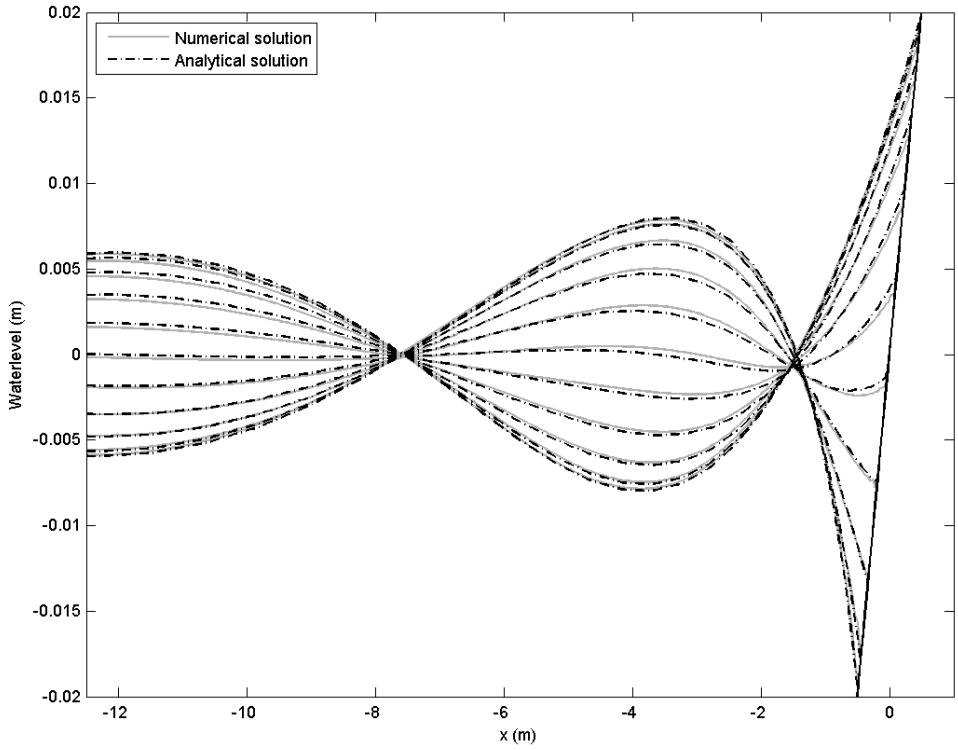
Implementation of a wetting and drying algorithm in a finite element model



Anna Kroon
September 17, 2009

Svašek Hydraulics

Delft University of Technology
Faculty of Civil Engineering
Section of Fluid Mechanics

SVAŠEK HYDRAULICS
COASTAL, HARBOUR AND RIVER CONSULTANTS

TU Delft

MSc Thesis
Delft University of Technology

Implementation of a wetting and drying algorithm in a finite element model

Anna Kroon

Supervising committee:
Prof.dr.ir. G.S. Stelling
Prof.dr.ir. C. Vuik
Dr.ir. R.J. Labeur
Ir. H. Talstra

Summary

In numerical modeling of water movement, wetting and drying is a well known problem. The governing equations are not valid in the dry part of the computational domain which may result in problems with mass conservation, negative water depths and artificially enlarged gradients.

In this study a wetting and drying algorithm is implemented in a 2DH finite element flow model based upon the wetting and drying algorithm proposed by Casulli [8]. It is investigated whether this procedure is mass conservative, efficient and robust. To this end a simple finite element discretization of the inviscid shallow water equations (SWE) is derived.

The assumption of inviscid flow enables the use of piecewise constant basis functions for the velocity. The advantage of this is that the velocity at the new time level can be derived directly from the momentum equation. In combination with the pressure correction method an uncoupled system is established in which only one matrix vector system needs to be solved.

The proposed wetting and drying procedure introduces an extra non-linear term in the system which needs to be solved iteratively.

Special attention is paid to the implementation of the advection term since its non-linearity causes several problems and in addition momentum conservation over the wet/dry interface is an important part of the wetting and drying procedure.

In order to solve the system additional measures should be taken to prevent the system matrix of becoming singular. Several measures are investigated on convergence speed of the wetting and drying iteration and mass conservation. The best working solution turned out to be the use of a minimum cell averaged depth in the linear part of the system matrix, to prevent singularities and enhance robustness, in combination with a lumped mass matrix in the dry part of the domain, to guarantee mass conservation, and a velocity that is put to zero for very small water depths.

The resulting procedure is validated with several one- and two-dimensional analytical solutions for: 1) a one dimensional dam break, 2) flow over a long crested weir, 3) a one-dimensional oscillating water surface in a parabolic basin, 4) the run-up of long waves on a beach, 5) a two-dimensional standing wave in a parabolic basin and 6) the spreading of a parabolic flood wave in two dimensions. In addition two laboratory experiments are used for validation: one experiment with solitary wave runup on a conical island and one experiment with a two dimensional dam break. The scheme is able to represent all the tests correctly except for the two-dimensional dam break where the front celerity is slightly too low and the lateral spreading is too large. However this is a severe test case and in the numerical model some crude assumptions have been made that may have caused this. Since the spreading of a parabolic flood wave, which exhibits comparable wetting and drying behavior, is represented well by the model it is not likely that the problems with the two-dimensional dam break are caused by the wetting and drying procedure.

In general the methods performance is good. However in two-dimensions it can be beneficial, in case of small gradients at the wet/dry interface, to use a lumped mass matrix at partially dry elements too. The wetting and drying iteration converges on average in 2 to 3 iteration steps. In some cases bifurcations and mass errors can occur. However mass errors are caused by rounding errors and can be resolved by using double precision. The occurrence of bifurcations is much less frequent in case of calculations in double precision and can be minimized by adjusting the BiCGSTAB settings.

Acknowledgements

This thesis is the result of the Master of Science program at the faculty of Civil Engineering and Geosciences at Delft University of Technology, The Netherlands. This study has been carried out under the guidance of the section of Environmental Fluid Mechanics in cooperation with Svašek Hydraulics.

I would like to thank Robert Jan Labeur, my supervisor, for his help and guidance and his never ending patience. I would not have come this far without him as my teacher. I would like to thank Harmen Talstra, my supervisor at Svašek Hydraulics for his help and disability to let a problem go before it is solved. If I came to him for help he would not rest before he found the answer to my problem. Furthermore I want to thank the other members of my committee and all employees of Svašek Hydraulics for their help and support. Moreover I could not have done this without the help of my friends and family. They have helped me through the hard times and listened to me even though they did not understand the majority of my problems.

Anna Kroon

Contents

Chapter 1

Introduction

In civil engineering practice it is often desired to obtain insight in the governing hydrodynamic conditions, in a certain area, for design, maintenance or construction purposes. Field measurements can be used to gather this data but this is often costly and time consuming. In addition to or even instead of measurements numerical models are widely used to define the hydrodynamic conditions of interest.

Numerical models are basically a translation of a physical model that describes the governing processes into a problem that a computer is able to solve. The conversion of a physical problem to a numerical problem comprises a number of steps:

- Describing the physical problem by a (set of) differential equation(s);
- Making assumptions regarding subjects still poorly understood or assumptions to reduce the extent of the problem;
- Establishing the boundaries of the area of interest;
- Divide the area of interest into a finite number of sub-domains.

Several numerical methods are available for the translation of a set of differential equations to a numerical model. The finite element method is a numerical method which makes it possible to follow the outline of shorelines, complicated bathymetries and hydraulic structures closely. Moreover a grid refinement in an area of interest can be applied locally. In this way accurate estimation of the governing hydrodynamics close to the structure of interest is enabled with relatively little extra computational efforts.

For a numerical model that describes the movement of water it is not hard to imagine that problems will arise. In real life hydrodynamic modeling of coastlines, estuaries, lagoons and other shallow areas, the shoreline is not continually at the same location in the model domain. It can vary over several cells within one time step. The main problem is that the governing equations are only valid in the wet part of the domain. Moreover problems are induced by the finite dimensions of the grid cells. Near (large) bottom gradients in a cross-section for example negative water depths can be induced. As a result mass conservation is not valid anymore. On top of that water depths approaching zero can lead to numerical instabilities.

Over the years numerous methods have been developed to solve the problem of wetting and drying. Each of them having their own advantages and disadvantages. Advantageous

properties can be simplicity, calculation effort, physical background or strict mass conservation. Disadvantageous properties could be poor mass conservation, negative water depths, artificially enlarged gradients and nonphysical wave reflection.

It is clear that wetting and drying is a nightmare (or a challenge) to many programmers, engineers and mathematicians. The search for the perfect wetting and drying algorithm has certainly not ended yet. Nevertheless it is hoped that this study can contribute somehow to this search.

1.1 Objective

The objective of this study is to develop a wetting and drying algorithm for a 2DH finite element flow model based upon the wetting and drying algorithm proposed by Casulli [8] and to investigate whether this procedure is mass conservative, efficient and robust. To this end three sub-objectives will be formulated:

- Develop a simple one-dimensional model that gives maximal insight into the occurring physical and mathematical processes.
- Implement a wetting and drying procedure in this one-dimensional model and investigate whether this procedure is mass-conservative, efficient and robust.
- Extend this model and wetting and drying procedure to two dimensions and investigate whether this procedure is mass-conservative, efficient and robust.

1.2 Outline

In the remainder of this chapter a short explanation of the wetting and drying problem will be given. Subsequently the governing equations will be given and several methods available in literature will be discussed. In Chapter 2 the basic principles of the finite element method will be presented. Accordingly, the SWE are discretized in space and time. In the end the wetting and drying algorithm is discussed. After that special attention is paid to the discretization of the advective terms in Chapter 3. Following this several ways of solving the system are discussed extensively in Chapter 4. In Chapter 5.3 the mass conservation of the wetting and drying procedure and of the scheme is discussed. Finally the resulting procedure will be compared to several analytical solutions and some measurements and its performance with respect to mass conservation, number of iterations and similarity to the analytical solution will be investigated in Chapter 5.

1.3 Notation

The notation which is adopted in this report is explained below. A vector is denoted with a bold lower-case letter.

$$\mathbf{u} \in \mathbb{R}^n,\ \mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ \vdots \\ u_{n-1} \\ u_n \end{bmatrix} \tag{1.1}$$

A matrix is indicated with upper-case italics.

$$A \in \mathbb{R}^{m\times n},\ A = \begin{bmatrix} a_{1,1} & a_{1,2} & \dots & a_{1,n-1} & a_{1,n} \\ a_{2,1} & a_{2,2} & & & a_{2,n} \\ \vdots & & \ddots & & \vdots \\ a_{m-1,n} & & & a_{m-1,n-1} & a_{m-1,n} \\ a_{m,1} & a_{m,2} & \dots & a_{m,n-1} & a_{m,n} \end{bmatrix} \tag{1.2}$$

1.4 The wetting and drying problem

The transition from wet to dry is a contact discontinuity. Contact discontinuities are surfaces that separate zones of different density and temperature. A contact discontinuity is a transition layer across which there is no particle transport. The contact discontinuity separates the wet state from the dry state (discontinuous density). A proper numerical representation of a wetting and drying problem encompasses two things, positive water depths and momentum conservation in the vicinity of the contact discontinuity. In a numerical model for water movement this should be translated in mass conservation and a momentum conserving formulation of the nonlinear advective terms. The earlier mentioned wetting and drying procedure is applied to keep the water depths positive in a mass-conserving manner. To fulfill the second aspect of wetting and drying special attention is paid to the formulation of the advective terms.

1.4.1 Governing equations

The applications subject to this study are problems of which the horizontal length scales are much larger than the vertical length scales. This assumption allows for a depth averaged approach. The governing equations are the shallow water equations (SWE). The shallow water equations are the depth averaged Navier Stokes equations, which can be derived from conservation laws. Mass conservation and momentum conservation together give a set of differential equations, which suffice to describe the motion of water. In combination with a set of boundary conditions on the boundary of the domain of interest a problem is defined with a unique solution.

A domain Ω in $\mathbb{R}^1$ or $\mathbb{R}^2$ is considered with a boundary Γ. The water depth is defined as the distance between the free surface and the bottom level given by $H = h - z$ as shown in figure 1.1.

The depth averaged continuity equation is given by

$$\frac{\partial h}{\partial t} + \nabla \cdot (H\mathbf{u}) = 0, \text{ in } \Omega \tag{1.3}$$

and originates from mass conservation for incompressible flow $\left(\frac{D\rho}{Dt} = 0\right)$. It means that a change in the water level of the water column is caused by a discharge gradient over the water column. The depth averaged inviscid momentum equations are given by

$$H\frac{\partial \mathbf{u}}{\partial t} + gH\nabla h + c_f \left|\mathbf{u}\right| \mathbf{u} + H\mathbf{u} \cdot \nabla \mathbf{u} = 0, \text{ in } \Omega. \tag{1.4}$$

In equations (1.3) and (1.4) the velocity vector is denoted by $\mathbf{u}$, which reduces to a scalar in one dimension and the system of equations (1.4) reduces to a single equation.

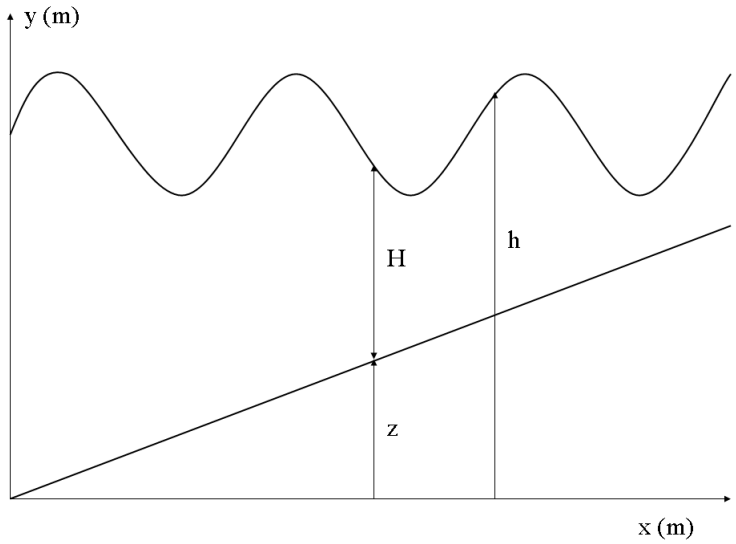


Figure 1.1: The bottom level and water surface level are measured positive above a certain reference level.

Neglecting diffusion significantly simplifies computations (solving one system instead of two as we will see later). Moreover the analytical solution of the test cases considered in Chapter 5 do not include diffusion. In the laboratory experiments diffusion is not necessarily negligible but that is a sacrifice we are willing to make for computational efficiency.

The boundary conditions needed to make the problem unique and well posed are given by equation (1.5) and (1.6).

$$h = g_1, \text{ on } \Gamma_1 \tag{1.5}$$

$$\mathbf{u} \cdot \mathbf{n} = g_2, \text{ on } \Gamma_2 \tag{1.6}$$

In two dimensions in addition to the above boundary conditions a tangential boundary condition is required on inflow boundaries which is given by

$$\mathbf{u} \cdot \mathbf{s} = g_3, \text{ on } \Gamma \text{ where } (\mathbf{u} \cdot \mathbf{n}) < 0 \tag{1.7}$$

where $\mathbf{n}$ is the outside unit normal vector, $\mathbf{s}$ is the unit tangent vector and $\Gamma_1 \cup \Gamma_2 = \Gamma$ and $\Gamma_1 \cap \Gamma_2 = \emptyset$.

In sections 2.3.1 to 2.3.3 we will see how equations (1.3) and (1.4) will be discretized and the boundary conditions (1.5) to (1.7) will be treated.

It should be noted that the form in which the momentum equation (1.4) is formulated, is not strictly momentum conserving. The momentum conserving formulation of the momentum equation is given in intermezzo 1.1. The choice for a non-conservative formulation does not seem to be very straightforward since it was emphasized in paragraph 1.4 that momentum conservation over the wet/dry interface is an important part of the wetting and drying problem. However the formulation in $\mathbf{q}$ exhibits its own problems with respect to wetting and drying. The discharge is given as $q = Hu$. When the water depths H go to zero this implicates that the velocities go to infinity near the wet/dry interface. Therefore the momentum conserving solution can often generate instabilities around the wet/dry interface. For this

reason the non-conserving formulation is chosen, in spite of of the fact that it will be more difficult to obtain a momentum conserving scheme.

Intermezzo 1.1. *The momentum conserving formulation is given by*

$$\frac{\partial \mathbf{q}}{\partial t} + \nabla \cdot \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) + gH \nabla z + c_f \left| U \right| \mathbf{u} = 0 \tag{1.8}$$

where I denotes the unit matrix. It can be proven that this is strictly conservative by assuming a domain Ω with a boundary Γ which is subdivided into several sub domains Ω_i with internal boundaries Γ_i. Integration over the domain and applying Gauss divergence theorem 2.1 results in

$$\begin{aligned} \int_\Omega \nabla \cdot \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) d\Omega &= \sum_i \int_{\Omega_i} \nabla \cdot \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) d\Omega \\ &= \sum_i \int_{\Gamma_i} \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) \cdot \mathbf{n} d\Gamma. \end{aligned} \tag{1.9}$$

It is not hard to see that all internal boundary integrals cancel out and that only the boundary integrals remain, resulting in

$$\int_\Omega \nabla \cdot \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) d\Omega = \int_\Gamma \left(\mathbf{q}\mathbf{u}^T + \frac{1}{2} g H^2 I \right) \cdot \mathbf{n} d\Gamma. \tag{1.10}$$

So the inflow in the domain must balance the outflow and no internal losses or production can occur.

1.4.2 Wetting and drying methods

In literature numerous methods can be found which deal with wetting and drying in various ways. In this section several methods are discussed to give an idea of various different methods without the intention of being complete.

One of the first wetting and drying techniques was based on putting up screens between cells. Today several well-known packages as Delft3D and ADCIRC still use screen-based techniques. ADCIRC is a finite element model developed by the US Army Corps of Engineers (USACE). Its wetting and drying technique turns on and of elements by putting removable barriers in the model. Elements connected to a dry node have their barriers up. In this way flow exchange between wet and dry elements is prevented. The time step is constraint since the wetting and drying speed cannot exceed one node per time step. A detailed description of the wetting and drying procedure of ADCIRC is given by Luettich and Westerink [18].

A more elegant way of implementing a wetting and drying procedure is to formulate the governing equations in such a way that negative water depths can not occur. Stelling and Zijlema [23] formulated a model in which the flow out of a control volume is based upon the water available in this control volume. Hence the outflow can never be more than the water available in this control volume thus preventing negative water depths in the control volume.

Van 't Hof and Vollebregt [11] use an artificial porosity function to allow a more gradual transition between dry and wet points. An artificial quantity is introduced, the pseudo-water level, which is, contrary to the true water level, free to drop below the bottom. Mass is

conserved, however an amount of water resides in the porous ground and as a result the ground is no longer impermeable and leakage can occur.

Casulli [8] developed a method in which the water depth is strictly positive while the surface elevation is allowed to attain negative values. In this way mass conservation is guaranteed and the propagation velocity of the wet/dry interface is not influenced by the positivity demand on the water depths. However, the resulting system of equations becomes nonlinear and has to be solved by an iteration procedure. Nevertheless Casulli's method is not completely new, it is a variation to a thin slot method or Preismann slot method. Thin slot methods assume a very thin water storing slot at the bottom of a cell that prevents the cell from becoming dry. This method is used amongst others by van der Molen [20].

Other types of techniques are thin layer techniques. A thin layer of water is maintained in dry elements but the velocities are set to zero. In this way mass is conserved, however momentum is not. This method is applied by Bunya *et al.* [6] and a comparable approach is applied by Luettich and Westerink [17]. Many other methods exist, some of them have a more theoretical foundation and some less.

In this study Casulli's method is chosen for further research. This is for its elegant formulation, ability to wet and dry without constraints on the time step and guaranteed mass conservation. In the next chapter the principle of Casulli's method will be explained in detail but first the finite element method will be introduced and the discretization of the SWE will be given accordingly.

Chapter 2

The finite element method

This chapter starts with a general introduction into the finite element method. It will proceed with the discretization of the SWE and conclude with an explanation of the wetting and drying algorithm. For an extensive description of the finite element method in general one is referred to Van Kan *et al.* [12]. For a more specific application to the SWE Pironneau [21] can be particularly useful.

The finite element method is a discretization technique for partial differential equations. Many different discretization techniques exist. Some of the most well known are listed below.

- *Finite difference techniques* use a Taylor series expansion to approximate the differential equation. The derivative expressions are replaced with approximately equivalent difference quotients. The domain of interest is split in regular sub domains.
- *Finite volume techniques* convert volume integrals, in a partial differential equation that contains a divergence term, to surface integrals. The states in control volumes are updated according to the in and outgoing fluxes. The domain of interest is split in regular sub domains: finite volumes.
- *Finite element techniques* can use any shape of sub domain and any order of discretization in space. With a weak formulation a discretized form of the differential equations is obtained.

The main advantage of the finite element method over the other methods is that it is well suited to be applied on irregular domains. However a drawback of the method is that the matrix structure of finite element problems is not as nicely organized as for finite volume and finite difference problems. In the last case the matrix contains five non-zero diagonals (for 2D problems) while in finite element problems the matrix is unstructured and often has a considerably larger bandwidth.

2.1 The weak form

In the finite element method the weak form of a partial differential equation (PDE) is considered. The problem is that derivatives have to be established of discrete functions. This difficulty is bypassed by moving the derivatives from the (non-smooth) discrete functions onto sufficiently smooth test functions. The discrete functions are then projected onto (sufficiently) smooth basis functions. The weak formulation of a PDE is obtained by multiplication with

a smooth (within certain bounds) test function, integration over the domain and the use of integration by parts, intermezzo 2.1. The result is an equation involving lower order derivatives, hence requiring less smoothness, LeVeque [15] p.27. The weak formulation is more easy to be satisfied since only global satisfaction of the PDE is required instead of point-wise satisfaction.

Intermezzo 2.1. *The divergence theorem of Gauss is given by*

$$\int_\Omega \nabla \cdot \mathbf{b}\, d\Omega = \int_\Gamma \mathbf{b} \cdot \mathbf{n}\, d\Gamma$$

where $\mathbf{n}$ *is the outward unit normal and* $\mathbf{b}$ *is a differentiable vector field on* Ω*.*

Integration by parts stems from the product rule of integration and is a rule that transforms the integral of multiplied functions into other integrals. For a scalar vector product this is given by

$$\int_\Omega \nabla \cdot (a\mathbf{b})\, d\Omega = \int_\Omega a\nabla \cdot \mathbf{b}\, d\Omega + \int_\Omega \nabla a \cdot \mathbf{b}\, d\Omega.$$

Together these rules can be combined into

$$\int_\Omega a\nabla \cdot \mathbf{b}\, d\Omega = \int_\Gamma a\mathbf{b} \cdot \mathbf{n}\, d\Gamma - \int_\Omega \mathbf{b} \cdot \nabla a\, d\Omega.$$

The same can be derived for a matrix vector product, $A\mathbf{b}$ *instead of the scalar vector product* $a\mathbf{b}$*.*

2.2 Galerkin methods

The problem can be further simplified by expressing the solution to the differential equation in terms of the test function ϕ. For the Laplace equation

$$\begin{cases} \nabla^2 \mathbf{c} = 0 \quad \text{on}\, \Omega \\ c|_\Gamma = 0 \end{cases} \tag{2.1}$$

the problem can be formulated as; find $c|_\Gamma = 0$ such that

$$\int_\Omega \nabla c \cdot \nabla \phi d\Omega = 0 \,\text{for all}\, \phi|_\Gamma = 0 \tag{2.2}$$

the solution can be represented by a linear combination of functions, the basis functions:

$$c^h(\mathbf{x}) = \sum_{j=1}^{n} c_j \phi_j(\mathbf{x}) \tag{2.3}$$

Where c^h denotes an approximation to the solution of the weak formulation. It is required that the basis functions $\phi_j(\mathbf{x})$ are linearly independent. The function c_0 must be chosen such that $c^h(\mathbf{x})$ satisfies the essential boundary conditions. It should be noted that $c^h \to c(\mathbf{x})$ as $n \to \infty$. The basis functions $\phi_j(x)$ are as smooth as demanded by the weak formulation of the differential equation. The resulting finite element representation of the weak form can be formulated as: Find the set of constants $\{c_1, ..., c_n\}$ such that

$$\sum_{j=1}^{n} \int_\Omega c_j \nabla \phi_j(\mathbf{x}) \cdot \nabla \phi_i(\mathbf{x}) d\Omega = 0,\, \forall\, i = 1, ..., n \tag{2.4}$$

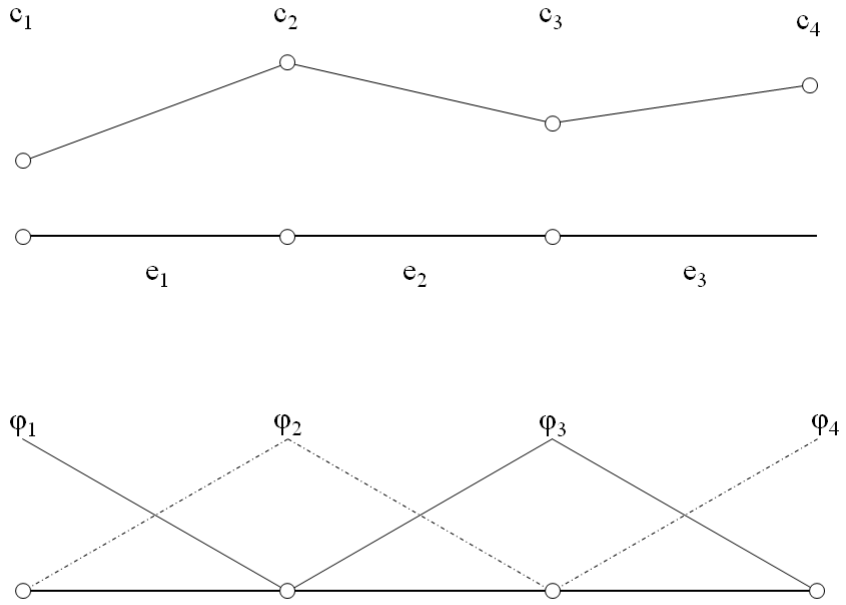


Figure 2.1: The discrete values c_j are represented piecewise linear by the basis functions ϕ_j according to the CG method.

2.2.1 Continuous Galerkin

The Continuous Galerkin (CG) method assumes that c^h is a continuous function. For $c^h = \sum_{j=1}^{n} c_j \phi_j$ to continuously represent the nodal values c_j the basis functions ϕ_j should have the value of 1 on node j and a zero value at all other nodes. In between ϕ can be described by a linear function, a quadratic function or even a higher order polynomial. This is graphically shown for a linear description of the basis functions in figure 2.1. The CG method does not allow for discontinuities as solution, while in an advection equation discontinuities are allowed. For this reason it is necessary to use upwind schemes to stabilize the solution of CG methods. A very common method is Streamline Upwind Petrov-Galerkin (SUPG). This method introduces artificial diffusion to stabilize the solution. This is done by adding an extra term to the test function, which may be discontinuous. For a more detailed description of the SUPG method one is referred to Van Kan *et al.* [12].

2.2.2 Discontinuous Galerkin

The Discontinuous Galerkin (DG) method allows c^h to be described by a discontinuous function and therefore discontinuous basis functions can be used. This results in multiple values per node, $c_{e,k}$, where e is the element index and $k = 1, ..., nn$ with nn the number of nodes of the element. Thus the approximate solution can be described by

$$c^h = \sum_{e=1}^{ne} \sum_{k=1}^{nn} c_{e,k} \phi_{e,k}. \tag{2.5}$$

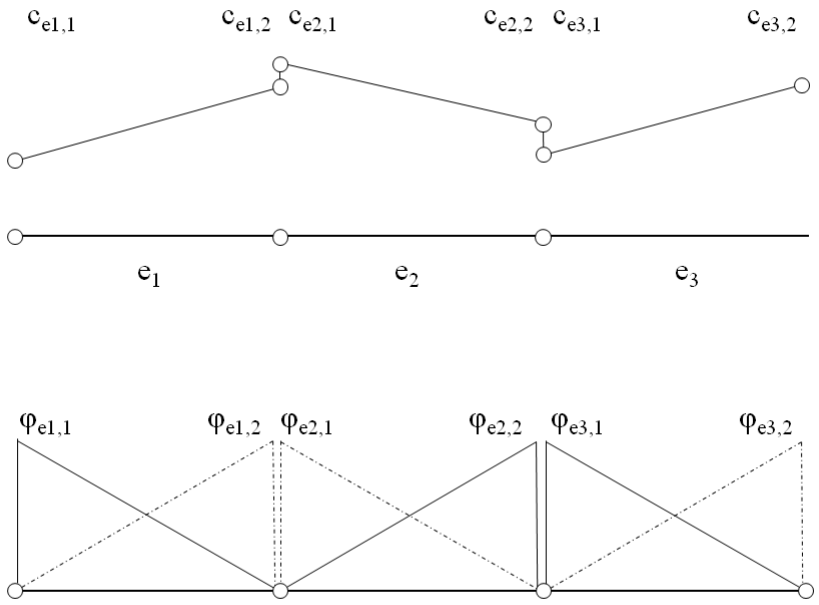


Figure 2.2: The discrete values c_j are represented piecewise linear by the basis functions $\phi_{e,k}$ according to the DG method.

This is illustrated in figure 2.2. By allowing a discontinuity in c on the element interfaces an extra degree of freedom is created in the gradient of c. For this reason the advective term behaves much more stable than in CG methods. However, as can be seen from equation (2.5) the total number of unknowns has changed from $ne + 1$ for CG methods to $ne \cdot nn$ for DG methods. This is obviously a severe drawback of DG methods.

Recently a method has been developed which combines the best of both methods. Discontinuous basis functions are allowed; and the extra computational effort is reduced to a minimum. For more information one is referred to Labeur [14].

2.3 Application of the finite element method to the instationary SWE

In this section the finite element discretization as discussed in the previous chapter will be applied to the SWE. First a semi-discretization in space will be derived and with the pressure correction method an uncoupled system is created. Thanks to the assumption of inviscid flow, it is allowed to use constant basis functions, resulting in an explicit velocity update. Next the implementation of the boundary conditions will be discussed. In the end the fully-discrete system is derived using the θ-method.

2.3.1 Discretization

The derivation of the discrete equation will be done for the two-dimensional equations. In one dimension the velocity vector $\mathbf{u}$ reduces to a scalar and the two dimensional discretization can be easily derived from the derivation given here. Equations (1.3) and (1.4) are multiplied with the test functions N^u respectively N^h. Subsequent integration over the domain Ω, application of Gauss divergence theorem and integration by parts results in the weak formulation. The problem can then be formulated as: find $\mathbf{u} \in \mathbb{R}^2$ and $H \in \mathbb{R}$ with

$$h = g_1, \text{ on } \Gamma_1 \qquad \mathbf{u}\cdot\mathbf{n} = g_2, \text{ on } \Gamma_2 \qquad \text{and} \qquad \mathbf{u}\cdot\mathbf{s} = g_3, \text{ on } \Gamma \tag{2.6}$$

such that

$$\int_\Omega \frac{\partial H}{\partial t} N^h d\Omega + \int_\Gamma H\mathbf{u}N^h\cdot\mathbf{n}d\Gamma - \int_\Omega H\mathbf{u}\cdot\nabla N^h d\Omega = 0 \tag{2.7a}$$

$$\int_\Omega H\frac{\partial \mathbf{u}}{\partial t}N^u d\Omega + \int_\Omega gH\nabla h N^u d\Omega + \int_\Omega c_f\left|\mathbf{u}\right|\mathbf{u}\,N^u d\Omega + \int_\Gamma H\mathbf{u}\cdot\mathbf{u}N^u\mathbf{n}\,d\Gamma - \int_\Omega \mathbf{u}\cdot\nabla\left(H\mathbf{u}N^u\right)d\Omega = 0 \tag{2.7b}$$

for all N^h, N^u such that $N^h, N^u = 0$ on Γ_1. In which $\mathbf{n}$ is the outside unit normal vector on an element boundary.

In equations (2.7) no derivatives of u and N^u are necessary, demanding u and N^u only to be integrable. First derivatives of H and N^h are necessary, demanding integrable derivatives too. This means that u and N^u can be taken piecewise constant while H and N^h have to be piecewise linear. Thus using a CG method for h and a DG method for u. The result is that both the favorable properties of CG and DG are combined. The discontinuous approach stabilizes the advection term. Yet since u is taken piecewise constant the number of independent variables is not increased in 1D and only slightly increased in 2D with respect to (pure) CG. Note that in case the diffusion term is not disregarded in the momentum equation, piecewise *linear* u and N^u would be necessary. As we will see later the use of piecewise constant u and N^u results in some favorable properties of the discrete system. With this choice u and H can be described by

$$1D: \qquad u(x) = \sum_{e_j=1}^{ne} u_{e_j}N^u_{e_j}(x) \quad \text{and} \quad h(x) = \sum_{j=1}^{nn} h_j N^h_j(x) \tag{2.8a}$$

$$2D: \quad \mathbf{u}(x,y) = \sum_{e_j=1}^{ne} u_{e_j}\mathbf{N}^u_{e_j u}(x,y) + v_{e_j}\mathbf{N}^u_{e_j v}(x,y) \quad \text{and} \quad h(x,y) = \sum_{j=1}^{nn} h_j N^h_j(x,y) \tag{2.8b}$$

$$\text{with } \mathbf{N}^u_{e_j u}(x,y) = \begin{pmatrix} N^u_{e_j}(x,y) \\ 0 \end{pmatrix} \quad \text{and} \quad \mathbf{N}^u_{e_j v}(x,y) = \begin{pmatrix} 0 \\ N^u_{e_j}(x,y) \end{pmatrix}.$$

In which ne the number of elements and nn the number of nodes. The basis function $N^u_{e_j}$ has value 1 on element e_j and 0 everywhere else and N^h_j has value 1 at node j and 0 at all other nodes, and varies linearly in between. The resulting values of $u(x)$ and $h(x)$ (1D case) are schematically shown in figure 2.3.

On continuity grounds we demand $q = uH$ to be equal on both sides of the element boundaries and since N^h is continuous over the element boundary, the boundary integral in equation (2.7a) cancels out on internal elements.

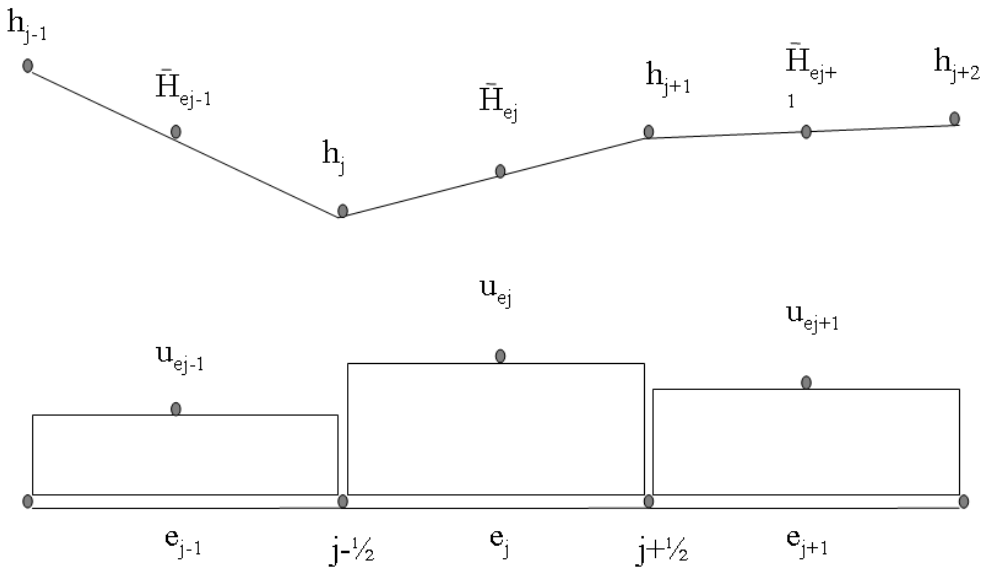


Figure 2.3: Interpolations of the discrete values u_j and h_j on the element domains.

