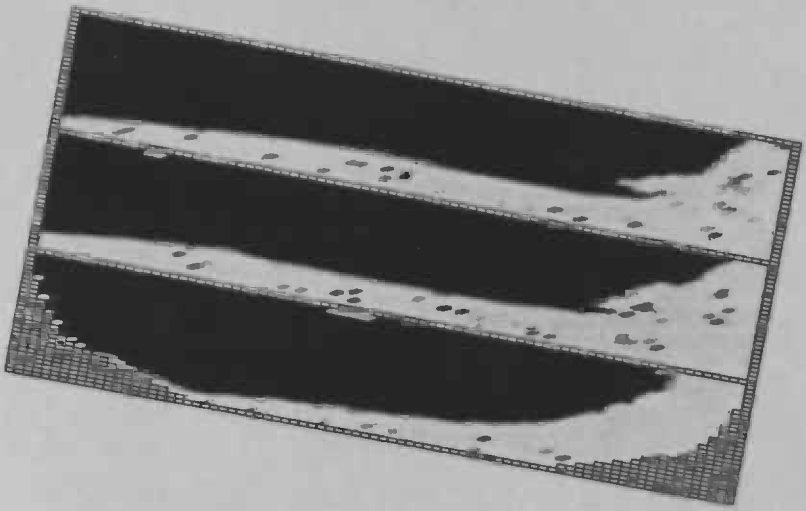
Bibl.

Two-Dimensional Simulation of an Anti Roll Tank

Arjan Bouma



Department of
Mathematics

RuG

Master's thesis

Two-Dimensional Simulation of an Anti Roll Tank

Arjan Bouma

University of Groningen
Department of Mathematics
P.O. Box 800
9700 AV Groningen

August 1998

SAVOF96 is based on the computer program SAVOF that has been developed at the National Aerospace Laboratory NLR under contract with the Netherlands Agency for Aerospace Programmes NIVR.

Contents

Chapter 1

Introduction

When a ship sails at sea it will perform a roll motion around its center of gravity, due to the interaction of the ship with waves. This roll motion can have unpleasant results for the ship and its cargo, and an anti roll tank can be placed in the ship to reduce this motion. In this report an anti roll tank was studied which consists of three tanks on top of each other, which were partially filled with water. Calculations were performed, and compared with results from MARIN, which they had available from tests.

Because the movement in the length direction is usually not very important for this kind of motion, the anti roll tank is approached with a two dimensional model. The calculations were performed with Savof96, a program to solve two dimensional fluid flow, with or without a free surface. The goal was to see how well the fluid flow in an anti roll tank could be described by a two dimensional model.

The flow of fluid is dictated by the *Navier-Stokes* equations (see chapter 2), which can only be solved analytically in very simple cases. Therefore they are discretized, and solved using an iterative method (see chapter 3). The results are in chapter 4, and in the appendix a description of Savof96 is given.

Chapter 2

Mathematical Model

2.1 The Navier-Stokes equations

The motion of fluid is prescribed by two conservation laws:

1. conservation of mass
2. conservation of momentum

The conservation laws lead to the following equations:

1. $\frac{\partial \rho}{\partial t} + \frac{\partial(\rho u)}{\partial x} + \frac{\partial(\rho v)}{\partial y} = 0$
2. $\frac{\partial(\rho u)}{\partial t} + \frac{\partial(\rho u^2)}{\partial x} + \frac{\partial(\rho uv)}{\partial y} = \rho F_x + \frac{\partial \sigma_{xx}}{\partial x} + \frac{\partial \sigma_{xy}}{\partial y}$

   $\frac{\partial(\rho v)}{\partial t} + \frac{\partial(\rho uv)}{\partial x} + \frac{\partial(\rho v^2)}{\partial y} = \rho F_y + \frac{\partial \sigma_{yx}}{\partial x} + \frac{\partial \sigma_{yy}}{\partial y}$

with F_x and F_y the components of an external force in $x-$ and $y-$ direction respectively. In this study we will assume the fluid to be incompressible. In that case we can replace

$$\begin{pmatrix} \sigma_{xx} & \sigma_{xy} \\ \sigma_{yx} & \sigma_{yy} \end{pmatrix} = \begin{pmatrix} -p & 0 \\ 0 & -p \end{pmatrix} + \mu \begin{pmatrix} 2\frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} \\ \frac{\partial u}{\partial y} + \frac{\partial v}{\partial x} & 2\frac{\partial u}{\partial y} \end{pmatrix}$$

With this definition the Navier-Stokes equations can be simplified to:

$$\begin{aligned} \frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} &= 0 \\ \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} + v\frac{\partial u}{\partial y} &= F_x - \frac{1}{\rho}\frac{\partial p}{\partial x} + \nu\left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}\right) \\ \frac{\partial v}{\partial t} + u\frac{\partial v}{\partial x} + v\frac{\partial v}{\partial y} &= F_y - \frac{1}{\rho}\frac{\partial p}{\partial y} + \nu\left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2}\right) \end{aligned}$$

with $\nu = \frac{\mu}{\rho}$, the kinematic viscosity.
In short notation:

$$\begin{aligned} \operatorname{div} \boldsymbol{u} &= 0 && (2.1) \\ \frac{\partial \boldsymbol{u}}{\partial t} + (\boldsymbol{u} \cdot \operatorname{grad})\, \boldsymbol{u} &= \boldsymbol{F} - \frac{1}{\rho} \operatorname{grad} p + \nu \operatorname{div} \operatorname{grad} \boldsymbol{u} && (2.2) \end{aligned}$$

2.2 Boundary conditions

The equations have to be completed with boundary conditions. We used two types:

2.2.1 solid boundary

The condition on a solid boundary is the *no-slip* condition: $u = 0$. This describes two things : The wall is impermeable, so the fluid cannot go through it, and due to the viscosity of the fluid it sticks to the wall.

2.2.2 free surface

At the free surface the conditions are:

$$\begin{aligned} -p + 2\mu \frac{\partial u_n}{\partial n} &= -p_0 + 2\gamma H \\ \mu\left(\frac{\partial u_n}{\partial t} + \frac{\partial u_t}{\partial n}\right) &= 0 \end{aligned}$$

where u_n is the velocity in the direction perpendicular to the free surface , and u_t is the velocity in the direction paralell to the surface. p_0 is the pressure of the surrounding gas, for example the atmospherical pressure. γ is the surface tension, and H is the mean curvature.

An equation is required for the displacement of the free surface. Suppose the position of the free surface is described by $s(x, t) = 0$, then the movement of the free surface becomes

$$\frac{Ds}{Dt} = \frac{\partial s}{\partial t} + \boldsymbol{u} \cdot \nabla s = 0$$

Chapter 3

Numerical Model

3.1 Poisson equation

If we set $\rho = 1$, and introduce $\boldsymbol{R} = -(\boldsymbol{u} \,.\, \mathrm{grad})\boldsymbol{u} + \nu \,\mathrm{div}\,\mathrm{grad}\,\boldsymbol{u} + \boldsymbol{F}$ the Navier-Stokes equations become:

$$\begin{aligned} \mathrm{div}\,\boldsymbol{u} &= 0 \\ \frac{\partial \boldsymbol{u}}{\partial t} + \mathrm{grad}\,p &= \boldsymbol{R} \end{aligned}$$

These equations are discretized with forward Euler :

$$\mathrm{div}\,\boldsymbol{u}^{n+1} = 0 \tag{3.1}$$

$$\frac{\boldsymbol{u}^{n+1} - \boldsymbol{u}^{n}}{\delta t} + \mathrm{grad}\,p^{n+1} = \boldsymbol{R}^{n} \tag{3.2}$$

$n + 1$ indicates the new time level, n the old. δt is the time step.
Equation (3.2) can be written in a different form:

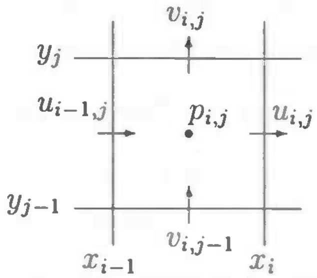
$$\boldsymbol{u}^{n+1} = \boldsymbol{u}^{n} + \delta t\, \boldsymbol{R}^{n} - \delta t\, \mathrm{grad}\,p^{n+1}$$

Substituting this in equation (3.1) gives the *Poisson equation* :

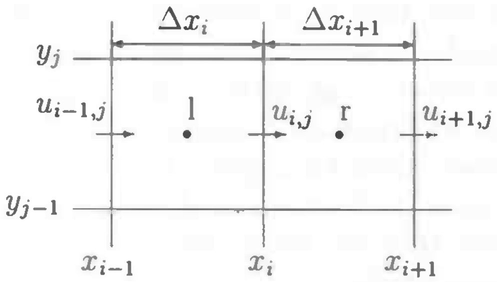
$$\mathrm{div}\,\mathrm{grad}\,p^{n+1} = \mathrm{div}\left(\frac{\boldsymbol{u}^{n}}{\delta t} + \boldsymbol{R}^{n}\right)$$

3.2 Discretization

A rectangular grid is used for the discretization, which can be stretched. The unknown variables u,v and p are placed according to the Marker-And-Cell method: the horizontal velocity u on the vertical edges of the cells, the vertical velocity v on the horizontal edges, and the pressure p in the centers.



