UiT
THE ARCTIC
UNIVERSITY
OF NORWAY

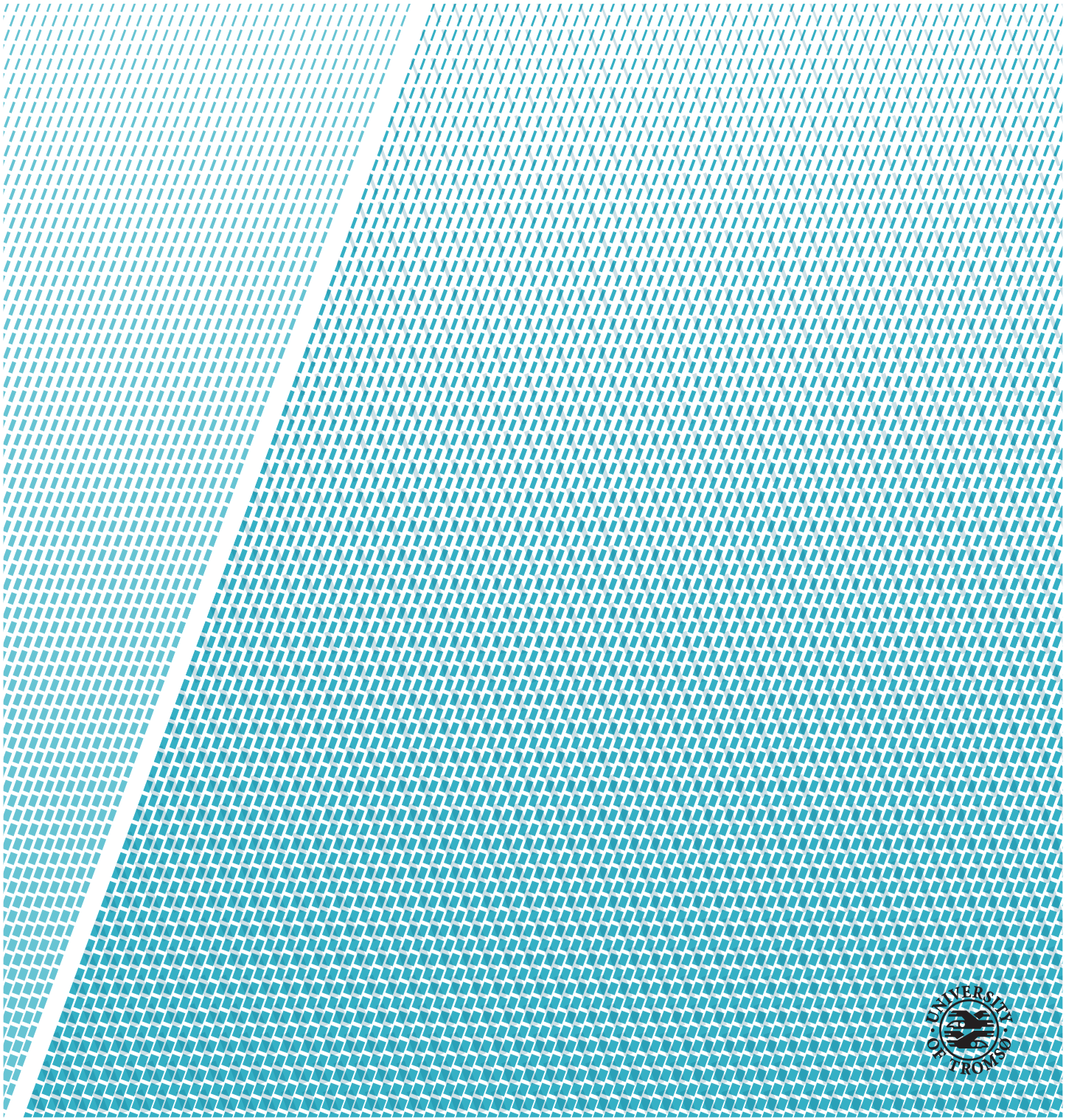
Faculty of Science and Technology
Department of Mathematics and Statistics

A boundary integral approach to the modeling of surface waves in a wave tank.

