaged upper bound for the relative error (a) and the time-averaged relative residue (b) of the continuity equation when increasing the problem size for test case b01.

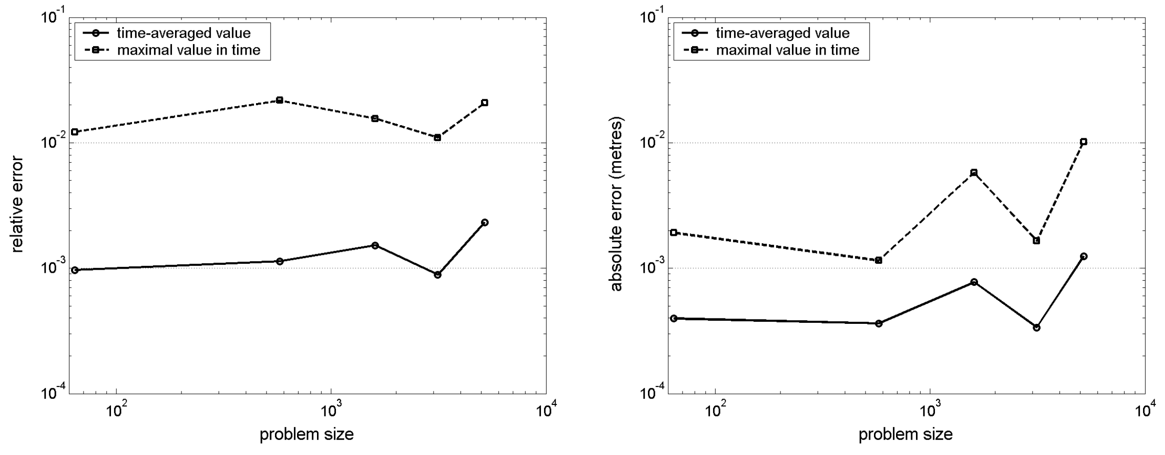
extrapolation of the the dotted line in Figure 6.25a still only ensures a relative error of the continuity equation between 1% and 10% for problem sizes beyond 10^5 grid points (and 10 σ-layers). For the explicit momentum equation the improvement is by far not so drastic: the condition number hardly decreases. This results in equivalent residues and upper bounds for the relative error if no row scaling is applied.

If we analyse the graph of the relative and absolute error in ζ, when row scaling is applied, we would expect a reduction with respect to the error, when row scaling is not applied. Figure 6.26 shows, however, otherwise. The errors are comparable to those in Figure 6.4.

In Section 6.3 on page 50 we briefly discussed the original version of this test case which has open and not closed boundaries. The absolute difference in ζ for this test case was unacceptably high for a considerable period of the simulation time. This period reduced when decreasing the time step or compute in double precision. The implementation of row scaling also leads to a reduction of this time period, especially in the case of single precision computation and a time step of ten minutes (see Figure 6.27a).

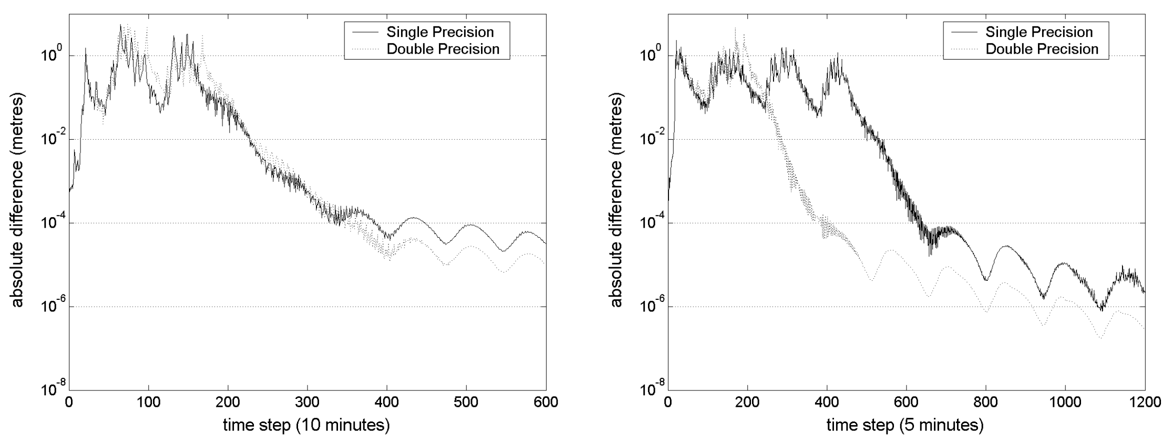
Test Case 3

For test case 3 more or less the same results arise when row scaling is applied. Again the upper bound on the relative error decreases (see Figure 6.28a), which leads to more accurate results. This is confirmed by a decrease in the relative residue (see Figure 6.28b). In contrary with



(a) (b)

Figure 6.26: The time-averaged relative error (a) and the absolute error (b) in the water level ζ when increasing the problem size for test case 2. Compare with Figure 6.4.

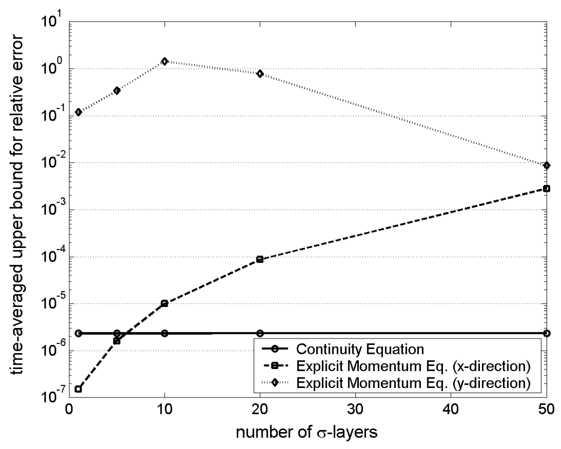


(a) (b)

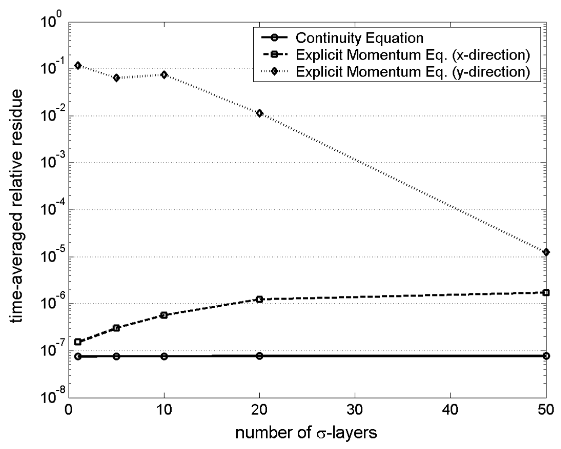
Figure 6.27: Absolute difference in ζ between a perturbed and an unperturbed problem with a time step of 10 (a) and 5 minutes (b). The perturbation is 1 mm on the initial water level and row scaling is applied. Compare with Figure 6.8c and Figure 6.8a, respectively.

test case 2, also the explicit momentum equation shows considerable improvement due to row scaling.

But also for this test case the absolute errors in the water level ζ and the flow velocity component v do not show any improvement if row scaling is applied. Possible reasons for this are most likely other processes which play a role in the numerical computation. However, if we choose a different turbulence closure model, the $k - L$ turbulence model instead of the $k - \epsilon$ turbulence model, row scaling still does not influence the absolute errors in ζ and v. The results acquired with the $k - L$ turbulence model has absolute errors for the equations as well as for the variables ζ and v in $\mathcal{O}(10^{-6})$, which is just above the machine precision. So, row scaling (or any other adjustment for that matter) is naturally not able to improve the accuracy.

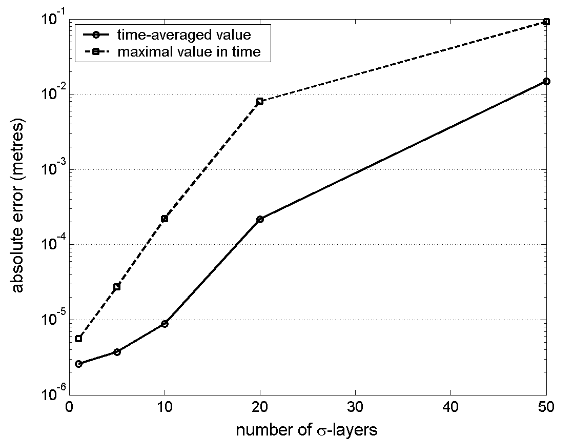


(a)

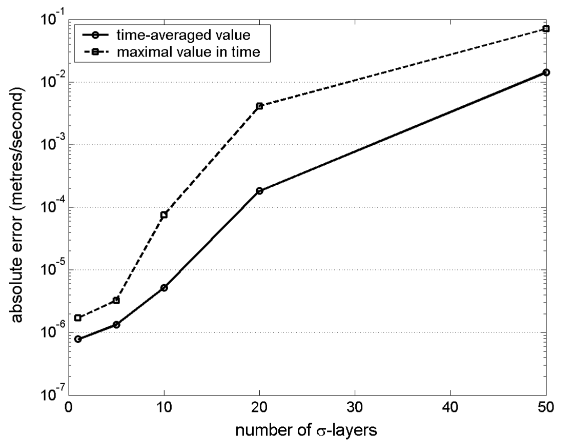


(b)

Figure 6.28: The time-averaged upper bound (a) and the time-averaged relative residue (b) for an increasing number of σ-layers when row scaling is applied. Compare with Figures 6.13a and 6.12a, respectively.



(a)



(b)

Figure 6.29: Time-averaged absolute error in ζ (a) and in the flow velocity in y-direction v (b) for an increasing number of layers when row scaling is applied. Compare with Figure 6.14.

6.8 Stopping Criteria

The explicit momentum equation (explicit with respect to the water level ζ, but implicit with respect to the flow velocity) is solved with a Gauss-Jacobi iterative scheme. An iterative process ends if its stopping criterion is met. The stopping criterion presently used in Delft3D-FLOW demands that the absolute difference between consecutive iterands is at most 10^{-6}. If the convergence speed is sufficient high, then this iteration will stop at the given criterion. Figure 6.30 shows that the contraction factor increases with the problem size. Extrapolation of the graph shows that for problems over 10^5 grid points the contraction factors assume high values. To avoid premature termination of the iteration the criterion must be more strict, e.g. according to equation(6.6). This stopping criterion requires at least two iterations due to calculation of the contraction factor. Other parameters, like time step, bed topography, number of σ-layers,

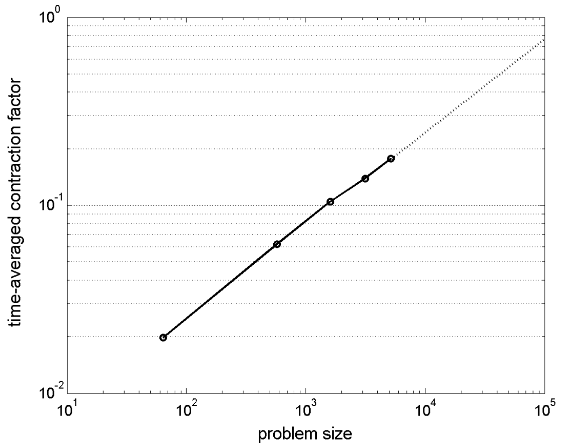


Figure 6.30: Time-averaged contraction factor for the explicit momentum equation when increasing the problem size (test case b01). The dotted line is an extrapolation of the graph.

machine precision and row scaling, seem to have far less influence on the contraction factor.