The term $\boldsymbol{R}$ is calculated in the way as described below.



First the diffusive terms. Only the discretization of $\frac{\partial^2 u}{\partial x^2}$ is demonstrated. The other terms are calculated similarly. $\frac{\partial^2 u}{\partial x^2}$ is discretized centrally. Define:

$$\begin{aligned}\frac{\partial u_{i,j}}{\partial x}\,|_l &= \frac{u^n_{i,j}-u^n_{i-1,j}}{\Delta x_i}\\ \frac{\partial u_{i,j}}{\partial x}\,|_r &= \frac{u^n_{i+1,j}-u^n_{i,j}}{\Delta x_{i+1}}\end{aligned}$$

Then the second derivative is defined:

$$\frac{\partial^2 u^n_{i,j}}{\partial x^2} = \frac{\frac{\partial u_{i,j}}{\partial x}\,|_r - \frac{\partial u_{i,j}}{\partial x}\,|_l}{\frac{1}{2}(\Delta x_i + \Delta x_{i+1})}$$

The convective terms are discretized with a parameter α ($\alpha = 1$ is full upwind).
Again only the discretization of one term is demonstrated, $u\,\frac{\partial u}{\partial x}$, because the other terms are calculated similarly.

$$u_{i,j}^n \frac{\partial u_{i,j}^n}{\partial x} = \begin{cases} \frac{u_{i,j}^n}{(1+\alpha)\Delta x_i+(1-\alpha)\Delta x_{i+1}} \left((1-\alpha)\Delta x_{i+1} \frac{\partial u_{i,j}}{\partial x} \mid_r +(1+\alpha)\Delta x_i \frac{\partial u_{i,j}}{\partial x} \mid_l\right) & , u_{i,j}^n \geq 0 \\ \frac{u_{i,j}^n}{(1-\alpha)\Delta x_i+(1+\alpha)\Delta x_{i+1}} \left((1+\alpha)\Delta x_{i+1} \frac{\partial u_{i,j}}{\partial x} \mid_r +(1-\alpha)\Delta x_i \frac{\partial u_{i,j}}{\partial x} \mid_l\right) & , u_{i,j}^n \leq 0 \end{cases}$$

Now the right-hand side of the Poisson equation can be calculated, and the equation can be solved by an iterative method.

3.3 Iteration-MILU

For presentational reasons, let us write the Poisson equation like:

$$A\,x = b$$

where

$$A = \text{ div grad}, \; x = p^{n+1}, \; b = \text{ div}\left(\frac{\boldsymbol{u}^n}{\delta t} + R^n\right)$$

Savof96 uses MILU (Modified Incomplete LU decomposition) to solve this equation. MILU is a combination of the Conjugate Gradient method with a suitable preconditioner. This preconditioner is an important advantage above other solving methods.

The matrix A is decomposed in a lower and an upper triangular matrix, L and U respectively. Both have the same structure as the lower and upper parts of A. The products of L and U has almost the same structure as A, except for two extra diagonals. The elements of these diagonals are called 'fill-in elements'. If these elements are ignored, then it is possible to find L and U whose product equals A. Those ignored fill-in elements can result in unreliable solutions when they are rather big. To compensate this effect the fill-in is subtracted from the diagonal entries.

With this preconditioner (called K) the algorithm used for the conjugate gradient method becomes:

1. Take $\boldsymbol{x}^{(0)} = \boldsymbol{p}$, and calculate $\boldsymbol{r}^{(0)} = \boldsymbol{b} - A\boldsymbol{x}^{(0)}$ and $\boldsymbol{z}^{(0)} = K^{-1}\boldsymbol{r}^{(0)}$

   Compute for $n = 0, 1, 2, \cdots$ the vectors $\boldsymbol{x}^{(n+1)}, \boldsymbol{r}^{(n+1)}and\boldsymbol{z}^{(n+1)}$ from

2. $\boldsymbol{x}^{(n+1)} = \boldsymbol{x}^{(n)} + \alpha_n \boldsymbol{z}^{(n)}$ with $\alpha_n = \frac{(\boldsymbol{r}^{(n)}, K^{-1}\boldsymbol{r}^{(n)})}{(\boldsymbol{z}^{(n)}, A\boldsymbol{z}^{(n)})}$

3. $\boldsymbol{r}^{(n+1)} = \boldsymbol{r}^{(n)} - \alpha_n A\boldsymbol{z}^{(n)}$

4. $\boldsymbol{z}^{(n+1)} = K^{-1}\boldsymbol{r}^{(n+1)} + \beta_n \boldsymbol{z}^{(n)}$ with $\beta_n = \frac{(\boldsymbol{r}^{(n+1)}, K^{-1}\boldsymbol{r}^{(n+1)})}{(\boldsymbol{r}^{(n)}, K^{-1}\boldsymbol{r}^{(n)})}$

This algorithm gives us p^{n+1}. The new velocity is calculated using
$u_{i,j}^{n+1} = u_{i,j}^n + \delta t R_{i,j}^n - \delta t$ grad $p_{i,j}^{n+1}$, where the gradient is discretized as:

$$\frac{p_{i+1,j}^{n+1} - p_{i,j}^{n+1}}{\frac{1}{2}\Delta x_i + \frac{1}{2}\Delta x_{i+1}}$$

Remark : For use on a parallel vector computers also a SOR solver is available, but we have not used it in the current investigation.

3.4 Free surface

A means of keeping track of the free surface of a fluid during calculation is necessary to know where to apply the momentum equation. Evidently, this only has to be done in cells that contain fluid. In Savof96, use is made of an *indicator function*. This is a function $F(i,j)$ that represents the part of cell (i,j) that is filled with fluid, for example F=1 if the cell is full (obviously $0 \leq F(i,j) \leq 1$). Besides the indicator function, there is the cell labeling, which gives more qualitative information about a cell. $NF(i,j)$ is a two-dimensional array in which this information for cell (i,j) is stored. It can have the following values:

- 0: full cell
- 1: surface cell with full cell to the left
- 2: surface cell with full cell to the right
- 3: surface cell with full cell at the bottom
- 4: surface cell with full cell at the top
- 5: degenerated cell
- 6: empty cell
- 7: 'outflow' cell
- 8: 'inflow' cell
- 9: obstacle or boundary cell

The cells with labels 7, 8 and 9 have constant labels, but the other labels can change every time step. First the cells are determined with no fluid, and they get the label 6. Then the cells that have an empty neighbour cell are labeled as surface cells, with a value between 1 and 5. All the other cells get the label 0, because they have no empty neighbour cells. Using the above labeling, the position where the momentum equations must be applied can be identified: at cell faces between full and/or surface and/or outflow cells. At faces between surface and empty cells boundary conditions are applied, to be described in the next section.

3.5 Boundary conditions

In this section the numerical treatment of the boundary conditions described in the previous chapter is shown:

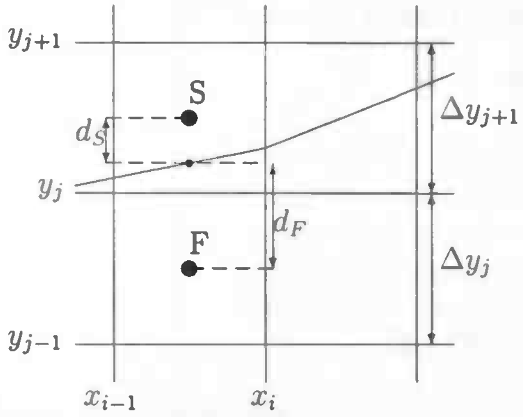
3.5.1 solid boundary

The no-slip condition prescribes $u = v = 0$, but since the velocities are not always defined on the solid boundary because of the placement of the variables, we have to interpolate and make use of virtual mirror points.
For example, consider the lower boundary ($i = 1$). The horizontal velocity u is not defined *on* the wall, but half a grid point higher. So we set $u(0,j) = -u(1,j)$, so that after interpolation of these two velocities the resulting velocity *on* the wall is zero.