—

Sander Bøe Thygesen

MAT-3240 Master thesis in applied physics and mathematics, June 2020



Abstract

Boundary integral equations (BIEs) are used to model surface waves in a wave tank. Assuming an ideal fluid, the velocity of the fluid can be considered as a potential flow and be modeled by the Laplace equation on the domain. The domain in this case will be a section of a wave channel with an incoming wave from the right, a rigid bottom, a reflective wall on the right and a time varying surface that represent the surface waves. A numerical solution is implemented and used to simulate a wave tank at the University of Oslo. The numerical solution to the BIEs is tested for accuracy against a known solution to the Laplace equation on the boundary and it was found that the BIEs gave a mostly accurate solution except for around the parts of the parametrized boundary that were non-smooth.

Two cases of instabilities were found:
1) when the number of discrete points in time was to low and
2) when the wave amplitude got to large.
Adding a diffuse term was tried in both cases and found to be ineffective.
In case 1) it was ineffective since increasing the number of discrete points in time already made the solution stable and adding the diffusive term introduced another relation ship between the discrete points in time and space which had to be satisfied to remain stable.
In case 2) adding the diffusive term had little to no effect, except for when the number of discrete points on time was to small, in which case the instability occured earlier. It is conjectured that improving the accuracies at the non-smooth parts of the boundary will improve the stability for incoming waves.

Acknowledgements

I would first like to thank my supervisor Per Kristen Jakobsen for great guidance despite being on sabbatical this last semester. There have been many frustrating phone calls were I'm trying to write down your brilliant ideas while you explain them with great enthusiasm. Your knowledge in so many areas of math and physics is truly inspiring.

I would also like to thank all my study buddies who has kept me sane through out these five year, especially Åsmund Sandland whom I think a have seen every weekend since i moved to Tromsø. In addition, Jørn Olav Jensen has been great company while sitting late at the office, but honestly I'm glad there's a while before I need to hear about you rant about some obscure concept in pure mathematics.

Lastly, I would like to thank my girlfriend Marikken Sofie Riis who has given me a lot of love despite me looking at the office screen more than I have looked at her this last year.

Contents

List of Figures

List of Tables

Abbreviations

BIE boundary integral equation

DE differential equation

MD mean difference

ODE ordinary differential equation

PDE partial differential equation

1 Introduction

It is known that, in the case of potential flow, surface waves can be modeled using a boundary integral formulation[1][2]. The goal of this thesis is to model surface waves in a wave tank of finite length and customizable bottom using boundary integral equations (BIEs). It will be assumed that there are no significant dynamics occurring in along the width of the channel such that the surface wave can be modeled using only two spatial coordinates (length and height). The physical boundary conditions on the water surface, derived form the fluid equation assuming an ideal fluid, gives a relation between the potential flow and the surface profile, however the spatial derivatives at the boundary need to be found in order the solve for the surface profile. The spatial derivatives of the potential flow at the surface can be expressed through the normal derivative of the potential flow. Since the BIEs gives a relation between the points of the potential flow and the normal derivative on the boundary of the domain it is used to solve for the normal derivate on the surface such that the spatial derivatives can be found.

The domain of the problem, shown in figure 1.1, is a part of a larger channel where the fluid velocity on the left boundary is known. This means that the way the waves form is not modeled such that it is done in [3], instead it is assumed that the waves are already formed and simply enter our domain after some time. The finite horizontal extent of the channel is an interval $[0, L]$ on the x-axis of a Cartesian coordinate system where z is used as the vertical coordinate. The channel is assumed wide enough that there are no significant dynamics occurring in the vertical coordinate and can therefore be ignored.