We have tested the three different stopping criteria mentioned in Section 6.1 on test case 3 with a varying number of equidistant σ-layers. In Figure 6.31 the relative residues for the explicit momentum equation is plotted for the x as well as the y-direction. We remind the reader that the residual criterion is met if the residue is smaller 10^{-4}. This is confirmed by the graphs. The critical value for the residual criterion is fairly high, but in single precision not all test cases could converge any further. The other two stopping criteria show comparable

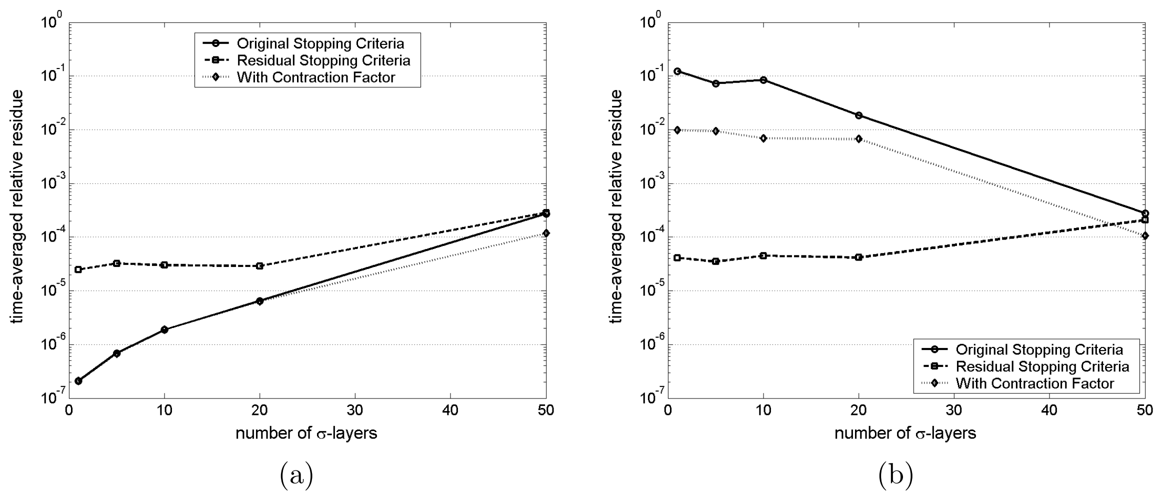
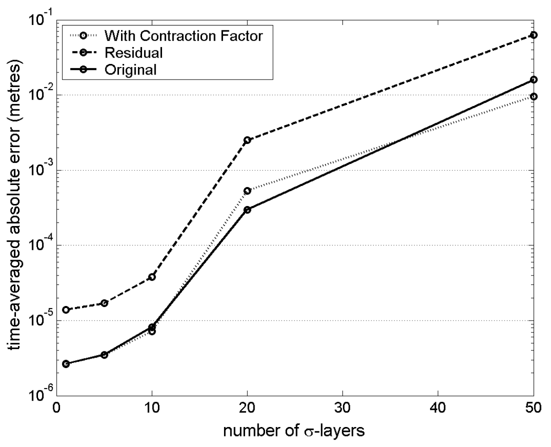


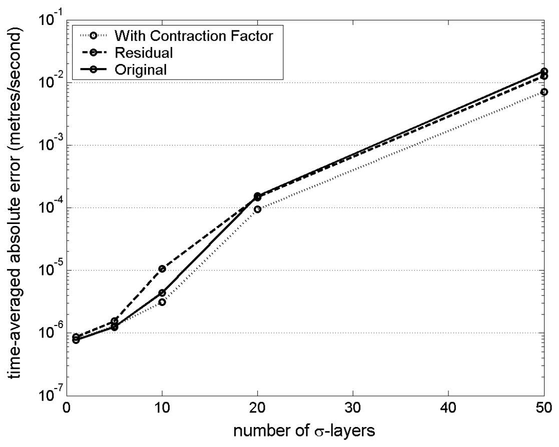
Figure 6.31: Time-averaged relative residue for the explicit momentum equations in x-direction (a) and in y-direction for an increasing number of σ-layers and different stopping criteria (test case 3).

residues. The iterative process with original stopping criterion only needs one iteration for the explicit momentum equation in y-direction, but in combination with the contraction factor at least two iterations are required. Hence, the difference in Figure 6.31b.

The different stopping criteria have little effect on the absolute error in ζ and v (see Figure 6.32).



(a)



(b)

Figure 6.32: Time-averaged absolute error in ζ (a) and in the flow velocity in y-direction v (b) for an increasing number of layers and different stopping criteria (test case 3).

6.9 Test Case 6: Westerschelde

The Westerschelde is an interesting test case, since the river bed is very uneven. At some points the water depth measures 40 metres, while at other points it is only a few meters, see Figure 6.33. The width varies between three and six kilometres. This test case is in fact nothing more than a large, existing example of test case 4.

This test case simulates the eastern part of the Westerschelde up to the Belgian border. In this area the estuary is very uneven, measuring water depths between only a few and 40 metres, see Figure 6.33. The width varies between three and six kilometres. This test case is used to analyse the accuracy for a large simulation.

This test case is set up with only one σ-layer, meaning that numerically speaking it is unconditionally stable. The horizontal grid is curvilinear. The initial water level is set to three metres. There are two open boundaries. At the west end (sea side) tidal conditions are imposed and at the east end (river side) a discharge boundary condition is imposed. The time step is set to 0.5 minutes.

The results in Table 6.8 show that only the continuity equation profits from row scaling and a higher machine precision. As is shown in Figures 6.12 and 6.13, with only one σ-layer the explicit momentum equation is solved very accurately. So, row scaling will have hardly any influence and a higher machine precision has no effect because the stopping criterion is set to 10^{-6}.

Table 6.8: Time-averaged data on the continuity equation and the explicit momentum equation for the Westerschelde test case.

precision; row scaling	continuity equation			explicit momentum equation		
	relative residue	condition number	upper bound rel. error	relative residue	condition number	upper bound rel. error
single; no	10^{-7}	10^{5}	10^{-2}	10^{-7}	10^{0}	10^{-7}
double; yes	10^{-16}	10^{2}	10^{-14}	10^{-7}	10^{0}	10^{-7}

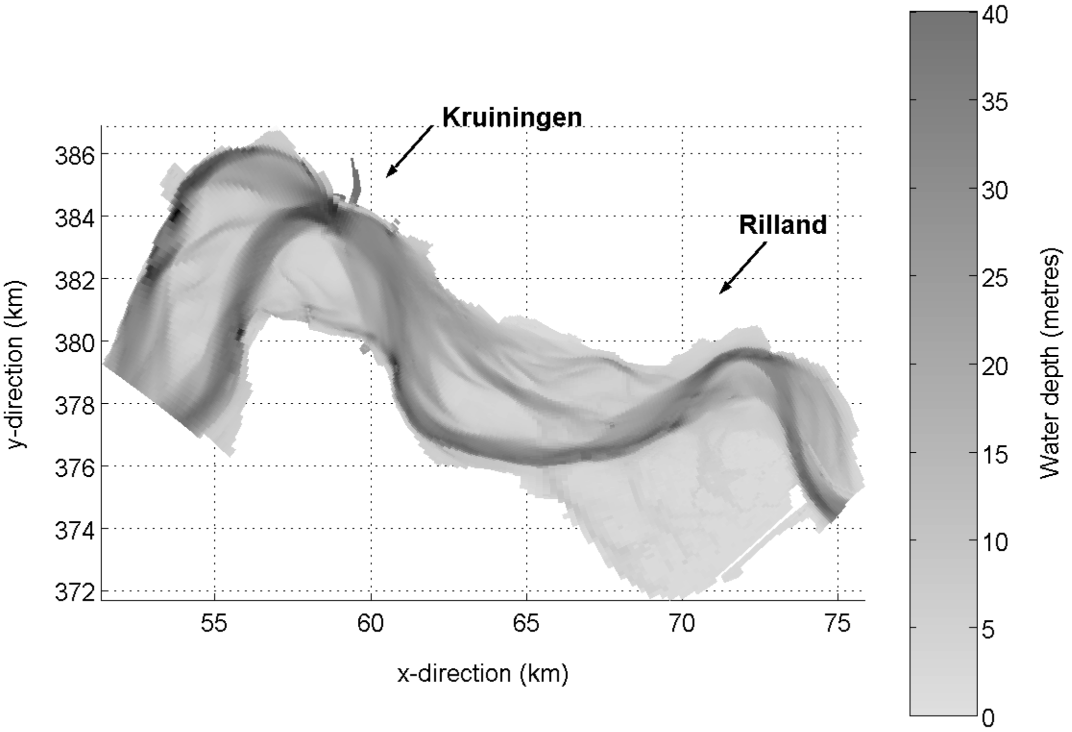


Figure 6.33: Bed topography of the Westerschelde.

Figure 6.34 shows a large error for both quantities. Furthermore, the absolute error in the water level corresponds with the frequency of the tidal condition. We have seen this before in Figure 6.8.

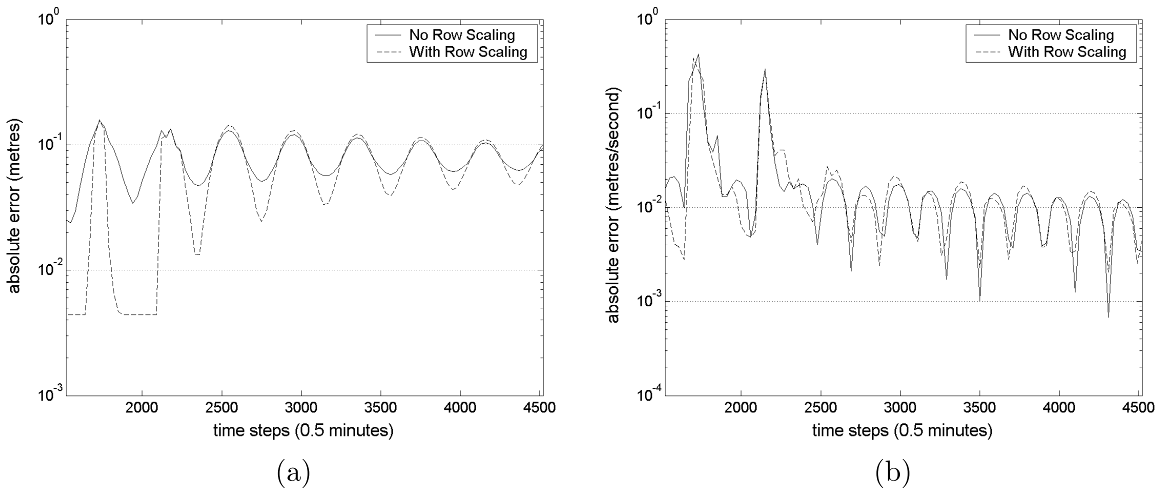


Figure 6.34: The absolute error in ζ (a) and in the flow velocity in y-direction v (b) for the Westerschelde. The first 1520 time steps are not shown.

But the tide does not explain the large absolute error in ζ. In Figure 6.35 the absolute error at a certain time step is plotted for the whole area. It is clear that the large error is very local and is most likely not due to cumulation of rounding errors. The peaks in the absolute error occur in very shallow areas (water depth is around 10 cm). This means that only small changes

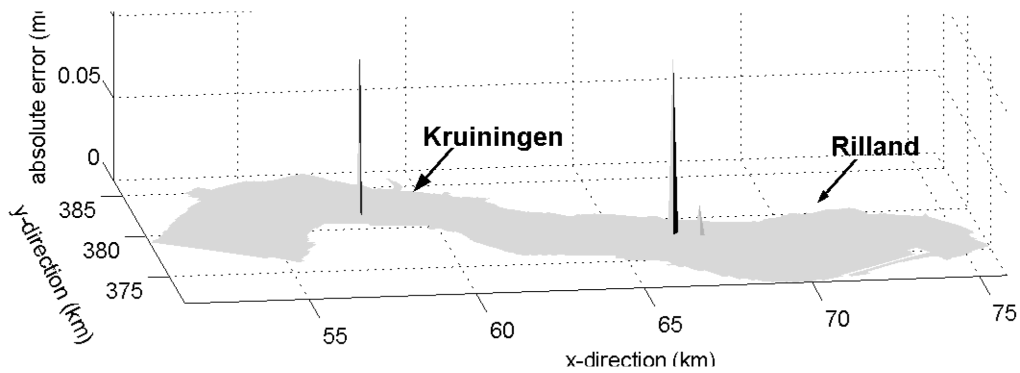
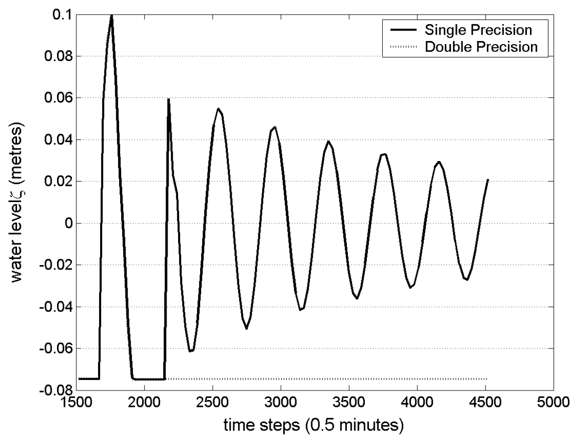
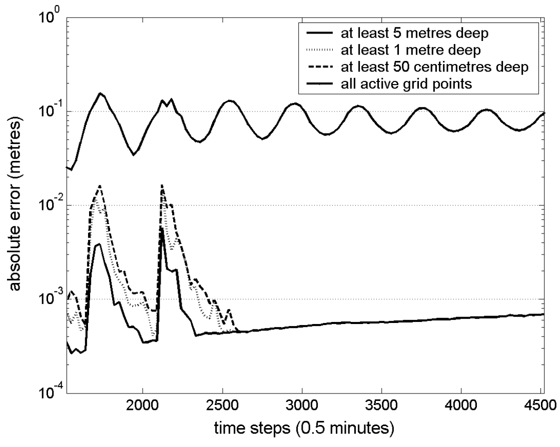


Figure 6.35: The absolute error in ζ for the Westerschelde at time step 3020.

to the water level will cause recession. Further inspection of the 'right' high peak in Figure 6.36a shows that this occurs. The recession and flooding criteria in Delft3D-FLOW are set at certain water depths ($\zeta + d$). So, small changes in the water level ζ can cause an area to recede or flood. In our case the water level in grid point (174, 115) remains dry when computed in double precision, but it is flooded in single precision from a certain point in time.