3.5.2 free surface

The free surface conditions are:

$$
\begin{aligned}
-p + 2\mu\frac{\partial u_n}{\partial n} &= -p_0 + 2\gamma H \qquad (3.3)\\
\mu(\frac{\partial u_n}{\partial t} + \frac{\partial u_t}{\partial n}) &= 0 \qquad (3.4)
\end{aligned}
$$



The pressure at the free surface needed in equation (3.3) is interpolated as follows. Suppose we have a full cell (with pressure p_F) with above it a surface cell (with pressure p_S). Then we can estimate the average height of the surface to be
$F_{i,j}\Delta y_j + F_{i,j+1}\Delta y_{j+1}$, so the distance from the center of the upper cell to the surface is $d_S := (1 - F_{i,j})\Delta y_j + (\frac{1}{2} - F_{i,j+1})\Delta y_{j+1}$, and the distance from the center of the lower cell to the surface is $d_F := (F_{i,j} - \frac{1}{2})\Delta y_j + F_{i,j+1}\Delta y_{j+1}$. Since the distance between the centers of the cells is $d := \frac{1}{2}\Delta y_j + \frac{1}{2}\Delta y_{j+1}$, we can now linearly interpolate the pressures in the two cells to obtain the pressure at the free surface:

$$p_f := \frac{d_S\, p_F + d_F\, p_S}{d}$$

The above formulation gives an accurate description of the pressure condition; it should be used in situations where details near the surface are relevant for the global flow dynamics (such as when the surface tension is relevant). When the free-surface details are not important, it is possible to simplify the above treatment to $p_S = -p_0 + 2\gamma H$.

In the present investigation we have followed the latter formulation. With reference to the use of MILU as a Poisson solver it has the advantage of yielding a symmetric iteration matrix.

Because the momentum equations have to be calculated in between surface cells as well, we need to establish extra velocities on those cell faces. To overcome this problem, we apply the continuity equation div $\boldsymbol{u} = 0$ in surface cells as well. If we have three velocities available, then this procedure gives the fourth. If less velocities are available, we take e.g. $\frac{\partial u}{\partial x} = \frac{\partial v}{\partial y} = 0$. The remaining velocities required, lying just outside the surface cell layer, are calculated using equation (3.4).

3.6 Adjusting δt

When the fluid is moving very wild, a very small time step is needed to retain stability of the proces, but during calm intervals a greater time step is permitted. A temporarily larger time step can of course mean a significant reduction of computation time.

These considerations are quantified in the *CFL-number*: The Courant-Friedrichs-Levy number, defined as:

$$CFL = \max\left(\frac{|u|\delta t}{h_x}, \frac{|v|\delta t}{h_y}\right)$$

where h_x and h_y are the distances between two grid points.

Fourier analysis shows that this number must not exceed 1. This has a physical explanation: If the CFL-number is less then 1, the fluid will not be moved over more then one cell in a time step.

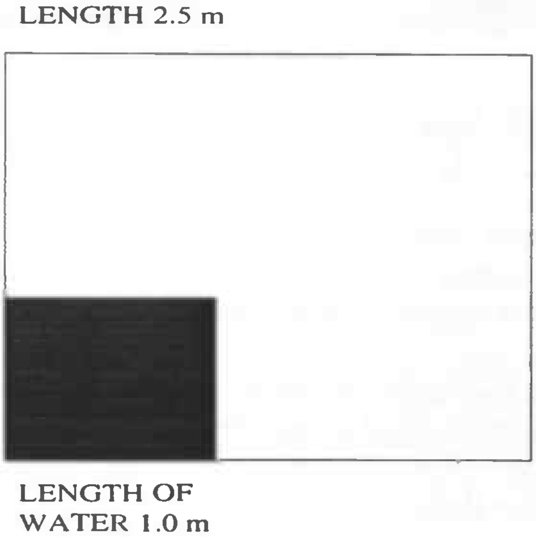
In the program, the CFL-number is calculated by taking the maximum of the number over all cells. The time-step is doubled if the CFL-number is small enough for a few consecutive cycles. If the CFL-number is greater then a user-specified constant, the time step is halved immediately. The constant should of course be less then 1.

Chapter 4

Results

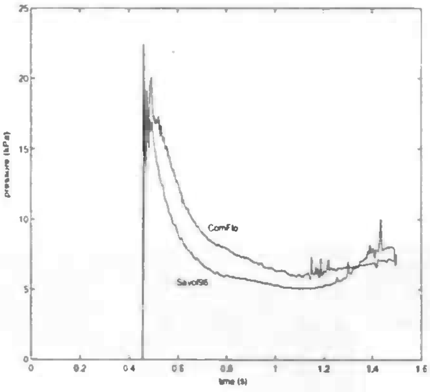
4.1 Dambreak Sabeur

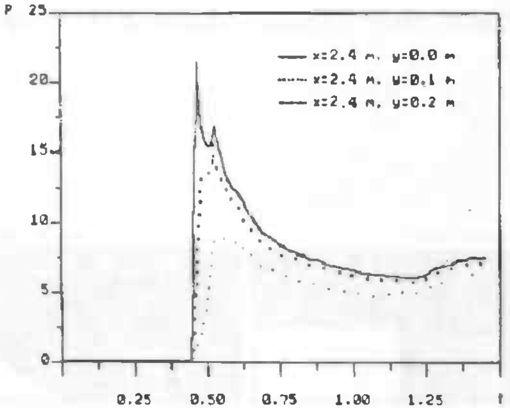
The first problem we studied was a dambreak, which Z.A.Sabeur also studied in 1995. The initial state is shown in the next figure.



At t=0 the water is released, a dambreak, and the pressure is measured in the lower right corner.
A grid of 100 points in $x-$ and $y-$direction was used. The computation time was about 15 minutes on a workstation with an average time step of about 1.3 10^{-3} seconds.

The goal was to compare the pressure in a qualitative way, i.e. to investigate whether the shape of the curves were similar. G. Fekken did the same computation with ComFlo, which is a 3D program. The pressures computed by Savof96 and ComFlo are plotted in the same figure, and in the adjacent figure the solution of Sabeur is shown.

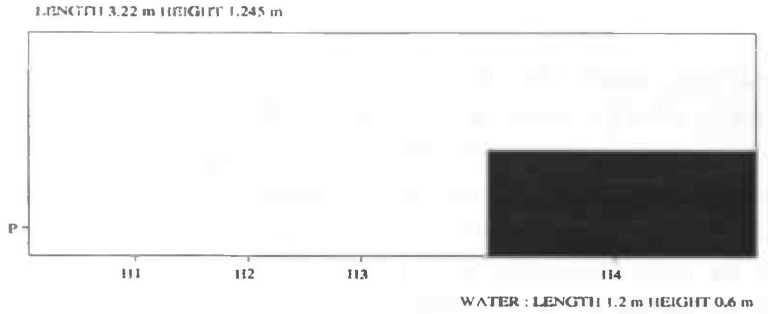




Although there are some differences, the results are quite similar. The moment of impact is the same, although the height of the first peak is a little different, but that is not so strange since all the impact happens in one time step, and this time step does not have to be the same in the three programs used.

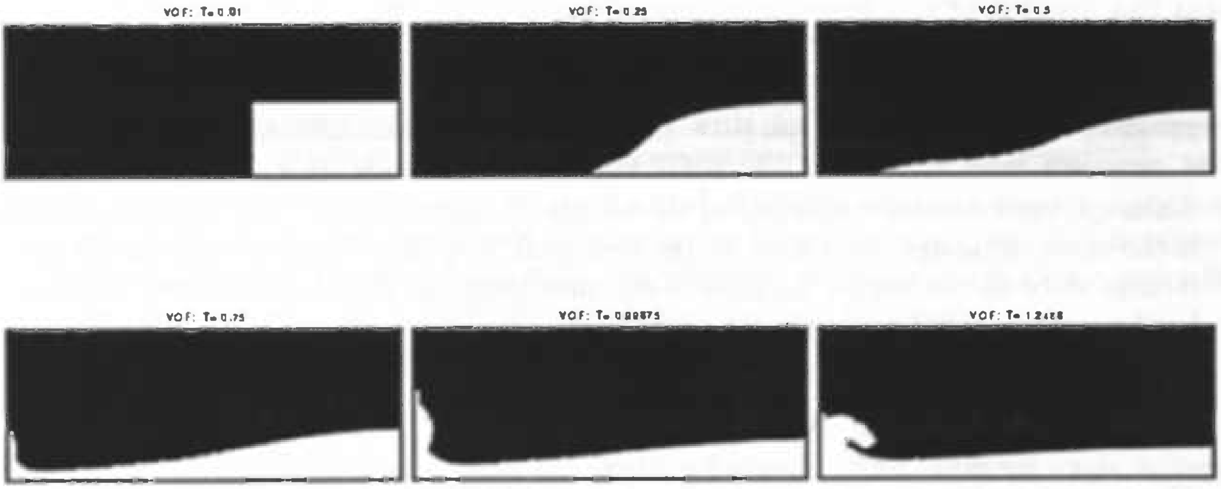
4.2 Dambreak MARIN

The second problem was also a dambreak, and MARIN had data available from experiments. We used a grid of 120x60 cells, and the total time of the computation is 4.5 seconds. The computation time was about 15 minutes on a workstation with an average time step of about 1.3 10^{-3} seconds.