By inserting the approximations of u and H given by equation (2.8b) and writing the definition of $\mathbf{u}$ as $\mathbf{u}(x,y) = \sum_{e_j=1}^{ne} \mathbf{u}_{e_j} N_{e_j}^u(x)$ for readability, the discrete problem can be written as

$$\int_\Omega \frac{\partial H_j}{\partial t} N_j^h N_i^h d\Omega + \int_{\Gamma_2} g_2 \bar{H}_{e_j} N_i^h d\Gamma - \int_\Omega \bar{H}_{e_j} N_{e_j}^u \mathbf{u}_{e_j} \cdot \nabla N_i^h d\Omega = 0, \qquad \forall\, i,j = 1,...,nn, \text{ and } e_j = 1,...,ne \tag{2.9a}$$

$$\int_\Omega \bar{H}_{e_j} \frac{\partial \mathbf{u}_{e_j}}{\partial t} N_{e_j}^u N_{e_i}^u d\Omega + \int_\Omega g \bar{H}_{e_j} h_j \nabla N_j^h \, N_{e_i}^u d\Omega + \int_\Omega c_f \mathbf{u}_{e_j} \left|\mathbf{u}_{e_j}\right| N_{e_j}^u N_{e_i}^u d\Omega + \int_\Gamma (\mathbf{u}\cdot\mathbf{n})^- H g_3 + ADV = 0, \qquad \forall\, j = 1,...,nn, \text{ and } e_j, e_i = 1,...,ne, \tag{2.9b}$$

with $\bar{H}_{e_j} = \frac{H_j + H_{j+1}}{2}$. The last two terms of equation (2.7b) are denoted here by ADV and will be elaborated further in Chapter 3. Equations (2.9) can be written as a semi-discrete system of equations, equation (2.10)[1].

$$\begin{aligned} M^H \frac{\partial \mathbf{H}}{\partial t} - D^H \mathbf{u} &= 0 \\ M^u \frac{\partial \mathbf{u}}{\partial t} + g D^u \mathbf{h} + F\mathbf{u} + ADV &= 0 \end{aligned} \tag{2.10}$$

In which the matrices are defined by

$$M_{ij}^H = \int_\Omega N_i^h N_j^h d\Omega \qquad M_{ij}^u = \int_\Omega N_{e_i}^u N_{e_j}^u d\Omega$$

$$D_{ij}^H = \int_\Omega \bar{H}_{e_j} N_{e_i}^u \frac{\partial N_j^h}{\partial x} d\Omega \qquad D_{ij}^u = \int_\Omega \nabla N_i^h N_{e_j}^u d\Omega$$

$$F_{ij} = \int_\Omega \frac{c_f \left|U_{e_j}\right|}{\bar{H}_{e_j}} N_{e_i}^u N_{e_j}^u d\Omega$$

The boundary integral in equation (2.9a) is not considered in the matrix vector formulation (2.10). Its treatment is discussed in the next paragraph. Equation (2.10) gives the semi-discrete system of equations. To calculate the evolution of the solution in time the equations still have to be discretized in time. The fully discrete equations will be presented at the end of this chapter.

2.3.2 Boundary conditions

Two categories of boundary conditions can be distinguished: essential and natural boundary conditions. Dirichlet boundary conditions belong to the first category, they prescribe the value of a certain quantity at the boundary. Neumann boundary conditions belong to the second category, they prescribe the value of a certain flux at the boundary. Dirichlet boundary conditions are called essential, since they should be satisfied explicitly. Neumann boundary conditions are called natural, since they are implicitly satisfied by the formulation of the

[1] This equation is now divided by H. For readability purposes this is taken into account for the advective term in Chapter 3 and not denoted here as $\frac{ADV}{H}$

equations. Dirichlet boundary conditions have to be imposed upon the solution. The first boundary condition in equation (2.6) is a Dirichlet boundary condition and the other two are Neumann boundary conditions.

In equation (2.9) the Neumann boundary conditions are substituted in the equations. The Dirichlet boundary conditions have to be imposed on the system. This is done by putting the row of the boundary element concerned in the system matrix to zero, the diagonal to one and the corresponding entry of the RHS vector to g_1.

2.3.3 Time discretization

The semi-discrete system of equations as derived earlier will be discretized in time. For the discretization in time the θ-method and the pressure-correction method are used. The θ-method is given by

$$\begin{aligned} M^u \frac{\mathbf{u}^{n+\theta} - \mathbf{u}^n}{\theta \Delta t} + g D^u \mathbf{h}^{n+\theta} &+ F^n \mathbf{u}^{n+\theta} \\ &+ ADV^{n+\theta} = 0, \qquad (2.11) \\ M^H \frac{\mathbf{H}^{n+\theta} - \mathbf{H}^n}{\theta \Delta t} &+ D^{h^n} \mathbf{u}^{n+\theta} = 0 \qquad (2.12) \end{aligned}$$

Where θ is an implicitness coefficient, $0 \leq \theta \leq 1$.

M^u and F^n are diagonal matrices[2], however the advective term is not. Implicit time stepping is often advantageous because of its unconditional stability. However, when the advective term is taken explicit, $\theta = 0$, $\mathbf{u}^{n+1}$ can be solved directly from equation (2.12) without matrix inversion. If all other terms are taken implicit, $\theta = 1$, this results in

$$\begin{aligned} \left[M^u + \Delta t F^n\right] \mathbf{u}^{n+1} &= M^u \mathbf{u}^n - g \Delta t D^u \mathbf{h}^{n+1} \\ &- \Delta t ADV^n. \qquad (2.13) \end{aligned}$$

The first and the last term of the RHS of equation (2.13) depend only on the known time level t^n. This means that a preliminary estimate of $\mathbf{u}^{n+1}$ can be made based on these two values. This value is denoted by $\mathbf{u}^*$ and is expressed by equation (2.14).

$$\mathbf{u}^* = \mathcal{B}^n M^u \mathbf{u}^n - \Delta t \mathcal{B}^n ADV^n \qquad (2.14)$$

With

$$\mathcal{B}^n = \left[M^u + \Delta t F^n\right]^{-1}$$

which is easily calculated since M^u and F^n are diagonal matrices. The expression for $\mathbf{u}^{n+1}$ is reduced to

$$\mathbf{u}^{n+1} = \mathbf{u}^* - g \Delta t \mathcal{B}^n D^u \mathbf{h}^{n+1}. \qquad (2.15)$$

The fully discrete formulation of the continuity equation is given by

$$M^H \mathbf{H}^{n+1} = M^H \mathbf{H}^n + \Delta t D^{H^n} \mathbf{u}^{n+1}. \qquad (2.16)$$

Inserting equation (2.15) in equation (2.16) and using $\mathbf{H}^n = \mathbf{h}^n - \mathbf{z}$ results in

$$M^H \mathbf{H}^{n+1} + g \Delta t^2 \mathcal{B}^n D^{H^n} D^u \mathbf{h}^{n+1} = M^H \mathbf{H}^n + \Delta t D^{H^n} \mathbf{u}^*. \qquad (2.17)$$

[2]Using piecewise constant basis functions for u results in this favorable feature of M^u.

Thereafter the velocity at time level t^{n+1} can be calculated by substitution of $\mathbf{h}^{n+1}$ and $\mathbf{u}^*$ in equation (2.15). By inserting the definition for $\mathbf{u}^*$ in equation (2.14) the complete system of equations can be written as

$$\begin{aligned} \underbrace{M^H}_{P}\mathbf{H}^{n+1} + \underbrace{g\Delta t^2 \mathcal{B}^n D^{H^n} D^u}_{T}\mathbf{h}^{n+1} = \\ \underbrace{M^H\mathbf{H}^n + \Delta t D^{H^n}\mathcal{B}^n M^u \mathbf{u}^n - \Delta t^2 D^{H^n}\mathcal{B}^n ADV^n}_{\mathbf{b}}. \end{aligned} \tag{2.18}$$

The above equation still contains two unknowns at time level t^{n+1}, $\mathbf{H}^{n+1}$ and $\mathbf{h}^{n+1}$. Although $\mathbf{H}^{n+1}$ and $\mathbf{h}^{n+1}$ can be expressed into one another this is not done yet. This is one of the main steps in the wetting and drying procedure as we will see in the next chapter.

2.4 Wetting and drying algorithm

The wetting and drying algorithm in this chapter follows the same reasoning as the algorithm presented by Casulli in [8]. Casulli has successfully implemented his algorithm in a finite volume discretization. He combines a mass conservative method for positive water depths with a Lagrangian advection scheme. Here an implementation of his method for positive water depths in a finite element discretization is derived. In the next chapter several flux splitting methods for the advective terms will be discussed.

2.4.1 Nonlinear system

The wetting and drying procedure is based on the idea that the water depth *cannot* become negative while the surface elevation on the other hand *is* able to become negative. Therefore it should not be regarded as a surface elevation but as a mathematical multiplier that influences the water level gradient. The positivity of the water depth means that the water depth can be defined by equation (2.19), but it can also be implemented by using a function $c(h_j)$ described by equation (2.20). This results in a definition of the water depth given by equation (2.21).

$$H_j^n = max\left[0, h_j^n - z_j\right] \tag{2.19}$$

$$c(h_j^n) = \begin{cases} 1 & \text{if } h_j > z_j \\ 0 & \text{if } h_j \leq z_j \end{cases} \tag{2.20}$$

$$H_j^n = c(h_j^n)\left[h_j^n - z_j\right] \tag{2.21}$$

When incorporating the first definition for H_j^n and the second definition for H_j^{n+1} in equation (2.18) we obtain the following nonlinear equation:

$$\begin{aligned} \left[M^H C(\mathbf{h}^n) + g\Delta t^2 \mathcal{B}^n D^{H^n} D^u\right]\mathbf{h}^{n+1} &= M^H C(\mathbf{h}^n)\mathbf{z} + M^H\mathbf{H}^n \\ &\quad + \Delta t D^{H^n}\mathcal{B}^n M^u\mathbf{u}^n - \Delta t^2 D^{H^n}\mathcal{B}^n ADV^n \end{aligned} \tag{2.22}$$

With $C(\mathbf{h}^n)$ the diagonal matrix with entries $c_{jj} = c(h_j^n)$ and $\mathbf{H}^n$ a vector with entries $H_j^n = max\left[0, h_j^n - z_j\right]$. Or in matrix vector notation as

$$\left[P(\mathbf{h}) + T\right]\mathbf{h} = P(\mathbf{h})\mathbf{z} + \mathbf{b}, \tag{2.23}$$

with P, T and b as defined in the previous chapter.

2.4.2 Finite Elements vs. Finite Volumes

For finite volumes, as described by Casulli [8], P is a diagonal matrix with positive entries. T has positive diagonal elements, negative off-diagonal elements and the sum of the elements over the row is exactly zero. Comparing these properties to the properties described in intermezzo 2.2 one can conclude that $A = P(\mathbf{h}) + T$ is a K-matrix. In combination with the irreducibility demand Casulli points out, A is an M-matrix and the existence of A^{-1} is guaranteed.

Intermezzo 2.2. *A matrix is called an M-matrix if A is nonsingular,*

$$\begin{aligned} A^{-1} &\geq 0, && \text{and} \\ a_{ij} &\leq 0, && \forall\, i,j = 1,...,n \text{ and } i \neq j. \end{aligned}$$

With $A^{-1} \geq 0$ is meant that all entries of the inverse matrix of A are larger than or equal to zero.

A matrix A is called a K-matrix if

$$\begin{aligned} a_{ii} &> 0, && i = 1,...,n, \\ a_{ij} &\leq 0, && i,j = 1,...,n, j \neq i \\ & && \text{and} \\ \sum_j a_{ij} &\geq 0, && i = 1,...,n. \end{aligned}$$

with strict inequality for at least one i.

A matrix is called irreducible if the system does not consist of independent subsystems. An irreducible K-matrix is an M-matrix, hence its inverse matrix exists [27]. Yet not all M-matrices are K-matrices. Consequently these properties are sufficient to prove existence of A^{-1}, yet not necessary to prove existence of A^{-1}.

For finite elements in 1D P is a tridiagonal matrix with strictly positive entries. T is a tridiagonal matrix with positive diagonal elements and negative off-diagonal elements. It is possible for A to fulfill the requirements described in intermezzo 2.2 however severe constraints will be posed on the minimum value of $\bar{H}_j^n$. For the off-diagonal terms of A to be negative, the following condition should be fulfilled:

$$\frac{1}{6}\frac{\Delta x}{\Delta t} \leq \alpha_{e_i}^n \frac{\Delta t}{\Delta x} g \bar{H}_{e_j}^n \quad \forall\, e_j.$$

After some manipulations (and assuming the friction to be negligible) the Courant-number can be recognized in this expression and this constraint transforms in

$$\frac{1}{6} \leq CFL^2.$$

Where the Courant-number is expressed as

$$CFL = \frac{\Delta t}{\Delta x} c, \text{ with } c = \sqrt{g \bar{H}_{e_j}^n}.$$

For a grid size $\Delta x = 1$ and a time step size $\Delta t = 0.1$ this means that $\bar{H}_{e_j}^n \geq 1.67 \forall\, j$ which is clearly impossible in case of wetting and drying. In general it is easily seen that when

$\bar{H}_{e_j} \to 0$, $\frac{\Delta x}{\Delta t}$ has to become very small to fulfill the constraint above. This is a condition that conflicts with accuracy and stability demands. However, this does not mean that A^{-1} does not exist but that its existence is not guaranteed and there are cases in which problems can arise with respect to this. This will be explained further in Chapter 4.

Chapter 3

Discretization of the advective terms

The advective terms can be discretized in many ways. The terms denoted with ADV in the previous chapter will be elaborated further here. The discontinuous velocity approximation makes it logical to use a Riemann type of method since Riemann solvers look at the propagation of discontinuities. They poses the advantageous property of using the information out of the direction that it travels from, even if there are multiple waves traveling in multiple directions. Sections 3.2 to 3.4 encompass the one-dimensional advection. In two dimensions another approach than for one dimension is because the one-dimensional approach would result in very complicated cross terms. The two-dimensional approach will be discussed in sections 3.6 and 3.7.

3.1 Discretization

The advective terms of the previous chapter are given by

$$ADV = \int_{\Gamma} \mathbf{u} \cdot \mathbf{u} N^u \mathbf{n} \, d\Gamma - \int_{\Omega} \mathbf{u} \cdot \nabla \left(\mathbf{u} N^u \right) d\Omega. \tag{3.1}$$

It might not seem straightforward to apply integration by parts to this term since the resulting integral still contains a derivative in $\mathbf{u}$. However by choosing the basis function for $\mathbf{u}$ piecewise constant the gradient of $\mathbf{u}$ would be zero and the advective term would disappear. By using integration by parts a new term appears which describes the incoming and outgoing fluxes in an element, making it possible to describe the velocity with piecewise constant basis functions. Next the discrete values for the water level and the velocity will be written as

$$\mathbf{u}(x) = \sum_{e_j=1}^{ne} \mathbf{u}_{e_j} N^u_{e_j}(x) \qquad \text{and} \qquad h(x) = \sum_{j=1}^{nn} h_j N^h_j(x)$$

and assuming piecewise constant basis functions for $\mathbf{u}$ the advective term becomes

$$ADV = \int_{\Gamma} N^u_{e_i} N^h_j \mathbf{u} \cdot \mathbf{n} \mathbf{u}_j \, d\Gamma \tag{3.2}$$

Or in matrix vector notation

$$ADV = \tilde{A} \mathbf{u} \tag{3.3}$$

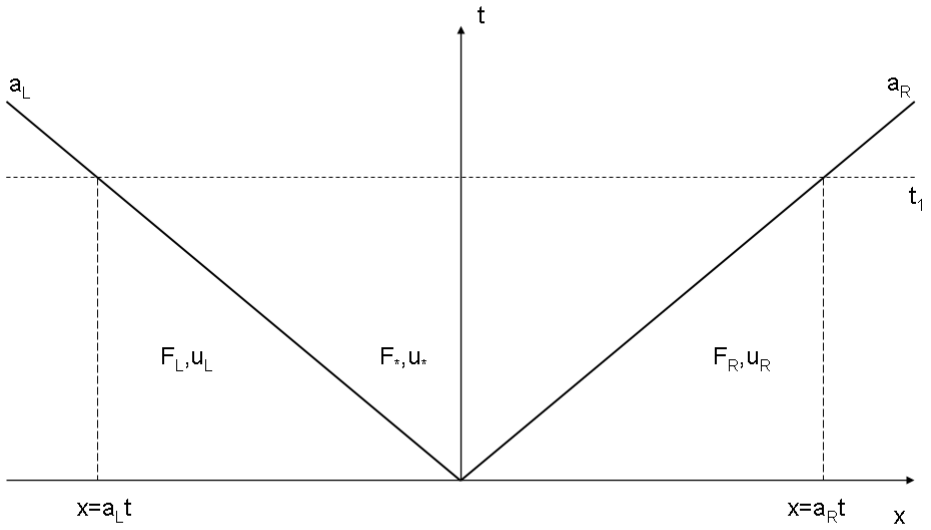


Figure 3.1: The different states of the solution to the Riemann problem.

with

$$\tilde{A}_{ij} = \int_{\Gamma} \mathbf{u} N_i^u N_j^h \mathbf{n} \, d\Gamma. \tag{3.4}$$

3.2 Riemann formulation

In one dimension the Riemann problem for the equations presented in section 1.4.1 is an initial-value problem described by

$$\frac{\partial h}{\partial t} + \frac{\partial}{\partial x}\left(F(u)\right) = 0 \tag{3.5}$$

with initial condition

$$U(x,0) = \begin{cases} U_L, \; x < 0 \\ U_R, \; x > 0. \end{cases} \tag{3.6}$$

The solution consists of two waves separating three states: the left state, the right state and the middle state. The last one is here denoted with a $*$, see figure 3.1.

If one looks at figure 3.2, the disturbance of the quantity U at the interface $j-\frac{1}{2}$ propagates with a speed a_L at the left side of the interface and with a speed a_R on the right side of the interface. If we consider only the right side of the interface, the disturbance $U_R - U_*$ travels with speed a_R where U_* is the state at the interface. The same holds at the left side of the interface. This results in

$$\begin{aligned} a_R \left(U_R - U_*\right) &= F_R - F_* \\ a_L \left(U_* - U_L\right) &= F_* - F_L. \end{aligned} \tag{3.7}$$

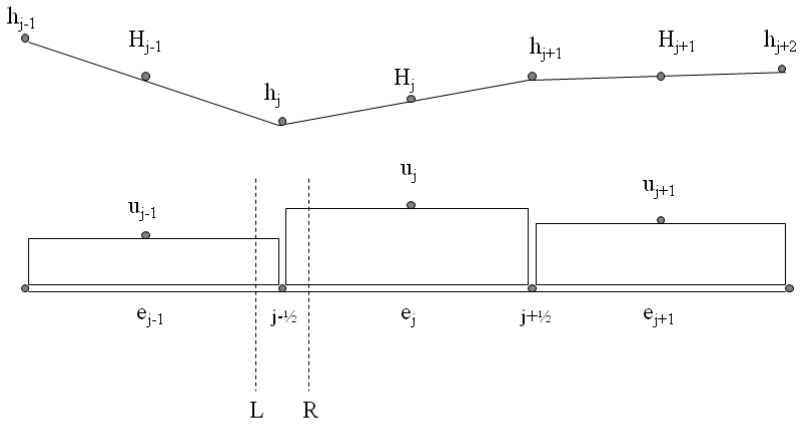


Figure 3.2: The different states to the left and to the right side of interface $j - \frac{1}{2}$

By multiplication of the first equation with a_L and the second with a_R and subtracting both, the interface flux F_* can be written as

$$F_* = \frac{a_R a_L (U_R - U_L) - F_R a_L + F_L a_R}{a_R - a_L} \tag{3.8}$$

With the non-conservative formulation of the equations as described in the previous chapter the fluxes are given by $F_L = \frac{1}{2} u_L u_L$ and $F_R = \frac{1}{2} u_R u_R$. The difference between different Riemann solvers is the choice of an appropriate definition of the propagation speeds, a_L and a_R.

In the next section more thought is given to different choices of the propagation speed a.

3.3 Choices of the propagation speed

The propagation speeds are given by the eigenvalues of the system (2.23). For a large matrix it can be quite laborious to determine the eigenvalues of this system. Obviously it would be convenient if it would not be necessary to calculate these eigenvalues. There is a range of estimates that can be made for a_L and a_R. A straightforward choice for a_L and a_R is $a_L = u_{e_{j-1}}$ and $a_R = u_{e_j}$. The eigenvalues of a simplified system could also serve as an approximation. The eigenvalues of the system

$$\begin{bmatrix} H \\ uH \end{bmatrix}_t + \begin{bmatrix} Hu \\ Hu^2 + \frac{1}{2} g H^2 \end{bmatrix}_x = 0 \tag{3.9}$$

can be approximated by writing the system (3.9) in the quasi-linear form $q_t + f'(q) q_x = 0$. The eigenvalues of the quasi-linear system are given by $\lambda_1 = u - \sqrt{gH}$ and $\lambda_2 = u + \sqrt{gH}$.

This could be discretized as $a_L = u_{e_{j-1}} - c_L$ and $a_R = u_{e_j} + c_R$. Furthermore it is possible to use a higher order estimate of the left and right interface velocities. This will be discussed in section 3.5.

3.4 Roe Linearization

It is also possible to circumvent the need for finding the eigenvalues by solving the Riemann problem approximately. Approximate Riemann solvers can give good results with much less effort. That is, the nonlinear flux term $f(u)_x$ will be replaced by a linearized term $\hat{A}_{j-\frac{1}{2}}\hat{u}_x$ defined at each cell interface. The matrix $\hat{A}_{j-\frac{1}{2}}$ is an approximation if $f'(u)$ at the cell interface. Combining this linearization with an upwind scheme, the interface flux at $j-\frac{1}{2}$ can be calculated as

$$\begin{aligned} F^* &= F_L + \hat{A}^-_{j-\frac{1}{2}}\left(u_{e_j} - u_{e_{j-1}}\right) \\ F^* &= F_R - \hat{A}^+_{j-\frac{1}{2}}\left(u_{e_j} - u_{e_{j-1}}\right). \end{aligned} \tag{3.10}$$

Averaging these expressions results in

$$F^* = \frac{1}{2}\left[F_{j-1} + F_j\right] - \frac{1}{2}\left|\hat{A}_{j-\frac{1}{2}}\right|\left(u_{e_j} - u_{e_{j-1}}\right) \tag{3.11}$$

which is often referred to as Roe's method.

3.5 Second Order Advective terms

Instead of approximating the advective flux immediately to the left of the interface based on the velocity left of the interface it is also possible to make an approximation of this velocity using linear interpolation. In figure 3.3 this approach is visualized, the values of u_R and u_L are indicated with black dots. A gradient in u is established, which is used to calculate an interface value of u.

A drawback of this method is that it is possible to calculate an interface value that is higher or lower than the velocities to the left and the right of the interface, as is u_R at the $j-\frac{1}{2}$ interface in figure 3.3. In this way a new local extrema is created. For this reason second order schemes always generate artificial wiggles. A simple solution to this problem is to say that in case the extrapolated value is outside the range of $u_{e_{j-1}}$ and u_{e_j} a first order estimate will be used. Several methods are illustrated in the remainder of this chapter based on this principle.

It should be noted that the limiters described in this chapter are no standard finite element method routines. However, the use of a piecewise constant basis function for the velocity results in lots of similarities between this finite element discretization and finite volume methods. This enables the use of these type of limiters, which are very common in finite volume computing, for finite element computations.

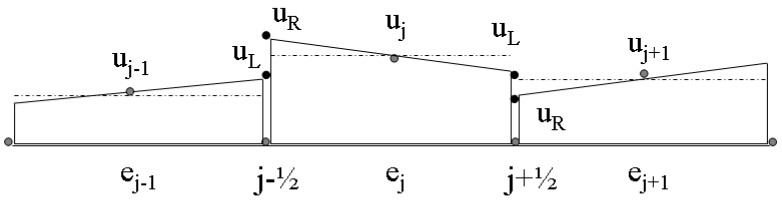


Figure 3.3: Visualisation of second order advective flux terms.

Minmod limiter

The minmod limiter is given by equation (3.12).

$$minmod\left(\Delta^+, \Delta^-\right) = \begin{cases} \Delta^+ & \text{if } |\Delta^+| < |\Delta^-| \text{ and } \Delta^+\Delta^- > 0, \\ \Delta^- & \text{if } |\Delta^-| < |\Delta^+| \text{ and } \Delta^+\Delta^- > 0, \\ 0 & \text{if } \Delta^+\Delta^- < 0. \end{cases} \tag{3.12}$$

$$\text{With } \Delta^+ = u_{e_{j+1}} - u_{e_j}$$
$$\Delta^- = u_{e_j} - u_{e_{j-1}}$$

The essence of the minmod limiter is sketched in figure 3.4. When the jump to the left and the jump to the right of element e_j are equal of sign, the gradient that results in a value closest to u_{e_j} is used for the higher order approximation, as indicated by the gray dots. In this way it is not possible to use an estimate that is outside the range of u_{e_j} and $u_{e_{j+1}}$. When the jumps are unequal of sign the first order approximation is used to prevent a higher order estimate that generates a local maximum, as shown in the lower part of figure 3.4.

Koren limiter

The limiter as described by Koren [13] is given in equation (3.13). By using this method the interface velocities will always be given by a first order upwind velocity or a second order

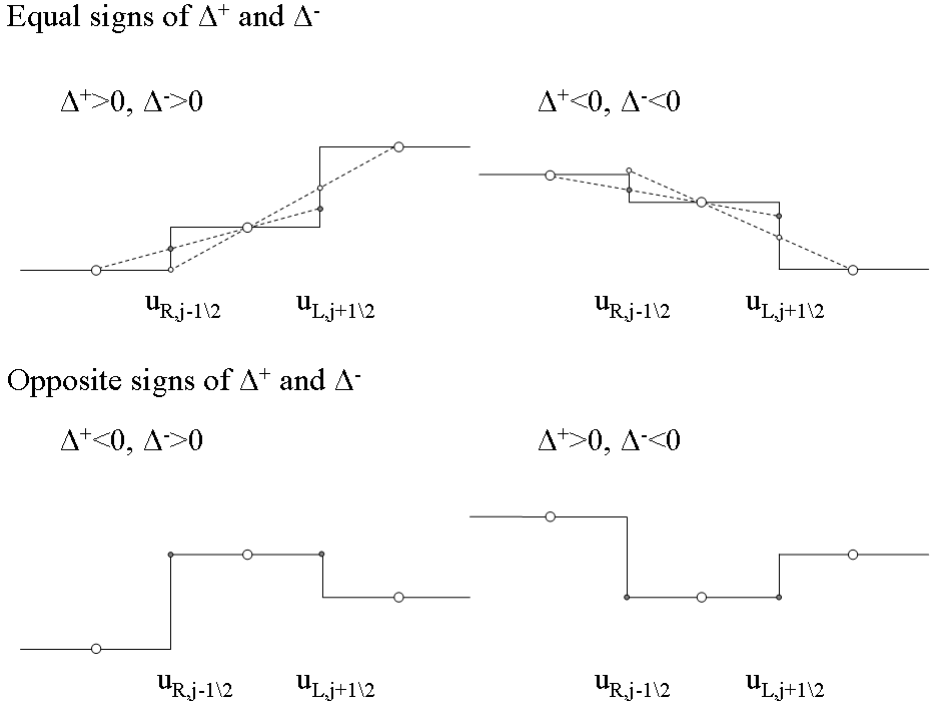


Figure 3.4: The basic principles of the minmod limiter.

interpolation which will be within the range of u_{e_j} and $u_{e_{j+1}}$.

$$\begin{aligned} r_j &= \frac{u_{e_{j+1}} - u_{e_j}}{u_{e_j} - u_{e_{j-1}}} \\ \phi(r_j) &= \max\left(0, \min\left(2r, \min\left(\frac{1}{3} + \frac{2}{3}r, 2\right)\right)\right) \\ u_{L,j+\frac{1}{2}} &= u_{e_j} + \frac{1}{2}\phi\left(r_j\right)\left(u_{e_j} - u_{e_{j-1}}\right) \\ u_{R,j+\frac{1}{2}} &= u_{e_{j+1}} - \frac{1}{2}\phi\left(r_{j+1}\right)\left(u_{e_{j+1}} - u_{e_j}\right) \end{aligned} \tag{3.13}$$

3.6 Advective terms in two dimensions

The extension of the previously used methodology into two dimensions is not as straightforward as one might think. In two dimensions the advective fluxes contain cross terms, $v\frac{\partial u}{\partial y}$ and $u\frac{\partial v}{\partial x}$. The fluxes do not necessarily work in the direction perpendicular to the interface anymore. For this reason another approach is chosen. In two dimensions we will look at flux differences, here denoted by ΔF.

In figure 3.5 a cross-section is taken perpendicular to the cell interface. As can be seen the flux differences can be split in the following manner:

$$\begin{aligned} \Delta\mathbf{F}_R &= a_R\left(\mathbf{u}_R - \mathbf{u}_*\right) \\ \Delta\mathbf{F}_L &= a_L\left(\mathbf{u}_* - \mathbf{u}_L\right) \\ \Delta\mathbf{F}_M &= \tilde{u}\left(\mathbf{u}_R - \mathbf{u}_L\right) \end{aligned}$$

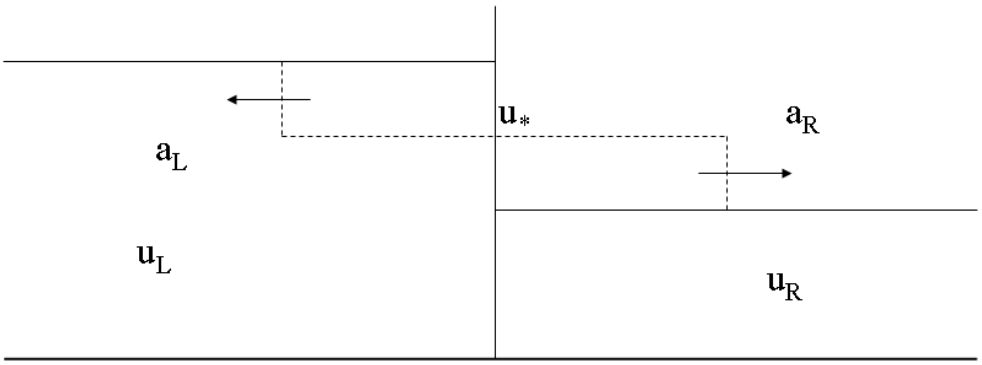


Figure 3.5: Left traveling wave and right traveling wave.

With $\Delta\mathbf{F}_R + \Delta\mathbf{F}_L = \Delta\mathbf{F}_M$ this results in

$$a_R\left(\mathbf{u}_R - \mathbf{u}_*\right) + a_L\left(\mathbf{u}_* - \mathbf{u}_L\right) = \tilde{u}\left(\mathbf{u}_R - \mathbf{u}_L\right). \tag{3.14}$$

This gives a formulation for the middle state $\mathbf{u}_*$:

$$\mathbf{u}_* = \frac{\tilde{u}\left(\mathbf{u}_R - \mathbf{u}_L\right) - a_R\mathbf{u}_R + a_L\mathbf{u}_L}{a_L - a_R} \tag{3.15}$$

Inserting this in the equations for the left and right flux differences ΔF_L and ΔF_R gives:

$$\begin{aligned} \Delta\mathbf{F}_R &= a_R\left(\frac{\tilde{u} - a_L}{a_R - a_L}\right)\left(\mathbf{u}_R - \mathbf{u}_L\right) \\ \Delta\mathbf{F}_L &= a_L\left(\frac{\tilde{u} - a_R}{a_L - a_R}\right)\left(\mathbf{u}_R - \mathbf{u}_L\right) \end{aligned} \tag{3.16}$$

Opposed to the one-dimensional formulation this advective scheme is not strictly conservative. In one dimension all contributions to the advective term were canceled out on internal elements. This formulation directly calculates the difference between the inflow and outflow, although for $\Delta x \to 0$ this is conservative, due to the finite size of the elements small differences can occur. Nevertheless no problems regarding the non-strict conservation of the advection will be encountered.

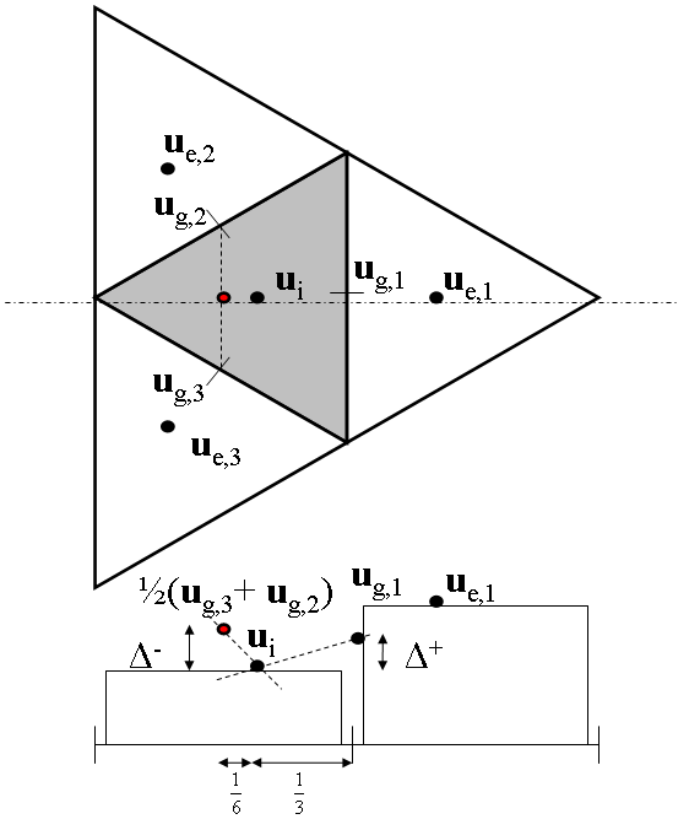


Figure 3.6: Establishing of velocity gradients in two dimensions.