(a)



(b)

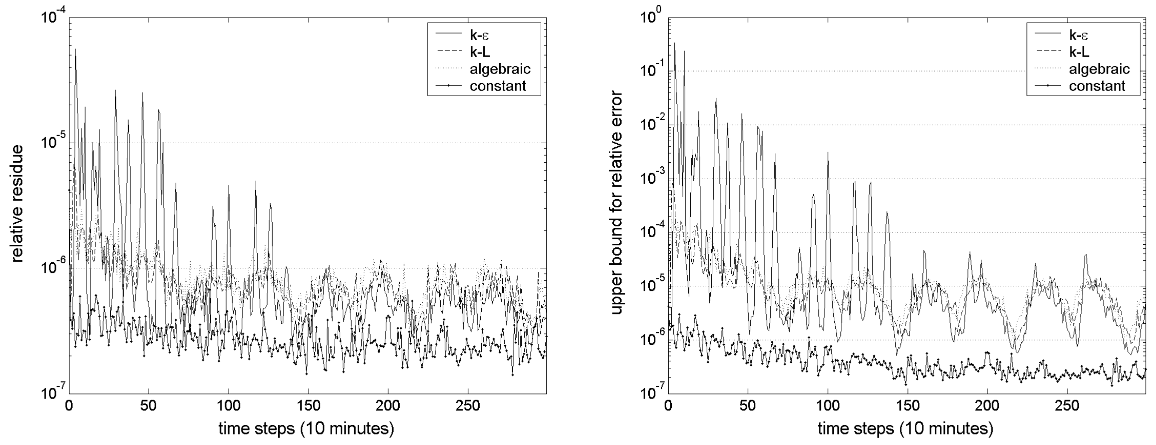
Figure 6.36: The water level ζ at grid point (174, 115) (a) and the absolute error in ζ taking into the water depth (b) for the Westerschelde.

Figure 6.36 shows that the absolute error in ζ in the deeper areas is $\mathcal{O}(10^{-3})$.

6.10 Turbulence Closure Model

In Figure 6.18 we have seen that the absolute errors in ζ and v are considerably smaller when the $k - L$ turbulence model is used instead of the $k - \epsilon$ turbulence model. In Figure 6.37 the $k - \epsilon$ model shows the most erratic behaviour for the relative residue and the upper bound for the relative error of the explicit momentum equation. The model with a constant value for the vertical eddy viscosity shows the best results in accurate sense with respect to the explicit momentum equation.

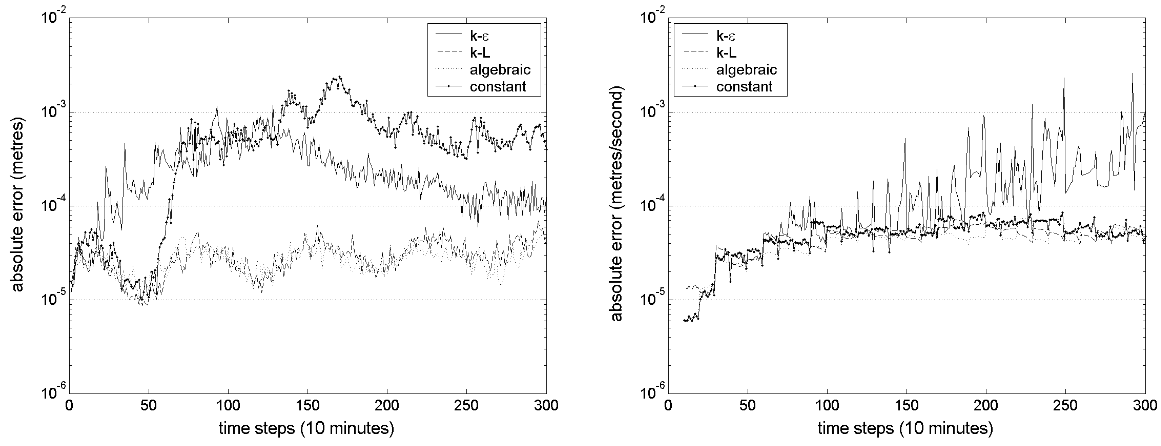
Figure 6.38 shows that the absolute error in ζ and u with the $k - \epsilon$ model is larger than with the other models, except for the constant value model, for which the absolute error in ζ



(a) (b)

Figure 6.37: The relative residue (a) and the upper bound for the relative error (b) of the explicit momentum equation for test case b21.

is the largest. From this we conclude that the use of the $k - \epsilon$ turbulence model leads to less accurate results than the other models with the exception of the constant value model.



(a) (b)

Figure 6.38: The absolute error in the water level ζ (a) and the flow velocity component u (b) for test case b21.

6.11 Stability

In Chapter 5 we proved unconditional numerical stability of the two-dimensional shallow-water equations. The consequence is that small errors in simulations in Delft3D-FLOW with a single σ-layer will dim. During the attempt to perform a similar stability analysis on the three-dimensional shallow-water equations complex matrix equations arose, which complicated the analytical approach. For that reason the amplification matrix (and its largest eigenvalue) of the complex equations is computed by means of a Matlab program, see Appendix D.

Early runs showed that for all the two-dimensional cases the largest eigenvalue of the amplification matrix is one, affirming the unconditional stability. But for some three-dimensional cases larger eigenvalues were not uncommon. We will first discuss the input parameters for the Matlab program and then show how they influence the largest eigenvalue.

The Matlab program uses a linearised version of the three-dimensional shallow-water equations. So, as opposed to Delft3D-FLOW it also requires input data with respect to the total water depth, the flow velocity profiles in the vertical for both directions, the relative vertical velocity, ω, and the vertical eddy viscosity. The latter two need to be given for every layer separately. The fact that the Matlab program requires more input data than Delft3D-FLOW makes the program less useful for an a priori stability check of a simulation. However, it does allow us to check the stability for more unusual situations.

We have used the flow results of test case c05, which shows unstable behaviour in Figure 6.11, and test case c02 to set up the necessary input data. The resulting largest eigenvalues are 1.0104 and 1.0065, respectively. According to the Matlab program both test cases would be unstable. It should be noted that the program does not give the answer to the question when a simulation will be stable or not. Though it can be used as an comparative tool to see, what changes or differences makes a simulation more stable or just less stable.

We have tested several "unnatural" settings on the Matlab program to gain some insight regarding the influence of the input data on the stability.

Flow velocity profiles An uniform horizontal velocity profile in the vertical direction results in a stable system. The slightest variation in the profile makes it unstable. Furthermore, a larger horizontal velocity results in a less stable or more unstable system. This can be compared with the cell-Péclet number, see Equation (6.7). A larger cell-Péclet number results in a less stable system

Relative vertical velocity The relative vertical velocity ω is set at most in $\mathcal{O}(10^{-4})$. If we neglect the vertical advective term by setting $\omega = 0$, the stability is hardly influenced.

Vertical eddy viscosity Computation of the vertical eddy viscosity can cause inaccurate results. Setting this parameter to zero in the Matlab program improves the stability. If we would set both the vertical eddy viscosity and the relative vertical velocity to zero, then the system would be stable according to the Matlab program. Setting all "vertical" terms to zero actually reduces the three-dimensional shallow-water equations to the two-dimensional case, of which we already know that it is unconditionally stable.

Number of σ-layers Increasing the number of σ-layers reduces the stability of the program.

From the previous remarks it should be noted that the vertical dimension is the cause of possible instabilities and unstable behaviour. The vertical eddy viscosity and the vertical profile for the horizontal velocities are the main contributors to these phenomena.

Stabilising the system for the various test cases is of course the next concern. Decreasing the time step for the test cases c02 and c05 has a positive effect on the stability, but does not seem to be sufficient, see Table 6.9. When increasing the horizontal eddy viscosity coefficient sufficiently, a stable system arises according to the analysis. Table 6.9 shows the minimal values which guarantee stability for three test cases. It appears that values for the cell-Péclet number between 15 and 25 "guarantee" numerical stability. The Courant numbers do not seem to matter in this respect.

Table 6.9: Stability tests on several test cases.

test case	b01		c02		c05		
ν_t^H	1	12	1	53	1	1	61
time step (s)	600	600	600	600	600	300	600
P	265	22	890	17	1197	1197	15
CFL_ζ	4.60	4.60	6.09	6.09	6.09	6.09	3.04
CFL_U	0.16	0.16	0.53	0.53	0.53	0.53	0.26
amplification	1.0025	1	1.0065	1	1.010	1.0024	1
stable	no	yes	no	yes	no	no	yes

It can be concluded that the Von Neumann stability analysis does not correspond one on one with results from Delft3D-FLOW. However, it does confirm for instance that test case c05 is less stable than test case c02. Also an increase in stability is shown when decreasing the time step or increasing the horizontal eddy viscosity coefficient.

6.12 Truncation Error

Rounding errors and iteration errors should be smaller than truncation errors. If this is not the case, then refinement of the grid would not produce more accurate results. Besides that, results with a comparable accuracy can be computed with a coarser grid in less (computation) time. In this respect we want to have a notion of the size of the truncation error.

When the discretisations have a truncation error of the order two, as is the case for the discretised three-dimensional shallow-water equations in Delft3D-FLOW, this error can be written as

$$\underline{x} - \underline{x}_{\Delta x} = K \Delta x^2, \tag{6.9}$$

where $\underline{x}$ is the exact solution to the (set of) differential equation(s) and K is a certain constant. Thus after grid refinement for instance by a factor 3, we have

$$\underline{x} - \underline{x}_{\frac{1}{3}\Delta x} = K(\tfrac{1}{3}\Delta x)^2. \tag{6.10}$$

Subtraction of (6.10) from (6.9) results in

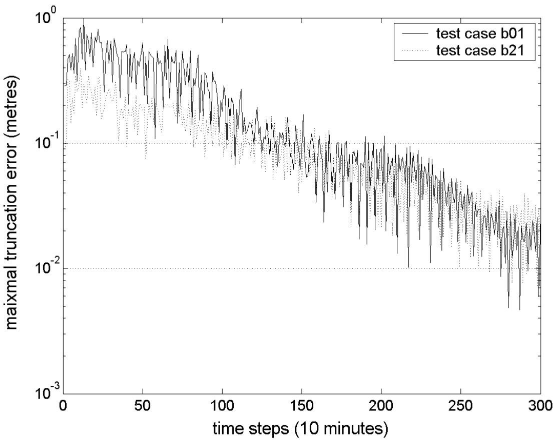
$$\underline{x}_{\frac{1}{3}\Delta x} - \underline{x}_{\Delta x} = \tfrac{8}{9} K \Delta x^2,$$

leading to

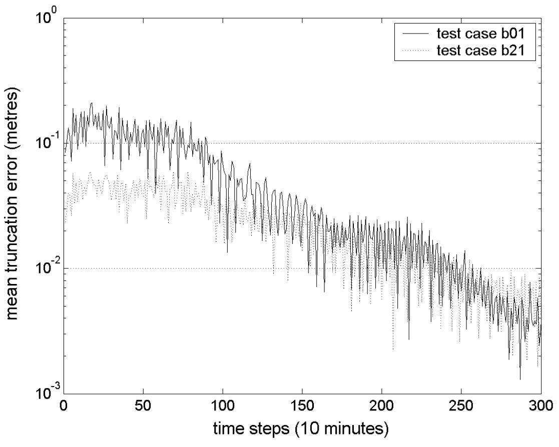
$$K = \frac{\underline{x}_{\frac{1}{3}\Delta x} - \underline{x}_{\Delta x}}{\frac{8}{9}\Delta x^2}. \tag{6.11}$$

Substitution of (6.11) into (6.9) and (6.10) gives the truncation errors.

