

# Convergence of QuickFlow

A Steady State Solver for the Shallow Water Equations

Femke van Wageningen-Kessels

Report literature study

December 2006

Supervisors

Dr.ir. C. Vuik	TU Delft
Dr.ir. M. van Gijzen	TU Delft
Dr.ir. B. van 't Hof	VORtech
Ir. J. Dijkzeul	VORtech





# Contents

<b>1</b>	<b>Introduction</b>	<b>5</b>
1.1	Background . . . . .	5
1.2	Contents of this report . . . . .	6
<b>2</b>	<b>Shallow water equations</b>	<b>7</b>
2.1	From Navier-Stokes to Reynolds equations . . . . .	8
2.2	From Reynolds to 3D shallow water equations . . . . .	11
2.2.1	Surface and bottom boundary conditions . . . . .	11
2.2.2	Hydrostatic pressure distribution . . . . .	13
2.3	From 3D to 2D . . . . .	15
2.4	From Cartesian to curvilinear coordinates . . . . .	16
<b>3</b>	<b>Spatial discretization</b>	<b>18</b>
3.1	Spatial grid . . . . .	18
3.2	Methods for spatial discretization . . . . .	20
3.3	Discretization of momentum equations . . . . .	21
3.3.1	Discretization of advection term . . . . .	21
3.3.2	Discretization of cross advection term . . . . .	21
3.3.3	Discretization of pressure gradients . . . . .	22
3.3.4	Discretization of horizontal viscous terms . . . . .	22
3.3.5	Discretization of the Coriolis force term . . . . .	23
3.3.6	Discretization of the bottom friction term . . . . .	23
3.4	Discretization of the continuity equation . . . . .	24
3.5	Lateral boundary conditions . . . . .	26
3.5.1	Closed boundaries . . . . .	27
3.5.2	Open boundaries . . . . .	28
3.5.3	Water level boundary . . . . .	29
3.5.4	QH-boundary . . . . .	30
3.5.5	Velocity boundary . . . . .	31

3.5.6	Discharge boundary . . . . .	32
3.5.7	Riemann invariant boundary . . . . .	32
3.5.8	Reflection coefficient . . . . .	34
3.5.9	Drying and flooding . . . . .	34
<b>4</b>	<b>Solution methods</b>	<b>38</b>
4.1	Solution Methods in WAQUA . . . . .	38
4.1.1	ADI . . . . .	38
4.1.2	ADI applied to the shallow water equations . . . . .	41
4.2	Solution methods in QuickFlow . . . . .	46
4.2.1	Euler Backward . . . . .	47
4.2.2	Newton's method with line search . . . . .	48
<b>5</b>	<b>Test problems</b>	<b>51</b>
5.1	Theoretical test problem . . . . .	51
5.2	Real test problems . . . . .	53
5.2.1	Lek . . . . .	53
5.2.2	Randwijk . . . . .	53
<b>6</b>	<b>Research goals and plan</b>	<b>60</b>
6.1	Goal . . . . .	60
6.2	Possible improvements . . . . .	61
6.2.1	Boundary conditions for open boundaries . . . . .	61
6.2.2	Time iteration method . . . . .	61
6.2.3	Iteration algorithm . . . . .	63
6.2.4	Adaptive time stepping . . . . .	65
6.2.5	Pseudo water level . . . . .	66
6.3	Plan . . . . .	66
<b>7</b>	<b>Conclusion</b>	<b>67</b>
	<b>Notation</b>	<b>71</b>
	<b>APPENDICES</b>	<b>71</b>
<b>A</b>	<b>Relation WAQUA and QuickFlow</b>	<b>72</b>
<b>B</b>	<b>Stencils for spatial discretization</b>	<b>76</b>
<b>C</b>	<b>Thomas' algorithm</b>	<b>82</b>

# Chapter 1

## Introduction

This report describes the literature study as part of my Master thesis project. The aim of this study is to describe the present development of QuickFlow, a solver for the stationary shallow water equations, and inventorise possible methods to improve the software. Furthermore a plan is developed to improve the convergence of QuickFlow during the second part of this project.

### 1.1 Background

Watermanagement is very important in a densely populated country with many rivers like the Netherlands. Maintenance and improvement of dykes, riverbeds etc. needs constant attention. In the design of these dykes and riverbeds software is used to predict the water flow and levels.

Rijkswaterstaat is a dutch governmental institute that is engaged in such projects. VORtech Computing develops software that is used to do the necessary computations. This project is conducted at VORtech, which is an engineering and software company with a lot of mathematical expertise.

WAQUA is a software package that has been developed by Rijkswaterstaat and other companies to predict flows of rivers, seas and oceans. WAQUA uses the two dimensional shallow water equations to compute the flow velocity and water level and its development over time very accurately. One of the major drawbacks of WAQUA is the large amount of time it takes to find a solution. This is especially disadvantageous when one only wants to know the stationary solution and is not interested in very accurate time dependent results.

For this purpose QuickFlow was developed. QuickFlow is a program that is intended to solve the shallow water equations for application to rivers in the Netherlands. Its purpose is to quickly find the new steady state solution after an

intervention like rising a dyke or a change in the level or structure of the bottom.

QuickFlow is tightly linked to WAQUA. QuickFlow is based on WAQUA, and needs some input from it, but it has the advantage that it finds a solution much quicker than WAQUA. Towards the end of the design process the proposed changes still need evaluation by WAQUA.

At this moment QuickFlow can find solutions that look pretty much like the WAQUA-solution for relatively easy problems, such as a small part of a river with a simple geometry. There are some problems, however, with more complex geometries, especially when parts of the river bed become dry. This report investigates these problems.

## 1.2 Contents of this report

In the first part of this report the mathematical background of QuickFlow and, where necessary, the background of WAQUA is described. In Chapter 2 the mathematical model of the river flow is described. In Chapter 3 the space discretization is discussed, as well as the boundary conditions (Section 3.5). In Chapter 4 we discuss the solution methods that are used by WAQUA, and the ones that are used by QuickFlow. These chapters focus on the situation in QuickFlow, which sometimes differs from the situation in WAQUA. Where necessary we make a remark on the differences between WAQUA and QuickFlow.

In the last part of this report we focus on how this project should be continued. Chapter 5 describes three test problems that will be used in the next phase of this project. Chapter 6 discusses the possible improvements of QuickFlow and gives an outline of how these improvements can be implemented.

In Appendix A the relation between QuickFlow and WAQUA is described in more detail.

## Chapter 2

# Shallow water equations

In this chapter we will describe the problem at hand. The shallow water equations in general describe the flow of a fluid, not necessarily water, where the thickness of a fluid layer is small compared to some horizontal length scale. In this project the flow in a river is described by the shallow water equations. They can be applied because the typical vertical scales (e.g. the water depth) are much smaller than any horizontal typical scale (e.g. the width of the river).

Hereafter the Reynolds equations for the statistical average of a turbulent flow are derived from the Navier Stokes equations. After several conditions these equations are integrated and we find the 2-dimensional shallow water equations.

Furthermore, the derivation of the 2D shallow water equations is described in more detail, the theory is taken from [1, Chapter 2]. The shallow water equations are applied to a part of a river. Figure 2.1 shows the coordinate system and the bottom and surface boundaries.

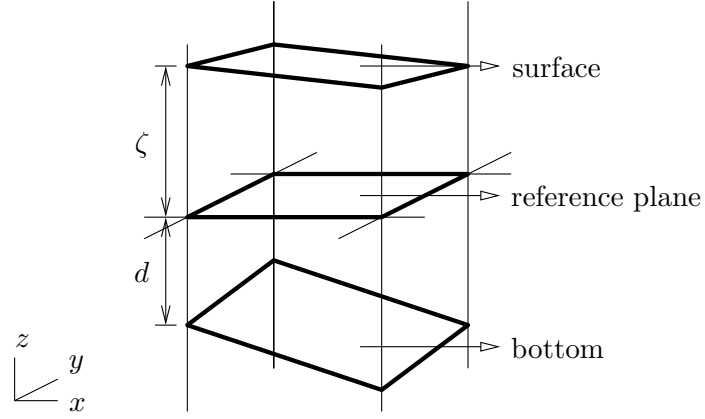


Figure 2.1: The coordinate system and the surface and bottom boundaries.

## 2.1 From Navier-Stokes to Reynolds equations

The shallow water equations can be derived from the incompressible Navier Stokes equations. These equations are given by:

$$\left. \begin{aligned} \frac{\partial}{\partial t}(\rho u) + \frac{\partial}{\partial x}(\rho u^2) + \frac{\partial}{\partial y}(\rho uv) + \frac{\partial}{\partial z}(\rho uw) - \rho f v + \frac{\partial p}{\partial x} \\ - \left( \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \right) &= 0, \\ \frac{\partial}{\partial t}(\rho v) + \frac{\partial}{\partial x}(\rho uv) + \frac{\partial}{\partial y}(\rho v^2) + \frac{\partial}{\partial z}(\rho vw) + \rho f u + \frac{\partial p}{\partial y} \\ - \left( \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \right) &= 0, \\ \frac{\partial}{\partial t}(\rho w) + \frac{\partial}{\partial x}(\rho uw) + \frac{\partial}{\partial y}(\rho vw) + \frac{\partial}{\partial z}(\rho w^2) + \frac{\partial p}{\partial z} + \rho g \\ - \left( \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \right) &= 0, \end{aligned} \right\} \quad (2.1)$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x}(\rho u) + \frac{\partial}{\partial y}(\rho v) + \frac{\partial}{\partial z}(\rho w) = 0. \quad (2.2)$$

### Viscous stresses

In the momentum conservation equations (2.1)  $\tau_{ij}$  represent the viscous stresses, which can be expressed in terms of the fluid deformation rate:

$$\frac{\tau_{ij}}{\rho} = \nu \left( \frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} \right), \quad \text{with } \nu \text{ the viscosity.} \quad (2.3)$$



### Density

In general the density  $\rho$  depends on the pressure  $p$ , the temperature and the salinity. The density depends on the location  $(x, y, z)$  only indirectly via the pressure, temperature and salinity. In our problem the temperature and salinity are taken constant in space and time. We consider an incompressible fluid (water) which means that the density  $\rho$  does not depend on the pressure  $p$ . Furthermore, since the density does not vary in time, the first term of the continuity equation (2.2) (also called mass conservation equation) is zero. There is only one term left where density variations are important: the gravity term in the  $z$ -momentum equation of (2.1). This is called the Boussinesq approximation. Summarizing we take the density  $\rho = \rho_0$  a constant reference density except in the gravity term  $\rho g$ .

### Coriolis force

The terms  $\rho f u$  and  $\rho f v$  deal with the Coriolis force. This force is due to the rotation of the earth and only has little influence on the flow in a relatively small system like a river. Nonetheless it is implemented because the implementation is easy and does not bring much cost. The Coriolis parameter  $f$  is computed as follows:

$$f = 2|\Omega| \sin \phi, \quad (2.4)$$

with  $\Omega$  the angular speed of the rotation of the earth,  
and  $\phi$  the geographic latitude.

### Splitting of velocity

In our application, one is only interested in large scale features, therefore we do not want to take into account the small scale structures (e.g. turbulence) as such. In section 2.1 we incorporate the turbulence in the eddy viscosity. We suppose we can split each velocity in some sort of 'mean' and a 'random' variation:

$$\begin{aligned} u &= U + u', \\ v &= V + v', \\ w &= W + w'. \end{aligned} \quad (2.5)$$

From now on we will only consider the 'mean' velocities  $U$ ,  $V$ , and  $W$  and neglect the random variations  $u'$ ,  $v'$  and  $w'$ , except for the case discussed below (advection terms and Reynolds stresses).

### Advection terms and Reynolds stresses

If we apply the splitting (2.5) on the advection terms Reynolds stresses will occur. This is discussed for the cross advection term  $\frac{\partial}{\partial y}(uv)$ . Applying the splitting on this cross advection term and on the viscous term  $\frac{\partial \tau_{xy}}{\partial x}$  we find:

$$\begin{aligned} \frac{\partial}{\partial x}(uv) &= \frac{\partial}{\partial x}(UV) + \underbrace{\frac{\partial}{\partial x}(Uv') + \frac{\partial}{\partial x}(u'V)}_{\text{can be neglected}} + \frac{\partial}{\partial x}(u'v') \\ &\approx \frac{\partial}{\partial x}(UV) + \frac{\partial}{\partial x}(u'v'), \end{aligned} \quad (2.6)$$

$$\begin{aligned} \frac{\partial \tau_{xy}}{\partial x} &= \rho \nu \frac{\partial}{\partial x} \left( \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) \\ &= \rho \nu \frac{\partial}{\partial x} \left( \frac{\partial U}{\partial y} + \frac{\partial V}{\partial x} + \underbrace{\frac{\partial u'}{\partial y} + \frac{\partial v'}{\partial x}}_{\text{can be neglected}} \right) \\ &= \rho \nu \frac{\partial}{\partial x} \left( \frac{\partial U}{\partial y} + \frac{\partial V}{\partial x} \right). \end{aligned} \quad (2.7)$$

The terms  $\frac{\partial}{\partial x}(Uv') + \frac{\partial}{\partial x}(u'V)$  can be neglected because  $U$  and  $v'$  are uncorrelated, as well as  $V$  and  $u'$ . We will take an integral over these terms when we apply the finite volume method discussed in Chapter 3. This integral will reduce to zero. A similar reasoning holds for  $\frac{\partial u'}{\partial y} + \frac{\partial v'}{\partial x}$ , which will reduce to zero as well when we apply the finite volume method.

Combining (2.6) and (2.7) we find:

$$\begin{aligned} \rho \frac{\partial}{\partial x}(uv) - \frac{\partial \tau_{xy}}{\partial x} &\approx \rho \left( \frac{\partial}{\partial x}(UV) + \frac{\partial}{\partial x}(u'v') - \nu \frac{\partial}{\partial x} \left( \frac{\partial U}{\partial y} + \frac{\partial V}{\partial x} \right) \right) \\ &= \rho \frac{\partial}{\partial x}(UV) + \rho \frac{\partial}{\partial x} \left( \nu \left( \frac{\partial U}{\partial y} + \frac{\partial V}{\partial x} \right) + (u'v') \right). \end{aligned} \quad (2.8)$$

In general we can write:

$$\rho \frac{\partial}{\partial x_i}(u_i u_j) - \frac{\partial \tau_{ij}}{\partial x_i} \approx \rho \frac{\partial}{\partial x_i}(U_i U_j) + \rho \frac{\partial}{\partial x_i} \left( \nu \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) + (u'_i u'_j) \right) \quad (2.9)$$

The terms  $u'_i u'_j$  are called Reynolds stresses. These stresses are due to the turbulence of the flow. Therefore we substitute the viscosity  $\nu$  with the effective turbulent or eddy viscosity  $\nu_t$ :

$$\tau_{ij} \approx \nu_t \left( \frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right). \quad (2.10)$$

We use this eddy viscosity  $\nu_t$  because we have a small scale turbulence, which can not be implemented in the model as such. Therefore we replace the turbulence by a fictitious diffusion term, the eddy viscosity. In general it depends quadratically on the step sizes  $\Delta x$ ,  $\Delta y$  and  $\Delta z$ . Here the eddy viscosity is taken to be constant.

### Reynolds equations

We apply the Boussinesq approximation, substitute the splitting of (2.5) in the momentum conservation equations (2.1) and the continuity equation (2.2) and we use the eddy viscosity as discussed above. We now find the Reynolds equations for the statistical average of a turbulent flow:

$$\begin{aligned}
\rho_0 \frac{\partial}{\partial t}(U) + \rho_0 \frac{\partial}{\partial x}(U^2) + \rho_0 \frac{\partial}{\partial y}(UV) + \rho_0 \frac{\partial}{\partial z}(UW) - \rho_0 fV + \frac{\partial p}{\partial x} \\
- \left( \frac{\partial \tau_{xx}}{\partial x} + \frac{\partial \tau_{xy}}{\partial y} + \frac{\partial \tau_{xz}}{\partial z} \right) &= 0, \\
\rho_0 \frac{\partial}{\partial t}(V) + \rho_0 \frac{\partial}{\partial x}(UV) + \rho_0 \frac{\partial}{\partial y}(V^2) + \rho_0 \frac{\partial}{\partial z}(VW) + \rho_0 fU + \frac{\partial p}{\partial y} \\
- \left( \frac{\partial \tau_{yx}}{\partial x} + \frac{\partial \tau_{yy}}{\partial y} + \frac{\partial \tau_{yz}}{\partial z} \right) &= 0, \quad (2.11) \\
\rho_0 \frac{\partial}{\partial t}(W) + \rho_0 \frac{\partial}{\partial x}(UW) + \rho_0 \frac{\partial}{\partial y}(VW) + \rho_0 \frac{\partial}{\partial z}(W^2) + \frac{\partial p}{\partial z} + \rho g \\
- \left( \frac{\partial \tau_{zx}}{\partial x} + \frac{\partial \tau_{zy}}{\partial y} + \frac{\partial \tau_{zz}}{\partial z} \right) &= 0, \\
\frac{\partial}{\partial x}(U) + \frac{\partial}{\partial y}(V) + \frac{\partial}{\partial z}(W) &= 0, \\
&\text{with } \tau_{ij} \text{ as in (2.10).}
\end{aligned}$$

## 2.2 From Reynolds to 3D shallow water equations

### 2.2.1 Surface and bottom boundary conditions

The equations derived above have to be complemented with boundary conditions. In this section the boundary conditions at the free water surface and at the solid bottom will be discussed. Other boundary conditions will be discussed in Section 3.5.

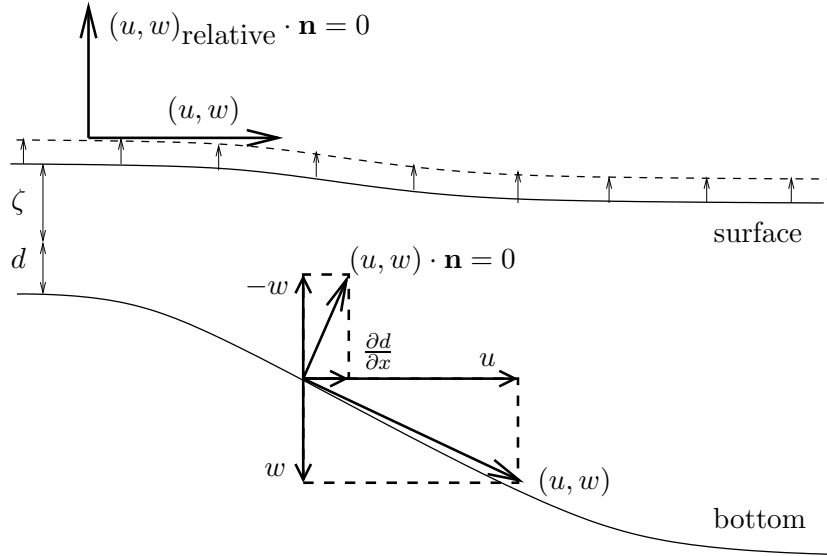


Figure 2.2: A schematic cross sectional view of the kinematic boundary conditions for the surface and bottom: a water particle can not cross the bottom or the surface.

### Kinematic boundary conditions

The kinematic boundary conditions prescribe that water particles can not cross the solid bottom nor the free water surface. For the bottom the normal velocity component must vanish. Since the free surface might be moving, the normal velocity of the fluid should equal the normal velocity of the surface. See also Figure 2.2.

$$\begin{aligned} u \frac{\partial d}{\partial x} + v \frac{\partial d}{\partial y} - w &= 0 \quad \text{for } z = -d, \quad \text{at the bottom,} \\ \frac{\partial \zeta}{\partial t} + u \frac{\partial \zeta}{\partial x} + v \frac{\partial \zeta}{\partial y} - w &= 0 \quad \text{for } z = \zeta, \quad \text{at the free surface.} \end{aligned} \quad (2.12)$$

### Dynamic boundary conditions

We have dynamic boundary conditions for the forces that act at the bottom and surface boundaries. For the bottom we have the no slip condition.

For the free surface we assume continuity of stresses: the stress just below the surface is equal to the stress just above it. This gives a condition on the pressure and on the shear stress. This shear stress is usually a result of the wind

over a sloping surface. This wind stress factor is neglected in QuickFlow (but it is implemented in WAQUA).

The resulting boundary conditions (neglecting the wind stress) are:

$$\begin{aligned} u = v = 0 & \quad \text{for } z = -d, & \quad \text{at the bottom,} \\ p = p_{\text{atm}} & \quad \text{for } z = \zeta, & \quad \text{at the free surface.} \end{aligned} \quad (2.13)$$

### 2.2.2 Hydrostatic pressure distribution

There are several length scales involved in the shallow water equations, they include water depth as a vertical scale and river width as a horizontal scale. Whenever the ratio of a typical vertical length  $\hat{L}$  to a typical horizontal length  $\hat{H}$  is smaller than about 0.05 the shallow water equations can be applied, see [1, section 2.3]. Furthermore we indicate the typical horizontal velocity  $\hat{U}$ .

