Comflow and Amazon-sc: Chalk and Cheese

– A Comparison of Two Numerical Methods for Fluid Flow –

Fieke Geurts

Department of Mathematics

RuG

Master's Thesis

Comflow and Amazon-sc: Chalk and Cheese

– A Comparison of Two Numerical Methods for Fluid Flow –

Fieke Geurts

Supervisor:
Prof.dr. A.E.P. Veldman
Department of Mathematics
University of Groningen
P.O. Box 800
9700 AV Groningen

August 2006

Contents

Chapter 1

Introduction

"As different as chalk and cheese". It is the British way of expressing that two things are alike in appearance, but totally different in nature.[1] Metaphorically, this phrase summarises the contents of this master's thesis. In less obscure and more scientific terms, the thesis deals with the following.

1.1 Historical perspective

To place the contents in its context, this thesis is concerned with the mathematical field of Computational Fluid Dynamics (CFD), which deals with the development of methods for the simulation of fluid flow. The application of CFD stretches from the flow of air around airplanes, to water past ships, to blood through arteries, to air in the atmosphere and to the sea impacting on dykes or sea walls. See Figure 1.1 for an example of the last.

Characteristic of all cases is that a practical problem, relating to some sort of flow, is represented by a *physical model*. The physics, and especially the simplifications of the physics, is largely determined by the type of situations the method aims to simulate (liquid or gas, the scale of the problem, etc.) Together with the simplifications, the physical model determines the *mathematical model*, which is given in the form of a set of partial differential equations. Typical as well is that these equations can not be solved analytically – by pen and paper – without very strong simplifications. More complex situations require a reformulation of the equations, by a *numerical method*, such that the brute calculation force of computers can be of help. These three ingredients (physical, mathematical and numerical model) constitute the three modelling pillars of CFD. In the flowchart of Figure 1.2 these steps are depicted graphically [26, p. V].

[1] The phrase is said to originate from the similarity between the superficial structure of chalk and cheese. Traditional English cheese is white, dry, crumbly and hard – it has the appearance of a block of raw chalk. Nevertheless, this impression of similarity soon makes way, given the profound dissimilarities between the two. (www.phrases.org.uk)

Figure 1.1: *Violently overtopping wave at a sea wall. Picture by G. Motyker, HR Wallingford.*

The mathematical model that gives a general description of continuous fluid flow has been known for over hundred-and-fifty years in the form of the Navier-Stokes equations. This set of partial differential equations is based on the mechanical conservation laws for mass, momentum and energy. They were first derived by the French engineer Claude-Louis Navier (1785 – 1836) in 1822, based on a molecular model. Independently, in 1845, the same equations were derived by the Irish mathematician Sir George Gabriel Stokes (1819 – 1903). Though this time by a more general approach based on the theory of continuum [1].

The first reported attempt to solve these equations numerically, was made in 1922 by the British mathematician Lewis Fry Richardson (1881–1953). Wanting to predict the weather, he divided physical space into grid cells, approximating the solution in each cell. The enormous amount of calculations involved, prevented him to achieve his aim. His attempt to calculate a weather forecast of eight hours took six weeks and, apart from being overtaken by the weather itself, produced wrong results. He predicted that over sixty thousand people would be needed to outrun the real time in a weather prediction – he envisaged that clerks in a theatre would produce numerical results under the direction of a conductor [29].

The brute force of computers is exactly the thing Richardson would have needed. In the fifties of the twentieth century the development of computers gave CFD a new impulse. During the last decades, mathematical theory on numerical methods has been developed rapidly, to use the newly available computational force to the fullest. Nowadays a large amount of methods is available for the simulation of flow related problems. Many situations can be accurately simulated due to the development of sophisticated numerical methods in addition to the improvement of computational speed. CFD is now used to test and develop cars, airplanes, ships, medical devises, dykes, dams and so on. However, the 'elusive' nature of fluid flow still makes it difficult to accurately simulate complex situations.

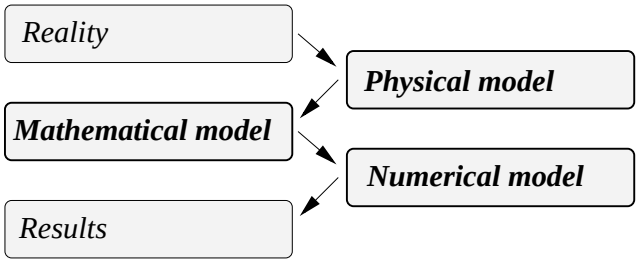

Figure 1.2: *Flowchart for modelling in Computational Fluid Dynamics.*

1.2 Problem definition

Within the field of CFD, this thesis focuses on two numerical methods for the simulation of flow problems – in particular flow problems with a free surface and moving objects. The two methods are the solvers COMFLOW and AMAZON-SC respectively.

In the Netherlands, the simulation tool COMFLOW has been developed in close cooperation between the university of Groningen (RuG), the Dutch maritime research institute (MARIN) and the Dutch national aerospace laboratory (NLR). Originally, the method is developed to simulate liquid sloshing on board spacecraft. Later, the method is extended to incorporate the impact of waves on (moving) ships. The combination of these two fields gives a method that is able to simulate flow problems with both moving solid bodies and free surfaces [7, 14, 15].

In the United Kingdom, the simulation tool AMAZON-SC has been developed at Manchester Metropolitan University (MMU). The code is part of the project to model violent overtopping by waves at sea walls. The method is also applied to simulate flow around wave energy devices and water entry phenomena with wedge shaped objects [5, 19, 21, 22].

The situations that these tools aim to simulate show a great similarity: both focus on moving objects and impacting waves. In view of these common interests, an exchange of knowledge and experience could facilitate the improvement of either method – thus motivating an MSc project to compares the two tools. The project is carried out at both universities: five months are spent in Groningen and five in Manchester. The aim of the project is thus as follows:

> Two computational methods, COMFLOW and AMAZON-SC, are compared to pinpoint their differences. It is studied why the different choices have been made and what the effect of these choices is on the numerical results. The possibility to enhance either solver by methods used in the other is investigated.

Figure 1.3: *Claude Navier (left), George Stokes (middle), Lewis Richardson (right).[1]*

The comparison of the two solvers shows that there are great differences in the theoretical basis. Nevertheless, in contrast to the differences in theoretical basis, the results of the simulations show (apart from some minor differences) a good similarity. What follows is thus a justification of the title: it shows that COMFLOW and AMAZON-SC are alike in appearance, but totally different in nature and thus why they are as different as chalk and cheese.

1.3 Outline of thesis

The thesis is organised as follows. Chapter 2 and Chapter 3 deal with the computational methods of COMFLOW and AMAZON-SC respectively. Both contain a description of the three modelling steps of CFD depicted in Figure 1.2, that is the physical, mathematical, and numerical model. Since the numerical basis of the two computational methods differs considerably, they are presented independent of each other. The two chapters do not refer to one another and can be read independently.

In Chapter 4 the results of simulations performed both with AMAZON-SC and COMFLOW are presented. The results are validated with experimental data if available. The test cases are wave generation in a tank, wedge entry phenomena and the flow around wave energy devices.

All information available is brought together in Chapter 5, where the two methods are again compared on the three levels of modelling. This theoretical comparison is used to further explain the differences in the numerical results.

The conclusions, with some recommendations for future research, are given in Chapter 6.

Chapter 2

Computational model: ComFlow

This chapter describes the computational model as implemented in the simulation tool Comflow. The physical, mathematical and numerical model are subsequently given in the following sections. The chapter concludes with a summary of the computational model.

2.1 Physical model

Comflow has been developed in close cooperation between the university of Groningen (RuG), the Dutch maritime research institute (MARIN) and the Dutch national aerospace laboratory (NLR). Originally, the method is developed to simulate liquid sloshing on board spacecraft. Later, the method is extended to incorporate impacting waves on (moving) ships. The combination of these two fields give a method that simulates flow problems with both moving solid bodies and free surfaces [7, 14, 15].

Given the complexity of free surface flow problems, some physical assumptions are necessary to make the resulting formulation trackable. An import assumption is the way the interface between the liquid (water) and gas (air) is treated. In Comflow the gas phase is neglected and only the liquid phase is considered. The interface between liquid and gas forms the (moving) boundary of the domain where the flow equations are solved. Additional assumptions are that the flow is incompressible and isothermal. Although most of the simulations performed with Comflow are convection-dominated and diffusive effects are of minor importance, the viscous effects are not neglected. The viscosity is considered constant.

The flow is simulated in three dimensions and the two dimensional case is considered as a special case of the three dimensional.

2.2 Mathematical model

A general description of the flow of a continuous fluid is given by mechanical conservation laws for mass, momentum and energy in addition to a thermodynamical equation of state. When the fluid is considered to be incompressible, isothermal and with constant viscosity, the mass and momentum equations suffice to describe the flow. These two equations are here referred to as the (incompressible) Navier-Stokes equations.[1]

2.2.1 Governing equations

In our mathematical model, the Navier-Stokes equations are presented in integral form for an arbitrary control volume Ω with boundary Γ. This presentation of the Navier-Stokes equations is natural since the equations are discretised with the finite volume method.

First, conservation of mass results in the equation

$$\oint_\Gamma \boldsymbol{u} \cdot \boldsymbol{n} \, \mathrm{d}\Gamma = 0, \tag{2.1}$$

which is commonly known as the continuity equation for incompressible flow. The symbols represent the velocity in three dimensions $\boldsymbol{u} = (u, v, w)^T$ and outward-pointing normal vector $\boldsymbol{n}$ of Γ.

Second, applying the conservation of momentum results in the system of equations

$$\frac{\mathrm{d}}{\mathrm{d}t} \int_\Omega \boldsymbol{u} \, \mathrm{d}\Omega + \oint_\Gamma (\boldsymbol{F}_{\text{inv}} - \boldsymbol{F}_{\text{vis}}) \cdot \boldsymbol{n} \, \mathrm{d}\Gamma = \int_\Omega \boldsymbol{g} \, \mathrm{d}\Omega. \tag{2.2}$$

The matrices $\boldsymbol{F}_{\text{inv}}$ and $\boldsymbol{F}_{\text{vis}}$ are the inviscid and viscous flux through the boundary of the control volume. The inviscid flux has a convective and a pressure term. The viscous flux has a diffusive term. They are given by

$$\boldsymbol{F}_{\text{inv}} \equiv \boldsymbol{u} \otimes \boldsymbol{u}^T + \frac{1}{\rho} p \boldsymbol{I} \quad \text{and} \tag{2.3}$$

$$\boldsymbol{F}_{\text{vis}} \equiv \frac{\mu}{\rho} \nabla \boldsymbol{u}. \tag{2.4}$$

The vector $\boldsymbol{g}$ denotes the acceleration due to body forces, which are only gravitational forces in the problems considered. The other symbols denote the pressure p, the density ρ, dynamical viscosity μ, the 3×3 identity matrix $\boldsymbol{I}$ and the vector tensor product $\otimes$.[2]

[1] The usage of the term 'Navier-Stokes equations' is ambiguous in literature. Here, they denote the (incompressible) equations for the conservation of mass and momentum.

[2] The equations reduce to the Euler equations when $\boldsymbol{F}_{\text{vis}} = \boldsymbol{0}$.

2.2.2 Free surface displacement

If there were no free surface to the fluid, the material part of the equations would now be given. To include a free surface, one more step is essential, since the position of the surface should be calculated. This implies that the free surface position must be tracked. This is achieved by describing the surface as $\boldsymbol{S}(\boldsymbol{x}, t) = 0$. In time, the motion is tracked with the material derivative

$$\frac{D\boldsymbol{S}}{Dt} = \frac{\partial \boldsymbol{S}}{\partial t} + \boldsymbol{u} \cdot \nabla \boldsymbol{S} = 0. \tag{2.5}$$

Interpreting this equation, the interface is moving with the liquid velocity.

2.2.3 Boundary conditions

In COMFLOW there are three types of boundaries. The first is a solid boundary, either stationary or moving. The second type consists of in- and outflow boundaries, where fluid flows in or out of the domain. The third is the free surface.

First, a no-slip boundary condition is applied at solid walls. This means that $\boldsymbol{u} = 0$ for fixed boundaries and $\boldsymbol{u} = \boldsymbol{u}_b$ for objects moving at velocity $\boldsymbol{u}_b$.

Second, incoming waves are generated by prescribing velocities at the inflow boundary. The term 'inflow boundary' is a little misleading, since the generation of waves by prescribing velocities implies that velocities can also become negative at this boundary. To determine the velocities at the inflow boundary, different kind of wave descriptions are used. The easiest is by means of , Airy, wave theory. Waves thus generated are only physically accurate in shallow water or for waves with a small height compared to their length. For steeper waves, nonlinear theory should be applied, for example in the form of 5th order Stokes waves. See [14] for more details.

At the outflow boundaries a condition is needed to prevent that waves re-enter the domain and distort the wave profile. This is achieved either by non-reflecting boundary conditions or by a dissipation zone where the wave is damped. The simplest form of the former is setting the normal derivatives of the velocity and pressure equal to zero at the outflow boundary. Since this simple procedure does not work very well, the Sommerfeld boundary condition is often applied. Here, the outflow wave velocity is determined as the wave velocity of a regular wave. It is therefore no surprise that the Sommerfeld condition only works for regular waves, that is for waves that are not too much disturbed by an object. To prevent the reflection of distorted waves, an alternative is a dissipation zone in the form of a numerical beach. The beach is usually one or two wavelengths long and the slope is formed by a damping function that depends on the frequency of the wave. The wave

is damped by adding a pressure term at the free surface which counteracts the wave motion.

Third, the free surface boundary conditions for pressure and velocity are calculated using the assumption that the stresses are continuous at the free surface. When the fluid is considered incompressible and the curvature of the free surface is neglected in the viscous stress terms, continuity of the normal stresses gives

$$-p + 2\mu \frac{\partial u_n}{\partial \boldsymbol{n}} = -p_0 + \sigma\kappa. \tag{2.6}$$

Analogously, continuity of the tangential stresses gives

$$\mu \left(\frac{\partial u_n}{\partial \boldsymbol{t}} + \frac{\partial u_t}{\partial \boldsymbol{n}} \right) = 0. \tag{2.7}$$

Here, $\boldsymbol{n}$ is the normal of the free surface and $\boldsymbol{t}$ the direction tangential to the free surface. Likewise, u_n and u_t are the velocity components normal and tangential to the free surface. Furthermore, p_0 is the atmospheric pressure, σ the surface tension and κ the curvature of the free surface. The curvature of the free surface is computed as $\kappa = \nabla \cdot \boldsymbol{n}$.

2.3 Numerical model

The discretisation of the mathematical model is presented in two dimensional terms, to avoid complex and unintelligible notations. The extension to three dimensions is straightforward. The discretisation takes into account moving bodies and cut cells. After the definition of the grid and geometry, the discretisation is given in space, time and with special attention to the free surface. In the closing section some attention is given to the cut cell method in three dimensions.

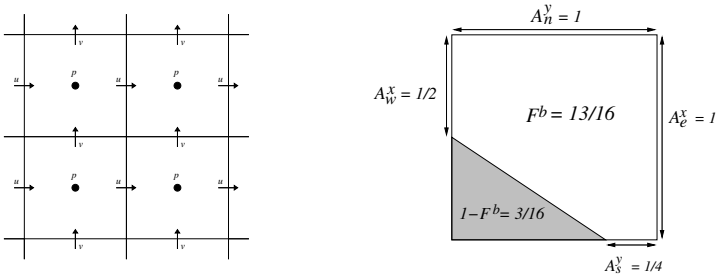

Figure 2.1: *Staggered arrangement of the velocity components and pressure (left) and an illustration of volume and edge apertures (right).*

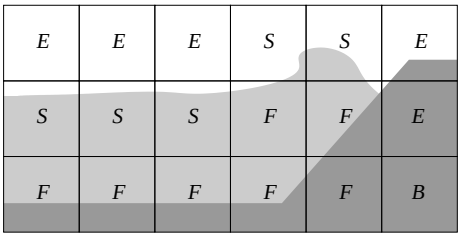

Figure 2.2: *Cell labelling: B(oundary), E(mpty), S(urface) and F(luid) cells.*

For a thorough description of the discretisation see the PhD-theses of Kleefsman [14] and Fekken [7]. This section is based on these two theses, but is less elaborate.

2.3.1 Grid and geometry definition

To solve the flow equations numerically, the computational domain is covered with a fixed Cartesian grid. The variables are staggered, which means that the velocities are defined on the cell faces and the pressure is defined in the cell centres. See the left of Figure 2.1 for an example. The advantage of a staggered grid is that there is no odd/even decoupling in the pressure terms.

The presence of moving bodies implies that the body's geometry cuts through the fixed rectangular grid. This results in cells that are partly filled with geometry. These cells are called cut cells. To store the information on the cut cells in the Cartesian grid, volume and edge apertures are used. A volume aperture F^b is the fraction of a cell that is open to fluid flow. This means that the fraction of solid body is given by $1 - F^b$. Edge apertures A^x, A^y and A^z give the fractions of the cell interface that are open to fluid flow in x, y and z-direction respectively. In the right of Figure 2.1 a two dimensional example is given.

Since the flow equations are not solved in the whole computational domain, the cells are labelled to distinguish between cells of different character. First, cells that are completely filled by geometry are called B(oundary) cells. These cells have volume aperture $F^b = 0$. Cells that do not contain fluid, but are available for possible fluid flow are labelled E(mpty). Subsequently, all cells adjacent to E-cells that contain fluid are S(urface) cells. Finally, the remaining cells are F(luid) cells. The flow equations are solved in all F and S cells. See Figure 2.2 for an example of cell labelling.

2.3.2 Spatial discretisation of the equations

The discretisation of the flow equations is described separately for all terms in the equations for the conservation of mass and the conservation of mo-

mentum. The discretisation is explained in x-direction on a uniform grid (thus Δx and Δy are constant) for one computational cell. The underlying philosophy of the discretisation is that the favourable (symmetric) properties of the continuous operators should also be properties of the discrete operators. See [28] for details.

Conservation of mass

For the discretisation of the conservation of mass, Equation (2.1) is applied to a computational cell. Without the presence of moving bodies the discretisation results in

$$\oint_\Gamma \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma \;\;\widehat{=}\;\; u_e A_e^x \Delta y + v_n A_n^y \Delta x - u_w A_w^x \Delta y - v_s A_s^y \Delta x \;=\; 0. \quad (2.8)$$

Here, the notation is used as depicted in the computational cell in the right of Figure 2.1, with u_* and v_* defined at the same edges where A_*^x and A_*^y are defined.

When a geometry moves in or out of the cell, Equation (2.8) has to be adjusted. Suppose that in the cell the geometry has a moving edge of length ℓ with normal $\boldsymbol{n}_b$ and velocity $\boldsymbol{u}_b = (u^b, v^b)$. Conservation of mass gives

$$u_e A_e^x \Delta y + v_n A_n^y \Delta x - u_w A_w^x \Delta y - v_s A_s^y \Delta x + \ell(\boldsymbol{u}_b\cdot\boldsymbol{n}_b) = 0. \quad (2.9)$$

The term $\ell(\boldsymbol{u}_b\cdot\boldsymbol{n}_b)$ is positive if the geometry moves out of the cell. When ℓ and $\boldsymbol{n}_b$ are expressed in terms of the edge apertures as $\ell = ||(\Delta y(A_w^x - A_e^x), \Delta x(A_s^y - A_n^y))||$ and $\boldsymbol{n}_b = (\Delta y(A_w^x - A_e^x), \Delta x(A_s^y - A_n^y))/\ell$, we have

$$\ell(\boldsymbol{u}_b\cdot\boldsymbol{n}_b) = u^b\ (A_w^x - A_e^x)\ \Delta y + v^b\ (A_s^y - A_n^y)\ \Delta x. \quad (2.10)$$

Conservation of momentum

In the discretisation of the momentum equations, the control volumes of the finite volume discretisation are formed around the velocity variables. Since the velocities are positioned at the cell edges, the control volumes employed are not the Cartesian cells: a control volume is defined as the sum of half the open parts of the neighbouring cells. See the left of Figure 2.3 for the control volume (of size F_c^b) around the variable u_c. The faces of the control volume have the uppercase subscripts E, W, N and S, denoting the east, west, north and south face respectively. The discretisation is performed around a central cell (subscript c). The neighbouring velocities and pressure terms have the lowercase subscripts e, w, n and s. See the right of Figure 2.3.

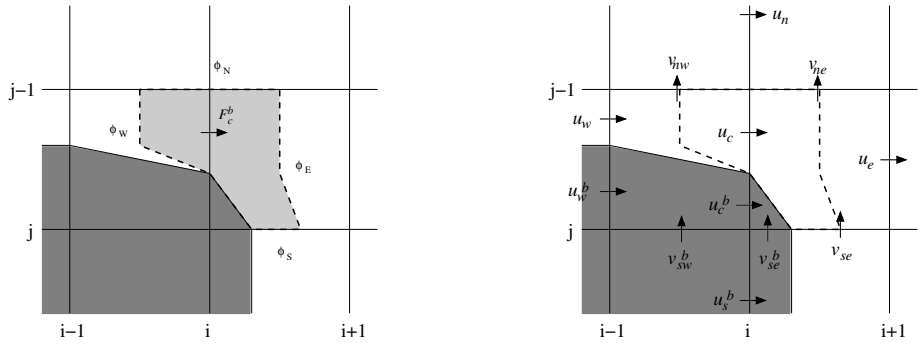

Figure 2.3: *Control volume near a cut cell (left). Names and position of the staggered variables near a moving boundary (right).*

(1) The volume integral of the time derivative is discretised in space using the midpoint rule. This results in

$$\int_\Omega \frac{\partial u}{\partial t}\,\mathrm{d}\Omega \quad \widehat{=} \quad \frac{\partial u_c}{\partial t}\,F_c^b\,\Delta x\,\Delta y. \tag{2.11}$$

(2) The convective term is computed by calculating the convective fluxes through the control volume's boundaries. As depicted in the left of Figure 2.3, ϕ_E, ϕ_W, ϕ_N and ϕ_S are the convective mass fluxes through the control volume's faces. The x-component of the convective term becomes

$$\oint_\Gamma u\boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma \quad \widehat{=} \quad u_E\phi_E - u_W\phi_W + u_N\phi_N - u_S\phi_S. \tag{2.12}$$

The velocity terms are computed by simply taking the average of the velocities at the cell faces

$$\begin{aligned} u_E &= \tfrac{1}{2}(u_e + u_c), & u_W &= \tfrac{1}{2}(u_w + u_c), \\ u_N &= \tfrac{1}{2}(u_n + u_c), & u_S &= \tfrac{1}{2}(u_s + u_c). \end{aligned} \tag{2.13}$$

The computation of the fluxes is similar to the calculation of the velocities. When A_*^x and A_*^y are the boundary apertures belonging to the cell faces where u_* and v_* are defines, this yields

$$\begin{aligned} \phi_E &= \tfrac{1}{2}\,(u_e A_e^x \Delta y + u_c A_c^x \Delta y) \;+\; \phi_E^{mov}, \\ \phi_W &= \tfrac{1}{2}\,(u_w A_w^x \Delta y + u_c A_c^x \Delta y) \;+\; \phi_W^{mov}, \\ \phi_N &= \tfrac{1}{2}\,(v_{ne} A_{ne}^y \Delta x + v_{nw} A_{nw}^y \Delta x) \;+\; \phi_N^{mov}, \\ \phi_S &= \tfrac{1}{2}\,(v_{se} A_{se}^y \Delta x + v_{sw} A_{sw}^y \Delta x) \;+\; \phi_S^{mov}. \end{aligned} \tag{2.14}$$

The additional terms ϕ_*^{mov} account for the influence of the moving boundary (velocity $\boldsymbol{u}_b$). They are given by

$$\begin{aligned}
\phi_E^{mov} &= \tfrac{1}{2}\Delta y(A_c^x - A_e^x)u_r^b, \\
\phi_W^{mov} &= \tfrac{1}{2}\Delta y(A_c^x - A_w^x)u_l^b, \\
\phi_N^{mov} &= \tfrac{1}{2}\Delta x(\max(0,(A_{se}^y - A_{ne}^y)v_r^b) + \max(0,(A_{sw}^y - A_{nw}^y)v_l^b)), \\
\phi_S^{mov} &= \tfrac{1}{2}\Delta x(\max(0,(A_{ne}^y - A_{se}^y)v_r^b) + \max(0,(A_{nw}^y - A_{sw}^y)v_l^b)).
\end{aligned} \tag{2.15}$$

When a body moves out of the cell, the 'moving mass flux' ϕ_*^{mov} is positive, thus letting mass flow into the cell. The max-terms that appear distinguish whether the body moves into or out of the cell.