For the test cases b01 and b21 we have calculated the truncation error using the results of test case b24 as a comparison, see Figure 6.39. It shows that a finer grid hardly contributes to a more accurate result. We also conclude that for a simulation with 10 σ-layers up to 600 grid points the truncation error is at best in $\mathcal{O}(10^{-3})$.



(a)



(b)

Figure 6.39: The maximal (a) and the mean truncation error (b) in the spatial directions for test case b01 and b21.

Chapter 7

Conclusions

In this report we have discussed the complete process from Navier-Stokes equations to the solution of the three-dimensional shallow-water equations. Residues, condition numbers and upper bounds for the relative errors of separate matrix equations have been computed as well as estimates of the absolute and the relative error of final results.

We will first summarise the conclusions from analyses of the test cases which have been performed without row scaling

Machine Precision

- For a small problem (100 grid points) the results between single and double precision match very well. However, they do diverge in time (Figure 6.1) for a closed boundary problem. After 100.000 time steps this difference may be in the order of millimetres.
- The residues and upper bounds for the relative error for double precision is about eight orders of magnitude smaller.
- The increase in computation time for double precision with respect to single precision on the Sun-Sparc is on average 13%.

Architecture

- Different computer architectures (Sun-Sparc, SGI, PC-Windows) produce very similar results in single precision for a simple test case. The accuracy for the different architectures is also very similar. In both cases the PC produces slightly better results than the Sun-Sparc and the SGI.

Problem size

- The upper bound on the relative error for a standard test case is too high to guarantee reliable results. Furthermore, the upper bound increases linearly with the problem size.
- The number of iterations for explicit momentum equation increases linearly with the number of grid points in x- and y-direction.
- The condition number of the continuity equation does not depend on the problem size.
- The absolute error in the results of Delft3D-FLOW increases slowly with the number of grid points in the horizontal plane.

Layer distribution

- The condition number and the upper bound for the relative error of the continuity equation are not dependent on the number of σ-layers.
- The condition number and subsequently the upper bound for the relative error of the explicit momentum equation increase exponentially with the number of σ-layers.
- The absolute error in the results of Delft3D-FLOW range from $\mathcal{O}(10^{-6})$ for one σ-layer to $\mathcal{O}(10^{-2})$ for fifty σ-layers.
- For two and three σ-layers Delft3D-FLOW shows erroneous results.

Bed topography

- Increasing slopes in the bed topography results in a slightly higher condition number and therefore slightly less accuracy.

Recession and flooding

- Due to the hard conditions for recession and flooding small errors and perturbations may cause large differences between single and double precision results, especially regarding the water level.

Turbulence Closure Model

- The choice of a turbulence model has consequences for the accuracy of the computed results. The $k-\epsilon$ turbulence model shows less accurate results than the $k-L$ turbulence model, the algebraic turbulence model and the constant value model. The difference is about one order of magnitude.

Stopping criteria

- The residual stopping criterion may either improve or worsen the relative residue compared to the original stopping criterion. This is due to the fact that the critical value for the residual criterion is set too high to ensure improvement of the computed result. This high critical value is not set higher, because then it is not met for test case due to the limited machine precision.
- The residual stopping criterion is expensive.
- The contraction factors are small enough for nearly all test cases to ensure convergence. However, for large problem sizes the contraction factor can increase close to one. A small adjustment, related to the contraction factor, to the original stopping criterion can still ensure convergence.

Stability

- The numerical scheme implemented in Delft3D-FLOW for the two-dimensional shallow-water equations is unconditionally stable. The numerical scheme for the three-dimensional shallow-water equations is not. Stability highly depends on the vertical profile for the horizontal flow velocities, the vertical eddy viscosity and the relative vertical velocity.

Row scaling

- Row scaling has a tremendous effect on the condition number and subsequently on the upper bound of the relative error of the continuity equation. The difference is, depending on the problem, between 2 and 5 orders of magnitude. The acquired upper bounds for the relative errors are in many cases so low that they guarantee small relative errors. The upper bounds range from $\mathcal{O}(10^{-7})$ to $\mathcal{O}(10^{-3})$ for the continuity equation and from $\mathcal{O}(10^{-7})$ to $\mathcal{O}(10^{-6})$ for the explicit momentum equation.
- The condition number of the continuity equation *does* depend on the problem size when row scaling is applied.
- The spin-up time decreases when row scaling is applied.

Chapter 8

Recommendations

With regard to the conclusions in the previous chapter, we recommend the following actions to be taken or considered.

- The acquired accuracy is not in every case acceptable. Thus measures should be taken. We strongly suggest to switch over from single to double precision.
- A preconditioning method will ensure that a sufficient accuracy is achieved. We suggest the implementation of a preconditioning method like row scaling. Row scaling hardly adds to the computation time and is very easy to implement.
- Other stopping criteria for the Gauss-Jacobi iteration are possible. In combination with the contraction factor the original stopping criterion can still be used. This is a simple adjustment and the extra computation time is negligible.
- Small rounding errors and high accuracy alone still do not guarantee good results. The truncation errors are supposedly larger than rounding errors and iteration errors. We recommend that these truncation errors are estimated and measures should be taken if they appear to be too large. To this end implementation of higher order numerical schemes, grid refinement and reduction of the time step are possibilities.
- The most common used $k-\epsilon$ turbulence model produces in general less accurate results than the other turbulence models. This 'problem' is already known and new ideas (e.g. $k-\tau$ turbulence model) are still developing.

Appendix A

Numerical Scheme

The numerical approximations will be given for both stages and if necessary due to the staggered grid averaged quantities are introduced. At the inner points of the grid each term of the equations (3.1a-d) is approximated as follows:

1. Discretisation of $\dfrac{\partial u}{\partial t}$ at $(m+\frac{1}{2}, n, k)$

   stage 1: $(u^{p+\frac{1}{2}} - u^p)/\frac{1}{2}\tau$

   stage 2: $(u^{p+1} - u^{p+\frac{1}{2}})/\frac{1}{2}\tau$

2. Discretisation of $\dfrac{\partial v}{\partial t}$ at $(m, n+\frac{1}{2}, k)$ is equivalent to the discretisation of $\dfrac{\partial u}{\partial t}$.

3. Discretisation of $\dfrac{\partial \zeta}{\partial t}$ at (m, n) is equivalent to the discretisation of $\dfrac{\partial u}{\partial t}$.

4. Horizontal advection term $u\dfrac{\partial u}{\partial x}$ at $(m+\frac{1}{2}, n, k)$

   stage 1: $u^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(u^p_{m+1\frac{1}{2},n} - u^p_{m-\frac{1}{2},n})/2\Delta x$

   stage 2: $$\begin{cases} u^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(3u^{p+1}_{m+\frac{1}{2},n} - 4u^{p+1}_{m-\frac{1}{2},n} + u^{p+1}_{m-1\frac{1}{2},n})/2\Delta x, & \text{if} \quad u^{p+\frac{1}{2}}_{m+\frac{1}{2},n} > 0 \\ u^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(-3u^{p+1}_{m+\frac{1}{2},n} + 4u^{p+1}_{m+1\frac{1}{2},n} - u^{p+1}_{m+2\frac{1}{2},n})/2\Delta x, & \text{if} \quad u^{p+\frac{1}{2}}_{m+\frac{1}{2},n} \leq 0 \end{cases}$$

   In the first stage the derivative is treated in an explicit way and approximated with a central difference scheme; in the second stage it is approximated with an implicit upwind scheme.

5. Horizontal advection term $v\dfrac{\partial v}{\partial y}$ at $(m, n+\frac{1}{2}, k)$

   stage 1: $$\begin{cases} v^p_{m,n+\frac{1}{2}}(3v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}} - 4v^{p+\frac{1}{2}}_{m,n-\frac{1}{2}} + v^{p+\frac{1}{2}}_{m,n-1\frac{1}{2}})/2\Delta y, & \text{if} \quad v^p_{m,n+\frac{1}{2}} > 0 \\ v^p_{m,n+\frac{1}{2}}(-3v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}} + 4v^{p+\frac{1}{2}}_{m,n+1\frac{1}{2}} - v^{p+\frac{1}{2}}_{m,n+2\frac{1}{2}})/2\Delta y, & \text{if} \quad v^p_{m,n+\frac{1}{2}} \leq 0 \end{cases}$$

   stage 2: $v^{p+1}_{m,n+\frac{1}{2}}(v^{p+\frac{1}{2}}_{m,n+1\frac{1}{2}} - v^{p+\frac{1}{2}}_{m,n-\frac{1}{2}})/2\Delta y$

   In the first stage the derivative is approximated with an implicit upwind scheme; in the second stage it is treated in an explicit way and approximated with a central difference scheme.

6. Horizontal advection term $v\dfrac{\partial u}{\partial y}$ at $(m+\frac{1}{2}, n, k)$

stage 1: $\bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(u^p_{m+\frac{1}{2},n+1} - u^p_{m+\frac{1}{2},n-1})/2\Delta y$

$$\text{stage 2:}\quad \begin{cases} \bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(3u^{p+1}_{m+\frac{1}{2},n} - 4u^{p+1}_{m+\frac{1}{2},n-1} + u^{p+1}_{m+\frac{1}{2},n-2})/2\Delta y, & \text{if } \bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n} > 0 \\ \bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n}(-3u^{p+1}_{m+\frac{1}{2},n} + 4u^{p+1}_{m+\frac{1}{2},n+1} - u^{p+1}_{m+\frac{1}{2},n+2})/2\Delta y, & \text{if } \bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n} \le 0 \end{cases}$$

where $\bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n} = \frac{1}{4}(v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}} + v^{p+\frac{1}{2}}_{m+1,n+\frac{1}{2}} + v^{p+\frac{1}{2}}_{m,n-\frac{1}{2}} + v^{p+\frac{1}{2}}_{m+1,n-\frac{1}{2}})$, the averaged value for $v^{p+\frac{1}{2}}$ at $(m+\frac{1}{2}, n, k)$.

7. Horizontal advection term $u\dfrac{\partial v}{\partial x}$ at $(m, n+\frac{1}{2}, k)$

$$\text{stage 1:}\quad \begin{cases} \bar{u}^p_{m,n+\frac{1}{2}}(3v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}} - 4v^{p+\frac{1}{2}}_{m-1,n+\frac{1}{2}} + v^{p+\frac{1}{2}}_{m-2,n+\frac{1}{2}})/2\Delta x, & \text{if } \bar{u}^p_{m,n+\frac{1}{2}} > 0 \\ \bar{u}^p_{m,n+\frac{1}{2}}(-3v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}} + 4v^{p+\frac{1}{2}}_{m+1,n+\frac{1}{2}} - v^{p+\frac{1}{2}}_{m+2,n+\frac{1}{2}})/2\Delta x, & \text{if } \bar{u}^p_{m,n+\frac{1}{2}} \le 0 \end{cases}$$

stage 2: $\bar{u}^{p+1}_{m,n+\frac{1}{2}}(v^{p+\frac{1}{2}}_{m+1,n+\frac{1}{2}} - v^{p+\frac{1}{2}}_{m-1,n+\frac{1}{2}})/2\Delta x$

where $\bar{u}^p_{m,n+\frac{1}{2}} = \frac{1}{4}(u^p_{m+\frac{1}{2},n} + u^p_{m+\frac{1}{2},n+1} + u^p_{m-\frac{1}{2},n} + u^p_{m-\frac{1}{2},n+1})$, the averaged value for u^p at $(m, n+\frac{1}{2}, k)$.