Scale and dimensional analysis (for details see [1, Section 2.3]) leads to three dimensionless numbers:

$$\begin{aligned} \text{Re}_t &= \frac{\hat{U}\hat{H}}{\nu_t} & \text{Reynolds number (viscosity),} \\ \text{Ro} &= \frac{\hat{U}}{f\hat{L}} & \text{Rossby number (Coriolis),} \\ \text{Fr} &= \frac{\hat{U}}{\sqrt{g\hat{H}}} & \text{Froude number (wave velocity).} \end{aligned} \quad (2.14)$$

For our application (Dutch rivers) we can use the following values:

$$\begin{aligned} \hat{U} &\approx 1 & \text{m/s,} \\ \hat{H} &\approx 20 & \text{m,} \\ \hat{L} &\approx 500 & \text{m,} \\ \nu_t &\approx 0.5 & \text{m}^2/\text{s,} \\ f &\approx 1 \times 10^{-4} & \text{s}^{-1}, \\ g &\approx 9.8 & \text{m/s}^2. \end{aligned} \quad (2.15)$$

We now find the following dimensionless numbers with their respective meaning:

$$\begin{aligned} \text{Re}_t &\approx 40 > 1 & \text{viscosity is small but can not be neglected,} \\ \text{Ro} &\approx 20 > 1 & \text{Coriolis force is of minor importance,} \\ \text{Fr} &\approx 0.07 \ll 1 & \text{wave velocity is much higher than water velocity}^1. \end{aligned} \quad (2.16)$$

Note that in the Reynolds number we include the turbulence in the viscosity (eddy viscosity), as we discussed in Section 2.1. If we do not include the turbulence and we use the viscosity of water  $\nu = 10^{-3}$  we find:

$$\text{Re} = \frac{\hat{U} \hat{H}}{\nu} \approx 2 \times 10^4. \quad (2.17)$$

This indicates that the flow is indeed turbulent: the Reynolds number is much higher than 2000.

Further analysis leads to the conclusion that all terms are small relative to the gravitational acceleration  $g$  and only the pressure gradient remains to balance it. Therefore we can apply the hydrostatic pressure gradient (for further details on this derivation see [1, Section 2.3-2.4]):

$$\frac{\partial p}{\partial z} = -\rho_0 g. \quad (2.18)$$

This hydrostatic pressure distribution together with the boundary condition  $p = p_{\text{atm}}$  from (2.13) gives:

$$\begin{aligned} p(x, y, z, t) &= \int_z^{\zeta(x, y, z, t)} \rho_0 g \, dz + p_{\text{atm}} \\ &= \rho_0 g (\zeta(x, y, z, t) - z) + p_{\text{atm}} \Rightarrow \\ &\quad \frac{\partial p}{\partial x} = \rho_0 g \frac{\partial \zeta}{\partial x} \quad \text{and} \\ &\quad \frac{\partial p}{\partial y} = \rho_0 g \frac{\partial \zeta}{\partial y}. \end{aligned} \quad (2.19)$$

---

<sup>1</sup>Because of the small Froude number we call the flow subcritical. This can be compared to subsonic velocity in gas dynamics: a velocity lower than the speed of sound.

### 3D Shallow water equations

Now we can write down the 3D shallow water equations: the horizontal momentum equations together with the continuity equation from (2.11):

$$\begin{aligned}
\frac{\partial}{\partial t}(U) + \frac{\partial}{\partial x}(U^2) + \frac{\partial}{\partial y}(UV) + \frac{\partial}{\partial z}(UW) - fV + g\frac{\partial\zeta}{\partial x} \\
- \frac{1}{\rho_0} \left( \frac{\partial\tau_{xx}}{\partial x} + \frac{\partial\tau_{xy}}{\partial y} + \frac{\partial\tau_{xz}}{\partial z} \right) &= 0, \\
\frac{\partial}{\partial t}(V) + \frac{\partial}{\partial x}(UV) + \frac{\partial}{\partial y}(V^2) + \frac{\partial}{\partial z}(VW) + fU + g\frac{\partial\zeta}{\partial y} \\
- \frac{1}{\rho_0} \left( \frac{\partial\tau_{yx}}{\partial x} + \frac{\partial\tau_{yy}}{\partial y} + \frac{\partial\tau_{yz}}{\partial z} \right) &= 0, \\
\frac{\partial}{\partial x}(U) + \frac{\partial}{\partial y}(V) + \frac{\partial}{\partial z}(W) &= 0,
\end{aligned} \tag{2.20}$$

with  $\tau_{ij}$  as in (2.10).

## 2.3 From 3D to 2D

Until now we have discussed the 3 dimensional case. We want to simplify the problem to a 2 dimensional one. Therefore we integrate the 3D shallow water equations (2.20) over depth  $H = d + \zeta$ .

We have to integrate the stresses over the depth. The depth integrated stresses can be modeled in many ways: the influence of many factors can be either implemented or neglected. In QuickFlow we chose to implement the bottom stress  $\tau_{\text{bottom}}$  and the viscosity. The bottom stress is split into two terms:  $\tau_{\text{bottom},x}$  in the  $x$ -direction and  $\tau_{\text{bottom},y}$  in the  $y$ -direction. Furthermore, the 3D eddy viscosity  $\nu_t$  is replaced by an horizontal 2D equivalent  $\nu_H$ .

More details on this derivation can be found in [1, Section 2.5]. The 2D shallow water equations can now be written as follows:

$$\begin{aligned}
\frac{\partial}{\partial t}(HU) + \frac{\partial}{\partial x}(HU^2) + \frac{\partial}{\partial y}(HUV) - fHV + gH\frac{\partial\zeta}{\partial x} + \frac{\tau_{\text{bottom},x}}{\rho_0} \\
- H\nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) &= 0, \\
\frac{\partial}{\partial t}(HV) + \frac{\partial}{\partial x}(HUV) + \frac{\partial}{\partial y}(HV^2) + fHU + gH\frac{\partial\zeta}{\partial y} + \frac{\tau_{\text{bottom},y}}{\rho_0} & \quad (2.21) \\
- H\nu_H \left( \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \right) &= 0, \\
\frac{\partial\zeta}{\partial t} + \frac{\partial}{\partial x}(HU) + \frac{\partial}{\partial y}(HV) &= 0.
\end{aligned}$$

Here the viscosity term is not conservative. Therefore we can not really call this the conservative form. However, a non-conservative form of the momentum equations can be obtained by substitution of the continuity equation in the momentum equations in (2.21). Hereby we also use:

$$\begin{aligned}
& \frac{\partial}{\partial t}(HU) + \frac{\partial}{\partial x}(HU^2) + \frac{\partial}{\partial y}(HUV) \quad \text{chain rule} \\
U \frac{\partial}{\partial t}\zeta + H \frac{\partial}{\partial t}U + U \frac{\partial}{\partial x}(HU) + HU \frac{\partial}{\partial x}U + U \frac{\partial}{\partial y}(HV) + HV \frac{\partial}{\partial y}U & \quad (2.21), \text{ continuity} \\
H \left( \frac{\partial}{\partial t}U + U \frac{\partial}{\partial x}U + V \frac{\partial}{\partial y}U \right). & \quad (2.22)
\end{aligned}$$

This results in a non-conservative form of the 2D shallow water equations:

$$\begin{aligned}
\frac{\partial U}{\partial t} + U \frac{\partial U}{\partial x} + V \frac{\partial U}{\partial y} - fV + g\frac{\partial\zeta}{\partial x} - \frac{\tau_{\text{bottom},x}}{\rho_0 H} - \nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) &= 0, \\
\frac{\partial V}{\partial t} + U \frac{\partial V}{\partial x} + V \frac{\partial V}{\partial y} + fU + g\frac{\partial\zeta}{\partial y} - \frac{\tau_{\text{bottom},y}}{\rho_0 H} - \nu_H \left( \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \right) &= 0 \quad (2.23) \\
\frac{\partial\zeta}{\partial t} + \frac{\partial H}{\partial x}U + H \frac{\partial U}{\partial x} + \frac{\partial H}{\partial y}V + H \frac{\partial V}{\partial y} &= 0.
\end{aligned}$$

## 2.4 From Cartesian to curvilinear coordinates

Until now we have used Cartesian coordinates. In WAQUA and QuickFlow an orthogonal curvilinear grid is used. One of the major advantages is that curvilinear gridlines can follow bending boundaries more smoothly. By demanding orthogonality we reduce the computational work compared to a nonorthogonal curvilinear grid.



The theory below is taken from [2, Section 2.4].  
The transformation is defined as follows:

$$\begin{aligned}x &= X(\xi, \eta), \\y &= Y(\xi, \eta).\end{aligned}\tag{2.24}$$

We have the following transformation coefficients:

$$\begin{aligned}g_{\xi\xi} &= \left(\frac{\partial x}{\partial \xi}\right)^2 + \left(\frac{\partial y}{\partial \xi}\right)^2, \\g_{\eta\eta} &= \left(\frac{\partial x}{\partial \eta}\right)^2 + \left(\frac{\partial y}{\partial \eta}\right)^2.\end{aligned}\tag{2.25}$$

The actual generation of the grid is not done within WAQUA or QuickFlow. Therefore it is not discussed here. In [2, Section 2.4] some more details are given and references to relevant documentation can be found.

Now we can write the 2D shallow water equations (2.23) in orthogonal curvilinear coordinates:

$$\begin{aligned}\frac{\partial}{\partial t}(U) + \frac{U}{\sqrt{g_{\xi\xi}}}\frac{\partial U}{\partial \xi} + \frac{V}{\sqrt{g_{\eta\eta}}}\frac{\partial U}{\partial \eta} + \frac{UV}{\sqrt{g_*}}\frac{\partial \sqrt{g_{\xi\xi}}}{\partial \eta} - \frac{V^2}{\sqrt{g_*}}\frac{\partial \sqrt{g_{\eta\eta}}}{\partial \xi} \\- fV + \frac{g}{\sqrt{g_{\xi\xi}}}\frac{\partial \zeta}{\partial \xi} + \frac{\tau_{\text{bottom},\xi}}{\rho_0 H} - \frac{\nu_H}{\sqrt{g_{\xi\xi}}}\frac{\partial F^1}{\partial \xi} + \frac{\nu_H}{\sqrt{g_{\eta\eta}}}\frac{\partial F^2}{\partial \eta} = 0, \\ \frac{\partial}{\partial t}(V) + \frac{U}{\sqrt{g_{\xi\xi}}}\frac{\partial V}{\partial \xi} + \frac{V}{\sqrt{g_{\eta\eta}}}\frac{\partial V}{\partial \eta} + \frac{UV}{\sqrt{g_*}}\frac{\partial \sqrt{g_{\eta\eta}}}{\partial \xi} - \frac{U^2}{\sqrt{g_*}}\frac{\partial \sqrt{g_{\xi\xi}}}{\partial \eta} \\+ fU + \frac{g}{\sqrt{g_{\eta\eta}}}\frac{\partial \zeta}{\partial \eta} + \frac{\tau_{\text{bottom},\eta}}{\rho_0 H} - \frac{\nu_H}{\sqrt{g_{\eta\eta}}}\frac{\partial F^1}{\partial \eta} - \frac{\nu_H}{\sqrt{g_{\xi\xi}}}\frac{\partial F^2}{\partial \xi} = 0, \\ \frac{\partial \zeta}{\partial t} + \frac{1}{\sqrt{g_*}}\frac{\partial}{\partial \xi}(HU\sqrt{g_{\eta\eta}}) + \frac{1}{\sqrt{g_*}}\frac{\partial}{\partial \eta}(HV\sqrt{g_{\xi\xi}}) = 0,\end{aligned}\tag{2.26}$$

$$\begin{aligned}\text{with } F^1 &= \frac{1}{\sqrt{g_*}}\left(\frac{\partial}{\partial \xi}(U\sqrt{g_{\eta\eta}}) + \frac{\partial}{\partial \eta}(V\sqrt{g_{\xi\xi}})\right), \\F^2 &= \frac{1}{\sqrt{g_*}}\left(\frac{\partial}{\partial \xi}(V\sqrt{g_{\eta\eta}}) - \frac{\partial}{\partial \eta}(U\sqrt{g_{\xi\xi}})\right), \\ \text{and } g_* &= \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \eta} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \eta},\end{aligned}\tag{2.27}$$

the determinant of the Jacobian of the transformation.

In the next chapters we discuss the spatial and time discretization. For simplicity we will not use this curvilinear transformation, instead Cartesian coordinates, like in (2.23), are used. The discretization for a curvilinear grid is almost the same, only some extra cross terms appear.

## Chapter 3

# Spatial discretization

In this chapter we describe the numerical discretization of the 2D shallow water equations in non-conservative form (2.23).

The discretization below is mostly taken from the Technical Documentation of WAQUA [2, Chapter 4 and 6]. In general QuickFlow uses the same spatial discretization as WAQUA. WAQUA, however, has some advanced options that can be switched on and off by the user. In such situations the option which is most suitable for steady state simulations is used.

The spatial discretization we describe in this chapter will be used in the next chapter on the Alternating Direction Implicit (ADI) method. Some knowledge of the ADI method is needed to understand the spatial discretization.

The ADI method is used for time discretization. The ADI method splits every timestep into two stages. In both stages the equations are solved. In the first stage some terms are taken implicitly while the other terms are taken explicitly. In the second stage this is the other way around. In this chapter we discuss this combination of implicit and explicit discretization.

### 3.1 Spatial grid

In Figure 3.1 the computational grid is shown. This grid is also known as the Arakawa-C grid.

Note that the grid is staggered: the velocities  $U$  and  $V$ , depth  $d$  and water level  $\zeta$  are located at different gridpoints. We use a staggered grid for numerical stability.

To prevent indices like  $(m + 1/2, n)$ ,  $(m, n + 1/2)$  and  $(m + 1/2, n + 1/2)$ , we

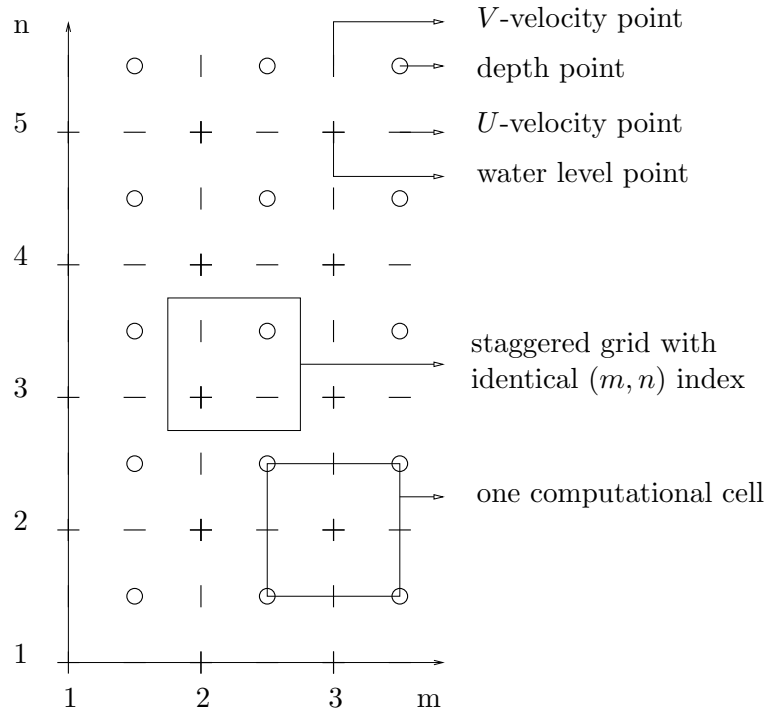


Figure 3.1: The computational grid (also known as the Arakawa-C grid).

only use integer-numbered indices:

$$\begin{aligned} U_{m+1/2,n} &\rightarrow U_{m,n}, \\ V_{m,n+1/2} &\rightarrow V_{m,n}, \\ d_{m+1/2,n+1/2} &\rightarrow d_{m,n}. \end{aligned} \tag{3.1}$$

Below we discuss the discretization on a Cartesian grid. If we have an orthogonal curvilinear grid, the discretization has to be adjusted.

### 3.2 Methods for spatial discretization

In WAQUA and QuickFlow both finite difference methods and finite volume methods are used for spatial discretization. Hereafter we will discuss which method is used for which terms. First the methods are discussed in general.

A finite difference method is used for the momentum equations. On a curvilinear grid it would be very difficult to achieve conservation of momentum, even with the finite volume method. Therefore, we use the finite difference method, which is easier to implement. In [3], a method is described to achieve conservation of mass, momentum and energy, using finite differences. However, this method is not implemented in WAQUA or in QuickFlow.

Several finite difference schemes are used for the terms in the momentum equations. We use central difference and upwind methods. The advantage of the central difference method is the second order accuracy (this accuracy can be checked using Taylor expansion). However numerical oscillations (wiggles) may occur near steep gradients. A first order upwind method is unconditionally wiggle free, but introduces a truncation error in the form of a second-order artificial viscosity term, which smoothens the computed solution. Third order upwinding is not free from wiggles but introduces fourth-order artificial viscosity  $\frac{\partial^4 U}{\partial y^4}$ . (This can be checked using Taylor expansion.) This artificial viscosity suppresses any wiggles without smoothing the solution too much.

A finite volume method is used for the continuity equation. This is because conservation of mass is very important here and can easily be achieved with the finite volume method.

A more detailed argumentation for the choice of methods used in WAQUA can be found in [4, Chapter 1].

The stencils for the spatial discretizations are shown in Appendix B.

### 3.3 Discretization of momentum equations

In this section we derive the spatial discretization of the U-momentum equation from (2.23):

$$\frac{\partial U}{\partial t} + U \frac{\partial U}{\partial x} + V \frac{\partial U}{\partial y} - fV + g \frac{\partial \zeta}{\partial x} - \frac{\tau_{\text{bottom},x}}{\rho_0 H} - \nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) = 0 \quad (3.2)$$

The spatial discretization of the V-momentum equation follows in a similar way.

#### 3.3.1 Discretization of advection term

On the advection term  $U \frac{\partial U}{\partial x}$  a central difference operator is applied. On a uniform grid this leads to:

$$U \frac{\partial U}{\partial x} \Big|_{m,n} \approx U_{m,n} \frac{U_{m+1,n} - U_{m-1,n}}{2\Delta x} \quad (3.3)$$

#### 3.3.2 Discretization of cross advection term

The cross advection term  $V \frac{\partial U}{\partial y}$  is approximated by the Cyclic method. This is a splitting of a third order upwind finite difference scheme for the first order derivative into two second order consistent discretizations. They are successively used in the stages of the ADI scheme which will be discussed in more detail in 4.

Before continuing with the Cyclic method we interpolate the  $V$ -velocity. Because the  $V$ -velocity gridpoints do not overlap the  $U$ -velocity gridpoints the  $V$ -velocity is approximated by an arithmetic average of the four surrounding  $V$ -velocity points  $\bar{V}$ . In an interior point this leads to:

$$\begin{aligned} V_{m,n} &= \bar{V}_{m,n} \\ &\approx \frac{V_{m+1,n} + V_{m,n+1} + V_{m-1,n} + V_{m,n-1}}{4}. \end{aligned} \quad (3.4)$$

The cross advection term is now discretized by third order upwinding:

$$V \frac{\partial U}{\partial y} \Big|_{m,n} \approx \begin{cases} \bar{V}_{m,n} \frac{U_{m,n+2} + 4U_{m,n+1} + 18U_{m,n} - 28U_{m,n-1} + 5U_{m,n-2}}{24\Delta y} & \text{if } V_{m,n} \geq 0, \\ \bar{V}_{m,n} \frac{-5U_{m,n+2} + 28U_{m,n+1} - 18U_{m,n} - 4U_{m,n-1} - U_{m,n-2}}{24\Delta y} & \text{if } V_{m,n} < 0. \end{cases} \quad (3.5)$$

This discretization is split into two second-order discretizations:

$$V \frac{\partial U}{\partial y} \Big|_{m,n}^{\text{ex}} \approx \bar{V}_{m,n} \frac{U_{m,n+2} + 4U_{m,n+1} - 4U_{m,n-1} - U_{m,n-2}}{12\Delta y}, \quad (3.6)$$

$$V \frac{\partial U}{\partial y} \Big|_{m,n}^{\text{im}} \approx \begin{cases} \bar{V}_{m,n} \frac{3U_{m,n} - 4U_{m,n-1} + U_{m,n-2}}{2\Delta y} & \text{if } V_{m,n} \geq 0, \\ \bar{V}_{m,n} \frac{-U_{m,n+2} + 4U_{m,n+1} - 3U_{m,n}}{2\Delta y} & \text{if } V_{m,n} < 0. \end{cases} \quad (3.7)$$

(3.6) is explicit and is used in the first stage of the ADI-method (see Section 4.1.1), (3.7) is implicit and is used in the second stage of the ADI-method.

Note that the 'average' of the explicit discretization (3.6) and the implicit discretization (3.7) is the third order upwind discretization (3.5):

$$\frac{1}{2} \left( V \frac{\partial U}{\partial y} \Big|_{m,n}^{\text{ex}} + V \frac{\partial U}{\partial y} \Big|_{m,n}^{\text{im}} \right) = V \frac{\partial U}{\partial y} \Big|_{m,n}. \quad (3.8)$$

### 3.3.3 Discretization of pressure gradients

The 2D pressure gradient is discretized with a central difference scheme:

$$g \frac{\partial \zeta}{\partial x} \Big|_{m,n} \approx g \frac{\zeta_{m+1,n} - \zeta_{m,n}}{\Delta x}. \quad (3.9)$$

### 3.3.4 Discretization of horizontal viscous terms

The horizontal viscous term  $\nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right)$  is discretized with the central difference method.

Therefore we first need an approximation of  $\frac{\partial^2 U}{\partial x^2}$ :

$$\begin{aligned} \frac{\partial^2 U}{\partial x^2} \Big|_{m,n} &\approx \frac{1}{\Delta x} \left( \frac{\partial U}{\partial x} \Big|_{m+1/2,n} - \frac{\partial U}{\partial x} \Big|_{m-1/2,n} \right) \\ &\approx \frac{1}{\Delta x} \left( \frac{U_{m+1,n} - U_{m,n}}{\Delta x} - \frac{U_{m,n} - U_{m-1,n}}{\Delta x} \right) \\ &= \frac{U_{m+1,n} - 2U_{m,n} + U_{m-1,n}}{\Delta x^2}. \end{aligned} \quad (3.10)$$

Similarly we find

$$\frac{\partial^2 U}{\partial y^2} \Big|_{m,n} \approx \frac{U_{m,n+1} - 2U_{m,n} + U_{m,n-1}}{\Delta y^2}. \quad (3.11)$$

Now we find the discretization of the horizontal viscous term

$$\nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right) \Big|_{m,n} \approx \nu_H \left( \frac{U_{m+1,n} - 2U_{m,n} + U_{m-1,n}}{\Delta x^2} + \frac{U_{m,n+1} - 2U_{m,n} + U_{m,n-1}}{\Delta y^2} \right). \quad (3.12)$$

### 3.3.5 Discretization of the Coriolis force term

The  $V$ -velocity in the Coriolis force term  $fV$  is approximated by an arithmetic average of the four surrounding  $V$ -velocity points, see (3.4).