Substituting the expressions (2.13)–(2.15) in Equation (2.12) gives the discrete convective operator

$$\oint_S u\boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}S \quad \widehat{=} \quad c_c u_c + c_e u_e - c_w u_w + c_n u_n - c_s u_s, \tag{2.16}$$

where the off-diagonal coefficients are given by

$$c_e = \tfrac{1}{4}\Delta y\Big(u_e A_e^x + u_c A_c^x + (A_c^x - A_e^x)u_r^b\Big) \tag{2.17a}$$

$$c_w = \tfrac{1}{4}\Delta y\Big(u_w A_w^x + u_c A_c^x + (A_c^x - A_w^x)u_l^b\Big) \tag{2.17b}$$

$$\begin{aligned}
c_n = \tfrac{1}{4}\Delta x\Big(&v_{ne}A_{ne}^y + v_{nw}A_{nw}^y + \max(0,(A_{se}^y - A_{ne}^y)v_r^b) \\
&+ \max(0,(A_{sw}^y - A_{nw}^y)v_l^b)\Big)
\end{aligned} \tag{2.17c}$$

$$\begin{aligned}
c_s = \tfrac{1}{4}\Delta x\Big(&v_{se}A_{se}^y + v_{sw}A_{sw}^y + \max(0,(A_{ne}^y - A_{se}^y)v_r^b) \\
&+ \max(0,(A_{nw}^y - A_{sw}^y)v_l^b)\Big),
\end{aligned} \tag{2.17d}$$

and the central coefficient c_c is given by

$$\begin{aligned}
c_c = {} & \tfrac{1}{4}\Big(\Delta y\big(u_c A_c^x - u_w A_w^x + (A_w^x - A_c^x)u_l^b\big) \\
&\quad + \Delta x\big(v_{nw}A_{nw}^y - v_{sw}A_{sw}^y + (A_{sw}^y - A_{nw}^y)v_l^b\big)\Big) \\
&+ \tfrac{1}{4}\Big(\Delta y\big(u_e A_e^x - u_c A_c^x + (A_c^x - A_e^x)u_r^b\big) \\
&\quad + \Delta x\big(v_{ne}A_{ne}^y - v_{se}A_{se}^y + (A_{se}^y - A_{ne}^y)v_r^b\big)\Big). \\
= {} & 0
\end{aligned} \tag{2.18}$$

To see that this coefficient reduces to zero, recognise the discrete continuity equations (see Equations (2.9) and (2.10)) for the eastern and western cell

in the expression. Since these are both zero, the total central coefficient reduces to zero.

Comparing the four off-diagonal coefficients, it becomes clear that the discrete convective operator is skew-symmetric. This is a favourable property since it implies that the skew-symmetric property of the continuous convective operator is conserved. This is also the reason that in the expressions of (2.13) simple averages are preferred over more accurate weighted averages [28].

The central discretisation described above has the disadvantage that spurious oscillations (wiggles) appear in the numerical solution. A remedy for these unphysical oscillations is adopting an upwind rather than a central discretisation. An upwind discretisation is equivalent to a central discretisation with an extra coefficient in the diffusive term. This artificial viscosity ν_a is added to the kinematic viscosity coefficient $\nu = \mu/\rho$ in the diffusive term [7, 14, 27].

(3) The x-component of the pressure term is discretised as a boundary integral, resulting in

$$\oint_\Gamma p n_x \, \mathrm{d}\Gamma \quad \widehat{=} \quad (p_e - p_w) \; A_c^x \; \Delta y. \tag{2.19}$$

As in the convective case, this discretisation preserves the symmetry of the underlying continuous operator. Writing the equations in differential form, reduces the continuity equation to $\nabla \cdot \boldsymbol{u} = 0$ and the pressure term to $-\nabla p$. Since analytically the gradient is the negative transpose of the divergence operator, that is $\nabla = -(\nabla\cdot)^T$, this should hold as well for the discrete gradient and divergence operators, which holds indeed (see for an example [7, p. 31]).

(4) Discretising the surface integral of the diffusive term is not straightforward since the derivatives of the velocities are needed at the control volume's boundary and these need not be zero. To overcome problems at the boundaries, the diffusive term (with artificial viscosity) is rewritten as a volume integral,

$$(\nu_a + \nu) \oint_\Gamma \nabla u \cdot \boldsymbol{n} \, \mathrm{d}\Gamma \; = \; (\nu_a + \nu) \int_\Omega \nabla \cdot \nabla u \, \mathrm{d}\Omega \tag{2.20}$$

and discretised with the midpoint rule. To derive the second order derivative at the position of u_c, the first order derivatives should be calculated at control volumes boundaries. A downside of calculating the first order derivatives this way is that it implicates division by possibly very short distances at cut cells. To prevent instabilities caused by these divisions, the geometry is handles in a 'staircase' manner – cut cells are thus treated as if they

were uncut. Although inaccurate, this way of discretising the diffusive term does not really influence the accuracy of the solver, since the simulations are convection driven. This results in

$$\nabla \cdot \nabla u \;\; \widehat{=} \;\; \frac{u_e - u_c}{(\Delta x)^2} - \frac{u_c - u_w}{(\Delta x)^2} + \frac{u_n - u_c}{(\Delta y)^2} - \frac{u_c - u_s}{(\Delta y)^2}, \tag{2.21}$$

such that the discrete diffusive term is written as

$$\begin{aligned}(\nu_a + \nu) \oint_\Gamma \nabla u \cdot \boldsymbol{n} \; \mathrm{d}\Gamma \;\; \widehat{=} \;\; & (\nu_a + \nu) F_c^b \Big(\frac{u_e + u_w}{\Delta x} \Delta y + \frac{u_n + u_s}{\Delta y} \Delta x \\ & - (\frac{2}{\Delta x} \Delta y + \frac{2}{\Delta y} \Delta x) \; u_c \Big). \end{aligned} \tag{2.22}$$

Comparing the coefficients of the five variables, it is clear that the discrete diffusive operator is symmetric. Since the central coefficient is negative and the sum of the other four coefficients equals the absolute central coefficient, the resulting discrete difference operator is also negative-definite. Again, the properties of the continuous discrete difference operator are conserved [28].

(5) The only external force is gravity (pointing in z-direction). Using a discretisation similar to the pressure term, that is using a boundary intergal, the z-component of the gravity term is given by

$$\int_\Omega g_z \; \mathrm{d}\Omega \;\; = \;\; \oint_\Gamma g z \boldsymbol{n} \; \mathrm{d}\Gamma \;\; \widehat{=} \;\; g \Delta z A_c^z \Delta x. \tag{2.23}$$

Schematic form of the discretisation

The discretisations given above yield the discrete operators on the vectors $\boldsymbol{u}_h$ (containing all discrete interior velocity term), $\boldsymbol{u}_b$ (containing all discrete boundary velocity term) and $\boldsymbol{p}_h$ (containing all discrete interior pressure terms).

For conservation of mass, the discrete divergence operator is given by the matrix $\mathcal{M}$. Splitting the operator $\mathcal{M}$ into a part $\mathcal{M}^0$ working on the interior and a part $\mathcal{M}^b$ working on the boundary, the discrete conservation of mass is also written as $\mathcal{M}^0 \boldsymbol{u}_h = -\mathcal{M}^b \boldsymbol{u}_b$. For the conservation of momentum, all control volumes $F_c^b \; \Delta x \; \Delta y$ are assembled in the diagonal matrix $\boldsymbol{\Omega}$. The discrete convective matrix is depending on the interior and exterior velocities and denoted as $\mathcal{C} = \mathcal{C}(\boldsymbol{u}_h, \boldsymbol{u}_b)$. Furthermore $\mathcal{G}$ and $\mathcal{D}$ are the matrices for the discrete gradient and diffusion. The vector $\mathcal{F}$ contains the gravity terms.

As observed in the discretisation given above, the discrete diffusive matrix $\mathcal{D}$ is symmetric negative definite and the convective matrix $\mathcal{C}(\boldsymbol{u}_h, \boldsymbol{u}_b)$ is skew-symmetric. The discrete gradient (working only on the interior) is

the negative transpose of the interior part of the divergence operator, that is $\mathcal{M}^0 = -\mathcal{G}^T$.

Recapitulating, the spatial discretisation of the Navier-Stokes equations is schematically given by

$$\mathcal{M}^0 \boldsymbol{u}_h = -\mathcal{M}^b \boldsymbol{u}_b, \tag{2.24a}$$

$$\boldsymbol{\Omega} \frac{\partial \boldsymbol{u}_h}{\partial t} = -\mathcal{C}(\boldsymbol{u}_h, \boldsymbol{u}_b)\boldsymbol{u}_h + \frac{1}{\rho}\left(\mathcal{D}\boldsymbol{u}_h - \mathcal{G}\boldsymbol{p}_h\right) + \mathcal{F}. \tag{2.24b}$$

2.3.3 Temporal discretisation and solution method

For the time discretisation of Equation (2.24) the explicit forward Euler method is adopted. This implies that all diffusive terms and external forces are considered at the previous time level. The pressure is considered at the current time level. At time level $t_{n+1} = \Delta t(n+1)$, this results in

$$\mathcal{M}^0 \boldsymbol{u}_h^{n+1} = -\mathcal{M}^b \boldsymbol{u}_b^{n+1}, \tag{2.25a}$$

$$\boldsymbol{\Omega} \frac{\boldsymbol{u}_h^{n+1} - \boldsymbol{u}_h^n}{\Delta t} = -\mathcal{C}(\boldsymbol{u}_h^n, \boldsymbol{u}_b^n)\boldsymbol{u}_h^n + \frac{1}{\rho}\left(\mathcal{D}\boldsymbol{u}_h^n - \mathcal{G}\boldsymbol{p}_h^{n+1}\right) + \mathcal{F}. \tag{2.25b}$$

To ensure that the velocity field stays divergence free, the continuity equation is discretised at the new time level t_{n+1}.

Since this system does not give the pressure explicitly, some rearrangements are necessary to solve it. It is natural to consider Equation (2.25b) as an equation for the velocity. Implicitly, the Equation (2.25a) yields an equation for the pressure in the form of the Poisson equation, that is derived as follows. Defining

$$\mathcal{R}^n \equiv \boldsymbol{u}_h^n - \Delta t \boldsymbol{\Omega}^{-1}\left(\mathcal{C}(\boldsymbol{u}_h^n, \boldsymbol{u}_b^n)\boldsymbol{u}_h^n - \frac{1}{\rho}\mathcal{D}\boldsymbol{u}_h^n - \mathcal{F}\right), \tag{2.26}$$

$$\mathcal{G}_{\rho,\Omega} \equiv \frac{1}{\rho}\boldsymbol{\Omega}^{-1}\mathcal{G}, \tag{2.27}$$

the Equations (2.25a) and (2.25b) are rewritten in matrix form as

$$\begin{pmatrix} \mathcal{M}^0 & 0 \\ \boldsymbol{I} & \Delta t \mathcal{G}_{\rho,\Omega} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_h^{n+1} \\ \boldsymbol{p}_h^{n+1} \end{pmatrix} = \begin{pmatrix} -\mathcal{M}^b \boldsymbol{u}_b^{n+1} \\ \mathcal{R}^n \end{pmatrix}. \tag{2.28}$$

Applying Gaussian elimination gives

$$\begin{pmatrix} 0 & \Delta t \mathcal{M}^0 \mathcal{G}_{\rho,\Omega} \\ \boldsymbol{I} & \Delta t \mathcal{G}_{\rho,\Omega} \end{pmatrix} \begin{pmatrix} \boldsymbol{u}_h^{n+1} \\ \boldsymbol{p}_h^{n+1} \end{pmatrix} = \begin{pmatrix} \mathcal{M}^0 \mathcal{R}^n + \mathcal{M}^b \boldsymbol{u}_b^{n+1} \\ \mathcal{R}^n \end{pmatrix}, \tag{2.29}$$

where a Poisson equation for the pressure is recognised in the first row. Once the Poisson equation is solved, the velocity is calculated using the second row. The Poisson equation for the pressure is solved iteratively using the Successive Over Relaxation (SOR) method. The relaxation parameter is determined and continually optimised during the iterations [2].

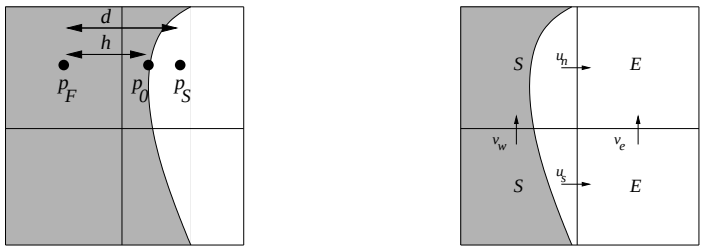

Figure 2.4: *Extrapolation of the pressure boundary condition (left) and the calculation of the EE-velocity boundary conditions (right)*

2.3.4 Discretisation at the free surface

The presence of a free surface is perhaps the most prominent feature of the flow simulated by COMFLOW and a correct discretisation is important for both the accuracy and robustness of the solver. Regarding the discretisation of the free surface, the boundary conditions and the displacement are the two important aspects.

Pressure boundary conditions

At the free surface, the boundary condition for the pressure is analytically given by the normal free surface condition (2.6). Since most simulations performed by COMFLOW are convection-dominated and gravity-driven, the viscous term ($2\mu\frac{\mathrm{d}u_n}{\mathrm{d}\boldsymbol{u}}$) and the surface tension ($\sigma\kappa$) are neglected. Hence, the pressure boundary condition at the free surface reduces to $p = p_0$.

While this boundary condition is positioned at the free surface, the pressure is positioned in the cell centre. Hence, extrapolation is applied. The pressure in a S-cell is calculated using a neighbouring F-cell and the pressure p_0 at the free surface. See the left of Figure 2.4 for an example. With the notation used in this figure, the extrapolated value p_S is given by

$$p_S = p_F - \frac{h}{d}(p_0 - p_F). \tag{2.30}$$

If a S-cell has more neighbouring F-cells, the orientation of the surface (in two dimensions either horizontal or vertical) determines which cell to make use of. When there are no neighbouring F-cells, $p_S = p_0$ is employed.

For simulation in micro-gravity environments, the surface tension must be taken into account, as it is the driving force in the absence of gravity. See Gerrits [8] for details.

Velocity boundary conditions

Since the velocities are positioned on the cell's edges, they can be classified depending on the labels of the two adjacent cells. Hence there are FF-, FS-, SS-, SE- and EE-velocities. The first three are calculated by solving the momentum equation (see Section 2.3.2). The last two occur in the neighbourhood of the free surface and need a special treatment since no equations are defined in the empty cells.

First, for the computation of the SE-velocities, there are two options. They can either be computed by an extrapolation from the interior, or by applying conservation of mass. The choices made in the discretisation of this term are rather delicate and influence the robustness and accurateness of the solver. A detailed description and a discussion of the options is given in [7] and [14].

Second, EE-velocities are needed when the momentum equations are solved at SS-velocities. Take for example the right frame of Figure 2.4, where v_e is needed to solve the momentum equations at v_w. As the EE-velocities are more or less tangential to the free surface, they are calculated with the tangential free surface condition given by Equation (2.7). When the derivatives are discretised in the directions of the Cartesian grid, the condition becomes (in two dimensions)

$$\mu \left(\frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \right) = 0. \tag{2.31}$$

Using a central discretisation results in

$$u_n = u_s - \frac{\Delta y}{\Delta x}(v_e - v_w). \tag{2.32}$$

Here, u_s is a known SS-velocity (from a previous time step) resulting from the momentum equation, and v_e and v_w are known SE-velocities calculated as described in the previous paragraph.

Displacement

Since the position of the free surface is not known in advance and varies in time, it should be computed as part of the solution. In Section 2.2.2, the function $\boldsymbol{S}$ is introduced and the free surface is tracked in time by Equation (2.5), that is $D\boldsymbol{S}/Dt = 0$.

The discrete version hereof is given by the Volume of Fluid (VOF) method, that was first introduced by Hirt and Nichols in 1981 [12]. The discrete VOF function F_S has values between zero and one, indicating the fraction of the cell that is filled with liquid. To calculate the evolution of the free surface in the next time step, the new VOF values are updated by

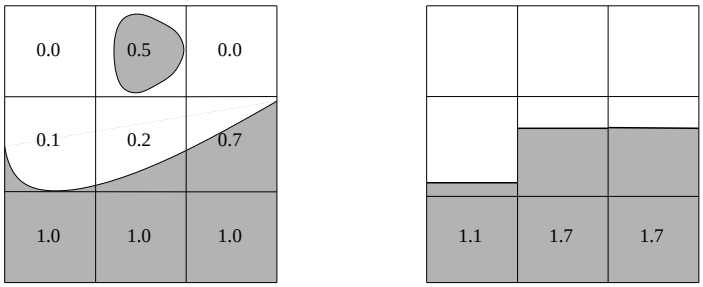

Figure 2.5: *VOF values (left) and VOF values after applying the local height function (right).*

applying $DF_S/Dt = 0$. Those cells that have $0 < F_S < 1$ are free surface cells, see the left of Figure 2.5 for an example.

Based on the VOF fractions, the free surface is reconstructed. This can either be done by a piecewise constant reconstruction where the interface is parallel with one of the coordinate axes (SLIC) or by a piecewise linear reconstruction (PLIC).

A disadvantage of the VOF method is that the calculation of the new values of F_S may yield $F_S < 0$ or $F_S > 1$. This can partly be prevented by applying limiters, but still the values of F_S may exceed the bounds. In the original method by Hirt and Nichols this is solved by truncating F_S. Unfortunately, this has the effect that mass is gained or lost. Another disadvantage of the VOF method is that small droplets disconnect from the free surface, causing an unphysical spray of liquid above the free surface (also called flotsam and jetsam). More details are found in [7, 12, 14].

To overcome these two disadvantages, the VOF method is adapted. Once the new values for F_S are calculated, a local height function is introduced in S-cells. Around the S cell a 3×3 (in two dimensions) block is considered. In this block the orientation of the free surface is determined as either horizontal or vertical. Subsequently, the local heights in each column (horizontal orientation) or row (vertical orientation) are determined by summing the F_S values. This is depicted in the right frame of Figure 2.5. The value of the concerning S cell is now determined by redistributing the fluid. Take for example the central cell of Figure 2.5, that had value $F_S = 0.2$ before applying the height function and $F_S = 0.7$ afterwards. The VOF method adapted with the height function is now mass conservative and produces less unphysical spray.

2.3.5 Grid: cut cell method

We have already seen that Comflow uses a Cartesian cut cell method to describe arbitrary moving geometries. The discretisation of the flow equa-

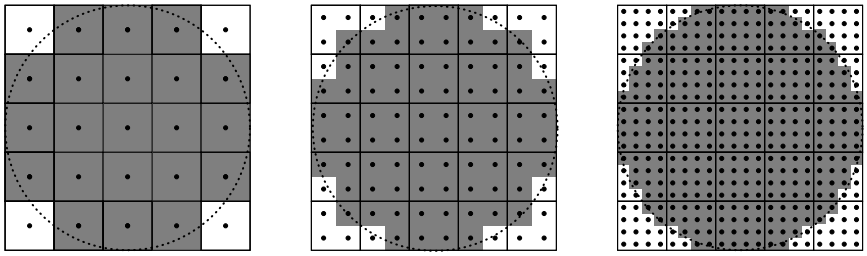

Figure 2.6: *Examples of the approximation of the volume of a solid body by markers.*

tions in cut cells is described above in Section 2.3.2. Here, special attention is given to small cut cells and (moving) cut cells in three dimensions.

First, small cut cells need special attention, since terms in $\mathbf{\Omega}$ can become very small, yielding an $\mathbf{\Omega}^{-1}$ with arbitrarily large eigenvalues. These large eigenvalues can cause problems regarding the stability of the method, especially near moving boundaries. In [14, §4] it is shown that these effects can be cancelled, without merging small cut cells with neighbouring cells.

Second, calculating cut cells is fairly straightforward in two dimensions. Also for moving solid bodies, the calculation of the new cut cells for the new time step is not too intricate. However, the calculation of cut cells for moving bodies in three dimensions is not as easy as it is in two. Since COMFLOW often deals with three dimensional geometries of arbitrary shape, an approximative method is implemented. When using an approximate instead of an exact calculation, it should be kept in mind that the approximative method should still model the conservation of fluid volume as accurately as possible.[3]

The first step in the approximate cut cell method is to describe the geometry with the use of markers. The markers are uniformly spread in each cell and each marker is the centre of a virtual rectangular box. The volume of the geometry is approximated by the total volume of those virtual boxes whose marker is inside the solid geometry, see Figure 2.6. The virtual boxes determine the cells' volume apertures F_b. In Figure 2.6 for example, the volume apertures of the cells in the upper left corner are $F_b = 0$, $F_b = \frac{3}{4}$ and $F_b = \frac{13}{16}$ from left to right.

Although the virtual boxes approximate the volume of a solid body quite accurately, they are poorly suited to describe the surface of the body. The second step in the approximate cut cell method is therefore to describe the body's surface using a piecewise linear interface reconstruction (PLIC). This

[3]Even though the figures used below to exemplify the method are given in two dimensions the approximate method is especially designed for three dimensional simulation: in two dimensions exact calculations are adopted.

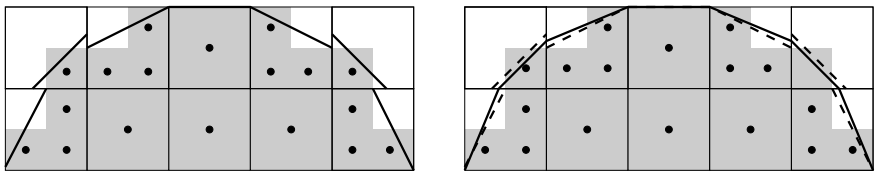

Figure 2.7: *Example of the piecewise linear reconstruction of the surface of a solid body.*

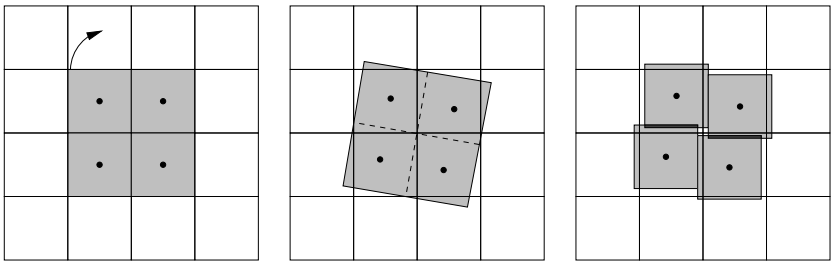

Figure 2.8: *Rotation of a rectangular body with one marker per cell; (left) the original position, (middle) the exact rotation, (right) approximate rotation.*

is essentially the same as the reconstruction of a free surface in the Volume-of-Fluid method. The PLIC method uses the fractions of solid body in each cell, $F_b - 1$, to discretely compute a cell's surface normal. In two dimensions this would for example mean that the fractions $F_b - 1$ of the eight surrounding cells are used to compute a cell's surface normal. In each cell the surface is approximated using the normal and the fractions of solid body of the cell. As is seen in the left of Figure 2.7, the reconstruction method does not guarantee a continuous surface. To assure the continuity the average is taken at the interfaces, see the right of Figure 2.7. The reconstructed surface is stored in the edge apertures.