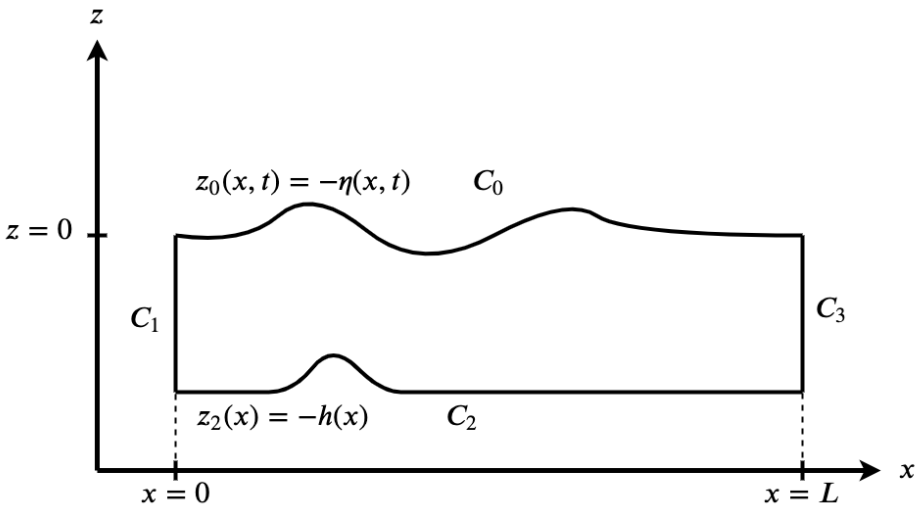


Figure 1.1: Shows the domain and each parametrized curve in the boundary.

The boundary in the domain is divided into four parts: surface (C_0), left side (C_1), bottom (C_2), right side (C_3). Further the surface and bottom profile are considered as a graph of the function $z_0(x,t) = \eta(x,t)$ and $z_2(x) = -h(x)$ respectively. Note that this approach means that there can be no breaking waves.

One of the goals in the thesis is to see if it possible to use parameters from a physical wave tank. Parameters for a wave tank at the University of Oslo is taken from the masters thesis' of Jorde and Raustøl [4][5]. These parameters will be present throughout the thesis in both testing, stability analysis and actual simulations.

2

Fluid description

The fluid is assumed to be an ideal fluid in a gravitational field. This means that there are no internal forces in the fluid and that the density is assumed to be constant. Such fluid is described by the Euler equations

$$\frac{D\boldsymbol{v}}{Dt} = -\frac{1}{\rho}\nabla p - \boldsymbol{g} \tag{2.1}$$

$$\nabla \cdot \boldsymbol{v} = 0 \tag{2.2}$$

where ρ is the constant density, p is the pressure, $\boldsymbol{g} = g\boldsymbol{e}_z = \nabla(gz)$ is the acceleration due to gravity in the direction of the z-coordinate and $\frac{D}{Dt} = \partial_t + (\boldsymbol{v} \cdot \nabla)$ is the material derivative. Here the notation $\partial_t := \frac{\partial}{\partial t}$ is used. Equation (2.1) is called the momentum equation and is a variation on Newton's second law, while equation (2.2) is called the mass conservation equation.

2.1 Potential flow

To show that an ideal fluid clan be modeled as potential flow consider the vorticity, $\boldsymbol{w}$, of the velocity field defined by

$$\boldsymbol{w} := \nabla \times \boldsymbol{v}$$

If the vorticity of the velocity field is zero it means that the velocity is a conservative vector field and can therefore be written as a scalar potential. If

$\boldsymbol{w} = 0$ at $t = 0$ it implies $\boldsymbol{w} = 0$ for all t. To show this consider the following computation: using the vector identity

$$(\nabla \times \boldsymbol{v}) \times \boldsymbol{v} = -\frac{1}{2}\nabla \boldsymbol{v}^2 + \boldsymbol{v} \cdot \nabla \boldsymbol{v}$$

equation (2.1) can be written as

$$\partial_t \boldsymbol{v} + \frac{1}{2}\nabla \boldsymbol{v}^2 + (\nabla \times \boldsymbol{v}) \times \boldsymbol{v} = -\frac{1}{\rho}\nabla p - \nabla(gz) \tag{2.3}$$

By taking the curl of equation (2.3), using that the curl of a gradient is zero and noting that $\boldsymbol{w} = \nabla \times \boldsymbol{v}$ it can be written as

$$\partial_t \boldsymbol{w} + \nabla \times (\boldsymbol{w} \times \boldsymbol{v}) = 0 \tag{2.4}$$