LENGTH 3.22 m HEIGHT 1.245 m
P
H1
H2
H3
H4
WATER : LENGTH 1.2 m HEIGHT 0.6 m

We compared the pressure in a point on the left wall at 16 cm height, and the water heights at four points (H1-H4) (see the figure above) with the results of MARIN.

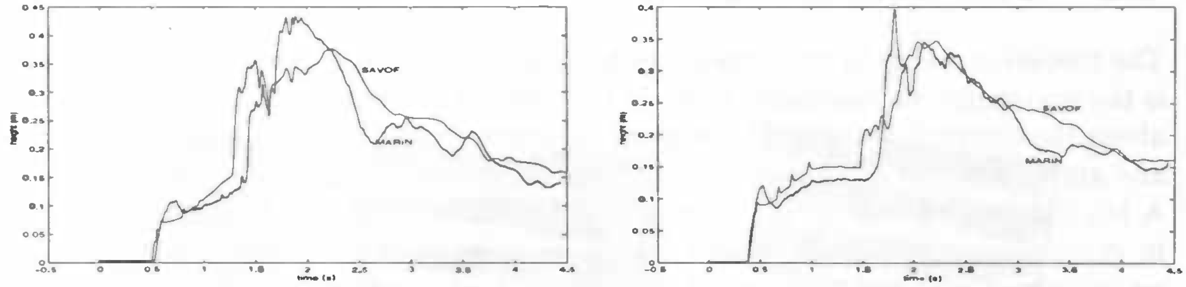
The next figure shows some snapshots starting at t=0, and every picture is 0.25 seconds later.



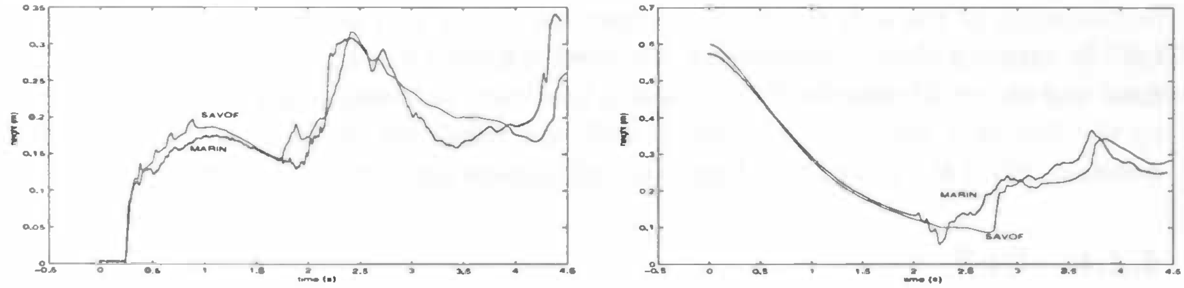
4.2.1 Water heights

The computed water heights look very much like the measured heights. The results are best until the point after the impact, where a wave, moving from the left wall to the right, reaches the point. In the figure this can be seen when the water level rises again: H1 at about 1.3 seconds, H2 at about 1.5 seconds, H3 at about 1.7 seconds, and H4 at about 2.5 seconds. This is not so strange since the water behaves very wild at impact, and little differences easily arise. However we only compared with one result of MARIN, and they did many more experiments, which all show small differences of about the same size as in the figure between Savof96 and MARIN. Further it seems that the water level at the experiment wasn't exactly 60 cm. This can be seen in the figure showing the water height at H4.

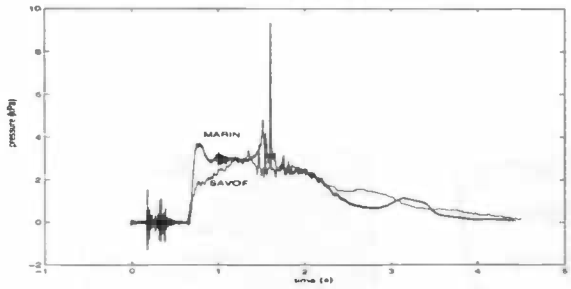
water height at H1,H2



Water height at H3,H4



4.2.2 The pressure



The pressure looks a lot like the pressure of the experiment, but there is one big difference. When the water hits the left wall at about $t = 0.5$, the pressure is different. There is a peak in the experiment, which does not appear in the computed pressure.

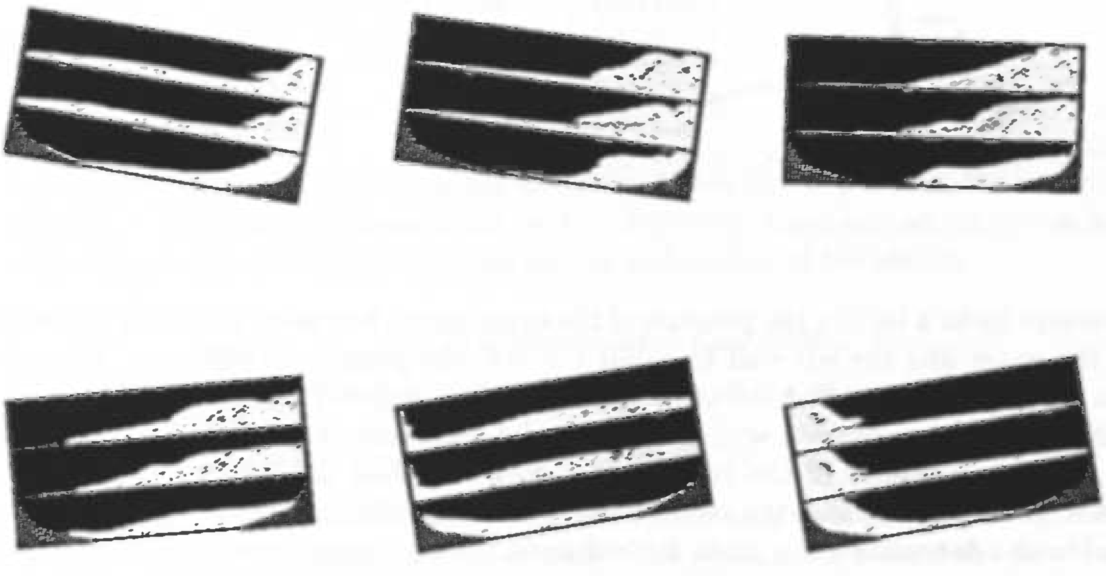
MARIN also did experiments with a small model dambreak, to test if there were scale effects, and the pressure of the small model does not show a peak at the moment of impact. G. Fekken computed the same problem with ComFlo, and the computed pressure of ComFlo also does not show a peak. His refinement study showed no improvement, so this difference between numerical simulation and experiment is not caused by the coarseness of the grid.

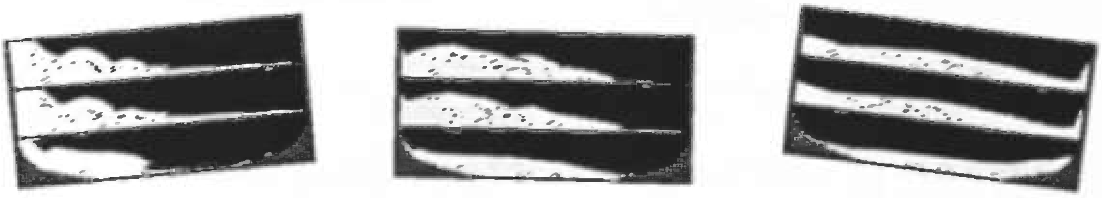
4.3 The anti-roll tank

The motion of a ship in two dimensions has three components: sway, heave and roll. Sway is the horizontal displacement, heave is the vertical displacement, and roll is the rotation about the centre of gravity. The motions are taken to oscillate with a prescribed frequency and amplitude. We computed the response of the wave amplitude to sway and roll
A big difference is that in the calculations the motion of the ship is prescribed, and that in experiments the motion follows from interaction with the water. For the roll motion we computed with the geometry of the anti-roll tank described in appendix A.4.1, and for the sway motion we used a rectangular geometry with the shape of one of the two upper rectangulars of the anti roll tank, because the upper two showed the most response, and are the same in these two motions. We used a grid of 62x62, and the average calculation time was about 30 minutes. Sometimes it was more and sometimes less, but that is caused by the fact that when the response is high, the velocities are high, and the time step will decrease. So if the response is high, the calculation time will be longer.