8. Vertical advection term $\dfrac{\omega}{H}\dfrac{\partial u}{\partial \sigma}$ at $(m+\frac{1}{2}, n, k)$

$$\text{stage 1:}\quad \frac{\bar{\omega}^p_k}{\frac{1}{2}h^p_{k-1}+h^p_k+\frac{1}{2}h^p_{k+1}}\left[\frac{h^p_k+h^p_{k+1}}{h^p_{k-1}+h^p_k}\left(u^{p+\frac{1}{2}}_{k-1} - u^{p+\frac{1}{2}}_k\right) + \frac{h^p_{k-1}+h^p_k}{h^p_k+h^p_{k+1}}\left(u^{p+\frac{1}{2}}_k - u^{p+\frac{1}{2}}_{k+1}\right)\right]$$

$$\text{stage 2:}\quad \frac{\bar{\omega}^{p+\frac{1}{2}}_k}{\frac{1}{2}h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k+\frac{1}{2}h^{p+\frac{1}{2}}_{k+1}}\left[\frac{h^{p+\frac{1}{2}}_k+h^{p+\frac{1}{2}}_{k+1}}{h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k}\left(u^{p+1}_{k-1} - u^{p+1}_k\right) + \frac{h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k}{h^{p+\frac{1}{2}}_k+h^{p+\frac{1}{2}}_{k+1}}\left(u^{p+1}_k - u^{p+1}_{k+1}\right)\right]$$

where

$$\bar{\omega}^p_k = \bar{\omega}^p_{m+\frac{1}{2},n,k} = \tfrac{1}{4}(\omega^p_{m,n,k-\frac{1}{2}} + \omega^p_{m,n,k+\frac{1}{2}} + \omega^p_{m+1,n,k-\frac{1}{2}} + \omega^p_{m+1,n,k+\frac{1}{2}}),$$

the averaged value of ω^p at $(m+\frac{1}{2}, n, k)$, and $h^p_{m+\frac{1}{2},n,k}$ is defined as $\Delta\sigma_k \bar{H}^p_{m+\frac{1}{2},n}$ with

$$\bar{H}^p_{m+\frac{1}{2},n} = \tfrac{1}{2}(\zeta^p_{m,n} + \zeta^p_{m+1,n} + d_{m+\frac{1}{2},n-\frac{1}{2}} + d_{m+\frac{1}{2},n+\frac{1}{2}}),$$

the averaged value of H^p at $(m+\frac{1}{2}, n)$.

9. Vertical advection term $\dfrac{\omega}{H}\dfrac{\partial v}{\partial \sigma}$ at $(m, n+\frac{1}{2}, k)$

$$\text{stage 1:}\quad \frac{\bar{\omega}^p_k}{\frac{1}{2}h^p_{k-1}+h^p_k+\frac{1}{2}h^p_{k+1}}\left[\frac{h^p_k+h^p_{k+1}}{h^p_{k-1}+h^p_k}\left(v^{p+\frac{1}{2}}_{k-1} - v^{p+\frac{1}{2}}_k\right) + \frac{h^p_{k-1}+h^p_k}{h^p_k+h^p_{k+1}}\left(v^{p+\frac{1}{2}}_k - v^{p+\frac{1}{2}}_{k+1}\right)\right]$$

$$\text{stage 2:}\quad \frac{\bar{\omega}^{p+\frac{1}{2}}_k}{\frac{1}{2}h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k+\frac{1}{2}h^{p+\frac{1}{2}}_{k+1}}\left[\frac{h^{p+\frac{1}{2}}_k+h^{p+\frac{1}{2}}_{k+1}}{h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k}\left(v^{p+1}_{k-1} - v^{p+1}_k\right) + \frac{h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k}{h^{p+\frac{1}{2}}_k+h^{p+\frac{1}{2}}_{k+1}}\left(v^{p+1}_k - v^{p+1}_{k+1}\right)\right]$$

10. Barotropic term $-g\dfrac{\partial \zeta}{\partial x}$ at $(m+\frac{1}{2}, n, k)$

stage 1: $-g\left(\zeta^{p+\frac{1}{2}}_{m+1,n} - \zeta^{p+\frac{1}{2}}_{m,n}\right)/\Delta x$

stage 2: the same expression as in the first stage.

Note that in the first stage the expression is implicit and it is explicit in the second stage.

11. Barotropic term $-g\dfrac{\partial \zeta}{\partial y}$ at $(m, n+\frac{1}{2}, k)$

stage 1: $-g\left(\zeta^{p}_{m,n+1} - \zeta^{p}_{m,n}\right)/\Delta y$

stage 2: $-g\left(\zeta^{p+1}_{m,n+1} - \zeta^{p+1}_{m,n}\right)/\Delta y$

Note that in the first stage the expression is explicit and it is implicit in the second stage.

12. Horizontal viscosity term $\nu_t^H\left(\dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}\right)$ at $(m+\frac{1}{2}, n, k)$

stage 1: 0

stage 2: $2\nu_t^H\left[S_{xx}(u^{p+1}_{m+\frac{1}{2},n}) + S_{yy}(u^{p+1}_{m+\frac{1}{2},n})\right]$

with

$$\begin{aligned} S_{xx}(u^{p+1}_{m+\frac{1}{2},n}) &= (u^{p+1}_{m-\frac{1}{2},n} - 2u^{p+1}_{m+\frac{1}{2},n} + u^{p+1}_{m+1\frac{1}{2},n})/\Delta x^2, \\ S_{yy}(u^{p+1}_{m+\frac{1}{2},n}) &= (u^{p+1}_{m+\frac{1}{2},n-1} - 2u^{p+1}_{m+\frac{1}{2},n} + u^{p+1}_{m+\frac{1}{2},n+1})/\Delta y^2. \end{aligned}$$

In the first stage this term is neglected, while in the second stage it is computed for the whole time step (thus twice as large).

13. Horizontal viscosity term $\nu_t^H\left(\dfrac{\partial^2 v}{\partial x^2} + \dfrac{\partial^2 v}{\partial y^2}\right)$ at $(m, n+\frac{1}{2}, k)$

stage 1: $2\nu_t^H\left[S_{xx}(v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}}) + S_{yy}(v^{p+\frac{1}{2}}_{m,n+\frac{1}{2}})\right]$

stage 2: 0

14. Vertical viscosity term $\dfrac{1}{H^2}\dfrac{\partial}{\partial \sigma}\left(\nu_t^V\dfrac{\partial u}{\partial \sigma}\right)$ at $(m+\frac{1}{2}, n, k)$

stage 1: $\dfrac{1}{h_k^p}\left[\nu^V_{k-\frac{1}{2}}\dfrac{u_{k-1}^{p+\frac{1}{2}} - u_k^{p+\frac{1}{2}}}{\frac{1}{2}(h^p_{k-1}+h^p_k)} - \nu^V_{k+\frac{1}{2}}\dfrac{u_k^{p+\frac{1}{2}} - u_{k+1}^{p+\frac{1}{2}}}{\frac{1}{2}(h^p_k+h^p_{k+1})}\right]$

stage 2: $\dfrac{1}{h_k^{p+\frac{1}{2}}}\left[\nu^V_{k-\frac{1}{2}}\dfrac{u_{k-1}^{p+1} - u_k^{p+1}}{\frac{1}{2}\left(h^{p+\frac{1}{2}}_{k-1}+h^{p+\frac{1}{2}}_k\right)} - \nu^V_{k+\frac{1}{2}}\dfrac{u_k^{p+1} - u_{k+1}^{p+1}}{\frac{1}{2}\left(h^{p+\frac{1}{2}}_k+h^{p+\frac{1}{2}}_{k+1}\right)}\right]$

The vertical eddy viscosity ν_t^V is calculated or determined through one of the turbulence closure models. This is done in an explicit way. In both stages only the flow velocity component is taken implicitly.

15. Vertical viscosity term $\dfrac{1}{H^2}\dfrac{\partial}{\partial \sigma}\left(\nu_t^V\dfrac{\partial v}{\partial \sigma}\right)$ at $(m, n+\frac{1}{2}, k)$ is discretised in a similar manner as $\dfrac{1}{H^2}\dfrac{\partial}{\partial \sigma}\left(\nu_t^V\dfrac{\partial u}{\partial \sigma}\right)$.

16. Coriolis term fv at $(m+\frac{1}{2}, n, k)$

stage 1: $f\bar{v}^{p+\frac{1}{2}}_{m+\frac{1}{2},n}$

stage 2: the same expression as in the first stage.

Note that in the first stage the expression is implicit and it is explicit in the second stage.

17. Coriolis term $-fu$ at $(m, n+\frac{1}{2}, k)$

stage 1: $-f\bar{u}^{p}_{m,n+\frac{1}{2}}$

stage 2: $-f\bar{u}^{p+1}_{m,n+\frac{1}{2}}$

Note that in the first stage the expression is explicit and it is implicit in the second stage.

18. $\dfrac{\partial HU}{\partial x}$ at (m, n)

stage 1: $\sum_k \Delta\sigma_k \left(\bar{H}^{p+\frac{1}{2}}_{m+\frac{1}{2},n,k} u^{p+\frac{1}{2}}_{m+\frac{1}{2},n,k} - \bar{H}^{p+\frac{1}{2}}_{m-\frac{1}{2},n,k} u^{p+\frac{1}{2}}_{m-\frac{1}{2},n,k} \right) / \Delta x$

stage 2: the same expression as in the first stage.

Note that in the first stage the expression is implicit and it is explicit in the second stage.

19. $\dfrac{\partial HV}{\partial y}$ at (m, n)

stage 1: $\sum_k \Delta\sigma_k \left(\bar{H}^{p}_{m,n+\frac{1}{2},k} v^{p}_{m,n+\frac{1}{2},k} - \bar{H}^{p}_{m,n-\frac{1}{2},k} v^{p}_{m,n-\frac{1}{2},k} \right) / \Delta y$

stage 2: $\sum_k \Delta\sigma_k \left(\bar{H}^{p+1}_{m,n+\frac{1}{2},k} v^{p+1}_{m,n+\frac{1}{2},k} - \bar{H}^{p+1}_{m,n-\frac{1}{2},k} v^{p+1}_{m,n-\frac{1}{2},k} \right) / \Delta y$

20. $\dfrac{\partial Hu}{\partial x}$ at (m, n, k)

stage 1: $\left(\bar{H}^{p+\frac{1}{2}}_{m+\frac{1}{2},n,k} u^{p+\frac{1}{2}}_{m+\frac{1}{2},n,k} - \bar{H}^{p+\frac{1}{2}}_{m-\frac{1}{2},n,k} u^{p+\frac{1}{2}}_{m-\frac{1}{2},n,k} \right) / \Delta x$

stage 2: the same expression as in the first stage.

Note that in the first stage the expression is implicit and it is explicit in the second stage.

21. $\dfrac{\partial Hv}{\partial y}$ at (m, n, k)

stage 1: $\left(\bar{H}^{p}_{m,n+\frac{1}{2},k} v^{p}_{m,n+\frac{1}{2},k} - \bar{H}^{p}_{m,n-\frac{1}{2},k} v^{p}_{m,n-\frac{1}{2},k} \right) / \Delta y$

stage 2: $\left(\bar{H}^{p+1}_{m,n+\frac{1}{2},k} v^{p+1}_{m,n+\frac{1}{2},k} - \bar{H}^{p+1}_{m,n-\frac{1}{2},k} v^{p+1}_{m,n-\frac{1}{2},k} \right) / \Delta y$

22. $\dfrac{\partial \omega}{\partial \sigma}$ at (m, n, k)

stage 1: $\left(\omega^{p+\frac{1}{2}}_{k-\frac{1}{2}} - \omega^{p+\frac{1}{2}}_{k+\frac{1}{2}} \right) / \Delta\sigma_k$

stage 2: $\left(\omega^{p+1}_{k-\frac{1}{2}} - \omega^{p+1}_{k+\frac{1}{2}} \right) / \Delta\sigma_k$

Appendix B

Matrix of the Explicit Momentum Equation

The construction of the matrix of the explicit momentum equation (3.8b) is discussed in Section 3.5. Generally, this matrix equation is of the form $A\underline{v} = b$, where $A \in \mathbb{R}^{MNK \times MNK}$ is given on the next page and $\underline{v} \in \mathbb{R}^{MNK}$ is the solution vector for the unknown velocity component v. The vector v is arranged in respectively the vertical and the horizontal (first in y- then in x-direction) directions.