The final step in the approximate cut cell method deals with the displacement of a body. When the solid body is moving the markers move accordingly. The volume boxes however, do not follow the movement of the solid body exactly. To avoid intricate calculations on volume apertures the orientation of the virtual volume boxes remains the same as the orientation of the Cartesian grid. The result of this approximate procedure is depicted in the right of Figure 2.8. The exact orientation of the volume boxes is given in the middle of 2.8. When assuring that no 'holes' appear in the solid body's interior cells, this approximate procedure turns out to be as good as volume accurate.

2.4 Summary

The underlying scheme of COMFLOW is based on the solution of the incompressible Navier-Stokes equations for a one fluid (liquid) system. The interface between liquid and gas is considered as a moving boundary. The equations are given in conservative integral form, since a finite volume approach is adopted in the discretisation.

The computational domain encompasses the fluid region on a stationary background Cartesian grid. Tracking the solid boundaries of moving bodies is accomplished using a cut cell method. The variables have a staggered arrangement, with the pressure in the cell centres and the velocities on the cell boundaries, in both cut and uncut cells.

The free surface is diplaced using the Volume of Fluid (VOF) method together with a local height function, resulting in a method that is mass conserving. The boundary conditions at the free surface are given by applying the continuity-condition for the tangential and normal stresses, as well as an extrapolation from the interior and the conservation of mass. The discretisation of the boundary conditions is crucial for the accuracy an accurateness of the method.

In space, an explicit discretisation is used. The discretisation is performed in such a way that the discrete operators have the same symmetry-properties as the underlying continuous ones. This means that the discrete convective operator is skew-symmetric and the discrete diffusive operator symmetric negative definite.

The temporal discretisation uses an explicit forward Euler discretisation. The resulting system of equations is solved using a Poisson equation for the pressure. Once the pressure is known, the discrete equation momentum equation yields the velocity-field

Chapter 3

Computational model: Amazon-sc

This chapter describes the computational model as implemented in the simulation tool Amazon-sc. The physical and mathematical model are given in the next two sections. Before turning to the numerical model, Section 3.3 contains some fundamental theoretical background. The chapter concludes with a summary of the computational model.

3.1 Physical model

Amazon-sc has been developed at the Manchester Metropolitan University (UK) to simulate flow problems with both moving solid bodies and free surfaces. The solver is employed to simulate the flow-field of violent waves interacting with sea walls. Furthermore it is applied to simulate wave energy devices such as the 'Oscillating wave surge converter' (OWSC) and the 'Limpet Oscillating Water Column Wave Power Device'. Water entry phenomena with wedge-shaped objects are also simulated. Details of these simulations are given for example in [5, 19, 21, 22].

Given the complexity of free surface flow problems, some physical assumptions are necessary to make the resulting formulation trackable. An import assumption is the way the interface between liquid (water) and gas (air) is treated. In Amazon-sc the fluid interface is treated as a contact discontinuity in the density field, which is captured automatically as part of the solution. The flow is thus solved in both the gas and liquid region. Furthermore, the flow is considered isothermal and incompressible. Since the simulations performed with Amazon-sc are convection-dominated and diffusive effects are of minor importance, the viscous effects are neglected.

The flow is simulated in two dimensions, although some parts of the solver have been extended to three dimensions.

3.2 Mathematical model

The most general mathematical model for describing viscous Newtonian flows are the Navier-Stokes equations, consisting of the continuity and momentum equations. As diffusive effects are ignored, the viscous flux is considered zero. This means that the mathematical model of Amazon-sc is based on a simplification of the Navier-Stokes equations: the Euler equations.

The mathematical model of Amazon-sc is presented in general three dimensional form (although the simulations are all in two dimensions). Moving bodies and cut cells are not accounted for, but extending the equations to accommodate moving boundaries and cut cells is fairly straightforward. A characteristic of Amazon-sc is that the governing equations are extended with artificial compressibility constraints.

3.2.1 Governing equations

In the mathematical model, the Euler equations are presented in integral form for an arbitrary control volume Ω with boundary Γ. This presentation of the Euler equations is natural since the equations are discretised with the finite volume method.

Although the fluid flow in Amazon-sc is considered to be incompressible, the density ρ is not constant. The density in air and water typically differs by a factor of a thousand. Owing to the varying ρ, the continuity and momentum equations are given in the compressible form.

First, the compressible continuity equation, resulting from the conservation of mass, is given by

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega} \rho \,\mathrm{d}\Omega + \oint_{\Gamma} \rho \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma = 0. \tag{3.1}$$

The symbols represent the velocity in three dimensions $\boldsymbol{u} = (u, v, w)^T$ and outward-pointing normal vector $\boldsymbol{n}$ of Γ. Nevertheless, the incompressibility constraint is valid, which means that

$$\oint_{\Gamma} \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma = 0 \tag{3.2}$$

should hold as well.

Second, applying the conservation of momentum results in the system of equations

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_{\Omega} \rho\boldsymbol{u}\,\mathrm{d}\Omega + \oint_{\Gamma} \boldsymbol{F}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma = \int_{\Omega} \rho\boldsymbol{g}\,\mathrm{d}\Omega. \tag{3.3}$$

The matrix $\boldsymbol{F} = \boldsymbol{F}_{\text{inv}}$ is the inviscid flux through the boundary of the control volume, given by

$$\boldsymbol{F} \equiv \rho \boldsymbol{u} \otimes \boldsymbol{u}^T + p\boldsymbol{I}. \tag{3.4}$$

The vector $\boldsymbol{g}$ denotes the acceleration due to body forces, which are only gravitational forces in the problems considered. The other symbols denote the pressure p, the density ρ, the 3×3 identity matrix $\boldsymbol{I}$ and the vector tensor product $\otimes$.

Perhaps the set of Equations (3.1)-(3.3) appears unusual, because of the presence of an equation for the density in (3.1). However, they do in fact form a solvable system of equations, as Kelecy and Pletcher ascertain in [13].

In addition to the discretisation by finite volume techniques, there is another argument for using the integral form of the Navier-Stokes equations in the case of a two-phase solver. The solution in the two-phase formulation has discontinuities at the free surface. These discontinuities necessitate the use of the integral formulation since in the divergence form a discontinuous solution would not be allowed. On the other hand, the integral formulation allows discontinuities in the form of weak solutions.

3.2.2 Equations with artificial compressibility

The ideas of artificial compressibility are incorporated in the mathematical model of Amazon-sc. This method was first introduced by Chorin in 1967 who applied it to solve the steady form of the equations [6]. He was inspired by the idea that solving the *compressible* form of the Euler equations has an apparent advantage compared to the *incompressible* form. This advantage is that the pressure appears in the compressible continuity equation, implying that there is an immediate link between the pressure and the velocity which is absent in the incompressible Euler equations. In view of this, he augmented the incompressibility constraint by a time derivative of the pressure, scaled with an appropriate constant β,

$$\frac{1}{\beta}\frac{\partial p}{\partial \tau} + \nabla \cdot \boldsymbol{u} = 0. \tag{3.5}$$

However, this method is only time accurate when a steady state solution is reached. Only for a steady state solution it is guaranteed that the velocity field remains divergence free, since then $\partial p/\partial \tau = 0$ for $\tau \to \infty$. To apply the method to the unsteady form of the equations, Chorin's method should be enhanced. This is done by introducing a dual-time stepping technique, which ensures that the method is time accurate for unsteady flow. The solution at every physical time step is computed by iterating in pseudo time until a steady solution is reached.

In the mathematical model of Amazon-sc the idea of artificial compressibility is not only used for the continuity equation, but for all three Equations (3.1), (3.2) and (3.3). They subsequently become

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega \rho \,\mathrm{d}\Omega + \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \rho \,\mathrm{d}\Omega = -\oint_\Gamma \rho \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma, \tag{3.6a}$$

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega \rho \boldsymbol{u} \,\mathrm{d}\Omega + \frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \rho \boldsymbol{u} \,\mathrm{d}\Omega = -\oint_\Gamma \boldsymbol{F}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma + \int_\Omega \rho \boldsymbol{g} \,\mathrm{d}\Omega \tag{3.6b}$$

$$\frac{1}{\beta}\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega p \,\mathrm{d}\Omega = -\oint_\Gamma \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma. \tag{3.6c}$$

The partial differential Equations (3.6) form the mathematical model for Amazon-sc. The same equations for multi-fluid flows are adopted by Pan and Chang [20]. Kelecy and Pletcher [13] suggest that both these equations and the equations without the pseudo-time terms in (3.6a) and (3.6b) can be used. They report that the former promotes the numerical stability of the iteration in pseudo-time. On the other hand it slows down the convergence of the subiteration.

With some adjustments in the notation – explained below – the set of Equations (3.6) is also given by the single equation

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega \boldsymbol{\phi} \,\mathrm{d}\Omega + \boldsymbol{I}_0\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \boldsymbol{\phi} \,\mathrm{d}\Omega = -\oint_\Gamma \boldsymbol{F}\cdot\boldsymbol{n}\,\mathrm{d}\Gamma + \int_\Omega \boldsymbol{G}\boldsymbol{\phi} \,\mathrm{d}\Omega. \tag{3.7}$$

The 5×1 vector $\boldsymbol{\phi}$ is the vector of the conserved variables density and momentum, augmented by the dimensionless pressure p/β :

$$\boldsymbol{\phi} \equiv (\rho, \rho u, \rho v, \rho w, p/\beta)^T. \tag{3.8}$$

The 5×5 matrix $\boldsymbol{G} \equiv [\ \boldsymbol{g}\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ \mathbf{0}\ ]$, with $\boldsymbol{g}$ the vector of the gravitational acceleration given by $\boldsymbol{g} = (0, g_x, g_y, g_z, 0)^T$. The values of g_x, g_y and g_z depend on the orientation of the system. The 5×5 matrix $\boldsymbol{I}_0$ is given by $\boldsymbol{I}_0 = \mathrm{diag}(1, 1, 1, 1, 0)$. Finally the 5×3 inviscid flux matrix is given by

$$\boldsymbol{F} \equiv \begin{bmatrix} \rho\boldsymbol{u}^T \\ \rho\boldsymbol{u}\otimes\boldsymbol{u}^T + p\boldsymbol{I} \\ \boldsymbol{u}^T \end{bmatrix}. \tag{3.9}$$

3.2.3 Boundary conditions

In the present solver there are two types of boundaries. The first is classified as a solid boundary, either stationary or moving. The second is an outlet or open boundary. No boundary conditions are needed at the free surface, since it is captured automatically as part of the solution.

At solid boundaries an impermeable condition is applied to the velocity and density. Thus, the normal component of the velocity on a boundary

is the same as the normal velocity of the boundary. When the boundary is stationary this means that the normal component of the velocity at the boundary is zero. There is however no no-slip condition for the velocity. This implies that the tangential boundary velocity may differ from the body's tangential velocity. Further, the density is assumed to have a zero normal gradient at the boundary, and is thus extrapolated form the interior. The pressure boundary condition is derived from the normal component of the momentum equations, evaluated at the wall. When the boundary is stationary this means $\nabla p = -\rho \boldsymbol{g}$.

At outlet boundaries the pressure is fixed and a zero gradient condition is applied to the velocity and density. This definition allows the fluid to enter the computational domain freely according to the local flow velocity and direction.

3.3 Theory on Godunov-type methods

The theoretical basis of AMAZON-SC's numerical scheme centres around the method presented in 1959 by Godunov, whose picture is given in Figure 3.1. He presented a method for solving non-linear systems of hyperbolic conservation laws such as the compressible Euler equations. It can be considered as a conservative extension of the first-order upwind scheme, introduced by Courant, Isaacson and Reeves in 1952. Upwind schemes utilise the physical properties of the flow to determine the direction of differencing: the *global* direction of the flow determines whether a upwind or downwind differencing is adopted. Godunov's method on the other hand presents a way of *locally* introducing the exact solution of the Euler equations into the discretisation. Thus taking the interaction between discretisation method and physical properties one step further [9, 17].

Following Hirsch [11, p.409] "the family of methods that call on exact or approximate local properties of the basic solutions to the Euler equations" are referred to as Godunov-type methods.

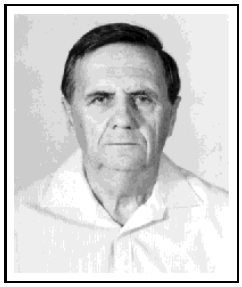

Figure 3.1: *Sergei Kanstantinovich Godunov (1929 –) [16].*

A thorough description of Godunov-type methods is given in the books by Hirsch [11] and Toro [26]. The following four sections are based on these two books.

3.3.1 Riemann problems

Fundamental to Godunov-type methods are solutions of Riemann problems. Before going into detail about Godunov's method, it is worth paying some attention to these problems.

A Riemann problem is defined by a partial differential equation with a specific set of initial conditions. The initial conditions are constant except for a single discontinuity. Take for example a partial differential equation for $\boldsymbol{\phi}(x,t)$. A Riemann problem is given by the initial conditions

$$\boldsymbol{\phi}(x,t_0) = \begin{cases} \boldsymbol{\phi}^- & \text{for } x < a \\ \boldsymbol{\phi}^+ & \text{for } x > a. \end{cases} \tag{3.10}$$

See for example the initial condition in the left of Figure 3.2 with $a = 0$.

To clarify the Riemann problem a little better, the solution for the Riemann problem of a hyperbolic system of equations with a constant coefficient matrix is considered. When the subscript denotes partial differentiation, the problem for the $m \times 1$ vector $\boldsymbol{\phi}$ is

$$\boldsymbol{\phi}_t + \boldsymbol{A}\,\boldsymbol{\phi}_x = 0, \tag{3.11a}$$

$$\boldsymbol{\phi}(x,t_0) = \begin{cases} \boldsymbol{\phi}^- & \text{for} \quad x < 0 \\ \boldsymbol{\phi}^+ & \text{for} \quad x > 0 \end{cases} \tag{3.11b}$$

Since the system is hyperbolic, $\boldsymbol{A}$ has real eigenvalues and linearly independent eigenvectors. This means that $\boldsymbol{A}$ can be diagonalised to $\boldsymbol{\Lambda} = \boldsymbol{K}^{-1}\boldsymbol{A}\boldsymbol{K}$. Here $\boldsymbol{\Lambda} = \operatorname{diag}(\lambda_1, \ldots, \lambda_m)$ is the matrix of ordered eigenvalues, that is $\lambda_1 < \cdots < \lambda_m$. The corresponding right eigenvectors are the columns of $\boldsymbol{K}$, denoted by $\boldsymbol{K}_1, \ldots, \boldsymbol{K}_m$. This diagonalisation gives us the opportunity to rewrite the equation in characteristic form. For $\boldsymbol{\psi} = \boldsymbol{K}^{-1}\boldsymbol{\phi}$ equation (3.11) becomes

$$\boldsymbol{\psi}_t + \boldsymbol{\Lambda}\,\boldsymbol{\psi}_x = 0, \tag{3.12a}$$

$$\boldsymbol{\psi}(x,t_0) = \boldsymbol{K}^{-1}\boldsymbol{\phi}(x,t_0). \tag{3.12b}$$

In this characteristic form the equations for each entry of $\boldsymbol{\psi}$ are decoupled. Considering each entry separately we see that the solution is constant along the characteristic curves satisfying $\mathrm{d}x/\mathrm{d}t = \lambda_i$, for $i = 1, \ldots, m$. The eigenvalues λ_i are the characteristic speeds at which the discontinuity propagates. The structure of the solution of this Riemann problem is thus as depicted in the right of Figure 3.2. The axes in the left and right frames of Figure 3.2 are not the same: the left is plotted with (x, ϕ) and the right with (x, t).

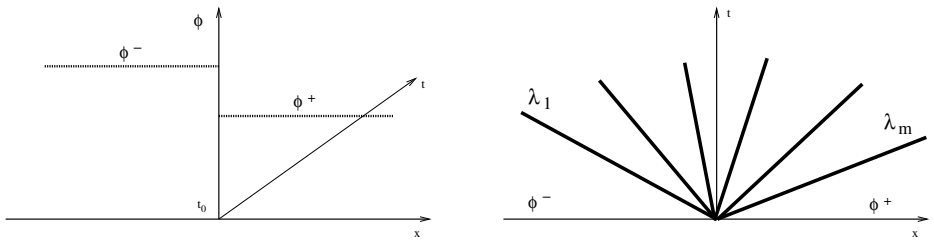

Figure 3.2: *Initial conditions for the Riemann problem (left). Development of the solution in time for a hyperbolic constant coefficient $m \times m$ Riemann problem (right).*

The solution $\boldsymbol{\phi}(x,t)$ in the wedge between λ_1 and λ_m is found by a superposition of waves. First note the solution only depends on the fraction x/t, since the solution is constant on lines through the origin. When the initial conditions are subsequently expressed in terms of the eigenvectors as

$$\boldsymbol{\phi}^- = \sum_{i=1}^{m} \alpha_i \boldsymbol{K}_i \quad \text{and} \quad \boldsymbol{\phi}^+ = \sum_{i=1}^{m} \beta_i \boldsymbol{K}_i, \tag{3.13}$$

the coefficients α_i and β_i are known. The solution is given by

$$\boldsymbol{\phi}(x/t) = \sum_{i=1}^{I} \beta_i \boldsymbol{K}_i + \sum_{i=I+1}^{m} \alpha_i \boldsymbol{K}_i. \tag{3.14}$$

Here, I is the largest integer with $i \leq I \leq m$ such that $x/t \geq \lambda_I$.

Although more complicated, the analytic solution of the Riemann problem for the compressible Euler equations can be found as well. As is the case in the example given above, the compressible Euler equations are hyperbolic, but the non-linearity of the equations makes the solution more complicated than the one described above. The solution does, however, exist and a detailed description of the solution is given in [26, chapter 4].

3.3.2 Original method of Godunov

Godunov's method typically consists of three distinct steps. These steps are illustrated in Figure 3.3 for a single variable in a one dimensional problem.

Step one is to define a piecewise constant approximation of the solution in each cell based on the known cell-centred data at t_n, see Figure 3.3.a. This reconstruction gives rise to a discontinuity at each cell face. In one, two and three dimensions this gives two, four and six discontinuities in every cell respectively: one for each boundary with an adjacent cell. Using the piecewise constant reconstruction, gives a method that is first order accurate in space.

In the second step, Figure 3.3.b, these discontinuous states are taken as the initial conditions to solve the partial differential equations locally. In other words, Riemann problems are solved at all cell faces. Since the exact solution of the local Riemann problem for the compressible Euler equations is known, this is introduced in the discretisation directly. This way, the method introduces the exact local properties of the flow into the solution, implying that the method uses information of the flow in an upwind manner.

In the third step, the solution at the next time step t_{n+1} is computed, see Figure 3.3.c. The evolution of the flow to the next time step results from wave interactions originating at the boundaries of the cell. The solution at t_{n+1} is computed by averaging these waves. It should be noted that the waves propagating from adjacent cells should not interfere. This condition limits the time interval over which the waves are allowed to propagate, resulting in a CFL-like condition for the time step.

After its introduction Godunov's scheme has been improved in a number of ways. The main idea behind the scheme consisting of the three steps is still the same. Godunov-type methods are therefore always of the form as depicted in the flowchart of Figure 3.4.

Improvements of the method mainly focus on the first and second step. When the exact solution to the Riemann problem is employed in the second step, the numerical accuracy of the method is determined by the first and last step. Using a piecewise reconstruction of the cell-centred data in the first step means that the method is first-order accurate in space. Of course, there are other ways of extrapolating the cell-centred data to the cell interfaces. To achieve higher order accuracy, the piecewise constant reconstruction in the first step is replaced by a higher order approximation, for example, by using a piecewise linear, instead of piecewise constant reconstruction. On the other hand, in the second step it is worth considering an approximate solution to the Riemann problem instead of using a time consuming exact solution. This is especially worth considering since some of the effort of computing an

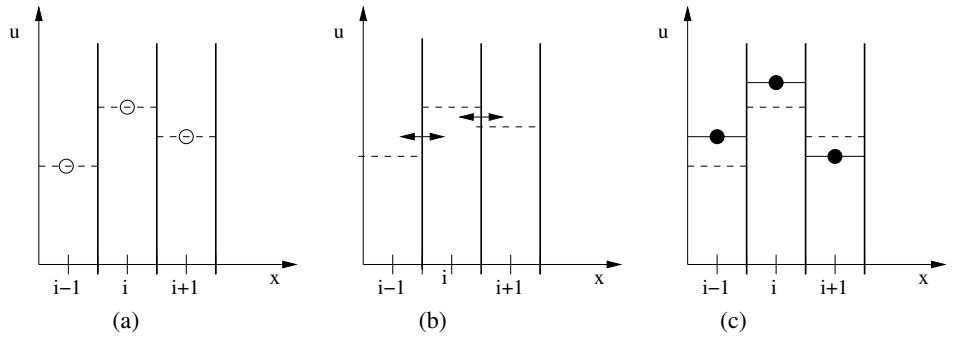

Figure 3.3: *Illustration of the three basic steps of Godunov's method.*

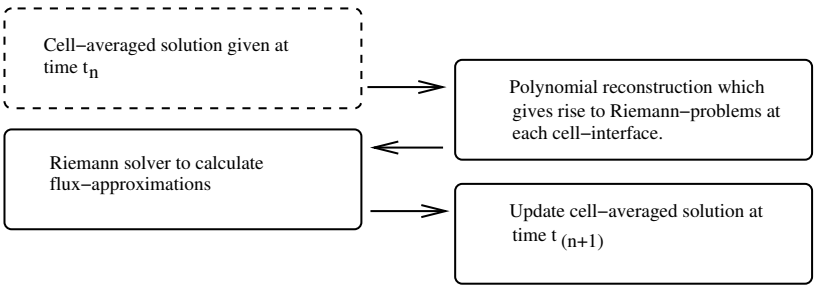

Figure 3.4: *Flowchart of the three steps in Godunov-type methods.*

exact solution is undone in the last step, when the exact solutions from all cell-interface fluxes are averaged over the cell. The former improvement is given in more detail in the next section, the latter subsequently in Section 3.3.4.

Note that the reconstruction in step one and the flux calculation in step two use dimensional splitting. Both steps are performed in each direction x,y or z separately. It implies that both steps 1 and 2 are essentially one-dimensional. In the following, the calculations are described in x-direction, with mesh width Δx and indices i.