### 3.3.6 Discretization of the bottom friction term

The bottom friction term  $\frac{\tau_{\text{bottom},x}}{\rho_0 H}$  is approximated using the Chézy coefficient  $C_{2D}$ . This is an empirical coefficient and can be determined in several ways:

$$C_{2D} = \begin{cases} \text{Chézy coefficient (m}^{1/2}/\text{s),} & \text{no conversion necessary,} \\ \frac{\sqrt[6]{H}}{n}, & \text{Chézy,} \\ & \text{with } n \text{ the Manning coefficient,} \\ & \text{Manning,} \\ 18^{10} \log \left( \max \left( \frac{12H}{k_s}, 1.0129 \right) \right), & \text{with } k_s \text{ the Nikuradse roughness length,} \\ & \text{White-Colebrook.} \end{cases} \quad (3.13)$$

In [2, Section 6.2] and in [5, Section 3.4.2.4-3.4.2.6] more detailed descriptions of the approximation of the bottom friction by the Chézy coefficient are given.

In WAQUA the user can define which approximation for the bottom friction he wants to use. The resulting Chézy coefficient differs over the spatial domain. The user furthermore defines how often (e.g. every 1 or every 10 minutes) should be recomputed. The Chézy coefficient that was used in the last WAQUA-iteration is subsequently read by QuickFlow and used in its computations.

Using this coefficient, the stress in  $U$ -velocity direction due to bottom friction can be expressed as follows:

$$\tau_{\text{bottom},x} = \frac{g\rho_0 U |\mathbf{U}|}{C_{2D,x}^2}, \quad (3.14)$$

with  $|\mathbf{U}| = \sqrt{U^2 + V^2}$  the magnitude of the horizontal velocity.

This leads to the following approximation of the bottom friction:

$$\begin{aligned} \frac{\tau_{\text{bottom},x}}{\rho_0 H} \Big|_{m,n} &= \frac{gU|\mathbf{U}|}{HC_{2D,x}^2} \Big|_{m,n} \\ &\approx \frac{gU_{m,n}\sqrt{U_{m,n}^2 + \bar{V}_{m,n}^2}}{H_{m,n}C_{2D,x}^2} \end{aligned} \quad (3.15)$$

Here the arithmetic average of the  $V$ -velocity  $\bar{V}$  is approximated in the same way as in the cross advectonal term, see (3.4).

### 3.4 Discretization of the continuity equation

We want to discretize the continuity equation of the 2D shallow water equations (2.23):

$$\frac{\partial \zeta}{\partial t} + \frac{\partial}{\partial x}(HU) + \frac{\partial}{\partial y}(HV) = 0. \quad (3.16)$$

We will use the finite volume method for the discretization of this equation. Note that this discretization is equal to the discretization that would be found with the finite difference method, though if a non-uniform grid (like the orthogonal curvilinear grid discussed in Section 2.4) would be used, the discretizations would be different.

We apply the finite volume method on the continuity equation (3.16), with a



uniform Cartesian grid:

$$\begin{aligned}
\int_{\Omega} \left( \frac{\partial \zeta}{\partial t} + \frac{\partial}{\partial x}(HU) + \frac{\partial}{\partial y}(HV) \right) d\Omega &= 0, \\
\text{Gauss' divergence theorem} &\Rightarrow \\
\int_{\Omega} \left( \frac{\partial \zeta}{\partial t} \right) d\Omega + \int_{\Gamma} \begin{pmatrix} HU \\ HV \end{pmatrix} \cdot \mathbf{n} d\Gamma &= 0, \\
\text{midpoint rule} &\Rightarrow \\
\Delta x \Delta y \frac{\partial \zeta_{m,n}}{\partial t} + \Delta y (HU)_{m,n} - \Delta y (HU)_{m-1,n} \\
+ \Delta x (HV)_{m,n} - \Delta x (HV)_{m,n-1} &= 0, \\
&\Rightarrow \\
\frac{\partial \zeta_{m,n}}{\partial t} + \frac{(HU)_{m,n} - (HU)_{m-1,n}}{\Delta x} + \frac{(HV)_{m,n} - (HV)_{m,n-1}}{\Delta y} &= 0, \quad (3.17)
\end{aligned}$$

with  $\Omega$  the computational cell (integration area),  
 $\Gamma$  the boundary of  $\Omega$  and  $\mathbf{n}$  the outward normal of  $\Gamma$ .

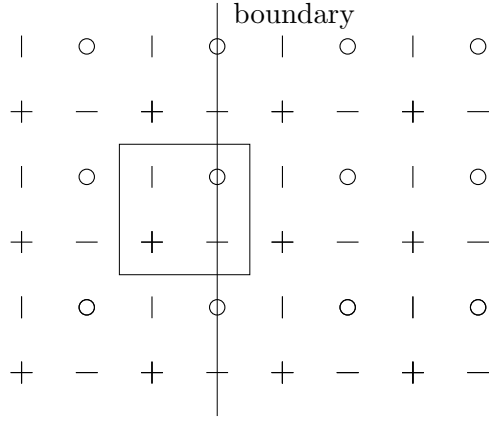
The total water depth  $H = d + \zeta$  needs to be determined at the  $U$ - and  $V$ -velocity points. The depth  $d$  is determined by the arithmetic average of the depth at the cell corners, similarly to the approximation of  $\bar{V}$  in (3.4). The water level  $\zeta$  for the total depth in a  $U$ -velocity point  $H^U$  can be determined in two ways:

$$\zeta_{m,n}^{\text{av}} \approx 0.5(\zeta_{m,n} + \zeta_{m+1,n}) \text{ or,} \quad (3.18)$$

$$\zeta_{m,n}^{\text{up}} \approx \begin{cases} \zeta_{m,n} & \text{if } U_{m,n} > 0 \\ \zeta_{m+1,n} & \text{if } U_{m,n} < 0 \\ \max(\zeta_{m,n}, \zeta_{m+1,n}) & \text{if } U_{m,n} = 0 \end{cases} \quad (3.19)$$

In (3.18) the depth is approximated by averaging over the velocity points, in (3.19) it is approximated by taking the water level in the upwind direction.

The latter approach for  $\zeta$  is a little more complicated but it is physically more realistic and performs better around low water in shallow areas in the neighbourhood of a deep channel or at high water near a summer dyke. In WAQUA there is some check which prescribes which approach should be used. QuickFlow uses this same check.

Figure 3.2: Boundary normal to  $U$ -velocity direction.

This results in the following discretization for  $H^U$ :

$$(H^U)_{m,n}^{\text{av}} \approx \frac{d_{m,n-1} + d_{m,n}}{2} + \frac{\zeta_{m,n} + \zeta_{m+1,n}}{2} \text{ or,} \quad (3.20)$$

$$(H^U)_{m,n}^{\text{up}} \approx \begin{cases} \frac{d_{m,n-1} + d_{m,n}}{2} + \zeta_{m,n} & \text{if } U_{m,n} > 0, \\ \frac{d_{m,n-1} + d_{m,n}}{2} + \zeta_{m+1,n} & \text{if } U_{m,n} < 0, \\ \frac{d_{m,n-1} + d_{m,n}}{2} + \max(\zeta_{m,n}, \zeta_{m+1,n}) & \text{if } U_{m,n} = 0. \end{cases} \quad (3.21)$$

Both approximations for  $H^U$  use the same stencil, which is shown in Figure B.9. For the  $V$ -velocity we use the same approach.

### 3.5 Lateral boundary conditions

In this section lateral boundary conditions are discussed. We already discussed the bottom and surface boundary conditions in Section 2.2.1. The lateral boundary conditions deal with the boundaries of the 2D-approximation, i.e. inflow and outflow boundaries and the shores and banks of a river. The theory in this section is mostly taken from [2, Section 6.3 - 6.5].

In the following the boundary is taken parallel to the  $V$ -velocity direction and normal to the  $U$ -velocity direction. The index of a boundary point is taken  $(m_f, n)$ , see Figure 3.2.

In general if a value is unknown because of the boundary, we will set it to zero. Therefore the order of discretization near a boundary usually is lower than

the discretization of an interior point. This means the solution might be less accurate. However it is found that this does not greatly influence the accuracy at the interior points [4, Section 4.3 - 4.4].

Lateral boundaries can be split into closed boundaries between water and land (Section 3.5.1) and non-physical open boundaries (Sections 3.5.2-3.5.8). Furthermore, we have moving boundaries: water-land boundaries for which the location depends on the water level (Section 3.5.9).

### 3.5.1 Closed boundaries

A closed boundary is located between water and land, it is for example a shore or quay. If the boundary is normal to the  $U$ -velocity direction the boundary conditions for this boundary are:

$$\begin{aligned}
 U_{m_f,n} &= 0 && \text{no flow through boundary,} \\
 \nu_H \frac{\partial V}{\partial x} \Big|_{m_f,n} &= 0 && \text{free slip along the boundary,} \\
 \frac{\partial \zeta}{\partial x} \Big|_{m_f,n} &= 0 && \text{water level at land is set equal to water level at boundary,} \\
 &&& \text{needed for well-posed problem.}
 \end{aligned} \tag{3.22}$$

In  $(m_f, n)$ ,  $U$  is prescribed and the  $V$ -velocity point is outside the domain. For the water level we find:

$$\begin{aligned}
 \frac{\zeta_{m_f+1,n} - \zeta_{m_f,n}}{\Delta x} &= 0 \Rightarrow \\
 \zeta_{m_f,n} &= \zeta_{m_f+1,n}.
 \end{aligned} \tag{3.23}$$

this is used in the pressure gradient and bottom friction terms and in the continuity equation.

We have to solve the  $U$ -momentum equation, the  $V$ -momentum equation and the continuity equation in  $(m_f + 1, n)$ .

Some terms in the shallow water equations need to be adapted if we want to solve these equations on  $(m_f + 1, n)$ . These adaptations are discussed below. For all other terms the discretization is straightforward, as described in Section 3.3 and 3.4.

***U*-momentum: Advection term**

Near the boundary the discretization of the advection term (3.3) needs to be adapted. We will use upwind discretization now instead of central discretization.

$$U \frac{\partial U}{\partial x} \Big|_{m_f+1,n} \approx \begin{cases} U_{m_f+1,n} \frac{U_{m_f+2,n} - U_{m_f+1,n}}{\Delta x} & \text{if } U_{m_f+1,n} \leq 0, \\ 0 & \text{if } U_{m_f+1,n} > 0. \end{cases} \quad (3.24)$$

***V*-momentum: Cross advection term**

The discretization of the cross advection term  $U \frac{\partial V}{\partial x}$  involves many gridpoints, as can be seen in Figures B.4 and B.5. If one or more of these gridpoints is on or outside the boundary, equation (3.5) and thus equation (3.6) and/or (3.7) should be adjusted for this situation. The computational stencil is then reduced by lowering the order of discretization.

For the cross advection term in the *V*-momentum equation we find:

$$U \frac{\partial V}{\partial x} \Big|_{m_f+1,n} = \bar{U}_{m_f+1,n} \frac{V_{m_f+2,n} - V_{m_f,n}}{2\Delta x}. \quad (3.25)$$

***V*-momentum: Viscous term**

In the discretization of  $\frac{\partial^2 V}{\partial x^2}$  in the viscous term first order derivatives are set to zero if they refer to points outside the boundary:

$$\begin{aligned} \frac{\partial^2 V}{\partial x^2} \Big|_{m_f+1,n} &\approx \frac{1}{\Delta x} \left( \frac{\partial V}{\partial x} \Big|_{m_f+1\frac{1}{2},n} - \underbrace{\frac{\partial V}{\partial x} \Big|_{m_f+\frac{1}{2},n}}_{=0} \right) \\ &\approx \frac{1}{(\Delta x)^2} (V_{m_f+2,n} - V_{m_f+1,n}). \end{aligned} \quad (3.26)$$

Note that we do not need to adapt the discretization of the viscous term  $\frac{\partial^2 U}{\partial x^2} \Big|_{m_f+1,n}$  in the *U*-momentum equation, because all points in (3.10) are on the boundary or inside the domain.

**3.5.2 Open boundaries**

The next sections deal with boundary conditions at open boundaries. These water-water boundaries are artificial and exist only in the mathematical model.

In general open boundaries can be located upstream or downstream the river. We can apply numerous boundary conditions. Here we only discuss:

- water level boundary,
- velocity boundary,
- discharge boundary,
- QH-boundary (relation between discharge ( $Q$ ) and depth ( $H$ )), and
- Riemann invariant boundary.

These boundary conditions are the driving forces of the system.

Furthermore, we can apply a reflection coefficient on water level and on velocity boundaries. The reflection coefficient ensures that waves do not reflect at the boundary but disappear from the domain. This will be discussed in Section 3.5.8.

We consider the situation in Figure 3.2, with the boundary normal to the  $U$ -velocity direction.

### 3.5.3 Water level boundary

A water level boundary is usually applied downstream, especially in situations where tide plays a role (i.e. close to the sea). Here we prescribe the water level:

$$\zeta = F_\zeta(t). \quad (3.27)$$

We further need:

$$\begin{aligned} \nu_H \frac{\partial V}{\partial x} \Big|_{m_f, n} &= 0 \quad \text{free slip along the boundary,} \\ V_{m_f, n} &= 0 \quad \text{only needed at inflow boundary.} \end{aligned} \quad (3.28)$$

We want to solve the  $U$ -momentum equation, and the continuity equation in  $(m_f, n)$  and  $(m_f + 1, n)$ . Some terms in the shallow water equations need to be adapted if we want to solve these equations on  $(m_f, n)$  and  $(m_f + 1, n)$ . These adaptations are discussed below. For all other terms the discretization is straightforward, as described in Section 3.3 and 3.4.

### Advection term

In the  $U$ -momentum equation the discretization of the advection term (3.3) needs to be adjusted. We use upwind discretization instead of central:

$$U \frac{\partial U}{\partial x} \Big|_{m_f, n} \approx \begin{cases} U_{m_f, n} \frac{U_{m_f+1, n} - U_{m_f, n}}{\Delta x} & \text{if } U_{m_f, n} \leq 0, \\ 0 & \text{if } U_{m_f, n} > 0, \end{cases} \quad (3.29)$$

$$U \frac{\partial U}{\partial x} \Big|_{m_f+1, n} \approx \begin{cases} U_{m_f+1, n} \frac{U_{m_f+2, n} - U_{m_f+1, n}}{\Delta x} & \text{if } U_{m_f+1, n} \leq 0, \\ U_{m_f+1, n} \frac{U_{m_f+1, n} - U_{m_f, n}}{\Delta x} & \text{if } U_{m_f+1, n} > 0. \end{cases} \quad (3.30)$$

### Pressure gradient in $U$ momentum equation

The pressure gradient is neglected:

$$g \frac{\partial \zeta}{\partial x} \Big|_{m_f, n} \approx 0. \quad (3.31)$$

### Viscous terms

The horizontal viscous terms are simplified by neglecting the second derivatives with respect to  $x$ :  $\frac{\partial^2 U}{\partial x^2}$  and  $\frac{\partial^2 V}{\partial x^2}$ :

$$\begin{aligned} \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \Big|_{m_f, n} &\approx \frac{\partial^2 U}{\partial y^2} \Big|_{m_f, n} \\ &\approx \frac{1}{\Delta y^2} (U_{m_f, n+1} - 2U_{m_f, n} + U_{m_f, n-1}), \\ \frac{\partial^2 V}{\partial x^2} + \frac{\partial^2 V}{\partial y^2} \Big|_{m_f, n} &\approx \frac{\partial^2 V}{\partial y^2} \Big|_{m_f, n} \\ &\approx \frac{1}{\Delta y^2} (V_{m_f, n+1} - 2V_{m_f, n} + V_{m_f, n-1}). \end{aligned} \quad (3.32)$$

#### 3.5.4 QH-boundary

A QH-boundary is usually applied downstream. A discharge is prescribed and related to a water level. The relation between water level and discharge is determined empirically and can be taken from a table.

In fact we apply a water level boundary indirectly:

$$Q = F_Q(t) \stackrel{\text{table}}{\Rightarrow} \zeta \quad (3.33)$$

The water level boundary condition that we find in this way is treated as described in Section 3.5.3.

### 3.5.5 Velocity boundary

A velocity boundary is not often used in practice. It can be applied for example when models are nested. E.g. when we know the velocities in a certain model from simulations, we can use these velocities as boundary conditions if we want to do a new simulation on a smaller part of this model. If the boundary is normal to the  $U$ -velocity direction we prescribe the  $U$ -velocity:

$$U = F_U(t) \quad (3.34)$$

We further prescribe:

$$\begin{aligned} \nu_H \frac{\partial V}{\partial x} \Big|_{m_f, n} &= 0 && \text{free slip along the boundary,} \\ \zeta_{m_f, n} &= \zeta_{m_f+1, n} && \text{needed for well-posedness,} \\ V_{m_f, n} &= 0 && \text{only needed at inflow boundary.} \end{aligned} \quad (3.35)$$

Furthermore, we can apply a reflection coefficient which is discussed in Section 3.5.8.

We want to solve the  $U$ - and  $V$ -momentum equation, and the continuity equations in  $(m_f + 1, n)$ ,  $V_{m_f, n}$ . Some terms in the shallow water equations need to be adapted if we want to solve these equations on  $(m_f, n)$  and  $(m_f + 1, n)$ . These adaptations are discussed below. For all other terms the discretization is straightforward, as described in Section 3.3 and 3.4.

#### Advection term in $U$ -momentum

The discretization of the advection term of the  $U$ -momentum equation (3.3) near the boundary is replaced by the upwind approximation given by (3.30).

#### Viscous term in $U$ -momentum

In the discretization of the viscous term (3.10) at the boundary we need  $\frac{\partial U}{\partial x} \Big|_{m_f - \frac{1}{2}, n}$ . Since it is not available this term is set to zero, just as in (3.26) and thus:

$$\frac{\partial^2 U}{\partial x^2} \Big|_{m_f, n} \approx \frac{1}{(\Delta x)^2} (U_{m_f+1, n} - U_{m_f, n}) \quad (3.36)$$

### 3.5.6 Discharge boundary

A discharge boundary is usually applied upstream. The discharge (or transport rate)  $Q$  is prescribed for the opening.

$$Q = F_Q(t), \quad (3.37)$$

with for every grid cell  $Q = H\Delta y U$ ,

The distribution of the discharge is then computed and relates the total discharge to the velocity. The distribution can be implemented in 2 ways:

- linear distribution over sections of the opening, the discharge is prescribed for the end points of the section (see Figure 3.3); or
- distribution over the opening where the total water depth and the bottom friction are taken into account (see Figure 3.4).

This last case is also called a 'distributed discharge opening'. The lower the total water depth and the higher the bottom friction, the less the discharge over a certain grid cell.

The implementation of this boundary condition is easy for the continuity equation:

$$(HU)_{m_f,n}(t) = \frac{F_Q(t)}{\Delta y}. \quad (3.38)$$

In the momentum equations we have two unknowns (velocity and water level). Therefore we take the total water depth from the previous step:

$$U_{m_f,n}(t + \frac{1}{2}\Delta t) = \frac{F_Q(t + \frac{1}{2}\Delta t)}{H_{m_f,n}(t)\Delta y}. \quad (3.39)$$

### 3.5.7 Riemann invariant boundary

At a Riemann invariant boundary a certain combination of the water level and the velocity is prescribed. This boundary is only applied in special cases for example to study the resonance of waves in a harbor. A Riemann invariant boundary condition can only be applied when  $\zeta$  is small compared to the depth  $d$ :  $\frac{|\zeta|}{d} \ll 1$ . For a boundary normal to the  $U$ -velocity we have:

$$F_R = \begin{cases} U + \zeta\sqrt{\frac{g}{d}} & \text{at left boundary,} \\ U - \zeta\sqrt{\frac{g}{d}} & \text{at right boundary.} \end{cases} \quad (3.40)$$



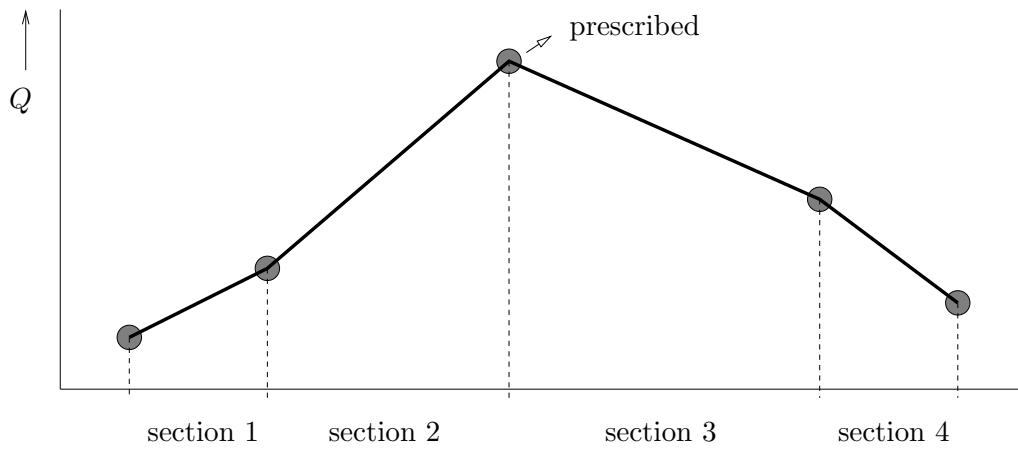


Figure 3.3: Discharge  $Q$  linearly distributed over opening.