The relation $\nabla \times (\boldsymbol{w} \times \boldsymbol{v})$ can be rewritten using the vector calculus identity

$$\nabla \times (\boldsymbol{w} \times \boldsymbol{v}) = \boldsymbol{v} \cdot \nabla \boldsymbol{w} - \boldsymbol{w} \cdot \nabla \boldsymbol{v} - (\nabla \cdot \boldsymbol{w})\boldsymbol{v} + (\nabla \cdot \boldsymbol{v})\boldsymbol{w}$$

and noting that $\nabla \cdot \boldsymbol{v} = 0$ and $\nabla \cdot \boldsymbol{w} = \nabla \cdot (\nabla \times \boldsymbol{v}) = 0$ equation (2.4) can be written as

$$\begin{aligned} &\partial_t \boldsymbol{w} + \boldsymbol{v} \cdot \nabla \boldsymbol{w} - \boldsymbol{w} \cdot \nabla \boldsymbol{v} = 0 \\ \implies &\frac{D\boldsymbol{w}}{Dt} = \boldsymbol{w} \cdot \nabla \boldsymbol{v} \end{aligned}$$

This equation shows that if the initial condition $\boldsymbol{w}(\boldsymbol{x}, t) = 0$ is imposed it implies that $\boldsymbol{w}(\boldsymbol{x}, t) = 0 \ \forall t$. Hence if the fluid is initially vorticity free it will stay vorticity free. Therefore it will be required that the fluid is initially vorticity free such that the velocity can be written as the gradient of a scalar potential ϕ

$$\boldsymbol{v} = \nabla \phi$$

Equation (2.2) can therefore be written as

$$\nabla^2 \phi = 0$$

The fluid velocity can therefore be described by solving the Laplace equation on our domain.

The momentum equation will now be transformed using potential flow. Substituting the velocity $\boldsymbol{v}$ with the scalar scalar potential $\nabla\phi$ in equation (2.1) gives

$$\partial_t \nabla \phi + (\nabla \phi \cdot \nabla)\nabla \phi = -\frac{1}{\rho_0}\nabla p - \nabla(gz) \tag{2.5}$$

Notice that $(\nabla\phi\cdot\nabla)\nabla\phi = \frac{1}{2}\nabla(\nabla\phi)^2$ by considering the following index notation derivation

$$\begin{aligned}\nabla(\nabla\phi)_i^2 &= \partial_{x_i}(\partial_{x_j}\phi\partial_{x_j}\phi)\\ &= \partial_{x_i}(\partial_{x_j}\phi)\partial_{x_j}\phi + \partial_{x_j}\phi\partial_{x_i}(\partial_{x_j}\phi)\\ &= \partial_{x_j}\phi\partial_{x_j}\partial_{x_i}\phi + \partial_{x_j}\phi\partial_{x_j}\partial_{x_i}\phi\\ &= 2(\nabla\phi\cdot\nabla)\nabla\phi\\ \implies (\nabla\phi\cdot\nabla)\nabla\phi &= \frac{1}{2}\nabla(\nabla\phi)^2\end{aligned}$$

where Einstein's summation convention is used. Equation (2.5) can therefore be written as

$$\nabla\left[\phi_t + \frac{1}{2}(\nabla\phi)^2 + \frac{p}{\rho_0} + gz\right] = 0 \tag{2.6}$$

where the notation $f_{x_i} := \partial_{x_i} f$ is used. From equation (2.6) it is clear that

$$\begin{aligned}\phi_t + \frac{1}{2}(\nabla\phi)^2 + \frac{p}{\rho_0} + gz &= \alpha(t)\\ \implies p &= \rho_0(-\phi_t - \frac{1}{2}(\nabla\phi)^2 - gz + \alpha(t))\end{aligned}$$

where $\alpha(t)$ is some arbitrary function of t. Since the pressure really measures pressure differences the pressure can be shifted by a constant p_0 without loss of generality to get

$$\frac{p-p_0}{\rho_0} = -\phi_t - \frac{1}{2}(\nabla\phi)^2 - gz + \alpha(t) \tag{2.7}$$

where p_0 represents air pressure at the top of the liquid in this case. Note that if $\phi(x,z,t)$ is a solution to the Laplace equation, so is $\phi(x,z,t) + \psi(t)$ since $\nabla(\phi(x,z,t) + \psi(t)) = \nabla\phi(x,z,t)$. Plugging such solution into equation (2.7) yields

$$\frac{p-p_0}{\rho_0} = -\phi_t - \psi'(t) - \frac{1}{2}(\nabla\phi)^2 - gz + \alpha(t) \tag{2.8}$$