3.3.3 Improving step 1: higher order reconstruction

An obvious way of improving the order of accuracy of Godunov's method is using higher order polynomial reconstruction instead of piecewise linear data. A possible way of doing this is displayed in Figure 3.5. In this case $\boldsymbol{\phi}^-$ and $\boldsymbol{\phi}^+$ of the Riemann problem at interface of $i+1/2$, are determined by linearly interpolating $\boldsymbol{\phi}_{i-1}$ and $\boldsymbol{\phi}_i$ for $\boldsymbol{\phi}^-_{i+1/2}$, and $\boldsymbol{\phi}_{i+1}$ together with $\boldsymbol{\phi}_{i+2}$ for $\boldsymbol{\phi}^+_{i+1/2}$. Note that this higher order reconstruction defines another Riemann problem than the one resulting from the piecewise constant reconstruction.

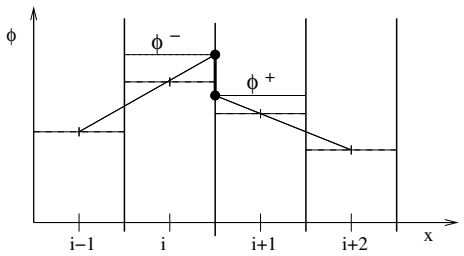

Figure 3.5: *Linear one-sided extrapolation of the interface values for* $\kappa = -1$ *in Equations (3.15).*

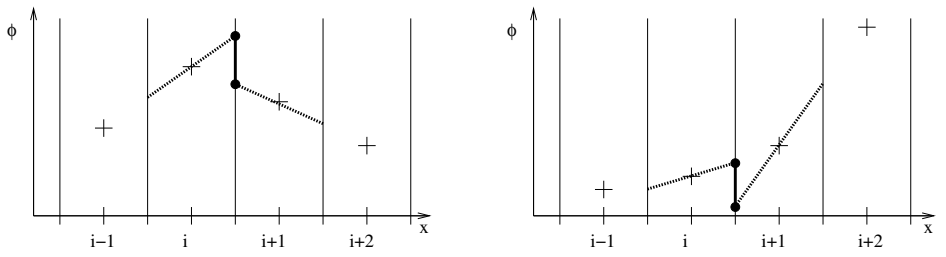

Figure 3.6: *Linear one-sided extrapolation at extremum (left) and steep gradient (right).*

Unfortunately there is a major drawback in replacing the piecewise constant data by higher order reconstructions. Around discontinuities these higher order reconstructions produce spurious oscillations, causing problems for example in the density at the free surface. This effect is similar to the problems that occur in central differencing schemes. Ironically, these are just the problems that upwind schemes intend to avoid. The numerical oscillations near discontinuities are however avoided when slope limiters are applied.

Theoretically the oscillation-preventing properties of slope limiters are quite hard to prove. Intuitively however, these properties are lucid enough when considering the following three observations.

To begin with, a general form of the interpolated values of $\boldsymbol{\phi}^-$ and $\boldsymbol{\phi}^+$ is given by

$$\boldsymbol{\phi}^-_{i+1/2} = \boldsymbol{\phi}_i + \tfrac{1}{4}[(1-\kappa)(\boldsymbol{\phi}_i - \boldsymbol{\phi}_{i-1}) + (1+\kappa)(\boldsymbol{\phi}_{i+1} - \boldsymbol{\phi}_i)] \tag{3.15a}$$

$$\boldsymbol{\phi}^+_{i+1/2} = \boldsymbol{\phi}_{i+1} - \tfrac{1}{4}[(1-\kappa)(\boldsymbol{\phi}_{i+2} - \boldsymbol{\phi}_{i+1}) + (1+\kappa)(\boldsymbol{\phi}_{i+1} - \boldsymbol{\phi}_i)]. \tag{3.15b}$$

The parameter κ defines the order and the type of the interpolant. Two

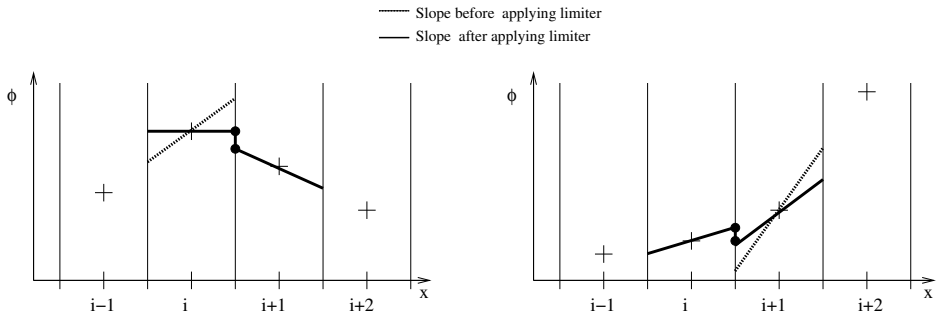

Figure 3.7: *Slope limiting at extremum (left) and for steep gradient (right).*

typical values are $\kappa = -1$ and $\kappa = \frac{1}{2}$. For $\kappa = -1$ the system gives the second-order, one-sided interpolation that is displayed in Figure 3.5. The interpolation is third order accurate when $\kappa = \frac{1}{2}$.

The second observation is that these reconstructions may cause over- and undershoots. Two typical examples are given in Figure 3.6. On the left the higher order reconstruction fails to recognise the maximum, resulting in an overshoot. On the right the steep gradient results in an undershoot. In both cases this results in a Riemann problem at $i+1/2$ that is not representative of the local conditions.

Third, observe that these over- and undershoots are prevented by adjusting the slope with a limiter function $0 \leq \Psi \leq 2$. Considering each scalar entry ϕ of $\boldsymbol{\phi}$ separately, the Equations (3.15) are therefore modified to

$$\phi^-_{i+1/2} = \phi_i + \tfrac{1}{4}[(1-\kappa)\Psi^+(\phi_i - \phi_{i-1}) + (1+\kappa)\Psi^-(\phi_{i+1} - \phi_i)] \tag{3.16a}$$

$$\phi^+_{i+1/2} = \phi_{i+1} - \tfrac{1}{4}[(1-\kappa)\Psi^-(\phi_{i+2} - \phi_{i+1}) + (1+\kappa)\Psi^+(\phi_{i+1} - \phi_i)]. \tag{3.16b}$$

Here Ψ^- and Ψ^+ are the left and right values for the limiter, defined as

$$\Psi^- \equiv \Psi(r^-_{i+1/2}), \qquad r^-_{i+1/2} = \frac{\phi_i - \phi_{i-1}}{\phi_{i+1} - \phi_i} \tag{3.17a}$$

$$\Psi^+ \equiv \Psi(r^+_{i+1/2}), \qquad r^+_{i+1/2} = \frac{\phi_{i+2} - \phi_{i+1}}{\phi_{i+1} - \phi_i}. \tag{3.17b}$$

From these definitions it is seen that, by introducing limiters, $\phi^-_{i+1/2}$ and $\phi^+_{i+1/2}$ become non-linear functions of ϕ_i, ϕ_{i-1}, ϕ_{i+1} and ϕ_{i+2}.

Within this structure, there are several effective limiters. Some limiters that are often applied are given in Table 3.1 and the effect of applying the minmod limiter is displayed in Figure 3.7. Here it is shown that after applying the limiter the Riemann problems are clearly more representative

Name	Limiter function
Van Leer limiter	$\Psi(r) = (r + \lvert r\rvert)(1 + r)^{-1}$
Superbee limiter	$\Psi(r) = \max(0, \min(2r, 1), \min(r, 2))$
Minmod limiter	$\Psi(r) = \max(0, \min(1, r))$
k-limiter	$\Psi(r) = \max(0, \min(1, kr))$
MUSCL-limiter	$\Psi(r) = \max(0, \min(2, 2r, \frac{1}{2} + \frac{1}{2}r))$

Table 3.1: *Definitions of slope limiters.*

of the local conditions, thus intuitively explaining the positive numerical effect of the limiter.

The reconstruction method described above is known as the MUSCL scheme, and was first introduced by Van Leer in 1979. MUSCL is an acronym for Monotone Upstream-centred Schemes for Conservation Laws. The theory that ensures that the MUSCL scheme is free of spurious oscillation centres around the *monotonicity* of the scheme. In the paragraphs below some details are presented.

Recall that a scheme is called monotone when for all discrete solutions v_i, u_i it holds that $v_i^n \geq u_i^n$ implies $v_i^{n+1} \geq u_i^{n+1}$ for all i,. Hence, monotone schemes mimic a basic property of exact solutions of partial differential equations. For non-linear conservation laws the monotonicity ensures that no new extrema are created and thus spurious oscillations do not appear.

Godunov was the first to show that higher order accuracy and monotonicity are contradictory requirements for linear schemes. He proved that there are no monotone, linear schemes that are second, or higher order accurate [26, theorem 13.5.4]. His own – first order – method is monotone and thus free of spurious oscillations. Circumventing Godunov's theorem is possible by using *non-linear* schemes. As observed above, applying limiters results in such a non-linear reconstruction scheme. This guarantees that using limiters might work; of course not that it will.

The theory that limiters yield monotonic schemes is developed from the total variation diminishing (TVD) condition. For a discrete function $\phi = \{\phi_i\}$ the total variation $TV(\phi)$ is defined as

$$TV(\phi) = \sum_i |\phi_{i+1} - \phi_i|. \tag{3.18}$$

Using linear reconstruction, the total variation is thus the sum of all interface jumps. In a TVD scheme the total variation of the solution at the new time step t_{n+1} is non-increasing with respect to the previous time step t_n:

$$TV(\phi^{n+1}) \leq TV(\phi^n). \tag{3.19}$$

For one-dimensional, linear conservation laws it is proven that the use of limiters guarantees that the TVD restriction holds and subsequently that the TVD condition ensures monotonicity. Unfortunately the theoretical basis is sound for these cases only. Experimental results show however, that satisfying the TVD condition by the adoption of limiters, is a good guideline to achieve monotonic schemes for more general conservation laws.

3.3.4 Improving step 2: approximate Riemann solvers

The exact Riemann solver used in the second step of Godunov's method, can be replaced by several approximate Riemann solvers. All of them approximate the exact flux function at the cell interfaces. One that is often used is

due to Roe, who first presented his method in 1981 [23]. The idea behind the method is to consider the exact solutions of an approximate problem, in contrast to approximating the exact solution. This is best illustrated when considering the one-dimensional homogeneous Euler equations in diverential form

$$\boldsymbol{\phi}_t + \boldsymbol{F}(\boldsymbol{\phi})_x = 0, \tag{3.20}$$

for a Riemann problem with initial states

$$\boldsymbol{\phi} = \begin{cases} \boldsymbol{\phi}^- & \text{for} \quad x < 0 \\ \boldsymbol{\phi}^+ & \text{for} \quad x > 0. \end{cases} \tag{3.21}$$

Equation (3.20) can also be written in the quasi-linear form

$$\boldsymbol{\phi}_t + \boldsymbol{A}(\boldsymbol{\phi})\boldsymbol{\phi}_x = 0, \qquad \text{with} \qquad \boldsymbol{A}(\boldsymbol{\phi}) = \boldsymbol{F}_{\boldsymbol{\phi}}. \tag{3.22}$$

Roe's idea is now to replace the quasi-linear, hyperbolic Equation (3.22) by an approximate *linear*, hyperbolic equation

$$\boldsymbol{\phi}_t + \widehat{\boldsymbol{A}}\ \boldsymbol{\phi}_x = 0, \tag{3.23}$$

where $\widehat{\boldsymbol{A}} = \widehat{\boldsymbol{A}}(\boldsymbol{\phi}^-, \boldsymbol{\phi}^+)$ has constant coefficients. $\widehat{\boldsymbol{A}}$ is chosen such that it is representative for the local conditions. To achieve this local resemblance of the Jacobian $\widehat{\boldsymbol{A}}$ and flux function $\widehat{\boldsymbol{F}}$ to $\boldsymbol{A}(\boldsymbol{\phi})$ and $\boldsymbol{F}(\boldsymbol{\phi})$, Roe gives the following restrictions:

1. The hyperbolicity of the original system (3.22) should be maintained, that is $\widehat{\boldsymbol{A}}$ should have real eigenvalues and linearly independent eigenvectors.

2. The approximative system (3.23) should be consistent with the original system (3.22). This means that $\widehat{\boldsymbol{A}}(\boldsymbol{\phi}, \boldsymbol{\phi}) = \boldsymbol{A}(\boldsymbol{\phi})$ and $\widehat{\boldsymbol{F}}(\boldsymbol{\phi}, \boldsymbol{\phi}) = \boldsymbol{F}(\boldsymbol{\phi})$ should both hold.

3. The conservation properties of the original system (3.22) should be mirrored in the approximate system (3.23), especially at discontinuities. Across discontinuities the local form of the conservation laws is described by the Rankine-Hugoniot relation. For the original quasi-linear system the relation is $\boldsymbol{F}(\boldsymbol{\phi}^+) - \boldsymbol{F}(\boldsymbol{\phi}^-) = S(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-)$, with S a matrix with the speeds of the jump discontinuities at the diagonal. For the approximate linear system the Rankine-Hugoniot relation simplifies to $\widehat{\boldsymbol{F}}(\boldsymbol{\phi}^+) - \widehat{\boldsymbol{F}}(\boldsymbol{\phi}^-) = \widehat{\boldsymbol{A}}(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-)$. To mirror the conservative properties along discontinuities of the original system in the the approximate system, $\widehat{\boldsymbol{A}}$ should be such that $\widehat{\boldsymbol{A}}(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-) = \boldsymbol{F}(\boldsymbol{\phi}^+) - \boldsymbol{F}(\boldsymbol{\phi}^-)$ holds.

For the moment assume that a matrix $\widehat{\boldsymbol{A}}$ that satisfies the three restrictions, is given. Once $\widehat{\boldsymbol{A}}$ is constructed, the flux $\widehat{\boldsymbol{F}}$ is constructed by applying theory on Riemann problems for linear hyperbolic systems with constant coefficients. To this end, recall Section 3.3.1 above.

Since the system is hyperbolic by restriction number 1, $\widehat{\boldsymbol{A}}$ is diagonalised to $\boldsymbol{\Lambda} = \boldsymbol{K}^{-1}\widehat{\boldsymbol{A}}\boldsymbol{K}$. With $\boldsymbol{\Lambda} = \mathrm{diag}(\lambda_1, \ldots, \lambda_m)$ the matrix of the ordered eigenvalues and the columns of $\boldsymbol{K}$ the corresponding right eigenvectors. Since the flux through $x = 0$ is desired, the solution $\boldsymbol{\phi}(x/t)$ given by Equation (3.14) is manipulated to form the two equations

$$\boldsymbol{\phi}(0) = \boldsymbol{\phi}^- + \sum_{\lambda_i \leq 0} (\beta_i - \alpha_i)\boldsymbol{K}_i \qquad \text{and} \tag{3.24a}$$

$$\boldsymbol{\phi}(0) = \boldsymbol{\phi}^+ - \sum_{\lambda_i \geq 0} (\beta_i - \alpha_i)\boldsymbol{K}_i. \tag{3.24b}$$

Using the average of both equations, a flux $\widehat{\boldsymbol{F}} = \widehat{\boldsymbol{F}}(\boldsymbol{\phi}(0)) = \widehat{\boldsymbol{A}}\boldsymbol{\phi}(0)$ is derived

$$\begin{aligned}
\widehat{\boldsymbol{F}} &= \tfrac{1}{2}(\widehat{\boldsymbol{A}}\boldsymbol{\phi}^- + \widehat{\boldsymbol{A}}\boldsymbol{\phi}^+) - \tfrac{1}{2}\sum_{i=1}^{m}(\beta_i - \alpha_i)|\lambda_i|\boldsymbol{K}_i \\
&= \tfrac{1}{2}(\widehat{\boldsymbol{A}}\boldsymbol{\phi}^- + \widehat{\boldsymbol{A}}\boldsymbol{\phi}^+) - \tfrac{1}{2}|\widehat{\boldsymbol{A}}|\sum_{i=1}^{m}(\beta_i - \alpha_i)\boldsymbol{K}_i \\
&= \tfrac{1}{2}(\widehat{\boldsymbol{A}}\boldsymbol{\phi}^- + \widehat{\boldsymbol{A}}\boldsymbol{\phi}^+) - \tfrac{1}{2}|\widehat{\boldsymbol{A}}|(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-).
\end{aligned} \tag{3.25}$$

Here $|\widehat{\boldsymbol{A}}| = \boldsymbol{K}|\boldsymbol{\Lambda}|\boldsymbol{K}^{-1}$, with $|\boldsymbol{\Lambda}|$ the matrix with the absolute value of the eigenvalues. Equation (3.25) shows that $\widehat{\boldsymbol{F}}$ is a function of $\boldsymbol{\phi}^-$ and $\boldsymbol{\phi}^+$.

We now have the exact flux of the approximate system (3.23). However, this flux is not consistent with the flux of the original system. For example

$$\widehat{\boldsymbol{F}}(\boldsymbol{\phi}^-, \boldsymbol{\phi}^-) = \widehat{\boldsymbol{F}}(\boldsymbol{\phi}^-) \neq \boldsymbol{F}(\boldsymbol{\phi}^-), \tag{3.26}$$

and restriction 2 is thus not satisfied. This is adjusted when taking $\boldsymbol{F}(\boldsymbol{\phi}^+)$ and $\boldsymbol{F}(\boldsymbol{\phi}^-)$ instead of $\widehat{\boldsymbol{A}}\boldsymbol{\phi}^-$ and $\widehat{\boldsymbol{A}}\boldsymbol{\phi}^+$ respectively. This results in Roe's approximate flux function:

$$\widehat{\boldsymbol{F}} = \tfrac{1}{2}\big(\boldsymbol{F}(\boldsymbol{\phi}^+) + \boldsymbol{F}(\boldsymbol{\phi}^-) - |\widehat{\boldsymbol{A}}|(\boldsymbol{\phi}^+ - \boldsymbol{\phi}^-)\big). \tag{3.27}$$

Now that we have derived Roe's approximate flux function – given in Equation (3.27) – it is left to show how to construct an $\widehat{\boldsymbol{A}}$ that satisfies the three restrictions. Unfortunately obvious candidates for $\widehat{\boldsymbol{A}}$ such as $\widehat{\boldsymbol{A}} = \tfrac{1}{2}(\boldsymbol{A}(\boldsymbol{\phi}^-) + \boldsymbol{A}(\boldsymbol{\phi}^+))$ do not satisfy the third restriction. It is not even clear that such an $\widehat{\boldsymbol{A}}$ actually exists. In 1984 Roe and Pike [24] showed that there

is an unique $\widehat{\boldsymbol{A}}$ satisfying the three restraints. This $\widehat{\boldsymbol{A}}$ is constructed when evaluating the original Jacobian $\boldsymbol{A}$ at the so called Roe's average values. The average values are given by

$$\widehat{\boldsymbol{\phi}} \equiv (\widehat{\rho}\ ,\ \widehat{\rho}\,\widehat{\boldsymbol{u}}\ ,\ \widehat{p}/\beta)^T. \tag{3.28}$$

Choosing

$$\widehat{\rho} \equiv \sqrt{\rho^+\ \rho^-} \tag{3.29a}$$

$$\widehat{\boldsymbol{u}} \equiv \frac{\boldsymbol{u}^+\sqrt{\rho^+} + \boldsymbol{u}^-\sqrt{\rho^-}}{\sqrt{\rho^+} + \sqrt{\rho^-}} \tag{3.29b}$$

$$\widehat{p} \equiv \tfrac{1}{2}(p^- + p^+). \tag{3.29c}$$

leads to, $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\widehat{\boldsymbol{\phi}})$.

In summary, Roe's approximate flux function for the hyperbolic homogeneous Euler equations (3.22) is given by Equation (3.27). Roe's approximate flux function is based on the solution of the approximate Equation (3.23) together with a consistency restrained. Roe gives an approximate matrix $\widehat{\boldsymbol{A}} = \boldsymbol{A}(\widehat{\boldsymbol{\phi}})$ that it representative of the local properties of $\boldsymbol{A}(\boldsymbol{\phi})$.

A few final remarks are necessary. The numerical flux that has just been derived in Equation (3.27) is the approximate flux of the homogeneous one-dimensional Euler equations, while we actually need a numerical inviscid flux for the non-homogeneous three dimensional Euler equations. First, the dimensional problem is solved by splitting the flux: the numerical inviscid fluxes are approached in x, y and z direction separately as one-dimensional fluxes. This is for example described by Toro in [26, chapter 16]. Second, the same sort of splitting is applied to handle the non-homogeneous term. The calculation of the inviscid flux is the same as described above, see for example [26, chapter 15].

3.4 Numerical model

Successively, the mathematical model of AMAZON-SC and the theory of Godunov-type methods have been presented. In consequence, all material is available to present the numerical model of AMAZON-SC. The discretisation of the equations is presented in general three dimensional terms. The discretisation does not take into account moving bodies and cut cells, but some attention to these matters is given subsequently in Section 3.4.4. The closing section of this chapter deals with the discretisation of the free surface.

3.4.1 Grid and geometry definition

To solve the flow equations numerically, the computational domain is covered with a fixed Cartesian grid. The variables have a collocated arrangement,

which means that the velocities and the pressure are defined in the cell centres. The control volumes are thus the Cartesian cells.

The body geometry is piecewise linear and cuts through the fixed rectangular grid. This results in some cells that are partly filled with a (moving) geometry. These cells are called cut cells, see Section 3.4.4.

3.4.2 Discretisation of the equations

To discretise the governing equations, recall that these are given by Equation (3.7)

$$\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega \boldsymbol{\phi}\ \mathrm{d}\Omega + \boldsymbol{I}_0\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \boldsymbol{\phi}\ \mathrm{d}\Omega \quad = \quad -\oint_\Gamma \boldsymbol{F}\cdot\boldsymbol{n}\ \mathrm{d}\Gamma + \int_\Omega \boldsymbol{G}\boldsymbol{\phi}\ \mathrm{d}\Omega.$$

The discretisation is given for one cell with volume Ω. The cell is considered to have N cell faces. For un-cut cells, $N = 6$ in three dimensions and $N = 4$ in two dimensions. Cut cells may have more cell faces. The flux through each cell face $j = 1, \ldots, N$ is given by $\boldsymbol{F}_j$. Evaluating the integral for the cell-centred data with the midpoint rule gives

$$\Omega\frac{\mathrm{d}}{\mathrm{d}\tau}\boldsymbol{\phi} + \Omega\boldsymbol{I}_0\frac{\mathrm{d}}{\mathrm{d}t}\boldsymbol{\phi} \quad = \quad R(\boldsymbol{\phi}), \tag{3.30}$$

with

$$R(\boldsymbol{\phi}) \quad \equiv \quad -\sum_{j=1}^{N}\boldsymbol{F}_j + \Omega\boldsymbol{G}\boldsymbol{\phi}. \tag{3.31}$$

The temporal discretisation of the physical time $t = n\Delta t$ is done by implicit Euler, which is first order accurate. In pseudo time $\tau = k\Delta\tau$ the same discretisation is implemented. This results in

$$\Omega\frac{\boldsymbol{\phi}^{n+1,k+1} - \boldsymbol{\phi}^{n+1,k}}{\Delta\tau} + \Omega\boldsymbol{I}_0\frac{\boldsymbol{\phi}^{n+1,k+1} - \boldsymbol{\phi}^{n}}{\Delta t} \quad = \quad R(\boldsymbol{\phi}^{n+1,k+1}). \tag{3.32}$$

Using the dual-time step procedure to ensure a time accurate solution, we need to have $|(\boldsymbol{\phi}^{n+1,k+1} - \boldsymbol{\phi}^{n+1,k})/\boldsymbol{\phi}^{n+1,k+1}| \leq \varepsilon$ for every pseudo-time iteration. In Amazon-sc the error bound usually takes values $10^{-4} \leq \varepsilon \leq 10^{-6}$. The restriction to ensure a time accurate solution, makes it more natural to rewrite the equation with the variable $\Delta\boldsymbol{\phi}^{k+1} = \boldsymbol{\phi}^{n+1,k+1} - \boldsymbol{\phi}^{n+1,k}$, using the superscript $\cdot^{\ k}$ as shorthand for $\cdot^{\ n+1,k}$.