3.7 Second order advective term in two dimensions

Following the same approach as in one dimension it is possible to derive a second order advective scheme in two dimensions. A gradient in the velocity is established by looking at the interface velocities of an element. For this purpose, the interface velocity is assumed to be the average velocity of the two adjacent elements. For an element i with its neighbor elements e, k with $k = 1, 2, 3$ as shown in figure 3.6. The interface velocity becomes

$$u_{g,k} = \frac{u_i + u_{e,k}}{2}. \tag{3.17}$$

If we consider interface 1 the forward difference is easily established as the difference between the element velocity and the velocity at interface 1. The backward difference is a little less straightforward. However, if an average is taken of the two remaining interface velocities, a velocity on the same line as the element velocity and the velocity at interface 1 is established. This is indicated with the red dot in figure 3.6. This velocity can then be used to establish the backward difference. It should be noted that the backward difference is taken over $\frac{1}{6}$ of the element height in that direction whilst the forward difference is taken over $\frac{1}{3}$ of the element height in that direction. With these backward and forward differences the minmod limiter can be applied in the same way as described in section 3.5, resulting in a (limited)

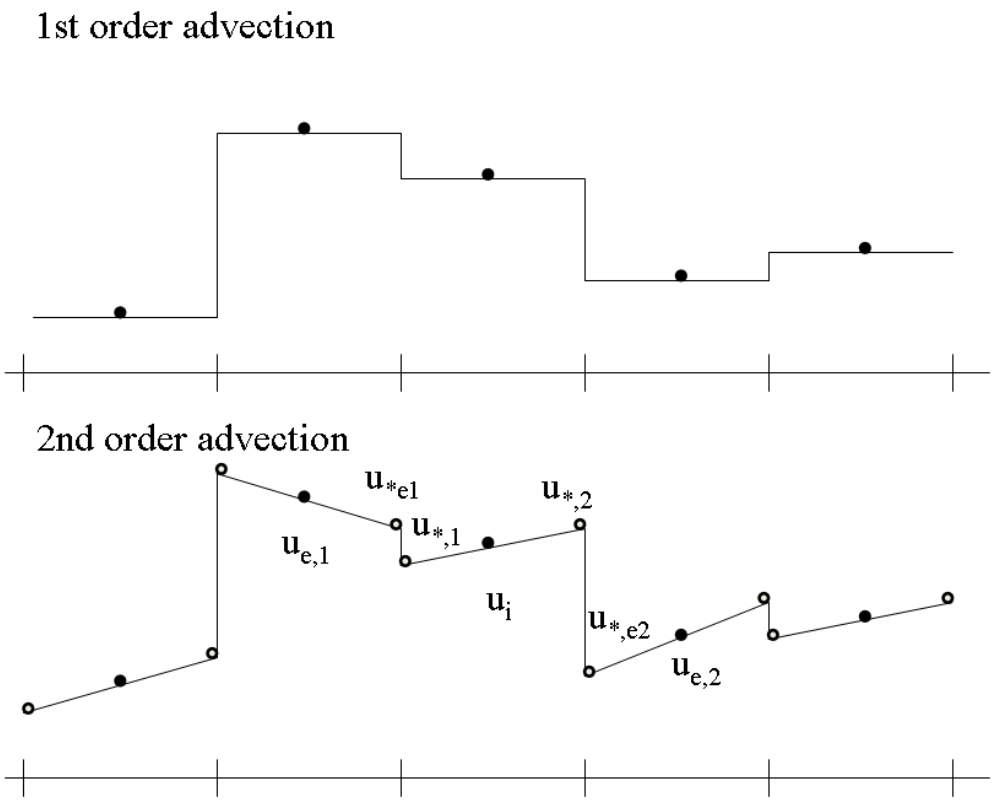


Figure 3.7: The difference between first order advection and second order advection in two dimensions (visualized on a one-dimensional grid).

second order estimation at the interface, u_*. Here the interface velocity is denoted with a * but should not be confused with the velocity estimate in the pressure correction method.

The next step is to add this to the equations. With respect to the first order terms we discussed before now a gradient over the element is established, see figure 3.7. This means that we cannot simply take the flux difference at the boundary to describe the advection but we need to consider the gradient over the element too. This results in an extra term in the advective flux:

$$\int_\Omega \mathbf{u}_i \cdot \nabla \mathbf{u}_{*,k} d\Omega + \int_\Gamma (\tilde{\mathbf{u}} \cdot \mathbf{n})^- \cdot [\mathbf{u}_{*,ek} - \mathbf{u}_{*,k}] \, d\Gamma \tag{3.18}$$

Where in first order only the second term was applied, since the first term was zero because of the constant velocity over an element. The above can be discretized as

$$-\mathbf{u}_i \cdot 2D\mathbf{u}_{*,k} + \sum_{e=1}^{k} \min\left(B\tilde{\mathbf{u}} \cdot \mathbf{n}, 0\right) \cdot \left[\mathbf{u}_{*,ek} - \mathbf{u}_{*,k}\right]. \tag{3.19}$$

The number two appears in the first term because u_* is a value in the middle of the interface and not at the nodes.

Chapter 4

Solution procedure

4.1 Nonlinear iteration procedure

Because of the nonlinear time derivative of the water level, the system derived in Chapter 2.4 must be solved with an iteration procedure. Casulli [8] uses a Newton iteration to solve the system. In [5] he proves that this iteration procedure converges. A Newton iteration procedure is based on the derivative of the system by

$$\mathbf{x}^{n+1} = \mathbf{x}^n - \frac{f(\mathbf{x})}{f'(\mathbf{x})}. \tag{4.1}$$

In finite volumes, where matrix P is a diagonal matrix it could be argued that its derivative is a point-wise derivative. However, in finite elements, where P also contains non-zero off-diagonal entries determining a derivative is difficult. It is possible to establish a derivative with a type of method as proposed by Broyden [3]. However it is questionable whether a derivative does even exist at the contact discontinuity of the wet/dry interface. Therefore a Picard Iteration procedure is described in this chapter.

4.1.1 Picard iteration

Picard Iteration is a fixed point method. A fixed point p of a function g is defined as $g(p) = p$. With this the iteration method is defined as $g(p^n) = p^{n+1}$ and for the right starting value this method will converge.

In our case the equation to be solved, equation (2.23) can be written as

$$\left[P(\mathbf{h}) + T\right]^{-1} \left[P(\mathbf{h})\mathbf{z} + \mathbf{b}\right] = \mathbf{h} \tag{4.2}$$

which is a function of the shape $g(h) = h$. The iteration procedure is subsequently written as

$$\left[P(\mathbf{h_m}) + T\right]^{-1} \left[P(\mathbf{h_m})\mathbf{z} + \mathbf{b}\right] = \mathbf{h_{m+1}} \tag{4.3}$$

or with the definitions for P, T and $\mathbf{b}$ as given in Chapter 2

$$\begin{aligned}\left[M^H C(\mathbf{h}_m) + g\Delta t^2 \mathcal{B}^n D^{H^n} D^u\right] \mathbf{h}_{m+1} = {} & M^H C(\mathbf{h}_m)\mathbf{z} + M^H \mathbf{H}^n \\ & + \Delta t D^{H^n} \mathcal{B}^n M^u \mathbf{u}^n + \Delta t^2 D^{H^n} \mathcal{B}^n A^n \mathbf{q}^n\end{aligned} \tag{4.4}$$

where the index m indicates the iteration step. The starting condition that is used to calculate h^{n+1} is $h_0 = h^n$.

4.2 Matrix solvers

After the complete discretization of the system of equations described in sections 2.3, 2.4 and 4.1 the result is a matrix vector system given by

$$A_m \mathbf{h}_{m+1} = \mathbf{B}_m. \tag{4.5}$$

With

$$A = P(\mathbf{h}_m) + T$$
$$\mathbf{B} = P(\mathbf{h}_m)\mathbf{z} + \mathbf{b}$$

This system can be solved directly or with an iterative matrix solver. In this chapter several ways of solving this system are explored. It will be explained in detail what the difficulties are for each method, why a certain method does or does not work or what extra measures have to be taken to make a certain method work or perform better. In the end it becomes clear that the system is solved best using a lumped mass matrix at dry nodes and a consistent mass matrix at wet nodes. For larger matrices it can be decided to use the iterative matrix solver BiCGSTAB instead of a direct method. The various options that are considered are summarized in table 4.1. In the first column the category of the method is given. In the second column the main method to prohibit singularities is described. In the last column extra measures taken to enhance the performance of the methods are explained.

4.2.1 Direct matrix solvers

The system can be solved with a direct matrix solver. The most common is Gaussian-elimination. This is a very robust way of solving the matrix vector system. Moreover the exact solution is obtained. However, for large systems it can be very time consuming. In addition, it is unable to find the solution when the matrix is singular, thus when the inverse matrix does not exist. Gaussian-elimination is roughly the same as $\mathbf{h} = A^{-1}\mathbf{b}$, it is then easily seen that when A is singular there is no solution to the system. This might seem trivial, however, when a matrix contains a row of zeros the determinant is zero hence the matrix is singular. In case of wetting and drying the matrix A will contain zero rows for elements of which all nodes are zero. Solving the system is therefore not straightforward. There are several ways to by-pass this difficulty. They will be discussed in the sections below.

Minimum cell averaged depth

The matrix A is constructed from a part P, that is only added to the system for wet nodes, and a part T, which contains the implicit pressure gradient which is zero for elements with $\bar{H}_j^n = 0$. To prevent singularity of A it is possible to guarantee a contribution of T to A by using a minimum cell averaged depth, d_{min}. In case of negligible friction and a value of $d_{min} = 10^{-4}$, at dry locations the following value of T is contributed to A

$$T_{e,min} = \alpha_i^n \frac{\Delta x}{\Delta t} g d_{min} \begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}.$$

The results obtained in this way can be very promising, they are however very sensitive to the value of d_{min}. The best results are obtained for $d_{min} = 10^{-4}$. For values of $d_{min} = 10^{-5}$

Category	Method	Extra Measures
Direct Matrix Solvers	1. Using a minimum value of the average water depth in T	-
	2. Remove dry nodes from the system	Reconstruct T every iteration step Limit u_j for small H_j
	3. Element-wise wetting and drying	-
	4. Manipulating matrix entries (set $h = z$ at dry nodes)	Reconstruct T every iteration step Limit u_j for small H_j
	5. Lumping the mass matrix for dry elements	-
Iterative Matrix Solvers	6. Preconditioned Conjugate Gradient method	In combination with the lumped mass matrix at dry nodes
	7. Bi-Conjugate Gradient Stabilization Method	In combination with the lumped mass matrix at dry nodes Limit u_j for small H_j

Table 4.1: Overview of methods, to prohibit matrix A from becoming singular, to be explored in 1D.

and smaller the matrix becomes too close to singular and the solution becomes instable. For values larger than $d_{min} = 10^{-4}$ the propagation speed of the wet/dry interface can be slowed down considerably which results in a bore at the wet/dry interface. The value of d_{min} has a considerable influence on the number of iterations needed. Since the number of iterations depends on the speed of wetting or drying and the wetness of the domain it is not surprising that the number of iterations depends on the value chosen for d_{min}.

Removal of dry points from the system

Another way of prohibiting matrix A from becoming singular is to solve the system only for the wet nodes. If this is done no minimum depth is needed to prevent matrix A from becoming singular, since the zero rows are omitted from the system. If k is the array containing the row numbers of non-zero diagonals of A, the system $A(k,k)h(k) = b(k)$ is solved.

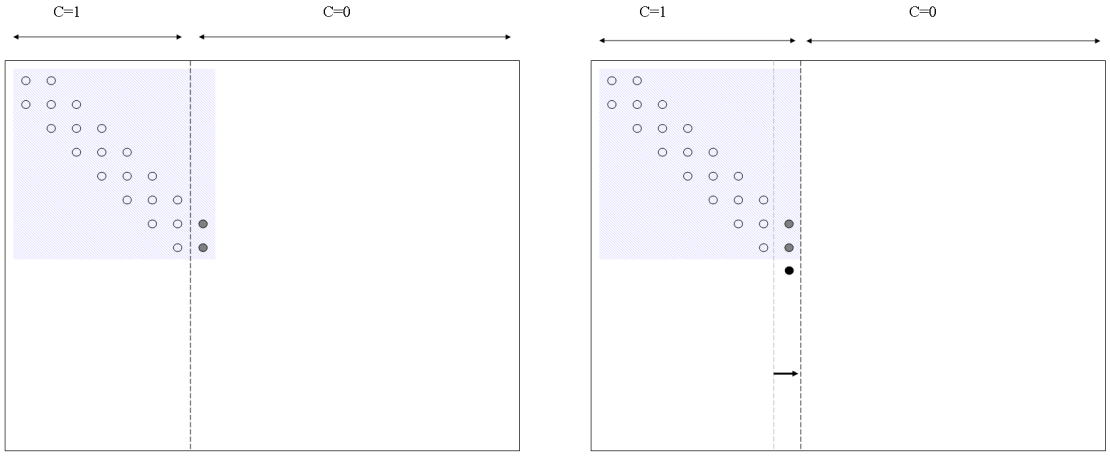
Some of the problems encountered are the existence of multiple wet domains and poor mass conservation. In many applications the layer thickness of the waterfront goes to zero. Therefore it is not a surprise that numerical difficulties arise with the physical correct representation of the solution. Whenever the computational domain consists of multiple wet sub-domains the matrix A is not irreducible anymore. Nonetheless this problem is quickly solved by solving all wet sub-domains separately. That mass is not conserved in this way is caused by the shape of matrix A and the construction of matrix A. When the system is only solved for wet nodes it is sufficient to determine which locations on the diagonal of A have a value larger than zero and remove the corresponding rows and columns from the system. Since the contribution of matrix P is zero for dry nodes and the contribution of matrix T is zero for elements connected to only dry nodes ($\bar{H}_j^n$ is the element averaged water level).

In figure 4.1 the structure of matrix A is given. The dots indicate the non-zero values of matrix A for a partially wet domain in one dimension. The dashed line indicates the wet/dry interface. The part of the matrix that is solved is shaded. There are two possible situations. The first situation always concerns the first iteration step. At that moment it is impossible to have a contribution of P to A and no contribution of T. However, it is possible to have a contribution of T at locations where the domain is dry. Whether or not T is larger than zero is determined by the element depth. The element depth is the average of the depths at the nodes of the element. If one node is dry (at that location there is no contribution of P) and the other node is wet there is a contribution of T to A for that element. This is indicated in figure 4.1(a) with the gray dots.

In case of flooding the wet/dry interface is proceeding to the right during the iteration procedure. It is possible to have the situation shown in figure 4.1(b), where the black dot indicates a contribution to A by P alone. It is clear that this entry of A is not used to solve the system, since its corresponding diagonal entry is zero. This causes a mass error.

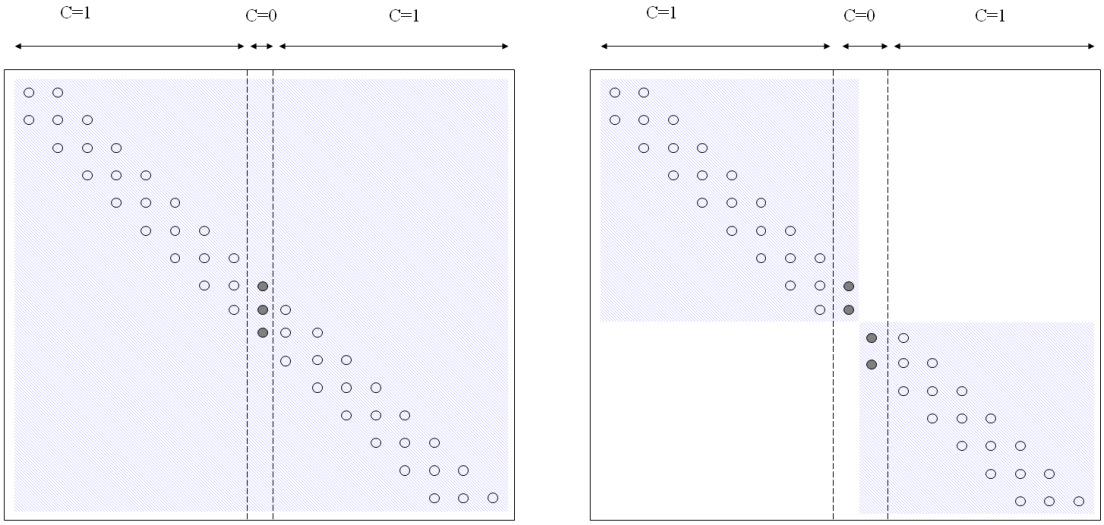
The solution to this problem seems obvious. When the matrix T is reconstructed in every wetting and drying iteration step, the contributions of T and P to system A will be consistent. So only situations as shown by figure 4.1(a) can occur. It is important to notice that this difficulty cannot occur in finite volumes since in that case P is a diagonal matrix.

In 2D the structure of matrix A will be slightly different however exactly the same problem can occur. For a completely wet element the corresponding diagonal entry will be non-zero. The surrounding wet elements will give an off-diagonal contribution, but not necessarily the off-diagonals next to the diagonal. After an iteration step it is possible that an off-diagonal contribution of P appears while the corresponding diagonal entry is still zero. Hence this



(a) (b)

Figure 4.1: Structure of matrix A in case of removal of the dry nodes from the system, the shaded area indicates the part of the matrix that is solved. (a) The initial situation at the start of the iteration procedure. (b) A situation that can occur after the wet/dry interface has proceeded to the right.



(a) (b)

Figure 4.2: Structure of matrix A, possible causes of trouble by very small entries (indicated by the gray dots). (a) Only one node is dry resulting in one column of very small entries. (b) Two adjacent nodes are dry resulting in a semi-uncoupled system.

contribution of P is not considered in the calculation, resulting in a mass error.

While reconstructing T every iteration step it is still possible to have very small diagonal entries that contain very small values, which can cause trouble. In figure 4.2 two examples are given of such cases. In figure 4.2(a) only one node is dry surrounded by wet nodes. The values of A marked gray are very small; they consist of a contribution of T only. However they are larger than zero, therefore the entire system is solved and no problems arise.

In figure 4.2(b) only two adjacent nodes are dry. Now problems do arise. This system consists of two parts, however neither solving the entire system nor solving the two individual parts results in a mass conserving solution. In fact both methods result in the same (non-conservative) solution with a very high peak at a node adjacent to the dry element. Some sort of semi-uncoupled system is composed that introduces mass errors. The essence of the problem is that the water depths become really small at the tip of the front while the velocities become very large. Bounding the velocity seems to resolve te problem. The best results are obtained by using $u_j^n = 0, \forall \bar{H}_j^n < 10^{-3}$, which will be denoted as $d_{u=0} = 10^{-3}$ from now on. For smaller threshold values mass is not always conserved. For larger threshold values the velocity is limited so much that a bore is created. It must be noted that the iteration behavior is much better compared to the situation in which a d_{min} is used, as discussed in the previous chapter. Where the average number of iterations was at first of $\mathcal{O}(10)$ it is here of $\mathcal{O}(1)$.

Another way of solving this problem of inconsistent contributions of P and T could be by turning the contribution of P on and off element-wise instead of node-wise. This will be discussed further in the next paragraph.

Element-wise contribution of matrix P

The matrix P is added to matrix A based on whether the entire element is dry or wet. The thought behind this approach is that it is never possible for a problem to occur as shown in figure 4.1. Since both contributions of P and T are established element-wise there will never be any non-zero entries of A that lay outside the solved part of matrix A.

The problem of this method is that it is not mass conservative. This problem is caused by partially wet elements. If a partially wet element has one node with a value of $h_j > 0$ and $h_{j+1} < 0$, its mass contribution is also based on the value $h_{j+1} < 0$. This means that there is negative mass added to the system. This can be visualized when we look at the core of the finite element method, the basis functions, as shown in figure 4.3. For elements 1 and 2 no difficulties arise since for all nodes of these elements $H = h - z$. The volume of water in an element is given by

$$V = \int_\Omega N_j^h h_j + N_{j+1}^h h_{j+1} dx = \frac{1}{2} h_j \Delta x + \frac{1}{2} h_{j+1} \Delta x.$$

Which corresponds to the volume that is found by looking at the mass matrix,

$$V = \Delta x \left[\frac{1}{3} h_j + \frac{1}{6} h_{j+1} + \frac{1}{6} h_j + \frac{1}{3} h_{j+1} \right] = \frac{1}{2} h_j \Delta x + \frac{1}{2} h_{j+1} \Delta x.$$

For element 3 it can be seen that the volume of water is

$$V_3 = \frac{1}{2} \Delta x h_3$$

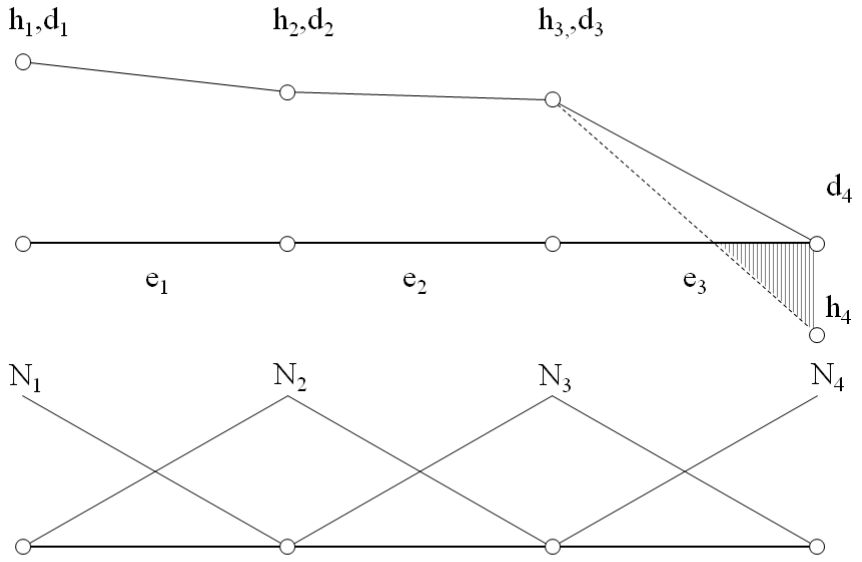


Figure 4.3: Determination of mass in the system, the shaded area indicates the negative volume that would be added if the mass matrix would be established element-wise.

which is not the same as the volume corresponding to the mass matrix,

$$V_3 = \int_\Omega (N_3 h_3 + N_4 h_4)\, dx = \frac{1}{2}\Delta x h_3 + \frac{1}{2}\Delta x h_4.$$

However it is the same as the volume that corresponds to a mass matrix with only entries in the column that correspond to the wet node,

$$M = \Delta \begin{bmatrix} \frac{1}{3} & 0 \\ \frac{1}{6} & 0 \end{bmatrix}.$$

The corresponding volume is thus

$$V_3 = \frac{1}{3}\Delta x h_3 + \frac{1}{6}\Delta x h_3 = \frac{1}{2}\Delta x h_3.$$

This corresponds to a mass matrix which is switched on and off based on whether a node is dry or wet. So mass conservation requires a node-wise wetting and drying approach.

Matrix manipulations

Another way to avoid matrix singularities is to manipulate the matrix. This is based on the principle that at dry nodes the water surface is equal to the bottom level. Only the dry nodes adjacent to the wet/dry interface are important in the establishment of the gradient at the interface. All nodes that are only connected to wet nodes can be assumed equal to the bottom level without consequences for the propagation of the waterfront. For these nodes the

equations reduce to $h_j^{n+1} = z_j$. When substituting this in the system matrix exactly all zero diagonal entries are filled and the result is a non-singular matrix A. For the same reason as for the case in which only the wet nodes where solved, it is necessary to reconstruct matrix T in every iteration step for this method to be mass conservative. For the same reason as we have seen by removing dry points from the system problems arise when the matrix A is not irreducible anymore, increasing the number of iterations per time step and the mass error.

It seems to be possible to reduce the mass error to $\mathcal{O}(10^{-14})$ however bifurcations can occur. That is, it is possible to have an iteration in which the solution varies between multiple solutions. In some cases one of these solutions is mass conserving while the others are not, it is then clear which one should be chosen. However in some cases none of the solutions is mass conserving. Again the solution can be found in limiting the velocity for small values of $\bar{H}_j^n$. The best results (considering the average number of iterations per time step and the resemblance to the analytical solution) are obtained for $d_{u=0} = 10^{-3}$ however smaller values are still mass conserving (as opposed to the case in which only the wet system is solved). The iteration behavior is as favorable as in the case the matrix is only solved for the wet part of the domain.

Manipulating the matrix leads to roughly the same results as solving only the wet part of the computational domain. Since a smaller system has to be solved when only the wet part of the computational domain is considered, the last seems to be the more favorable choice of the two. Yet, in two dimensions solving only the wet computational domain is not as straightforward as in one dimension. In one dimension a tridiagonal matrix has to be considered but for two dimensions a large sparse matrix has to be solved which severely complicates the matter. Thus in 2D it can be preferable to manipulate the matrix as described above.

Lumping the mass matrix

Instead of reconstructing the matrix T in every iteration step it is also possible to guarantee mass conservation, in combination with solving only the wet part of the system or manipulation of the matrix, by lumping the mass matrix. When the mass matrix is lumped, the contribution of P to the system outside the solved part of the matrix is zero (as in the right drawing of figure 4.1). Hence no mass error is introduced. The mass matrix for a one-dimensional element is given by equation (4.6). The lumped mass matrix is derived by summing the values of the consistent mass matrix over the rows and inserting these on the diagonals. This can be interpreted as assigning the value of the half of the volume of water in an element to one node and the other half to the other node instead of spreading it more smoothly over the element. The result is given by equation (4.7).

$$M_C = \frac{\Delta x}{6} \begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} \tag{4.6}$$

$$M_L = \frac{\Delta x}{2} \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \tag{4.7}$$

It is possible to write the consistent mass matrix as a contribution of the lumped mass matrix and another matrix, which is quickly recognized as the diffusion matrix, see equation (4.8). Since the lumped mass matrix is a diagonal matrix the method using a lumped mass matrix bears more resemblance to Casulli's finite volume problem. As discussed in Chapter

2.4 in case of a diagonal matrix P, as for instance when the lumped mass matrix is used, the existence of A^{-1} is guaranteed. Using the lumped mass matrix adds diffusion to the system. Diffusion is a stabilizing term, for its ability to smoothen irregularities, which can be a desirable property. However the lumped mass matrix gives less accurate results. Hence, aimed is at minimum use of the lumped mass matrix. Since problems arise only at the wet/dry interface it is possible to use the consistent mass matrix at wet or partially wet elements and the lumped mass matrix at dry elements.

$$M_C = M_L - \frac{\Delta x}{6\Delta t}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix} \tag{4.8}$$

All possible non-zero entries of matrix A are shown in figure 4.4 where the initial wet/dry front is indicated by the dashed line (this means that the values of the mass matrices are put to zero to the right of the dashed line). With possible non-zero values the shape of matrix A is meant that would result if after a couple of iteration steps the entire domain would be wet. In case of flooding the dashed line will progress to the right, while doing this there cannot occur a situation in which an off-diagonal term is non-zero while the corresponding diagonal term in the same row is zero as a result of lumping the mass matrix at dry elements. Consequently mass is conserved in case only the wet domain is solved or when ones are placed in case of zero diagonals and the RHS is set equal to the bottom. In 2D the same holds. Again lumping of the mass matrix at initially dry nodes prohibits non-zero off-diagonal terms with a corresponding diagonal entry that is zero.

An advantage of this method over reconstruction of matrix T is that whenever A contains multiple adjacent wet domains it does not result in a non-conservative solution. This seems rather arbitrary since in this case there does occur again a large extreme value however this value is negative this time. Thanks to the definition of the water depth this does not influence the solution. Although this seems to be random, this extreme value is never positive when a lumped mass matrix is used and very often positive when it is not. Another advantage of this method is that it takes less computational effort since it is not necessary to reconstruct T every iteration step.

This method has again very favorable iteration behavior. Moreover it behaves very robust, no extra measures are necessary for smooth behavior or unconditional mass conservation.

4.2.2 Iterative matrix solvers

For large computational domains the matrix A rapidly expands, having a size of $n \times n$, where n is the number of nodes, while its non-zero entries are only $\approx 3n$. In two dimensions this difference is even more pronounced having a total number of $4n^2$ entries in the matrix with $\approx 7n$ non-zero entries. Iterative matrix solvers are able to use this property and find an approximate solution with less computational effort than a direct solver.

Preconditioned Conjugate Gradient Solver, PCG

This method is applicable to symmetric positive definite matrices. The number of iterations required is proportional to $\sqrt{cond(A)}$. To make $cond(A)$ smaller the matrix A is preconditioned.

Intermezzo 4.1. *A matrix is symmetric if it holds that* $A = A^T$.

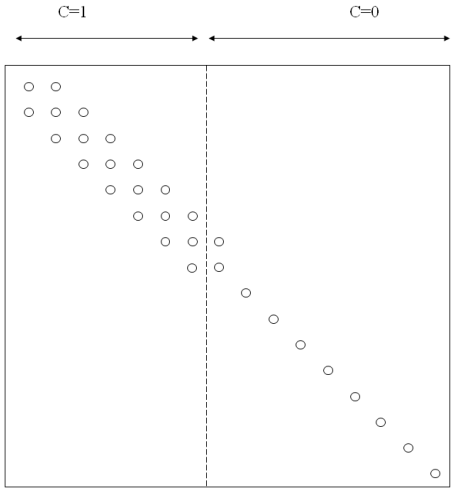


Figure 4.4: All possible non-zero entries matrix A in case the mass matrix is lumped for completely dry elements. The initial wet/dry interface is indicated with the dashed line.

A matrix is positive definite if $x^T Ax > 0$. *The determinant of a positive definite matrix is always positive and therefore a positive definite matrix is always nonsingular.*

The condition number of a symmetric matrix A *is given by* $cond(A) = \frac{max|\lambda(A)|}{min|\lambda(A)|}$.

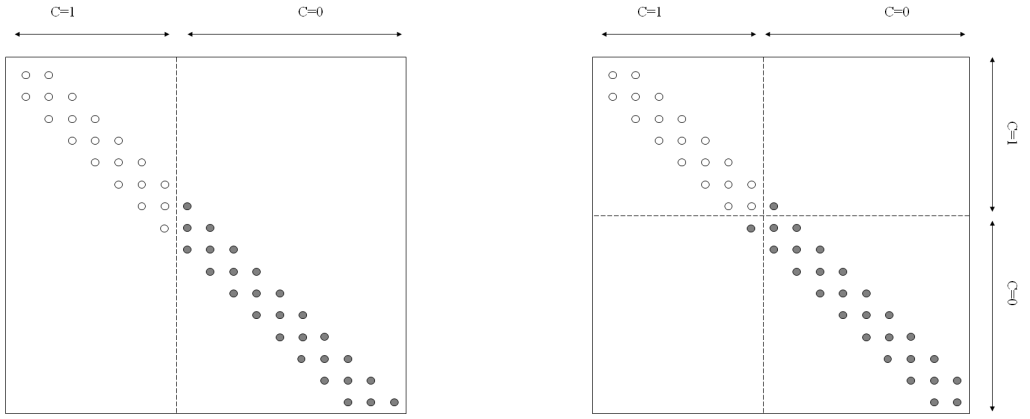
The main problem of this method is that matrix A has to be symmetric, which it is not in case of wetting and drying. The element matrices P and T are symmetric but when the global matrix P is added per wet node its contribution to the global matrix A can be asymmetric. There are several ways to obtain a symmetric matrix A.

It is possible to establish a symmetric matrix A by adding matrix P and T element-wise. Yet we have seen in Chapter 4.2.1 that this is not mass-conservative. Another way of obtaining a symmetric matrix A is to add $\left(P^T diag\,(c)\right)^T diag\,(c)$ instead of $P diag\,(c)$. The effect of this can be seen in figure 4.5. In figure 4.5(a) matrix A is sketched where the contribution of P is canceled for all dry nodes, resulting in an asymmetric matrix. In figure 4.5(b) the contributions of dry nodes to wet nodes are canceled too. Consequently a symmetric matrix is created. The drawback of this way of creating a symmetric matrix is that it is not mass conserving. A preconditioned conjugate gradient solver is not a suitable solution algorithm because the symmetry requirement makes it impossible for the scheme to be mass conserving.

Bi-conjugate Gradient Stabilization Solver, BiCGSTAB

BiCGSTAB does not require a symmetric matrix, it takes slightly more computational effort than PCG and convergence is not proven. However the convergence behavior often resembles that of PCG.

For mass conservation and to prevent the matrix from becoming singular, BiCGSTAB is



(a) (b)

Figure 4.5: Structure of matrix A. (a) The asymmetric original structure of matrix A. (b) The structure of matrix A after symmetrization. The gray circles indicate the turned off contributions of P.

used in combination with a lumped mass matrix at dry elements and manipulation of the matrix. BiCGSTAB is preconditioned with the Jacobi preconditioner, here denoted by R_{ij}. The Jacobi preconditioner is given by

$$R_{ij} = \begin{cases} A_{ij} & \text{if} \quad i = j \\ 0 & \text{otherwise.} \end{cases} \tag{4.9}$$

In combination with limited velocities for very small water depths, so that the thin front does not shoot ahead too fast, becoming thinner and thinner, this method gives quite good results. This value at which the velocities are put to zero is taken $d_{u=0} = 10^{-4}$. For smaller values mass conservation becomes less, $\mathcal{O}(10^{-7})$, and the robustness of the solution is compromised. Iteration behavior is slightly less than for the previously discussed methods, but can be considerably improved by adjusting the BICGSTAB settings.

A difficulty that arises in some cases is that there can occur bifurcations in the iteration procedure (that do give mass conserving solutions). The right solution could then be chosen on mass conservation grounds. However the bifurcation can contain many different solutions. To establish the occurrence of a bifurcation and choose the best solution it is necessary to store all solutions temporarily and go trough all bifurcation steps, increasing the number of iterations and the amount of storage space required. The occurrence of these bifurcations can be diminished by adjusting the BICGSTAB settings (lowering its tolerance and enlarging its maximum number of iterations), however this is more costly in terms of calculation time. Using d_{min} instead of lumping to prevent the system from becoming singular solves the bifurcation problem completely. The extra iterations needed in this case are unfavorable but can probably level with the costs of adjusting BiCGSTAB settings. The fact that it resolves the problem instead of diminishing only is of course decisive here.

Another solution to the bifurcation problem could be to use a stopping criterion in case of a bifurcation loop. Such a procedure is described in section 4.3.3.

4.2.3 Overview

From table 4.2 it becomes clear that several of these methods give good results. Method 2 and 4 give the best results with respect to iterations and mass conservation. However, the velocity must be limited for robustness reasons and the results are very sensitive to the value of the water depth where u is limited for. The use of the lumped mass matrix requires only slightly more iterations and the method seems to be much more robust. These methods can be combined with BiCGSTAB to speed up the calculations. However the robustness is compromised as it is possible for bifurcations to occur and mass conservation is slightly less. These bifurcations can be resolved by using BiCGSTAB in combination with a minimum cell averaged depth.

4.3 Extension to two dimensions

In this section all findings of the previous sections will be discussed for two dimensions. Although there are direct matrix solvers that are able to solve large systems with comparable computational effort to indirect matrix solvers it is felt that that is outside the scope of this research and only the iterative solver BiCGSTAB will be considered. As discussed earlier on in this chapter element-wise contributions to the mass matrix are very hard to implement in a mass conserving manner and will therefore not be discussed here either.

This section starts with the idea that the most robust and reliable method in combination with BiCGSTAB to solve the system in one dimension was using the lumped mass matrix at dry elements, limiting the velocity for small depths and working with a cell averaged minimum depth to enhance the robustness. Implementing this combination of measures results indeed in a working two-dimensional model. This section will therefore concentrate on this combination of measures and the influence of varying certain factors on the performance. In table 4.3 the different variations on this combination of measures are summarized. It should be noted that the Jacobi preconditioner as used in one dimension does not perform well in two dimensions. Therefore an incomplete LU (ILU) preconditioner will be used here.