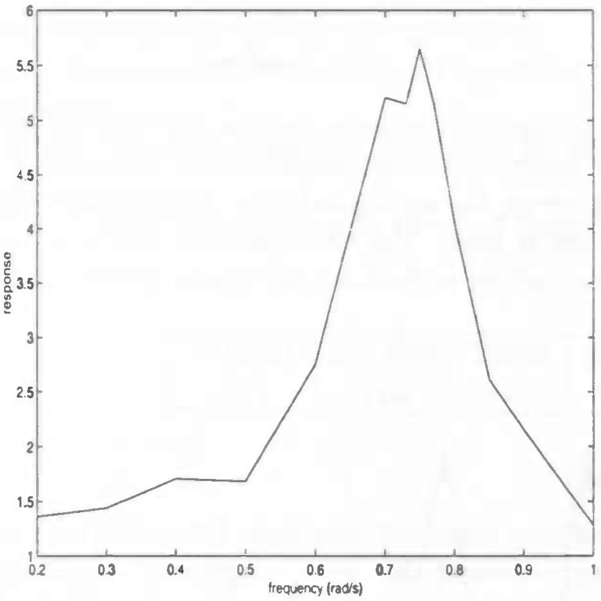
4.3.1 Roll

The roll motion is performed around the centre of gravity. The maximal angle is kept constant for all frequencies at about 10 degrees. Some snapshots are shown below for the roll motion with frequency 0.7.



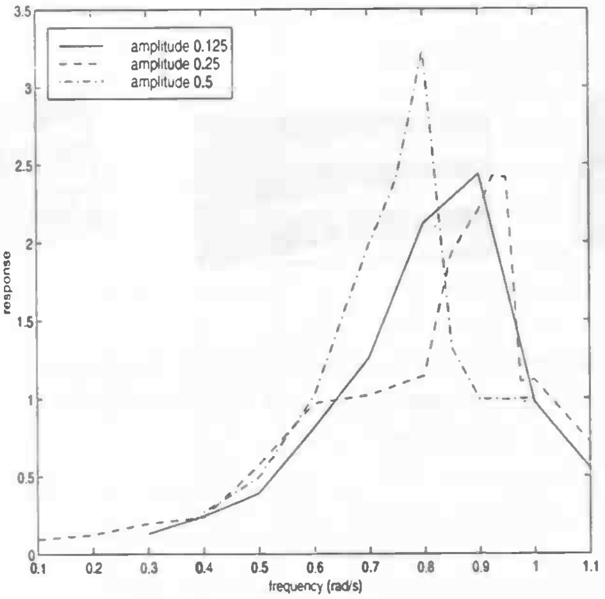


The response is plotted to the roll motion for some frequencies. We used a grid of 82x82. The frequency is on the horizontal axis, and the amplitude of the water wave is on the vertical axis.



In the picture it can be seen that there is a high response for the frequencies around 0.8.

4.3.2 Sway



Here the response is plotted to the sway motion for different amplitudes. The amplitude of the water wave is scaled with the amplitude of the motion, and put on the vertical axis. The frequencies are on the horizontal axis. The picture shows that at the frequencies around 0.8 the response is high. For the frequency 0.8, with amplitude 0.5, calculations were performed on finer grids, to look for influences of the coarseness of the grid:

grid	62x62	82x82	102x102	122x122
scaled ampl.	1.14	1.11	1.17	1.24

4.3.3 Damping

For the roll motion we have regarded damping. When the ship moves, the water inside the ship exerts a momentum around the center of gravity. Damping occurs if this is in phase with the velocity. A simplified model for the roll motion of the ship is:

$$(I + a_{\phi\phi})\ddot{\phi} + b_{\phi\phi}\dot{\phi} + c_{\phi\phi}\phi = M \tag{4.1}$$

In Savof96 the roll-motion is prescribed: $\phi = \phi_a cos\, \omega t$, so we can write M as:

$$M = (-\omega^2(I + a_{\phi\phi}) + c_{\phi\phi})\phi_a cos\, \omega t - b_{\phi\phi}\omega\phi_a sin\, \omega t$$

A force in $x-$ and $y-$ direction is calculated that results from integrating the pressure over the surface of obstacles and outer walls. The momentum around the centre of gravity $(x_0, y_0) = (0, 4.47)$ is obtained from this force :

$$\begin{aligned} m &= \textstyle\sum_{i,j} \boldsymbol{F} \times r \\ &= \textstyle\sum_{i,j} f_i(y_j - y_0) - f_j(x_i - x_0) \end{aligned}$$

where the sum is taken over i and j for which (x_i, y_j) is on the surface of an obstcale or outer wall. f_i and f_j are the forces in $x-$ and $y-$ direction respectively, $\boldsymbol{F} = (f_i, f_j, 0)$, and $r = (x_i - x_0, y_j - y_0, 0)$. This momentum m can be decomposed with a Discrete Fourier Transformation (DFT) :

$$m = a_0 + \sum_{n=1}^{N} a_n \cos(\omega nt) + b_n \sin(\omega nt)$$

For a vector m with length N, the DFT is a length N vector X, with elements

$$X(k) = \sum_{n=1}^{N} m(n) * e^{-i2\pi(k-1)\frac{n-1}{N}}$$

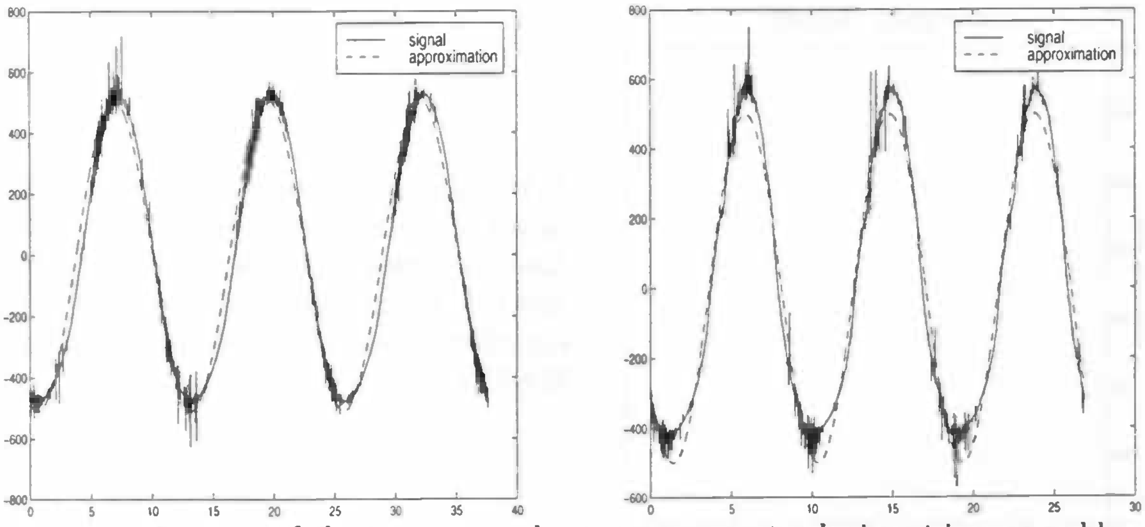
The coefficients a_i and b_i can be calculated :

$$a_0 = \frac{2\,X(1)}{N}, \; a_i = \frac{2\,RE(X(i+1))}{N}, \; b_i = \frac{2\,IM(X(i+1))}{N}$$