To give an explicit equation for $\Delta\boldsymbol{\phi}^{k+1}$, the implicitly defined $R(\boldsymbol{\phi}^{n+1,k+1})$ needs to be linearised. The Taylor expansion around the previous pseudo-time step is given by

$$\begin{aligned} R(\boldsymbol{\phi}^{k+1}) \quad &= \quad R^k + \left(\frac{\partial R}{\partial \tau}\right)^k \Delta\tau + \mathcal{O}(\Delta\tau^2) \\ &= \quad R^k + \left(\frac{\partial R}{\partial \boldsymbol{\phi}}\right)^k \Delta\boldsymbol{\phi}^{k+1} + \mathcal{O}(\Delta\tau). \end{aligned} \tag{3.33}$$

Equation (3.32) is now rewritten as

$$\Omega(\frac{1}{\Delta\tau}\boldsymbol{I}+\frac{1}{\Delta t}\boldsymbol{I_0})\Delta\boldsymbol{\phi}^{k+1}-\left(\frac{\partial R}{\partial\boldsymbol{\phi}}\right)^k\Delta\boldsymbol{\phi}^{k+1}=\Omega\frac{1}{\Delta t}\boldsymbol{I_0}(\boldsymbol{\phi}^k-\boldsymbol{\phi}^n)+R^k. \quad (3.34)$$

The left hand side contains all unknown, the right all the known terms. It now remains to show what the two terms in the linearisation of $R(\boldsymbol{\phi}^{k+1})$ look like.

First, R^k is given straightforward by the values at the previous time step

$$R^k \equiv -\sum_{j=1}^{N}\widehat{\boldsymbol{F}}_j^k+\Omega\boldsymbol{G}\boldsymbol{\phi}^k, \quad (3.35)$$

with the numerical inviscid flux $\widehat{\boldsymbol{F}}$ given by Roe's approximate flux function given by Equation (3.27)

$$\widehat{\boldsymbol{F}}(\Delta\boldsymbol{\phi}^-,\Delta\boldsymbol{\phi}^+)=\tfrac{1}{2}\big(\boldsymbol{F}(\Delta\boldsymbol{\phi}^+)+\boldsymbol{F}(\Delta\boldsymbol{\phi}^-)-|\widehat{\boldsymbol{A}}|(\Delta\boldsymbol{\phi}^+-\Delta\boldsymbol{\phi}^-)\big).$$

Second, the computation of the Jacobians, especially the inviscid term, is a little more demanding. For the inviscid term, recall the theory on Godunov-type method, described in Section 3.3. The theory is applicable since we have a hyperbolic conservative flux term.

Using Roe's numerical flux again, Equation (3.27), we have

$$\begin{aligned}\left(\frac{\partial\boldsymbol{F}_j}{\partial\boldsymbol{\phi}}\right)^k\Delta\boldsymbol{\phi}^{k+1} &\widehat{=} \left(\frac{\partial\widehat{\boldsymbol{F}}_j}{\partial\boldsymbol{\phi}^-}\right)^k(\Delta\boldsymbol{\phi}^-)^{k+1}\\ &\quad+\left(\frac{\partial\widehat{\boldsymbol{F}}_j}{\partial\boldsymbol{\phi}^+}\right)^k(\Delta\boldsymbol{\phi}^+)^{k+1}. \quad (3.36)\end{aligned}$$

The Jacobians of the numerical flux are approximated by substituting Equation (3.27) and neglecting derivatives of $|\widehat{\boldsymbol{A}}|$ with respect to $\boldsymbol{\phi}^-$ and $\boldsymbol{\phi}^+$. The resulting expressions are written as

$$\frac{\partial\widehat{\boldsymbol{F}}_j}{\partial\boldsymbol{\phi}^-} \widehat{=} \tfrac{1}{2}(\boldsymbol{A}(\Delta\boldsymbol{\phi}^-)+|\widehat{\boldsymbol{A}}|) \equiv \boldsymbol{B}_j^- \quad (3.37a)$$

$$\frac{\partial\widehat{\boldsymbol{F}}_j}{\partial\boldsymbol{\phi}^+} \widehat{=} \tfrac{1}{2}(\boldsymbol{A}(\Delta\boldsymbol{\phi}^+)-|\widehat{\boldsymbol{A}}|) \equiv \boldsymbol{B}_j^+. \quad (3.37b)$$

This results in

$$\left(\frac{\partial\boldsymbol{F}_j}{\partial\boldsymbol{\phi}}\right)^k\Delta\boldsymbol{\phi}^{k+1} \widehat{=} (\boldsymbol{B}_j^-)^k(\Delta\boldsymbol{\phi}^-)^{k+1}+(\boldsymbol{B}_j^+)^k(\Delta\boldsymbol{\phi}^+)^{k+1}. \quad (3.38)$$

The inviscid term is now properly linearised, but still some attention has to be given to the way $\Delta\boldsymbol{\phi}^-$ and $\Delta\boldsymbol{\phi}^+$ are constructed. Remember that

the MUSCL scheme gives non-linear reconstructions, see Equations (3.16). Since we need a linear system for $\Delta\boldsymbol{\phi}^{k+1}$ these non-linear reconstructions are not used for $(\Delta\boldsymbol{\phi}^-)^{k+1}$ and $(\Delta\boldsymbol{\phi}^+)^{k+1}$. In this case the piecewise constant reconstruction based on the cell-centred data is used. The non-linear MUSCL scheme can however be applied for all reconstructions of $\Delta\boldsymbol{\phi}^-$ and $\Delta\boldsymbol{\phi}^+$ used to derive $(\boldsymbol{B}_j^-)^k$, $(\boldsymbol{B}_j^+)^k$ and $\widehat{\boldsymbol{F}}_j^k$. In the case of AMAZON-SC $\kappa = 1$, resulting in the central differencing (in x-direction)

$$\Delta\boldsymbol{\phi}^-_{i+1/2} = \Delta\boldsymbol{\phi}_i + \tfrac{1}{2}\Psi^-(\Delta\boldsymbol{\phi}_{i+1} - \Delta\boldsymbol{\phi}_i) \tag{3.39a}$$

$$\Delta\boldsymbol{\phi}^+_{i+1/2} = \Delta\boldsymbol{\phi}_{i+1} - \tfrac{1}{2}\Psi^+(\Delta\boldsymbol{\phi}_{i+1} - \Delta\boldsymbol{\phi}_i). \tag{3.39b}$$

The limiter that prevents spurious oscillations, should be able to keep a sharp interface at the free surface. This implies it is advantageous to employ a highly compressive limiter for the density and less compressive limiters for the momentum and pressure. For most simulations the k-limiter is used with $k = 1$ for the momentum components and the pressure, and $k = 2$ for the density.

We are now in a postion to substitute the given linearisation into Equation (3.34). After straightforward manipulations the resulting linear system is

$$\begin{aligned}\Omega(\frac{1}{\Delta\tau}\boldsymbol{I} + \frac{1}{\Delta t}\boldsymbol{I_0} - \boldsymbol{G})\Delta\boldsymbol{\phi}^{k+1} + \sum_{j=1}^{N}(\boldsymbol{B}^-)_j^k(\Delta\boldsymbol{\phi}^-)_j^{k+1} + (\boldsymbol{B}^+)_j^k(\Delta\boldsymbol{\phi}^+)_j^{k+1} \\ = \Omega\frac{1}{\Delta t}\boldsymbol{I_0}(\boldsymbol{\phi}^k - \boldsymbol{\phi}^n) + \Omega\boldsymbol{G}\boldsymbol{\phi}^k - \sum_{j=1}^{N}\widehat{\boldsymbol{F}}_j^k.\end{aligned} \tag{3.40}$$

After introducing the boundary conditions and the interpolants for $\Delta\boldsymbol{\phi}^-$ and $\Delta\boldsymbol{\phi}^+$, the system of equations reduces to a block-banded system. This system is solved with an approximate LU decomposition scheme.

The parameters in the discretisation have, of course, a significant role in the stability and accuracy of the method. The parameter β is of key importance for the performance of the method. Taking larger values implies a better resemblance to the original incompressible equations. On the other hand, this will make the equations stiff numerically, implying a small pseudo-time step. The parameters used in the simulation of AMAZON-SC are usually in the ranges given in Table 3.2.

3.4.3 Discretisation at the free surface

The discretisation at the free surface comes quite naturally in AMAZON-SC, since it is captured automatically as a part of the solution. Also without having to adopt boundary conditions at a moving interface, there are some

Parameter	Lower bound	Upper bound
β	200	2000
Δt	$5 \cdot 10^{-5}$	$5 \cdot 10^{-4}$
$\Delta \tau$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-2}$

Table 3.2: *Bounds for numerical parameters in Amazon-sc.*

points that need special attention – points that are typical for two-phase solvers.

First of all, the density is of interest. A consequence of using ρ to capture the interface is a jump at the free surface. To prevent numerical problems at this discontinuity, the Godunov method is applied. Still, in view of the size of the jump (a factor 1000) it is nearly impossible to maintain a sharp interface; a transition zone of several cells is inevitable to cover the jump. The choice of slope limiter plays an important role in this case. The more compressive ones, for example hyperbee, maintain a transition zone of two to three cells over a longer period of time, while the Van Leer limiter is found to be more diffusive. The limiter most generally applied in Amazon-sc is the k-limiter, choosing a compressive one ($k = 2$) for the density and a less compressive one ($k = 1$) for the velocity and pressure [22, p. 27-33].

The second central point in the discretisation of the free surface is the pressure term. The pressure term at the free surface can cause instabilities. This becomes clear when considering the stationary problem. In this case, with all time derivatives and velocities zero, the momentum equations (3.6b) reduce to

$$\nabla p = \rho \boldsymbol{g}. \tag{3.41}$$

This means that the pressure gradient should be balanced by the density term. In the direction normal to the free surface – assume this to be the y-direction – the jump in the density is also present in the pressure gradient. The pressure is thus piecewise linear, with a kink at the interface. This is depicted in Figure 3.8, where point A is the correct pressure value at the interface. However, by using the neighbouring cells to calculate the pressure at the interface, errors are introduced. For example, the incorrect value B is found, when using the values for the pressure at j and $j + 1$.

As a result, the gravity term is not exactly balanced by the normal pressure gradient, introducing errors in the discretisation. Since the gravity term is a source term in the governing equations, the error may accumulate with time. Notably, this has a serious impact on the accuracy and stability

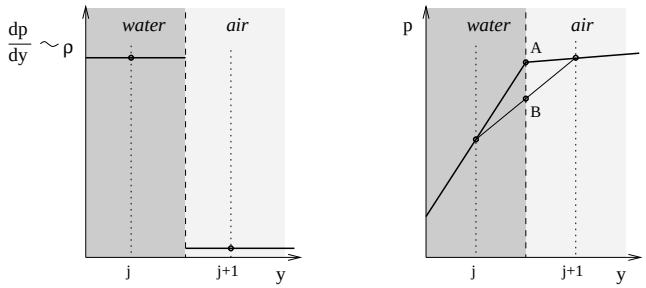

Figure 3.8: *Pressure gradient across the interface (left). Pressure across the interface (right).*

of the method. To tackle this problem, a pressure splitting procedure is adopted [22].

The normal pressure gradient (with interface normal $\boldsymbol{n}$) is split into an hydrostatic and a kinematic part as

$$\nabla p \cdot \boldsymbol{n} = (\nabla p \cdot \boldsymbol{n})_{hyd} + (\nabla p \cdot \boldsymbol{n})_{kin}. \tag{3.42}$$

The hydrostatic term is calculated directly, using the the local density values. The kinematic term is calculated with a second order scheme. When the normal and the gravity point in the y-direction, this results in

$$\frac{p^{kin}_{j+1} - p^{kin}_{j}}{\Delta y} = \frac{p_{j+1} - p_j}{\Delta y} - \tfrac{1}{2}(\rho_{j+1} + \rho_j)g. \tag{3.43}$$

The splitting makes sure that the gravity source term is exactly balanced by the normal hydrostatic pressure gradient, thus preventing the accumulation of errors.

3.4.4 Grid: cut cell method

Although AMAZON-SC uses a Cartesian cut cell method to describe arbitrary moving geometries, the discretisation of the flow equations is not given ex-

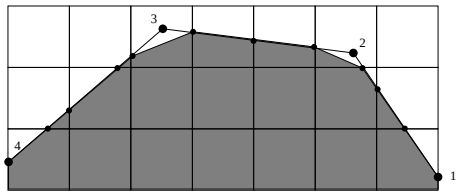

Figure 3.9: *Cut cells in two dimensions for a solid boundary.*

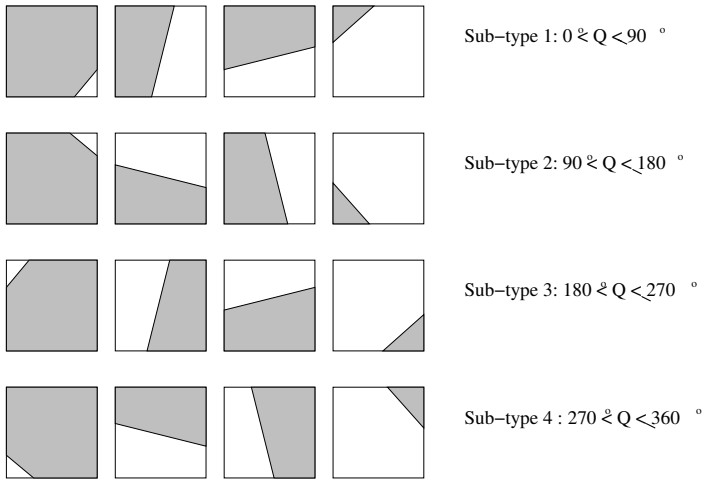

Figure 3.10: *Four sub-types of two dimensional cut cells, see [4, Fig. 4].*

plicitly for cut cells. Here, the general idea of the cut cell method is presented and furthermore, special attention is given to small cut cells and (moving) cut cells in three dimensions. More details of AMAZON-SC 's cut cell method are given in [3, 4, 30].

The Cartesian cut cell method uses a stationary Cartesian background mesh. To cope with complex geometries, solid regions are cut out of the mesh, with their boundaries represented by different types of cut cells. These odd cells are singled out for special treatment and the remainder of the flow cells are treated in a straightforward manner. Moving boundaries are accommodated by recomputing the intersections of mesh cells and boundary locally. In two dimensions, the boundary of the geometry is represented by a piecewise linear interpolation between data points. The set of data points is ordered in an anti-clockwise direction, such that the solid body is alway to the left of the boundary, see Figure 3.9 for an example. Initially all cells

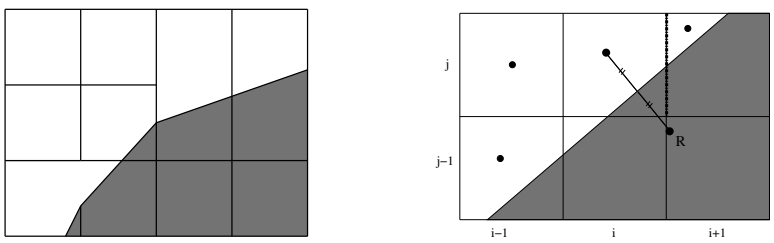

Figure 3.11: *Cell merging procedure (left). Reconstruction in cut cells (right).*

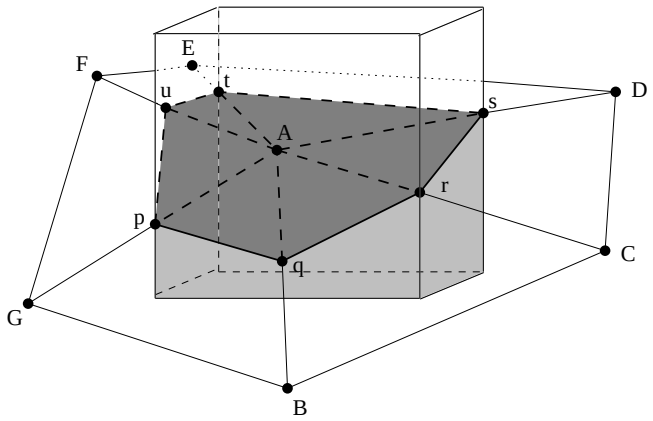

Figure 3.12: *Exact three dimensional cut cell, with the cut boundary formed by the planes Apq, Aqr, Ars, Ast, Atu and Aup.*

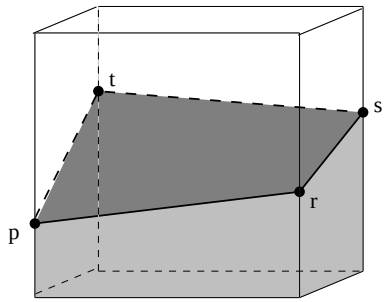

Figure 3.13: *Approximate three dimensional cut cell, with the cut boundary formed by the non-planar quadrilateral prst.*

are flagged as either flow or solid. To determine which cells are cut cells the intersections of the boundary with the grid lines are calculated. This also determines the type of the cut cell. Depending on the four possible directions – corresponding to the quadrants – for the slope of a cut-boundary, the cut cells are categorised into four types. Each category has four subtypes, depending on the number of faces that are (partly) solid. The sixteen subtypes are depicted in Figure 3.10.

The discretisation given in Section 3.4.2 should be slightly adjusted for cut cells. Given that a cut cell has several fluid faces and a solid face, the reconstruction technique applied to calculate the interface fluxes in fluid cells (see Section 3.3.3) can not automatically be applied to cut cells. At cut cells, solid boundary conditions must be incorporated into the reconstruction. The procedure, for the sake of simplicity given for the x-direction only, is as follows.

First, the reconstruction of the fluid part, $\boldsymbol{\phi}^f$, is calculated as before; using Equations (3.16a) for the left and Equation (3.16b) for the right interface value. Second, to account for the boundary conditions, the reconstruction of the solid part $\boldsymbol{\phi}^s$ is done with the use of ghost cells in the solid's interior. Almost the same reconstruction is used, except that the values in cut or solid cells are replaced by the values of ghost cells. To make this a little more concrete, see the right of Figure 3.11 for a clarifying example. To reconstruct $(\boldsymbol{\phi}^-_{i+1/2})^s$, $\boldsymbol{\phi}_{i+1,j}$ is replaced by $\boldsymbol{\phi}_R$. In this specific case the reconstruction formula ($\kappa = 1$) changes to

$$(\phi^-_{i+1/2})^s = \phi_i + \tfrac{1}{2}\Psi^-(\phi_R - \phi_i). \tag{3.44}$$

The value $\boldsymbol{\phi}_R$ is taken such that the boundary conditions are satisfied when averaging $\boldsymbol{\phi}_R$ and $\boldsymbol{\phi}_{i,j}$. Finally, the general reconstruction based on $\boldsymbol{\phi}^f$ and $\boldsymbol{\phi}^s$ is obtained by a length-averaging technique. When A^x_* is the fraction that is open to fluid at interface $*$, we have

$$\boldsymbol{\phi}_* = A^x_* \boldsymbol{\phi}^f_* + (1 - A^x_*)\boldsymbol{\phi}^s_* \tag{3.45}$$

Looking at the two extremes $A^x = 0$ and $A^x = 1$, we see that using this equation in flow cells, $A^x = 1$, gives the reconstruction as before. Alternatively, when one of the cell interfaces is completely solid, $A^x = 0$, the equation gives the boundary conditions directly. For the values of $0 < A^x < 1$, Equation (3.45) thus incorporates the boundary conditions into the reconstruction.

The conceptual idea of the cut cell method in AMAZON-SC is hereby given. Forthwith, we elaborate of the discretisation near small cut cells and the cut cell method in three dimensions.

In practice a cut cell may become arbitrary small. This can be problematic for the numerical stability, since arbitrary small cells may demand a very small time step. Recall, that in Godunov-type methods the time step is limited by a CFL-like condition to prevent the interaction of waves propagating from adjacent cells. To overcome this problem, a cell merging technique is applied. The basic idea is to combine several neighbouring cells, such that waves are able to propagate in the combined larger cells without reducing the global time step. See the left of Figure 3.11 for an example. In the newly merged cells the flow solver has to be slightly adjusted. First the new cell-centred data is constructed from a volume-weighted average of the two former cell-centred data. When the finite volume discretisation is subsequently applied with the new cell volume, this ensures the process to be fully conservative.

In three dimensions it becomes quite complex to accurately represent a surface triangulation with cut cells. See Figure 3.12 for an exact cut cell in three dimensions. To simplify matters, the intersections of the surface triangulation with the grid cell's faces are not considered: only the intersections with the grid cell's edges are taken into account. Applying this to

the surface triangulation in Figure 3.12, the intersection points q and u are ignored, resulting in the cut cell depicted in Figure 3.13. The solid boundary is now approximated by the non-planar quadrilateral $prst$. This procedure yields a classification of cut cells in three dimensions that is similar to – but more extensive than – the classification in two dimensions shown in Figure 3.10.

3.5 Summary

The underlying scheme of AMAZON-SC is based on the solution of the incompressible Euler equations for a two fluid (air and liquid) system. This results in variable values for the density, though constant in each fluid. The fluid interface is treated as a contact discontinuity in the density field, which is captured automatically as part of the solution. The equations are given in conservative integral form, since a finite volume approach is used for the discretisation. By using artificial compressibility terms, changes in the velocity field are coupled to changes in the pressure field. The artificial compressibility terms also ensure that the mathematical model is given by a hyperbolic system.

The computational domain fully encompasses both fluid regions on a stationary background Cartesian grid. Tracking the solid boundaries of moving bodies is accomplished using a cut cell method. The data are cell-centred in both cut and uncut cells.

The temporal discretisation uses dual-time stepping to achieve a time accurate, steady state solution at every physical time step. This is necessary since the equations with artificial compressibility terms are only accurate for steady state solutions.

The numerical solution of the hyperbolic conservative system is found by a high-resolution Godunov-type method, which is an upwind finite volume technique. Information about the data at the cell interfaces is constructed using a second order accurate central interpolation of the cell-centred data. Slope limiters are applied to prevent spurious oscillation originating from over- or undershoots in the interpolated data. The fluxes are calculated by approximating the solution of the local Riemann problems with Roe's approximate Riemann solver. The discretisation results in a block diagonal system. This system is solved using an approximate LU-decomposition method.