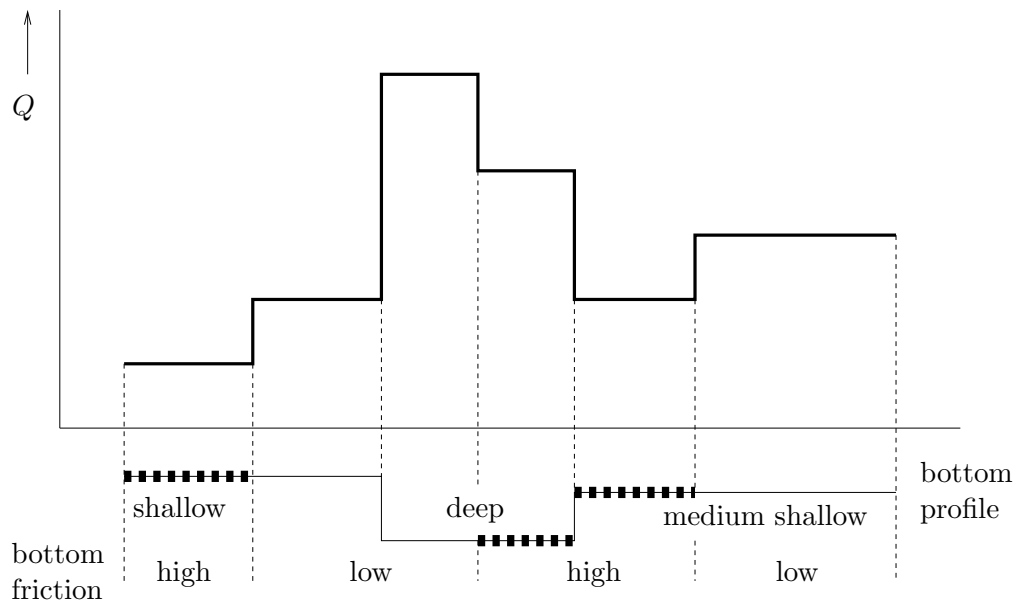


Figure 3.4: Discharge  $Q$  depends on total water depth and on bottom friction.

### 3.5.8 Reflection coefficient

A reflection coefficient is applied to prevent reflection of waves at open boundaries. It is an artefact used in the modelling, and prevents non-physical oscillations. These oscillations are very common in the first time period after the initial solution. The bottom friction and viscosity will ensure some damping, but usually this would take too long. Therefore, a term consisting of the time derivative of the Riemann invariant multiplied by the reflection coefficient  $\alpha$  can be added to a water level or velocity boundary condition to reduce the spin up time of a model:

$$\begin{aligned} F_\zeta(t) &= \zeta + \alpha \frac{\partial}{\partial t} \begin{cases} (U + 2\sqrt{gH}) & \text{at left boundary,} \\ (U - 2\sqrt{gH}) & \text{at right boundary,} \end{cases} \\ F_U(t) &= U + \alpha \frac{\partial}{\partial t} \begin{cases} (U + 2\sqrt{gH}) & \text{at left boundary,} \\ (U - 2\sqrt{gH}) & \text{at right boundary.} \end{cases} \end{aligned} \quad (3.41)$$

The reflection coefficient  $\alpha$  is chosen small enough to damp short oscillations. More details on the reflection coefficient can be found in [2, Section 6.3].

This method is implemented in WAQUA, in QuickFlow we do not use such methods to reduce oscillations.

### 3.5.9 Drying and flooding

The location of a moving boundary can shift due to drying and flooding. The location depends on the water level, see also Figure 3.5. The movement of the front is in the same direction as the flow:

$$\begin{aligned} \mathbf{U}_{\text{front}} &= \mathbf{U} \cdot \mathbf{n} \\ &\text{with } \mathbf{U} = \begin{pmatrix} U \\ V \end{pmatrix} \end{aligned} \quad (3.42)$$

This is illustrated in Figure 3.7.

The theory below is taken from [2, Section 8.1].

In Figure 3.6 the discretized bottom is shown. Now, if the water is very shallow, the influence of the bottom is relatively large and the horizontal velocity will be small. Therefore, we define a threshold value  $\delta$  and describe the drying and

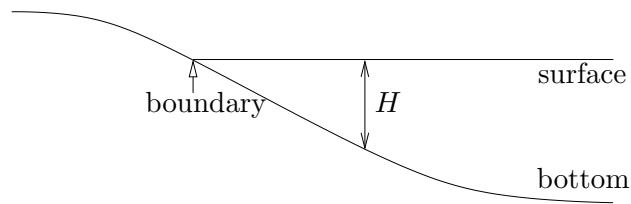


Figure 3.5: A moving boundary (cross sectional view), its location depends on the water level  $H$ .

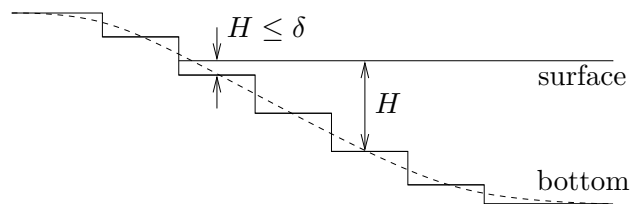


Figure 3.6: Discretization of the bottom with a moving boundary in Figure 3.5.

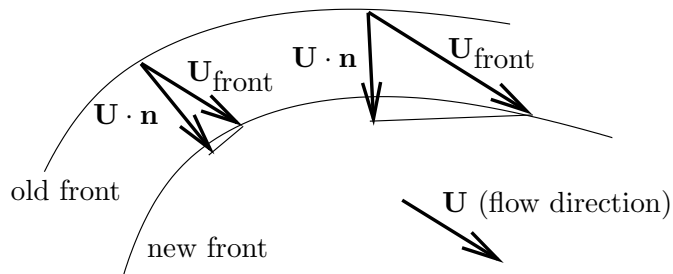


Figure 3.7: A moving boundary (top view), it moves in the same direction as the flow.

flooding procedure:

$$\begin{aligned}
 \text{drying: } U_{m_f,n} &= 0 & \text{if } H_{m_f,n}^U &\leq \frac{1}{2}\delta, \\
 V_{m_f,n} &= 0 & \text{if } H_{m_f,n}^V &\leq \frac{1}{2}\delta, \\
 \text{flooding: } U_{m_f,n} &\neq 0 & \text{if } H_{m_f,n}^U &\leq \delta, \\
 V_{m_f,n} &\neq 0 & \text{if } H_{m_f,n}^V &\leq \delta,
 \end{aligned} \tag{3.43}$$

with  $H_{m,n}^U$  the total depth at the  $U$ -point  $(m,n)$ , see (3.20) and (3.21),  
and  $H_{m,n}^V$  the total depth at the  $V$ -point  $(m,n)$ .

Note that drying only takes place when the water level is below half of the threshold value. It will become wet again only when the water level is above the threshold value  $\delta$ . This way we prevent too fast successive drying and flooding, so called 'flip-flop'.

This method basically comes down to placing and removing of 'screens' whenever necessary. By 'placing a screen' we mean that in the model we place a 'wall' or a 'screen' in such a way that the water can not flow in a certain direction. If we place a screen in the  $U$ -velocity direction the  $V$ -velocity is set to zero. 'Screens' can also be used to assure that at the open boundaries the velocity is normal to the boundary, see Figure 3.8.

This method is easy to implement and the procedure described above to prevent 'flip-flopping' reduces some instabilities. Still it can be unstable, especially when  $\delta$  is chosen small. In [2, Section 8.2] and in [5, Section 3.6.2] more advanced methods are described to prevent instabilities. These methods can be used in WAQUA.

In QuickFlow the placing and removing of screens is implemented approximately. It is assumed that this has not much influence on the solution because we are only interested in the stationary case. Furthermore, we expect the solution to be less accurate in these areas anyhow.

An alternative possible approach to simulate drying and flooding of tidal flats has been designed by VORtech. The shallow water equations have a lot in common with the compressible Navier-Stokes equations. Therefore methods that were developed for free surfaces in the Navier-Stokes equations can be applied on tidal flats in the shallow water equations. This resulted in a method based on artificial porosity. We do not need to use screens with this approach, which is one of its major advantages. This method is (shortly) described in [6].

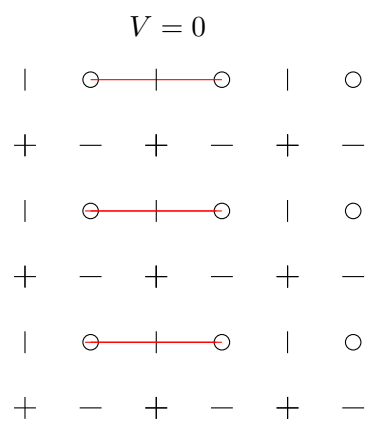


Figure 3.8: Screens (red lines) prevent water to flow in the direction normal to the screen.

# Chapter 4

## Solution methods

After discretization in space and implementation of the boundary conditions, the system of equations is solved using an iterative method. QuickFlow is intended to find the stationary solution, still we use time stepping and a method based on ADI is implemented. We will first introduce the ADI method and explain how it is used in WAQUA. Afterwards we will discuss the time iteration methods used in QuickFlow.

It is important to understand the time iteration in both WAQUA as in QuickFlow because it appears to be one of the major causes for differences, between the two.

### 4.1 Solution Methods in WAQUA

#### 4.1.1 ADI

For time-discretization the Alternating Direction Implicit (ADI)-method is used. This method was first introduced for the shallow water equations in 1967 by Leendertse [7]. The theory below is mostly taken from [2, Chapter 5 and 7].

The ADI-method splits every time step into two stages. In both stages the equations are solved. In the first stage some terms are taken implicitly while the other terms are taken explicitly. In the second stage this is the other way around. Which terms are taken implicitly and which explicitly depends on the application.

This method is used because it is a good combination of an implicit and an explicit method. A completely explicit method would only be stable for a very small time step. On the other hand, an implicit method would use too much computer time and memory, which was particularly important in the early days of WAQUA in the 1970's.

The ADI-method is based on the Crank-Nicholson method for solving the linear matrix vector equation  $\frac{d\mathbf{u}}{dt} = \mathbf{P}\mathbf{u}$ :

$$\frac{\mathbf{u}^{l+1} - \mathbf{u}^l}{\Delta t} = \frac{1}{2}\mathbf{P}\mathbf{u}^l + \frac{1}{2}\mathbf{P}\mathbf{u}^{l+1}. \quad (4.1)$$

The ADI-method can be written as:

$$\begin{aligned} \frac{\mathbf{u}^{l+1/2} - \mathbf{u}^l}{\Delta t} &= \frac{1}{2}\mathbf{P}_1\mathbf{u}^{l+1/2} + \frac{1}{2}\mathbf{P}_2\mathbf{u}^l && \text{stage 1,} \\ \frac{\mathbf{u}^{l+1} - \mathbf{u}^{l+1/2}}{\Delta t} &= \frac{1}{2}\mathbf{P}_1\mathbf{u}^{l+1/2} + \frac{1}{2}\mathbf{P}_2\mathbf{u}^{l+1} && \text{stage 2,} \end{aligned} \quad (4.2)$$

with  $\mathbf{u} = \begin{pmatrix} U \\ V \\ \zeta \end{pmatrix}$  and functions  $\mathbf{A}_1$  and  $\mathbf{A}_2$ .

In this application  $\mathbf{P}_1$  contains all derivatives in the  $x$ -direction,  $\mathbf{P}_2$  contains the derivatives in the  $y$ -direction. Furthermore  $\mathbf{P}_1 + \mathbf{P}_2 = \mathbf{P}$ .

The equations for the ADI method (4.2) can also be written as:

$$\begin{aligned} \frac{d\mathbf{u}}{dt} &= \mathbf{A}_1(\mathbf{u}^l, \mathbf{u}^{l+1/2}) && \text{stage 1,} \\ \frac{d\mathbf{u}}{dt} &= \mathbf{A}_2(\mathbf{u}^{l+1/2}, \mathbf{u}^{l+1}) && \text{stage 2.} \end{aligned} \quad (4.3)$$

In Section 4.1.2 we will discuss how  $\mathbf{A}_1$  and  $\mathbf{A}_2$  look like for the shallow water equations.

With  $\mathbf{u}(t) = \mathbf{u}^l$ ,  $\mathbf{u}(t + \frac{1}{2}\Delta t) = \mathbf{u}^{l+1/2}$  and  $\mathbf{u}(t + \Delta t) = \mathbf{u}^{l+1}$  we can also write for the first stage:

$$\mathbf{u}\left(t + \frac{1}{2}\Delta t\right) = \mathbf{u}(t) + \frac{1}{2}\Delta t\mathbf{A}_1\left(\mathbf{u}(t), \mathbf{u}\left(t + \frac{1}{2}\Delta t\right)\right). \quad (4.4)$$

For the equations in the second stage we can similarly write:

$$\mathbf{u}(t + \Delta t) = \mathbf{u}\left(t + \frac{1}{2}\Delta t\right) + \frac{1}{2}\Delta t\mathbf{A}_2\left(\mathbf{u}\left(t + \frac{1}{2}\Delta t\right), \mathbf{u}(t + \Delta t)\right). \quad (4.5)$$

By substituting (4.4) in (4.5) we find one equation for the ADI method:

$$\mathbf{u}(t + \Delta t) = \mathbf{u}(t) + \frac{1}{2}\Delta t \left[ \begin{array}{l} \mathbf{A}_1(\mathbf{u}(t), \mathbf{u}(t + \frac{1}{2}\Delta t)) \\ + \mathbf{A}_2(\mathbf{u}(t + \frac{1}{2}\Delta t), \mathbf{u}(t + \Delta t)) \end{array} \right]. \quad (4.6)$$

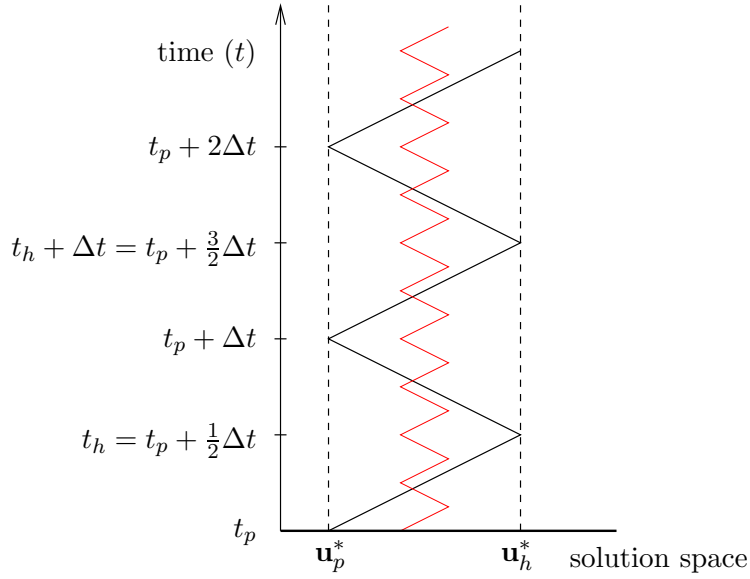


Figure 4.1: The stationary solution we find with the ADI method used in WAQUA exists of 2 solutions:  $\mathbf{u}_p^*$  at primary time steps and  $\mathbf{u}_h^*$  at half time steps. The red line gives an indication of the 'stationary' solution if we take a smaller time step  $\Delta t$ .

If we now let  $\Delta t \rightarrow 0$  then

$$\begin{aligned} \mathbf{u}(t + \frac{1}{2}\Delta t) &\rightarrow \mathbf{u}(t), & \text{and} \\ \mathbf{u}(t + \Delta t) &\rightarrow \mathbf{u}(t + \frac{1}{2}\Delta t), & \text{and thus} \\ \mathbf{u}(t + \Delta t) &\rightarrow \mathbf{u}(t). \end{aligned} \quad (4.7)$$

Note that in practice the time step can never be equal to zero. Therefore we will always find two coupled stationary solutions, one for the primary (integer valued) time steps ( $p$ ) and one for the half time steps ( $h$ ):

$$\begin{aligned} \mathbf{u}_p^* &= \mathbf{u}^*(t) = \mathbf{u}^*(t + \Delta t), \\ \mathbf{u}_h^* &= \mathbf{u}^*(t - \frac{1}{2}\Delta t) = \mathbf{u}^*(t + \frac{1}{2}\Delta t). \end{aligned} \quad (4.8)$$

These solutions are not necessarily close to each other:  $|\mathbf{u}_p^* - \mathbf{u}_h^*|$  might be large, especially for large time steps  $\Delta t$ . This subject is illustrated in Figure 4.1.

Consider the stationary solution for the primary time steps  $\mathbf{u}_p^*$ . If we now take a very small time step ( $\Delta t \rightarrow 0$ ) then the stationary solution for the half time step  $\mathbf{u}_h^* \rightarrow \mathbf{u}_p^*$ .



This way we find the stationary solution in WAQUA  $\mathbf{u}^{*W}$  with:

$$\frac{d\mathbf{u}^{*W}}{dt} = \frac{1}{2} (\mathbf{A}_1(\mathbf{u}^{*W}, \mathbf{u}^{*W}) + \mathbf{A}_2(\mathbf{u}^{*W}, \mathbf{u}^{*W})). \quad (4.9)$$

WAQUA does not store the solution from the half time steps, therefore we will find  $\mathbf{u}^{*W} = \mathbf{u}_p^*$  and we do not know how far this solution is away from the solution in the half time steps, in other words:  $|\mathbf{u}^{*W} - \mathbf{u}_h^*|$  is unknown.

#### 4.1.2 ADI applied to the shallow water equations

We can now apply the ADI method to the shallow water equations (2.23) with its discretization described in Chapter 3. This is described in more detail in [2, Chapter 5]. In the first stage of the ADI method we solve the following equations, in this order:

$$\begin{aligned}
& \frac{V^{l+1/2} - V^l}{\frac{1}{2}\Delta t} + \bar{U}^l \frac{\partial}{\partial x} V^{l+1/2} + V^l \frac{\partial}{\partial y} V^{l+1/2} \\
& + \nu_H \left( \frac{\partial^2}{\partial x^2} V^{l+1/2} + \frac{\partial^2}{\partial y^2} V^{l+1/2} \right) - g V^{l+1/2} \frac{\sqrt{(\bar{U}^l)^2 + (V^l)^2}}{\bar{H}^l C_{2D,y}^2} \\
& + g \frac{\partial}{\partial y} \zeta^l + f \bar{U}^l = 0, \quad (4.10) \\
& \Rightarrow V^{l+1/2},
\end{aligned}$$

$$\left. \begin{aligned}
& \frac{U^{l+1/2} - U^l}{\frac{1}{2}\Delta t} + U^{l+1/2} \frac{\partial}{\partial x} U^l + \bar{V}^{l+1/2} \frac{\partial}{\partial y} U^l \\
& + \nu_H \left( \frac{\partial^2}{\partial x^2} U^l + \frac{\partial^2}{\partial y^2} U^l \right) - g U^{l+1/2} \frac{\sqrt{(U^l)^2 + (\bar{V}^l)^2}}{\bar{H}^l C_{2D,x}^2} \\
& + g \frac{\partial}{\partial x} \zeta^{l+1/2} - f \bar{V}^{l+1/2} = 0, \\
& \frac{\zeta^{l+1/2} - \zeta^l}{\frac{1}{2}\Delta t} + \frac{\partial}{\partial x} (H^{l+1/2} \bar{U}^{l+1/2}) + \frac{\partial}{\partial y} (H^l \bar{V}^l) = 0, \\
& \Rightarrow U^{l+1/2}, \zeta^{l+1/2}.
\end{aligned} \right\} \quad (4.11)$$

In the second stage the following equations are solved in this order:

$$\begin{aligned}
& \frac{U^{l+1} - U^{l+1/2}}{\frac{1}{2}\Delta t} + U^{l+1/2} \frac{\partial}{\partial x} U^{l+1} + \bar{V}^{l+1/2} \frac{\partial}{\partial y} U^{l+1} \\
& + \nu_H \left( \frac{\partial^2}{\partial x^2} U^{l+1} + \frac{\partial^2}{\partial y^2} U^{l+1} \right) - g U^{l+1} \frac{\sqrt{(U^{l+1/2})^2 + (\bar{V}^{l+1/2})^2}}{\bar{H}^{l+1/2} C_{2D,x}^2} \\
& + g \frac{\partial}{\partial x} \zeta^{l+1/2} - f \bar{V}^{l+1/2} = 0, \quad (4.12) \\
& \Rightarrow U^{l+1},
\end{aligned}$$

$$\left. \begin{aligned}
& \frac{V^{l+1} - V^{l+1/2}}{\frac{1}{2}\Delta t} + \bar{U}^{l+1} \frac{\partial}{\partial x} V^{l+1/2} + V^{l+1} \frac{\partial}{\partial y} V^{l+1/2} \\
& + \nu_H \left( \frac{\partial^2}{\partial x^2} V^{l+1/2} + \frac{\partial^2}{\partial y^2} V^{l+1/2} \right) - g V^{l+1} \frac{\sqrt{(\bar{U}^{l+1/2})^2 + (V^{l+1/2})^2}}{\bar{H}^{l+1/2} C_{2D,y}^2} \\
& + g \frac{\partial}{\partial y} \zeta^{l+1} + f \bar{U}^{l+1} = 0, \\
& \frac{\zeta^{l+1} - \zeta^{l+1/2}}{\frac{1}{2}\Delta t} + \frac{\partial}{\partial x} (H^{l+1/2} \bar{U}^{l+1/2}) + \frac{\partial}{\partial y} (H^{l+1} \bar{V}^{l+1}) = 0, \\
& \Rightarrow V^{l+1}, \zeta^{l+1}.
\end{aligned} \right\} \quad (4.13)$$

Now we can define  $A_1$  and  $A_2$  from (4.3). We can write for the first stage:

$$A_1 \begin{pmatrix} U^l, & U^{l+1/2} \\ V^l, & V^{l+1/2} \\ \zeta^l, & \zeta^{l+1/2} \end{pmatrix} = \begin{pmatrix} a_1 & b_1 & c_1 \\ 0 & b_2 & 0 \\ 0 & 0 & 0 \end{pmatrix} \begin{pmatrix} U^{l+1/2} \\ V^{l+1/2} \\ \zeta^{l+1/2} \end{pmatrix} + \begin{pmatrix} d_1 \\ d_2 \\ d_3 \end{pmatrix} + \begin{pmatrix} 0 \\ f_2(V^{l+1/2}) \\ f_3(U^{l+1/2}, \zeta^{l+1/2}) \end{pmatrix} \quad (4.14)$$

with:

$$\begin{aligned} a_1 &= \frac{\partial U^l}{\partial x} - g \frac{\sqrt{(U^l)^2 + (V^l)^2}}{\bar{H}^l C_{2D,x}^2} \\ b_1 &= \frac{\partial U^l}{\partial y} - f \\ c_1 &= g \frac{\partial}{\partial x} \\ d_1 &= \nu_H \left( \frac{\partial^2}{\partial x^2} U^l + \frac{\partial^2}{\partial y^2} U^l \right) \\ b_2 &= \bar{U}^l \frac{\partial}{\partial x} + V^l \frac{\partial}{\partial y} - g \frac{\sqrt{(\bar{U}^l)^2 + (V^l)^2}}{\bar{H}^l C_{2D,y}^2} \\ d_2 &= g \frac{\partial \zeta^l}{\partial y} + f \bar{U}^l \\ f_2(V^{l+1/2}) &= \nu_H \left( \frac{\partial^2}{\partial x^2} V^{l+1/2} + \frac{\partial^2}{\partial y^2} V^{l+1/2} \right) \\ d_3 &= \frac{\partial}{\partial y} (H^l \bar{U}^l) \\ f_3(U^{l+1/2}, \zeta^{l+1/2}) &= \frac{\partial}{\partial x} (H^{l+1/2} \bar{U}^{l+1/2}) \end{aligned}$$

$A_2$  can be defined similarly. Note that  $A_1$  consists of a linear and a nonlinear part. The submatrices  $a_1$ ,  $b_1$ ,  $c_1$  and  $b_2$  are pentadiagonal matrices, as we will show in the next section.