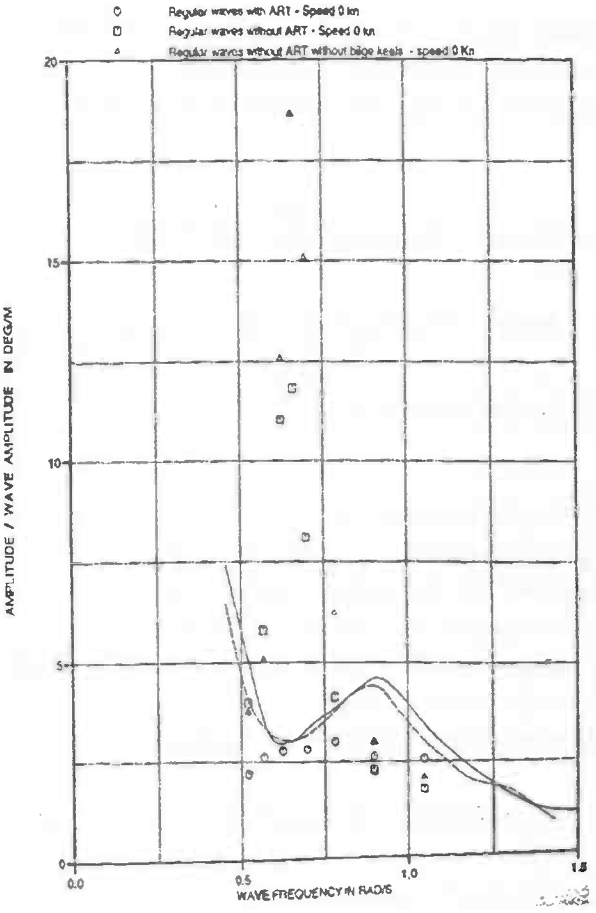
To calculate the damping factor, only use is made of a_1, and b_1. The other terms are neglected, and presumed small.
If we take $m = a_1 \cos\, \omega t + b_1 \sin\, \omega t$,the following must hold for a_1 and b_1:

$$a_1 = (-\omega^2(I + a_{\phi\phi}) + c_{\phi\phi})\phi_a \text{ and } b_1 = b_{\phi\phi}\omega\phi_a$$

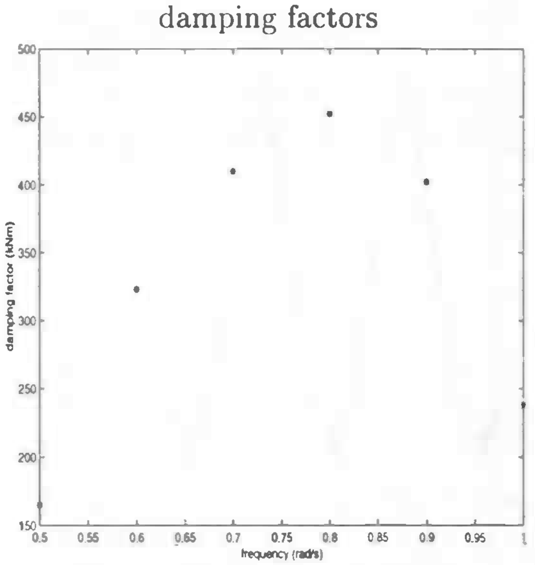
b_1 is the damping factor.



The approximation of the momentum by $m = a_1 \cos\, \omega t + b_1 \sin\, \omega t$ is reasonable, which is shown in the above figures. There the momentum is plotted for three periods for the motion with $w = 0.5$ in the left, and $w = 0.7$ in the right figure, and the approximation is plotted in the same figure.



MARIN did tests with the anti-roll tank, and the results are shown in the adjacent figure. The response of the ship was tested with and without an anti roll tank. The ship has a high response around the frequency 0.7 without an anti roll tank, and with the anti roll tank the reponse is much lower, so then a lot of damping occurs. The damping of the anti roll tank is only important for the frequencies around 0.7, because for other frequencies the response of the ship is low, so the movement does not need damping. In the previous subsections we have seen that there is indeed a high response for these frequencies, so now we have to calculate the damping factor.



In the adjacent figure the calculated damping factor is plotted for some frequencies. The damping factor is large for the frequencies 0.7 and 0.8, which corresponds with the damping effect of the anti-roll tank in the tests of MARIN.

The constants in equation (4.1) are presumed to be constant for frequencies close to 0.7, and can be estimated from the calculations. With these constants a damping factor ν_{art} can be calculated: $\nu_{art} = \frac{b_{\phi\phi}}{2\sqrt{(I+a_{\phi\phi})c_{\phi\phi}}} = 0.36$. This can be compared with a factor ν calculated from the simulations of MARIN, $\nu = 0.21$. This factor is obtained from the ship without an anti roll tank, and the fraction of the measured and the calculated should give the reduction of the roll motion. This means that the roll motion should be reduced by a factor 0.58. In the picture of the tests of MARIN it can be seen that the motion at the frequency 0.7 is reduced by about a factor 0.3. A cause for the difference is that the calculation of ν is rather inaccurate, because two close numbers are subtracted. Also the calculation of ν_{art} depends on some assumptions, so little differences easily arise.

Chapter 5

Conclusions

As we have seen in the last chapter, the motion of fluid in three dimensions, with everything the same in the third direction can well be described by Savof96. The differences are small, when there is no motion of the geometry. Bigger differences arise when the motion is introduced, but an important reason is that the motion is prescribed in Savof96, and in experiments the motion is influenced by interaction with the fluid inside. The approximation of the anti roll tank is reasonable, and could easily be used to test the effect of changes in the geometry.

A big advantage is that the calculation times are much smaller with a two dimensional program. In order to find out if the differences are caused by numerical noise or differences in modelling the problem, some points of interest in the near future could be:

- Dynamical Interaction: The fluid inside the geometry exerts a force on it, which influences the motion.
- Two mediums: This means adding another medium, for example air, to solve the problems with air bubbles in the fluid.

Appendix A

Program description

This appendix contains information about the calling sequence, important variables, subroutines, inputfile, and outputfiles of Savof96.

A.1 Calling sequence

This is the main calling sequence, excluding postprocessing routines:

```
initialisation  SETPAR   GRID
                SETFLD   SURDEF
                         MKGEOM
                         RDGRID
                         LDSTAT
                         LABEL
                         BC          BCBND
time step       INIT     BCBND
                PETCAL
                TILDE    BDYFRC
                SOLVEP   COEFF
              ------------------------------
                1        CONVRT
                         PRESIT      SLAG
                                     SVSTAT
              ------------------------------
                2        MILU
              ------------------------------
                         BC          BCBND
                CFLCHK
                DTADJ
                VFCONV   LABEL
                PRT
                SVSTAT
```

where 1 and 2 denotes a choice between SOR- and MILU-iteration.
The postprocessing files are omited for presentational reasons. The subroutine PRT is invoked every **prtdt** time units, to be specified in the input file.. PRT calls all the desired subroutines that produce output. The subroutines are discussed in appendix A.2.

A.2 Subroutines

In this section a short description is given of every subroutine:

AVS	: generates output to be processed by AVS.
BC	: sets boundary conditions for velocity components.
BCBND	: boundary conditions at outer boundary.
BDYFRC	: computes the apparent body force.
CFLCHK	: monitors CFL number and sets flag for time-step adjustment.
COEFF	: defines coefficient matrix for Poisson equation including boundary conditions at wall and free surface.
CONVRT	: converts 2-dim datastructure into 1-dim datastructure for more efficient implementation of pressure solver.
DTADJ	: halves/doubles time step (old time step will be repeated).
EXTRA	: saves velocities in a specified region.
FILOUT	: determines and saves the fill ratios of specified areas.
FLXOUT	: determines and saves the fluxes through specified areas.
FRCOUT	: computes force by integrating the pressure on the walls in both x- and y-direction, and computes the momentum around a point.
GNUPLT	: writes data to a pipe to be displayed 'live' by Gnuplot.
GRID	: makes (non-uniform) grid.
INIT	: starts new time step.
LABEL	: labels empty, surface (preliminary) and full cells.
LDSTAT	: reads restart file (produced by SVSTAT).
MILU	: solves Poisson equation through MILU algorithm.
MKGEOM	: defines the geometry.
MKSTRL	: keeps track of specified particles by integrating velocities.
MPTOUT	: interpolates velocities in points specified in the geometry file.
PETCAL	: labels surface cells and computes pressure at free surface.
PRESIT	: solves Poisson equation and controls SOR relaxation factor.
PRT	: prints and writes results.
PRTCNF	: saves the velocities, the pressure, and the VOF-function.
PRTFLD	: produces a sequence of characters representing the entire geometry including obstacles (#=obstacle, *=fluid, I=inflow and 0=outflow).

PRSOUT : integrates the pressure over a given circle, and writes to file.

RDGRID : reads 'broidary' file defining geometry.

SETFLD : initializes fluid configuration.

SETPAR : reads input file.

SLAG : performs one SOR sweep. It uses a one-dimensional implementation.

SOLVEP : organizes pressure calculation and updates velocity.

SURDEF : generates liquid configuration.

SVSTAT : writes restart file.

TILDE : integrates momentum equations.

VFCONV : moves fluid, i.e. adjusts VOF-function, and re-labels.

A.3 Common block variables

In Fortran global variables are defined in common blocks. The most important variables are listed here.

/CASE/ contains external body forces and their controls:

Omega : rotation rate.

DelOme : initial rotation relative to Omega.

XO, YO : point about which the rotation takes place.

TwUp, TwDown : time to start and end rotation.

UO, VO : initial velocities in x- and y-direction respectively.