Chapter 4

Comparison of results

In this chapter results of simulations done with AMAZON-SC and COMFLOW are compared. Initially the test cases have been performed to test and validate AMAZON-SC, see the articles [5, 19, 21, 22]. The test cases described in these articles are repeated with COMFLOW and the results are compared with those of AMAZON-SC. The test cases subsequently are wave generation in a wave tank (Section 4.1), wedge entry phenomena (Section 4.2) and the flow around wave energy devices (Section 4.3). The test results are compared with experimental and theoretical data when available

4.1 Wave generation in a tank

One of the simplest tests involving moving bodies and free surface motions is wave generation in a wave tank, where an oscillating piston paddle is used for generating waves. Due to their simplicity they provide an excellent way of testing and comparing the two flow solvers. Both numerical methods are compared to the experiments performed by Gao at Brighton University

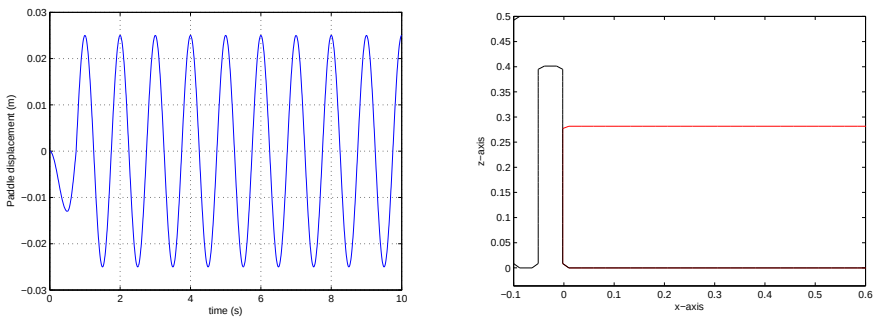

Figure 4.1: *Displacement (left) and initial position (right) of the paddle.*

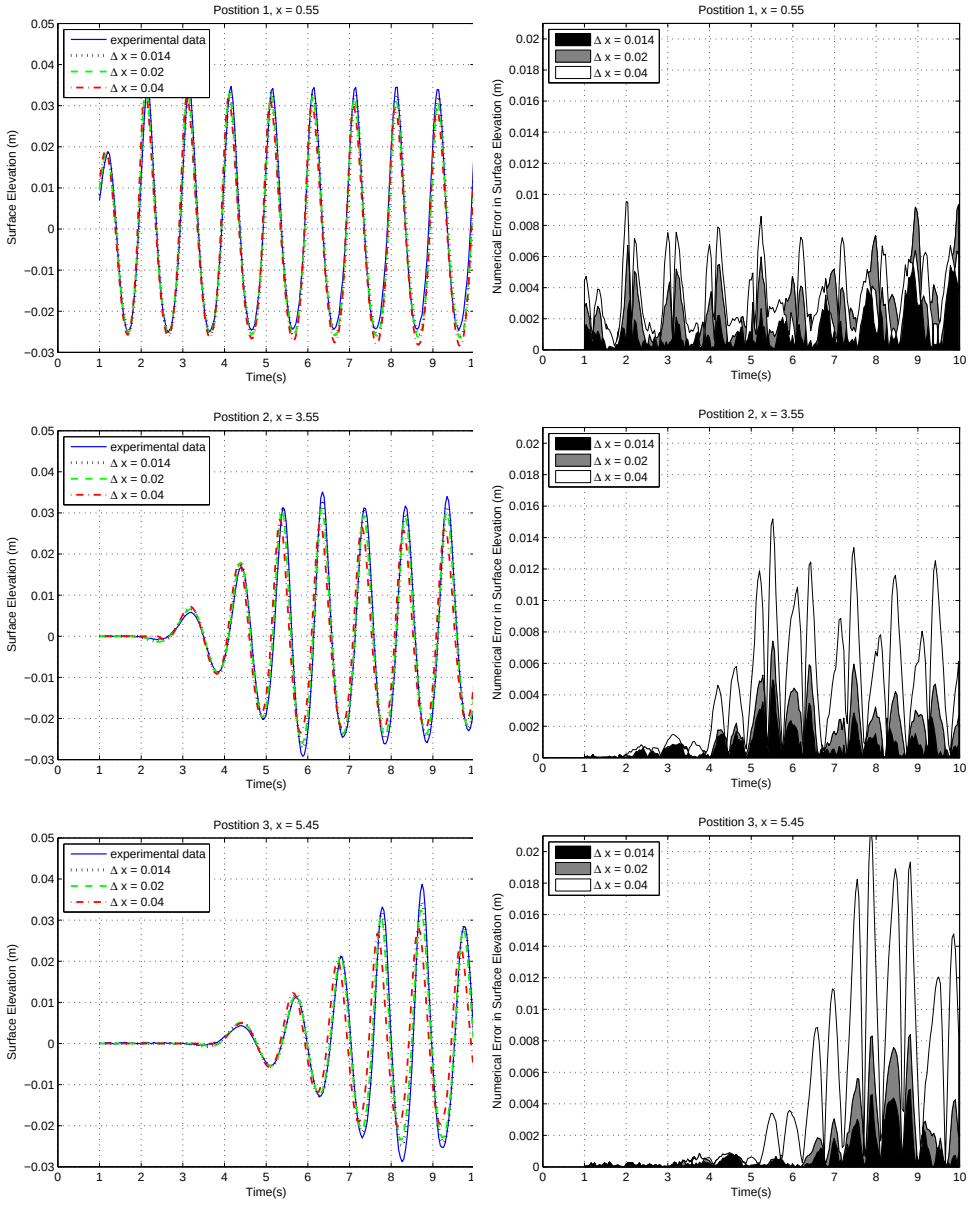

Figure 4.2: *Results of Comflow in the wave paddle test. Surface elevation (left) and numerical error in surface elevation compared to experimental results (right) at $x = 0.55$ (top), $x = 3.55$ (middle) and $x = 5.45$ (bottom).*

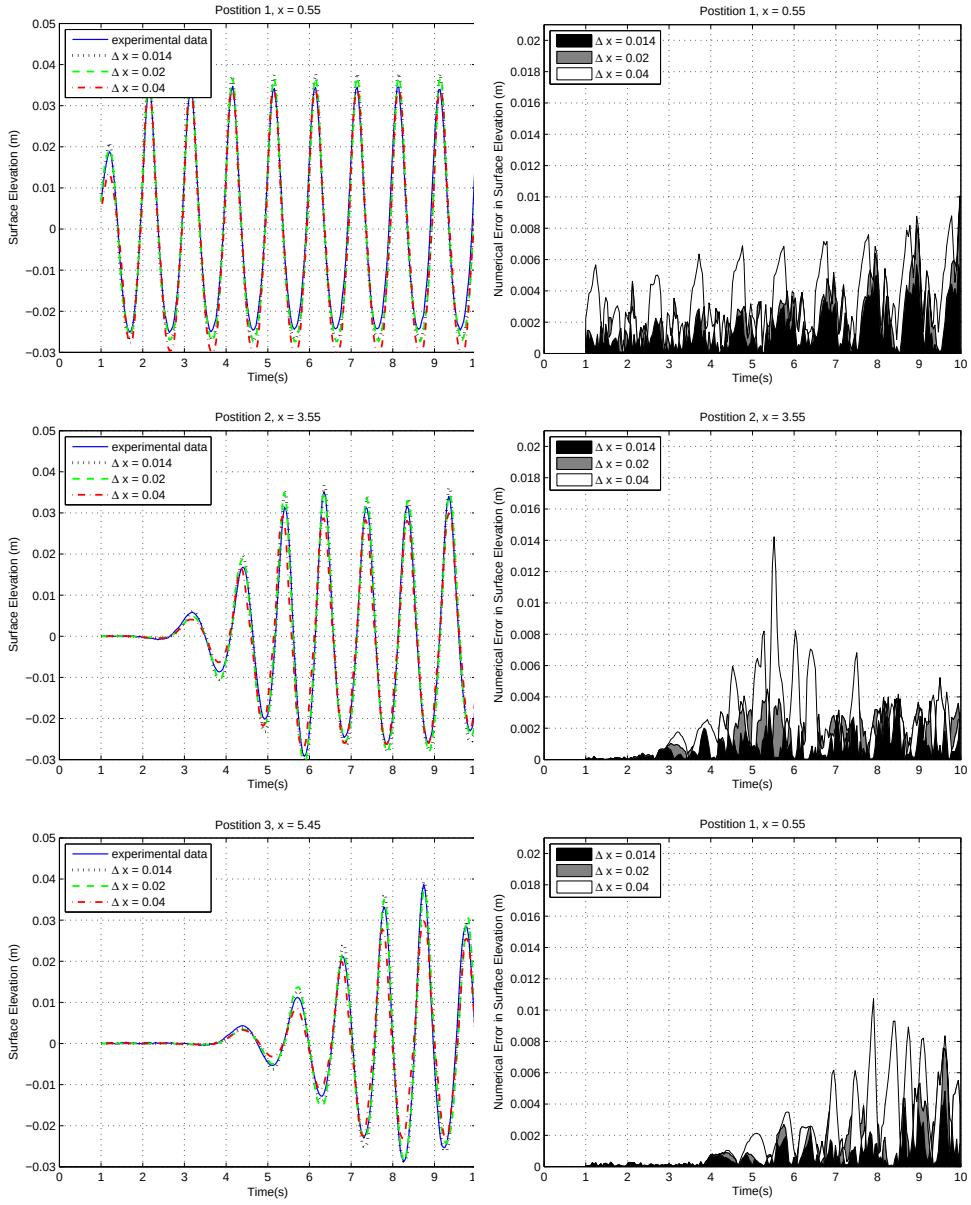

Figure 4.3: *Results of Amazon-sc in the wave paddle test. Surface elevation (left) and numerical error in surface elevation compared to experimental results (right) at $x = 0.55$ (top), $x = 3.55$ (middle) and $x = 5.45$ (bottom).*

		Δx			*convergence*	
		0.014	0.02	0.04	*rates*	
$x = 0.55$	C	4.2	4.0	6.0	*1.0*	*1.5*
	A	3.5	4.2	5.9	*1.2*	*1.7*
$x = 3.55$	C	1.9	3.4	8.5	*1.8*	*2.5*
	A	2.7	2.7	4.9	*1.0*	*1.8*
$x = 5.45$	C	2.0	3.6	10.6	*1.8*	*2.9*
	A	2.4	2.9	4.6	*1.2*	*1.6*

Table 4.1: *L^2-norms (times 10^{-2}) for the difference between numerical and experimental results of the wave paddle test, plus convergence factors.*

In Gao's experiment waves are produced in an 8.85 meters long wave tank, with a still water depth of 0.28 meters. To generate the waves, a piston paddle situated at the left boundary moves as depicted in Figure 4.1 with a period of 1.0 seconds and an amplitude of 2.5 centimeters. The development of the surface elevation is monitored during 10 seconds at three positions: $0.55, 3.55$ and 5.45 meters from the initial position of the paddle. The experiment is performed with both methods on an grid with grid size $\Delta x = \Delta y = 0.04, 0.02$ and 0.014 respectively. The time step is fixed, that is $\Delta t = 1 \cdot 10^{-4}$.

The results of both AMAZON-SC and COMFLOW resemble the experimental results. The results of COMFLOW are given in Figure 4.2, the results of AMAZON-SC in Figure 4.3. In both figures the frames in the left column show the surface elevation at the three monitor points as measured experimentally by Gao and as predicted on the different grids by the numerical methods. The frames show that both methods accurately resemble the experimental data. Unsurprisingly, the peak values of the surface elevation on the coarsest grid are either under-predicted or over-predicted, but the grid refinement gives a satisfactory agreement with the experimental data. It should be noted that COMFLOW generally underestimates the peak values, while AMAZON-SC gives an overprediction.

The convergence to the experimental data for the grid refinement is depicted in the frames of the right columns of both Figure 4.2 and 4.3, where the absolute difference between the numerical and experimental data is plotted (with a fixed range on the y-axis for all the frames). The frames show that COMFLOW is less accurate than AMAZON-SC on the coarsest grid. However, on the finer grid the two methods give similarly accurate results, implying that grid refinement gives a higher convergence rate in COMFLOW.

The effects of the grid refinement for both methods are further examined by considering the (convergence rate of the) L^2-norm. The results for

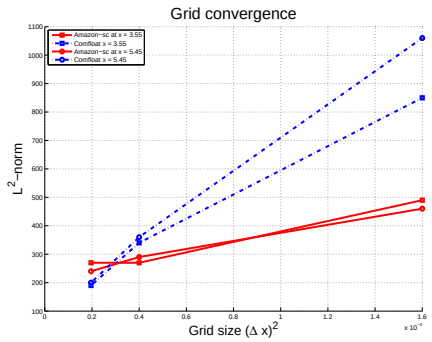

Figure 4.4: *Convergence rates for grid refinement.*

Comflow (C) and Amazon-sc (A) are given in Table 4.1. Comparing the norms on the coarsest grid, $\Delta x = 0.04$, it becomes clear that Amazon-sc is significantly more accurate. Especially further away from the paddle, the L^2-norm of Amazon-sc is almost twice as small. This changes on the finer grids, since the convergence rate of Comflow is higher than the convergence rate of Amazon-sc. For $\Delta x = 0.014$, Comflow gives slightly better results.

To compare the convergence rates of the two methods, consider Figure 4.4. Taking the averages of the covergence rates then gives a rate of 1.7 for Amazon-sc and 2.2 for Comflow when doubling the grid size (quadrupling the number of cells). Since both are roughly first order accurate, Comflow performs a little better than expected, while Amazon-sc performes a little less than expected. Having this said, it should be kept in mind that Amazon-sc generates more accurate results on coarser grids than Comflow does. A possible explanation for this good performance of Amazon-sc on the coarser grids, is that Amazon-sc uses a higher order upwind reconstruction, while Comflow employs a first order upwind method, see Section 5.3.5.

Concluding, both methods are able to handle this simple test problem pretty well. While Amazon-sc is able to handle the coarser grid a little better than Comflow, the results on the finer grids are similar.

4.2 Wedge entry phenomena

Water impact of two dimensional rigid wedge-shaped objects has been studied extensively both theoretically, experimentally and numerically. For this reason, Amazon-sc has been validated by wedge entry test cases [5, 22]. Wedge entry phenomena have also been studied extensively with Comflow [7]. To compare the numerical results of both numerical methods two wedge

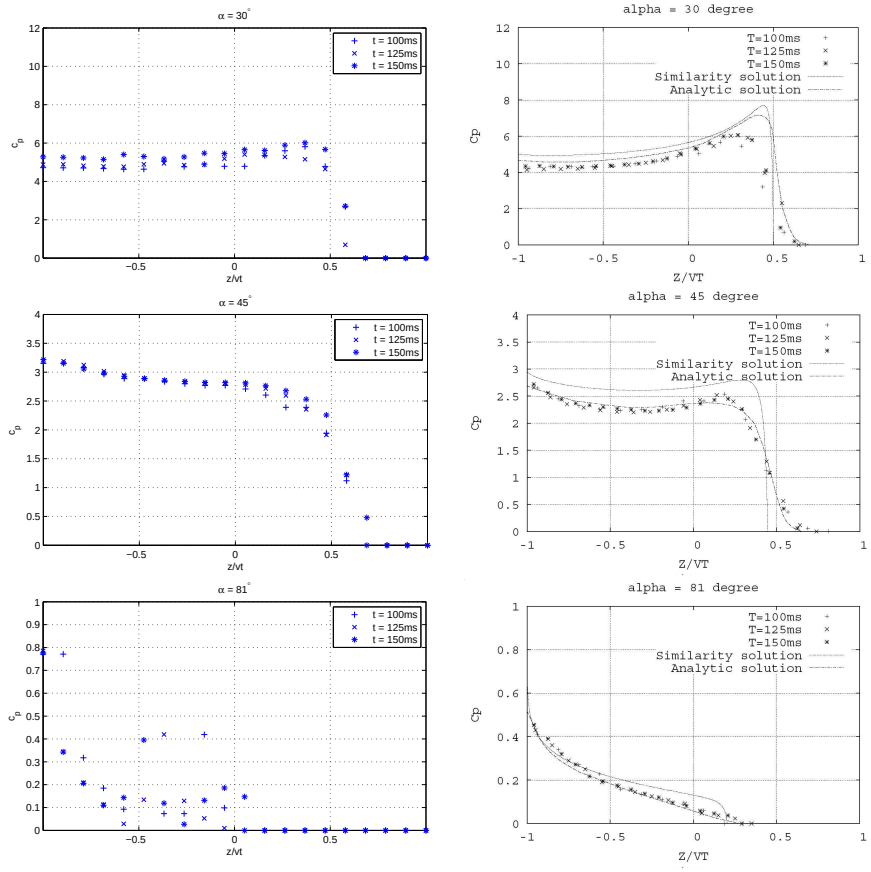

Figure 4.5: *c_p along the sloping edge of a rigid wedge as simulated by Comflow (left) and Amazon-sc (right, [22]).*

entry tests described in [5] and [22] are repeated by COMFLOW. For both testcases the wedge has a prescribed velocity. In the first test (Section 4.2.1) the pressure coefficient along the sloping edge of the wedge during the first moments of the impact is studied. The second test (Section 4.2.2) focuses on the interface behaviour when the wedge is totally immersed into water.

4.2.1 Water entry of a wedge

The pressure coefficient, $c_p = 2p/\rho v^2$, is studied along the sloping edge of a wedge entering a water surface. The wedge has three different deadrise angles (the angle between the wedge slope and a horizontal line) of $30°, 45°$ and $81°$ respectively. When the penetration depth $z = vt$ is small, analytical expressions for c_p can be found based on potential flow assumptions, see for example the analytical solutions described by Mei et al. in [18]. In theory, the solutions for different times at these early stages are self-similar if they are properly non-dimensionalised. During the first stages of impact the analytical solutions accurately resemble experimental results. In our tests we identify similarity solutions during the first 0.15 seconds of impact and compare them to the analytical solutions.

The experiment is conducted in a computational domain of 2×2 meters ($0 \leq x \leq 2$ and $-1 \leq z \leq 1$) which is half filled with water. The domain is discretised with 160×160 grid cells, implying a mesh width of 0.0125. The base angle of the wedge is initially situated at $(x, z) = (0, 0)$ and the wedge moves downwards with a prescribed velocity of $v = 1\ m/s$. To identify similarity solutions, z and x are non-dimensionalised by the penetration depth vt. The pressure coefficient along the sloping edge of the wedge is thus plotted for $-1 \leq z/vt \leq 1$. Note that in experimental test cases the pressure measured at the sloping edge is the sum of the hydrodynamic and hydrostatic pressure, $p_{tot} = p + ghz$. The analytical solution only accounts for the hydrodynamic part, which means the simulations are done with $g = 0\ m/s^2$. Note that ignoring the gravitational forces is only physically accurate since the impact is simulated for a very short period of time: the simulations is stopped before waves start propagating or droplets disconnect from the surface.

The results of the simulation are given in Figure 4.5, with COMFLOW on the left and AMAZON-SC on the right. The frames on the right show that the numerical results of AMAZON-SC are in agreement with the analytic solution described by Mei et al [18]. From this it is concluded that in this case AMAZON-SC is effective and accurate [5, 22].

In the left frames of Figure 4.5 it is shown that the results of COMFLOW give nice similarity results as well for $\alpha = 30°$ and $45°$. However, two small differences should be noticed. First, the value for c_p along the part of the edge that is below the still water level ($-1 \leq z/vt \leq 0$) is a little higher in COMFLOW than in AMAZON-SC. Second, the increase in c_p for

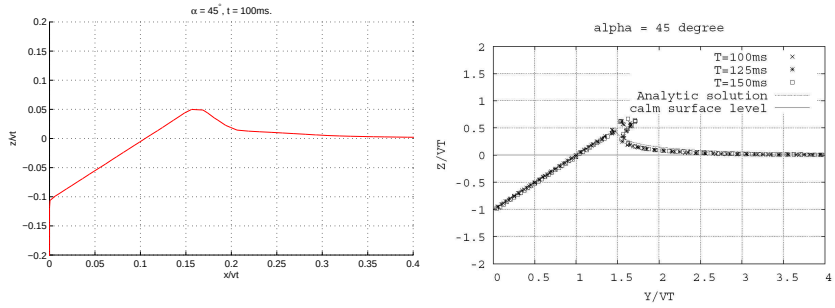

Figure 4.6: *Jet development for a rigid wedge with a deadrise angle of* $45°$ *as simulated by Comflow (left) and Amazon-sc (right, [22]).*

$0 \leq z/vt \leq 0.3$ that is clearly present on the right is slightly less on the right. This is explained by the magnitude of the jets that are produced by the impact. Grid refinement studies in [14, Figure 2.61] and [7] show that the jets are only captured accurately when the grid resolution is fine enough. On coarser grids however the magnitude of the jets is underestimated – they are smaller than expected. The decline in pressure coefficient for $z/vt > 0$ implies that on this grid the jets are not completely developed. From the frames in Figure 4.6 it becomes clear that AMAZON-SC indeed gives better developed jets. As is the case in the wave generation described in Section 4.2 the results of AMAZON-SC on a coarse grid are more accurate than those of COMFLOW.

Another – and more conspicuous – difference is the irregular behaviour of c_p for $\alpha = 81°$ in COMFLOW. This irregular behaviour is explained by the spiky behaviour of the pressure. This can clearly be seen in the left of Figure 4.7, where the development of the pressure coefficient in time is monitored on the base angle of the wedge ($\alpha = 81°$). Altough not really visable, these irregularities are also present in the other two cases. This can be seen in the right of Figure 4.7, where the development of the pressure coefficient in time is plotted (with the same range) for $\alpha = 30°$. It can be seen that the spikes for $\alpha = 30°$ are just as big (or bigger) as for $\alpha = 81°$. These wiggles are not so disastrous for $\alpha = 30°$ since they are smaller than the general level of the pressure coefficient. However for $\alpha = 81°$ the spikes are of the same order as the physical pressure signal, resulting in a numerical solution that is completely dominated by the spikes.

The spiky pressure behaviour of c_p in COMFLOW is not easily explained. To begin with, the pressure is rather sensitive to irregularities in the flow since the pressure calculated by the Poisson equation is such as to maintain a divergence free velocity field. Every disturbance in the flow thus has to be 'mended' by the pressure. This in contrast to the equation of state $p = p(\rho)$

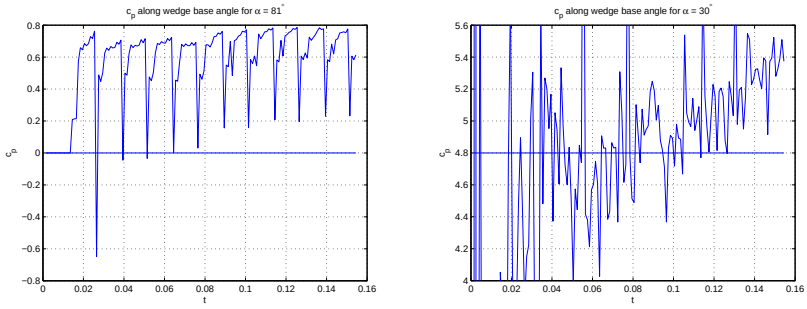

Figure 4.7: *For $\alpha = 81°$, the spiky c_p behaviour in time at the base angle of the wedge (left) and a third order polynomial reconstruction of c_p along the sloping edge of the wedge (right).*

used in compressible flow.

To explain the occurrence of these irregularities, Fekken identifies three causes [7, section 2.4]. First, attention has to be paid to the boundary conditions at the free surface. Fekken effectively gives methods to prevent spikes originating from this cause. As a second cause for pressure spikes, moving objects with sharp corners are identified. The changing geometry apertures and labels cause aperture 'jumps'. In combination with the no-slip condition at the object's boundary – that is the fluid is forced to move according to the moving body – this causes spikes in the pressure terms. A couple of solutions are proposed, but no general method is clearly effective. The third cause is the positioning of the pressure in cut cells. Especially in bodies that move diagonally through the grid, pressure spikes occur. At the moment no measures are taken and it is clear that in our wedge entry testcases the pressure spikes can largely be attributed to this last cause.