### Solving nonlinear equations

Below we describe how to solve the  $U$ -momentum equations in the second stage (4.12) and the  $U$ -momentum equation in combination with the continuity equa-

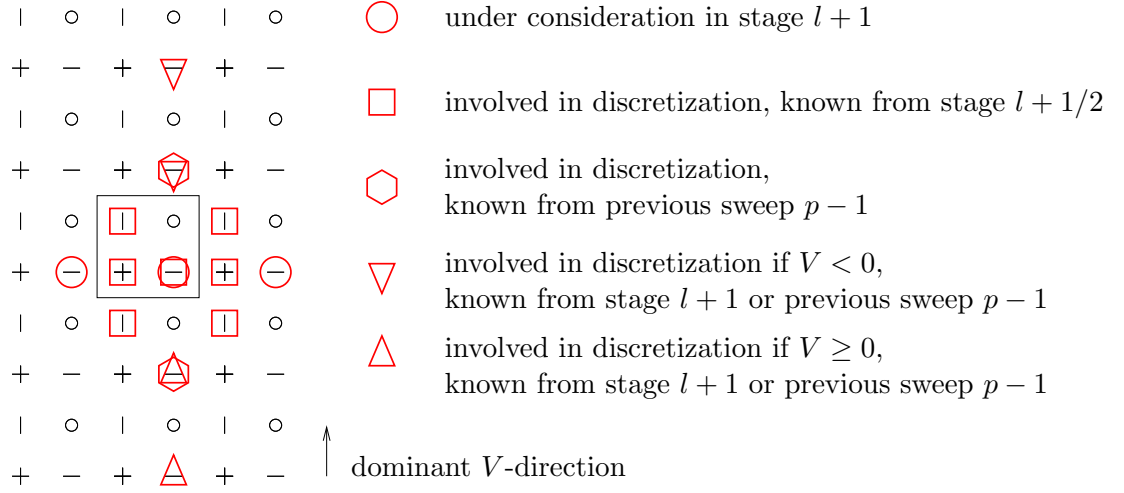


Figure 4.2: Stencil for time discretization of  $U$ -momentum equation in the second stage of ADI method.

tion from the first stage (4.11). The solutions of the  $V$ -momentum and continuity equations (4.10) and (4.13) can be obtained using similar techniques.

We start with the discretization of the  $U$ -momentum equation in second stage of ADI method (4.12). The stencil is shown in Figure 4.2. We can not solve this equation directly, we will use 'sweeps'. The first sweep  $p = 1$  is in the dominant  $V$ -direction. The dominant  $V$ -direction is determined by:

$$\text{dominant } V\text{-direction} = \begin{cases} y\text{-direction} & \text{if } \sum_{m,n} V_{m,n}^{l+1/2} \geq 0, \\ -(y\text{-direction}) & \text{if } \sum_{m,n} V_{m,n}^{l+1/2} < 0. \end{cases} \quad (4.15)$$

The next sweep is in the other direction, then again in the dominant  $V$ -direction, etc. Usually 2 sweeps are enough, but this can be adjusted by the user. If the  $V$ -direction for gridpoint  $(m, n)$  is the sweep direction,  $U_{m,n-1}^{l+1}$  and  $U_{m,n-2}^{l+1}$  can be used. If the  $V$ -direction is not the sweep direction  $U_{m,n+1}^{p-1}$  and  $U_{m,n+2}^{p-1}$  from the previous sweep have to be used. This results in tridiagonal matrix equation for each  $n$  in each sweep:

$$a_{m-1,n}U_{m-1,n} + b_{m,n}U_{m,n} + c_{m+1,n}U_{m+1,n} = d_{m,n} \quad (4.16)$$

with  $a_{m,n}$ ,  $b_{m,n}$ ,  $c_{m,n}$ ,  $d_{m,n}$  constants.

In the first stage of the ADI-method the  $U$ -momentum and the continuity equations (4.11) are solved simultaneously. In Figures 4.3 and 4.4 the stencils are

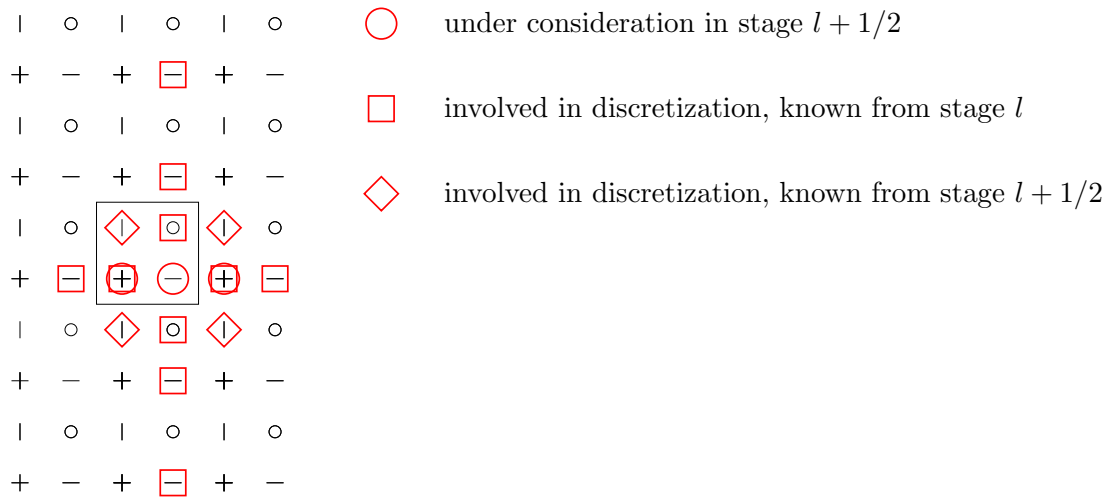


Figure 4.3: Stencil for time discretization of  $U$ -momentum equation in the first stage of ADI method.

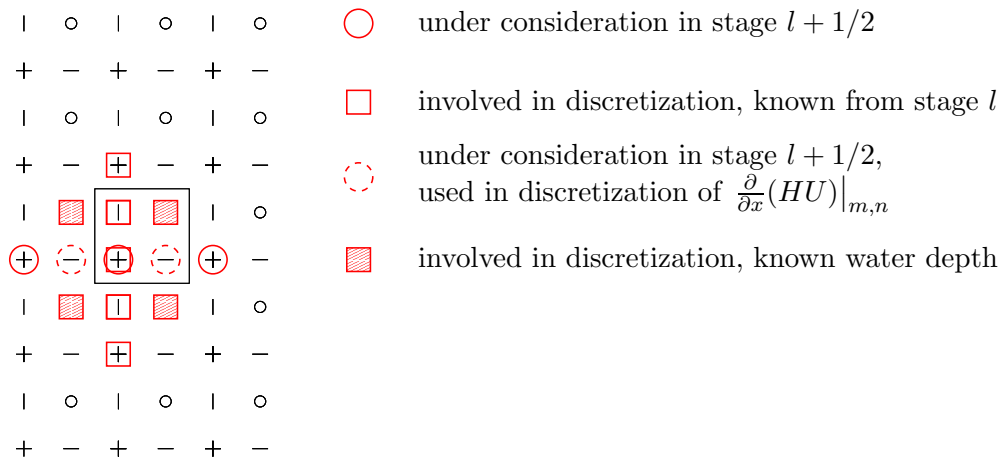


Figure 4.4: Stencil for time discretization of continuity equation in the first stage of ADI method.

shown. The equations can not be solved in a straightforward way. Therefore we first use the  $U$ -momentum equation to express  $U$  in  $\zeta$ :

$$\begin{aligned} a_{m,n}\zeta_{m,n} + b_{m,n}U_{m,n} + c_{m+1,n}\zeta_{m+1,n} &= d_{m,n} \Rightarrow \\ U_{m,n} &= U_{m,n}(\zeta_{m,n}, \zeta_{m+1,n}), \\ \text{with } a_{m,n}, b_{m,n}, c_{m,n}, d_{m,n} &\text{ constants.} \end{aligned} \quad (4.17)$$

Now this  $U$  can be substituted in the continuity equation. This results in a tridiagonal matrix equation for each  $n$  in each sweep:

$$\begin{aligned} e_{m-1,n}\zeta_{m-1,n} + f_{m,n}\zeta_{m,n} + g_{m+1,n}\zeta_{m+1,n} &= h_{m,n} \\ \text{with } e_{m-1,n}, f_{m,n}, g_{m+1,n}, h_{m,n} &\text{ constants.} \end{aligned} \quad (4.18)$$

After solving (4.18) for  $\zeta$  the velocities  $U$  can be obtained by back substitution of  $\zeta$  in (4.17).

The tridiagonal matrix equations (4.16) and (4.18) can be solved using Thomas' algorithm. This algorithm is discussed in Appendix C. It follows there that we have the restriction:

$$\max \left( \frac{\Delta t |U_{m,n}^l|}{\Delta x}, \frac{\Delta t |V_{m,n}^l|}{\Delta y} \right) \leq 4, \quad \forall m, n. \quad (4.19)$$

## 4.2 Solution methods in QuickFlow

QuickFlow only computes the stationary solution of the shallow water equations. Therefore it is not necessary to do all the time steps as accurately as in WAQUA. Still time stepping is used. The initial solution is taken from WAQUA and is assumed to be stationary. First we use QuickFlow to check if this solution is indeed stationary. Then an adjustment is made to, for example, the bottom. QuickFlow now computes the new stationary solution by taking time steps until a stationary solution is found. In this section we will focus on how QuickFlow finds a stationary solution.

Remember from Section 4.1.1 that we can find an approximation of the stationary solution in WAQUA  $\mathbf{u}^{*W}$  by solving equation (4.9):

$$\begin{aligned} \frac{d\mathbf{u}^{*W}}{dt} &= \frac{1}{2} (\mathbf{A}_1(\mathbf{u}^{*W}, \mathbf{u}^{*W}) + \mathbf{A}_2(\mathbf{u}^{*W}, \mathbf{u}^{*W})), \\ \text{with } \mathbf{A}_1 \text{ and } \mathbf{A}_2 &\text{ as in (4.14).} \end{aligned} \quad (4.20)$$

### 4.2.1 Euler Backward

WAQUA uses the ADI-method to solve (4.20). In QuickFlow we solve this equation with the Euler Backward method.

In general  $\frac{dx}{dt} = f(t, x(t))$  is approximated with Euler Backward as follows:

$$\begin{aligned} \frac{dx}{dt} &\approx \frac{x(t) - x(t - \Delta t)}{\Delta t}, \Rightarrow \\ x^{l+1} &= x^l + \Delta t f(t^{l+1}, x^{l+1}). \end{aligned} \quad (4.21)$$

We now apply Euler Backward (4.21) on the differential equation (4.20):

$$\begin{aligned} \mathbf{u}^{n+1} = \mathbf{u}^n + \frac{1}{2}\Delta t \left[ \begin{array}{c} \mathbf{A}_1(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}) \\ + \mathbf{A}_2(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}) \end{array} \right], \end{aligned} \quad (4.22)$$

with  $\mathbf{u}^0 = \mathbf{u}^{*W}$  the initial solution.

We can also write this as:

$$\frac{1}{2}\Delta t M(\mathbf{u}^{n+1} - \mathbf{u}^n) + \mathbf{A}(\mathbf{u}^{n+1}) = \mathbf{b}, \quad (4.23)$$

with mass matrix  $M$  the identity matrix, with some ones (referring to equations without time derivative) replaced by zeros, function  $\mathbf{A}(\mathbf{u}^{n+1}) = -(\mathbf{A}_1(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}) + \mathbf{A}_2(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}))$ , and right hand side vector  $\mathbf{b}$  known from boundary conditions and previous steps.

With  $\mathbf{u}(t) = \mathbf{u}^n$  and  $\mathbf{u}(t + \Delta t) = \mathbf{u}^{n+1}$  we can write (4.23) also as:

$$\frac{1}{2}\Delta t M(\mathbf{u}(t + \Delta t) - \mathbf{u}(t)) + \mathbf{A}(\mathbf{u}(t + \Delta t)) = \mathbf{b}. \quad (4.24)$$

#### Stop criterium outer iteration

We need to set a stop criterium for the Euler Backward method.

For the stationary solution  $\mathbf{u}^{*Q}$  of (4.23) we have:

$$\mathbf{A}(\mathbf{u}^{*Q}) = \mathbf{b}. \quad (4.25)$$

We compute the residual  $R_{\text{outer}}$  in the  $n$ -th time step using the 2-norm:

$$R_{\text{outer}}^n = \|\mathbf{A}(\mathbf{u}^n) - \mathbf{b}\|_2. \quad (4.26)$$

If this residual is small enough the iteration is complete and we have approximated the stationary solution close enough. This stopping criterium is set at:  $R_{\text{outer}}^n < 10^{-7}$ .

Note that there is a close relation between the initial residual and the difference, in the 2-norm, between the initial and the stationary solution. We will make this clear for the linear case  $\mathbf{Ax} = \mathbf{b}$  with stationary solution  $\mathbf{x}^*$ .

$$\begin{aligned} R^n &= \|\mathbf{Ax}^n - \mathbf{b}\| \\ &= \|\mathbf{Ax}^n - \mathbf{Ax}^*\| \\ &\leq \|\mathbf{A}\| \|\mathbf{x}^n - \mathbf{x}^*\|. \end{aligned} \quad (4.27)$$

If we take  $n = 0$ , we consider the initial residual  $R^0$  and the difference between the initial and the stationary solution  $\mathbf{x}^0 - \mathbf{x}^*$ . We find:

$$\begin{aligned} R^0 &\leq \|\mathbf{A}\| \|\mathbf{x}^0 - \mathbf{x}^*\| \Rightarrow \\ \frac{R^0}{\|\mathbf{A}\|} &\leq \|\mathbf{x}^0 - \mathbf{x}^*\|. \end{aligned} \quad (4.28)$$

So, this gives an indication of how far the initial solution is, at least, from the stationary solution.

### Time step

We solve the nonlinear equation (4.24) using Newton's method with line search (see Section 4.2.2). Note that the Euler Backward method is stable even for large time steps. However, Newton's method performs better if we take small time steps. Therefore we adjust the time step  $\Delta t$  every outer iteration. If Newton's method needs only a few (inner) iterations, this is an indication that we can take a larger time step. If it needs a lot of iterations or if the residual does not decrease, then the time step should be taken smaller. We apply the following criterium:

$$\Delta t^{n+1} = \begin{cases} 2\Delta t^n & \text{if } K \leq 2 & \text{and } R_{\text{outer}}^n \leq 1.01R_{\text{outer}}^{n-1} \\ \Delta t^n & \text{if } 2 < K < 5 & \text{and } R_{\text{outer}}^n \leq 1.01R_{\text{outer}}^{n-1} \\ \frac{1}{2}\Delta t^n & \text{if } K < 5 & \text{and } R_{\text{outer}}^n > 1.01R_{\text{outer}}^{n-1} \\ \frac{1}{2}\Delta t^n & \text{if } 5 \leq K < 7 \\ \frac{1}{4}\Delta t^n & \text{if } K \geq 7 \end{cases} \quad (4.29)$$

with  $K$  the number of Newton iterations to find  $\mathbf{u}^n$ ,

and  $R_{\text{outer}}^n$  the residual of the  $n$ -th outer iteration as defined in (4.26).

### 4.2.2 Newton's method with line search

We will solve equation (4.23) using Newton's method, also known as the Newton Raphson method. In general Newton's method approximates the solution of



$\mathbf{F}(\mathbf{x}) = \mathbf{0}$  by:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \left(\mathbf{F}'(\mathbf{x}^k)\right)^{-1} \mathbf{F}(\mathbf{x}^k), \quad (4.30)$$

with  $\mathbf{x}^0$  the initial iterate, and  $\mathbf{F}'$  the Jacobian:

$$\mathbf{F}'(\mathbf{x}) = \begin{pmatrix} \frac{\partial F_1(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial F_1(\mathbf{x})}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial F_n(\mathbf{x})}{\partial x_1} & \dots & \frac{\partial F_n(\mathbf{x})}{\partial x_n} \end{pmatrix}. \quad (4.31)$$

Note that we need to compute the Jacobian  $\mathbf{F}'(\mathbf{x})$  and its inverse in every iteration step. In practice we do not compute  $(\mathbf{F}'(\mathbf{x}))^{-1}$  explicitly. Instead we first find a vector  $\mathbf{s}$  that satisfies  $\mathbf{F}'(\mathbf{x}^k)\mathbf{s} = -\mathbf{F}(\mathbf{x}^k)$ . We will compute  $\mathbf{s}$  using the Matlab routine `s=F'\F`. Then we compute the new approximation using the line search algorithm described below.

We apply Newton's method on the nonlinear equation (4.23). So we are actually solving:

$$\mathbf{F}(\mathbf{x}) = \frac{1}{2}\Delta t M(\mathbf{u}^{n+1} - \mathbf{u}^n) + \mathbf{A}(\mathbf{u}^{n+1}) - \mathbf{b} = \mathbf{0}, \quad (4.32)$$

with  $\mathbf{x} = \mathbf{u}^{n+1}$ .

In order to simplify notation we will use within this inner iteration  $\mathbf{u}^n = \mathbf{x}^0$ , iteration index  $k$ . So Newton's method will give us the following sequence:  $\mathbf{u}^n = \mathbf{x}^0, \mathbf{x}^1, \mathbf{x}^2, \dots$

### Line searches

In QuickFlow we do not use Newton's method exactly as described above in (4.30). We combine it with a line search. This means that we do not necessarily take a complete Newton step  $\mathbf{s}^k = (\mathbf{F}'(\mathbf{x}^k))^{-1} \mathbf{F}(\mathbf{x}^k)$ . The line search method instead adjusts the length of the step in this descent direction  $\mathbf{s}$ . Therefore we minimize over  $\alpha$

$$\phi(\alpha) = \mathbf{F}(\mathbf{x}^k - \alpha \mathbf{s}^k). \quad (4.33)$$

This minimization problem would be too complex to solve exactly. Therefore we start with  $\alpha^0 = 1$  (complete Newton step) and we do an iteration process:

- step 0.  $\alpha^i = \alpha^0$
1.  $\alpha^{i+1} = \frac{1}{2}\alpha^i$
  2. if  $\phi(\alpha^{i+1}) < \phi(\alpha^i)$   
 $i := i + 1$   
return to step 1.
  3.  $\alpha = \alpha^i$

We use the resulting  $\alpha$  to compute the solution in the next iteration:

$$\mathbf{x}^{k+1} = \mathbf{x}^k - \alpha \mathbf{s}^k \quad (4.34)$$

with  $\mathbf{s}^k = \left(\mathbf{F}'(\mathbf{x}^k)\right)^{-1} \mathbf{F}(\mathbf{x}^k)$ .

One of the major advantages of line search methods is that the convergence is global.

### Stop criterium inner iteration

The inner iterations of Newton's method with line search also need a stopping criterium.

We are looking for an approximation of the solution of (4.23). Every  $k$ -th iteration of Newton's method we compute the residual:

$$R_{\text{inner}}^k = \left\| \mathbf{M}(\mathbf{x}^k - \mathbf{x}^0) + \mathbf{A}(\mathbf{x}^k) - \mathbf{b} \right\|_2. \quad (4.35)$$

For the stopping criterium we use the relative residual:

$$\frac{R_{\text{inner}}^k}{R_{\text{inner}}^0} < 0.01. \quad (4.36)$$

# Chapter 5

## Test problems

In this section we will describe the test problems for the next phase of this project. During the coming months we will look at these test problems more closely in order to find out where and why problems in solving the shallow water equations occur. Furthermore, we will try to adjust the solution methods to find a more accurate solution and/or to achieve a faster convergence.