In general it can be said that the performance of the algorithm is not very robust and mass conservation is severely compromised by the increase in scales of the two-dimensional tests, which will be further explained in chapter 5.3. What in one dimension seemed to be an algorithm with some difficulties that could be bypassed has turned into an algorithm that needs to be steered and molded into the right shape and direction. Most problems can be resolved, although the robustness of the one-dimensional model cannot be matched.

4.3.1 Working with a minimum cell averaged depth

The use of a threshold depth is in two dimensions compulsory, while in one dimension it was possible to obtain a nonsingular system by manipulation of the matrix or removal of dry nodes from the system. In two dimensions the removal of dry nodes from the system is not as straightforward as in one dimension since every element is linked to approximately six other elements instead of two in one dimension. Moreover the matrix is not as well organized as in one dimension which complicates matters as well.

Manipulation of the matrix is just as straightforward as in one dimension. If a diagonal entry is zero a one should be placed on the diagonal, all other values in that row to zero and the RHS equal to the bottom level. Despite its simplicity it does not work in two

Category	Method	Extra Measures	Mass Conservation	Iterations	Advantages	Disadvantages
Direct Matrix Solvers	1. Using a minimum value of the average water depth	-	$\mathcal{O}(10^{-14})$	$\mathcal{O}(10)$	Behaves very robust	Needs many iterations per time step
	2. Remove dry nodes from the system	Reconstruct T every iteration Limit u_j for small H_j	$\mathcal{O}(10^{-14})$	$\mathcal{O}(1)$	Solving smaller system	Sensitive to value H_j, u_j is set to 0 for
	3. Element-wise wetting and drying	-	$\mathcal{O}(10^{-1})$	$\mathcal{O}(1)$	Symmetric matrix	Never mass conservative
	4. Manipulating matrix entries	Reconstruct T every iteration Limit u_j for small H_j	$\mathcal{O}(10^{-14})$	$\mathcal{O}(1)$	Easily Extended to 2 D	Sensitive to value H_j, u_j is set to 0 for
	5. Lumping the mass matrix for dry elements	-	$\mathcal{O}(10^{-14})$	$\mathcal{O}(1)$	Behaves Very Robust	Inconsistent scheme
Iterative Matrix Solvers	6. Preconditioned Conjugate Gradient Method	[2]	$\mathcal{O}(10^{1})$	$\mathcal{O}(1)$		Never mass-conservative
	7. Bi-Conjugate Gradient Stabilization Method	[2]Limit u_j for small H_j	$\mathcal{O}(10^{-11})$	$\mathcal{O}(1)$	Suitable for large systems	Poor mass conservation Not very robust

Table 4.2: Summary of the different measures used and their performance in 1D.

2 In combination with the lumped mass matrix at dry nodes

Basic measures	**Variations on the basic measures**		
BiCGSTAB	-		
Lumping the mass matrix at dry elements	Lumping the mass matrix everywhere	Lumping the mass matrix at dry and partially dry elements	No lumping
d_{min}	Matrix manipulations	Solve only the wet part of the domain	
$d_{u=0}$	Different values		

Table 4.3: Overview of methods, to prohibit matrix A from becoming singular, to be explored in 2D.

dimensions and after a few time steps the solution explodes. When trying this on an almost one-dimensional test with only one element in the lateral direction the performance of the two-dimensional model is just as good as that of the one-dimensional model. So most probably the matrix becomes close to singular because of many small entries in the matrix. Nonetheless BiCGSTAB does not experience any trouble with this, since it does not use the inverse of a matrix but only matrix vector products. In case of an almost one-dimensional problem small values appear only around the wet/dry interface, only at one location in the matrix. However for a fully two-dimensional problem very small values will appear in the matrix at many locations and BiCGSTAB will explode eventually. This implicates that the use of a threshold depth is necessary to limit the occurrence of too many small entries in the matrix and thus prevents BiCGSTAB from exploding. This theory is also supported by the fact that bifurcations occur more frequently and the convergence speed of BiCGSTAB is much lower for a fully two dimensional problem.

The influence of different values for d_{min} is not as pronounced as in one dimension. However for values of $d_{min} > 10^{-3}$ m a bore develops at the wet/dry interface. The number of iterations is not completely indifferent to the value of d_{min}, but a trend as clear as in one dimension is not to be observed.

4.3.2 Lumping the mass matrix

The influence of four different ways of lumping the mass matrix are investigated.

- Lumping the mass matrix nowhere,
- lumping the mass matrix everywhere,
- lumping the mass matrix only for dry or partially dry elements, and
- lumping the mass matrix only for dry elements.

Using the consistent mass matrix everywhere leads to a considerable increase in the number of iterations necessary and in some cases even to an explosion of the solution. Lumping the mass matrix everywhere decreases the accuracy of the solution. The schemes loses its ability to perfectly follow the analytical solution. An artificial bore is generated at the wet/dry interface. In some cases the amount of iterations is reduced, but not in all. In some cases lumping even causes the system to blow up. Only partly lumping seems to be a solution. However it is case dependent whether the mass matrix should be lumped only for completely

dry elements or also for partially dry elements. On the contrary in cases with small gradients at the wet/dry interface complete lumping or lumping when at least one node of an element is dry can improve the iteration behavior. Positively, friction is a stabilizing term in this respect. In cases with friction it does not considerably influence the solution whether the mass matrix is lumped on dry elements only or at partially dry elements aswell.

It may be concluded that the solution and robustness depend to a large extent on when the mass matrix is lumped. What criterion should be used for lumping is again highly case dependent. This is a highly undesirable feature in civil engineering practice. A model should be able to represent a wave on a sloping beach and a dam break at the same time. Luckily friction plays a stabilizing role here. For civil engineering applications, friction can rarely be neglected, making this method still practically applicable.

4.3.3 Occurrence of bifurcations

Bifurcations in the solution of BiCGSTAB were resolved in one dimension by using a minimum value for the cell averaged depth. In spite of using this d_{min} in two dimensions bifurcations still occur. These bifurcation loops can consist of more than 10 different solutions. It is necessary to implement a routine that establishes the occurrence of a loop and selects the best solution. As a stopping criterion the minimization of the mass difference between two subsequent time steps is taken. This procedure is perfectly able to exit bifurcation loops and reduce the number of iteration steps.

When this stopping criterion is used the occurrence and the size of bifurcations is significantly reduced. It seems that choosing the *wrong* answer once causes a chain reaction of bifurcations and more *wrong* choices until the system explodes.

Using double precision significantly reduces the occurrence of bifurcations. Using double precision does reduce the occurrence of bifurcations but does not resolve the problem completely. However it does reduce the number of iterations and storage space required.

A sloping bottom has a negative influence on the occurrence of bifurcations. Even for calculations performed with double precision. The stopping criterion does reduce the occurrence of bifurcations however a difference in iteration behavior can be noticed between sloping and flat bottoms.

The exit criterion discussed here is only a solution to cure the symptoms, not to the cause. It works in obtaining a mass conservative solution but unnecessary iteration steps have to be performed and all solutions to these steps have to be stored in order to be able to choose the right one. A solution might be found in choosing a formulation for H that is continuous instead of discontinuous. Thus instead of an abrupt change from $H = 0 \,\text{where}\, h < 0$ to $H = h \,\text{where}\, h \geq 0$ there could be a formulation which changes gradually.

4.3.4 Overview

In table 4.4 the different measures discussed in the last section are summarized. As a method to prevent matrix A of becoming singular only the use of a minimum cell averaged depth worked. Moreover the performance with a threshold depth for limiting the velocity was best for $d_{u=0} = 10^{-4}$. The methods perform similar with respect to mass conservation and iteration behavior. Nevertheless if the mass matrix is lumped at dry elements only for large interface gradients or if the mass matrix is lumped at dry and partially dry elements in cases with small interface gradients the number of iterations will increase to $\mathcal{O}(10)$.

	Method	Performance	Mass conservation % of the initial water volume	Iterations
1	Lumping the mass matrix everywhere	-Decrease in accuracy -Increase in robustness	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$
	Lumping the mass matrix at dry and partially dry elements	Works only for small gradients at the wet/dry interface	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$
	Lumping the mass matrix at dry elements	Works only for large gradients at the wet/dry interface	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$
	Lumping the mass matrix nowhere	- Increase in the number of iterations - Occurrence of instabilities	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$
2	Matrix manipulations	Unstable	-	-
	Solve only the wet part of the domain	Not straightforward to implement in 2D	-	-
	d_{min}	Works best for $d_{min} = 10^{-3}$	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$
3	$d_{u=0}$	Works best for $d_{u=0} = 10^{-4}$	$\mathcal{O}(10^{-6})$	$\mathcal{O}(1)$

Table 4.4: Performance of different solution procedures in two dimensions. In every case $d_{min} = 10^{-4}$, $d_{u=0} = 10^{-4}$ and BiCGSTAB is used to solve the matrix.

Chapter 5

Validation and discussion

To test the performance of the wetting and drying procedure with respect to its solution, stability properties, mass conservation and iteration convergence the numerical solutions of different tests are compared to several corresponding analytical solutions and measurements from literature.

In the first part of this chapter one-dimensional cases are discussed. These are compared to analytical solutions and we will see that the results are very satisfying. In the second part of this chapter several two-dimensional test are discussed. Some tests are compared to analytical results and some are compared to laboratory measurements. Despite of the difficulties encountered upon in the two-dimensional model the results look quite well. It should be noted that the results of the tests appeared to be very sensitive to the numerical settings. The last part of this chapter is devoted to mass conservation since this is one of the main aspects of wetting and drying.

5.1 1D Test Cases

In this section several one-dimensional test cases will be considered of which the numerical results will be compared to the analytical solution. At first a classical one-dimensional dam break problem will be considered. This test will be performed with a dry bed to test the wetting and drying procedure, and with a wet bed to assess the performance of the advective scheme. This is quite a severe test case because of the large initial gradient at the wet/dry interface. Secondly a test with flow over a long crested weir will be done. This is again not to assess the performance of the wetting and drying algorithm but to assess the performance of the advective schemes. As explained in section 1.4 the wetting and drying problem comprises both positivity of water depths (mass conservation) and momentum conservation over a contact discontinuity. So it is just as important to assess the momentum conservation properties of the advective terms. After this a parabolic basin with an oscillating plane surface will be assessed. In the end a test with non-breaking waves on a sloping beach will be looked at.

All results are given for a second order Riemann advection scheme with a minmod limiter. The mass matrix will be lumped at dry elements. All results are calculated both for Gaussian-elimination and for BiCGSTAB unless stated otherwise.

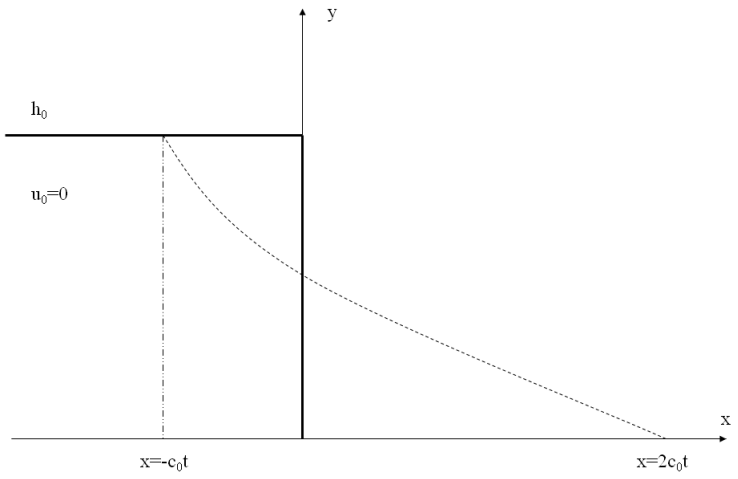


Figure 5.1: Initial state (solid line) and solution at time t (dashed line) for the dam break problem.

5.1.1 Dam Break test

The dam break test is a classical test widely used to validate the performance of wetting and drying procedures. Examples in literature are Stelling and Duinmeijer [22], Fraccarollo and Toro [9], LeVeque [15] and Toro [26].

Two regions with different water levels, initially separated by a vertical wall, are considered. At $t = 0$ s the wall is suddenly removed and a flood wave enters the downstream domain.

This test case is used to assess the performance of the advective discretization and the ability of the model to handle the flooding of several cells per time step. The difficulties that arise for the dry dam break problem are

- the thickness of the water layer at the front becomes infinitely small in the analytical solution. This is of course difficult in a numerical representation,

- the gradient of the water front is very large, flooding the dry part of the domain at a very high speed.

Analytical description dry bed case

The analytical solution to the dam break problem is derived in Stoker [24] for the inviscid shallow water equations in which friction is neglected. For an upstream water mass that is initially at rest and a downstream water level equal to zero, as shown in figure 5.1, the

solution between $x = -c_0 t$ and $x = 2c_0 t$ is described by

$$\begin{aligned} h &= \frac{1}{g}\left(\frac{2}{3}c_0 - \frac{x}{t}\right)^2, \\ u &= \frac{2}{3}\left(c_0 + \frac{x}{t}\right), \text{ with} \\ c_0 &= \sqrt{g h_0} \end{aligned} \tag{5.1}$$

Numerical setup

This test is performed three times. First the system is solved direct, secondly the system is solved iteratively with BiCGSTAB. For these cases the mass matrix is lumped at fully dry elements. The velocity is set to zero for $d_{u=0} = 10^{-3}\ m$. The resolution is set to $\Delta x = 1\ m$ and $\Delta t = 0.05\ s$.

In the third test grid refinement is applied, resulting in $\Delta x = 0.1\ m$ and $\Delta t = 0.001\ s$. The system is solved direct and the velocity is set to zero for $d_{u=0} = 10^{-3}$ again. The upstream water level, h_0 is set to 1 m. All nodes with $x_i \leq 20$ are initially wet; the others are dry.

Results and discussion dry bed case

The results obtained with Gaussian-elimination are presented in figure 5.2. The average number of iterations per time step is 1.65, figure 5.3. In figure 5.4 and 5.5 the results are shown using BiCGSTAB. The results with respect to the analytical solution and the number of iterations are quite similar in both cases.

The scheme is able to represent the analytical solution disregarding some wiggles in the velocity profile. These wiggles in the velocity profile can be suppressed with grid refinement as is clearly visible in figure 5.15(a). However the lag in velocity that was visible in figure 5.2 and 5.4 does not vanish for increasing resolution. Apparently the velocity does not converge completely to the analytical solution. The use of an explicit advective term could be causing this lag in the velocity profile. When there is such a large discontinuity in velocity it is not hard to imagine that using the velocity from the old time step causes the lagging propagation of the shock wave. Since this is not of major importance for the testing of the wetting and drying algorithm no implicit advective formulation is tested in this study.

No artificial bore is created at the wet/dry interface. This means on the one hand that the model does not experience any trouble with the flooding of more than one element per time step. That does not mean that the time step can be taken any size. The Courant condition still has to be fulfilled due to the explicit advective term. On the other hand the advective term performs well, since the advective term is largely responsible for the correct representation of the shape of the wet/dry interface.

If one looks at the mass conservation in figures 5.2 and 5.4 a strange thing occurs which is most apparent in figure 5.2. If a mass error occurs, the difference with respect to the preceding time step is of the same size or a multiple of it, in case of Gaussian-elimination, $3.5527 \cdot 10^{-15}\ m^2$. Here this does not seem to be very disturbing, however we see later on that the size of this error increases significantly for larger scales in two dimensions. In section 5.3 special attention is paid to this phenomenon.

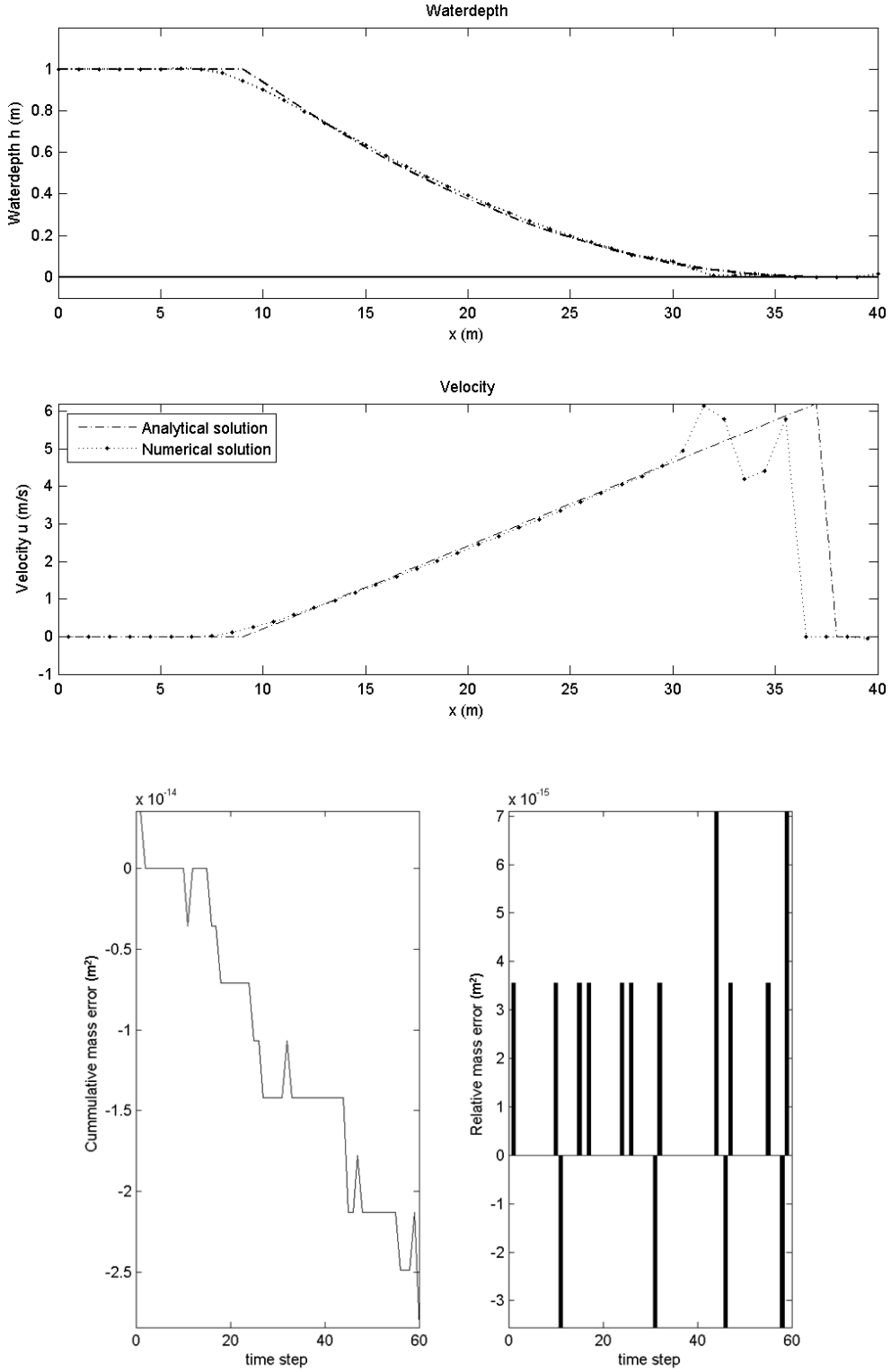


Figure 5.2: Results for the dam break problem at $t = 3\ s$, with M_L used at the dry nodes, solved direct and $d_{u=0} = 10^{-3}\ m$.

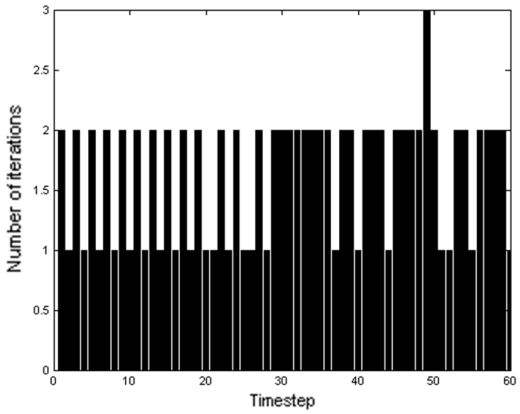


Figure 5.3: Number of iterations for the dam break problem at $t = 3\ s$, with M_L used at the dry nodes, solved direct and $d_{u=0} = 10^{-3}\ m$, the average number of iterations is 1.65.

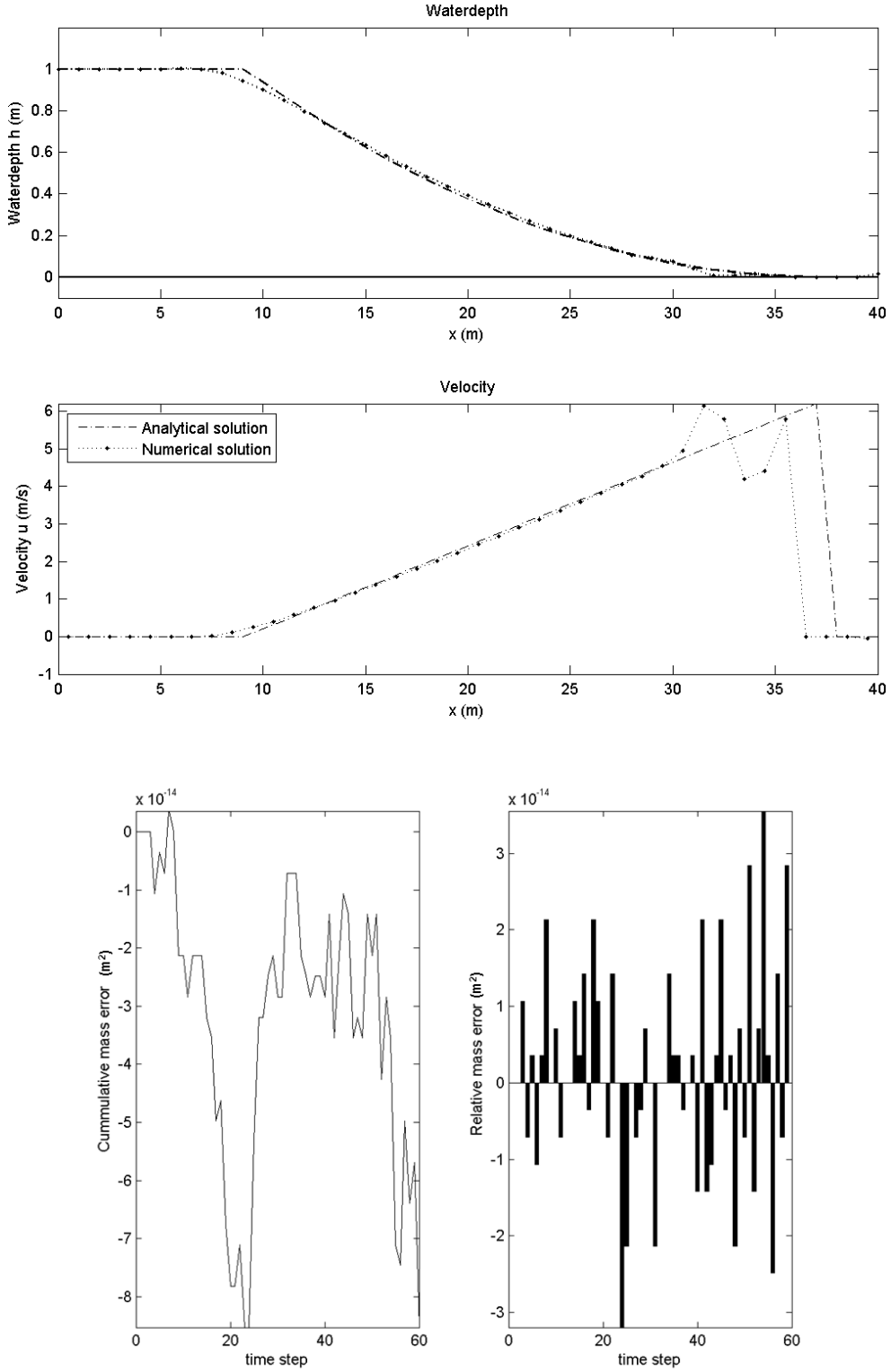


Figure 5.4: Results for the dam break problem at $t = 3\ s$, with M_L used at the dry nodes, solved iteratively with BiCGSTAB and $d_{u=0} = 10^{-3}\ m$.

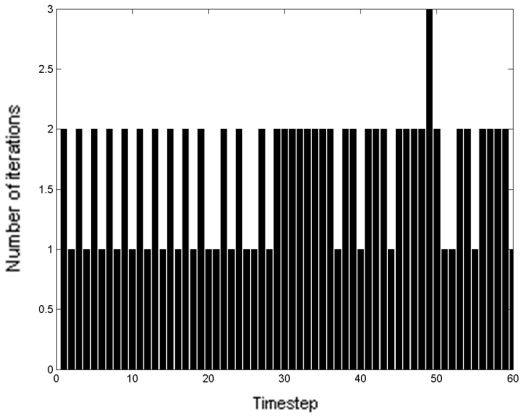


Figure 5.5: Number of iterations for the dam break problem at $t = 3\ s$, with M_L used at the dry nodes, solved iteratively with BiCGSTAB and $d_{u=0} = 10^{-3}\ m$, the average number of iterations is 1.65.

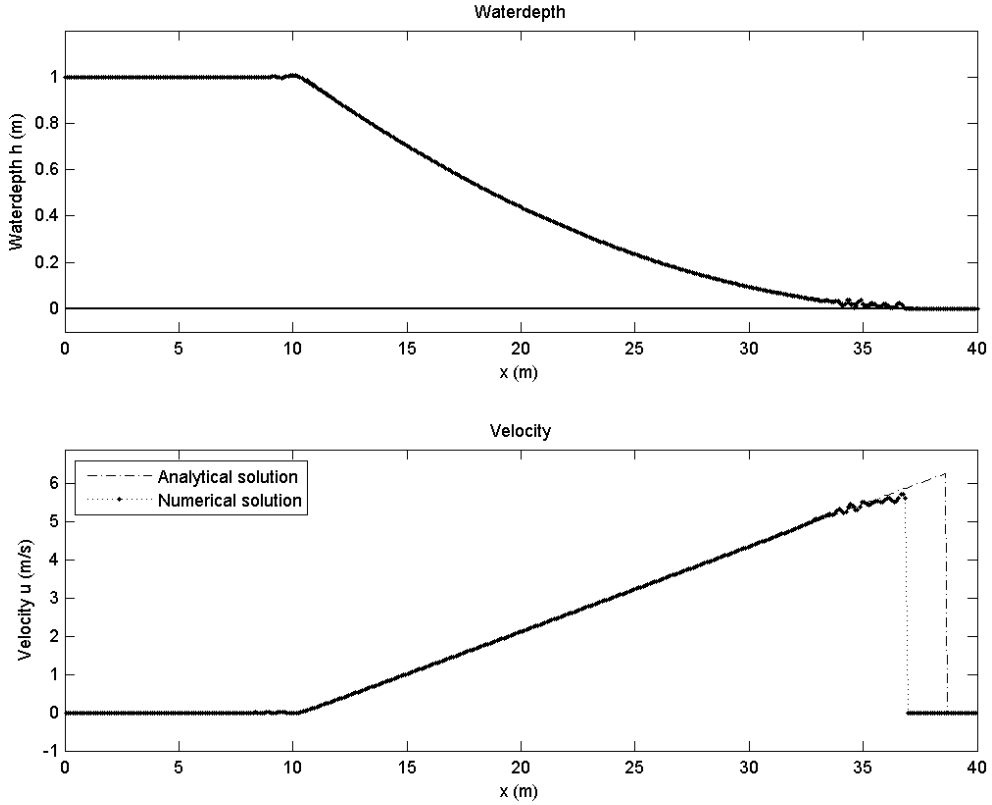


Figure 5.6: Results for the dam break problem at $t = 3\ s$, with M_L used at the dry nodes, solved direct and $d_{u=0} = 10^{-3}\ m$, with $\Delta t = 0.001\ s$ and $\Delta x = 0.1\ m$.

Analytical description wet bed case

The solution with a wet bed downstream is slightly different. Now four separate zones can be recognized. The upstream area that has not been affected by the dam break, a parabolic shaped area, a constant state shock wave area and an unaffected downstream area. These four zones are indicated in figure 5.7. The solution for region I is equal to the solution for the wet bed case. The solution in area II is

$$\begin{aligned} h_{II} &= \tfrac{1}{g}\left[-\tfrac{1}{3}\left(u_{II}-c_{II}\right)+\tfrac{2}{3}c_0\right]^2 \\ u_{II} &= \xi\left(1-\tfrac{d_{III}}{d_{II}}\right). \end{aligned} \tag{5.2}$$

The propagation velocity of the front is given by

$$\xi = \sqrt{g\frac{h_{II}}{h_{III}}\frac{h_{II}+h_{III}}{2}}. \tag{5.3}$$

The solutions in region 0 and III are equal to the left and right initial conditions.

This test case does not verify the solution with respect to wetting and drying. However the correct representation of the height of the hydraulic jump indicates whether the scheme is momentum conservative.

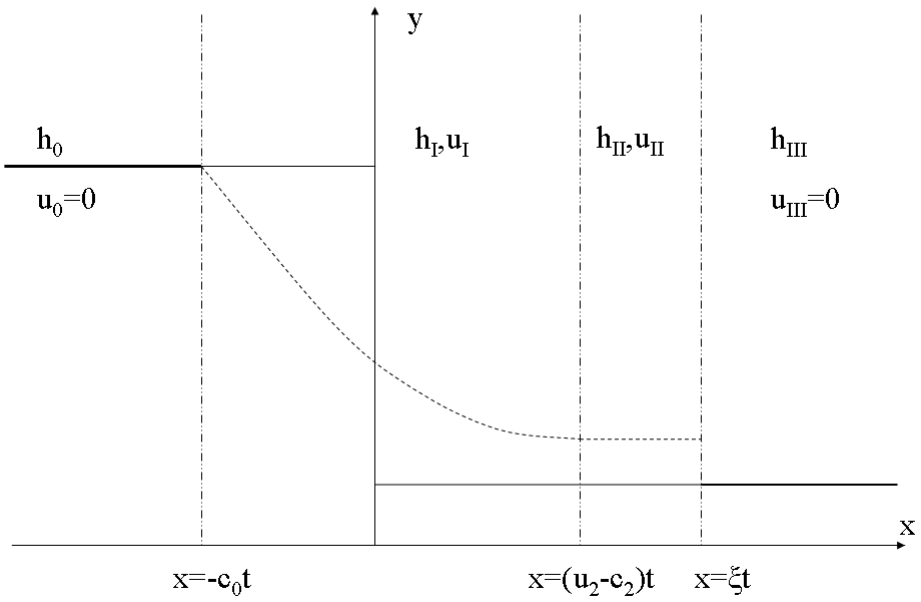


Figure 5.7: Initial state (solid line) and solution (dashed line) for the dam break problem with a non-zero water depth at both sides of the dam.

Numerical setup

The numerical parameters for this test are set to $\Delta x = 0.5\ m$ and $\Delta t = 0.05\ s$. The mass matrix is lumped at completely dry elements and solved with Gaussian-elimination. The

upstream water depth was set to $h_0 = 1\ m$ at all nodes with $x_i \leq 20$. The downstream water depth was set to $h_{III} = 0.1\ m$ at nodes with $x_i > 20$.

Results and discussion wet bed case

As can be seen in figure 5.8 the height of the jump is not represented correct. It is too high therefore the particle velocity of the water behind the front is too low and the propagation speed of the front of the numerical solution is considerably lower than of the analytical solution. This test indicates that momentum is not conserved properly. This is most probably caused by the approximations for the advective velocities. However since in the previous test the scheme was able to model the dry-bed dam break test without a bore being generated it is assumed that these losses are not of too much importance.

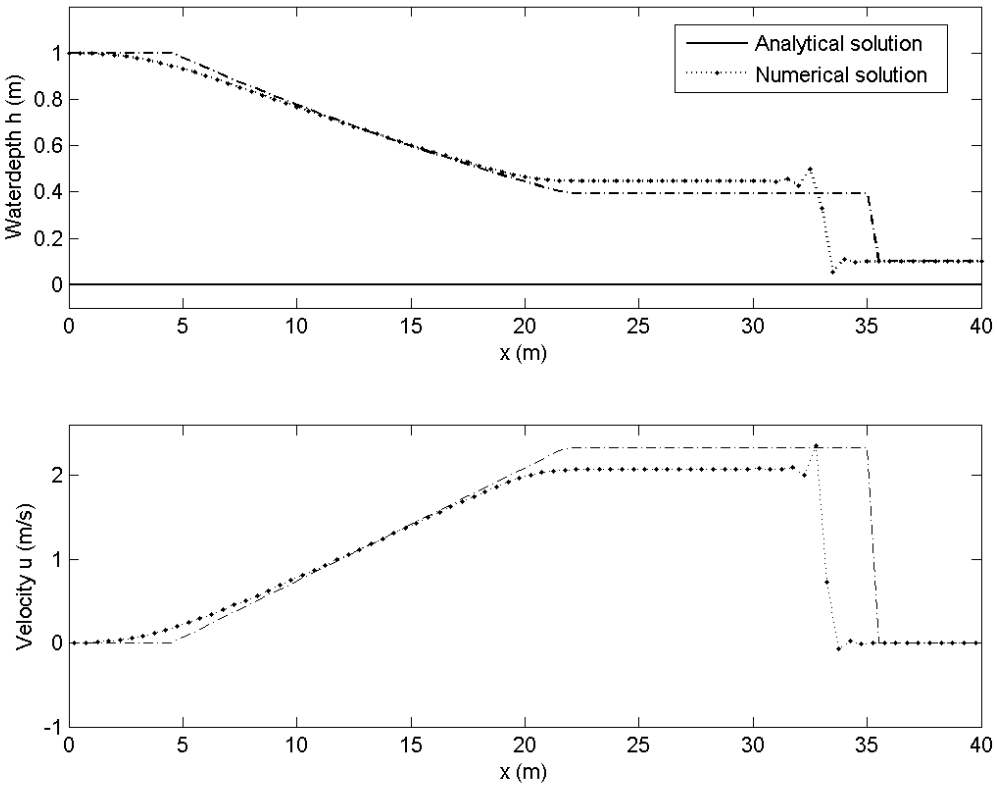
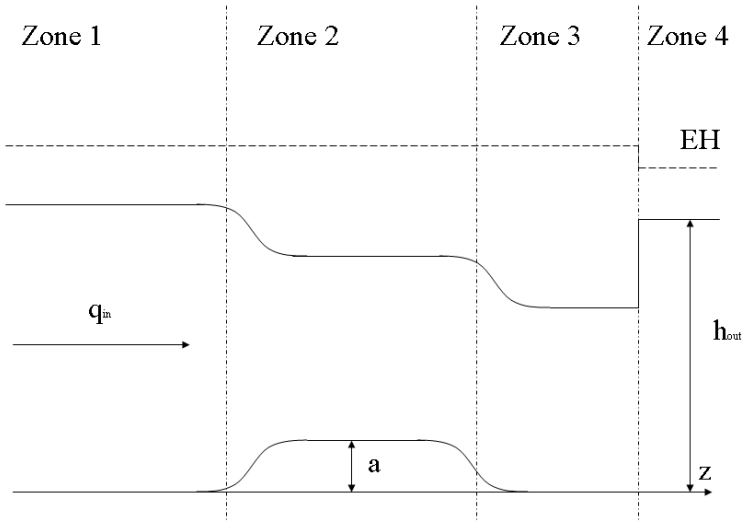


Figure 5.8: Results for the dam break problem at $t = 5\ s$ with a non-zero water depth at both sides of the dam, with M_L used at the dry nodes, solved direct.