The submatrix $T_i \in \mathbb{R}^{K \times K}$ is a tridiagonal matrix with the coefficients (if $i = (m-1)N + n$) for $v_{m,n,k}$ $\forall k$. The submatrix $D^{(jy)}_{i,i+j} \in \mathbb{R}^{K \times K}$ is a diagonal matrix with the coefficients for $v_{m,n+j,k}$ $\forall k$ with $i = (m-1)N + n$ and $j \in \{-2, -1, 1, 2\}$. The submatrix $D^{(jx)}_{i,i+j\cdot N} \in \mathbb{R}^{K \times K}$ is a diagonal matrix with the coefficients for $v_{m+j,n,k}$ $\forall k$ with $i = (m-1)N+n$ and $j \in \{-2, -1, 1, 2\}$. Note that $D^{(-2x)}$, $D^{(-2y)}$, $D^{(2y)}$ and $D^{(2x)}$ can have zeros on the diagonal due to the second order upwind schemes.

$$\begin{pmatrix}
T_1 & D_{12}^{(1y)} & D_{13}^{(2y)} & \emptyset & \dots & \emptyset & D_{1,N+1}^{(1x)} & \emptyset & \dots & \emptyset & D_{1,2N+1}^{(2x)} & \emptyset & & \\
D_{21}^{(-1y)} & T_2 & D_{23}^{(1y)} & D_{24}^{(2y)} & \emptyset & \dots & \emptyset & D_{2,N+2}^{(1x)} & \emptyset & \dots & \emptyset & D_{2,2N+2}^{(2x)} & \emptyset & \\
D_{31}^{(-2y)} & D_{32}^{(-1y)} & T_3 & D_{34}^{(1y)} & D_{35}^{(2y)} & \emptyset & \dots & \emptyset & D_{3,N+3}^{(1x)} & \emptyset & \dots & \emptyset & D_{3,2N+3}^{(2x)} & \ddots \\
\emptyset & D_{42}^{(-2y)} & D_{43}^{(-1y)} & T_4 & D_{45}^{(1y)} & D_{46}^{(2y)} & \emptyset & \dots & \emptyset & D_{4,N+4}^{(1x)} & \emptyset & \dots & \emptyset & \ddots \\
\vdots & \emptyset & D_{53}^{(-2y)} & D_{54}^{(-1y)} & T_5 & D_{56}^{(1y)} & D_{57}^{(2y)} & \emptyset & \dots & \emptyset & D_{5,N+5}^{(1x)} & \emptyset & \dots & \ddots \\
\emptyset & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \\
D_{N+1,1}^{(-1x)} & \emptyset & \dots & \emptyset & D_{N+1,N-1}^{(-2y)} & D_{N+1,N}^{(-1y)} & T_{N+1} & D_{N+1,N+2}^{(1y)} & D_{N+1,N+3}^{(2y)} & \emptyset & \dots & \emptyset & D_{N+1,2N+1}^{1x} & \emptyset \\
\emptyset & D_{N+2,2}^{(-1x)} & \emptyset & \dots & \emptyset & D_{N+2,N}^{(-2y)} & D_{N+2,N+1}^{(-1y)} & T_{N+2} & D_{N+2,N+3}^{(1y)} & D_{N+2,N+4}^{(2y)} & \emptyset & \dots & \emptyset & \ddots \\
\vdots & \emptyset & D_{N+3,3}^{(-1x)} & \emptyset & \dots & \emptyset & D_{N+3,N+1}^{(-2y)} & D_{N+3,N+2}^{(-1y)} & T_{N+3} & D_{N+3,N+4}^{(1y)} & D_{N+3,N+5}^{(2y)} & \emptyset & \dots & \ddots \\
\emptyset & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \ddots & \\
D_{2N+1,1}^{(-2x)} & \emptyset & \dots & \emptyset & D_{2N+1,N+1}^{(-1x)} & \emptyset & \dots & \emptyset & D_{2N+1,2N-1}^{(-2y)} & D_{2N+1,2N}^{(-1y)} & T_{2N+1} & D_{2N+1,2N+2}^{(1y)} & D_{2N+1,2N+3}^{(2y)} & \emptyset \\
\emptyset & D_{2N+2,2}^{(-2x)} & \emptyset & \dots & \emptyset & D_{2N+2,N+2}^{(-1x)} & \emptyset & \dots & \emptyset & D_{2N+2,2N}^{(-2y)} & D_{2N+2,2N+1}^{(-1y)} & T_{2N+2} & D_{2N+2,2N+3}^{(1y)} & \ddots \\
 & \emptyset & D_{2N+3,3}^{(-2x)} & \emptyset & \dots & \emptyset & D_{2N+3,N+3}^{(-1x)} & \emptyset & \dots & \emptyset & D_{2N+3,2N+1}^{(-2y)} & D_{2N+3,2N+2}^{(-1y)} & T_{2N+3} & \ddots \\
 & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & & \ddots & \ddots & \ddots & \ddots
\end{pmatrix}$$

Appendix C

Matrices of the Linearised Three-Dimensional Shallow-Water Equations

In Section 5.3 a linearised version of the three-dimensional shallow-water equations is given in the system (5.7)-(5.8). When looking at solutions for these equations of the form

$$\begin{bmatrix} \tilde{u}^p_{mn,1} \\ \tilde{u}^p_{mn,2} \\ \vdots \\ \tilde{u}^p_{mn,K} \\ \tilde{v}^p_{mn,1} \\ \tilde{v}^p_{mn,2} \\ \vdots \\ \tilde{v}^p_{mn,K} \\ \tilde{\zeta}^p_{mn} \end{bmatrix} = \begin{bmatrix} \hat{u}^p_1 \\ \hat{u}^p_2 \\ \vdots \\ \hat{u}^p_K \\ \hat{v}^p_1 \\ \hat{v}^p_2 \\ \vdots \\ \hat{v}^p_K \\ \hat{\zeta}^p \end{bmatrix} e^{i(\alpha_1 m\Delta x+\alpha_2 n\Delta y)} = \underline{\hat{w}}^p \, e^{i(\alpha_1 m\Delta x+\alpha_2 n\Delta y)}$$

with $\alpha_1, \alpha_2 \in \mathbb{R}$, then for $K = 3$ (i.e. three σ-layers) the matrix form reads

$$\begin{aligned} A\underline{\hat{w}}^{p+\frac{1}{2}} &= B\underline{\hat{w}}^p, \\ C\underline{\hat{w}}^{p+1} &= D\underline{\hat{w}}^{p+\frac{1}{2}}. \end{aligned}$$

where the vector $\underline{\hat{w}}^p = (\hat{u}^p_1,\, \hat{u}^p_2,\, \hat{u}^p_3,\, \hat{v}^p_1,\, \hat{v}^p_2,\, \hat{v}^p_3,\, \hat{\zeta}^p)^T$. These equations represent the system (5.7)-(5.8), respectively. The matrices A and C contain the coefficients of all the implicit terms for respectively the first and second stage. The matrices B and D contain those of all the explicit terms.

Consider the derivatives in vertical direction, $S_\sigma(u^p_k)$ and $S_{\sigma\sigma}(u^p_k)$, in (5.9). They are included in the matrix form as

$$S_\sigma(\tilde{u}^p_k) = \left(\hat{D}^l_\sigma(k)\cdot \hat{u}^p_{k-1} + \hat{D}^m_\sigma(k)\cdot \hat{u}^p_k + \hat{D}^u_\sigma(k)\cdot \hat{u}^p_{k+1}\right) e^{i(\alpha_1 m\Delta x+\alpha_2 n\Delta y)}$$

and

$$S_{\sigma\sigma}(\tilde{u}^p_k) = \left(\hat{D}^l_{\sigma\sigma}(k)\cdot \hat{u}^p_{k-1} + \hat{D}^m_{\sigma\sigma}(k)\cdot \hat{u}^p_k + \hat{D}^u_{\sigma\sigma}(k)\cdot \hat{u}^p_{k+1}\right) e^{i(\alpha_1 m\Delta x+\alpha_2 n\Delta y)},$$

turning up in *three* (two at bottom and surface boundaries) matrix entries. At the end of this appendix the symbols $\hat{D}^l_\sigma(k)$, $\hat{D}^l_\sigma(k)$, etc. are explained. Derivatives in the horizontal direction however are still contained in *one* matrix entry of, e.g.

$$S_{+x}(\tilde{u}_k^p) = \hat{D}_{+x} \cdot \hat{u}_k^p \, e^{i(\alpha_1 m\Delta x + \alpha_2 n \Delta y)}.$$

The matrices $A, B, C, D \in \mathbb{R}^{7\times 7}$ are multiplied by a factor $\frac{\tau}{2}$ and as such given by

$$A = \begin{pmatrix} A_{11} & \emptyset & \underline{a}_{13} \\ \emptyset & A_{22} & \underline{0} \\ \underline{a}_{31}^T & \underline{0}^T & a_{33} \end{pmatrix} \tag{C.1}$$

with

$$A_{11} = \begin{pmatrix} 1+\frac{\tau}{2}\frac{\omega_1}{H}\hat{D}^m_\sigma(1) - \frac{\tau}{2}\frac{\nu_t^V(1)}{H^2}\hat{D}^m_{\sigma\sigma}(1) & \frac{\tau}{2}\frac{\omega_1}{H}\hat{D}^u_\sigma(1) - \frac{\tau}{2}\frac{\nu_t^V(1)}{H^2}\hat{D}^u_{\sigma\sigma}(1) & 0 \\ \frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^l_\sigma(2) - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^l_{\sigma\sigma}(2) & 1+\frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^m_\sigma(2) - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^m_{\sigma\sigma}(2) & \frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^u_\sigma(2) - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^u_{\sigma\sigma}(2) \\ 0 & \frac{\tau}{2}\frac{\omega_3}{H}\hat{D}^l_\sigma(3) - \frac{\tau}{2}\frac{\nu_t^V(3)}{H^2}\hat{D}^l_{\sigma\sigma}(3) & 1+\frac{\tau}{2}\frac{\omega_3}{H}\hat{D}^m_\sigma(3) - \frac{\tau}{2}\frac{\nu_t^V(3)}{H^2}\hat{D}^m_{\sigma\sigma}(3) \end{pmatrix},$$

A_{22} given columnwise, so

$$(A_{22})_1 = \begin{pmatrix} 1 + \frac{\tau}{2}U\hat{D}_{+x} + \frac{\tau}{2}V\hat{D}_{+y} + \frac{\tau}{2}\frac{\omega_1}{H}\hat{D}^m_\sigma(1) - \tau\nu_t^H[\hat{D}_{xx} + \hat{D}_{yy}] - \frac{\tau}{2}\frac{\nu_t^V(1)}{H^2}\hat{D}^m_{\sigma\sigma}(1) \\ \frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^l_\sigma(2) - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^l_{\sigma\sigma}(2) \\ 0 \end{pmatrix}$$

$$(A_{22})_2 = \begin{pmatrix} \frac{\tau}{2}\frac{\omega_1}{H}\hat{D}^u_\sigma(1) - \frac{\tau}{2}\frac{\nu_t^V(1)}{H^2}\hat{D}^u_{\sigma\sigma}(1) \\ 1 + \frac{\tau}{2}U\hat{D}_{+x} + \frac{\tau}{2}V\hat{D}_{+y} + \frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^m_\sigma(2) - \tau\nu_t^H[\hat{D}_{xx} + \hat{D}_{yy}] - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^m_{\sigma\sigma}(2) \\ \frac{\tau}{2}\frac{\omega_3}{H}\hat{D}^l_\sigma(3) - \frac{\tau}{2}\frac{\nu_t^V(3)}{H^2}\hat{D}^l_{\sigma\sigma}(3) \end{pmatrix}$$

$$(A_{22})_3 = \begin{pmatrix} 0 \\ \frac{\tau}{2}\frac{\omega_2}{H}\hat{D}^u_\sigma(2) - \frac{\tau}{2}\frac{\nu_t^V(2)}{H^2}\hat{D}^u_{\sigma\sigma}(2) \\ 1 + \frac{\tau}{2}U\hat{D}_{+x} + \frac{\tau}{2}V\hat{D}_{+y} + \frac{\tau}{2}\frac{\omega_3}{H}\hat{D}^m_\sigma(3) - \tau\nu_t^H[\hat{D}_{xx} + \hat{D}_{yy}] - \frac{\tau}{2}\frac{\nu_t^V(3)}{H^2}\hat{D}^m_{\sigma\sigma}(3) \end{pmatrix}$$