$\psi'(t)$ can now always be chosen as the cancel $\alpha(t)$ such that equation (2.8) can be written in its final form

$$\frac{p-p_0}{\rho_0} = -\phi_t - \frac{1}{2}(\nabla\phi)^2 - gz \tag{2.9}$$

2.2 Surface waves

The description of the surface waves in this paper does not not include breaking waves, therefore it is assumed that the surface of the fluid can be written as a graph of a function as shown in figure 1.1. With the assumption that no important dynamics occur along the width of the wave channel the surface is modeled as 1-dimensional horizontal motion such that the height of the fluid is written as

$$z = \eta(x(t), t)$$

Thus, parts of the boundary of the domain is time dependent. The bottom is modeled in a similar way, namely

$$z = -h(x)$$

As with the surface this description does not allow overhangs. Several boundary conditions is needed to compute the surface profile and will now be derived.

2.2.1 Kinematic surface boundary condition

Since the surface is defined as the interface between water and air, water cannot cross the boundary. Thus, the fluid velocity at the surface must be equal to the velocity of the surface. Let $\boldsymbol{x}(t) = (x(t), z(t))$ be a time dependent position vector for a point on the surface. Its velocity, $\frac{d}{dt}\boldsymbol{x}(t) = (x'(t), z'(t))$, must be equal to the velocity of the fluid at the surface $\boldsymbol{v}(\boldsymbol{x}(t), t)$, that is

$$\boldsymbol{x}'(t) = \boldsymbol{v}(\boldsymbol{x}(t), t), \quad z(t) = \eta(x(t), t)$$

Differentiating $z(t)$ with respect to time at the surface using the chain rule we get that

$$z' = \eta_x x' + \eta_t \tag{2.10}$$

Using that $(x', z') = \boldsymbol{v} = (\phi_x, \phi_z)$ at the surface equation (2.10) becomes

$$\eta_t = \phi_z - \eta_x \phi_x, \quad z = \eta(x, t)$$

Thus, the surface profile is modeled using the potential flow $\phi(x, z, t)$.

2.2.2 Dynamic surface boundary condition

Assuming that the surface is massless and that there are no surface tension the net force on the surface fluid element is $p - p_0$. Since the surface element is

massless this force must be zero, that is $p - p_0 = 0$. Therefore equation (2.9) can be written, on the surface, as

$$\phi_t = -\frac{1}{2}(\nabla\phi)^2 - gz, \quad z = \eta$$

2.2.3 Bottom boundary condition

The bottom is assumed to be impenetrable to water. This implies that the velocity cannot have a component that is normal to the surface, that is

$$\boldsymbol{v} \cdot \boldsymbol{n} = 0, \quad z = -h(x)$$

where $\boldsymbol{n}$ is the unit normal vector at the bottom.

3

Boundary integral formulation

Using the boundary conditions on the surface a solution for the surface profile η can be found by solving the following system of coupled partial differential equations (PDEs)

$$\eta_t = \phi_z - \eta_x \phi_x \tag{3.1}$$

$$\phi_t = -\frac{1}{2}\left(\phi_x^2 + \phi_z^2\right) - g\eta, \quad \phi \in C_0 \tag{3.2}$$

This system will be numerically solved by considering discrete values of η and ϕ along the graph $z(x,t) = \eta(x,t)$ and treating each discrete point as a separate functions of time. The finite partial derivatives in space couples discrete points in space together, therefore equation (3.1) and (3.2) will be a coupled system of ordinary differential equations (ODEs) in time. To solve this system numerically the values of the partial derivatives in space must be determined to step the numerical integration process forward. The partial derivatives of ϕ cannot be determined by a finite difference in space since that require values of ϕ in immediate proximity in the respective spatial variable, which is in general is not known since the values of ϕ is only known on C_0. However, consider the unit tangent $\boldsymbol{t}$ and unit normal $\boldsymbol{n}$ on the curve $z = \eta(x,t)$ and define the

tangential and normal derivative respectively as

$$\partial_{\boldsymbol{t}}\phi := \nabla\phi \cdot \boldsymbol{t} = (\phi_x, \phi_z) \cdot \boldsymbol{t}$$
$$\partial_{\boldsymbol{n}}\phi := \nabla\phi \cdot \boldsymbol{n} = (\phi_x, \phi_z) \cdot \boldsymbol{n}$$