We will consider three test problems. One is theoretical, the other two are models of river parts in the Netherlands. They are chosen such that we can start with an easy problem. Afterwards we convert to a real life problem with little difficulties in geometry and boundary conditions. Finally, we consider a more complex real life problem.

Some typical characteristics of models are shown in Table 5.1.

### 5.1 Theoretical test problem

The first test problem is theoretical: it consists of a fictitious straight canal. We will refer to this problem as the 'Chézy gutter', named after the french hydraulic engineer Antoine de Chézy (1718-1798). It has been developed to test software and is ready to use. The grid of this model is Cartesian, there is a similar test problem with curvilinear grid.

The stationary solution as it is calculated by WAQUA and by QuickFlow at this moment is shown in Figures 5.1 and 5.2. Near the closed boundaries the velocities computed by QuickFlow are much larger than the velocities computed by WAQUA. In Figure 5.3 the residuals of the WAQUA-solution as they are computed by QuickFlow are shown. Note that there are large residuals near the open boundaries.

Table 5.1: Characteristics of test problems.

	Chézy	Lek	Randwijk
<b>Geometry</b>			
length $\times$ width (km)	0.75 $\times$ 0.33	$\pm$ 4.5 $\times$ 0.5	$\pm$ 24 $\times$ 0.5
length $\times$ width (number of water level points = number of cells +1)	26 $\times$ 12	43 $\times$ 35	288 $\times$ 27
gridsize (m $\times$ m)	30 $\times$ 30	$\pm$ 100 $\times$ 15	$\pm$ 85 $\times$ 20
number of weirs	0	156	2326
depth under reference plane ( $d$ (m))		6,...,-6	10,...,-15
total depth ( $H$ (m))	4,...,4.5		
eddy viscosity (m <sup>2</sup> /s)	1	0.5	0.5
<b>Boundary conditions</b>			
bottom: Manning coefficient ( $n$ )	0.0316		
bottom: Nikuradse roughness length ( $k_s$ ) (mostly around 0.2) (needed for White- Colebrook)		0,...,4	0,...,40
treshold value drying procedure (m)	0.3	0.01	0.01
inflow: discharge boundary ( $F_Q$ (m/s <sup>2</sup> ))		-3,...,-400	-100,...,-200
inflow: velocity boundary ( $F_U$ (m/s))	2.93		
outflow: water level boundary ( $F_\zeta$ (m))	0.45	4.35	10.85
<b>WAQUA input</b>			
initial water level ( $\zeta$ (m))	0.75	4.5	14.5
reflection coefficient ( $\alpha$ ) at inflow boundary	100	100	100
reflection coefficient ( $\alpha$ ) at outflow boundary	0	100	100
time step (minutes)	0.5	0.5	0.2
time domain (minutes)	120	1440	2000

## 5.2 Real test problems

The other two test problems correspond to real parts of rivers in the Netherlands. Their location is shown in Figure 5.4.

### 5.2.1 Lek

We consider a part of the river 'Lek' as shown in Figure 5.5. The stationary solution as it is calculated by QuickFlow at this moment is shown in Figure 5.7.

The situation here is simple: there is no flooding and drying and the boundary conditions on the closed boundaries are easy to implement. Furthermore, there are no branches, splittings or sharp curves and only a few weirs. However, we find a relatively large difference between the solutions of QuickFlow and of WAQUA in the first three grid cells from the outflow boundary. The differences are shown in Figure 5.8, the residuals are shown in Figure 5.6. The differences are only in the order of a few promille of the calculated velocities. Since the differences are much smaller in other areas, this indicates that something does go wrong in the computations in this area. Note that in Figure 5.6 the residuals in the continuity equation are much smaller than in the momentum equations. In the momentum equations large residuals can be seen in the first 3 grid cells from the left boundary.

### 5.2.2 Randwijk

We consider a part of the Rhine river just south of the city Wageningen, as shown in Figure 5.9. The model of this part is called after the nearby town 'Randwijk'. The stationary solution as it is calculated by QuickFlow at this moment is shown in Figure 5.11.

The situation is more complex than in the Lek model. The geometry is more complex, there are a lot of weirs and some parts can dry and flood. Several of these complexities can be observed in the more detailed picture of Figure 5.10.

Until recently the solution we found in QuickFlow did not converge. We have found that it does converge if we use an other initial solution, which is probably closer to the stationary solution. Still we find differences in the solutions from WAQUA and from QuickFlow, as is shown in Figure 5.12.

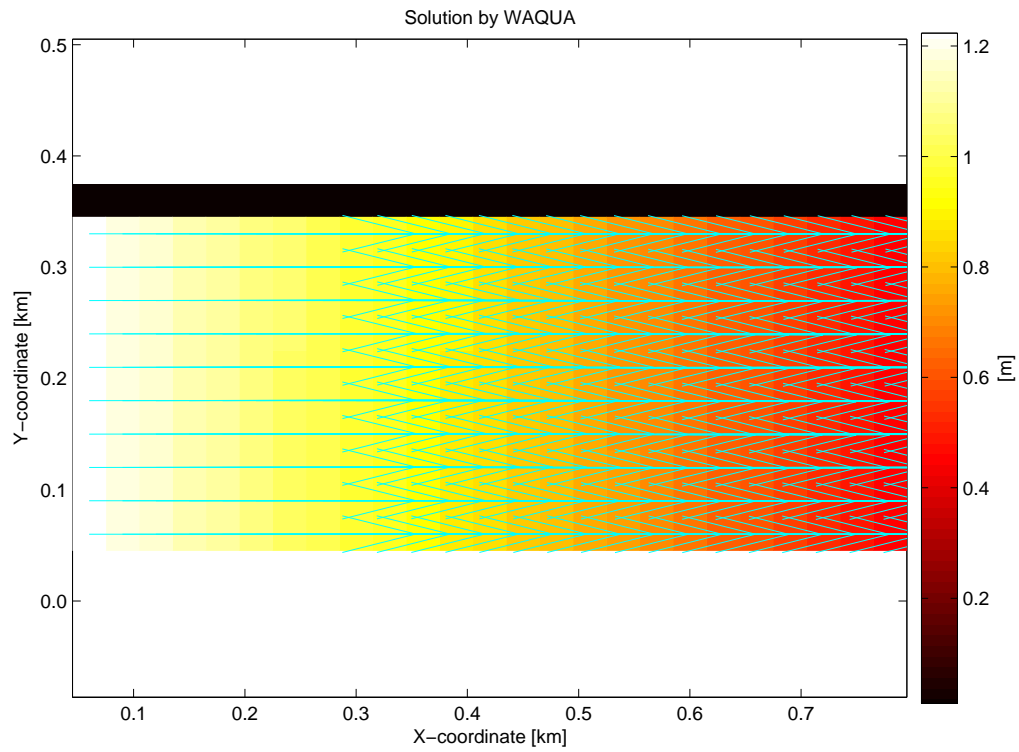


Figure 5.1: Chézy gutter, stationary solution computed by WAQUA.

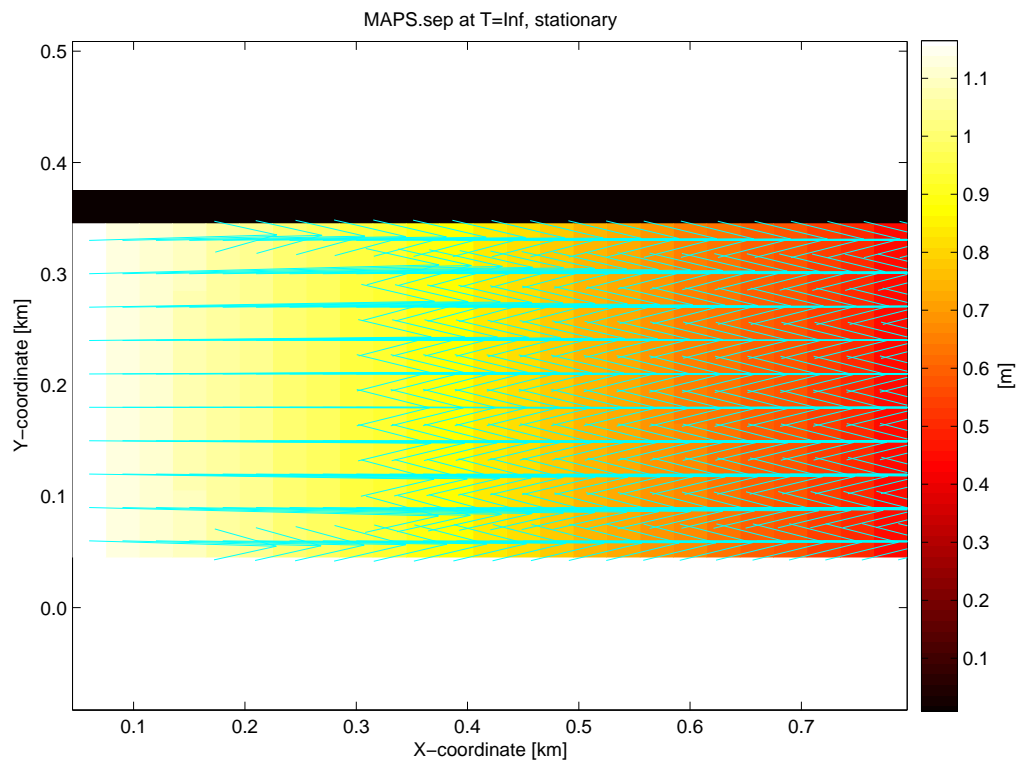


Figure 5.2: Chézy gutter, stationary solution computed by QuickFlow.

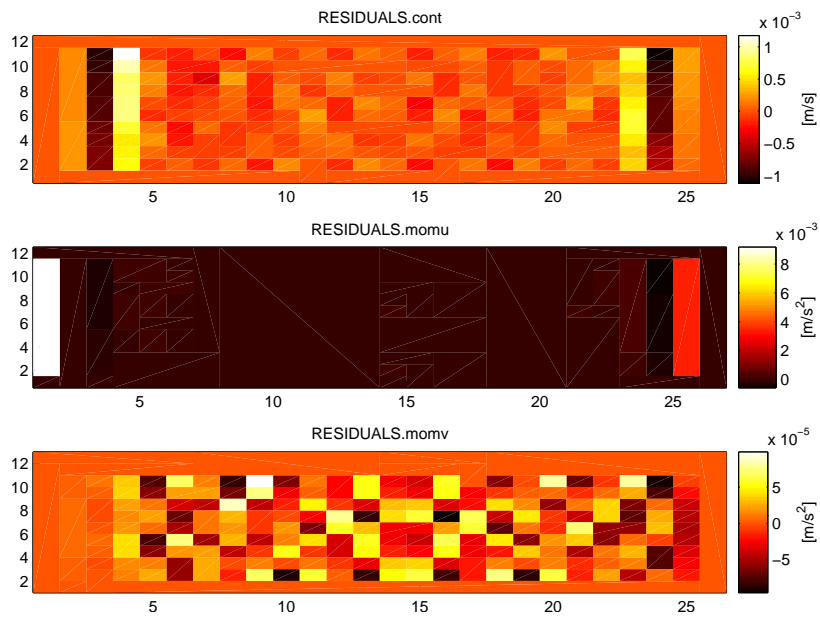


Figure 5.3: Chézy gutter, residuals of the WAQUA-solution, computed by Quick-Flow.



Figure 5.4: Overview of part of the Netherlands with some main rivers. River parts that are used as test model are indicated.



Figure 5.5: Satellite picture of the river part that is modelled in the Lek model and its surroundings.

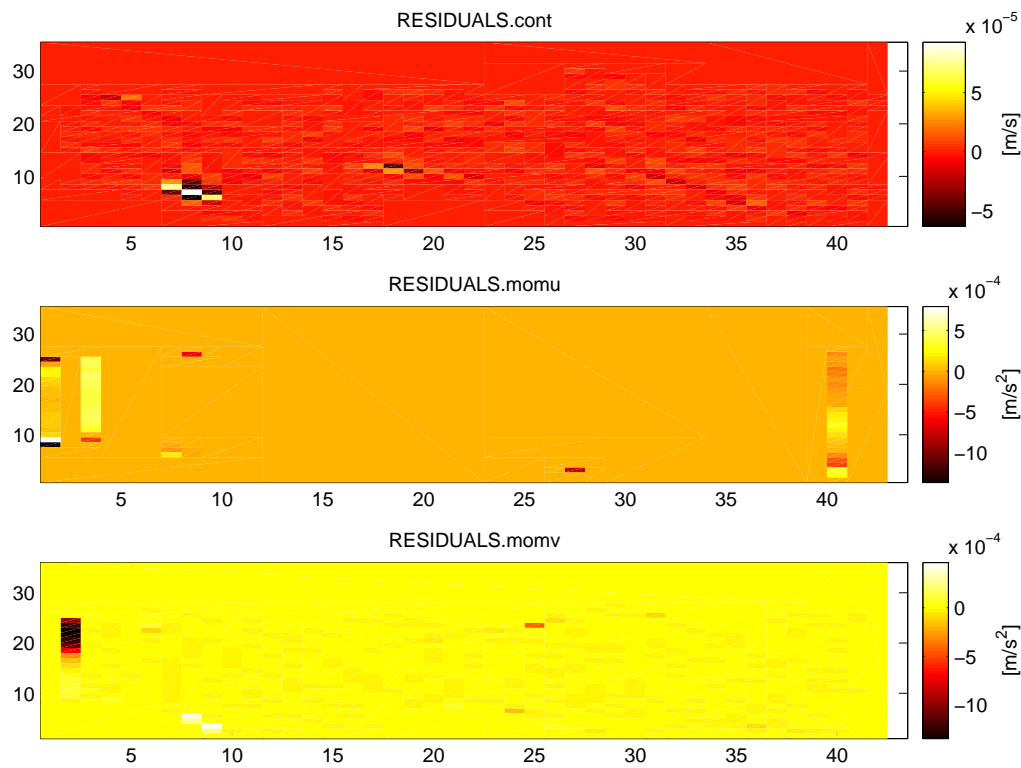


Figure 5.6: Lek, upper plot: residuals of continuity equation, center plot: residuals of  $U$ -momentum equations, lower plot: residuals of  $V$ -momentum equations.



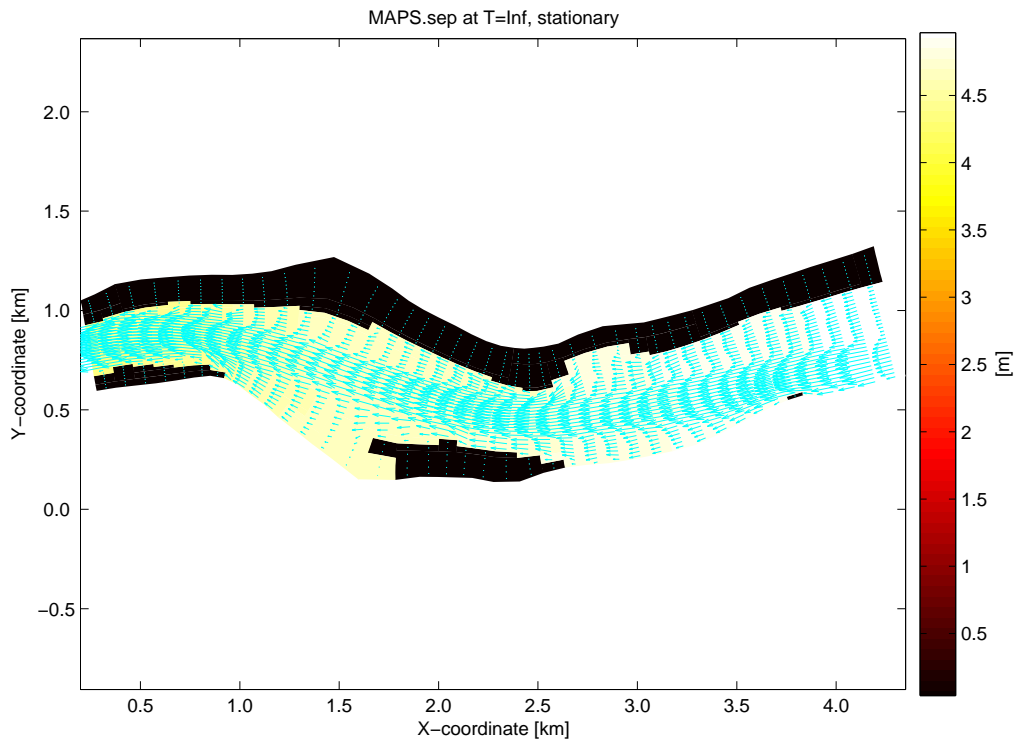


Figure 5.7: Lek, stationary solution computed by QuickFlow.

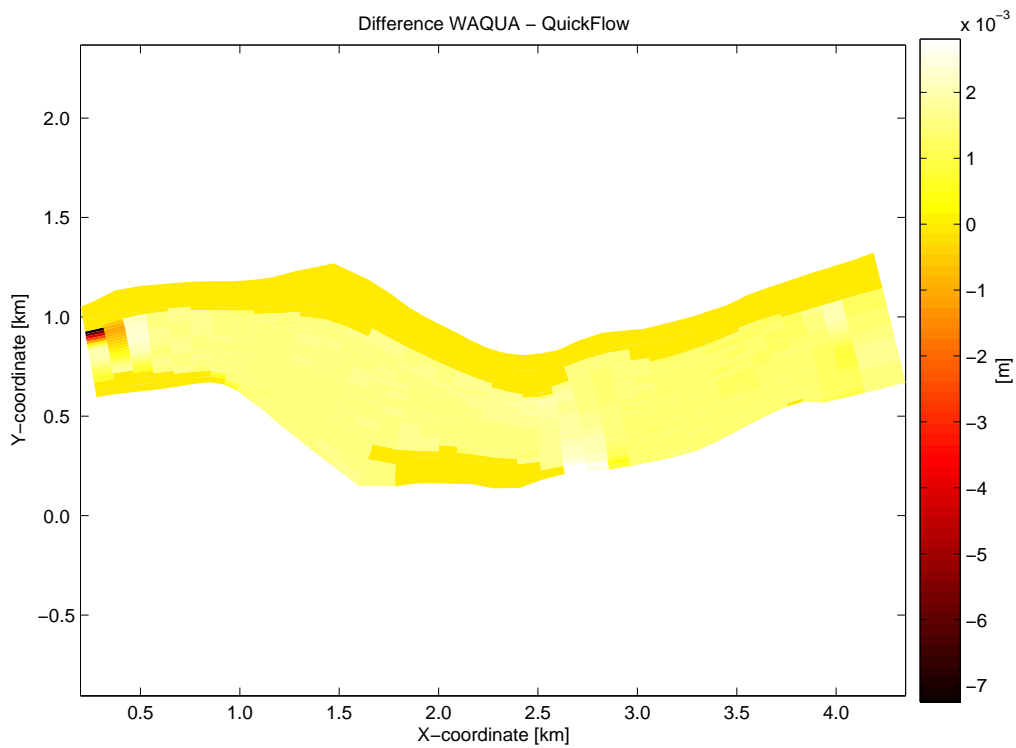


Figure 5.8: Lek, difference in water level computed by WAQUA and stationary solution computed by QuickFlow.



Figure 5.9: Satellite picture of the river part that is modelled in the Randwijk model and its surroundings.



Figure 5.10: Detail of the river part modelled in the Randwijk model.

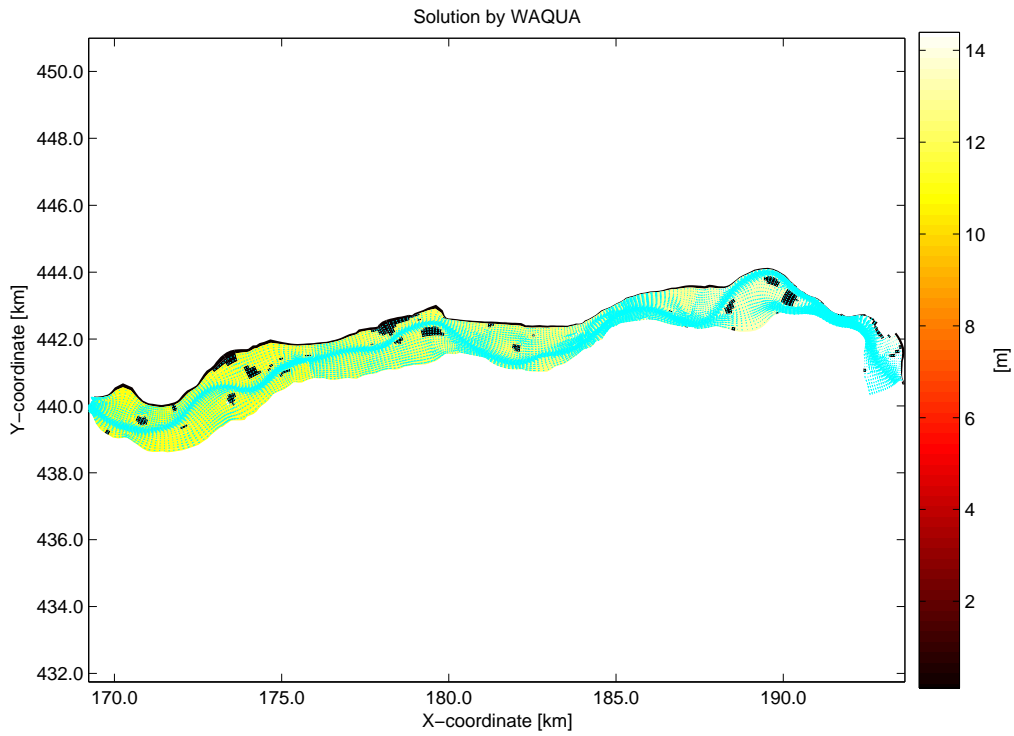


Figure 5.11: Randwijk, stationary solution computed by QuickFlow.