$$\underline{a}_{13} = \tfrac{\tau}{2}g\hat{D}_{0x} \cdot (1,1,1)^T,$$

$$\underline{a}_{31} = (\tfrac{\tau}{2}\Delta\sigma_1 H\hat{D}_{0x},\ \tfrac{\tau}{2}\Delta\sigma_2 H\hat{D}_{0x},\ \tfrac{\tau}{2}\Delta\sigma_3 H\hat{D}_{0x})^T,$$

$$a_{33} = 1 + \tfrac{\tau}{2}U\hat{D}_{1x},$$

$$B = \begin{pmatrix} B_{11} & \emptyset & \underline{0}^T \\ \emptyset & I_3 & \underline{b}_{23} \\ \underline{0} & \underline{b}_{32}^T & b_{33} \end{pmatrix} \tag{C.2}$$

with I_3 the 3×3 identity matrix,

$$B_{11} = \begin{pmatrix} 1-\frac{\tau}{2}U\hat{D}_{1x} - \frac{\tau}{2}V\hat{D}_{1y} & 0 & 0 \\ 0 & 1-\frac{\tau}{2}U\hat{D}_{1x} - \frac{\tau}{2}V\hat{D}_{1y} & 0 \\ 0 & 0 & 1-\frac{\tau}{2}U\hat{D}_{1x} - \frac{\tau}{2}V\hat{D}_{1y} \end{pmatrix},$$

$$\underline{b}_{23} = -\tfrac{\tau}{2} g \hat{D}_{0y} \cdot (1,1,1)^T,$$

$$\underline{b}_{32} = (-\tfrac{\tau}{2}\Delta\sigma_1 H \hat{D}_{0y},\ -\tfrac{\tau}{2}\Delta\sigma_2 H \hat{D}_{0y},\ -\tfrac{\tau}{2}\Delta\sigma_3 H \hat{D}_{0y})^T,$$

$$b_{33} = 1 - \tfrac{\tau}{2} V \hat{D}_{1y}.$$

The matrix C which is given below is comparable with/to the matrix A. The ADI scheme is responsible for the fact that in C the submatrices A_{11} and A_{22} are interchanged, because the discretisations which are used on $\hat{u}_k\ \forall k$ in one half time step are exactly the same as the ones used on $\hat{v}_k\ \forall k$ in the other half time step. So, we have

$$C = \begin{pmatrix} A_{22} & \emptyset & \underline{0} \\ \emptyset & A_{11} & \underline{c}_{23} \\ \underline{0} & \underline{c}_{32}^T & c_{33} \end{pmatrix} \tag{C.3}$$

with

$$\underline{c}_{23} = \tfrac{\tau}{2} g \hat{D}_{0y} \cdot (1,1,1)^T,$$

$$\underline{c}_{32} = (\tfrac{\tau}{2}\Delta\sigma_1 H \hat{D}_{0y},\ \tfrac{\tau}{2}\Delta\sigma_2 H \hat{D}_{0y},\ \tfrac{\tau}{2}\Delta\sigma_3 H \hat{D}_{0y})^T,$$

$$c_{33} = 1 + \tfrac{\tau}{2} V \hat{D}_{1y},$$

and

$$D = \begin{pmatrix} I_3 & \emptyset & \underline{d}_{13} \\ \emptyset & B_{11} & \underline{0} \\ \underline{d}_{31}^T & \underline{0} & d_{33} \end{pmatrix} \tag{C.4}$$

with

$$\underline{d}_{13} = -\tfrac{\tau}{2} g \hat{D}_{0x} \cdot (1,1,1)^T,$$

$$\underline{d}_{31} = (-\tfrac{\tau}{2}\Delta\sigma_1 H \hat{D}_{0x},\ -\tfrac{\tau}{2}\Delta\sigma_2 H \hat{D}_{0x},\ -\tfrac{\tau}{2}\Delta\sigma_3 H \hat{D}_{0x})^T,$$

$$d_{33} = 1 - \tfrac{\tau}{2} U \hat{D}_{1x}.$$

The symbols $\hat{D}_{xx}$ and $\hat{D}_{yy}$in the horizontal viscosity terms $-\tau\nu_t^H \hat{D}_{xx}$ and $-\tau\nu_t^H \hat{D}_{yy}$ represent second order accurate second derivatives in respectively the x- and y-direction. Thus

$$\hat{D}_{xx} = 2[\cos(\alpha_1 \Delta x) - 1]/\Delta x^2 \quad \text{and} \quad \hat{D}_{yy} = 2[\cos(\alpha_2 \Delta y) - 1]/\Delta y^2.$$

The vertical derivatives are represented by $\hat{D}_\sigma^l$, $\hat{D}_\sigma^m$ and $\hat{D}_\sigma^u$ for the vertical advective terms and $\hat{D}_{\sigma\sigma}^l$, $\hat{D}_{\sigma\sigma}^m$ and $\hat{D}_{\sigma\sigma}^u$ for the vertical viscosity terms. For the top layer ($k = 1$) the advective terms are represented by

$$\hat{D}_\sigma^m(1) = \frac{1}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)} \quad \text{and} \quad \hat{D}_\sigma^u(1) = -\frac{1}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)},$$

for the bottom layer ($k = K$) by

$$\hat{D}_\sigma^l(K) = \frac{1}{\frac{1}{2}(\Delta\sigma_{K-1} + \Delta\sigma_K)} \quad \text{and} \quad \hat{D}_\sigma^m(K) = -\frac{1}{\frac{1}{2}(\Delta\sigma_{K-1} + \Delta\sigma_K)},$$

and for the other layers by

$$
\begin{aligned}
\hat{D}^l_\sigma(k) &= \frac{2}{\Delta\sigma_{k-1} + 2\Delta\sigma_k + \Delta\sigma_{k+1}} \cdot \frac{\Delta\sigma_k + \Delta\sigma_{k+1}}{\Delta\sigma_{k-1} + \Delta\sigma_k}, && 0 < k < K, \\
\hat{D}^m_\sigma(k) &= \frac{2}{\Delta\sigma_{k-1} + 2\Delta\sigma_k + \Delta\sigma_{k+1}} \left[-\frac{\Delta\sigma_k + \Delta\sigma_{k+1}}{\Delta\sigma_{k-1} + \Delta\sigma_k} + \frac{\Delta\sigma_{k-1} + \Delta\sigma_k}{\Delta\sigma_k + \Delta\sigma_{k+1}} \right], && 0 < k < K, \\
\hat{D}^u_\sigma(k) &= -\frac{2}{\Delta\sigma_{k-1} + 2\Delta\sigma_k + \Delta\sigma_{k+1}} \cdot \frac{\Delta\sigma_{k-1} + \Delta\sigma_k}{\Delta\sigma_k + \Delta\sigma_{k+1}}, && 0 < k < K.
\end{aligned}
$$

For the top layer ($k = 1$) the vertical viscosity terms are represented by

$$
\hat{D}^m_{\sigma\sigma}(1) = -\frac{1}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)\Delta\sigma_1} \quad \text{and} \quad \hat{D}^u_{\sigma\sigma}(1) = \frac{1}{\frac{1}{2}(\Delta\sigma_1 + \Delta\sigma_2)\Delta\sigma_1},
$$

for the bottom layer ($k = K$) by

$$
\hat{D}^l_{\sigma\sigma}(K) = \frac{1}{\frac{1}{2}(\Delta\sigma_{K-1} + \Delta\sigma_K)\Delta\sigma_K} \quad \text{and} \quad \hat{D}^m_{\sigma\sigma}(K) = -\frac{1}{\frac{1}{2}(\Delta\sigma_{K-1} + \Delta\sigma_K)\Delta\sigma_K},
$$

and for the other layers by

$$
\begin{aligned}
\hat{D}^l_{\sigma\sigma}(k) &= \frac{1}{\frac{1}{2}(\Delta\sigma_{k-1} + \Delta\sigma_k)\Delta\sigma_k}, && 0 < k < K, \\
\hat{D}^m_{\sigma\sigma}(k) &= -\left(\frac{1}{\frac{1}{2}(\Delta\sigma_{k-1} + \Delta\sigma_k)\Delta\sigma_k} + \frac{1}{\frac{1}{2}(\Delta\sigma_k + \Delta\sigma_{k+1})\Delta\sigma_k} \right), && 0 < k < K, \\
\hat{D}^u_{\sigma\sigma}(k) &= \frac{1}{\frac{1}{2}(\Delta\sigma_k + \Delta\sigma_{k+1})\Delta\sigma_k}, && 0 < k < K.
\end{aligned}
$$