5.1.2 Long crested weir

This test case is used to verify the ability of the method to correctly represent the water levels and energy head for flow over a long crested weir. The downstream flow is supercritical, the flow at the crest of the weir is critical, and upstream the flow is subcritical. For a good



[!htp]

Figure 5.9: Different solution zones for a Long-crested weir. The water level is indicated by the solid line and the energy head by the dashed line.

representation of different flow regimes it is assessed whether the height of the hydraulic jump and the energy head loss over the hydraulic jump is represented correctly and whether the Froude number is exactly one at the head of a "perfect" weir.

Analytical description

The solution can be split in four regions as indicated in figure 5.9. The solution is determined by the boundary conditions q_{in} and h_{out}. In zones 1 to 3 energy conservation applies, in the transition from zone 3 to 4 only momentum conservation is applicable because energy is lost in the hydraulic jump.

The Froude number is given by

$$Fr = \frac{u}{\sqrt{gh}}. \tag{5.4}$$

The water level at the crest is critical, this means that $Fr = 1$ and can be calculated with

$$d_2 = \left(\frac{q_{in}^2}{g}\right)^{\frac{1}{3}}. \tag{5.5}$$

The water depth in zone 1 can be related to the water depth in zone 2 by energy head conservation,

$$d_1 + \frac{u_1^2}{2g} = d_2 + a + \frac{\left(\frac{q_{in}}{d_2}\right)^2}{2g}. \tag{5.6}$$

The water depth in zone 3 is related to the water depth in zone 4 by momentum conservation

which is written as

$$\frac{d_3 + d_4}{2} d_3 d_4 = \frac{q_{in}^2}{g}. \tag{5.7}$$

Numerical setup

In this test the lumped mass matrix is used at dry elements and the system was solved with Gaussian-elimination. The grid size was set to $\Delta x = 0.5\ m$ except for some refinement at the steps of the weir to $\Delta x = 0.1\ m$. The time step was set to $\Delta t = 0.05\ s$. The height of the weir was $a = 0.4\ m$, the incoming discharge was set to $q_{in} = 8\ \frac{m^3}{s}(/m)$ and the downstream water depth was set to $h_{out} = 2.5\ m$.

Results and discussion

The numerical results are shown in figure 5.10. In table 5.1 the analytical results are compared to the numerical results in the four different zones. The analytical and numerical depths do not differ much. At the top of the weir where the flow gently changes from subcritical to supercritical the Froude number goes through 1, as is shown in figure 5.11. It is clear that the scheme is able to represent the height of the hydraulic jump, the energy head and the critical water depth very well.

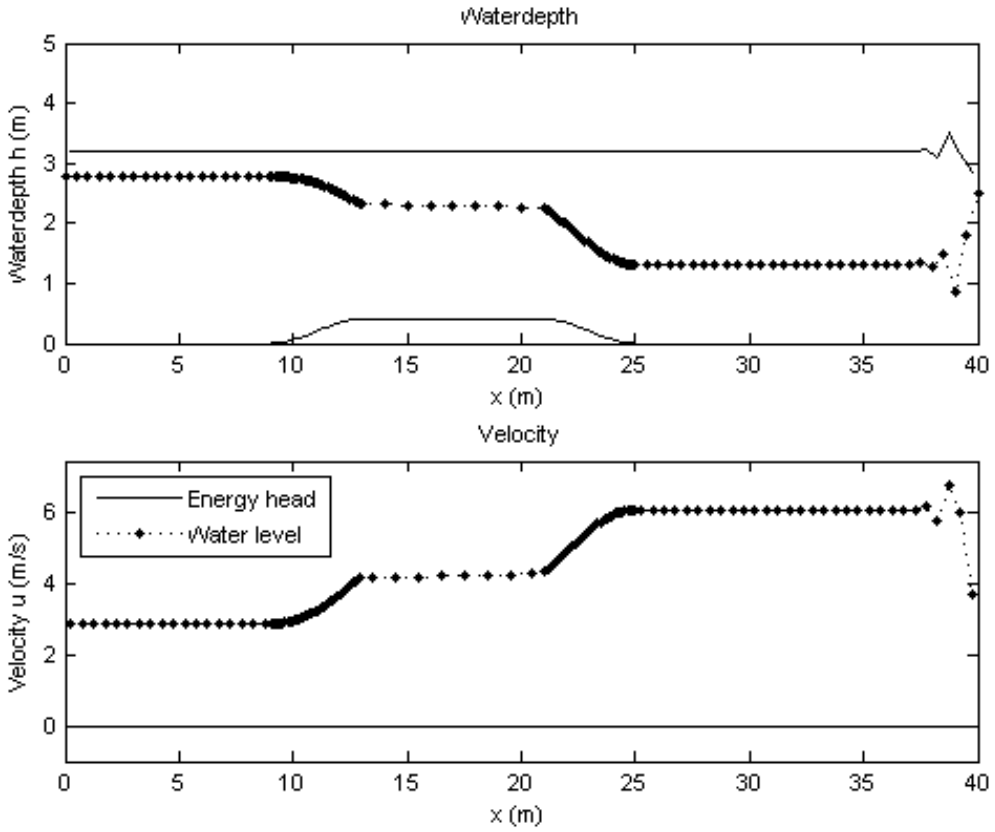


Figure 5.10: Results for flow over a long crested weir.

Zone	Analytical solution	Numerical solution	Absolute difference (%)
1	$d_1 = 2.78$	$d_1 = 2.81$	1.08
	$E_1 = 3.20$	$E_1 = 3.23$	0.94
2	$d_2 = 1.87$	$d_2 = 1.94$	3.74
	$E_2 = 3.20$	$E_2 = 3.23$	0.94
3	$d_3 = 1.35$	$d_3 = 1.33$	1.48
	$E_3 = 3.20$	$E_3 = 3.22$	0.63
4	$d_4 = 2.50$	$d_4 = 2.50$	0.00
	$E_4 = 3.09$	$E_4 = 3.10$	0.32

Table 5.1: Comparison of numerical results to analytical results.

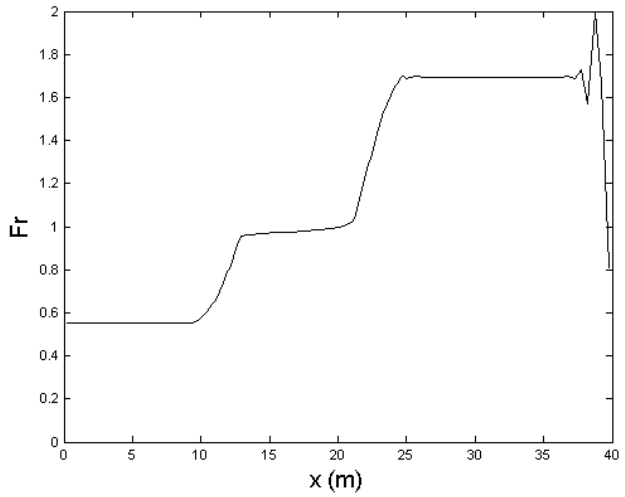


Figure 5.11: The Froude number for the flow over a long crested weir. The flow changes slowly from subcritical to supercritical.

5.1.3 Parabolic Basin

The analytical solution that describes the oscillating movement of an initially tilted plane in a parabolic basin is derived by Thacker [25]. For an example of this test, used to verify a wetting and drying algorithm in a finite element method, one can take a look at Bokhove [1]. Again the inviscid SWE in which friction is neglected are considered.

This test case is used to assess the ability of the model to handle sloping bottoms and the amount of numerical diffusion in the model. The difficulties that arise with this test case are

- that flooding and drying occurs at the same time, and
- that oscillations of the water surface are easily generated while the surface should remain planar.

Analytical description

The bottom level, water level and velocity are given respectively by equations (5.8), (5.9) and (5.10).

$$z(x) = -D_0 \left(1 - \frac{x^2}{L^2}\right) \tag{5.8}$$

$$h(x,t) = 2\eta D_0 cos(\omega t) \left(x - \frac{\eta}{2L} cos(\omega t)\right) \tag{5.9}$$

$$u(x,t) = -\eta\omega sin(\omega t) \tag{5.10}$$

With

$$\omega = \left(\frac{2gD_0}{L^2}\right)^{\frac{1}{2}}$$

D_0, the maximum depth of the basin

$2L$, the length of the water surface when horizontal

η, a coefficient that determines the initial amplitude

Numerical setup

This test is performed three times, twice solved directly and once using BiCGSTAB. In each case the mass matrix was lumped at dry elements and $d_{u=0} = 10^{-3}\ m$. The third test was with a higher resolution. The test case related parameters are set to $D_0 = 50\ m$, $L = 10\ m$ and $\eta = 0.1$. Two grids are used for this test, a uniform grid with $\Delta x = 0.5\ m$ and one nonuniform grid for the last test where $\Delta x = 0.1\ m$ for $-11 \leq x_i \leq -9$ and $9 \leq x_i \leq 11$ and $\Delta x = 0.5\ m$ everywhere else. The time step was set to $\Delta t = 0.05\ s$.

Results and discussion

In figure 5.12 the numerical solution compared to the analytical solution is shown at $t = 2T$. It can be seen that the solution suffers a lot from numerical damping. The numerical damping in the system is proportional to Δt and by reducing the time step size a solution with less damping is obtained. The damping in the system is not (solely) caused by the wetting and

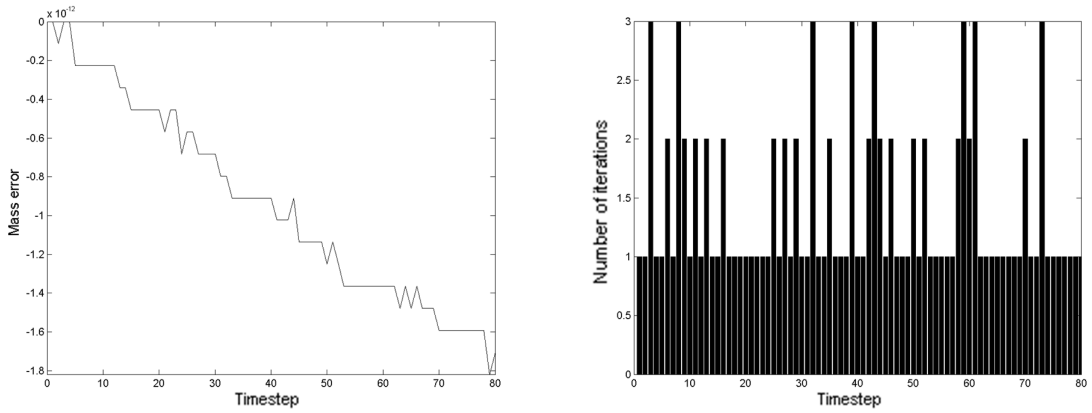


Figure 5.13: The mass error as a function of the time step and the number of iterations needed per time step for the parabolic basin test, the average number of iterations is 1.41. The computational parameters were set to $\Delta t = 0.05\ s$, M_L used at the dry nodes, solved directly and $d_{u=0} = 10^{-3}\ m$.

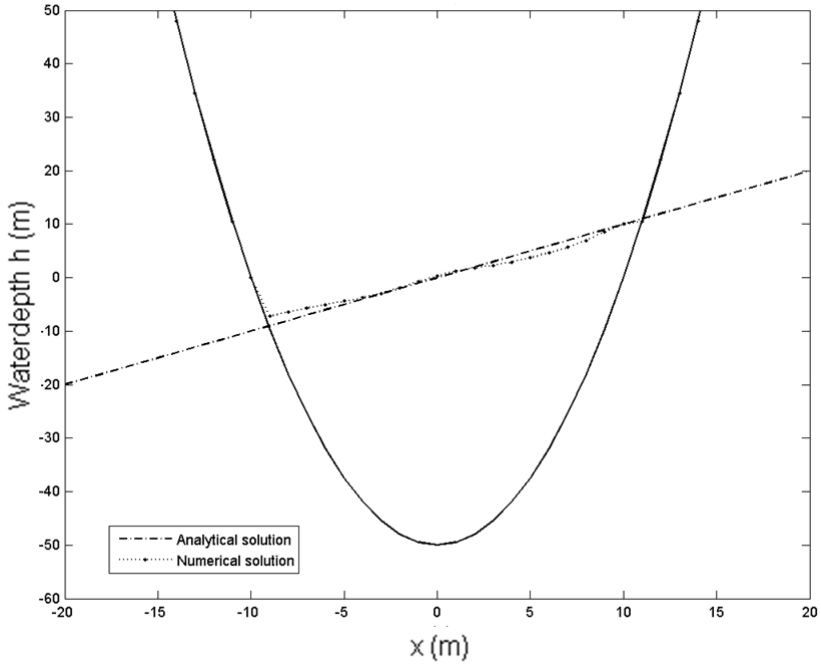
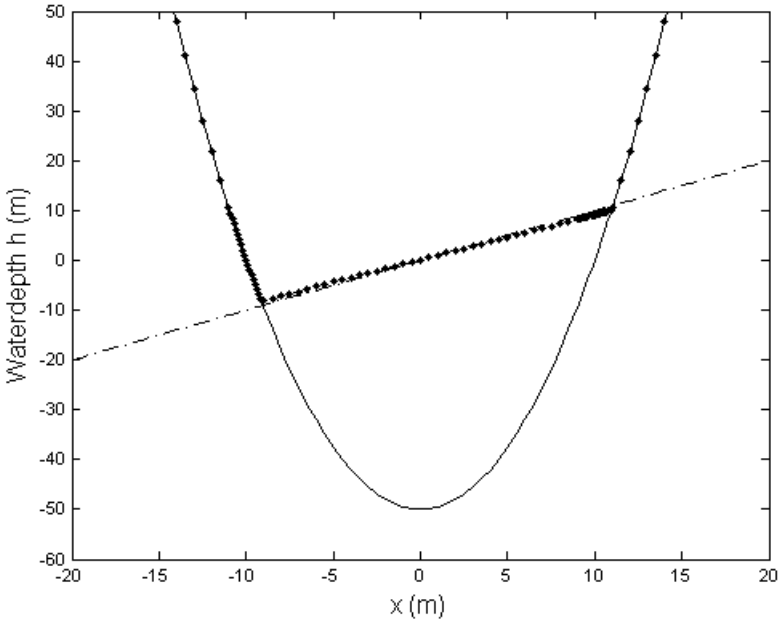
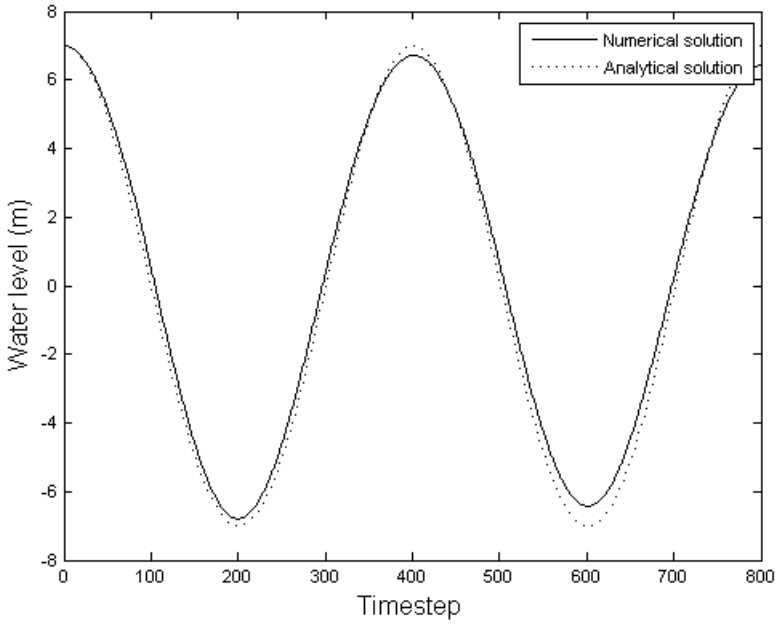


Figure 5.14: The analytical solution for the parabolic basin compared to the numerical solution at $t = 2T\ s$ with $\Delta t = 0.005\ s$, M_L used at the dry nodes, solved directly and $d_{u=0} = 10^{-3}\ m$.



(a)



(b)

Figure 5.15: The analytical solution for the parabolic basin compared to the numerical solution with $\Delta t = 0.005\ s$, $\Delta x = 0.1\ m$ in wetting and drying region, M_L used at the dry nodes, solved directly and $d_{u=0} = 10^{-3}\ m$. (a) The surface elevation for the basin at $t = 2T\ s$. (b) The surface elevation at $x = 7\ m$.

drying procedure since for a standing wave in a basin with vertical walls the motion is damped quite fast as well.

In figure 5.14 the solution is given, at $t = 2T$, with a time step of $\Delta t = 0.005\ s$. The performance of the numerical wetting and drying algorithm is good. The mass error is at machine accuracy and the average number of iterations per time step is only 1.41, see figure 5.13. The scheme is able to handle the sloping bottom, however some wiggles may be generated near the dry wet interface. Near the shoreline there can be a kink in the water surface. When an element becomes dry this kink suddenly changes shape and this sudden movement translates through the domain. The remedy is found in grid refinement near the shore. In figure 5.15(a) it can be seen that no wiggles occur. In figure 5.15(b) the water level variation at $x = 7\ m$ is plotted. Although visually in figure 5.15(a) the reduction of the time step has resolved the damping problem it can be seen in figure 5.15(b) that there is still some damping in the system.

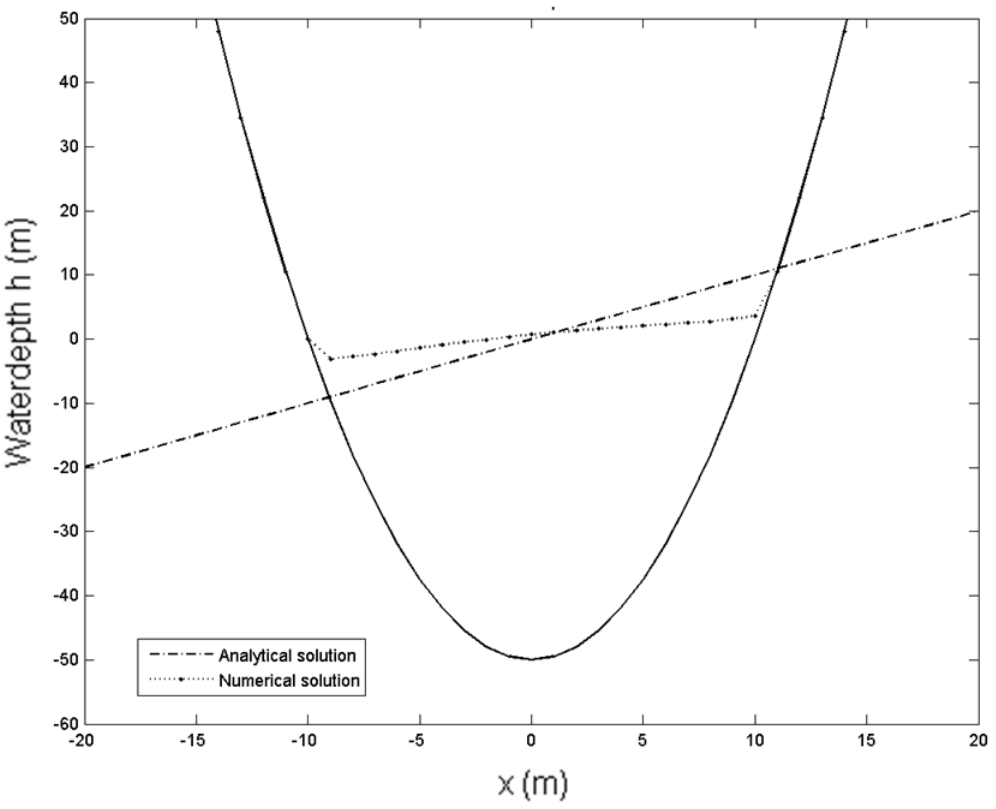


Figure 5.12: The analytical solution for the parabolic basin compared to the numerical solution at $t = 2T\ s$ with $\Delta t = 0.05\ s$, M_L used at the dry nodes, solved directly and $d_{u=0} = 10^{-3}\ m$.

BiCGSTAB does not perform well on this test case. The occurrence of bifurcations makes the calculation slow. Moreover, bifurcations occur in which none of the found solutions are mass conservative. The performance can be improved by using a smaller tolerance and increasing the maximum number of iterations, yet bifurcations can still occur. In fact using

d_{min} instead of lumping the mass matrix against matrix singularities results in the most improvements. No bifurcations occur and mass conservation is improved. The results are given in figure 5.16. The average number of iterations per time step might seem to be more than in figure 5.12, but it is only 1.52. In case of a large matrix this will probably be compensated by the use of BiCGSTAB.

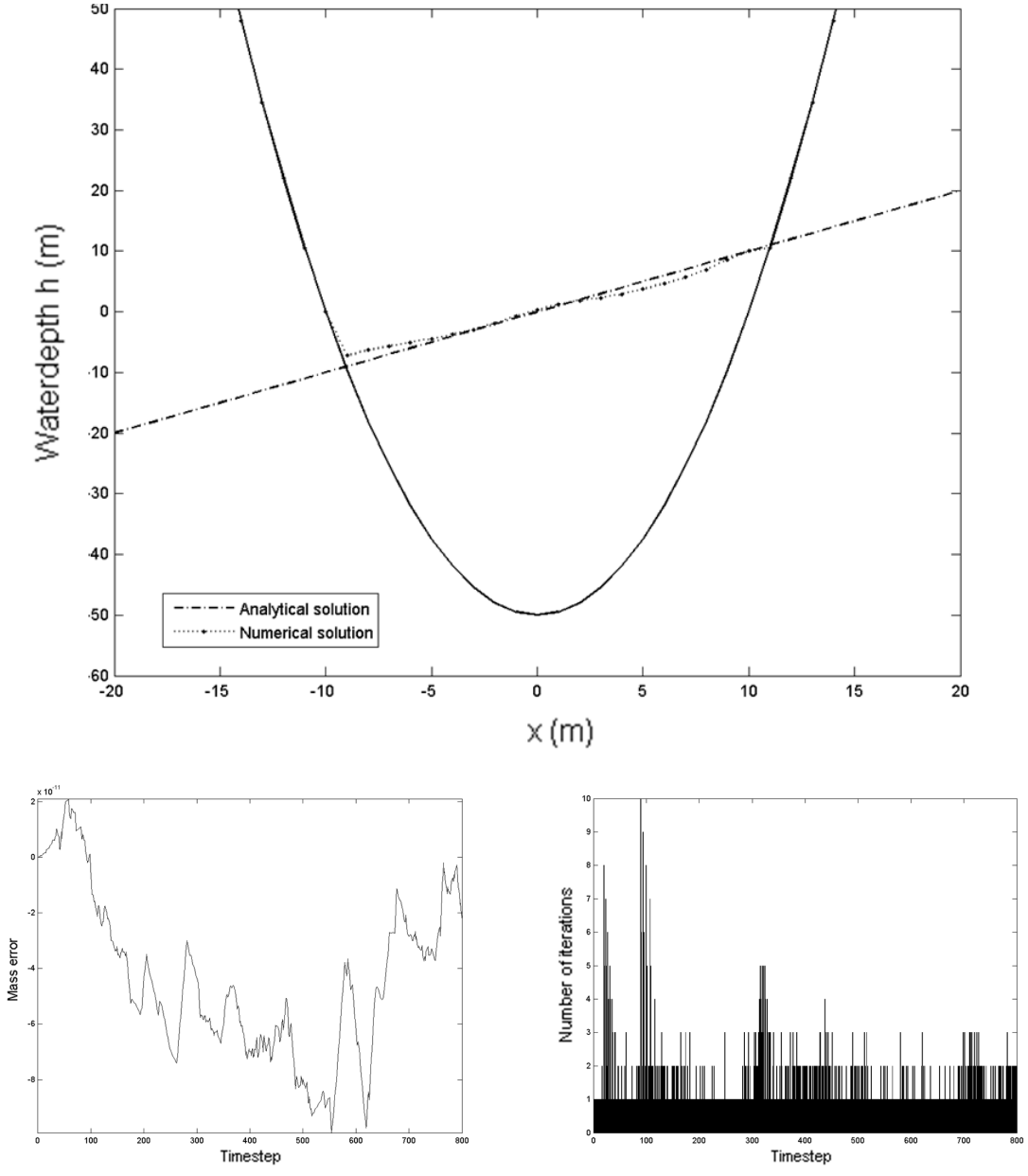


Figure 5.16: The analytical solution for the parabolic basin compared to the numerical solution at $t = 2T\ s$ with $\Delta t = 0.005\ s$ with $d_{min} = 10^{-4}\ m$, BiCGSTAB to solve the system and $d_{u=0} = 10^{-3}\ m$, the average number of iterations is 1.52.

5.1.4 Waves on a sloping beach

In this test a sinusoidal wave climbs up a sloping beach without breaking. The initial shape and velocity distribution determine whether a given wave will break. The analytical solution to this problem is derived by Carrier and Greenspan [7]. While Carrier and Greenspan derive their solution to determine a breaking criterion in the first place, their solution has been used numerous times to verify wetting and drying problems, amongst others by Stelling and Zijlema [23] and Bokhove [1].

This test case is particularly suited to verify the accuracy of the calculated shoreline movement and the correct representation of the location of the nodes of the standing wave.

Analytical description

Carrier and Greenspan come to their solution by using dimensionless quantities and writing the solution in terms of the independent variables λ and σ.

$$\begin{aligned} \lambda &= 2\,(v+t) \\ \sigma &= 4c \end{aligned} \tag{5.11}$$

Where v, t and c are respectively the dimensionless velocity, time and propagation speed. The solution is based on the inviscid shallow-water equations in which friction is neglected. The velocity, water surface elevation, distance and time with respect to these variables are then written as (in dimensionless form)

$$\begin{aligned} v &= \sigma^{-1}\phi_\sigma\,(\sigma,\lambda)\,, \\ x &= \frac{\phi_\lambda}{4} - \frac{\sigma^2}{16} - \frac{v^2}{2}, \\ \eta &= \frac{\phi_\lambda}{4} - \frac{v^2}{2}, \\ t &= \frac{\lambda}{2} - v, \text{ with} \\ \phi &= AJ_0\,(\sigma)\cos\,(\lambda)\,. \end{aligned} \tag{5.12}$$

Where J_0 is the Bessel function of the first kind.

Numerical setup

In this case all the solution methods discussed in Chapter 4.2 do not give a proper solution except for the solution with a minimum value for the cell averaged water depth in combination with a direct matrix solver. The value of a minimum element averaged depth is set to $d_{min} = 10^{-5}\ m$. The parameters used are the same as used by Stelling and Zijlema [23] and Madsen *et al.* [19]. A sinusoidal wave with a wave period 10 s and a wave height of 0.006 m propagates onto a beach with a slope of $1:25$, the maximum water depth is 0.5 m. The step size is $\Delta x = 0.04\ m$ and the time step is $\Delta t = 0.01\ s$.

Results and discussion

The results for this test are presented in figure 5.17, 5.18 and 5.19. The average number of iterations is surprisingly low, 1.15, considering that using d_{min} usually results in a relatively

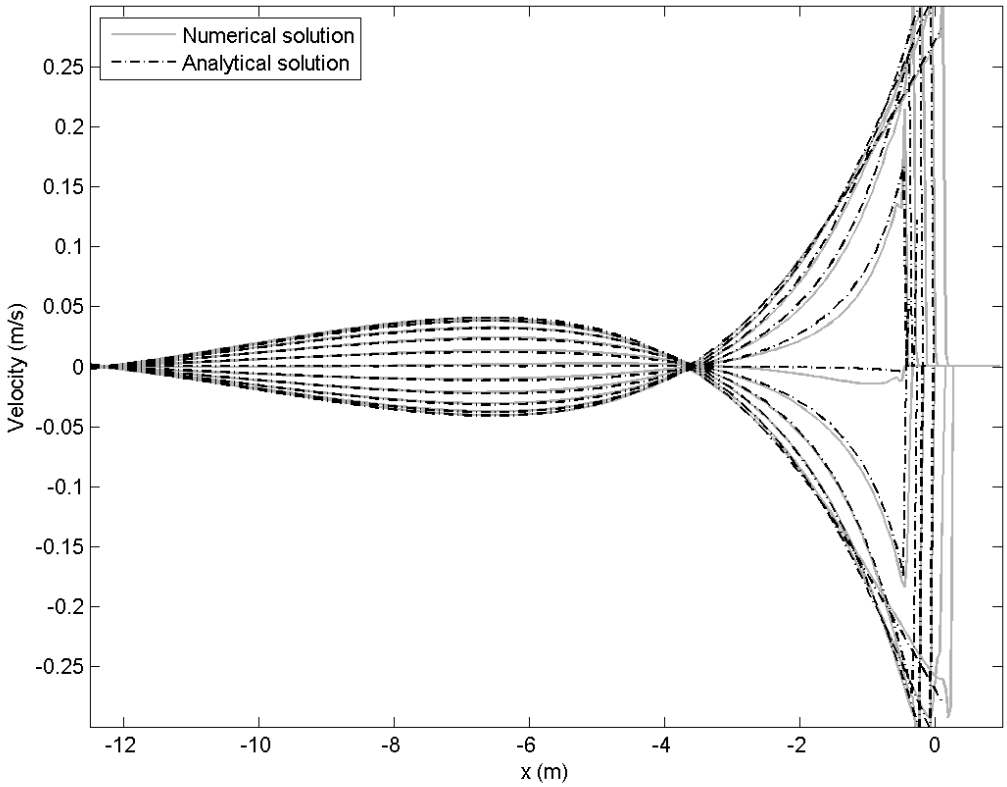


Figure 5.18: The analytical velocity compared to the numerical velocity for the Carrier and Greenspan test with $d_{min} = 10^{-5}\ m$, solved direct and $d_{u=0} = 10^{-5}\ m$.

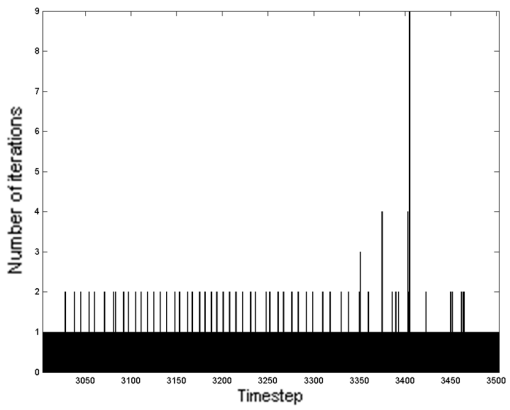


Figure 5.19: The iteration behavior for the Carrier and Greenspan test, the average number of iterations is 1.15.

high number of iterations. The model is quite well able to represent the wave shape, the location of the node and the wave velocity, but only for a relatively small grid size. For the parameters considered some irregularities at the shoreline are already observed for $\Delta x > 0.1$ m. The value of d_{min} also influences the solution considerably. For larger values than $d_{min} = 10^{-5}$ m a small bore develops at the wet/dry interface. However, if the velocity is not limited for small water depths, large and irregular velocity peaks occur at the dry wet interface.

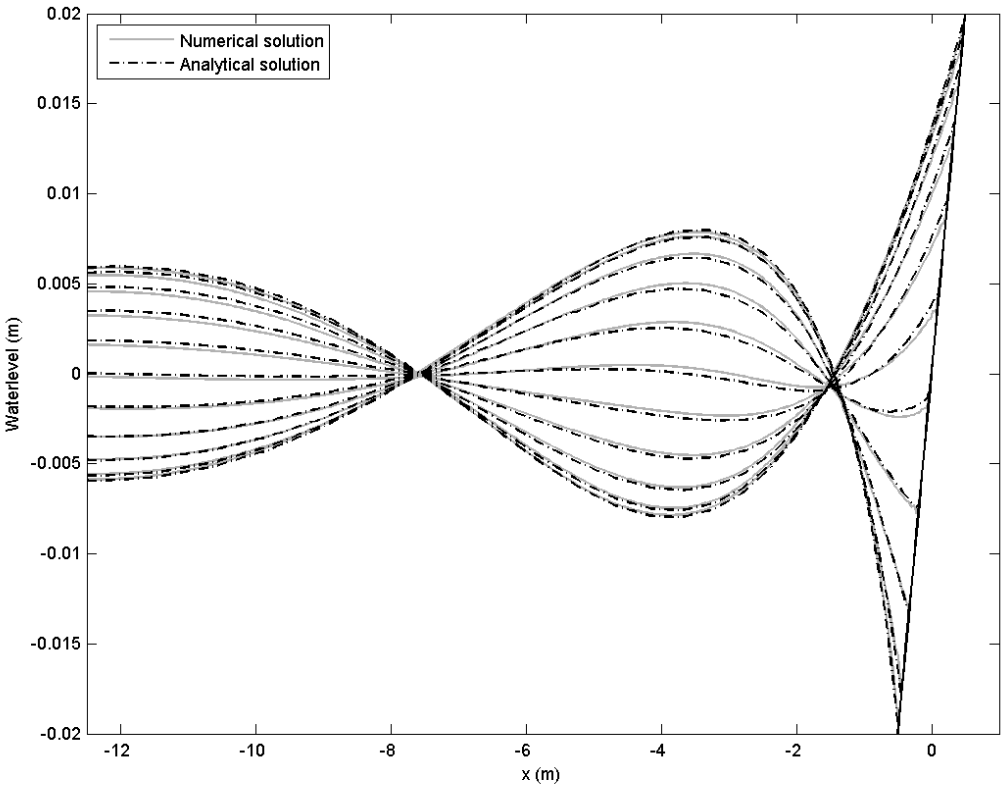


Figure 5.17: The analytical water level compared to the numerical water level for the Carrier and Greenspan test with $d_{min} = 10^{-5}$ m, solved direct and $d_{u=0} = 10^{-5}$ m.

5.2 2D Test Cases

In this section several two-dimensional numerical test results are compared to analytical solutions as well as to measurements. First a standing wave in a parabolic basin is compared to an analytical solution. It is a test with relatively small gradients at the wet/dry interface and the dry area is relatively small compared to the wet area. Second a parabolic water mass is released on a horizontal plane which is again compared to an analytical solution. In this case the gradient at the wet/dry interface is significantly larger than in the first test and the ratio wet area to dry area is approximately one.

After that the model is compared to measurements by a test in which a solitary wave runs

upon a conical island. This test bears quite some resemblance to the parabolic basin test. It also has relatively small gradients at the wet/dry interface and the wet area is large compared to the dry area. The only difference is that in this test friction is included. It can be seen as a logical next step after a good performance on the parabolic basin test. At last the models performance is compared to the measurements of a two dimensional dam break. In this case large gradients at the wet/dry interface exist as well as a large dry area.

All tests are performed with the use of a minimum element averaged depth, d_{min}, the velocity will be limited for a depth $d_{u=0}$, the system will be solved with BiCGSTAB, the mass matrix will be lumped for all elements which are partially dry or fully dry (depending on the test case), and an exit mode to prevent unnecessary iterations will be adopted.

5.2.1 Standing wave in a parabolic basin

In this test a parabolic shaped water surface is released in a parabolic basin inducing a standing wave pattern. Because friction is neglected this wave does not damp out. The analytical solution to this test was derived by Thacker [25]. This test is amongst others used by Casulli [8] and by Fuhrman and Madsen [10] to verify their wetting and drying procedures.

The difficulties of this test case encompass

- the generation of wiggles by the sudden transition of an element from dry to wet and vice versa,
- the occurrence of flooding and drying at the same time.