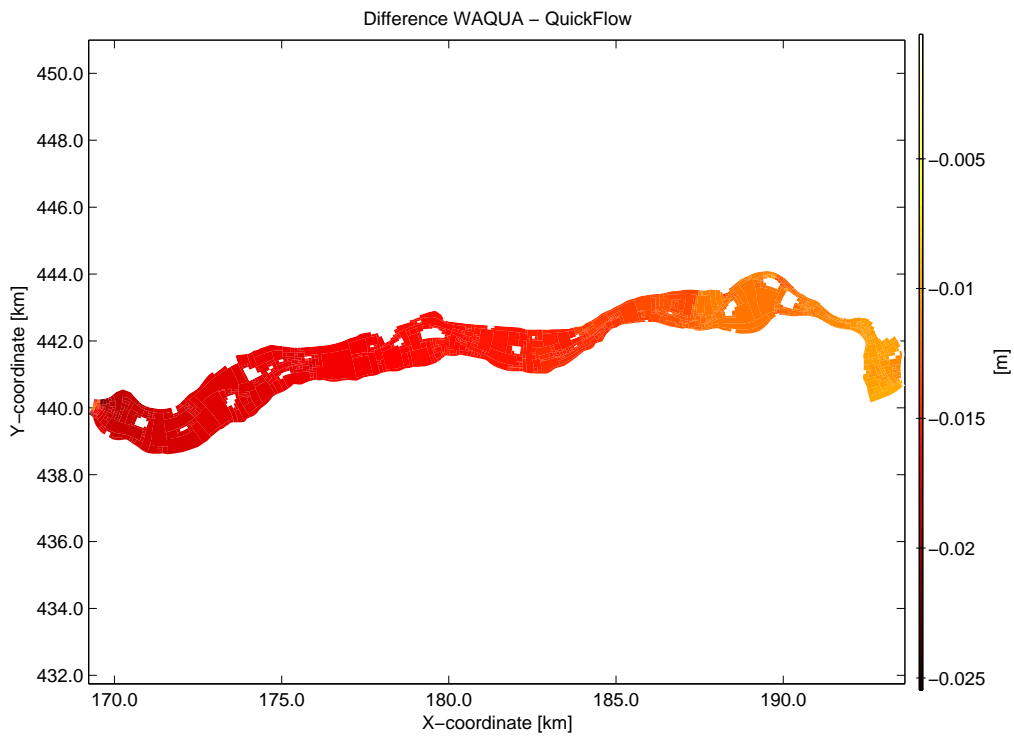


Figure 5.12: Randwijk, difference in water level computed by WAQUA and stationary solution computed by QuickFlow.

## Chapter 6

# Research goals and plan

### 6.1 Goal

In the next phase of this project we will try to improve QuickFlow. We want QuickFlow to compute a stationary solution for the shallow water equations applied on the Dutch rivers. Furthermore the solution should be accurate and the convergence fast.

We will restrict ourselves to the test problems given in Chapter 5. For these test problems we will compare the accuracy of the QuickFlow results with the results given by WAQUA. QuickFlow has several tools to visualize residuals and differences between solutions. We will use these tools, and whenever necessary, make more tools to visualize and/or measure how good a result is.

First we will focus on QuickFlow to compute a stationary solution for the same situation as in WAQUA, i.e., without any adjustments to the bottom or any other conditions. Secondly we will study the results if these adjustments are made. We expect that QuickFlow will give good results if a small adjustment is made, but the results will be worse for bigger adjustments. We will study how 'big' an adjustment can be. For example if a one meter deep hole of 20 by 20 meters will be digged in the bottom, we expect QuickFlow to give an accurate result within limited time. The convergence will be slower if this hole will be ten or hundred times wider and deeper.

A similar approach will be used for the initial conditions. We will first focus on problems for which the initial condition from WAQUA is rather stationary. This will be checked by computing the residuals. Later we will focus on situations where the residual of the initial condition is larger.

Note that it would be beyond the scope of this study to compare our simulation results with the physical reality. Therefore, we assume the results from

WAQUA to be realistic, which is a valid assumption in the cases under consideration.

## 6.2 Possible improvements

We have several suggestions to improve the performance of QuickFlow. In arbitrary order:

- apply (critical) damping boundary conditions to open boundaries;
- use an alternative method for time iteration which is not completely implicit like Euler Backward but implicit only for 'appropriate' terms;
- use an other iteration algorithm, which is less time consuming than the standard Newton's method;
- adaptive time stepping;
- adjust the shallow water equations with a 'pseudo water level' such that it is better capable of handling drying and flooding.

We will discuss these options in more detail below. In the Section 6.3 we will discuss how to determine which of these improvements will have priority.

### 6.2.1 Boundary conditions for open boundaries

At this moment the boundary conditions for the open boundaries are taken 'hard' as in Sections 3.5.3-3.5.4. This will introduce nonphysical reflections of waves at these boundaries. It takes time for these waves to leave the system. The system would more quickly stabilize if these reflections were reduced. This can be achieved by applying a reflection coefficient as described in Section 3.5.8 or by so called Sommerfeld boundary conditions, see [8].

Other interesting sources for this topic are [9, page 226], [10, page 161], [11] and [12, Sections 7.3.1 and 21.8.5].

### 6.2.2 Time iteration method

At this moment we use Euler Backward and Newton's method to do the time iteration in QuickFlow, as described in Section 4.2. The Euler Backward method is completely implicit. We can have a better performance if we consider well which terms we should take implicit and which explicit.

As an example we consider the advection equation:

$$u_t + \underbrace{u \frac{\partial u}{\partial x}}_{=\frac{1}{2} \frac{\partial u^2}{\partial x}} = 0. \quad (6.1)$$

We assume  $u > 0$  and use upwind discretization in space. We apply Euler Backward:

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \mathbf{A}(\mathbf{u}^{n+1}), \\ \text{with } \mathbf{A}(u_m^{n+1}) &= u_m^{n+1} \frac{u_m^{n+1} - u_{m-1}^{n+1}}{\Delta x}. \end{aligned} \quad (6.2)$$

Newton's method now gives:

$$\begin{aligned} \mathbf{u}_{k+1}^{n+1} &= \mathbf{u}_k^{n+1} - \left( (\mathbf{u}_k^{n+1} - \mathbf{u}_k^n - \Delta t \mathbf{A}(\mathbf{u}_k^{n+1}))' \right)^{-1} (\mathbf{u}_k^{n+1} - \mathbf{u}_k^n - \Delta t \mathbf{A}(\mathbf{u}_k^{n+1})) \\ &= \mathbf{u}_k^{n+1} - (\mathbf{F}(\mathbf{u})')^{-1} \mathbf{F}(\mathbf{u}) \\ \text{with } \mathbf{F}(\mathbf{u}) &= \mathbf{u}_k^{n+1} - \mathbf{u}_k^n - \Delta t \mathbf{A}(\mathbf{u}_k^{n+1}) \end{aligned} \quad (6.3)$$

It has been found<sup>1</sup> that we can find a more accurate result in fewer time steps. We therefore only take the point under consideration implicitly, the rest is taken explicitly. This leads to an semi-explicit method with an approximation of the function  $\mathbf{A}$ :

$$\begin{aligned} \mathbf{A}(\mathbf{u}^{n+1}) &= \mathbf{u}_m^{n+1} \frac{\mathbf{u}_m^{n+1} - \mathbf{u}_{m-1}^n}{\Delta x} = \mathbf{B}(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}) \\ &\approx \frac{\mathbf{u}_m^n \mathbf{u}_m^{n+1} - \mathbf{u}_m^n \mathbf{u}_{m-1}^n}{\Delta x} = \mathbf{B}(\mathbf{u}^n, \mathbf{u}^{n+1}) \end{aligned} \quad (6.4)$$

We can implement this semi-explicit method in 2 ways: either in the Euler Backward iteration or in the Newton iteration, which are explained in the next paragraphs.

### Semi-explicit time step

If we apply the above described method on the Euler Backward iteration we compute the next iteration as follows:

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \mathbf{B}(\mathbf{u}^n, \mathbf{u}^{n+1}), \\ \mathbf{u}_{k+1}^{n+1} &= \mathbf{u}_k^{n+1} - (\mathbf{G}(\mathbf{u})')^{-1} \mathbf{G}(\mathbf{u}), \\ \text{with } \mathbf{G}(\mathbf{u}) &= \mathbf{u}_k^{n+1} - \mathbf{u}_k^n - \Delta t \mathbf{B}(\mathbf{u}_k^n, \mathbf{u}_k^{n+1}). \end{aligned} \quad (6.5)$$

---

<sup>1</sup>We have discussed this informally with G.S. Stelling, a professor in fluid mechanics at the University of Technology Delft.

Note that the Euler Backward steps are adjusted, but we take complete Newton steps.

### Semi-explicit Newton

For the other implementation we take incomplete Newton steps, but we do not change the Euler Backward step:

$$\begin{aligned} \mathbf{u}^{n+1} &= \mathbf{u}^n + \Delta t \mathbf{B}(\mathbf{u}^{n+1}, \mathbf{u}^{n+1}) = \mathbf{u}^n + \Delta t \mathbf{A}(\mathbf{u}^{n+1}), \\ \mathbf{u}_{k+1}^{n+1} &= \mathbf{u}_k^{n+1} - (\mathbf{G}'(\mathbf{u}))^{-1} \mathbf{G}(\mathbf{u}), \\ &\text{with } \mathbf{G}(\mathbf{u}) = \mathbf{u}_k^{n+1} - \mathbf{u}_k^n - \Delta t \mathbf{A}(\mathbf{u}_k^{n+1}). \end{aligned} \quad (6.6)$$

This latter method (6.6) has not been studied yet. We revert our discussion to the method in (6.5). The upper equation of (6.5) can also be written as:

$$\frac{u_m^{n+1} - u_m^n}{\Delta t} + \frac{u_m^n u_m^{n+1} - u_m^n u_{m-1}^n}{\Delta x} = 0. \quad (6.7)$$

This discretization results in a matrix which has a oblique stencil and which is more diagonally dominant. This might be the reason why this 'trick' works so well, but it has not been thoroughly studied.

It has been found that this method performs well if it is applied to the advection term in the momentum equations. We do not know whether we can apply it to other terms and what the results would be.

In [13] and [14] semi-implicit schemes for three dimensional shallow water flow are presented. [15] presents for the shallow water equations multilevel schemes (time scheme adapted for spatial scales) and multistep schemes (time scheme adapted for operator under consideration).

### 6.2.3 Iteration algorithm

When we choose to apply an iteration algorithm, we need to consider its characteristics. One of the most important characteristics is the rate of convergence, see also [16, Section 4.1].

Suppose we want to solve  $\mathbf{F}(\mathbf{x}) = \mathbf{0}$ , with  $\mathbf{x}^*$  the exact solution. The numerical method is linearly converging in some norm  $\|\cdot\|$  if:

$$\lim_{n \rightarrow \infty} \frac{\|\mathbf{x}_{n+1} - \mathbf{x}^*\|}{\|\mathbf{x}_n - \mathbf{x}^*\|} = \mu, \text{ for some } \mu \in (0, 1). \quad (6.8)$$

The numerical method is superlinearly converging in some norm  $\|\cdot\|$  if  $\mu = 0$  in (6.8).

The method is superlinearly converging with order  $q$  in some norm  $\|\cdot\|$  if:

$$\lim_{n \rightarrow \infty} \frac{\|\mathbf{x}_{n+1} - \mathbf{x}^*\|}{\|\mathbf{x}_n - \mathbf{x}^*\|^q} = \mu, \text{ for some } \mu \in (0, 1). \quad (6.9)$$

In particular if  $q = 2$  in (6.9) we say that the method converges quadratically.

The convergence of Newton's method is at least quadratic, for a proof see [16, Section 5.1]. Drawbacks of Newton's method are the costs for the determination of the Jacobian  $\mathbf{F}'$  and of its LU-decomposition. This LU-decomposition is used to compute an  $\mathbf{s}$  such that  $\mathbf{F}'\mathbf{s} = -\mathbf{F}$ .

Broyden's method handles this drawbacks. This method is a quasi-Newton method. In a quasi-Newton we do not compute the Jacobian every Newton iteration. We only update the approximation of the Jacobian. If  $\mathbf{B} \approx \mathbf{F}'$  we can find the next iterate:

$$\begin{aligned} \mathbf{x}^{k+1} &= \mathbf{x}^k - (\mathbf{B}^k)^{-1}\mathbf{F}(\mathbf{x}^k), \\ &\text{with } \mathbf{B}^0 \approx \mathbf{F}'(\mathbf{x}^0) \text{ such that } \mathbf{s}, \text{ with } \mathbf{F}'\mathbf{s} = -\mathbf{F}, \\ &\text{can easily be determined.} \end{aligned} \quad (6.10)$$

This initial approximation of the Jacobian is called a preconditioner.

The quasi Newton method is determined by the way it updates the approximation of the Jacobian. In Broyden's method the approximation to the Jacobian is computed as follows:

$$\begin{aligned} \mathbf{B}^{k+1} &= \mathbf{B}^k + \frac{(\mathbf{F}(\mathbf{x}^{k+1}) - \mathbf{F}(\mathbf{x}^k) - \mathbf{B}^k \mathbf{y}) \mathbf{y}^T}{\mathbf{y}^T \mathbf{y}} \\ &= \mathbf{B}^k + \frac{(\mathbf{F}(\mathbf{x}^{k+1})) \mathbf{y}^T}{\mathbf{y}^T \mathbf{y}}, \\ &\text{with } \mathbf{y} = \mathbf{x}^{k+1} - \mathbf{x}^k. \end{aligned} \quad (6.11)$$

Broyden's method is locally superlinearly convergent but it uses a lot of computer storage. More details on Broyden's method can be found in [16, Chapter 7]. Broyden's method as it is described here is well suited for dense matrices. If we want to apply Broyden's method in QuickFlow we will, however, need a variant for sparse matrices.

An other way to improve the Newton algorithm in QuickFlow is to use a Krylov method, see for example [16, Chapter 2 and 3]. We expect, however, that this will not result in a substantial improvement as long as most other problems have not been solved.



### 6.2.4 Adaptive time stepping

#### Space dependent time step

Some sections of the rivers that we consider are more complicated than others. For example the flow in sections that can flood and dry is difficult to compute. In order to save computing time we can take large time steps in relatively easy regions, and in other, more complex regions we can take a small time step.

One way to find an appropriate time step is to determine a suitable CFL-number  $\nu \equiv \left| \frac{\mathbf{u}\Delta t}{\Delta x} \right|$  (compare with (4.19)) and adjust the timestep in every grid cell.

However this would take a lot of computer time and therefore it might be better to divide the domain in regions and determine different time steps for every region. We can also consider changing this time step every iteration or only after some number of iterations.

In [17] a local time stepping procedure is described which allows variation of the time step in both space and time. In [18] a method is described for steady flow simulations. These articles originated from aerospace related studies, [19] describes a method for space and time step adaptation in general hyperbolic partial differential equations.

#### Adjustment of time step for continuity equation

Furthermore we might improve the performance by taking the time step larger for the continuity equation than for the momentum equations. Consider the continuity equation in one dimension:

$$\frac{\partial \zeta}{\partial t} + \frac{\partial}{\partial x}(Hu) = 0. \quad (6.12)$$

If we take a time step  $\Delta t \rightarrow \infty$  we will find a divergence free discharge field:

$$\frac{\partial}{\partial x}(Hu) = 0. \quad (6.13)$$

If we now try to solve the system of equations with the momentum and continuity equations we will only find solutions with a divergence free discharge field. In the stationary solution the discharge field is divergence free as well. By limiting the solution space in this way we hope to find the stationary solution easier.

### 6.2.5 Pseudo water level

The last suggestion to improve the performance of QuickFlow is to implement a pseudo water level. This way the shallow water equations are adapted and the model will better handle drying and flooding. Implementation would take a lot of adjustments and is not within reach for this project.

## 6.3 Plan

In the next phase of this project we will implement some of the suggestions for improvement described in Section 6.2. But, we will start with an other adaptation. The functions  $\mathbf{A}_1$  and  $\mathbf{A}_2$  as introduced in Chapter 4 both only depend on  $\mathbf{u}(t + \Delta t)$ , see (4.22). Therefore they can be combined into one function  $\mathbf{A}(\mathbf{u}(t + \Delta t))$ . This will make the program easier to overview, but not necessarily to perform better. We will implement this first because it will make other adjustments easier.

Next, we will study which of the above described adaptations will substantially improve QuickFlow's performance. We can use several techniques to find out which adaptation should be made first. For example we will look where large residuals in the initial solution occur and where the solution by QuickFlow differs a lot from the solution by WAQUA.

We will start looking for difficulties that occur in the easiest test problems, e.g. the Chézy gutter, with a small initial residual. Later we can expand our view to the other test problems and to problems with a larger initial residual and to problems with adapted bottom.

Part of the numerical problems may be caused by programming errors. For example we have seen a large residual near the open boundaries in the Chézy gutter and in the Lek model. A programming error might well be the cause. One of our first actions will be to analyse the cause for the large residual and fix the problem.

We will also have to take into account the machine precision. The WAQUA software only computes in single precision. While QuickFlow is programmed in Matlab and computes in double precision.

# Chapter 7

## Conclusion

In this report, we have discussed how river flows can be modelled by the shallow water equations together with appropriate boundary conditions. The numerical discretization in space that is used by WAQUA and QuickFlow is discussed. We have seen that major differences between WAQUA and QuickFlow occur in the methods to compute the stationary solution that were discussed in Chapter 4. WAQUA uses the ADI-method, which gives a good result for accurate computation of time dependent solutions. It is, however, not well suited for computing a stationary solution. The solution methods used by WAQUA have been adapted for use by QuickFlow. QuickFlow uses Euler Backward for the time steps. The resulting equations are linearized and solved using Newton's method with line search.

Now that we know the most important characteristics of WAQUA and QuickFlow, we will start to improve the convergence of QuickFlow. In the previous chapter we have discussed the most important possible methods:

- reduce oscillations by applying damping boundary conditions;
- use a semi-explicit time scheme;
- use a more efficient iteration algorithm; and
- adaptive time stepping.

First we will study the possibilities to implement these improvements in more detail. Subsequently we will implement the chosen improvements and test them using the test problems described in Chapter 5.

Our goal is to improve QuickFlow in such a way that it can find the stationary solution for all test problems within a reasonable time.

Below a time schedule for the rest of this project is shown.

<b>Action</b>	<b>duration (weeks)</b>	<b>start</b>	<b>end</b>
Analyze and solve problems in Chézy gutter	6	2 January	9 February
Analyze and solve problems in Lek model	6	12 February	30 March <sup>1</sup>
Analyze and solve problems in Randwijk model	6	2 April	11 May
Finish report & extra time	5	14 May	15 June
Make, prepare and give presentation	2	18 June	30 June

---

<sup>1</sup>Including 1 week vacation.



# Notation

$\mathbf{A}_{1,2}$	function in first (second) ADI stage	
$C_{2D}$	(2D) Chézy coefficient	$\text{m}^{1/2}/\text{s}$
$C_{2D,x,y}$	(2D) Chézy coefficient in $x(y)$ -direction	$\text{m}^{1/2}/\text{s}$
$d$	water depth below some horizontal plane	$\text{m}$
$F_{\zeta,U,V,Q,R}$	boundary value for water level ( $U$ -velocity, $V$ -velocity, discharge or QH, Riemann invariant) boundary	
$f$	Coriolis parameter	$\text{s}^{-1}$
$\text{Fr}$	Froude number	
$g$	acceleration due to gravity	$\text{m}/\text{s}^2$
$\hat{H}$	typical vertical length	$\text{m}$
$H = d + \zeta$	total water depth	$\text{m}$
$H^{U,V}$	total depth at $U(V)$ -velocity point	$\text{m}$
$\hat{L}$	typical horizontal length	$\text{m}$
$l$	time iteration indicator	
$m$	grid index in $x(\xi)$ -direction	
$m_f$	grid index in $x(\xi)$ -direction at boundary	
$n$	grid index in $y(\eta)$ -direction	
$p$	pressure	$\text{kg}/\text{m}/\text{s}^2$
$p_{\text{atm}}$	atmospheric pressure	$\text{kg}/\text{m}/\text{s}^2$
$Q$	discharge (2-dimensional)	$\text{m}^2/\text{s}$
$\text{Re}$	Reynolds number	
$\text{Re}_t$	Reynolds number with turbulence included	
$\text{Ro}$	Rossby number	

$T_{xx,yy,xy}$	lateral (or Reynold's) stresses	kg/m/s <sup>2</sup>
$t$	time coordinate	s
$\Delta t$	time step	s
$\hat{U}$	typical velocity in any horizontal direction	m/s
$U, V, W$	averaged velocity in $x(y, z)$ -direction	m/s
	averaged velocity in $\xi(\eta)$ -direction	m/s
$\bar{U}, \bar{V}$	arithmetic average of surrounding $U(V)$ -velocity points	m/s
$ \mathbf{U}  = \sqrt{U^2 + V^2}$	magnitude of velocity	m/s
$u, v, w$	velocity in $x(y, z)$ -direction	m/s
	velocity in $\xi(\eta)$ -direction	m/s
$u', v', w'$	random variation on velocity in $x(y, z)$ -direction	m/s
$\mathbf{u}$	solution vector ( $U, V, \zeta$ )	
$\mathbf{u}^{*W}, \mathbf{u}^{*Q}$	stationary solution found with WAQUA (Quick-Flow)	
$x, y, z$	Cartesian coordinates	m
$\Delta x, \Delta y$	grid distance in $x(y)$ - direction	m
$\alpha$	reflection coefficient for open boundaries	
$\delta$	treshold value for drying and flooding	m
$\xi, \eta$	coordinates of orthogonal curvilinear grid	
$\zeta$	water level above some horizontal plane	m
$\nu$	viscosity	m <sup>2</sup> /s
$\nu_t$	eddy viscosity	m <sup>2</sup> /s
$\nu_H$	horizontal eddy viscosity	m <sup>2</sup> /s
$\rho$	density	kg/m <sup>3</sup>
$\rho_0$	constant reference density	kg/m <sup>3</sup>
$\tau$	shear stress	kg/m/s <sup>2</sup>
$\tau_{\text{bottom},x,y,\xi,\eta}$	shear stress at bottom (in $x(y, \xi, \eta)$ -direction)	kg/m/s <sup>2</sup>
$\tau_{i,j}$	viscous stresses	

## Appendix A

# Relation WAQUA and QuickFlow

In Figure A.1 the processes in WAQUA and its relation with QuickFlow is shown. The input for WAQUA consists for example of GIS (Geographic Information System) maps, the grid (which is generated by an other program), measurement data (for example the relation between water level and discharge at a certain location), and model parameters (for example the bottom roughness at a certain location).