From this definition it is clear that if $\partial_{\boldsymbol{t}}\phi$ and $\partial_{\boldsymbol{n}}\phi$ can be found ϕ_x and ϕ_z can be determined. In this chapter BIEs will be used to determine $\partial_{\boldsymbol{n}}\phi$ and later on it will be explained how values for $\partial_{\boldsymbol{t}}\phi$ can be determined. BIEs uses a Green's function together with an integral identity to relate points on the boundary to each other.

3.1 Green's functions and BIEs

Let $\mathcal{L}$ be a linear differential operator on functions $\phi : \mathbb{R}^2 \to \mathbb{R}$. A Green's function, $\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})$, for $\mathcal{L}$ is then defined by the equation

$$\mathcal{L}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) = \delta(\boldsymbol{x} - \boldsymbol{\xi})$$

where $\boldsymbol{x} = (x, z)$ and $\delta(\boldsymbol{x})$ is the two-dimensional Dirac delta generalized function defined as the outer product between two one-dimensional Dirac delta generalized functions

$$\delta(\boldsymbol{x}) := \delta(x)\delta(z)$$

with the property

$$\delta(\boldsymbol{x}) = \begin{cases} +\infty & , \quad \boldsymbol{x} = 0 \\ 0 & , \quad \boldsymbol{x} \neq 0 \end{cases}$$

This is, however, just is a heuristic definition as no function defined on the real numbers has these properties but since the rigorous detail are not important they will be omitted.

A Green's function for the Laplace operator and a BIE is derived in Appendix A.1 and A.2 respectively and are found to be

$$\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) = \frac{1}{2\pi}\ln(|\boldsymbol{x} - \boldsymbol{\xi}|)$$
$$\frac{1}{2}\phi(\boldsymbol{\xi}, t) = \text{PV}\oint_C \left[\phi(\boldsymbol{x}, t)\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \partial_{\boldsymbol{n}}\phi(\boldsymbol{x}, t)\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\right] d\boldsymbol{x}, \quad \boldsymbol{\xi} \in C$$

Where "PV" means that the integral should be evaluated as a principal value integral. This integral identity shows the connection between any points $\boldsymbol{\xi}$ on the boundary to the rest of the boundary. The principal value integral is

needed to integrate over the Green's function since $\boldsymbol{\xi} \in C$ which means that at some point $\mathcal{G}(\boldsymbol{\xi};\boldsymbol{\xi})$ must be evaluated which is undefined since $\mathcal{G}(\boldsymbol{\xi};\boldsymbol{\xi}) = \frac{1}{2\pi}\ln(0)$.

As illustrated in figure 1.1 the path C is divided into four parts. Let ϕ evaluated at C_i be written as $\phi_j := \phi|_{C_j}$ and similarly let $(x_i, z_i) := (x, z)|_{C_i}$. The path C is parametrized using a parameter $s \in [0, 1]$. Note that parts of C is time dependent and is therefore denoted by $C_i = C_i(s, t)$. Thus $\phi_j(\boldsymbol{\xi}, t)$ can be found evaluating

$$\frac{1}{2}\phi_j(\boldsymbol{\xi}, t) = \sum_{i=0}^{3}\int_I [\phi_i(x_i(s,t), z_i(s,t), t)\partial_{\boldsymbol{n}}\mathcal{G}(x_i(s,t), z_i(s,t); \boldsymbol{\xi}) \\ -\partial_{\boldsymbol{n}}\phi_i(x_i(s,t), z_i(s,t), t)\mathcal{G}(x_i(s,t), z_i(s,t); \boldsymbol{\xi})]\,|\partial_s C_i(s,t)|ds \tag{3.3}$$

where I is the interval s is integrated over and a principal value integral must be used when $i = j$. This equation can be simplified by noting that since $\boldsymbol{x}$ is a function of s then ϕ_i is simply a function of s. The same is also true about $\mathcal{G}$. The two components of $\boldsymbol{\xi}$ will also vary only on the parametrization variable, but will generally be a different value than s and will be called s'. Equation (3.3) can therefore be written as