Analytical description

The bottom of the basin is described by

$$z(x,y) = -h_0 \left(1 - \frac{\sqrt{x^2+y^2}}{L^2}\right) \tag{5.13}$$

where h_0 denotes the maximum depth of the basin measured from the zero level and L is the radius of the basin measured from the zero level. The water depth is given by

$$H = h_0 \left[\frac{\sqrt{1-A^2}}{1 - A\cos(\omega t)} - \frac{r^2(1-A^2)}{L^2(1 - A\cos(\omega t)^2)}\right] \tag{5.14}$$

where A is given by

$$A = \frac{(h_0+n_0)^2 - h_0^2}{(h_0+n_0)^2 + h_0^2} \tag{5.15}$$

and n_0 is the initial surface elevation at $x, y = 0$.

Numerical setup

This test is performed with a minimum element averaged value of $d_{min} = 10^{-4}\ m$, the mass matrix is lumped for dry and partially dry elements and the velocity is set to zero for $d_{u=0} = 10^{-4}\ m$. For this test the test constants are taken $h_0 = 50\ m$, $n_0 = 5$ and $L = 500\ m$. The total number of elements used is 11686, the approximate area of an element is 625 m^2. In the wetting and drying region, a refinement was applied up to an approximate element size

of 125 m^2. The resulting mesh is shown in figure 5.20. The time step was set to $\Delta t = 0.5$ s. In the end a comparison is made to a case in which the mass matrix is lumped everywhere, all other settings are the same as described above.

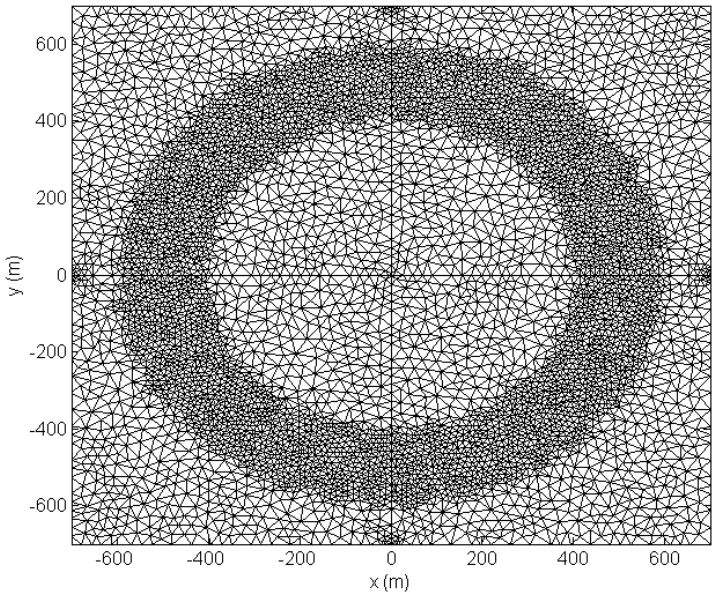


Figure 5.20: Computational grid for the two-dimensional parabolic basin test

Results and discussion

In figure 5.21 the numerical results are compared to the analytical results. The results are quite similar. Grid refinement in the wetting and drying region is necessary to minimize the occurrence of wiggles. However it is obvious that the scheme suffers from numerical damping. At first the analytical solution and the numerical solution lay nicely on top of each other, while after already less than a half period a little difference is visible between the numerical and the analytical solution. The average number of iterations per time step is 2.46.

The periodic character of the solution is nicely visible by the iteration behavior, figure 5.22. $T = 50.15$ s, which is approximately 100 time steps. The moments at which the velocity is zero are at time step 50, 100, 150, 200, 250 and 300 become visible by multiple subsequent time steps of only 2 iterations or even 1. The maximum speed of flooding occurs approximately at time steps 25, 125 and 225. Around these time steps 3 to 4 iterations are needed. The maximum speed of drying is approximately at time steps 75, 175 and 275, and results in several time steps in which 3 iterations are necessary. It is clear that the speed of flooding and drying does influence the iteration behavior and apparently flooding is more difficult than drying.

When the mass matrix is lumped only for fully dry elements the system blows up after a few time steps. When the mass matrix is lumped everywhere the results are pretty much the same as for the case of lumping at dry elements and partially dry elements. However the occurrence of a mass error is less frequent, see figure 5.23, and the average number of

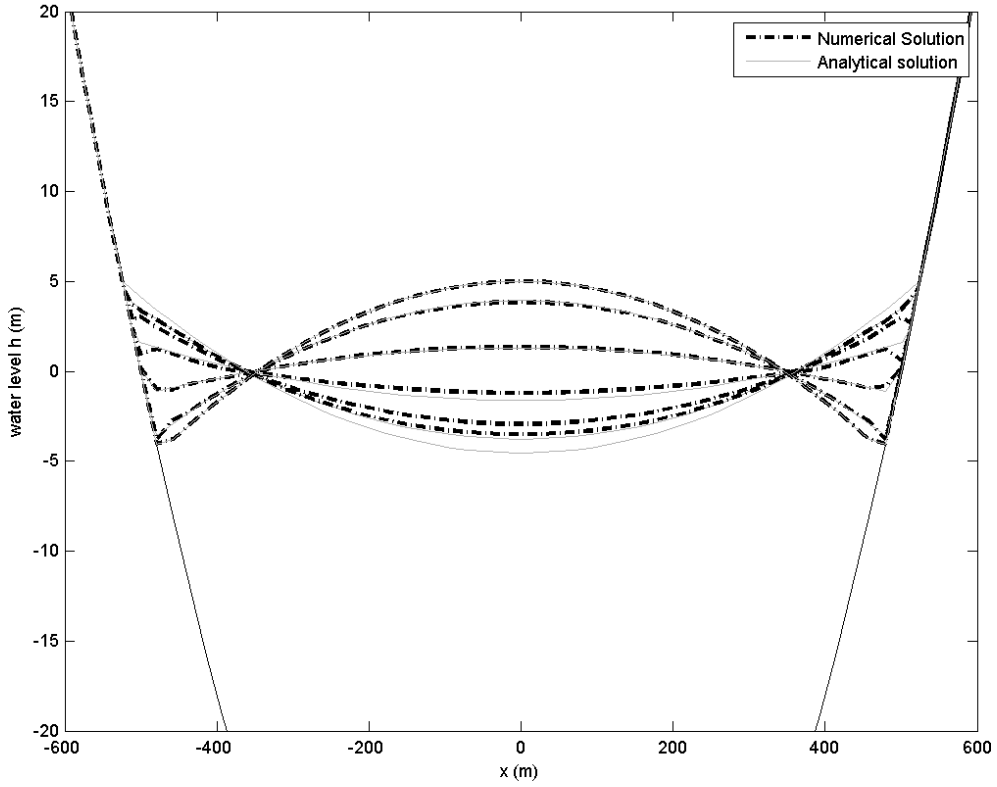


Figure 5.21: Numerical solution compared to the analytical solution at $t \approx 0$, $\frac{1}{10}T$, $\frac{2}{10}T$, $\frac{3}{10}T$, $\frac{4}{10}T$ and $\frac{5}{10}T$ s for the cross-section $y = 0$.

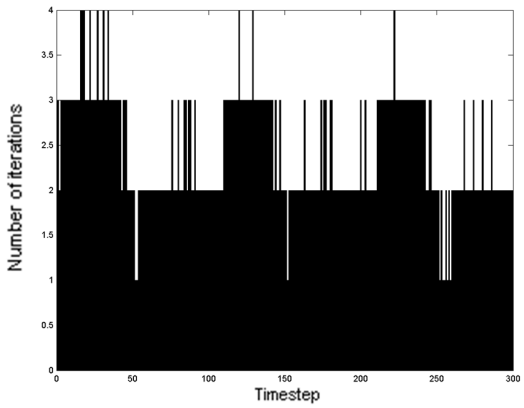


Figure 5.22: Iterations per time step for the standing wave in a parabolic basin, the average number of iterations per time step is 2.46.

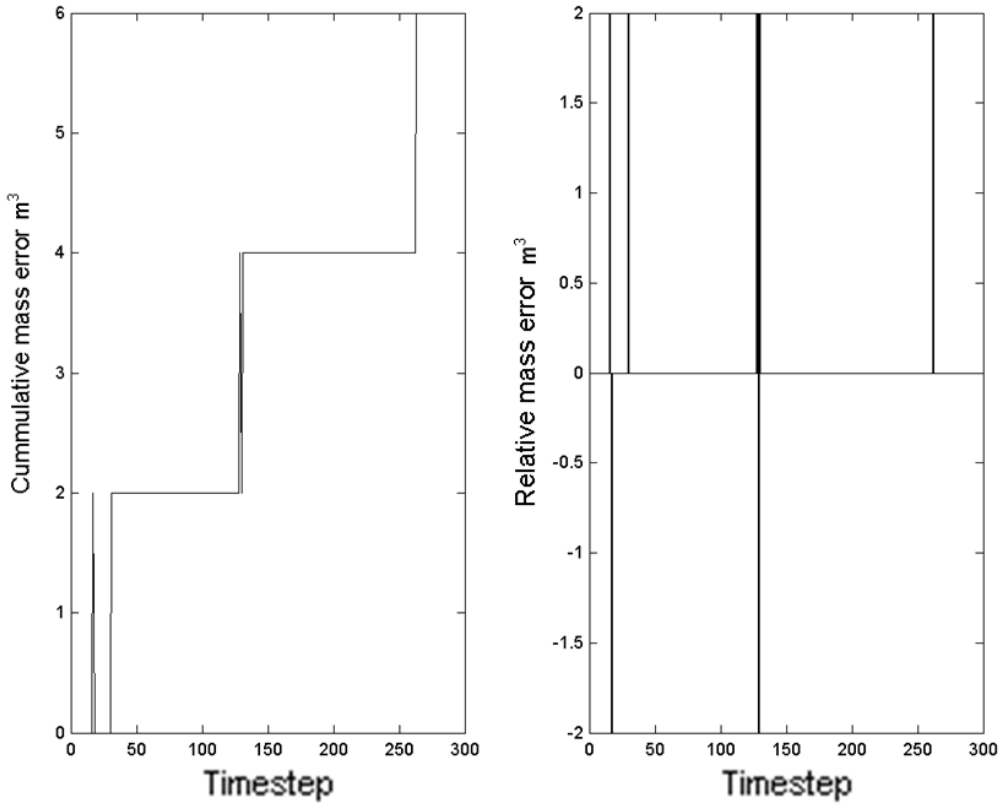


Figure 5.23: Mass conservation for the standing wave in a parabolic basin in case the mass matrix is lumped everywhere. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

iterations per time step is slightly reduced to 2.15. So in this case lumping everywhere does lead to computational improvements.

Mass conservation is not very satisfactory for this test as can be seen in figure 5.24. If one takes a close look at the mass conservation figures in section 5.1, figures 5.2 and 5.12 it is visible that at every time step the same absolute mass jump in the mass error occurs. In one dimension this was not very disturbing because these jumps were of $\mathcal{O}(10^{-14})$ to $\mathcal{O}(10^{-7})$, whereas the mass differences in this test are of $\mathcal{O}(1)$. However the percentage of the total water volume is only $10^{-5}\%$. Nevertheless a trend is clearly visible and in case of long computations this can lead to severe mass errors. In section 5.3 it is shown that this phenomenon is not a feature of the wetting and drying procedure.

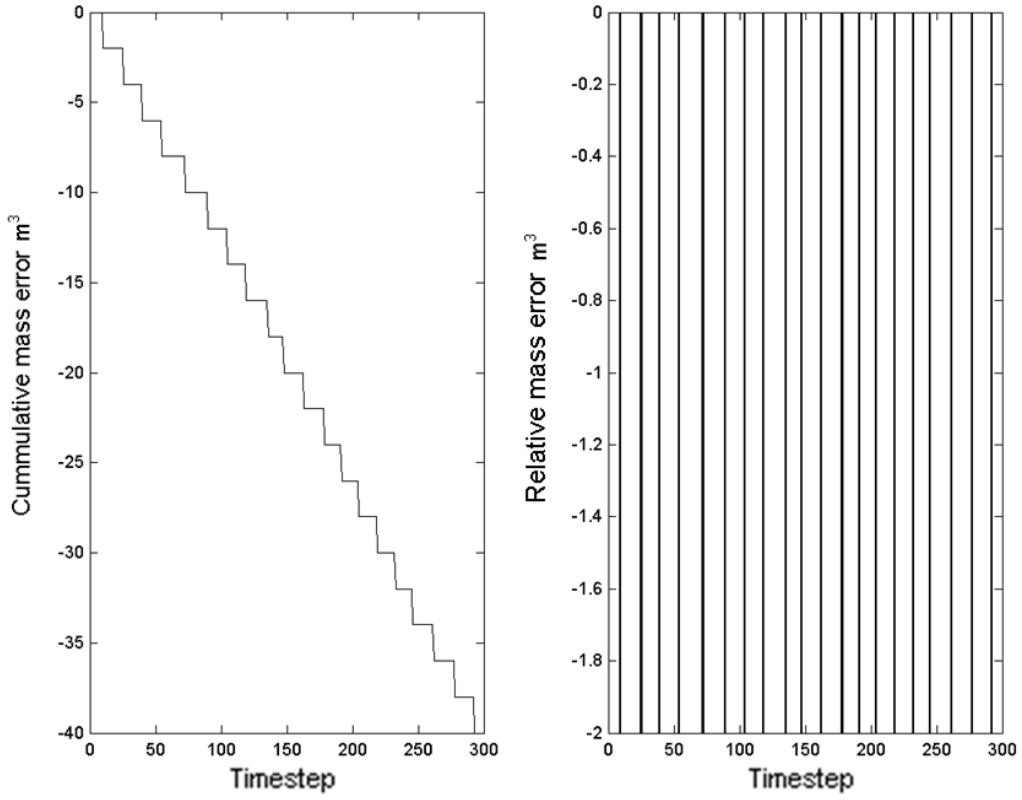


Figure 5.24: Mass conservation for the standing wave in a parabolic basin in case the mass matrix is lumped for dry and partially dry elements. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

5.2.2 Parabolic flood wave

A water mass with a parabolic shape which is initially at rest is released on a flat bed without friction. The analytical solution to this test was derived by Thacker [25].

The difficulty of this test case is that the interface gradient is very strong, causing the flooding process to go very fast. Moreover the water level at the dry nodes can become very

negative. As seen earlier on this could lead to many iterations per time step.

Analytical description

The initial shape of the water mass is described by

$$h_0 = \eta \left(1 - \frac{x^2 + y^2}{R_0^2}\right). \tag{5.16}$$

In which R_0 is the initial radius of the water mass and η is the initial height of the mount. The development in time of the mount is described by

$$h = \eta \left[\frac{T^2}{t^2 + T^2} - \frac{x^2 + y^2}{R_0^2}\left(\frac{T^2}{t^2 + T^2}\right)^2\right]. \tag{5.17}$$

In which T is the time after which the initial height η has been halved and is given by

$$T = \frac{R_0}{\sqrt{2g\eta}}. \tag{5.18}$$

Numerical setup

This test is performed with a minimum element averaged value of $d_{min} = 10^{-4}\ m$, the mass matrix is lumped whenever an element is completely dry and the velocity is set to zero for $H_j^n < 10^{-4}\ m$. The test parameters are set to $\eta = 0.5\ m$, $R_0 = \sqrt{10}\ m$ resulting in a half-period $T = 1.0096\ s$. The time step is taken $0.025\ s$, the total number of elements used is 1681 and the area of an element is $\Delta t = 0.125\ m^2$. The mesh used is shown in figure 5.25. In the end a comparison is made to the performance of the model for complete lumping and for lumping the mass matrix for dry and partially dry elements. The remaining settings are kept the same.

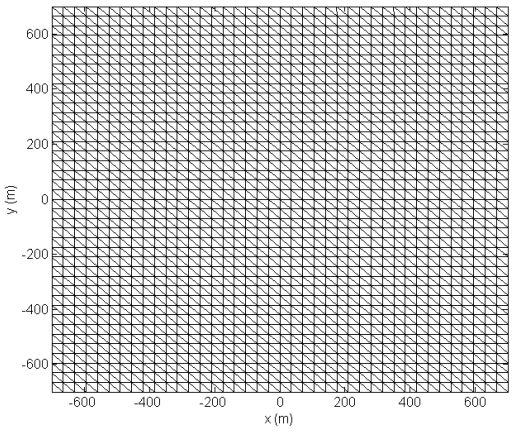


Figure 5.25: Computational grid for the parabolic water mass test.

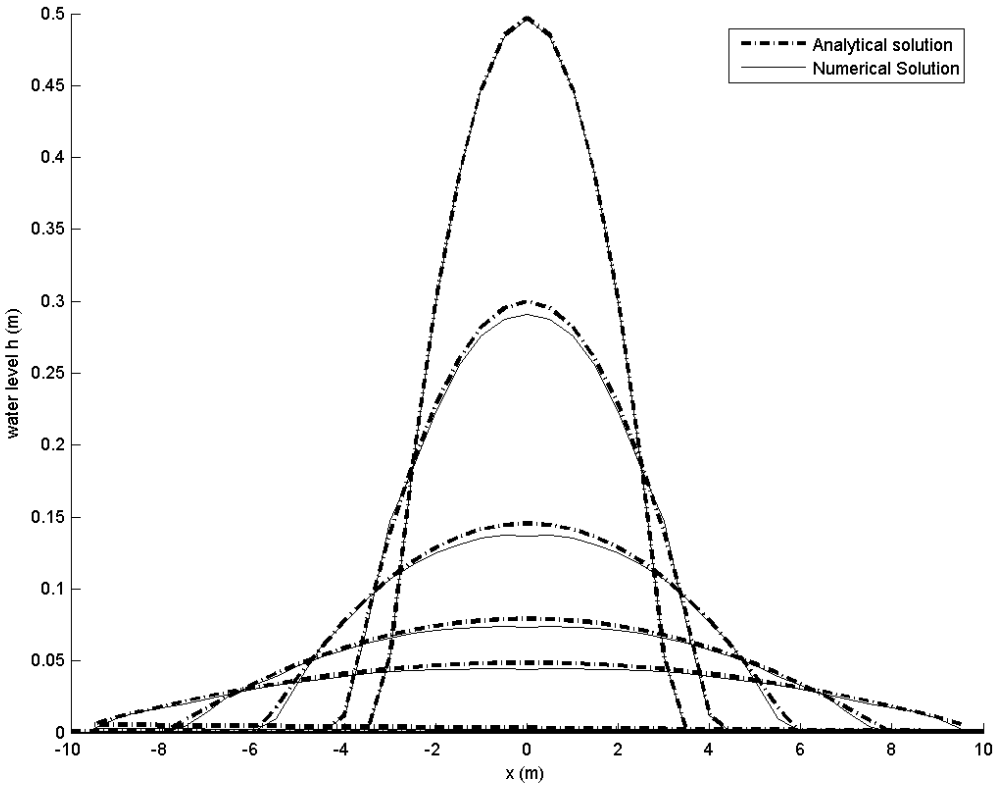


Figure 5.26: Numerical solution compared to the analytical solution at $t = 0$, 0.75, 1.5, 2.25 and 3 s, when lumping the mass matrix for dry elements only.

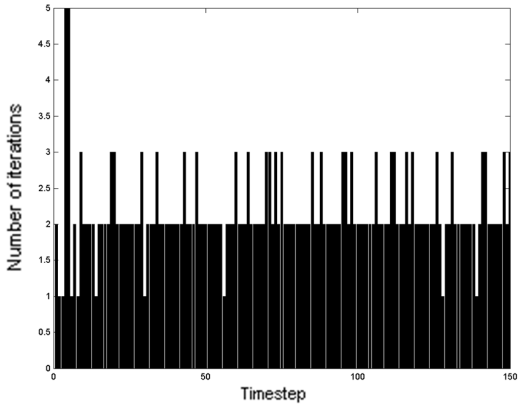


Figure 5.27: Iterations per time step for the spreading of a parabolic water mass when lumping the mass matrix for dry elements only, the average number of iterations per time step is 2.17.

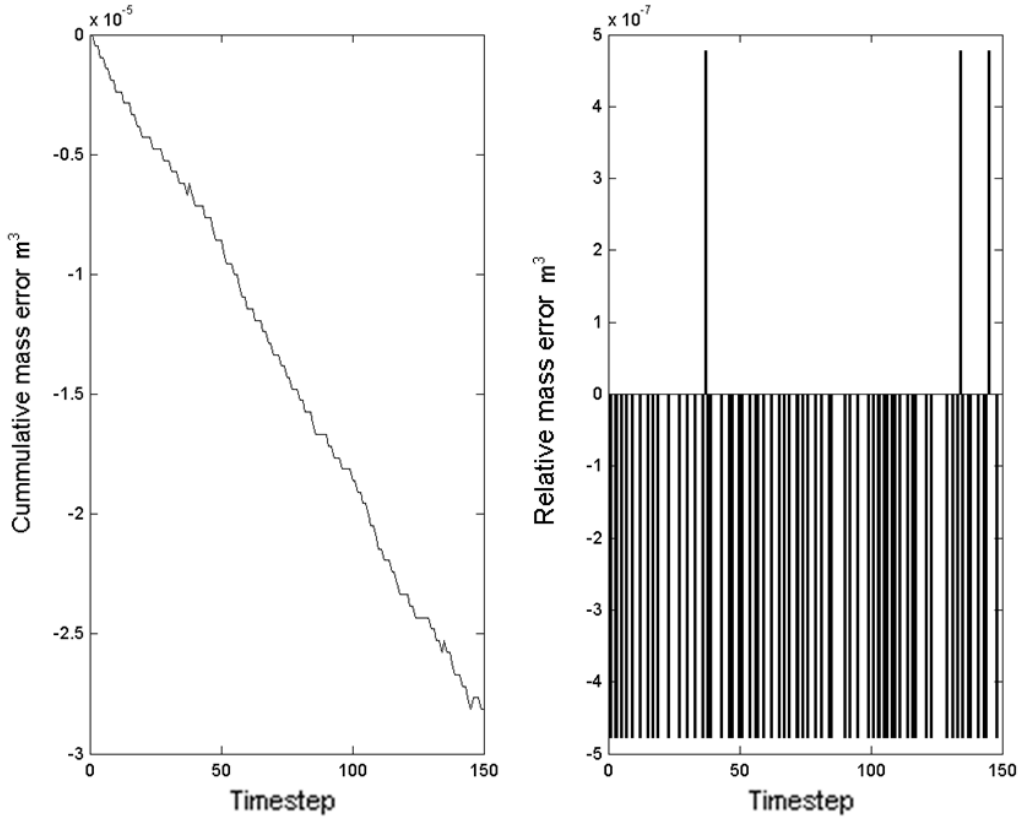


Figure 5.28: Mass conservation for the spreading of a parabolic water mass. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

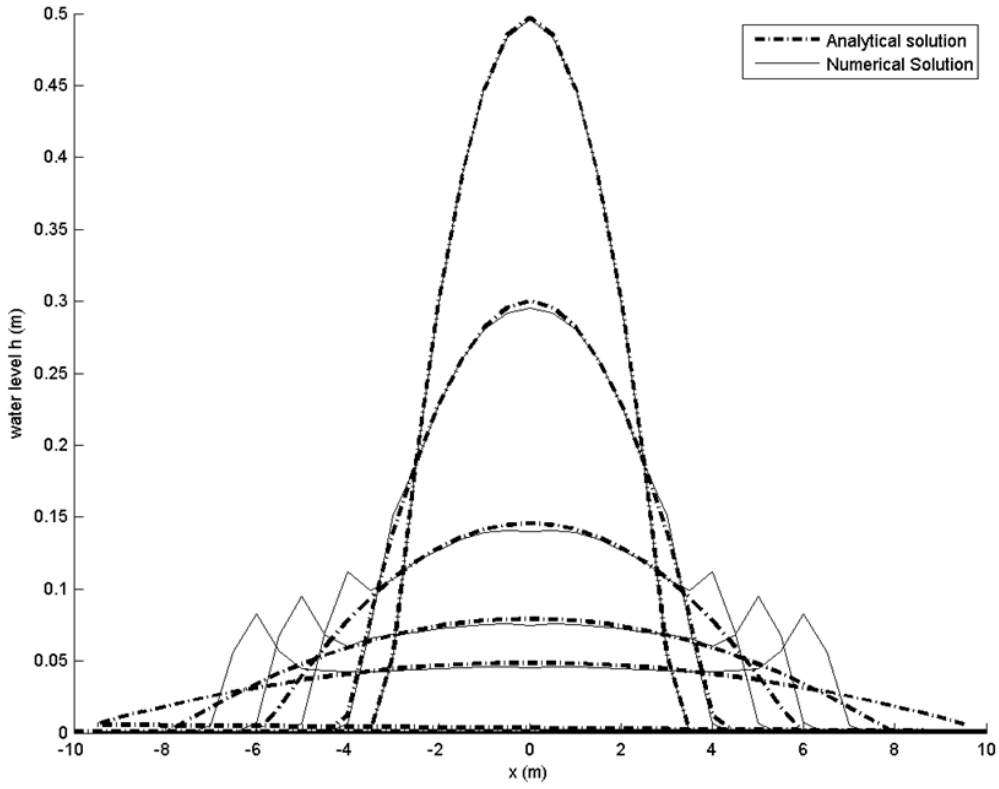
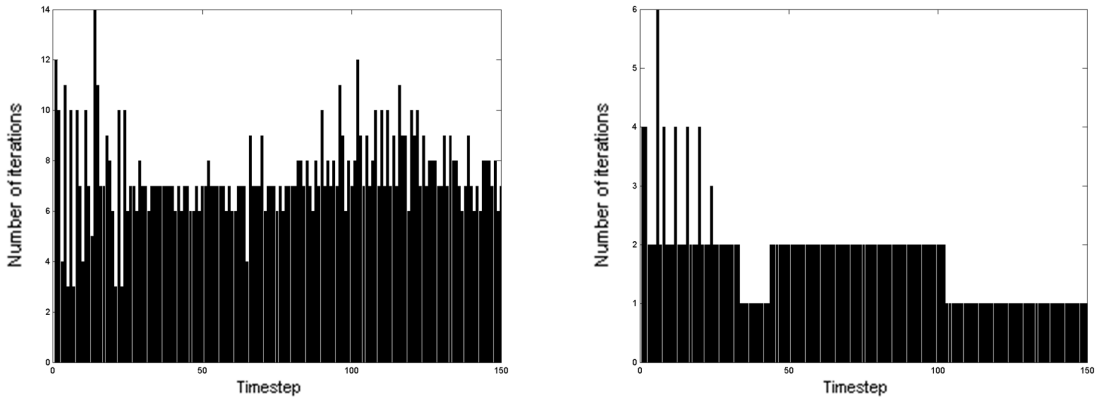


Figure 5.29: Numerical solution compared to the analytical solution at $t = 0$, 0.75, 1.5, 2.25 and 3 s in case of a lumped mass matrix.



(a) (b)

Figure 5.30: Iterations per time step for the spreading of a parabolic water mass. (a) Lumping at dry and partially dry elements, the average number of iterations is 7.47. (b) Lumping everywhere, the average number of iterations is 1.73.

Results and discussion

In figure 5.26 the results are shown for $t = 0$, 0.75, 1.5, 2.25 and 3 s. The models ability to represent this analytical solution is very good. The spreading of the numerical solution is a little bit larger than that of the analytical solution, as shown by the slightly lower top and the wider toe. This is not quite unexpected since the advection scheme is only first order accurate. The iteration behavior as shown in figure 5.27 is also very good, with an average of 2.17 iterations per time step it does not perform worse than the one-dimensional tests. What is clearly visible is the decreasing number of iterations with a decreasing gradient at the wet/dry interface or decreasing flooding velocity. However figure 5.28 shows that mass conservation is not quite as good as it was for one-dimensional tests. Again the error, if any, is of the same size. There is a clear trend visible and for long computations this could result in a serious mass error.

It is possible to lump the mass matrix for partially dry elements too. No instabilities occur, but the iteration behavior is seriously compromised as can be seen in figure 5.30(a), resulting in an average number of iterations per time step of 7.46. When using the lumped mass matrix everywhere a bore is created at the wet/dry interface as can be seen from figure 5.29. The average number of iterations per time step is slightly decreased to 1.73 when lumping the mass matrix everywhere, see figure 5.30(b). In these two figures the decreasing number of iterations for a decreasing flooding velocity is visualized. The mass conservation is practically the same for complete lumping of the mass matrix as for lumping only at fully dry elements. Here again lumping might be preferred with respect to computational effort. However, one should realize that the accuracy of the solution is seriously compromised.

5.2.3 Solitary wave runup on a conical island

In this test a solitary wave is generated by a wave maker and sent into a water basin in which a conical shaped island is located. This laboratory test is performed by the Coastal and Hydraulics Laboratory of the USACE. The results have been used as a source for multiple publications, amongst others Liu *et al.* [16], Briggs *et al.* [2] and Fuhrman and Madsen [10]. The tests were performed as a result of two tsunami events where unexpectedly large runup heights in the lee of small islands were observed. As it was felt that more knowledge was necessary on the parameters that influenced the runup process three different tests were carried out. In this study a comparison is made to only one of these measurements since the other two tests fall outside the application range of the shallow water equations.

Experimental setup

The test was performed in a rectangular basin of 30 m by 25 m as shown schematically in figure 5.31. The center of the island was located at $x, y = (15m, 13m)$. The slope of the island was 1 : 4 and the diameter of the basis of the island was 7.2 m. A wave maker was installed along the x-axis. Twenty seven gauges were installed to measure the surface elevation. Of eight of these gauges the data series were available, which are shown in figure 5.32. The still water level is indicated by the dashed line.

Gauges 1 to 4 measure the incoming generated solitary wave and are placed at a distance of a half wavelength from the foot of the island. Gauge 6 was placed at the foot of the island at the 270° transect. Gauges 9, 16 and 22 were all placed at 0.08 m depth at the 0, 90 and 270°

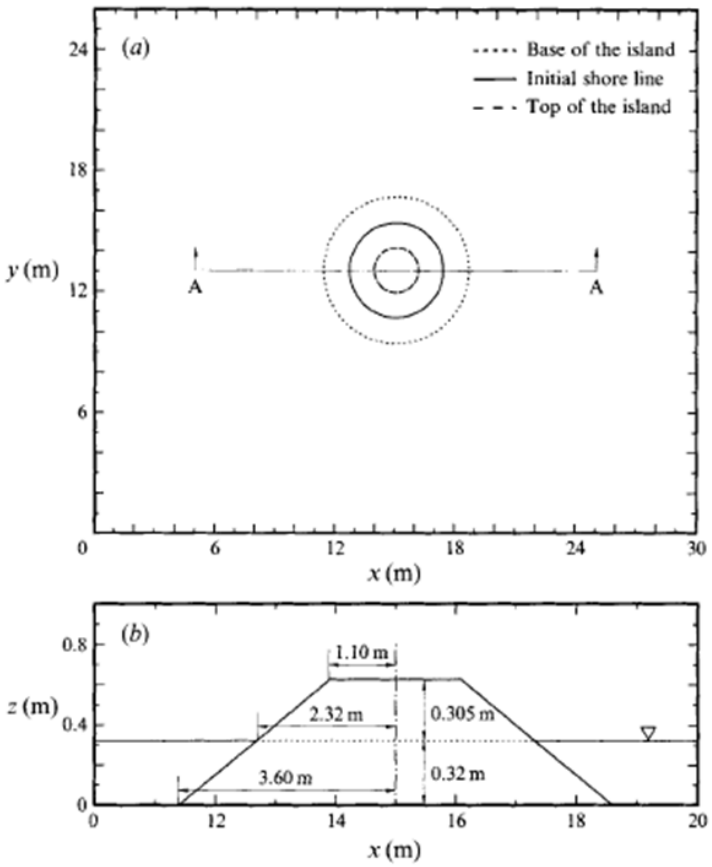


Figure 5.31: Experimental setup of the solitary wave runup on a conical island test. (a) Top view. (b) Cross-section A-A'. [16].

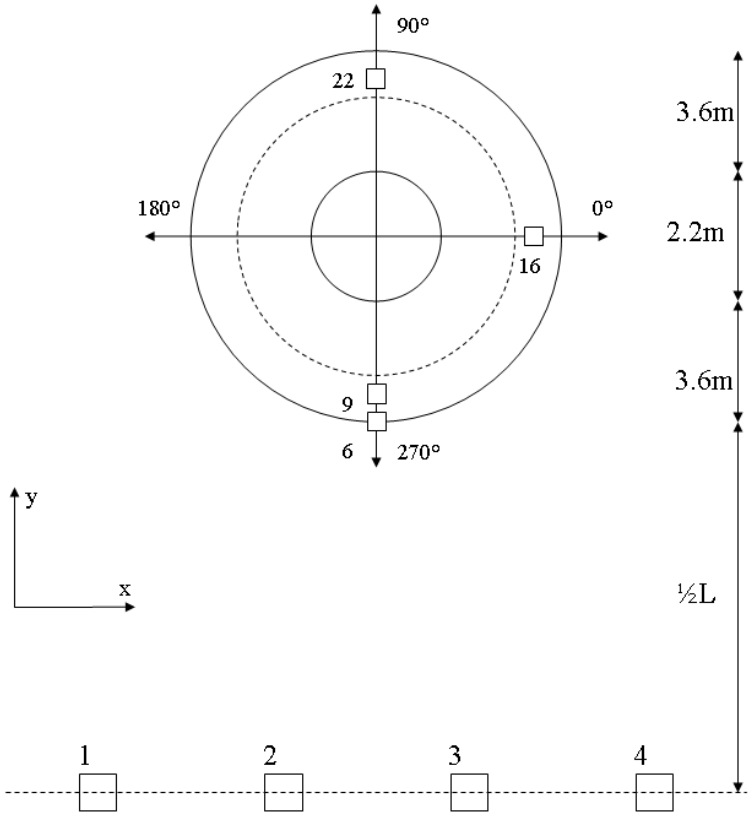


Figure 5.32: Location of the wave gauges.

transect. The still water depth was 0.32 m. The height to depth ratio of the solitary wave was $\varepsilon = 0.05$, resulting in a (solitary) wavelength of 6.55 m.

Numerical setup

Only part of the basin was modeled for the numerical computations. The lower boundary in x-direction was chosen at the line of gauges 1 to 4 such that the measured incoming wave could function as a boundary condition. A total width of the computational domain of 14.4 m was chosen as it was felt that the fully reflecting lateral boundaries would not have a significant influence on the solution at that distance from the island. This while assuming that the wave is perpendicular again to the sides of the basin at that distance. The time step used was 0.02 s. The grid consisted of 60126 elements of which the approximate area was 0.04 m^2 which changed gradually to an area of $2.25 \cdot 10^{-5}$ m^2 around the still water level as shown in figure 5.33. Mannings friction parameter is set to $n = 0.01$ $m^{-\frac{1}{3}}s$. The incoming wave that was measured was set to zero after 15 s since after that the signal was disturbed by the wave reflection on the island. The resulting boundary condition is shown in figure 5.34.