Appendix D

Listing for Stability Program

The stability of the three-dimensional shallow-water equations for various cases has been tested by means of Matlab. Below follows the listing of the Matlab program. For clarity the comments in this program are included.

```
function [lev] = stabilitySWE(deltasigma,H,U,V,omega,nutH,nutV,...
delta_x,delta_y,tau,resolution)
% Stability.m calculates the largest eigenvalue in absolute sense
% of the amplification matrix for the discrete three-dimensional
% shallow-water equations. This is performed using a Von Neumann
% stability analysis (or Fourier analysis).
% A stability field is plotted on the plane where alpha1 varies
% between -pi/delta_x and pi/delta_x. For alpha2 similar bounds
% hold.
%
% SYNTAX: STABILITYSWE(deltasigma, H, U, V, omega, nutH, nutV,...
%                       nutV, delta_x, delta_y, tau, resolution)
%
% deltasigma is a row containing the vertical layer distribution.
% H is the total water depth in metres.
% U is s row containing the advective horizontal velocities in
%   x-direction in metres/second for every sigma-layer.
% V is s row containing the advective horizontal velocities in
%   y-direction in metres/second for every sigma-layer.
% omega is vertical velocity in the top layer. omega is zero at the
%   bottom. The interpolation is linear.
% nutH is the constant horizontal eddy viscosity in
%   metres^2/second.
% nutV is the vertical eddy viscosity in the middle of the depth
%   in metres^2/second. nutV is zero at the top and at the bottom.
%   The interpolation is done with a sine function.
% tau is the time step in minutes.
% delta_x is the grid size in x-direction in metres.
% delta_y is the grid size in y-direction in metres.
% resolution is an integer determining how many points in x- as
%   well as y-direction of the 'stability field' are going to be
%   calculated. Higer resolution means longer calculation time, but
%   better restults. Range: 10 (quick and dirty) to 30 (accurate).
%
% OUPUT: [lev] = STABILITYSWE(...);
%
% lev is the largest eigenvalue of the amplification matrix.
```

```
%
% RESTRICTIONS
%
% This function can only be used with rectilinear grids and con-
%   stant horizontal eddy viscosity.
% Salt, pollutants, temperature, flooding and drying and structures
%   are not considered.
%
% REMARKS
%
% Boundary conditions are not considered in a Fourier analysis as
%   such, because right-hand sides are not necessary for a Fourier
%   analysis.
% The values in deltasigma should add up to one.
% deltasigma, U and V should be of the same sizes.
%
% EXAMPLE
%
% deltasigma = 0.1*ones(1,10);
% H = 10;
% U_top = 1.2; U_bottom = 0.5;
% U = (1-log(0.1+0.9*(1-cumsum(deltasigma)))/log(0.1))*(U_top-U_bottom)...
%                                                               + U_bottom;
% V = 0.2*ones(size(deltasigma));
% omega = 1e-4; nutH = 1; nutV = 2e-2;
% delta_x = 1000; delta_y = 1000; tau = 5;
% [lev] = STABILITYSWE(deltasigma,H,U,V,omega,nutH,nutV,...
%                                                  delta_x,delta_y,tau,20);


% CHECK INPUT

if (abs(sum(deltasigma)-1) > 1e-8)
error('ERROR: Relative sigma-layers do not add up to one!'); end;
kmax = size(deltasigma,2);
if (size(U,2) ~= kmax)
  error('ERROR: Number of velocities in U-profile do not match the number...
                                                            of sigma-layers!'); end;
if (size(V,2) ~= kmax)
  error('ERROR: Number of velocities in V-profile do not match the number
                                                            of sigma-layers!'); end;
if (H   <= 0)
  error('ERROR: Water depth is negative or zero!'); end;
if (nutH < 0)
  error('ERROR: Horizontal eddy viscosity is negative!'); end;
if (nutV < 0)
  error('ERROR: Vertical eddy viscosity is negative!'); end;
if (delta_x <= 0)
  error('ERROR: Grid size in x-direction is negative or zero!'); end;
if (delta_y <= 0)
  error('ERROR: Grid size in y-direction is negative or zero!'); end;
if (tau <= 0)
  error('ERROR: Time step is negative or zero!'); end;
resolution = round(resolution);
if (resolution < 3)
```

```
  error('ERROR: Resolution is too small! Must be at least 4.'); end;


% INITIALISATION

% --- construct arrays for the vertical eddy viscosity (nutV) and
%     the vertical velocity (omega) if more than one layer is
%     defined.
if (kmax > 1)
  nutV  = nutV*sin(cumsum(deltasigma)*pi);
  omega = omega*(1 - cumsum(deltasigma));    % linear increase of omega
  %omega = omega*sin(cumsum(deltasigma)*pi);  % sine curve for omega
else
  nutV  = 0;
  omega = 0;
end;

% --- calculate depth-integrated advective currents
U_di = dot(U,deltasigma);
V_di = dot(V,deltasigma);

% --- set time step unit to seconds
tau = tau*60;
% --- define gravitation constant (in m/s^2)
g   = 9.81;
% --- calculate CFL number
CFL = sqrt(g*H)*tau/min(delta_x, delta_y);

%
% PRINT DATA TO SCREEN
%
fprintf('Time step:         tau = %5.1f s\n',tau);
fprintf('Grid size:     Delta_x = %5d m\n',delta_x);
fprintf('               Delta_y = %5d m\n',delta_y);
fprintf('Water depth:         H = %5.2f m\n',H);
fprintf('Advective currents: U = %5.2f m/s\n',U_di);
fprintf('                     V = %5.2f m/s\n',V_di);
fprintf('Number of layers:        %5d \n',kmax);
fprintf('Vertical velocity, eddy viscosity, layer distribution, velocity profiles\n');
fprintf('        omega           nutV        layers        U           V   \n');
fprintf('   %12.3e    %12.3e     %6.4f     %6.3f      %6.3f \n',...
                                             [omega; nutV; deltasigma; U; V]);
fprintf('Viscosity:        nutH = %5.2f m^2/s\n\n',nutH);
fprintf('                   CFL = %5.2f \n',CFL);
fprintf('            Resolution = %5d \n\n',resolution);

%
% MATRIX CONSTRUCTION AND CALCULATION OF LARGEST EIGENVALUE OF G
%
% --- define index ranges
indexrange = [0:resolution];
range1 = 2*pi/(delta_x)*(indexrange/resolution-0.5);
range2 = 2*pi/(delta_y)*(indexrange/resolution-0.5);
% --- initialise matrices
p = 2*kmax+1;
```

```
A = zeros(p);
B = A; C = A; D = A;

for index1=indexrange
alpha1 = range1(index1+1);
for index2=indexrange
alpha2 = range2(index2+1);

  % --- central difference schemes first derivative
  D_0x = 2*i*sin(0.5*alpha1*delta_x)/delta_x;
  D_0y = 2*i*sin(0.5*alpha2*delta_y)/delta_y;
  D_1x = i*sin(alpha1*delta_x)/delta_x;
  D_1y = i*sin(alpha2*delta_y)/delta_y;
  % --- central difference schemes second derivative
  D_xx = 2*(cos(alpha1*delta_x)-1)/(delta_x^2);
  D_yy = 2*(cos(alpha2*delta_y)-1)/(delta_y^2);
  % --- second order upwind schemes
  for k=1:kmax
    D_ux(k) = (sign(U(k))*(1-cos(alpha1*delta_x))^2 +...
      i*sin(alpha1*delta_x)*(2-cos(alpha1*delta_x)))/delta_x;
    D_uy(k) = (sign(V(k))*(1-cos(alpha2*delta_y))^2 +...
      i*sin(alpha2*delta_y)*(2-cos(alpha2*delta_y)))/delta_y;
  end;

  % --- vertical advective and viscosity terms
  if (kmax == 1)
    D_lds(1) = 0;
    D_mds(1) = 0;
    D_uds(1) = 0;
  else
    D_mds(1) =  2*(omega(1)/H) / (deltasigma(1)+deltasigma(2))...
              + 2*nutV(1)/(H^2*(deltasigma(1)+deltasigma(2))*deltasigma(1));
    D_uds(1) = -D_mds(1);
    for k=2:kmax-1
      hulp1 = 2*(omega(k)/H)/(deltasigma(k-1)+2*deltasigma(k)+deltasigma(k+1));
      sigmalaag = deltasigma(k-1)+deltasigma(k);
      sigmahoog = deltasigma(k)+deltasigma(k+1);
      D_lds(k) =  hulp1*sigmahoog/sigmalaag -...
        2*nutV(k)/(H^2*sigmalaag*deltasigma(k));
      D_uds(k) = -hulp1*sigmalaag/sigmahoog -...
        2*nutV(k)/(H^2*sigmahoog*deltasigma(k));
      D_mds(k) = -D_lds(k) - D_uds(k);
    end;
    D_lds(kmax) =  2*(omega(kmax)/H) / (deltasigma(kmax-1)+deltasigma(kmax))...
       - 2*nutV(kmax) / (H^2*(deltasigma(kmax-1)+deltasigma(kmax))*deltasigma(kmax));
    D_mds(kmax) = -D_lds(kmax);
  end;

  % --- compute matrix elements
  for k=1:kmax
    A(k,k) = 2/tau + D_mds(k);
    if (k>1)    A(k,k-1) = D_lds(k); end;
    if (k<kmax) A(k,k+1) = D_uds(k); end;
    A(k,p) = g*D_0x;
    A(p,k) = deltasigma(k)*H*D_0x;
    B(k,k) = 2/tau - U(k)*D_1x - V(k)*D_1y;

    C(k,k) = 2/tau + U(k)*D_ux(k) + V(k)*D_uy(k) + D_mds(k) - 2*nutH*(D_xx+D_yy);
    if (k>1)    C(k,k-1) = D_lds(k); end;
    if (k<kmax) C(k,k+1) = D_uds(k); end;

    D(k,k) = 2/tau;
    D(k,p) = -g*D_0x;
    D(p,k) = -deltasigma(k)*H*D_0x;
  end;
  for k=kmax+1:2*kmax
    A(k,k) = 2/tau + U(k-kmax)*D_ux(k-kmax) +...
      V(k-kmax)*D_uy(k-kmax) + D_mds(k-kmax) - 2*nutH*(D_xx+D_yy);
    if (k>kmax+1) A(k,k-1) = D_lds(k-kmax); end;
    if (k<2*kmax) A(k,k+1) = D_uds(k-kmax); end;

    B(k,k) = 2/tau;
    B(k,p) = -g*D_0y;
    B(p,k) = -deltasigma(k-kmax)*H*D_0y;

    C(k,k) = 2/tau + D_mds(k-kmax);
    if (k>kmax+1) C(k,k-1) = D_lds(k-kmax); end;
    if (k<2*kmax) C(k,k+1) = D_uds(k-kmax); end;
    C(k,p) = g*D_0y;
    C(p,k) = deltasigma(k-kmax)*H*D_0y;

    D(k,k) = 2/tau - U(k-kmax)*D_1x - V(k-kmax)*D_1y;
  end;
  A(p,p) = 2/tau + U_di*D_1x;
  B(p,p) = 2/tau - V_di*D_1y;
  C(p,p) = 2/tau + V_di*D_1y;
  D(p,p) = 2/tau - U_di*D_1x;

  A = tau/2*A;
  B = tau/2*B;
  C = tau/2*C;
  D = tau/2*D;

  G = (inv(C)*D)*(inv(A)*B);

  largestev(index1+1, index2+1) = max(abs(eig(G)));

end;
end;

% --- calculate largest eigenvalue
lev = max(max(largestev));

% --- print largest eigenvalue to the screen
fprintf('Largest eigenvalue : %.8f\n',lev);

% --- plotting the graoh of the stability field.
figure(gcf+1)
[c,h] = contour(range1, range2, largestev, [.7 .8  .9 .94 .96 .98 .99 .995 1 1.02]);
```

```
colormap([0 0 0]);
set(gca, 'fontsize',16,'fontname','Times New Roman');
title(strcat('Stability Field. LEV=',num2str(lev,9)));
clabel(c,h,'fontsize',14,'labelspacing',300);
xlabel('\alpha_1','fontsize',16,'fontname','Times New Roman');
ylabel('\alpha_2','fontsize',16,'fontname','Times New Roman');
set(gcf, 'position', [1 29 1152 768]);
```

Appendix E

Settings in Delft3D-FLOW

Delft3D-FLOW is used to simulate (three-dimensional) shallow-water flows. For a single simulation many parameters can be set. We will distinguish them in three groups: physical, numerical and output related. The physical parameters can again be split into geometry (e.g. domain, boundary conditions), flow properties (e.g. initial conditions, turbulence model, roughness) and other processes (e.g. salinity, temperature, wind). The numerical parameters are time step and drying and flooding criteria. The output parameters determine which data and results are going to be written to the output files. We will discuss the parameters for every group separately.

Physical parameters

The geometry is different for every test case. The domain and boundary conditions are treated in the text. The initial conditions on the flow velocity components is hard-coded set to zero. The water level on simulation start must be given by the user. In our test cases this is always an uniform condition. Furthermore, we have used, unless mentioned otherwise:

- a horizontal eddy viscosity of 1 m^2/s,
- the $k - \epsilon$ turbulence model with no background value,
- the Chezy bottom roughness formula specified by a constant (65 in both directions),
- no partial slip conditions.

Table E.1 shows the values of the relevant physical constants.

Table E.1: Values of the physical constants.

Quantity	Value
Gravity	9.81 m^2/s
Temperature	15 °C
Water Density	1000 kg/m^3
Salinity	31 *ppt*

Numerical parameters

For the time step several values have been used. They are mentioned in the text. By default the maximum criterion has been used for the drying and flooding mechanism. The threshold depth is set at 0.10 m.

Output parameters

By default we let Delft3D-FLOW write out all calculated variables for every time step.

Bibliography

AIAA, *Guide for the verification and validation of computational fluid dynamics simulations*, AIAA Guide G-077-1998 (Reston, VA, 1998).

Axelsson, O., *Iterative Solution Methods* (Cambridge University Press, 1994).

Bijvelds, M., *Numerical modelling of estuarine flow over steep topography*, Ph.D. thesis, Delft University of Technology (Delft, 2001).

Forsythe, G.E. and Moler, C.B., *Computer Solution of Linear Algebraic Systems* (Englewood Cliffs, NJ: Prentice-Hall, 1967).

Golub, G.H. and Van Loan, C.F., *Matrix Computations*, 2nd edn. (Baltimore: John Hopkins, 1989).

Jin, X.-Y., *Quasi-three-dimensional numerical modelling of flow and dispersion in shallow water*, Ph.D. thesis, Delft University of Technology (Delft, 1993).

Kester, J.A.Th.M. van and Stelling, G.S., 'Versnellen van TRISULA-3D', Delft Hydraulics Report Z81, WL | Delft Hydraulics (1992).

Launder, B.E. and Spalding, D., *Lectures in mathematical models of turbulence* (New York: Academic Press, 1972).

Mitchell, A.R. and Griffiths, D.F., *The Finite Difference Method in Partial Differential Equations*, pp. 59–70, 148–154 (New York: John Wiley, 1980).

Oberkampf, W.L. and Blottner, F.G., 'Issues in computational fluid dynamics code verification and validation', *AIAA Journal*, vol. 36, 687–695 (1998).

Phillips, N.G., 'A coordinate system having some special advantages for numerical forecasting', *Journal of Meteorology*, vol. 14, 184–185 (1957).

Rodi, W., 'Calculation of stably stratified shear-layer flows with a buoyancy-extended k-ϵ turbulence model', in Hunt, J.C.R. ed., *Turbulence and diffusion in stable environments*, pp. 111–140 (Oxford: Clarendon, 1985).

Stelling, G.S., *On the construction of computational methods for shallow water flow problems*, Ph.D. thesis, Delft University of Technology (Delft, 1984).

Uittenbogaard, R.E., Van Kester, J.A.Th.M. and Stelling, G.S., 'Implementation of three turbulence models in TRISULA for rectangular horizontal grids', Delft Hydraulics Report Z162, WL | Delft Hydraulics (Delft, 1992).

Varga, R.S., *Matrix Iterative Analysis* (New York: Prentice-Hall, 1962).

Wesseling, P., *Principles of Computational Fluid Dynamics*, Springer Series in Computational Mathematics, Vol.29 (Heidelberg: Springer, 2000).