$$\frac{1}{2}\phi_j(s', t) = \\ \sum_{i=0}^{3}\int_I [\phi_i(s,t)\partial_{\boldsymbol{n}}\mathcal{G}_i(s;s') - \partial_{\boldsymbol{n}}\phi_i(s,t)\mathcal{G}_i(s;s')]\,|\partial_s C_i(s,t)|ds \tag{3.4}$$

where $\mathcal{G}_i(s;s') = \mathcal{G}(x_i(s,t), z_i(s,t); \boldsymbol{\xi})$. Equation (3.4) shows a connection between the points of ϕ and $\partial_{\boldsymbol{n}}\phi$ on C, however the equation does not give a solution to ϕ on C since ϕ also appear inside the integral. An approximate solution can be found, however, by discretizing the integral.

3.2 Discretization

To get a solution for ϕ the integral is discretized. Let I be divided into N_i equally spaced intervals denoted I^l, where $l \in [0, N_i - 1]$. Equation (3.3) can then be written as

$$\frac{1}{2}\phi_j(s', t) = \\ \sum_{i=0}^{3}\sum_{l=0}^{N_i-1}\int_{I^l} [\phi_i(s,t)\partial_{\boldsymbol{n}}\mathcal{G}_i(s;s') - \partial_{\boldsymbol{n}}\phi_i(s,t)\mathcal{G}_i(s;s')]\,|\partial_s C_i(s,t)|ds$$

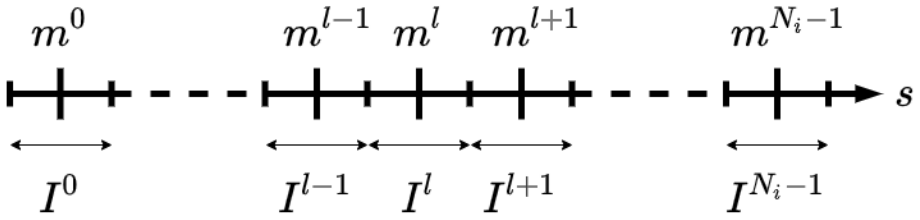


Figure 3.1: Shows the partitions of I and the mid points.

Let m^l be the midpoint of the interval I^l as shown in figure 3.1. Given sufficiently large N_i it is assumed that ϕ_i will wary little such that the integral can be approximated using the midpoint rule. The considered values of ϕ is therefore those evaluated at the midpoint. Discrete points of ϕ_j can therefore be approximated as

$$\begin{aligned}\frac{1}{2}\phi_j\left(m^k,t\right) \approx \\ \sum_{i=0}^{3}\sum_{l=0}^{N_i-1}\Bigg(&\phi_i\left(m^l,t\right)\int_{I^l}\partial_{\boldsymbol{n}}\mathcal{G}_i\left(s;m^k\right)\left|\partial_s C_i^l\left(s,t\right)\right|ds \\ &-\partial_{\boldsymbol{n}}\phi_i\left(m^l,t\right)\int_{I^l}\mathcal{G}_i\left(s;m^k\right)\left|\partial_s C_i^l\left(s,t\right)\right|ds\Bigg)\end{aligned} \tag{3.5}$$

Note that the principal value integral is now only needed when $i = j$ and $l = k$. To shorten the notation the following definitions are made

$$\begin{aligned}A_{ij}^{lk} &:= \int_{I^l}\partial_{\boldsymbol{n}}\mathcal{G}_i\left(s;m^k\right)\left|\partial_s C_i^l\left(s,t\right)\right|ds \\ B_{ij}^{lk} &:= \int_{I^l}\mathcal{G}_i\left(s;m^k\right)\left|\partial_s C_i^l\left(s,t\right)\right|ds\end{aligned}$$

Equation (3.5) can then be written as

$$\frac{1}{2}\phi_j\left(m^k,t\right) = \sum_{i=0}^{3}\sum_{l=0}^{N_i-1}\left(A_{ij}^{lk}\phi_i\left(m^l,t\right) - B_{ij}^{lk}\partial_{\boldsymbol{n}}\phi_i\left(m^l,t\right)\right) \tag{3.6}$$