Ampl,Freq : amplitude and frequency of the oscillation.

AAngle : angle under which the oscillation takes place.

GX, GY : acceleration in x- and y-direction respectively.

TxOn, TxOff,
TyOn, TyOff : times to turn accelerations in x- and y-direction on and off.

/COEFP/ contains the coefficients for the pressure in the Poisson equation:

CC(I,J) : coefficient of $p_{i,j}$.

CN(I,J) : coefficient of $p_{i,j+1}$.

CS(I,J) : coefficient of $p_{i,j-1}$.

CE(I,J) : coefficient of $p_{i+1,j}$.

CW(I,J) : coefficient of $p_{i-1,j}$.

DIV(I,J) : right-hand side of the Poisson equation.

/GRIDAR/ contains parameters involving gridsize:

X(I)	:	x-coordinates of the grid points.
XI(I)	:	x-coordinates of the cell centers.
DelX(I)	:	distance in x-direction between two subsequent grid points.
Y(J)	:	y-coordinates of the grid points.
YJ(J)	:	y-coordinates of the cell centers.
DelY(J)	:	distance in y-direction between two subsequent grid points.
RX(I),RXI(I), RDX(I)	:	inverse of X(I), XI(I) and DELX(I) respectively.
RDY(J)	:	inverse of DELY(J).
CYL,ICYL	:	floating-point and integer version of the 2D/axisymmetric switch. (CYL=0 for 2D and CYL=1 for axisymmetric geometries)
IMaxUs,JMaxUs	:	number of grid points in x- and y-direction.
IM1Us,JM1Us	:	IM1Us=IMaxUs-1, JM1Us=JMaxUs-1.
IM2Us,JM2Us	:	IM2Us=IMaxUs-2, JM2Us=JMaxUs-2.

/ORGA/ contains cell labeling information.

NF(I,J)	:	cell labels for the current time level.
NFN(I,J)	:	cell labels for the previous time level.
PETA(I,J)	:	coefficient used to interpolate the pressure.

/PHYS/ contains pressure and velocities at the current and previous time level. An N means 'at the previous time level'.

F(I,J),FN(I,J)	:	VOF/indicator function.
U(I,J),UN(I,J)	:	horizontal velocity.
V(I,J),VN(I,J)	:	vertical velocity.
P(I,J),PN(I,J)	:	pressure.
PS(I,J)	:	pressure at the free surface.

/TIMES/ contains parameters related to time levels and steps.

Cycle	:	time step number.
T	:	current time.
DelT	:	time step.
DelTMx	:	maximum allowed timestep.
TStart	:	start time.
TFin	:	end time.

A.4 Pre- and Postprocessing

A.4.1 Input files

Savof96 uses at least one input file, the main input file **savof96-2.in**:

```
SAVOF96-2.0
***** tank geometry *************************************************
icyl     Xmin      Xmax     Ymin     Ymax    SpGeom  SpMot
  0     -5.72      5.72      0.0     5.72       1     0

***** liquid configuration ******************************************
LiqCnf    xp    yp     r     xq     yq
  3      -5.72  0.0    2    5.72   0.64
         -5.72  1.96        5.72   2.60
         -5.72  3.86        5.72   4.50

***** grid definition ***********************************************
iMaxUs   jMaxUs   cx    cy    Xpos  Ypos
    82      82    0.0   0.0   0.0   0.0

***** liquid properties *********************************************
Sigma    CAngle     Nu
72e-6      90.0    1e-5

***** body forces and external motion: 2D ***************************
Gx   TxOn      TxOff      u0       Gy    TyOn    TyOff    v0
0.0      0.0      1000   0.0     -9.81     0.0    100.0   0.0
Ampl Freq   Angle
0.0   0.7    0.0
Rpm   DelOme  TwUp    TwDown     x0     y0
1.1667  0.0     0.0    100.0     0.0    5.34

***** body forces and external motion: axisymmetric *****************
     Gy    TyOn    TyOff    v0    Ampl    Freq
 0.0     0.0   100.0    0     0.0     0.0
 Rpm   DelOme     TwUp TwDown
 0.0     0.0    0.0   0.0

***** boundary conditions and inflow characteristics ****************
left    right   top   bottom   UIn  VIn   Freqin  IPIN  PIn
   2        2      2       2   0.0  0.0     0.0   0     0
```

```
***** upwind parameter and Poisson iteration parameters *************
Alpha      Epsi     ItMax   OmStrt  IMilu
  1.0    3.0e-5       50      1.9     1

***** time step and restart control *********************************
TFin    DelT    CFLMax    PrtDt     svst   svdt
 10     0.01      0.1      0.1       1     50.0

***** print / plot control ******************************************
 gnu    avs   uvpf  velop   height  force press
  1     0      1      0        0      1     0

***** stream lines **************************************************
nrx   nry   nrdt    xps     yps     xqs     yqs      t   dt/delt
  0     0      0    -1.0    0.0     1.0     1.0     0.0     1e3

***** fluxes ********************************************************
number of fluxes to be printed
    0
   p1     p2     p3     hor

***** fill ratios ***************************************************
number of ratios to be printed
   2
  xpf    ypf    xqf    yqf
 -0.5    1.91   0.5    3.76
 -0.5    3.86   0.5    5.72
```

The input file will be explained following the example.

tank geometry

iCyl is the switch between 2D and axisymmetric geometries. 0 for two dimensional, and 1 for axisymmetric calculations. (Xmin,Ymin) and (Xmax,Ymax) are the lower left and the upper right corner of the tank respectively. If a special geometry is used, SpGeom should be greater then 0, and if a special motion is desired SpMot should be 1.

liquid configuration

here the initial configuration of the fluid is specified. The meaning of xp,yp,r,xq and yq depend on LiqCnf:

`LiqCnf`

0 : no fluid whatsoever.
1 : lower part filled with surface shaped as a semi-circle at average height yp.
2 : drop with center `(xp,yp)` and radius `r`.
3 : `r+1` rectangles with lower left point `(xp,yp)` and upper right point `(xq,yq)`.
4 : drop with radius `r` falling along vertical axis at y =yp into a pool of depth yq.
5 : fluid filament with width `r` and height yp including a semisphere.
6 : soliton with amplitude yq on a pool of depth yp.

grid definition

`iMaxUs` and `jMaxUs` are the number of grid points in x- and y-direction respectively. `cx` and `cy` are the stretch parameters. if zero, no stretching is performed. Positive values result in smaller cells near x =Xpos and y =Ypos respectively, and negative values in smaller cells near the outer walls.

liquid properties

`Sigma` specifies the kinematic surface tension $\frac{\sigma}{\rho}$, `CAngle` the contact angle, and `nu` the kinematic viscosity $\frac{\nu}{\rho}$.

body forces and external motion

These parameters are all described in the common block `/CASE/` in section A.3.

boundary conditions and inflow characteristics

For every side the desired treatment can be stated here. Values `1` or `2` set the boundary conditions *slip* or *no-slip* respectively and values `7` or `8` make the entire side an opening for out- or inflow respectively. The inflow velocity, which is obtained from `UIn` and `VIn`, oscillates with frequency `Freqin`. IPIN states the side where the pressure condition is applied. The pressure is set to `PIn`.

upwind parameter and Poisson iteration parameters

`Alpha` is the upwind parameter described on page 7. With `Epsi` the Poisson convergence criterion is controlled. `ItMax` is the maximum number of iterations allowed in one time step. If SOR-iteration is used `OmStrt` is the initial relaxation factor. `IMilu` is the switch between SOR- and MILU-iteration. `0` means SOR-iteration, and any other value means MILU-iteration.

time step and restart control

`TFin` is the end time, and `DelT` is the initial time step, which will be halved if the CFL number exceeds `CFLMax`, or doubled if it is less then 0.4*`CFLMax` during a few consecutive

time steps. Every `PrtDt` time units a printout is given on the screen, and the subroutines that produce output are called. Every 20*`PrtDt` time units a large printout is given on the screen.
`svst` can have three values: `0` if no restart backups are required, `1` to save the program state every `svdt` time units and `2` if a saved state from a previous run is to be read at startup, and after that the state of the program should be saved every `svdt` time units.

print / plot control

All parameters here are switches used to obtain the desired output. `1` enables, and `0` disables the respective output option. The switches are explained below:

`gnu`	:	live Gnu-plot animation of liquid configuration.
`avs`	:	velocity and pressure data in AVS-format.
`uvpf`	:	velocity, pressure and VOF-function data.
`velop`	:	additional velocity and pressure output.
`height`	:	height of the free surface
`force`	:	force integrated over the solid boundaries, and the momentum about a given point.
`press`	:	pressure data in a specified point.