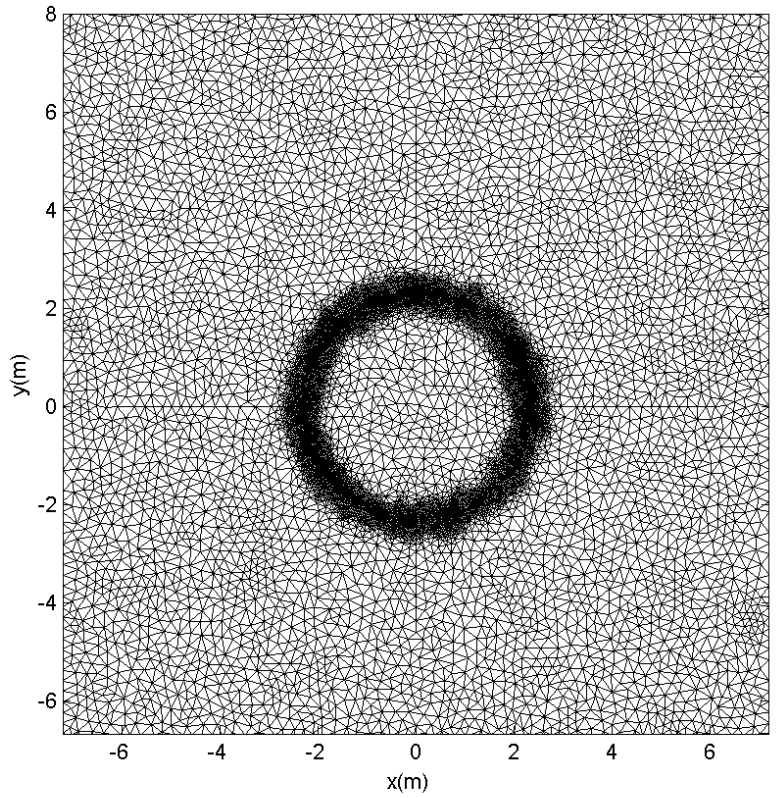


Figure 5.33: Computational grid for the solitary wave runup.

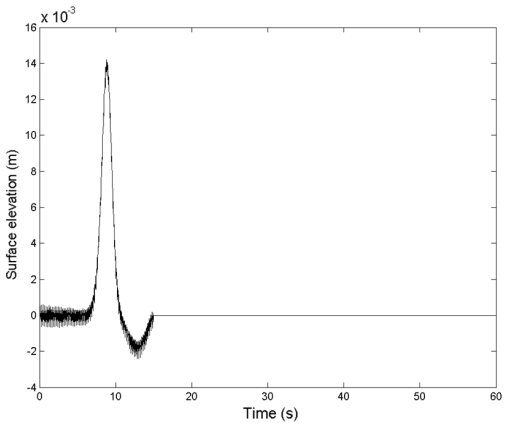


Figure 5.34: Boundary condition for the solitary wave.

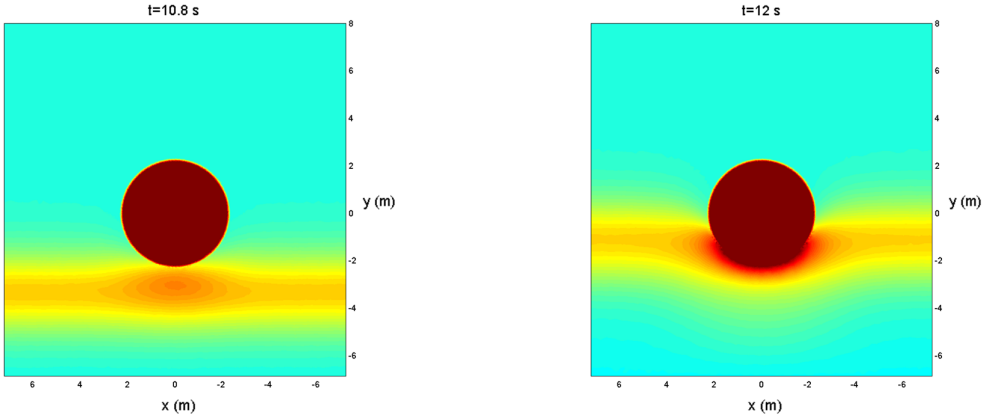
Results and discussion

When approaching the island the solitary wave starts to shoal, thus becoming steeper and shorter, as can be nicely seen in figure 5.35(a). Next refraction and diffraction cause the wave to bend around the island, see figure 5.35(b). The wave is (partially) reflected at the front of the island causing a depression, see figure 5.35(c). Where the two bent sides of the wave meet at the back of the island a larger runup is induced by wave-wave interference, see figure 5.35(d).

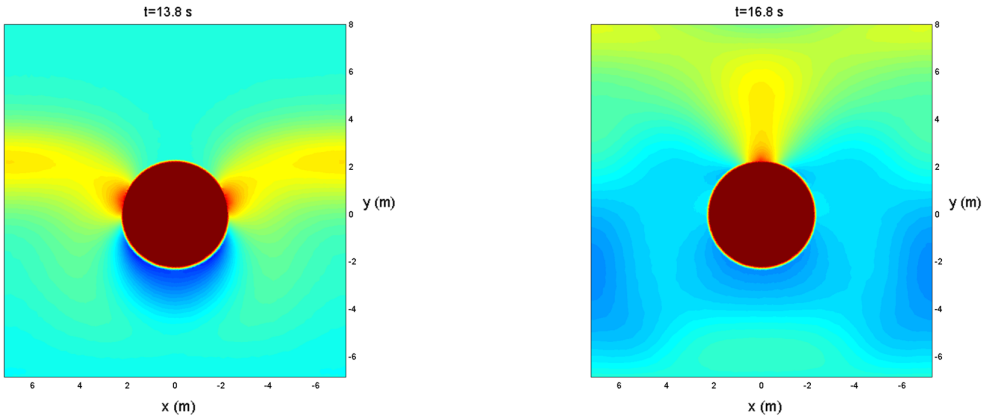
In figure 5.36 the numerical results compared to the measurements at the wave gauges are plotted. It can be seen that the model represents the arrival time and the height of the incoming wave very well. The wave is slightly steeper than measured which is not unexpected. Because the waves in the SWE travel with wave speed $c = \sqrt{gH}$ the top of the wave will travel faster than the lower part of the wave. In reality the applicability of the SWE comes to an end close to the island. With decreasing depth the vertical accelerations are no longer negligible, which causes the deviation in the steepness of the wave top. The reflected wave is not represented well by the numerical model, since the trough is not as deep as it should be according to the measurements. Nevertheless the shape is followed quite well only not reaching the same extrema. This is probably caused by the numerical damping that we have seen before in the test of the standing wave in the parabolic basin.

In figure 5.37 the number of iteration per time step are given. The average number of iterations per time step is 1.95.

The results with respect to runup are not as good as for the gauges. In figure 5.38(a) the horizontal runup is given. The gray dots indicate the initially dry nodes and the black dots indicate the nodes which remain dry throughout the entire simulation. The solitary wave is coming in from the left side of the figure. In the southwest area of the island the runup is much higher than at the north west side. On further investigation it is discovered that the water depth in this area is really small, $\mathcal{O}(10^{-3})$, and can be caused by a bias in the grid. In one dimension the problem of a thin layer shooting ahead of the wet/dry interface was already encountered. In one dimension it was solved by limiting the velocity for too small water depths. If we change the value of the water depth where we apply this for from $d_{u=0} = 10^{-4}\ m$ to $d_{u=0} = 10^{-3}\ m$ this does indeed solve the problem, figure 5.38(b). If



(a) (b)



(c) (d)

Figure 5.35: Numerical solution at times $t = 10.8$ (a) , $t = 12$ (b), $t = 13.8$ (c) and $t = 16.8$ (d) s.

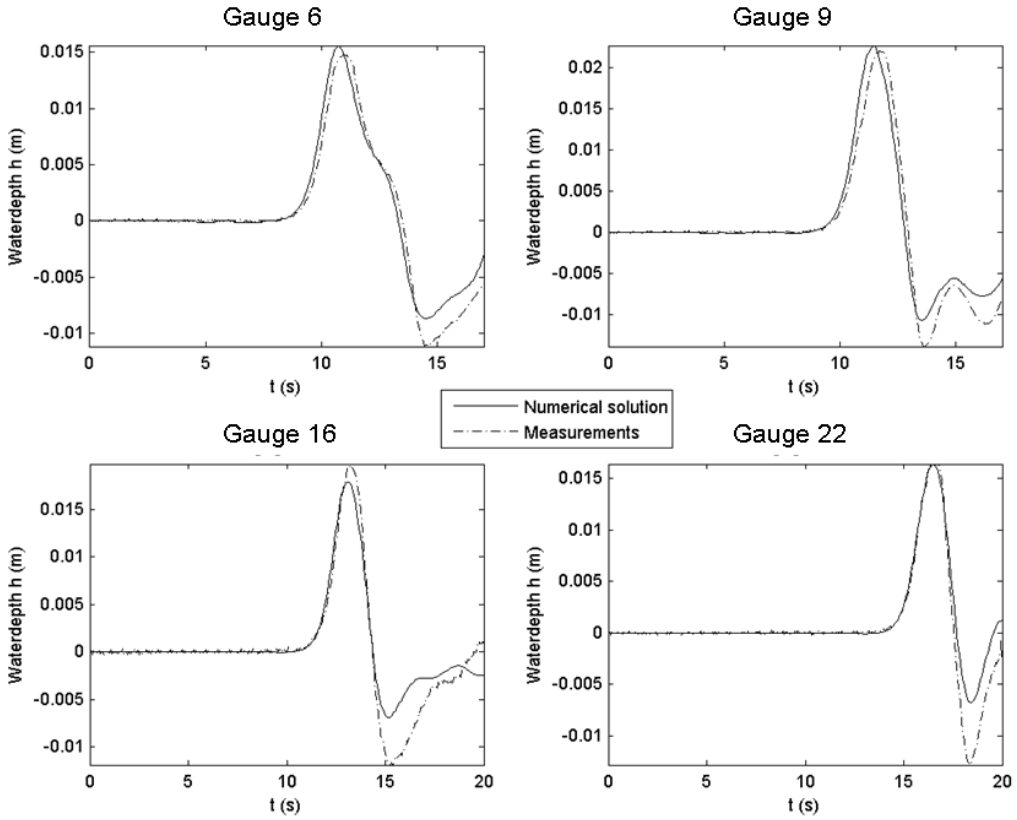


Figure 5.36: Numerical solution compared to the analytical solution at gauges 6,9,16 and 22 for the solitary wave runup on a conical island test.

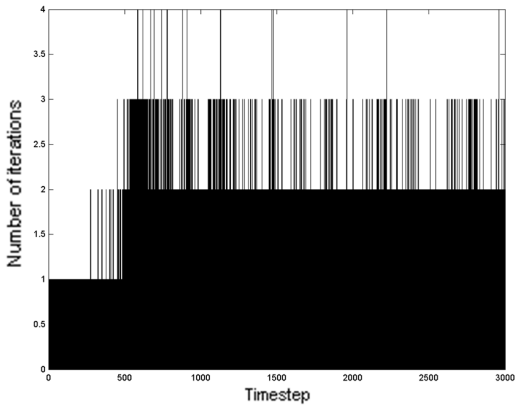
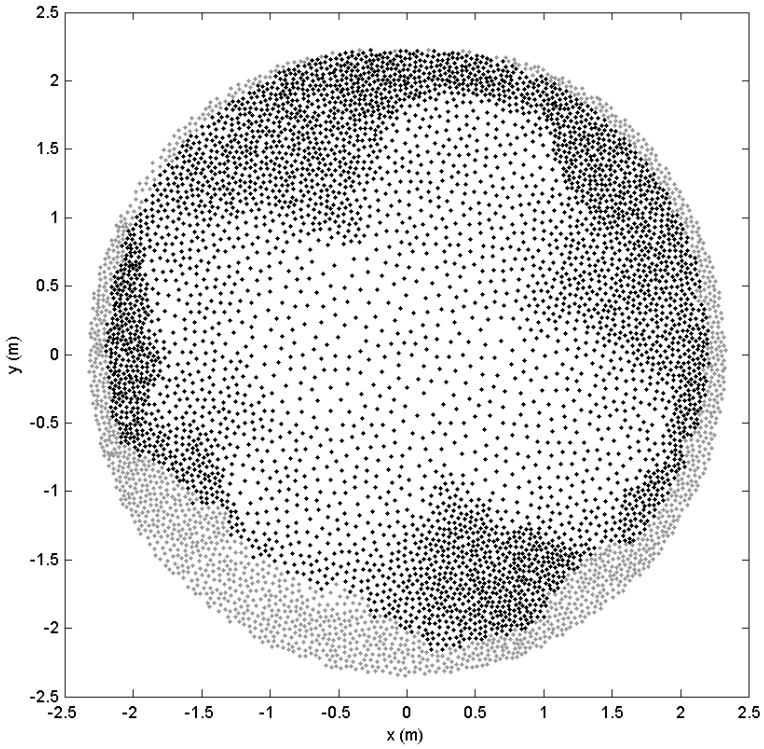
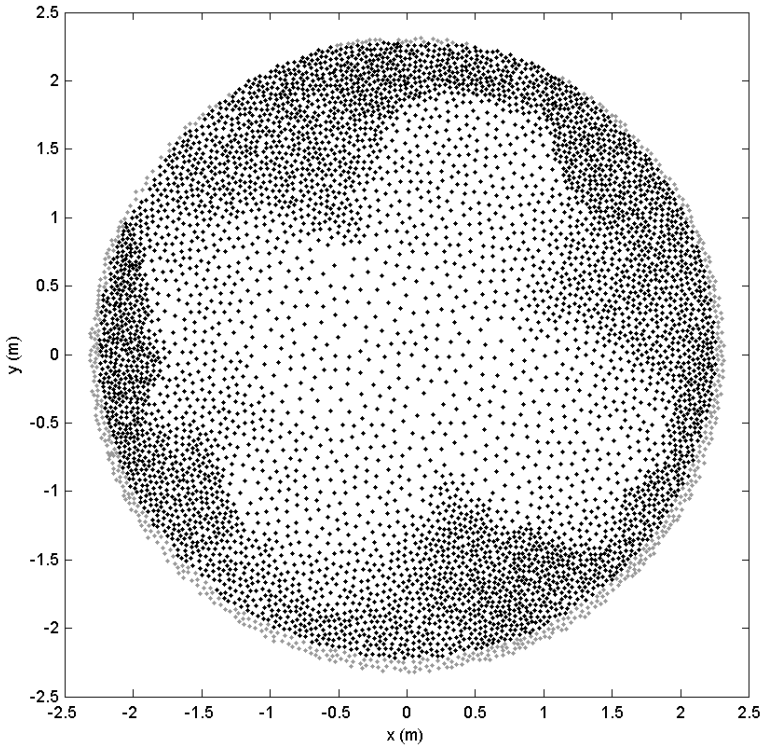


Figure 5.37: Iterations per time step for the solitary wave runup on a conical island, the average number of iterations is 1.95.



(a)



(b)

Figure 5.38: The horizontal runup for (a) The velocity is limited for $d_{u=0} = 10^{-4}$. Due to a thin layer of water the calculated horizontal runup shows large (artificial) peaks (b) The velocity is limited for $d_{u=0} = 10^{-3}$ the runup shows a more physical realistic image.

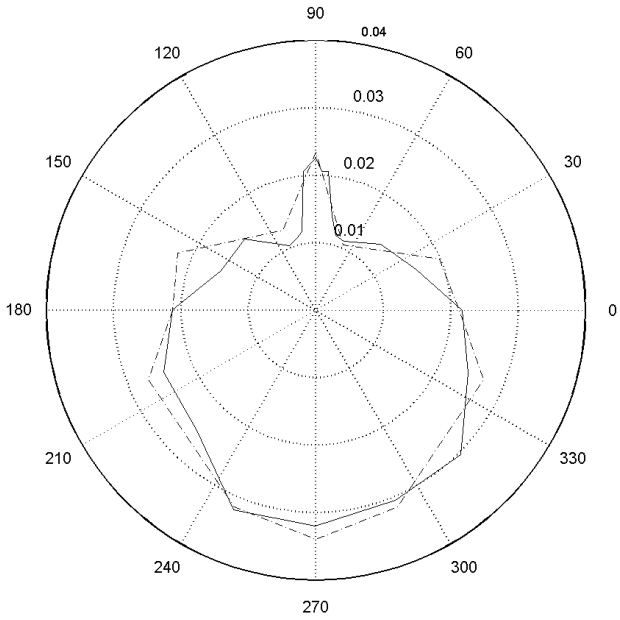


Figure 5.39: The vertical runup while lumping the mass matrix at dry and partially dry elements.

the horizontal runup is plotted together with the measured runup some resemblance can be observed. The shape of the shoreline and the enhanced runup at the lee side of the island is represented well, see figure 5.39. However, this result is highly dependent of $d_{u=0}$. In figure 5.39 $d_{u=0} = 5 \cdot 10^{-3}$, for smaller values the runup is larger and for larger values the runup is smaller than the measured runup.

5.2.4 Two-dimensional dam break

As a last test to the wetting and drying procedure presented in this study a two-dimensional dam break is simulated. Dam break problems have been the subject of many research studies. Brufau and Garcia-Navarro [4] developed a model based on multi dimensional upwind schemes that is suited to model a dam break into a bent channel. Stelling and Duijnmeijer [22] also use an upwind scheme to simulate a two-dimensional dam break which they compare to measurements with good results. The same measurements will be used here to compare the simulation to.

Experimental setup

This test is performed with a large basin of which a relatively small compartment contains water. At $t = 0\ s$ a gate is lifted with 16 cm/s and the dry area is flooded. The water containing part has a width of approximately 8 m by 2.4 m and has 0.6 m water in it. The dry basin is 8 m wide and almost 30 m long. The opening between the two basins has a width of 0.40 m. The experimental setup is shown in figure 5.40. The propagation and shape of the water front was measured with a video camera. The thickness of the water layer is measured with wave gauges which were positioned at the symmetry axis of the basin.

Numerical setup

To perform this test with minimum calculation capacity a symmetry axis is chosen in the middle of the basin in the long direction and only half of the basin is modeled. The numerical layout is shown in figure 5.41. The approximate area of the grid cells in the outflow basin is 0.04 m^2 and the approximate area of the elements in the water basin is 0.1 m^2. In the outflow opening mesh refinement is applied to an approximate area of 0.01 m^2. The mesh consists in total of 18268 elements. The time step is set to $\Delta t = 0.0004\ s$ and the computed time is 6 s, $d_{min} = 10^{-4}\ m$, $d_{u=0} = 10^{-4}\ m$ and the mass matrix is lumped for completely dry elements. The nodes with $y_i \leq 0$ have an initial water depth of 0.6 m. The slow opening of the gate is not taken into account and the water is released at once at $t = 0\ s$. Mannings friction parameter is set to $n = 0.01\ m^{-\frac{1}{3}}s$.

Results and discussion

In figure 5.42 and 5.43 the numerical solution to the dam break problem at times $t = 1, 2, 3, 4, 5 \text{ and } 6\ s$ is shown. The model is able to model such a severe test case without any numerical artificialities. Just as in the one-dimensional dam break two types of waves can be observed. One wave traveling into the water basin and one wave traveling into the outflow basin. The water accelerates mainly in the y-direction from the pressure out of the basin. When flowing further out of the basin it slowly starts to spread in lateral direction too.

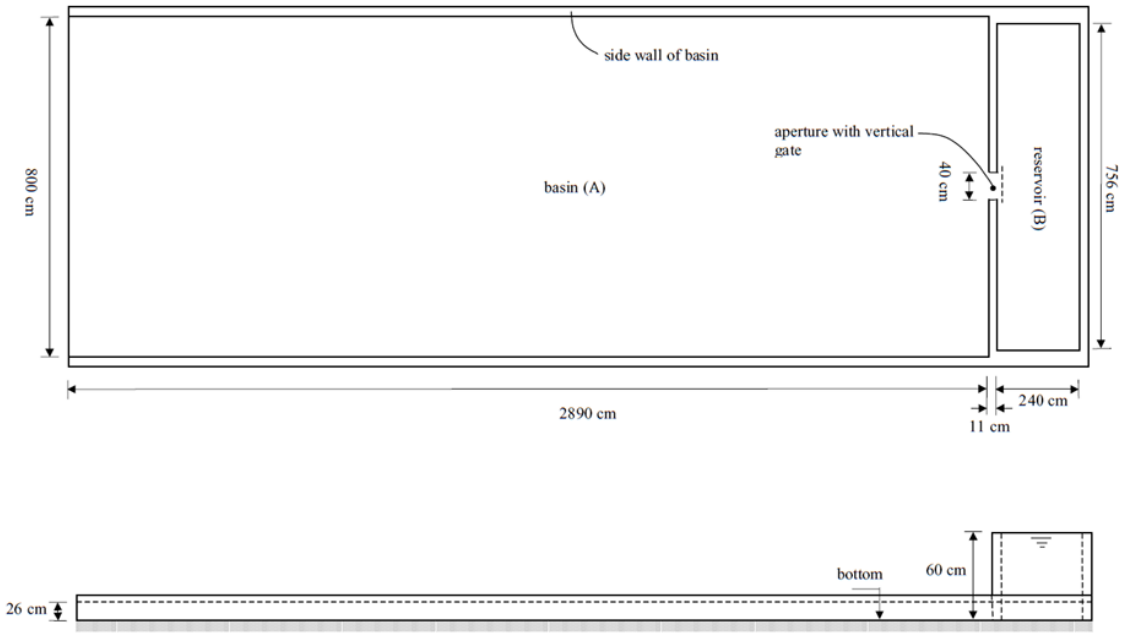


Figure 5.40: Experimental setup of the two-dimensional dam break experiment.

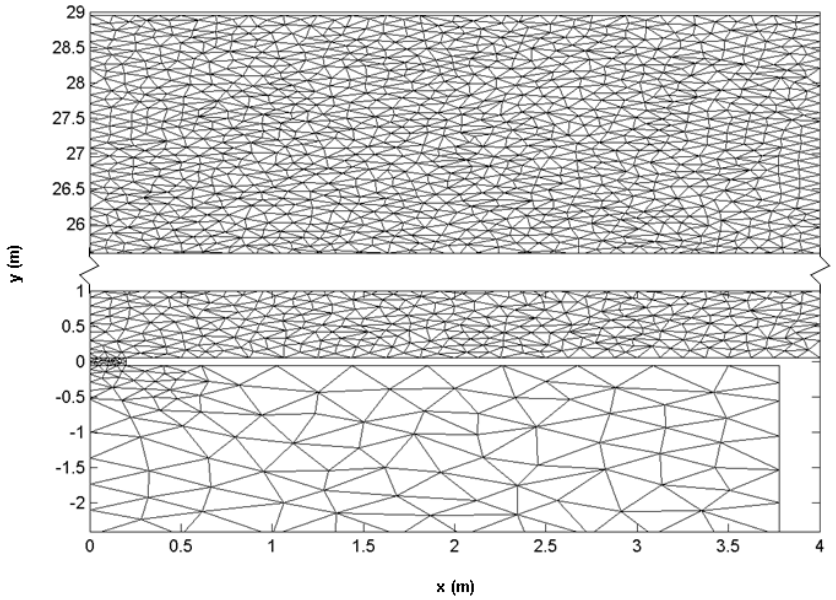


Figure 5.41: Computational meshes for the dam break test case.

In figure 5.44 the numerical results for the location of the front is shown at $t = 1, 2, 3$ and 4 s compared to the measured values and the numerical results of Stelling and Zijlema [22]. It is obvious that the propagation of the front and the measured values in y-direction is very well initially however after a few seconds a slight lag becomes visible. This lag can also be observed in 5.45 for the gauges at a distance of 6 and 9 m from the aperture. This is not a surprising observation since we have seen this phenomenon already in several one-dimensional test cases.

The lateral movement of the front is not as satisfying as the longitudinal movement. The spreading of the front in lateral direction is too large. The cause lays within the assumption of the instantaneous opening of the gate instead of an opening speed of 16 cm/s and in the assumption of hydrostatic flow in the outflow opening.

The average number of iterations per time step is 1.32, the iterations per time step are shown in figure 5.46. The calculation time is relatively high compared to the previous tests due to poor convergence of BiCGSTAB. For other tests BiCGSTAB needed around 40 iterations and for this test it could even be up to 400.

The mass error is again of $\mathcal{O}(10^{-6})\%$ of the total amount of water, as is visible in figure 5.47.

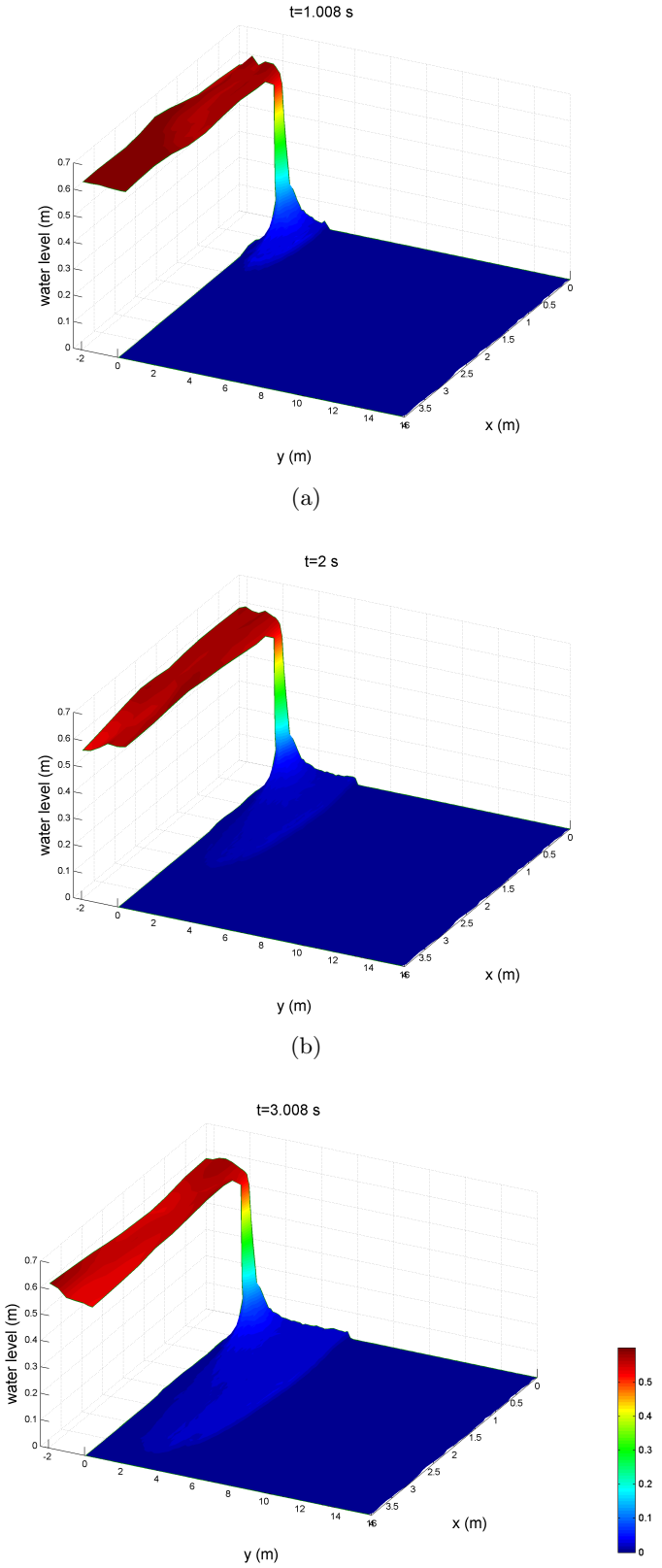


Figure 5.42: The numerical solution for the two-dimensional dam break problems at (a) $t = 1$ s, (b) $t = 2$ s and (c) $t = 3$ s.

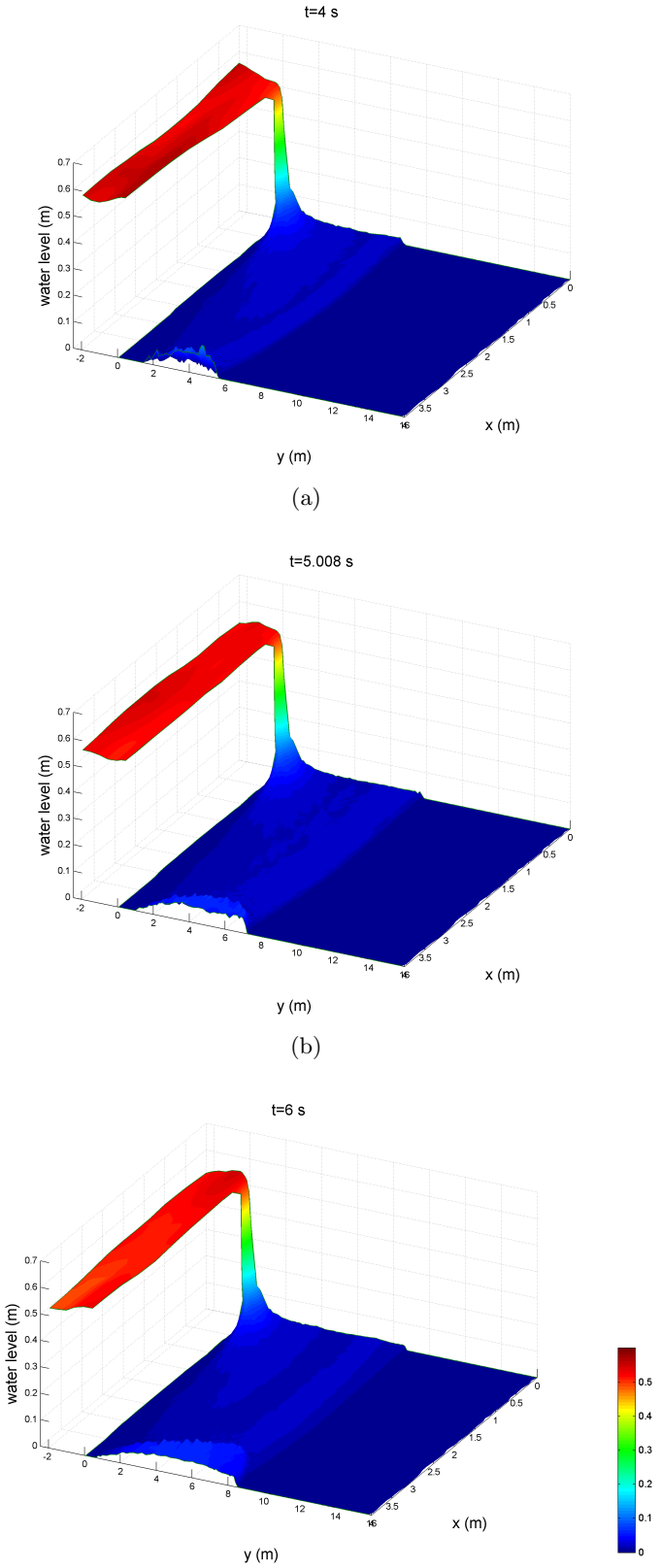


Figure 5.43: The numerical solution for the two-dimensional dam break problems at (a) $t = 4$ s, (b) $t = 5$ s, and (c) $t = 6$ s.

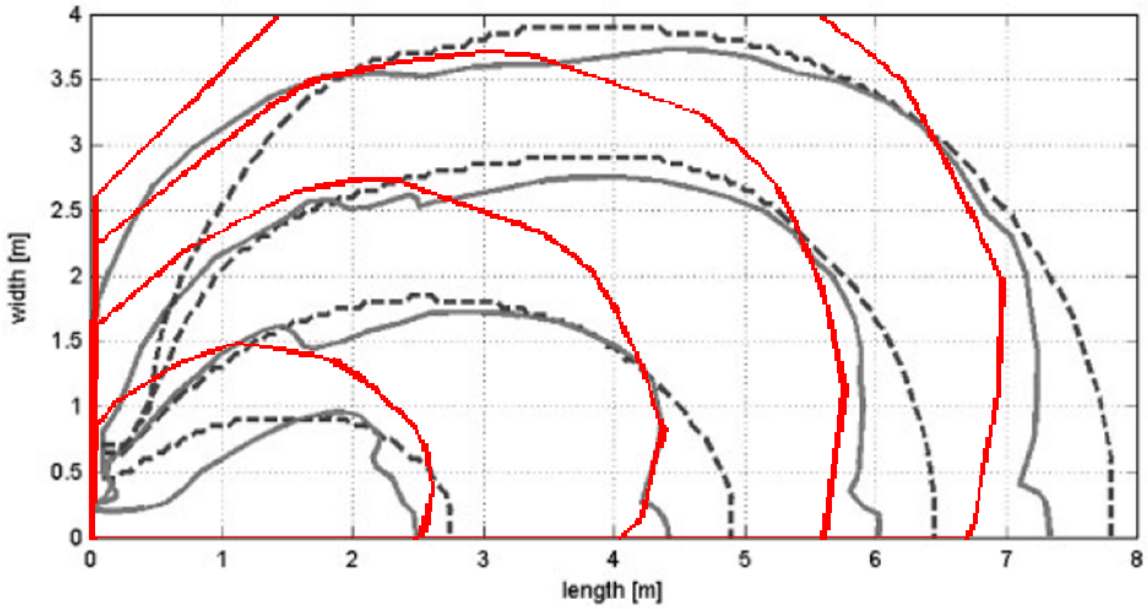


Figure 5.44: The length and width of the front for the two-dimensional dam break test at $t = 1, 2, 3, 4$ s the gray line indicates the measured values, the dashed line indicates the numerical results of Stelling and Duinmeijer and the red line indicates the results obtained with the model presented here. In the numerical results $n = 0.01\ m^{-\frac{1}{3}}s$ is used.

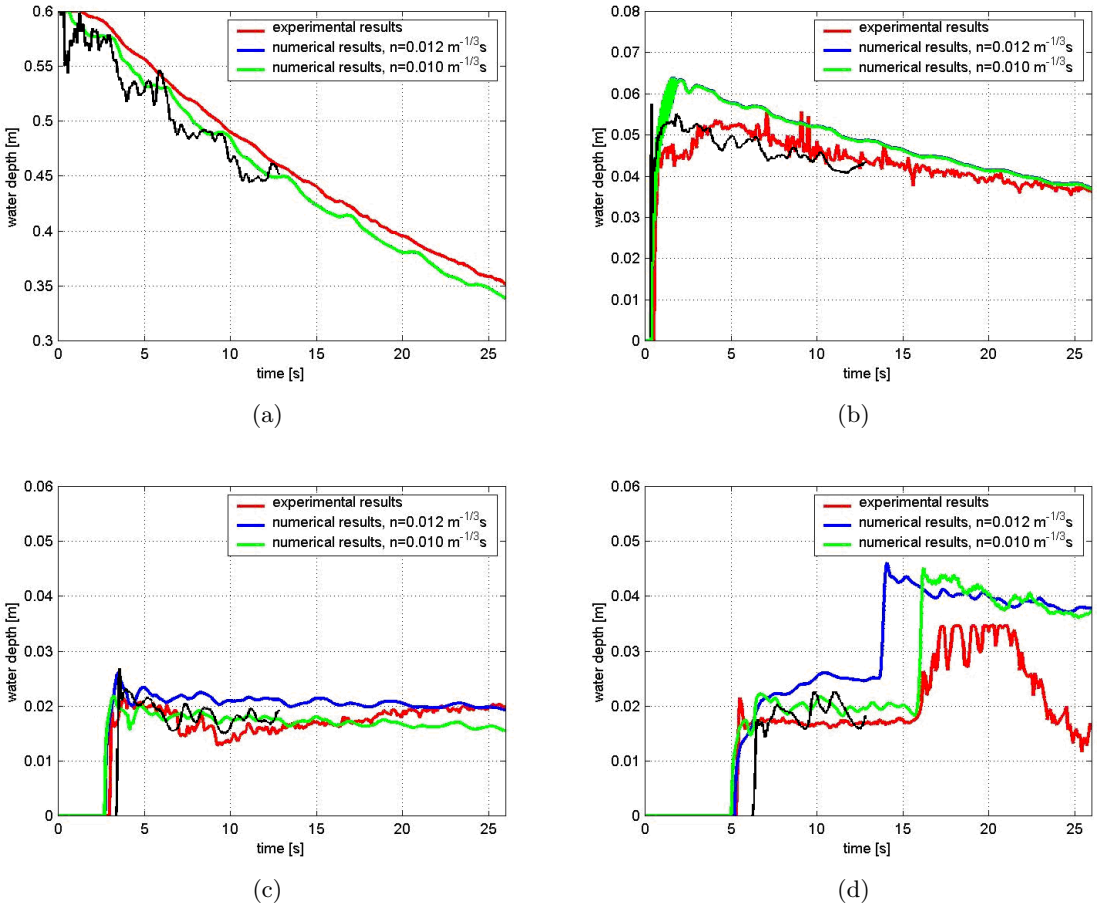


Figure 5.45: Water levels at (a) gauge 0, at $y = -1\ m$, (b) gauge 1 at $y = 1\ m$, (c) gauge 2 at $y = 6\ m$ and (d) gauge 3 at $y = 9\ m$. The red line represents the experimental results, the green and the blue line represent the numerical results of Stelling and Duinmeijer with respectively $n = 0.01\ m^{-\frac{1}{3}}/s$ and $n = 0.012\ m^{-\frac{1}{3}}/s$ and the black line are the numerical results obtained with the model presented here and $n = 0.01\ m^{-\frac{1}{3}}/s$.