This input is stored in the siminp-file for one specific model. This is an ASCII-file. Waqpre uses the data from the siminp-file for preprocessing. It prepares and puts ready the data that will be used later by waqpro. For example, when we use time dependent boundary conditions, it computes and stores the boundary conditions for every time step.

The output of waqpre is stored in the SDS-file. This data is used by waqpro. Waqpro discretizes the equations on the given grid and computes the velocities and water levels for the demanded time interval. After every time step it writes its solution in the SDS-file and uses it in the next time step.

Whenever necessary postprocessing software can be used to make the WAQUA solution in the SDS-file visible in a plot.

QuickFlow uses the data from the SDS-file as well. In Figure A.2 the processes in QuickFlow are shown. QuickFlow takes the solution of the last time step from the SDS-file. It is assumed that this is the stationary solution. However, in practice it is only an approximation of the stationary solution and QuickFlow first determines whether this solution is 'good enough'. Therefore, QuickFlow computes the initial residual and computes the stationary solution. In the ideal case QuickFlow would find exactly the same stationary solution as WAQUA and



the residual would be zero. In practice this will rarely be the case and the results are postprocessed for comparison. The residual and the difference between the approximate stationary solution from WAQUA and the stationary solution from QuickFlow are plotted. At this moment the user has to inspect these results visually and decide whether he can use the approximate stationary from WAQUA as initial solution for further computations.

QuickFlow was designed to compute new stationary solutions after adjustments to the bottom and boundaries. These simulations can start now. The user provides QuickFlow with information on the adjustments by way of a GUI (Graphical User Interface). QuickFlow computes the new stationary solution after these adjustments. Again the results are postprocessed and made visible for the user in a plot.

In practical situations the end user would try different adjustments and inspect the results. Then he chooses which adjustment(s) give(s) the desired velocities and water levels. He can check this results with WAQUA, which is supposed to give a more accurate result. Moreover, WAQUA is able to compute time dependent solutions, where QuickFlow only gives the stationary solution.

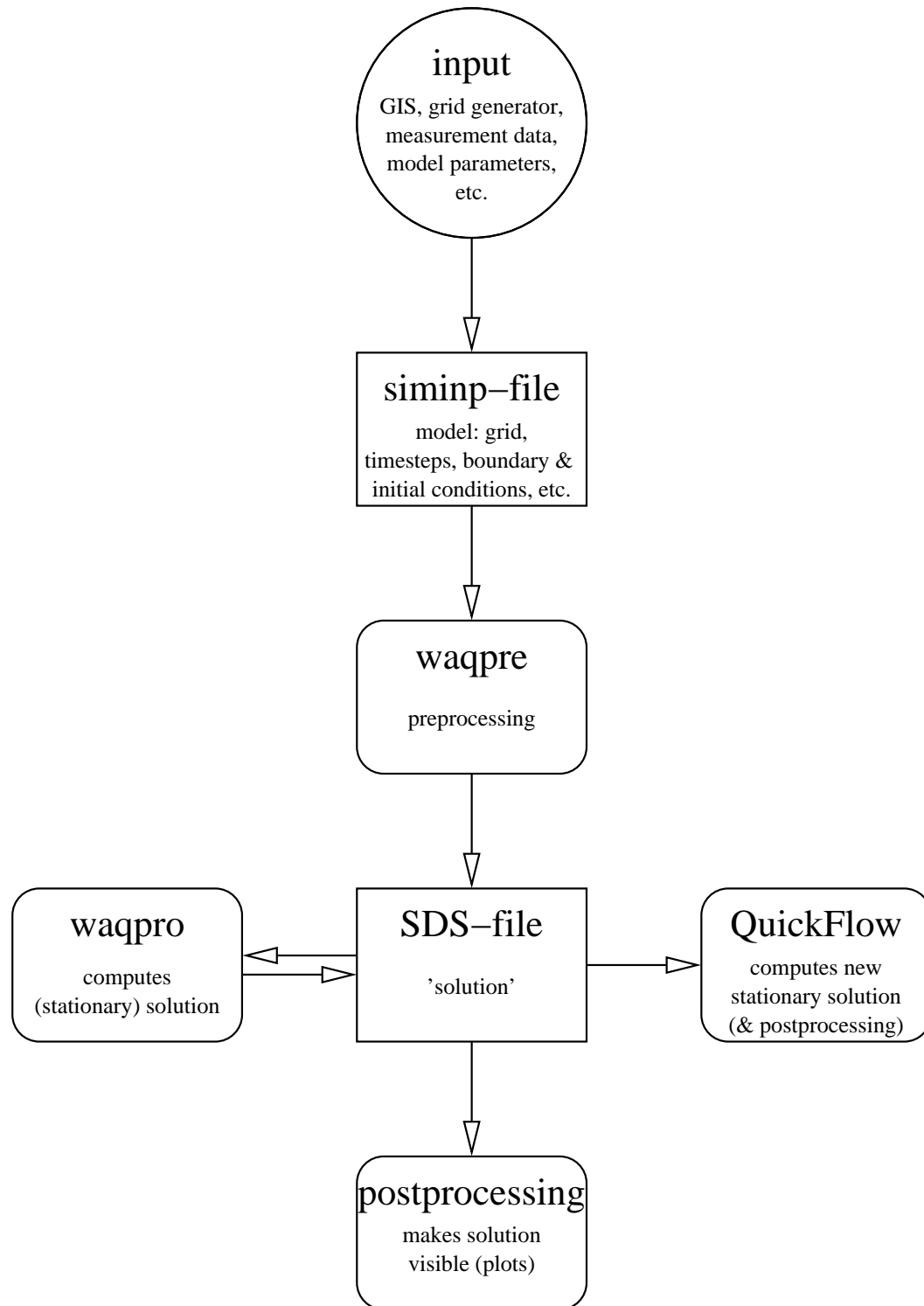


Figure A.1: Scheme of the relation between WAQUA and QuickFlow.

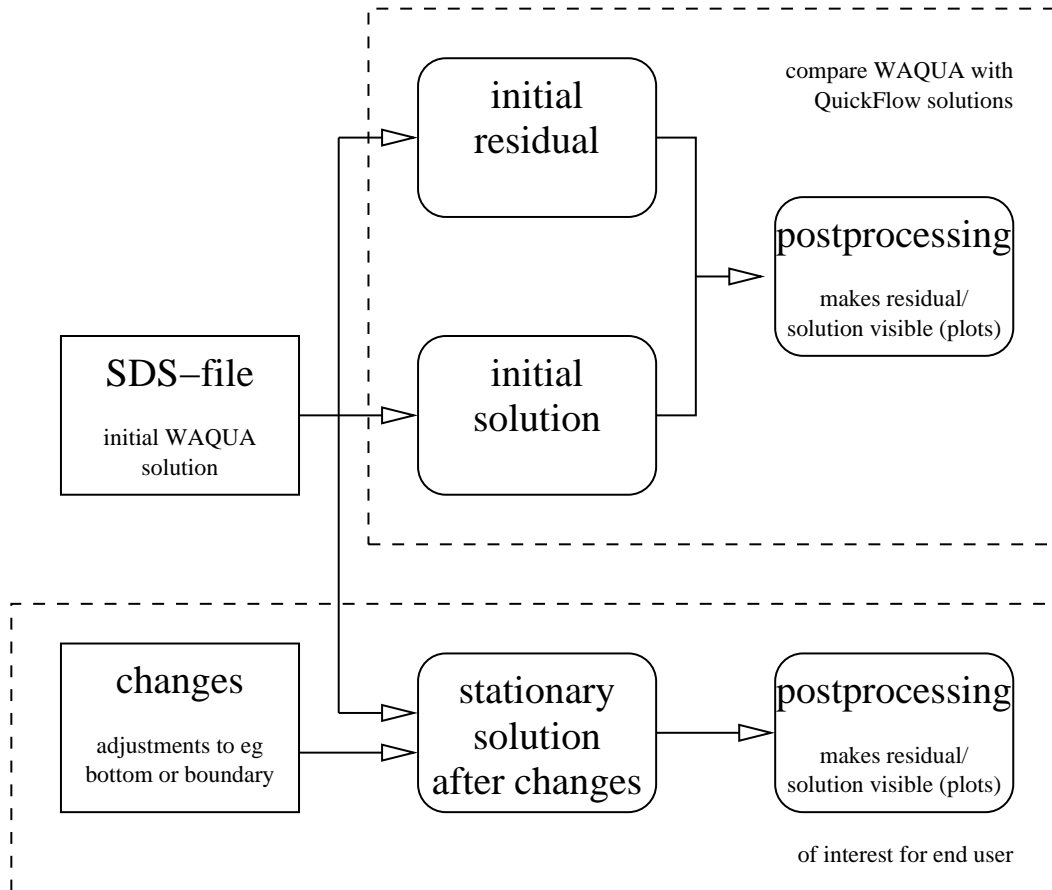


Figure A.2: Scheme of the processes in QuickFlow.

## Appendix B

# Stencils for spatial discretization





-  under consideration
-  involved in discretization
-  involved in discretization if  $V < 0$
-  involved in discretization if  $V \geq 0$

Figure B.1: Legend for stencils below: Figure B.2-B.10.

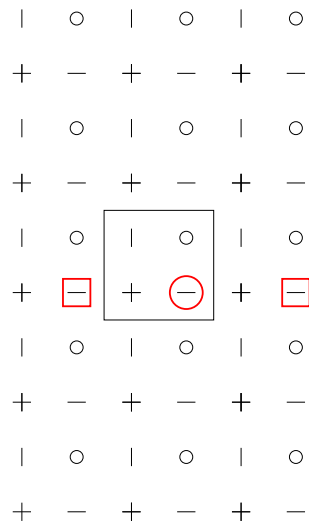


Figure B.2: Points involved in the discretization of the advection term  $U \frac{\partial U}{\partial x}$ .

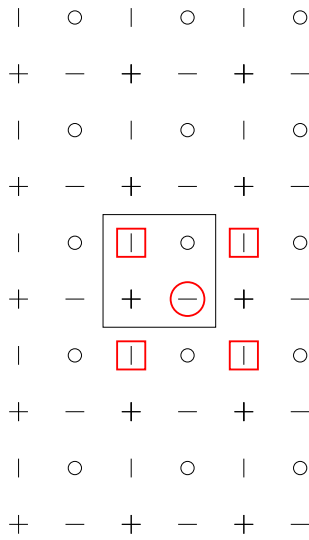


Figure B.3: Points involved in the discretization of  $\bar{V}$ .

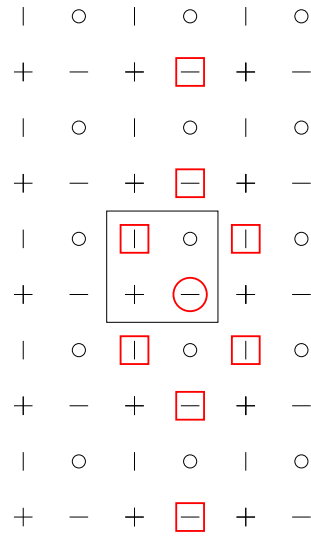


Figure B.4: Points involved in the discretization of the explicit part of the cross advection term  $V \frac{\partial U}{\partial y}$ .

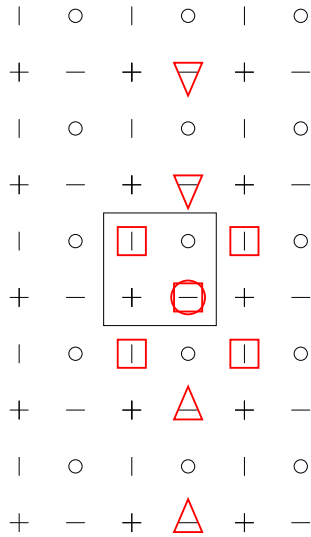


Figure B.5: Points involved in the discretization of the implicit part of the cross advection term  $V \frac{\partial U}{\partial y}$ .

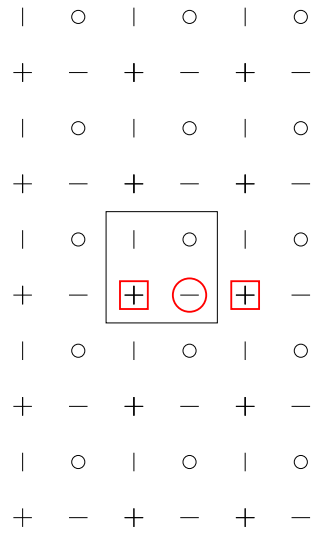


Figure B.6: Points involved in the discretization of the pressure gradient  $\frac{\partial \zeta}{\partial x}$ .

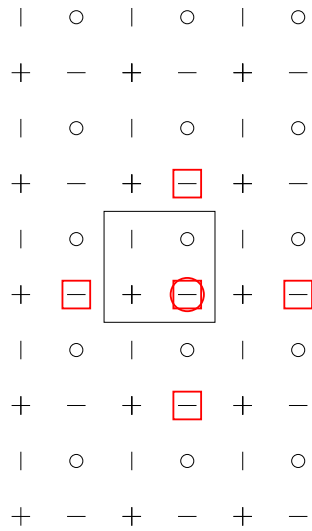


Figure B.7: Points involved in the discretization of the horizontal viscous term  $\nu_H \left( \frac{\partial^2 U}{\partial x^2} + \frac{\partial^2 U}{\partial y^2} \right)$ .

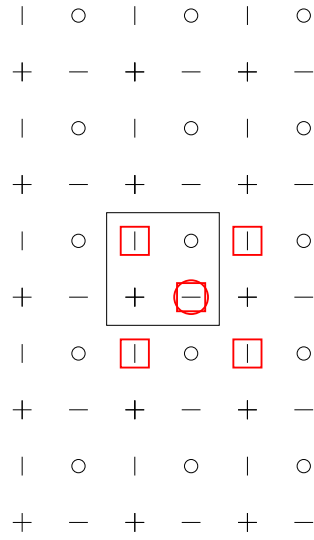


Figure B.8: Points involved in the discretization of the bottom friction term  $\tau_{\text{bottom},x}$ .

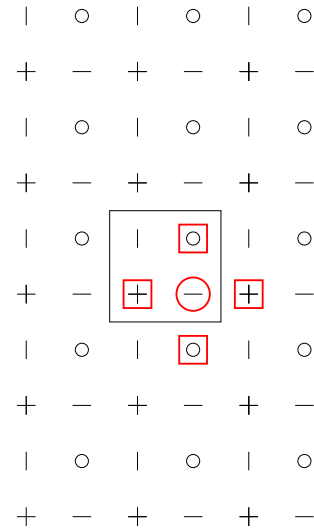


Figure B.9: Points involved in the approximation of  $H^U$  in the discretization of the continuity equation.



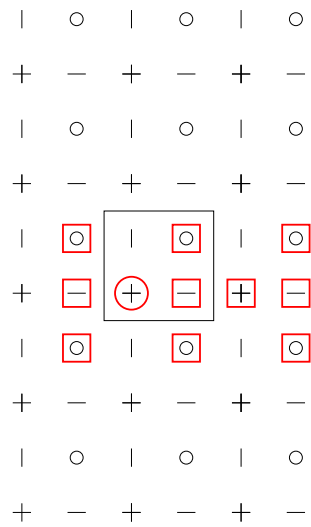


Figure B.10: Points involved in the approximation of  $(HU)_x$  in the discretization of the continuity equation.

## Appendix C

# Thomas' algorithm

Systems like  $\mathbf{Ax} = \mathbf{d}$  can be solved using the Thomas' algorithm (also known as tridigonal matrix algorithm) if  $\mathbf{A}$  is tridiagonal:

$$A = \begin{pmatrix} b_1 & c_1 & & \emptyset \\ a_2 & b_2 & \ddots & \\ & \ddots & \ddots & c_{k-1} \\ \emptyset & & a_k & b_k \end{pmatrix}. \quad (\text{C.1})$$

The theory below is taken from [2, Section 7.4].

The tridiagonal system is reduced to a bidiagonal system:

$$\begin{pmatrix} 1 & \gamma_1 & & \emptyset \\ & \ddots & \ddots & \\ & & \ddots & \gamma_{k-1} \\ \emptyset & & & 1 \end{pmatrix} = \delta, \quad (\text{C.2})$$

$$\begin{aligned} \text{with } \gamma_1 &= \frac{c_1}{b_1}, \\ \gamma_i &= \frac{c_i}{b_i - a_i \gamma_{i-1}} \quad \text{for } i = 2, 3, \dots, k, \\ \delta_1 &= \frac{d_1}{b_1}, \\ \delta_i &= \frac{d_i - a_i \delta_{i-1}}{b_i - a_i \gamma_{i-1}} \quad \text{for } i = 2, 3, \dots, k. \end{aligned} \quad (\text{C.3})$$

Now the solution of  $\mathbf{Ax} = \mathbf{d}$  can be obtained by backward substitution:

$$\begin{aligned} x_k &= \delta_k, \\ x_i &= \delta_i - \gamma_i x_{i+1} \quad \text{for } i = k-1, k-2, \dots, 1. \end{aligned} \tag{C.4}$$

In this method rounding off errors might amplify. This is prevented if the matrix  $\mathbf{A}$  is diagonal dominant:

$$|b_i| > |a_i| + |c_i| \quad \text{for } i = 1, 2, \dots, k. \tag{C.5}$$

For the systems solved in WAQUA and QuickFlow this leads to the restriction:

$$\max \left( \frac{\Delta t |U_{m,n}^l|}{\Delta x}, \frac{\Delta t |V_{m,n}^l|}{\Delta y} \right) \leq 4. \tag{C.6}$$

# Bibliography

- [1] C.B. Vreughdenhil. *Numerical Methods for Shallow-Water Flow*. Kluwer Academic Publishers, Dordrecht, 1994.
- [2] WL|Delft Hydraulics, Delft. *Technical documentation WAQUA*, April 2005. Version 1.2.
- [3] B. van 't Hof, A.E.B. Veldman, and E.A.H. Vollebregt. MoMEC discretizations; applied to shallow water simulations on curvilinear grids. Memo, VORtech Computing, August 2005. Not yet published.
- [4] G.S. Stelling. On the construction of computational methods for shallow water flow problems. Rijkswaterstaat Communications 35, Rijkswaterstaat, The Hague, 1984. Also appeared as PhD thesis at University of Technology Delft, 1983.
- [5] MX.Systems / Ministry of Transport, Public Works and Water Management; Directorate-General for Public Works and Water Management, Rijswijk. *User's guide WAQUA; technical information*, November 2005. Version 10.41.
- [6] B. van 't Hof. Artificial porosity; drying and flooding without screens. PowerPoint presentation, 2005.
- [7] J.J. Leendertse. *Aspects of a computational model for long-period water-wave propagation*. PhD thesis, University of Technology Delft, 1967.
- [8] R. Vichnevetsky and E.C. Pariser. High order numerical sommerfeld boundary conditions: theory and experiments. *Computers and mathematics with applications*, 11(1-3):67–78, 1985.
- [9] J.A. Battjes. *Vloeistofmechanica; collegehandleiding CTme2100*. Technische Universiteit Delft; Faculteit Civiele Techniek en Geowetenschappen; Sectie Vloeistofmechanica, March 2000.

- [10] J.A. Battjes. *Stroming in waterlopen; collegehandleiding CTwa3310*. Technische Universiteit Delft; Faculteit Civiele Techniek en Geowetenschappen; Sectie Vloeistofmechanica, January 2000.
- [11] J. van Kester, G. Stelling, A. Bijlsma, and T van der Kaaij. *Syllabus Randvoorwaarden WAQUA en TRIWAQ*, January 2001.
- [12] R.J. LeVeque. *Finite Volume Methods for Hyperbolic Problems*. Cambridge texts in Applied Mathematics. Cambridge University Press, Cambridge, 2002.
- [13] V. Casulli and E. Cattani. Stability, accuracy and efficiency of a semi-implicit method for three-dimensional shallow water flow. *Computers and mathematics with applications*, 27(4):99–112, 1994.
- [14] V. Casulli and P. Zanolli. Semi-implicit numerical modeling of nonhydrostatic free-surface flows for environmental problems. *Mathematical and Computer Modelling*, 36:1131–1149, 2002.
- [15] T. Dubois, F. Jauberteau, R.M. Teman, and J. Tribbia. Multilevel schemes for the shallow water equations. *Journal of computational physics*, 207:660–694, 2005.
- [16] Kelley. *Iterative Methods for Linear and Nonlinear Equations*. Number 16 in Frontiers in Applied Mathematics. Society for Industrial and Applied Mathematics, Philadelphia, 1995.
- [17] S. Chang, Y. WU, V Yang, and X. Wang. Local time-stepping procedures for the space-time conservation element and solution element method. *International Journal of Computational Fluid Dynamics*, 19(5):359–380, 2005.
- [18] T. Imamura, K. Suzuki, T. Nakamura, and M. Yoshida. Acceleration of steady-state lattice boltzmann simulations on non-uniform mesh using local time step methos. *Journal of Computational Physics*, 202:645–663, 2005.
- [19] P. Lötstedt, S. Söderberg, A. Ramage, and L. Hemmingsson-Frändén. Implicit solution of hyperbolic equations with space-time adaptivity. *BIT*, 42(1):134–158, 2002.