Fekken finally concludes that smearing the interface over a few cells prevents pressure spikes and a less smooth pressure signal is the price to be paid for keeping a sharp interface. This conclusion is in accordance with the results of Amazon-sc. The free surface of Amazon-sc is less sharp and at the same time this method does not really suffer from the pressure spikes.

Comparing the two methods it is concluded that for this wedge entry case the results of Amazon-sc are a little better than Comflow. First, the spiky pressure behaviour in Comflow gives inaccurate results. Second, the underdeveloped jets indicate that Comflow needs more accurate grids to produces the same results as Amazon-sc.

4.2.2 Water entry and total immersion of a wedge

The second experiment on wedge entry phenomena includes a two dimensional body with a wedged shaped bottom (deadrise angle of 45°) that is

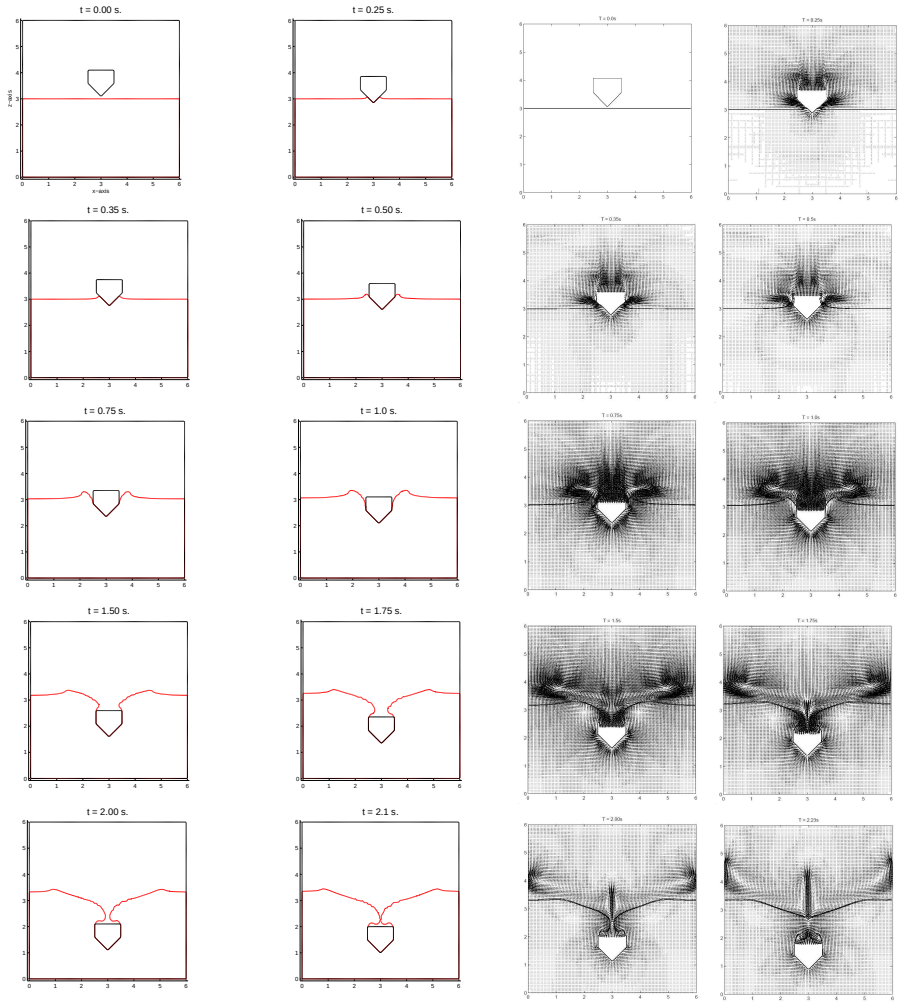

Figure 4.8: *Interface for a 2D wedge entering the water surface as simulated by Comflow (left) and Amazon-sc (right, [22]).*

entering and immersing into water. The geometry and initial position of the body are shown in the first frame of Figure 4.8. The body has a constant downwards velocity of 1 m/s. The gravitational acceleration is taken as 2 m/s^2, and the characteristic length of the body is 1 m. This results in a Froude number, $Fr = v/\sqrt{gl}$, of 0.71. A grid of 120×120 is used for a computational domain of 6×6 meters that is half filled with water, implying a mesh width of 0.05.

Although there is no experimental data available for comparison, this experiment shows how the methods handle moving bodies in addition to interface break-up and reconnection. We therefore compare the general behaviour of the surfaces.

The simulation was first performed with AMAZON-SC, and the results are described in [5] and [22]. When the simulation is repeated with COMFLOW, the results show a good general resemblance. See the left and right frames of Figure 4.8 for comparative snapshots of the development of the free surface. As the body pierces through the water two jets are formed on each side of the wedge. When the wedge travels further down into the water, the overturning waves start closing the gap behind the wedge. The surfaces finally reconnects, leaving some air bubbles trapped behind the body.

On a global level the main difference is the reconnection of the surface. In the last frame of Figure 4.8 it can be seen that with COMFLOW the surface reconnects at $t \approx 2.1$ s, which is a little earlier than the $t \approx 2.23$ s that AMAZON-SC predicts. This is to be expected since COMFLOW does not model the air in the gap between the two overturning waves. It should however be noted that COMFLOW detects the reconnection a little earlier since it models a sharper boundary of the free surface.

4.3 Flow around wave energy devices

The numerical code AMAZON-SC has been used in the research to the design of wave power devices. In cooperation with Queen's University Belfast (QUB) the possibilities have been investigated to develop an oscillating wave surge converter (OWSC) wave power device. This is a device to exploit the energy of near shore waves by means of an interactive oscillating flap. The flap either has a bottom hinge or a top hinge, as shown in Figure 4.9.

In this section the interactive motion as simulated with COMFLOW is described. Interactive motions between fluid and bodies have been implemented in COMFLOW, but so far have not been extensively tested. The interactive motion of a rotating vane is one of the motions, that had not been simulated up to now. Without striving to get quantitative results to test the design, the vane is simulated to see the applicability of COMFLOW.

In Fekken [7, § 2.5], a description is given of the algorithm used for coupled fluid-body motion. In COMFLOW the algorithm is implemented

Figure 4.9: *Interactive oscillation vane, top hinged (left) and bottom hinged (right)*

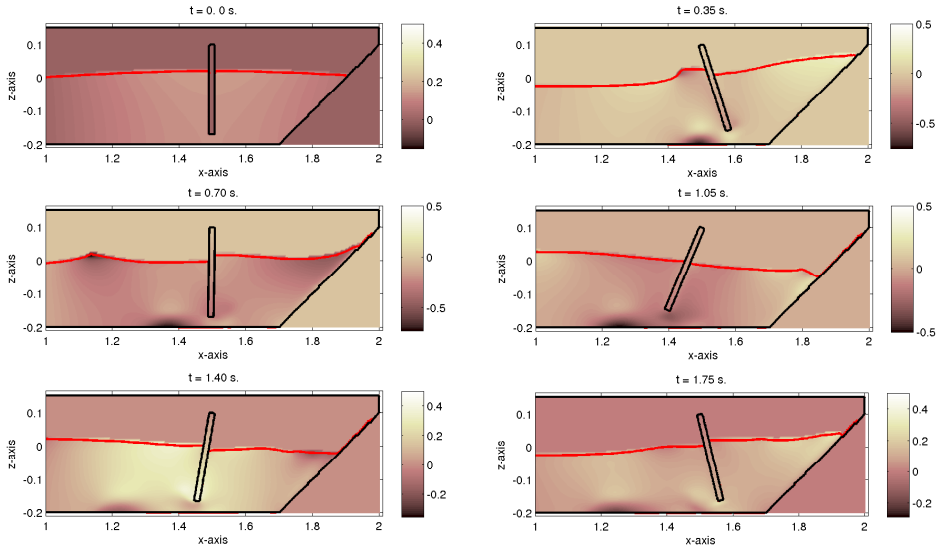

Figure 4.10: *The horizontal velocity profile around an interactive moving vane with top hinge for one period at* $t = 0,\ 0.35,\ 0.7,\ 1.05,\ 1.4,\ 1.75\ s.$

in the routine `dynint.f`. To simulate the interactive rotating vane with COMFLOW, Fekken's routine for coupled interactive fluid-body motion has to be slightly adjusted. In the previous version of the routine it was assumed that the centre of mass and the centre of rotation coincide, implying that the gravity has no influence on the torque (and thus on the angular acceleration and angular velocity). For the vane with a fixed hinge, the centre of mass and the centre of rotation are clearly not the same. Hence, gravity terms should be taken into account in the calculation of the torque. Besides, the velocity of the centre of rotation should be set to zero. The resulting (adjustment to the) subroutine is given in Appendix A.

In mathematical terms the adjustments are as follows. First the angular acceleration (around the three axes) due to the incoming wave is calculated as $\boldsymbol{\alpha} = J^{-1}N$, with J the diagonal matrix containing the moments of intertia around the three axes and N the torque vector. Discretely, the torque vector

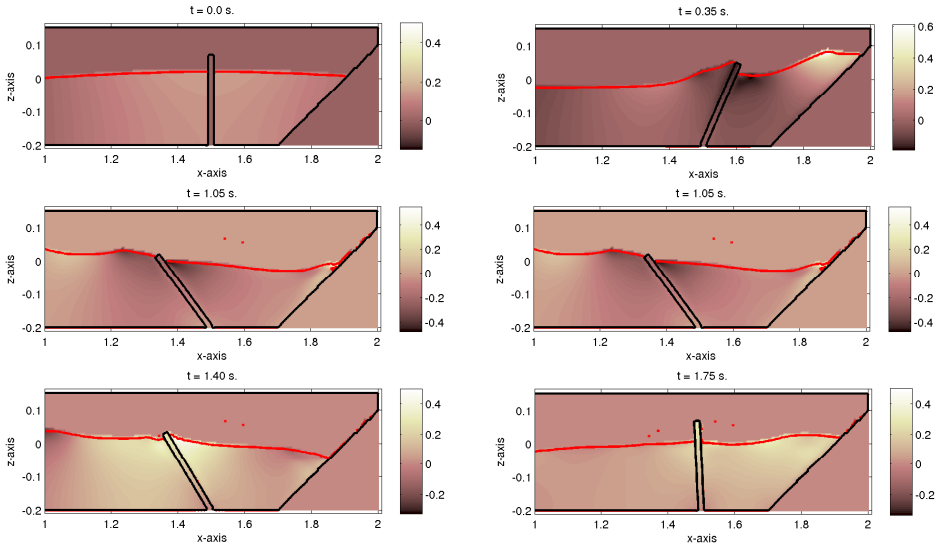

Figure 4.11: *The horizontal velocity profile around an interactive moving vane with bottom hinge for one period at* $t = 0,\ 0.35,\ 0.7,\ 1.05,\ 1.4,\ 1.75\ s$.

N around a fixed point positioned at $\boldsymbol{x}_c$ is calculated as

$$N = \sum_{\boldsymbol{x} \in \Omega_b} (\boldsymbol{f}_b(\boldsymbol{x}) + \boldsymbol{g}) \times (\boldsymbol{x} - \boldsymbol{x}_c), \tag{4.1}$$

with $\boldsymbol{g} = (0, 0, g_z)$ the gravity, $\boldsymbol{x}$ the coordinates of the computational cells and Ω_b the volume of the moving body. The force term $\boldsymbol{f}_b(\boldsymbol{x})$ is calculated by taking the integral of the pressure over the boundary of the moving body. When $\boldsymbol{x}$ is in the interior of the moving body, the force is zero. The moments of intertia are calculated as

$$\mathrm{J} = m_b \sum_{\boldsymbol{x} \in \Omega_b} \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} (\boldsymbol{x} - \boldsymbol{x}_c)^2, \tag{4.2}$$

with m_b the mass of the moving body.

With the adjusted routine for interactive motion, the first test case is the vane with the top hinge. This vane has a density of $\rho_{vane} = 1.2\rho_w$, with $\rho_w = 1000\ kg/m^3$ is the density of the water. In the scaled model, the vane is 0.27 meters long and the hinge is positioned 0.1 meters above the still water level, such that the clearance between the paddle and the bottom is 3 centimetres. In the first frame of Figure 4.10, the initial position of the vane is depicted. The incoming linear wave (which already has a fully

developed wave profile for $t = 0$) has a period of $T = 1.6$ seconds and an amplitude of 0.02 meters. Since the still water depth h is 0.2 meters, the dispersion relation gives a wave length of $\lambda \approx 2.1\ m$. Since this implies that the shore on the right of the domain (see first frame of Figure 4.10) is positioned approximately a quarter of a wavelength away from the vane, this wave profile ensures that the waves reflecting from the shore do not damp the motion of the vane. The discretisation is performed in two dimensions on a grid with $\Delta x = \Delta y = 0.005$. The simulation results in a vane that moves nicely in accordance with the incoming wave.

The second test case is the vane with the bottom hinge, which has a density of $\rho_{vane} = 0.9\rho_w$ to prevent it from sinking to the bottom. The vane is attached to the bottom and is 0.27 m long, such that it sticks out 7 cm above the still water depth, see the first frame of Figure 4.11. Furthermore, the configuration (incoming wave and grid) is the same as for the top hinge. The simulation also results in a vane that moves according to the incoming wave. Since this vane is lighter than the top-hinged one, the amplitude is a little bigger.

The frames show that COMFLOW gives promising results for the interactive body-fluid motion of the OWSC. Still, validation with experimental results is needed to further investigate the accurateness of the current solution.

Chapter 5

Theoretical comparison

In the previous chapters two examples of computational methods for fluid flow have been described. In addition some numerical results have been given. In this chapter all information is brought together. The two methods are compared to emphasise the different choices that have been made in the modelling and discretisation. It is studied why these choices have been made and what the effect of the choices is on the numerical results. The possibility to enhance either solver by methods used in the other is investigated. Mirroring the outline of Chapters 2 and 3, the subsequent sections deal with the physical, mathematical and numerical properties.

5.1 Physical model

Owing to the great similarity in the practical problems that Amazon-sc and Comflow aim to simulate, the physical models of the two methods correspond for the larger part. Both are methods for incompressible free surface flow with moving solid bodies, with an emphasis on the impact of waves on (moving) structures. The main physical differences are the way the free surface is handled, the flow domain, the viscosity properties of the fluid and the surface tension.

5.1.1 Handling the free surface

The differences in the way the free surface is handled, are made more precise by the three main categories for handling free surface problems that are described in the article of Kelecy and Pletcher [13].

The first category of modelling free surfaces consists of *surface fitting* methods. These are Lagrangian methods in which the moving surface is accounted for by fitting the grid to the moving boundary. The computational domain is therefore the same as the physical fluid region and the equations are typically solved in one phase only. When the free surface is strongly

deformed, the transformation of the domain may become problematic. Since the fluid flow both AMAZON-SC and COMFLOW aim to simulate involves highly deformed interfaces, this type of method is not applicable.

Second, *surface tracking* methods are distinguished. They are Eulerian methods, where the location of the boundary is defined relative to a fixed grid. At each time step grid cells are typically flagged as containing either gas, liquid or free surface and often the equations are solved only in cells containing fluid and free surface. The free surface forms the moving boundary of the flow domain. Since the potential domain for fluid flow should be taken into account, the computational grid for surface tracking methods is larger than for surface fitting methods. COMFLOW is a typical example of a surface tracking method.

Third, *surface capturing* methods solve the flow field simultaneously within both gas and liquid. The location of the free surface is then captured as a discontinuity in density. The computational grid is the same as in surface tracking methods, with the difference that the flow equations are also solved in the gas region. Inherently, this type of method is computationally more demanding than surface fitting and tracking methods. At the same time they are potentially more accurate. AMAZON-SC is such a surface capturing method.

The physical approach to the boundary between liquid and gas is thus quite different in both methods. This different approach – surface capturing against surface tracking – is the major source for the different choices in the mathematical and numerical model. Besides, these physically different approaches to the free surface, result in computational methods that solve different physical situations. Differences in the numerical results of the two methods are therefore to be expected.

5.1.2 Flow domain

The question what the flow domain should be (that is modelling both liquid and gas or only liquid), is closely linked to way the free surface is handled. They are, however, not completely equivalent[1]. The apparent advantage of a two-phase method is the improved physical accuracy compared to the one-phase method. Especially when flow conditions get more violent, using a two-phase solver is strongly recommendable. Take for example impacting waves, where air bubbles are present in the water. Because of the mixing air and water the pressure levels and velocity field are seriously affected. The downside of using a two-phase method is of course that it is computationally more demanding.

[1] Of course a surface capturing method is inherently a two-phase method, but not all two-phase solvers are surface capturing method. A surface tracking method could, for example, also be a two-phase method.

The other question concerning the flow domain is whether to simulate arbitrary three dimensional flow, or only two dimensional flow. In Amazon-sc, most simulations are two dimensional, although parts of the solver have been extended to three dimensions. This choice for a two dimensional solver is partly made because two-phase simulations in three dimensions are computationally very demanding. On the other hand, Comflow is designed to simulated arbitrary three dimensional flow. This difference influences for example the discretisation of the cut cells (see Section 5.3.6).

5.1.3 Viscosity

In Amazon-sc the viscous effects are neglected, while in Comflow they are not. Since the flow that is simulated with both methods is convection dominated, that is for large Reynolds numbers, this difference does not have a big influence.

5.1.4 Surface tension

The surface tension does not have a big influence in the simulations considered in this thesis, since the effects of surface tension are dominated by gravitational effects, but in the absence of gravity (that is in space) surface tension is the driving force. In Comflow surface tension is modeled, as the solver is also applied to study liquid sloshing on board space craft. In Amazon-sc surface tension is not considered. This implies that Amazon-sc can not be used to accurately model fluid flow in micro-gravity environments.

5.2 Mathematical model

Mathematically, the main similarity is that both flow equations are given in conservative form. That is, both methods are *mathematically* conservative, but Comflow is not *physically* conservative, since the momentum equations of Comflow, see Equation (2.2), are given for the velocity $\boldsymbol{u}$ and not for the momentum $\rho\boldsymbol{u}$. Furthermore, the conservative flow equations differ to the extend that in Amazon-sc they are described by the Euler equations (inviscid) and in Comflow by the Navier-Stokes equations (viscous), but this is not a material difference for the problems considered. The main mathematical differences concern the boundary conditions and the mathematical character of the flow equations.

5.2.1 Boundary conditions

The most influential consequence of neglecting the viscosity is the effect this has on the boundary conditions. When viscous effects are ignored, the viscous shear stress at the boundary does not exists. This means that the

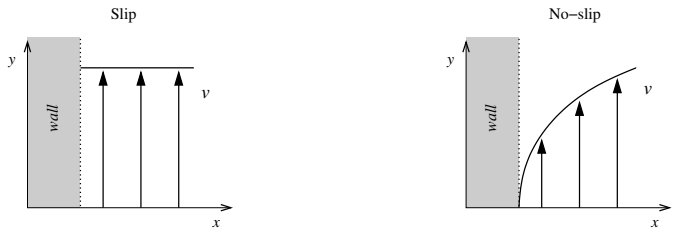

Figure 5.1: *Illustration of slip and no-slip boundary conditions for the tangential velocity.*

boundary condition for the velocity tangential to the boundary 'slips'. On the other hand, not neglecting the viscous terms results in a no-slip boundary condition for the velocity tangential to the boundary. This difference is depicted in Figure 5.1.

5.2.2 Mathematical character

Considering the continuity equations for COMFLOW and AMAZON-SC (see Equations 2.1 and 3.2 respectively), the mathematical character of both equations is elliptic. However, the mathematical character of the continuity equation in AMAZON-SC changes from elliptic to hyperbolic after adding the artificial compressibility term.

In the equations for the conservation of momentum the convective term is hyperbolic. The diffusive term (present in AMAZON-SC, but not in COMFLOW) has a parabolic character. The system of governing equation in AMAZON-SC is thus hyperbolic, while in COMFLOW they have a mixed elliptic, parabolic and hyperbolic character.

5.3 Numerical model

The main numerical similarity is that a finite volume discretisation is used on a stationary background Cartesian grid. To account for moving bodies both methods adopt a cut cell method. For the rest, there are a number of differences. Below, the differences are discussed regarding the grid arrangement, the underlying philosophy of the discretisation, the artificial compressibility, the temporal discretisation, the upwind schemes and the cut cell method.

5.3.1 Grid arrangement

In the discretisation the first difference is the arrangement of the variables, see Figure 5.2. In AMAZON-SC all variables are stored in the cell centres, called a collocated arrangement. Alternatively, in COMFLOW the variables

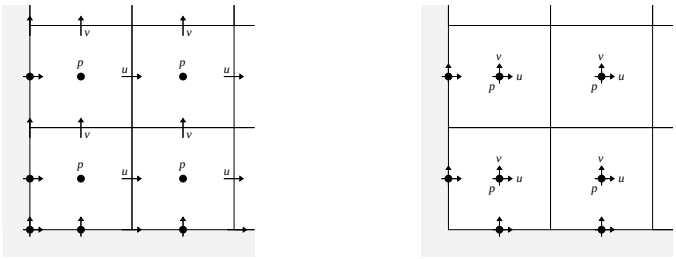

Figure 5.2: *Illustration of a staggered (left) and a collocated (right) arrangement of velocity and pressure.*

have a staggered arrangement, with the velocities stored on the cell's boundaries and the pressure stored in the cell centre.

The staggered grid was introduced in 1965 by Harlow and Welch as part of their Marker and Cell method [10]. The advantage of a staggered arrangement opposed to the collocated arrangement, is that an odd-even decoupling in the pressure terms does not occur, due to the strong coupling between velocities and pressure.

5.3.2 Underlying discretisation philosophy

The different physical approaches to the free surface result in different requirements of the numerical scheme. In Amazon-sc the numerical scheme is required to be stable around discontinuities, resulting in a Godunov-type method. In Comflow, these discontinuities are not present and the discretisation is done with the philosophy that the symmetric properties of the underlying continuous operator should be conserved to get a stable numerical scheme. The discretisation is thus based on two different philosophies.

5.3.3 Artificial compressibility

One of the most apparent differences between the two numerical models is the usage of artificial compressibility in Amazon-sc, and a Poisson equation for the pressure in Comflow. In this section it is considered what using artificial compressibility in Comflow would imply. The artificial compressibility method of Amazon-sc is then further examined.

To begin with, recall the development of the use of artificial compressibility terms. Chorin was the first to employ artificial compressibility terms to solve the steady form of the incompressible flow equations (either Euler or Navier-Stokes) in 1967 [6]. He was inspired by the idea that solving the *compressible* form of the flow equations has an apparent advantage compared to the *incompressible* form. This advantage is that the pressure appears in the compressible continuity equation, implying that there is an immediate link

between the pressure and the velocity which is absent in the incompressible flow equations. In Section 2.3.3 it is shown that this problem is solved by using a Poisson equation for the pressure. Instead of using a Poisson equation, Chorin augmented the incompressibility constraint $\nabla \cdot \boldsymbol{u} = 0$ by a (pseudo) time derivative of the pressure. In divergence form the equation for artificial compressibility is

$$\frac{1}{\beta}\frac{\partial p}{\partial \tau} + \nabla \cdot \boldsymbol{u} = 0, \tag{5.1}$$

with β an appropriate constant.