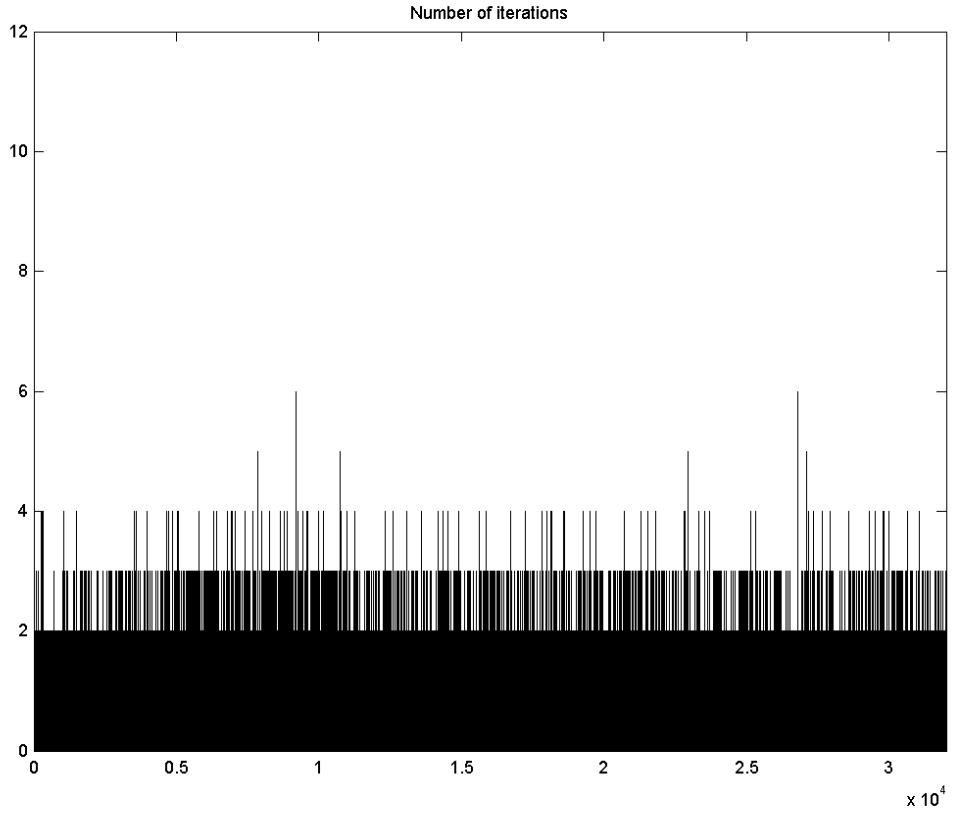


Figure 5.46: Iterations per time step for the two-dimensional dam break problem. The average number of iterations per time step is 1.32.

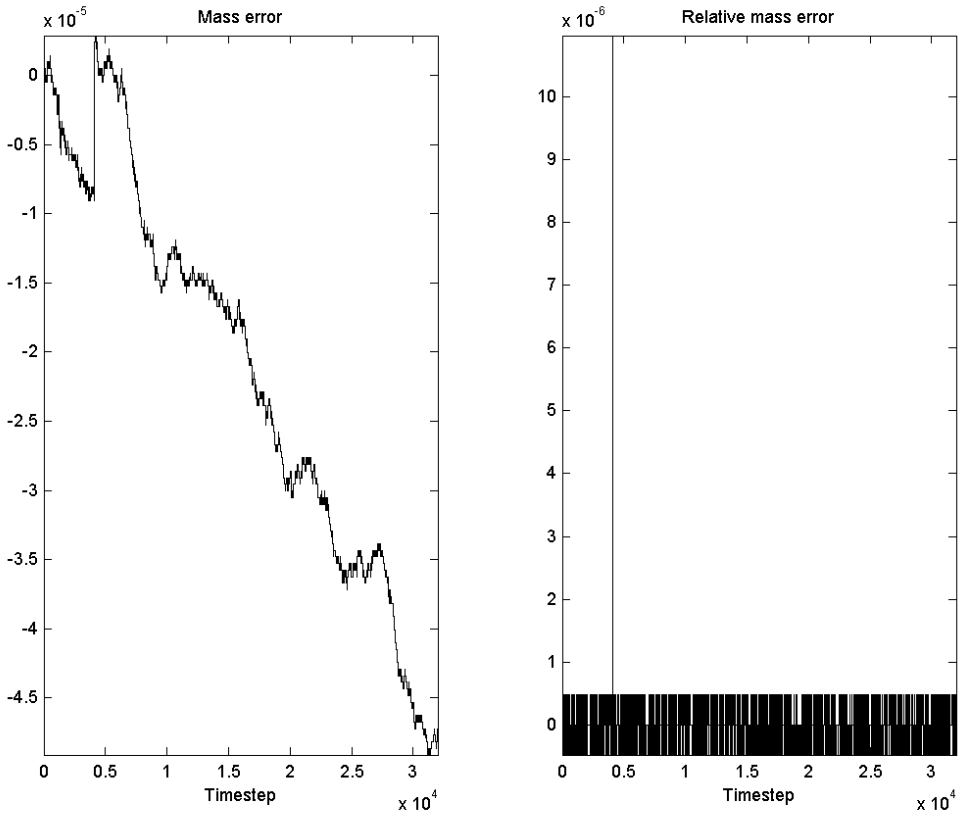


Figure 5.47: Mass conservation for the two-dimensional dam break problem. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

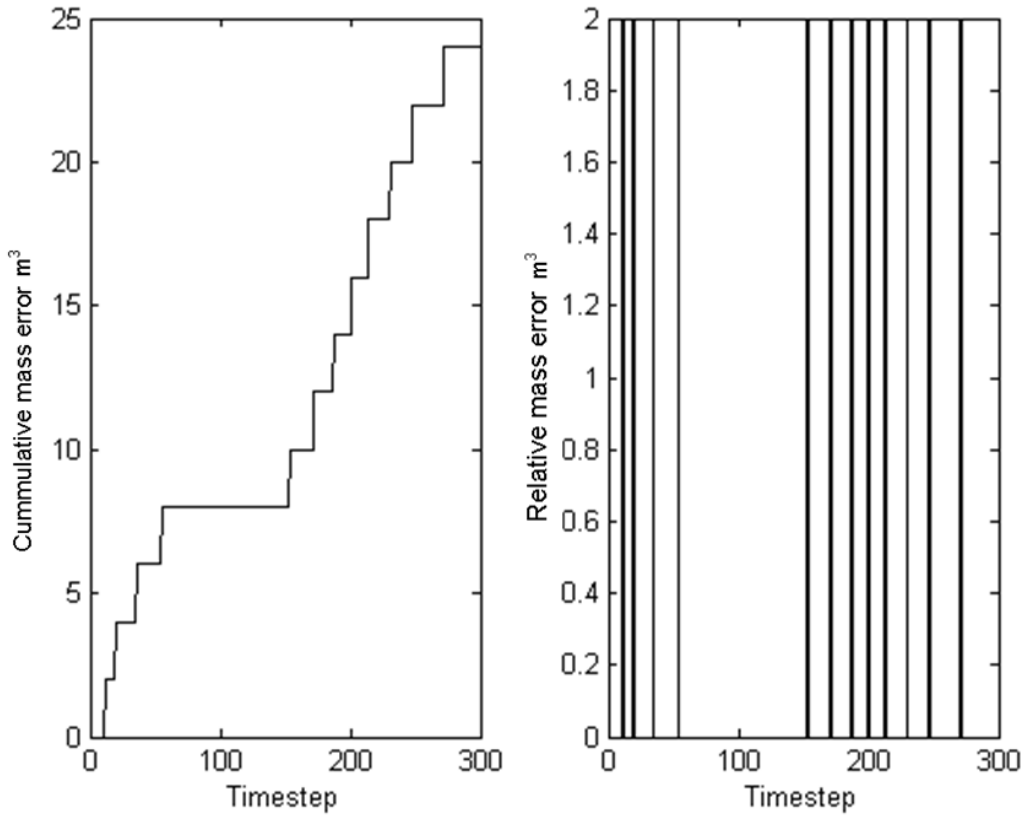


Figure 5.48: Mass error for a still water test in a two-dimensional parabolic basin. The total amount of water in the basin is $1.96 \cdot 10^7\ m^3$.

5.3 Mass conservation

When considering the examples in sections 5.1 and 5.2 mass conservation of the wetting and drying procedure appears to be good in general, of $\mathcal{O}(10^{-6}\%$. Mass errors for completely wet tests are of the same order as for tests that include wetting and drying. In fact, in case of a still water test the occurring mass errors are of the same order as in case of wetting and drying.

Nevertheless after a few cases it becomes apparent that a rather strange phenomenon with respect to mass conservation can be observed. In every test, if an error occurs between two subsequent time steps, it is of exactly the same size. In one dimension this was not considered as troublesome since the error was of $\mathcal{O}(10^{-14})\ m^2$ however in two dimensions this error becomes considerably larger. For large domains with a total mass of $\mathcal{O}(10^7)\ m^3$ this error can even grow to $\mathcal{O}(1)\ m^3$.

To illustrate this phenomenon a still water test is used. A two dimensional parabolic basin with a horizontal water level is assumed without external forcing. At $t = 0$ s the simulation is started. Since no external forcing is applied all fluctuations will be numerical artificialities and rounding errors. Moreover since the wet area is constant in time the system reduces to a linear system. The total amount of water in the system is $1.96 \cdot 10^7\ m^3$.

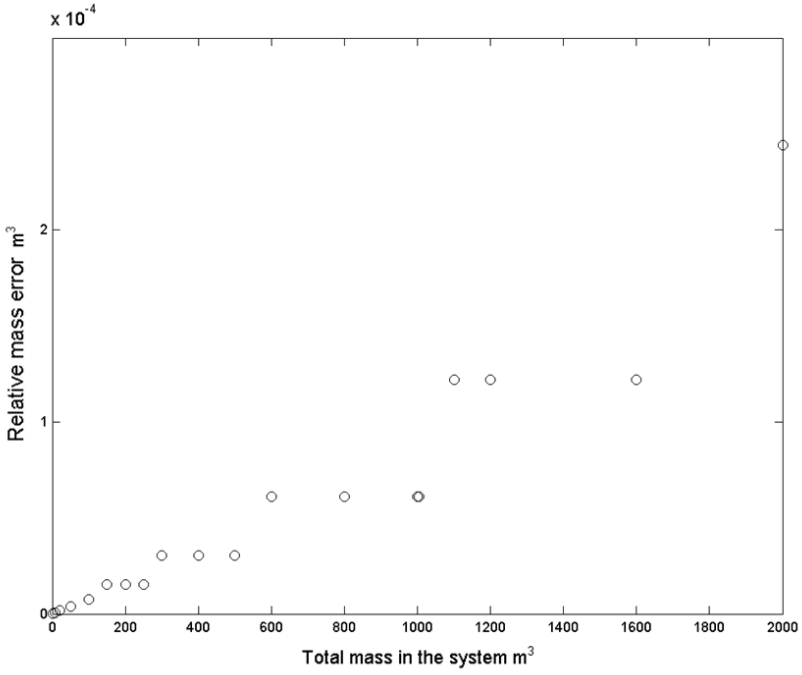


Figure 5.49: Absolute value of the relative mass error plotted against the initial volume of water in the system for a still water test in a cubic basin with different water depths.

In figure 5.48 the mass error is shown. In the left plot the cumulative mass error is given and in the right plot the mass error between two subsequent time steps can be seen. The mass error in this test does not differ significantly from a mass error plot with water movement or shoreline movement as we have seen in section 5.2.

This mass error is not influenced in size and only slightly in occurrence by using a d_{min} or not, or different values for d_{min}. It does not matter whether the mass matrix is lumped completely, at dry elements or at dry elements and partially dry elements. Moreover there is no connection between the occurrence of bifurcation and the occurrence of the mass error. Furthermore the mass error is indifferent to the mesh used. It does not matter how many elements the mesh consists of or whether the mesh is structured or unstructured. The only relation that is to be found is with the amount of mass in the system.

To visualize this a new still water test is performed. The tests is performed on a cubic basin with four closed walls. The water level in the basin is horizontal and no external forcing is applied. The test is performed several times with different water volumes. The volume in the basin is adjusted by adjusting the water depth. The total area of the basin is 100 m^2.

In figure 5.49 the absolute value of the mass error between two subsequent time steps is plotted against the total amount of mass in the system. The occurring mass error per time step is constant until a certain threshold mass is reached. Then the relative mass error becomes twice its size for a certain range in mass again until it exceeds another threshold and the error is doubled once more. In table 5.2 the values that are plotted are given with the enlargement factor with respect to the previous value. As can be seen the enlargement factors are always multiples of 2. However in terms of percentage the mass errors might not be so disturbing after all, since these are always of $\mathcal{O}(10^{-5})$. Even in the case shown in figure

Mass	Absolute value of the relative mass error	Enlargement factor with respect to previous error	Percentage of mass
2	1.1921e-07	-	5.9605e-06
7.8375	4.7684e-07	4	6.0840e-06
20	1.9073e-06	4	9.5367e-06
50	3.8147e-06	2	7.6294e-06
100	7.6294e-06	2	7.6294e-06
150	1.5259e-05	2	1.0173e-05
200	1.5259e-05		7.6294e-06
250	1.5259e-05		6.1035e-06
300	3.0518e-05	2	1.0173e-05
400	3.0518e-05		7.6294e-06
500	3.0518e-05		6.1035e-06
600	6.1035e-05	2	1.0173e-05
800	6.1035e-05		7.6294e-06
1000	6.1035e-05		6.1035e-06
1004	6.1035e-05		6.0792e-06
1100	1.2207e-4	2	1.1097e-05
1200	1.2207e-4		1.0173e-05
1600	1.2207e-4		7.6294e-06
2000	2.4414e-4	2	1.2207e-05

Table 5.2: Mass in the system and relative mass error for a still water test in a cubic basin with different water depths.

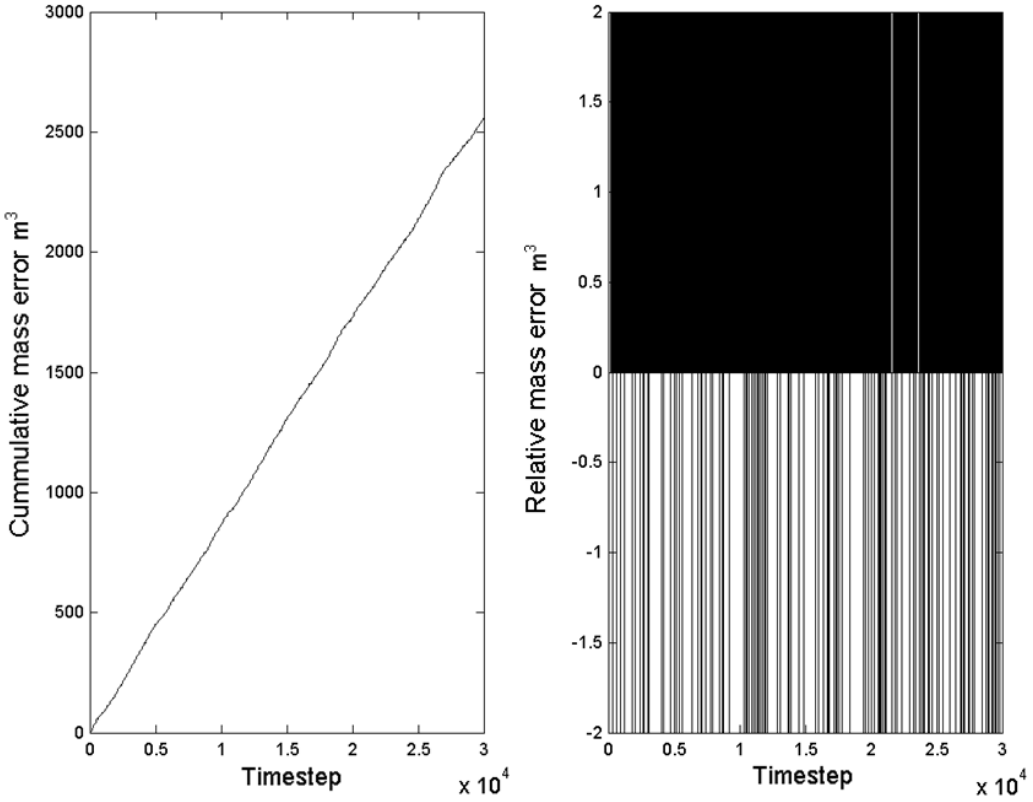


Figure 5.50: Long term mass loss for a still water test in a two-dimensional parabolic basin after 15000 s.

5.48 where the total amount of mass is $1.96 \cdot 10^7\ m^3$, the loss in terms of percentage is still of $\mathcal{O}(10^{-5})$. Nevertheless a distinct trend is visible in the mass errors and for long computations problems will most certainly arise. In figure 5.50 the mass loss for the still water test in a parabolic basin is shown with a calculated period of 15000 s in 30000 time steps. It can be seen that the loss per time step is equal in absolute value. For the cumulative mass error, a pronounced trend can be observed which does not differ from the short term trend. Indeed it can be concluded that this mass loss can cause some serious problems.

The fact that the occurrence is indifferent to so many parameters and even to water movement does point in the direction of an error at system accuracy that is enlarged by the scheme. Moreover the fact that the occurrence is indifferent to water movement indicates that the problem is not caused by the wetting and drying procedure.

To support this assertion a two tests will be done with double precision. The first test will be the same as discussed above. A two dimensional parabolic basin with a horizontal water level is assumed without external forcing. At $t = 0$ s the simulation is started. The second test is performed to show that double precision also diminishes the mass error in case of wetting and drying and is the same test only now a parabolic shaped initial water level is released.

In figure 5.51 the mass error for the still water test is presented which is now of $\mathcal{O}(10^{-9})$

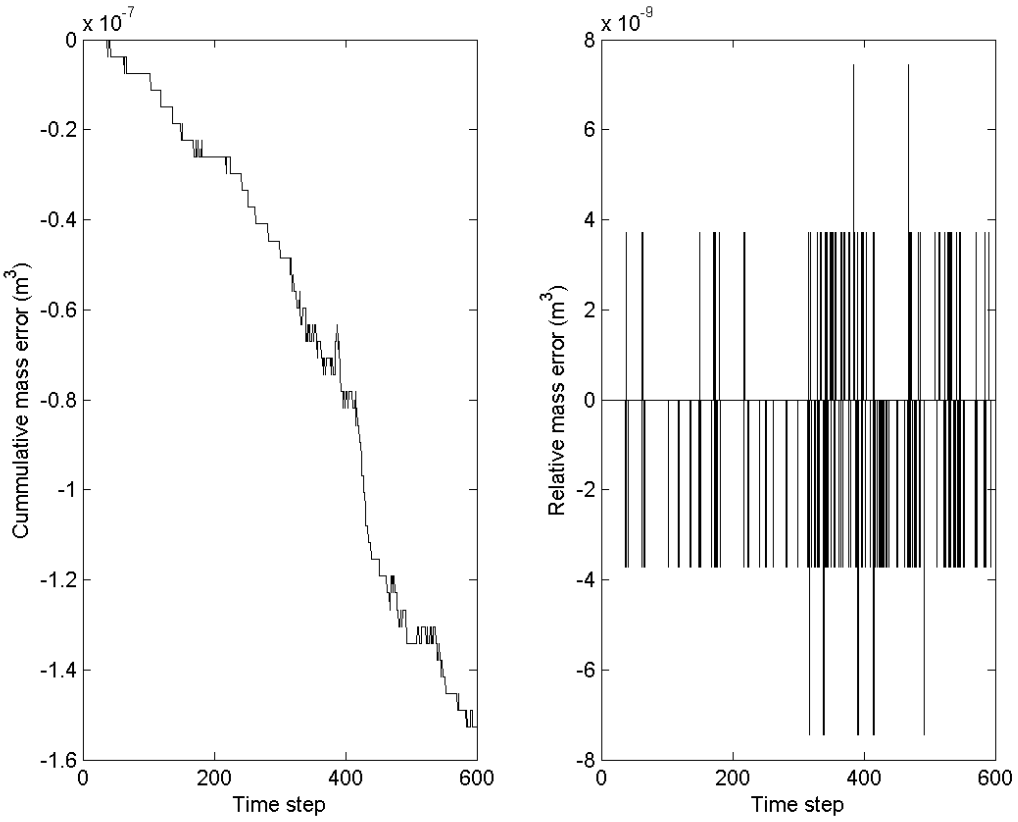


Figure 5.51: Still water test in a parabolic basin performed with double precision. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

per time step what means that it is reduced to $\mathcal{O}(10^{-14})\,\%$ which is system accuracy. In figure 5.52 the mass error for the parabolic basin with the parabolic shaped initial water level is shown. Now mass conservation is a little less however it is still of $\mathcal{O}(10^{-12})\,\%$ which is still not more than system accuracy. Moreover the cumulative mass error is in both cases of the same order. It is obvious that the usage of double precision resolves the mass error problem.

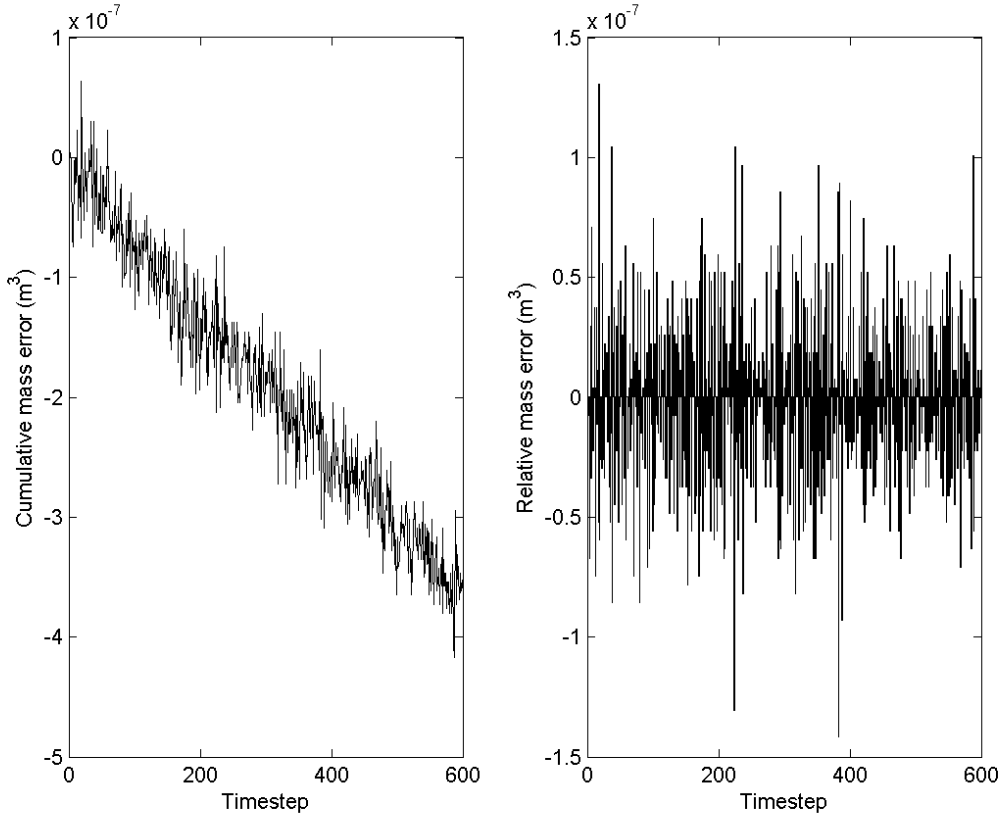


Figure 5.52: Test of a standing wave in a parabolic basin performed with double precision. Left: the total mass error as a function of the time step. Right: the mass difference with respect to the previous time step.

Chapter 6

Conclusions and recommendations

The objective of this study was to develop a wetting and drying algorithm for a 2DH finite element method based upon the wetting and drying algorithm proposed by Casulli [8] and investigate whether this procedure is a mass conservative, efficient and robust method to include wetting and drying. In this chapter conclusions with respect to its performance will be discussed and recommendations with respect to further research will be given.

6.1 Conclusions

The wetting and drying procedure is mass conservative. Occurring mass errors were caused by rounding errors and not necessarily by the wetting and drying procedure. However mistreated bifurcations can lead to mass errors which are a consequence of the wetting and drying procedure. The occurrence of bifurcations is a serious problem in practical applications. Their occurrence can be minimized, but not resolved, by using double precision and use a small tolerance and a large maximum number of iterations as BiCGSTAB settings.

The procedure costs in general 2 to 3 iterations in case of wetting and drying, except for some peaks in case of seriously large gradients at the wetting and drying interface or large flooding or drying velocities. The model is very robust in one dimension and slightly less in two dimensions. However using double precision has a considerable positive influence on robustness and on the iteration behavior of the wetting and drying iteration.

The main problems encountered with the model were numerical damping, mass conservation errors and wrong velocity estimations near large discontinuities in velocity. The first two problems are not properties of the wetting and drying procedure or of the momentum conservation over the contact discontinuity, but of discretization of the SWE. This implicates that the wetting and drying procedure itself does not suffer from these problems. The problem in the estimation of the velocity near the contact discontinuity is, of course, a problem of the wetting and drying algorithm. However this can most probably be solved by using an improved formulation for the advective terms.

6.1.1 Conclusions with respect to one-dimensional modeling

The wetting and drying algorithm presented by Casulli for finite volumes transformed to finite elements gives quite reasonable results. Compared to the finite volume case there are several difficulties that arise. The main problem is that the matrix properties are less favorable in case

of finite elements. To prevent the matrix A from becoming singular in a mass conservative way several measures can be taken:

- use a minimum cell averaged depth;
- manipulation of the matrix in combination with a lumped mass matrix at dry elements;
- removal of the dry nodes from the system in combination with a lumped mass matrix at dry elements;
- manipulation of the matrix in combination with updating the linear (T) and the non-linear part (P) of the system matrix every iteration step;
- removal of the dry nodes from the system in combination with reconstruction of matrix T every iteration step.

To be mass conservative there are three options:

- make use of a minimum water depth d_{min} in T to guarantee non-zero diagonal entries in the system matrix;
- use the lumped mass matrix at dry elements;
- reconstruct T every iteration step.

The use of the lumped mass matrix costs slightly less iterations than using the consistent mass matrix and the method seems to behave much more robust. BiCGSTAB can be used to speed up the calculations. However the solution behaves not as robust as with a direct solver; it is possible for bifurcations to occur and mass conservation is compromised a little.

The most general applicable and robust method is using d_{min}. It performs best on sloping topographies and it improves the robustness of BiCGSTAB significantly.

6.1.2 Conclusions with respect to two-dimensional modeling

In two dimensions all the above mentioned notions for prohibiting matrix singularities and for mass conservation hold. However for a robust model some more care should be taken. The use of d_{min} is necessary to obtain a robust model.

Lumping the mass matrix is also something that should be done for robustness, however not necessarily in the entire domain. The criterion for lumping depends on the type of problem. In case of large gradients lumping should be done for entirely dry elements, in case of gentle gradients the mass matrix should also be lumped for partly dry elements. Using a lumped mass matrix in the entire domain can result in some improvements with respect to computational costs, less wetting and drying iterations and better mass conservation (in some cases), however the model is not able anymore to produce the exact solution.

Including friction in the calculations diminishes the sensitivity to lumping of the mass matrix for partially dry elements. Still the convergence of the wetting and drying iteration and of BiCGSTAB is very sensitive to this however no instabilities are generated.

6.2 Recommendations

The wetting and drying algorithm performance is reasonably well on the testcases presented here. However only analytical solutions and measurement data from laboratory experiments are used. To obtain a complete picture of the performance of the wetting and drying algorithm it is recommended to do a simulation from civil engineering practice to assess its performance in non-ideal situations.

If computation time is a very important factor in the calculation this method is probably not the best option, since it introduces a nonlinearity which has to be solved iteratively. This means that the solution procedure to solve the system has to be done twice as often (on average). Since this part is often the most expensive part of the calculation it will increase the computational effort considerably. On the other hand many other methods impose time step limitations on the solution procedure increasing the computational effort in another way. Therefore this method is absolutely not inferior to other methods available.

Many problems encountered upon in this study seem to be related to the finite element method. Off course it is not possible to draw conclusions with respect to finite volume modeling from this study but this method seems to be a more elegant fit for a finite volume method. Nevertheless it is possible to implement it in a finite element method but it will not behave robust in all circumstances.

For practical applications the occurrence of bifurcations is a serious problem since it increases the number of iteration steps and the amount of storage space required. It is advised to perform more research with respect to this phenomenon. For example test whether a continuous formulation of H would solve this problem. Moreover it is important to find out why sloping bottoms induce bifurcations or at least enhance the occurrence of bifurcations

The criterion stated here whether to lump the mass matrix only for dry elements or also for partially dry elements is only based on four tests. Founding such a conclusion on only four observations is rather simplistic inductive reasoning. It is therefore recommended to perform more research on why lumping the mass matrix for partially dry elements is in some cases necessary to obtain results and in some cases the cause that no results are obtained.

If one is determined to use the entire schematization presented in this study and not only the wetting and drying algorithm, it is advised to implement the advective term implicit and to use double precision in all the calculations to diminish the mass error. However taking the advective term implicit will cancel the biggest advantage of the scheme presented here, namely that only one system needs to be solved instead of two coupled systems.

Bibliography

[1] O. Bokhove. Flooding and drying in finite-element discretizations of shallow-water equations. part 1: One dimension. Memorandum No. 1683, 2003.

[2] M. J. Briggs, J. C. Borrero, and C. E. Synolakis. Tsunami disaster mitigation research in the united states. In *International Symposium on Tsunami Disaster Mitigation in Future*, 1995.

[3] C. G. Broyden. A class of methods for solving nonlinear simultaneous equations. *Mathematics of Computation*, 19(92):577–593, 1965.

[4] P. Brufau and P. Garcia-Navarro. Two-dimensional dam break flow simulation. *Int. J. Nummer. Meth. Fluids*, 2000.

[5] L. Brugnano and V. Casulli. Iterative solution of piecewise linear systems. *SIAM J. Sci. Comput.*, 30(1):463–472, 2007.

[6] S. Bunya, E. J. Kubatko, J. J. Westerink, C. Dawson, and S. Yoshimura. A mass conserving moving boundary method for discontinuous galerkin solutions to the shallow water equations. Technical report, ICES, Institute for Computational Engineering and Sciences, 2007.

[7] G.F. Carrier and H.P. Greenspan. Water waves of finite amplitude on a sloping beach. *J. of Fluid Mech.*, 4:97–109, 1958.

[8] V. Casulli. A high resolution wetting and drying algorithm for free-surface hydrodynamics. *Int. J. Numer. Meth. Fluids*, 60:391–408, 2008.

[9] L. Fraccarollo and E. F. Toro. Experimental and numerical assessment of the shallow water model for two-dimensional dam-break type problems. *J. of hydraulic research*, 33(6):843–863, 1995.

[10] D.R. Fuhrman and P.A. Madsen. Simulation of nonlinear wave run-up with a high-order boussinesq model. *Coastal Engineering*, 55:139–154, 2008.

[11] B. van 't Hof and E. A. H. Vollebrecht. Modelling of wetting and drying of shallow water using artificial porosity. *Int. J. Numer. Meth. Fluids*, 48:1199–1217, 2005.

[12] J. van Kan, A. Segal, and F. Vermolen. *Numerical Methods in Scientific Computing*. VSSD, 2005.

[13] B. Koren. A robust upwind discretisation method for advection, diffusion and source terms. In *In C.B. Vreugdenhil and B. Koren, editors, Numerical Methods for Advection-Diffusion Problems*, pages 117–137. Vieweg, Braunschweig, 1993.

[14] R. J. Labeur. *Finite element modelling of transport and non-hydrostatic flow in environmental fluid mechanics*. PhD thesis, Delft University of Technology, 2009.

[15] R. J. LeVeque. *Numerical Methods for Conservation Laws*. Birkhauser, Basel, 1992.

[16] P. L.-F. Liu, Y.-S. Cho, M. J. Briggs, U. K. LU, and C. E. Synolakis. Runup of solitary waves on a circular island. *J. Fluid Mech.*, 302:259–285, 1995.

[17] R. A. Luettich and J. J. Westerink. An assessment of flooding and drying techniques for use in the adcirc hydrodynamic model. Technical report, CERC, 1995.

[18] R. A. Luettich and J. J. Westerink. Implementation and testing of elemental flooding and drying in the adcirc hydrodynamic model. Technical report, CERC, 1995.

[19] P.A. Madsen, O. R. Sorensen, and H.A. Schaffer. Surf zone dynamics simulated by a boussinesq type model. part i. model description and cross-shore motion of regular waves. *Coastal Engineering*, 32:255–287, 1997.

[20] J. van der Molen. *Tides in a Salt-Marsh (Great Marshes, Barnstable, Cape Cod, USA)*. PhD thesis, Vrije Universiteit, 1997.

[21] O. Pironneau. *Finite Element Methods for Fluids*. John Wiley and Sons, 1989.

[22] G. S. Stelling and S. P. A. Duinmeijer. A staggered conservative scheme for every froude number in rapidly varied shallow water flows. *Int. J. Numer. Meth. Fluids*, 43:1329–1345, 2003.

[23] G. S. Stelling and M. Zijlema. Efficient computation of surf zone waves using the nonlinear shallow water equations with non-hydrostatic pressure. *Coastal Engineering*, 55:780–790, 2008.

[24] J. J. Stoker. *Water Waves*. Interscience, New York, 1957.

[25] W. C. Thacker. Some exact solutions to the nonlinear shallow-water wave equations. *J. of Fluid Mech.*, 107:499–508, 1981.

[26] E. F. Toro. *Shock-Capturing Methods for Free-Surface Shallow Flows*. Wiley, 2001.

[27] P. Wesseling. *Principles of computational fluid dynamics*. Springer, 2001.

List of Figures

List of Tables