stream lines

In the box defined by `(xps,yps)` and `(xqs,yqs)` as lower-left and upper-right corner respectively, `nrx*nry` particles are placed: `nrx` in x-direction, and `nry` in y-direction. These particles are followed, starting at t =`t`, and after `dt/delt` time steps 'new' particles are released until the number of releases has reached `nrdt`. If `nrx` ,`nry` or `nrdt` is zero, no stream lines are created.

fluxes

First the number of fluxes to be measured has to be specified. Then for every flux a line needs to be added to the input file specifying the location to record the flux. Those locations are defined by horizontal or vertical lines. The line is defined by the first three co-ordinates, and whether the line is horizontal or vertical depends on the value of `hor`. If `hor=1` the line has startpoint `(p1,p3)` and endpoint `(p2,p3)` (a horizontal line), and if `hor=2` these coordinates are `(p3,p1)` and `(p3,p2)` respectively (a vertical line).

fill ratios

First also the number of areas, of which the fill ratio is to be monitored, has to be stated. Each rectangular area is determined by a line with four parameters xpf,ypf,xqf,yqf: (xpf,ypf) is the lower-left corner, and (xqf,yqf) is the upper-right corner.
It is important to note that all parameters must be specified, and can not be omitted. Also no extra lines should appear before the end of the last input section, because Savof96 expects them at a constant place. This concludes the description of the main input file.

To create a special geometry, another file can be used : **savof96.geo**.

```
ANTI ROLL TANK
Icyl    Xmin     Xmax     Ymin     Ymax
  0    -5.72     5.72     0.0      5.72
arcs    lines    inflows    outflows
   2       8          0           0
arcs:
   xm    ym      r    teta1 teta2 kant links rechts boven onder meetpntn
  4.21  1.91   1.51   280   360   1      0      1     0     1      0
 -4.21  1.91   1.51   180   260   1      1      0     0     1      0
lines:
   x1     y1     x2     y2     kant links rechts boven onder   meetpntn
 -5.72  5.72   5.72   5.72    1      1      1      1     0       0
 -5.72  3.76   5.72   3.86    2      1      1      0     0       0
 -5.72  1.86   5.72   1.96    2      1      1      0     0       0
 -2.89  0.0    2.89   0.0     0      0      0      0     1       0
 -4.75  0.49  -2.89   0.0     0      0      0      0     1       0
  2.89  0.0    4.75   0.49    0      0      0      0     1       0
 -5.72  1.91  -5.72   5.72    0      1      0      1     1       0
  5.72  1.91   5.72   5.72    1      0      1      1     1       0
inflows:
   p1      p2   side (left=1,right=2,top=3,bottom=4)
uitstromen:
   p1      p2   side (left=1,right=2,top=3,bottom=4)
```

This is an example of an Anti roll tank. On the first line, a name for the geometry can be stated. Then the next five parameters have the same meaning as in Savof96-2.in, and should be the same. In the next line, the number of each object is specified. Then the objects are defined per class:

arcs

A circular arc is defined by its center (xm,ym), which does not have to lie inside the space of the geometry, radius r, start angle teta1 and end angle teta2. The parameter side determines where the obstacle cells are put: 0 results in obstacle cells in the inner part of the circle, and 1 in obstacle cells in the outer part, the next four parameters determine to which outerwall the object is to be extended, where 0 means no extension , and 1 means extension to the respective wall. mpoints defines the number of measure points. these are equally divided over the surface at a distance of 1.5 gridpoint from it. These points are numbered, and every time step interpolated velocities are written to corresponding files.

lines

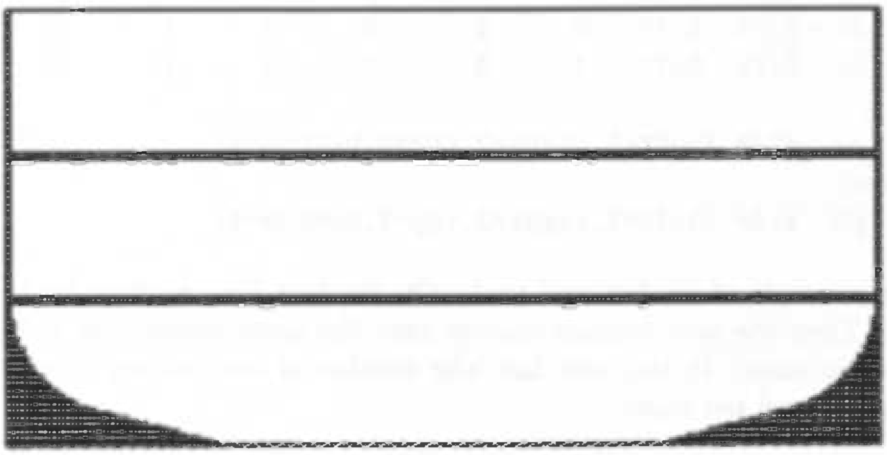
A line is defined by its start point (x1,y1) and end point (x2,y2) side determines where obstacle cells are placed:
0 means obstacle cells are put underneath, or -if the line is vertical- to the left of the line. 1 means obstacle cells are put above, or -if the line is vertical- to the right of the line. 2 means a rectangle is created with left-lower corner (x1,y1) and right-upper corner (x2,y2), and obstacle cells are placed in the inner part. The other parameters have the same meaning as for arcs.

inflows and outflows

These openings can only be situated in the outer walls, so three parameters suffice: side determines the wall in which the opening is placed. 1 or 2 means the wall is vertical, so then p1 and p2 are the y-coordinates of the opening. 3 or 4 means the wall is horizontal, so then they are the x-coordinates.

Here it is also important to notice that no lines should be added or omitted, because the parameters are expected on a constant place.

The geometry file used above results in the following geometry:



A.4.2 Output files

Savof96 can produce many output files. The most important are discussed here.

`CONFIG.DAT`
This file contains information about the fluid configuration, the grid and the parameters for the external motion.

`FILL##.OUT and FLUX##.OUT`
These files are created by the subroutines FILOUT and FLXOUT, and contain the fill ratio through an area or the flux through a line respectively. ## denotes a number correponding to the line number in the input file where the respective quantities are specified. Both type of files contain two columns. The first one is the time, and the second one the fill ratio or the flux.

`FORCES.OUT`
This file is created by the subroutine FRCOUT, and contains four columns. The first is the time, the second and third the force on the geometry in x- and y-direction respectively, and the fourth is the momentum about a specified point.

`PIPE and VOF`
These files are required by Gnuplot to illustrate the calculation *on-line*

`PRES.OUT`
This file is produced by PRSOUT and contains four columns. The first is the time, the second the pressure in a point, the third the average pressure over a specified line, and the fourth is the average pressure over a specified circle.

`SAVOF96.OUT`
This is the main output file of Savof96. It contains the coordinates of the gridpoints, rough plots of the liquid inside the geometry at equidistant points in time, followed by the number of iterations that was required to obtain the desired accuracy, the relative change in volume and an array representing the level of fluid.

`UVPF####.DAT`
This file contains five columns. The first is the time, the second and third are the horizontal and vertical velocity respectively, the fourth is the pressure and the fifth is the VOF-function. If uvpf=1 in the input file, every PrtDt a new file is created with a consecutive number.

`UVPF.tim`
This file contains one column which represents the time. The line number corresponds with the number of UVPF####.dat.

A.4.3 Postprocessing tools

To ceate al sorts of plots, and movies a Matlab menu **liqmenu** is used. It uses `CONFIG.DAT` and `UVPF####.DAT` mainly as input files, and can produce plots of velocity, pressure and the VOF-function. Also movies can be created of consecutive plots.
The other output files can easily be loaded in Matlab, and processed with the standard tools.

Bibliography

[1] J.J. van den Bosch and J.H. Vugts, *Roll damping by free surface tanks*, Report. (1966)

[2] E.F.F. Botta, *Eindige differentiemethoden*, Lecture notes R*u*G (1992)

[3] B. Buchner and G. van Ballegoyen, *Joint industry project: F(P)SO green water loading.* Volume A3: Scale effect tests. (1997)

[4] G. Fekken, *Numerical Simulation of Green Water Loading on the Foredeck of a Ship*, Master's thesis R*u*G (1998)

[5] B. de Groot, *SAVOF96 - Simulation of free-surface liquid dynamics in moving complex geometries*, Master's thesis R*u*G (1996)

[6] Z.A. Sabeur, *Development and use of an numerical model using the volume of fluid method for the design of coastal structures.* In K.W. Morton and M.J. Baines, editors, *Numerical Methods for Fluid Dynamics*, Volume V, pages 565-573. (1995)

[7] A.E.P. Veldman, *Numerieke Stromingsleer*, Lecture notes R*u*G (1994).