By introducing an equation of the form (5.1), changes in the velocity field are coupled to changes in the pressure field. To maintain a divergence free velocity field it should hold that $\partial p/\partial \tau = 0$ for $\tau \to \infty$. This condition is satisfied when solving the steady Navier-Stokes equations. For the unsteady form however, the restriction is not necessarily satisfied and Chorin's method should be enhanced. This is done by introducing a dual-time stepping technique. When the solution at $t_n \equiv n\Delta t$ is known, the solution at t_{n+1} is calculated by iterating in pseudo time, say with steps $\Delta\tau$, until a steady state is achieved. The solution at the new time step t_{n+1} is then given by the steady state solution in pseudo time. In doing so it is ensured that the method is time accurate for unsteady flow. This technique was among others introduced by Soh and Goodrich in 1988 [25].

In Section 3.2.2 it has been shown how this technique is applied in Amazon-sc. But how can this technique be incorporated in Comflow? To begin with the governing equations are now the momentum equations (2.2) and the continuity equation (2.1) augmented with a artificial compressibility term[2]. With a few rearrangements they are given by

$$\frac{1}{\beta}\frac{\mathrm{d}}{\mathrm{d}\tau}\int_\Omega p \,\mathrm{d}\Omega + \oint_S \boldsymbol{u}\cdot\boldsymbol{n}\,\mathrm{d}S = 0, \tag{5.2a}$$

$$\frac{\mathrm{d}}{\mathrm{d}t}\int_\Omega \boldsymbol{u}\,\mathrm{d}\Omega + \oint_S \frac{1}{\rho}p\boldsymbol{I}\cdot\boldsymbol{n}\,\mathrm{d}S = \boldsymbol{R}(\boldsymbol{u}), \tag{5.2b}$$

with

$$\boldsymbol{R}(\boldsymbol{u}) \equiv \oint_S \left(\frac{\mu}{\rho}\nabla\boldsymbol{u} - \boldsymbol{u}\otimes\boldsymbol{u}^T\right)\cdot\boldsymbol{n}\,\mathrm{d}S + \int_\Omega \boldsymbol{g}\,\mathrm{d}\Omega. \tag{5.2c}$$

The equations are discretised in space using the methods and notation described in Section 2.3.2. Using the explicit forward Euler method to discretise time, see also Section 2.3.3, the following equations are obtained

[2]For simplicity the momentum equations are not augmented with the artificial compressibility terms, but with a few adjustment the argument given below, is still valid in this case.

$$\begin{aligned} \boldsymbol{p}^{k+1} &= \boldsymbol{p}^k - \beta\Delta\tau\mathcal{M}\boldsymbol{u}^k, && \text{(5.3a)} \\ \boldsymbol{u}^{n+1} + \Delta t\mathcal{G}\boldsymbol{p}^{n+1} &= \mathcal{R}^n. && \text{(5.3b)} \end{aligned}$$

Here $\mathcal{M}$ is the discrete divergence matrix working on the whole domain and $\mathcal{G} \equiv \mathcal{G}_{\rho,\Omega}$ the discrete gradient. The rest term $\mathcal{R}^n$ is defined by Equation (2.26). The variables $\boldsymbol{p} \equiv \boldsymbol{p}_h$ and $\boldsymbol{u} \equiv \boldsymbol{u}_h$ are the discrete vectors of the pressure terms and velocity term. The indices k and n are for the pseudo and real time respectively.

The solution at the new physical time step $n+1$ is given when a steady state is reached in pseudo time. This means that Equation (5.3a) is iterated for every physical time step and $\boldsymbol{p}^{n+1,k+1} = \boldsymbol{p}^{n+1}$ once $|\boldsymbol{p}^{n+1,k+1} - \boldsymbol{p}^{n+1,k}| < \varepsilon$ for a prescribed accuracy bound ε. Now suppose that an estimate $\boldsymbol{p}^{n+1,k}$ for the pressure is given, then Equation (5.3b) gives an estimate for the velocity $\boldsymbol{u}^{n+1,k}$ and Equation (5.3a) yields the new pressure estimate. This leads to the following algorithm

$$\left\{ \begin{array}{lcl} \boldsymbol{u}^{n+1,k} &=& \mathcal{R}^{n,k} - \Delta t\mathcal{G}\boldsymbol{p}^{n+1,k}, \\ \boldsymbol{p}^{n+1,k+1} &=& \boldsymbol{p}^{n+1,k} - \beta\Delta\tau\mathcal{M}\boldsymbol{u}^{n+1,k}. \end{array} \right. \tag{5.4}$$

At first sight, the technique above seems quite different from using the Poisson equation for the pressure. Taking a closer look, it becomes clear that in fact it is not.

Inserting the expression for $\boldsymbol{u}^{n+1,k}$ in the expression for $\boldsymbol{p}^{n+1,k+1}$ gives, with a little rearranging,

$$\boldsymbol{p}^{n+1,k+1} = \left(\boldsymbol{I} + \beta\Delta t\Delta\tau\mathcal{M}\mathcal{G}\right)\boldsymbol{p}^{n+1,k} - \beta\Delta\tau\mathcal{M}\mathcal{R}^{n,k}. \tag{5.5}$$

In this equation, the iterative scheme of the Jacobi method with overrelaxation (JOR) is recognised[3], that is Equation (5.5) is the JOR-scheme for

$$\mathcal{M}\mathcal{G}\boldsymbol{p}^{n+1} = \frac{1}{\Delta t}\mathcal{M}\mathcal{R}^n, \tag{5.6}$$

with the unsteady relaxation parameter given by

$$\omega = -\beta\ \Delta t\ \Delta\tau\ m_i. \tag{5.7}$$

[3]The solution of the matrix equation $Ax = b$, is found iteratively with the the JOR-scheme as $x^{n+1} = (I - \omega D^{-1}A)x^n + \omega D^{-1}b$, where D is the diagonal of A. The JOR-scheme converges for the underrelaxation values $0 \leq \omega \leq 1$ whenever A is irreducible and weakly diagonally dominant [27, p. 67].

Here m_i is the diagonal element of $\mathcal{MG}$ at the ith grid point. Since the Laplace operator $\mathcal{MG}$ is irreducible and weakly diagonally dominant, an error bound for the real time step Δt is found. When $M = \max(|m_i|)$,

$$0 \leq \Delta t \leq \frac{1}{\beta\ \Delta\tau\ M}. \tag{5.8}$$

This means that the convergence rate of Equation (5.5) is largely dependant on the value of β.

Using an explicit forward Euler discretisation of time, the method of artificial compressibility is thus equivalent to solving the Poisson equation with a unsteady JOR-scheme. There are, however, more efficient methods than JOR to solve the Poisson equation, such as the SOR (successive over relaxation) method. In a SOR-scheme the optimal relaxation parameter is determined, and adjusted, during the iterations. Clearly, using the method of artificial compressibility in COMFLOW in the form described above would not be efficient.

In other words, the above implies that artificial compressibility with dual-time stepping is effectively an iterative scheme. The physical arguments for using this method do, however, not guarantee that it is the most effective iterative scheme. In this light, it seems that using artificial compressibility in AMAZON-SC might not be such a good idea. At least, for COMFLOW there are more efficient iterative schemes than using artificial compressibility. If this holds as well in the case of AMAZON-SC, is examined in the following by means of two observations.

First observe that two-phase solvers that automatically capture the interface as a discontinuity in the density, must be able to cope with large discontinuities. Methods that are able to do this are typically Godunov-type methods. They introduce information from the local exact or approximate solution of the Euler equations in the numerical method. Thus bringing about a closer interaction between the discretisation method and the physical properties of the flow. However, Godunov-type methods are only available for *hyperbolic* equations.

The second observation is that, from a mathematical point of view, the major difference between the equations of compressible and incompressible flow is their mathematical character. The compressible Euler equations are hyperbolic. By contrast, the incompressible equations have a mixed hyperbolic-elliptic character. This implies that most numerical methods for compressible flow can not be applied to incompressible flow. Not unless the elliptic character of the continuity equation is modified to form a hyperbolic equation. Incorporating an artificial compressibility term is a straightforward means of doing this.

Following this line of reasoning, the implementation of an artificial compressibility Godunov-type method comes naturally for AMAZON-SC. In view

of the above, the argument given in the case of COMFLOW against the use of artificial compressibility, is not sustainable in the case of AMAZON-SC. Nevertheless, it is likely to be one of the reasons why AMAZON-SC takes more computational time than COMFLOW.

As noticed before, the value for the artificial compressibility parameter β is one of the keys to the performance of this method – and thus for the amount of computational time. In case of the explicit forward Euler discretisation in time, Δt dependent of β, see Equation (5.8). Unfortunately, such a bound has not been found for the discretisation of AMAZON-SC . Intuitively, larger β seem to be preferable, since the larger the value of β, the more the 'incompressible' the equations become. On the other hand, larger β make the equations stiff numerically, forcing the time step to be small. The value for β that yields a good convergence, thus has to be found heuristically.

Concluding, whether or not to implement artificial compressibility is closely linked with the physics the methods aim to model. When the aim is a two-phase surface capturing model, the use of artificial compressibility can be seen as trade off between an increase of computational time and correctly modelling the fluid interface. Artificial compressibility is then the catalyst that enables the usage of a method that is able to cope with large discontinuities. On the other hand, the aim of the model could also be a surface tracking (one-phase) rather than a surface capturing method. In this case there is no specific need to adopt a Godunov-type method, since large discontinuities will not occur. In this case the artificial compressibility method functions 'just' as an iterative scheme, for which there are most likely faster alternatives.

5.3.4 Temporal discretisation

The temporal discretisation of COMFLOW is done with explicit Euler, while AMAZON-SC uses implicit Euler.

5.3.5 Upwind discretisation

Both COMFLOW and AMAZON-SC adopt upwind discretisations in their numerical method to circumvent spurious oscillations. On the one hand, COMFLOW uses an upwind method in the discretisation of the diffusive operator. To guarantee a monotonic behaviour of this operator, the central discretisation is changed to an upwind discretisation by adding an artificial viscosity term. This method is first order accurate. On the other hand, AMAZON-SC employs an upwind method in the reconstruction of the free surface. To get the values at the cell boundaries a non-linear higher order upwind reconstruction is used by means of applying limiters. This implies that the upwind method used in AMAZON-SC is more accurate than the one used in

COMFLOW.

This difference between the upwind methods can possibly explain the difference between the behaviour of AMAZON-SC and COMFLOW on coarser grids. The use of the non-linear higher order upwind method could explain why AMAZON-SC is more accurate on a coarse grid than COMFLOW is.

5.3.6 Cut cell method

The first difference in the cut cell methods of AMAZON-SC and COMFLOW is concerned with the flow domain. Due to the two dimensional nature of most simulations of AMAZON-SC, an exact cut cell method is adopted. Since COMFLOW is designed to simulate arbitrary three dimensional geometries, and approximate cut cell method is adopted. For three dimensional simulations in AMAZON-SC, cut cells are also treated in a approximate manner, but without using volume and edge apertures as in COMFLOW. Instead the cut cells are approximated by allowing for one cut face per cell in the form of a non-planar quadrilateral.

The second difference in the cut cell method of AMAZON-SC and COMFLOW is the merging of small cells. In COMFLOW the discretisation is performed in such a way that it is stable for small cut cells. In AMAZON-SC small cut cells are merged with other cells.

Chapter 6

Conclusions

In the previous chapters the theoretical basis of Amazon-sc and Comflow is presented and compared. Additionally results of simulations performed with both Amazon-sc and Comflow are presented. The conclusion deals with a summary of the differences and similarities of the two methods and some recommendations for future work.

6.1 Similarities and differences

The two methods roughly aim at the same sort of simulations, that is, the simulation of flow with a free surface around (moving) objects. Due to the resemblance in the type of simulations, there are a number of elements in both solvers that are similar. In Table 6.1 all these similarities are summarised.

Despite of the similarities, there are quite a number of differences between the two solvers (see Table 6.2). For the larger part these differences can be traced back to the different approach to handle the free surface. In Comflow the free surface is considered as the moving boundary of the domain. An essential part of the accuracy and robustness of the solver depends on the surface boundary conditions and the way the surface is tracked. On the other hand, in Amazon-sc the free surface is considered as a discontinu-

Similarities

Physics	*Mathematics*	*Numerical*
moving solid bodies	integral formulation	finite volume
free surface	conservative formulation	stationary Cartesian grid
incompressible		cut cell method
isothermal		

Table 6.1: *Similarities between ComFlow and Amazon-sc.*

Differences

	COMFLOW	AMAZON-SC
Physics		
flow domain	three dimensional	two dimensional
flow equations	one-phase	two-phase
free surface	moving boundary	discontinuity in ρ
viscosity	viscous flow	inviscid flow
Mathematics		
$\boldsymbol{v}$ boundary condition	no-slip	slip
character equations	parabolic, hyperbolic and elliptic	hyperbolic
flow equations	Euler	Navier Stokes
integral formulation	math. conservative	phys. conservative
Numerical		
variable arrangement	staggered	collocated
underlying philosophy	symmetry preserving	stable at discontinuity
p-$\boldsymbol{v}$ link	Poisson equation	artificial compres.
time integration	explicit	implicit
small cut cells	no merging	merging

Table 6.2: *Differences between ComFlow and Amazon-sc.*

ity in the density. As a result, the 'problems' with boundary conditions and tracking the movements of the free surface, do not occur in AMAZON-SC. Nevertheless, AMAZON-SC has other numerical issues, since the numerical scheme of AMAZON-SC is required to be stable around the large discontinuity at the free surface. Moreover, the discontinuity should stay sharp during the simulation. Numerical schemes that can handle large discontinuities are Godunov-type methods. However, Godunov-type methods are only available for hyperbolic equations. In AMAZON-SC the governing equations are expanded with an artificial compressibility constraint to obtain hyperbolic equations. The downside of artificial compressibility is that it is essentially an rather slow iterative method, see Section 5.3.3, to solve the Poisson equation.

To conclude on the difference and similarities of the two methods, we return once more to the title of the thesis. Although COMFLOW and AMAZON-SC show a superficial resemblance they differ considerably in their theoretical basis. They are as different as chalk and cheese.

6.2 Future work

The comparison leads to two recommendation for the improvement of the two methods. Future work for both methods could be:

- COMFLOW can possibly be improved by replacing the first order upwind method by a higher order non-linear upwind method, like the one used in AMAZON-SC. This would hopefully have the result that COMFLOW is able to handle coarser grids better than it does at the moment.

- AMAZON-SC can possibly be improved by reconsidering the use of artificial compressibility. Instead, the equations could be rearranged such that a Poisson equation for the pressure is formed. The rest of the hyperbolic equations can be solved with Godunov's method. Using a fast iterative method to solve the Poisson equation for the pressure would lessen the computational time needed.

Bibliography

[1] *The MacTutor History of Mathematics archive*, School of Mathematics and Statistics, University of St Andrews, Scotland, URL: http://www-history.mcs.st-andrews.ac.uk, 2006.

[2] E. Botta and M. Ellenbroek, *A modified SOR method for the Poisson eqaution in unsteady free-surface flow calculations*, Journal of Computational Physics, 60 (1985), pp. 119–134.

[3] D. Causon, D. Ingram, and C. Mingham, *A Cartesian cut cell method for shallow water flows with moving boundaries*, Advances in Water Resources, 24 (2001), pp. 899–911.

[4] D. Causon, D. Ingram, C. Mingham, G. Yang, and R. Pearson, *Calculation of shallow water flows using a Cartesian cut cell approach*, Advances in Water Resources, 23 (2000), pp. 545–562.

[5] D. Causon, C. Mingham, D. Ingram, and L. Qian, *Numerical experiments on slamming of rigid wedge-shaped bodies*, in Proceedings of the Fourtheenth International Offshore and Polar Engineering Conference, 2004, pp. 546 – 553.

[6] A. Chorin, *A numerical method for solving incomprenssible viscous flows problems.*, Journal of Computational Physics, 2 (1967), pp. 12–26. (Reedited in 1997, volume 135, pages 118–125).

[7] G. Fekken, *Numerical simulations of free-surface flow with moving rigid bodies*, PhD thesis, University of Groningen, the Netherlands, URL: http://irs.ub.rug.nl/ppn/260616745, 2004.

[8] J. Gerrits, *Dynamics of liquid-filled spacecraft*, PhD thesis, University of Groningen, the Netherlands, URL: http://irs.ub.rug.nl/ppn/234864656, 2001.

[9] S. Godunov, *Reminiscences about difference schemes*, Journal of Computational Physics, 153 (1999), pp. 6–25.

[10] F. Harlow and J. E.Welch, *Numerical calculations of time-dependent visous incompressible flow of fluid with free surface*, The Physics of Fluids, 8 (1965), pp. 2182 – 2189.

[11] C. Hirsch, *Numerical computation of internal and external flows. Volume 2: Computational methods for inviscid and viscous flows.*, Wiley Series in Numerical Methods in Engineering, 1990.

[12] C. Hirt and B. Nichols, *Volume of fluid (vof) method for the dynamics of free boundaries*, Journal of computational physics, 39 (1981), pp. 201–225.

[13] F. Kelecy and R. Pletcher, *The development of a free surface capturing approach for multidimensional free surface flows in closed containers.*, Journal of Computational Physics, 138 (1997), pp. 939–980.

[14] K. Kleefsman, *Water impact loading on offshore structures - a numerical study*, PhD thesis, University of Groningen, the Netherlands, URL: http://irs.ub.rug.nl/ppn/288227360, 2005.

[15] K. Kleefsman, G. Fekken, A. Veldman, B. Iwanowski, and B. Buchner, *A volume-of fluid based simulation method for wave impact problems*, Journal of Computational Physics, 206 (2005), pp. 262–393.

[16] B. V. Leer, *Erratum: Godunov's method for gas dynamics: Current applications and future development*, Journal of Compuational Physics, 134 (1997), p. 199.

[17] ——, *An introduction to the article "reminiscences about difference schemes" by S.K.Godunov*, Journal of Compuational Physics, 135 (1999), pp. 1–5.

[18] X. Mei, Y. Liu, and D. Yue, *On the water impact of general two-dimensional sections*, Applied Ocean Research, 21 (1999), pp. 1–15.

[19] C. Mingham, L. Qian., D. Causon, and D. Ingram, *Non-linear simulation of wave energy devices*, in Proceedings of the Fourtheenth International Offshore and Polar Engineering Conference, 2004, pp. 194 – 201.

[20] D. Pan and C. Chang, *The capturing of free surfaces in incompressible multi-fluid flow*, International Journal for Numerical Methods in Fluids, 33 (2000), pp. 203–222.

[21] L. Qian, D. Causon, D. Ingram, and C. Mingham, *Cartesian cut cell two-fluid solver for hydraulic flow problems*, ASCE Journal of Hydraulic Engineering, 129 (2003), pp. 688–696.

[22] L. Qian, D. Causon, C. Mingham, and D. Ingram, *A free-surface capturing method for two fluid flows with moving bodies*, Proceedings of the Royal Society A, 462 (2006), pp. 21–42.

[23] P. Roe, *Approximate Riemann solvers, parameter vecotors, and difference schemes*, Journal of Computational Physics, (1981).

[24] P. Roe and J. Pike, *Efficient construction and utilisation of approximate Riemann solutions*, in Computing methods in applied sciences and engineering, R. Glowinski and J. Lions, eds., North Holland, Amsterdam, 1984.

[25] W. Soh and J. Goodrich, *Unsteady solution of incompressible Navier-Stokes equations*, Journal of Computational Physics, 79 (1988), pp. 113–134.

[26] E. Toro, *Riemann solvers and numerical methods for fluid dynamics: A practical introduction*, Springer–Verlag Berlin Heidelberg, 1997.

[27] A. Veldman, *Lecture notes Computational Fluid Dynamics*, Rijksuniversiteit Groningen, Mathematics Department, 2001.

[28] R. Verstappen and A. Veldman, *Symmetry-preserving discretization of turbulent flow*, Journal of Computational Science, 187 (2003), pp. 343–368.

[29] C. Vreugdenhil, *Forcasting weather and climate since Richardson, 1922: Richardson, mathematical modeller*, Dynamics of Atmospheres and Oceans, 20 (1994), pp. 211–225.

[30] G. Yang, D. Causon, and D. Ingram, *Calculation of compressible flows about complex moving geometries using a three-dimensional Cartesian cut cell method*, Intenational Journal for Numerical Method in Fluids, (2000), pp. 1121–1151.

Appendix A

ComFlow-code for OWSC

In Comflow the coupled fluid-body motion is implemented in the subroutine `dynint.f`, see Fekken [7, § 2.5] for a description of the algorithm. This subroutine is slightly adjusted to simulate the interactive moving vane. The parts that have bee adjust are given below.

Discrete calculation of the mass and the moments of intertia, after initialisation:

```
      DO 11, K = 5, KMAX-4
      DO 11, J = 3, JMAX-2
      DO 11, I = 5, IMAX-4
        IF (MOVE(I,J,K).EQ.1) THEN
          M = M + RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K)
          IXX = IXX + RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K
     &         )*(((Y(J-1)+Y(J))/2 - YCOM)**2 +((Z(K-1)+Z(K))/2 - ZCOM)
     &         **2)
          IYY = IYY + RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K
     &         )*(((X(I-1)+X(I))/2 - XCOM)**2 +((Z(K-1)+Z(K))/2 - ZCOM)
     &         **2)
          IZZ = IZZ + RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K
     &         )*(((X(I-1)+X(I))/2 - XCOM)**2 +((Y(J-1)+Y(J))/2 - YCOM)
     &         **2)
        ENDIF
 11   CONTINUE
      IYY = - IYY
```

Discrete calculation of the forces and torque as caused by gravity, after initialisation:

```
      DO 12, K = 2, KMAX-1
      DO 12, J = 2, JMAX-1
      DO 12, I = 2, IMAX-1

       IF (MOVE(I,J,K).EQ.1) THEN
          GFX = RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K)*GRAVX
          GFY = RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K)*GRAVY
          GFZ = RHO_B * (1.0-FB(I,J,K))*DXP(I)*DYP(J)*DZP(K)*GRAVZ
          GFORCEX = GFORCEX + GFX
          GFORCEY = GFORCEY + GFY
```

```
          GFORCEZ = GFORCEZ + GFZ
          CALL CROSS2D(GFY,GFZ,0.5*(Y(J)+Y(J-1))-YCOM,0.5*(Z(K)+Z(K-1))
     &          -ZCOM,GMOMX)
          GMOMENTX = GMOMENTX - GMOMX
          CALL CROSS2D(GFX,GFZ,0.5*(X(I)+X(I-1))-XCOM,0.5*(Z(K)+Z(K-1))
     &          -ZCOM,GMOMY)
          GMOMENTY = GMOMENTY - GMOMY
          CALL CROSS2D(GFX,GFY,0.5*(X(I)+X(I-1))-XCOM,0.5*(Y(J)+Y(J-1))
     &          -YCOM,GMOMZ)
          GMOMENTZ = GMOMENTZ - GMOMZ
        ENDIF
 12     CONTINUE
```

Calculation of the (interactive part of the) velocity of the object:

```
      UCOM = 0.0
      VCOM = 0.0
      WCOM = 0.0
      ANGVELX = RELAX * ANGVELXNN + (1.-RELAX) * ANGVELXN +
     &          RELAX * DT * (MOMENTX+GMOMENTX) / IXX
      ANGVELY = RELAX * ANGVELYNN + (1.-RELAX) * ANGVELYN +
     &          RELAX * DT * (MOMENTY+GMOMENTY) / IYY
      ANGVELZ = RELAX * ANGVELZNN + (1.-RELAX) * ANGVELZN +
     &          RELAX * DT * (MOMENTZ+GMOMENTZ) / IZZ
```