This can now be considered a linear system of equations with respect to the discrete values of ϕ and $\partial_{\boldsymbol{n}}\phi$. Observe that the system has $N_0 + N_1 + N_2 + N_3$ equations with $2(N_0 + N_1 + N_2 + N_3)$ variables. Thus, to get a unique solution it is then required to supply enough boundary conditions which in particular means that enough values of ϕ and $\partial_{\boldsymbol{n}}\phi$ must be known before attempting to solve (3.6). One should keep in the back of the mind that the ultimate goal of solving this system is to find values for $\partial_{\boldsymbol{n}}\phi_0$ so that a solution to (3.1) can be found. It should therefore be noted that when equation (3.1) is solved numerically system (3.6) must be solved at every discrete time step.

3.3 Parametrization of C

It is at this point appropriate to infrom how $C_i = (x_i, z_i)$ is parametrized specifically. The boundary integral will be performed counter clockwise starting at the upper right corner of the domain. All the parametrizations uses the parameter $s \in [0, 1]$ and are therefore parametrized as

$$x_i(s,t) = \begin{cases} L(1-s) & , \quad i = 0 \\ 0 & , \quad i = 1 \\ Ls & , \quad i = 2 \\ L & , \quad i = 3 \end{cases}$$

$$z_i(s,t) = \begin{cases} \eta(x_0(s,t),t) & , \quad i = 0 \\ \eta(0,t)(1-s) - sh(0) & , \quad i = 1 \\ -h(x_2(s,t)) & , \quad i = 2 \\ -h(L)(1-s) + s\eta(L,t) & , \quad i = 3 \end{cases}$$

3.4 Boundary conditions for the linear system

The boundary conditions for each region of C will now be decided such that enough values of ϕ and $\partial_{\boldsymbol{n}}\phi$ is know to solve system (3.6).

3.4.1 C_0

When calculating a new time step ϕ_0 is known from the previous calculation. While, as mentioned, $\partial_{\boldsymbol{n}}\phi_0$ is unknown.

3.4.2 C_1

Since $\boldsymbol{n} = (-1, 0)$ on the left hand side it implies that $\partial_{\boldsymbol{n}}\phi_1 = -\phi_x$. Since ϕ_x is the horizontal component of the fluid velocity $\partial_{\boldsymbol{n}}\phi_1$ will be set as known such that the incoming fluid velocity can be decided. ϕ_1 will be set as unknown.

3.4.3 C_2

In section 2.2.3 it is stated that, at the bottom, $\boldsymbol{v} \cdot \boldsymbol{n} = 0$. Using that $\boldsymbol{v} = \nabla\phi$ and the definition of the normal derivative it is equivalent to the bottom boundary condition to say that $\partial_{\boldsymbol{n}}\phi_2 = 0$. Thus $\partial_{\boldsymbol{n}}\phi_2$ is known. No data for ϕ_2 is set, thus it is unknown.

3.4.4 C_3

So far 3 of the 8 boundary values are known, which means that one more boundary value must be known. To solve this issue the boundary value $\partial_{\boldsymbol{n}}\phi_3 = 0$ is set. Physically this means that there is a wall on the right hand side of the channel which will reflect all incoming waves. There is a slight problem with this approach, namely that when the reflected wave reach the left hand side of the channel it might be inconsistent with the boundary condition at the left side. This problem can be avoided if the simulation stops before this happens. This implies that the channel has to be very long to observe interesting phenomenon before the reflected wave interfere. With the known boundary conditions out of the way ϕ_3 is set to be unknown.

3.5 Formulation of the linear system

Let $\boldsymbol{\phi}_j$ be a vector whose components are the the discrete values of ϕ_j, that is $\boldsymbol{\phi}_j := \left(\phi_j\left(m^0\right), \phi_j\left(m^1\right), \cdots, \phi_j\left(m^{N_i-1}\right)\right)$. Similarly let $\partial_{\boldsymbol{n}}\boldsymbol{\phi}_j$ be a vector with the discrete values of $\partial_{\boldsymbol{n}}\phi_j$. Now let $\boldsymbol{\Phi}$ be the stacked vector of the unknown values, that is

$$\boldsymbol{\Phi} = (\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3)$$

and similarly

$$\boldsymbol{\Phi}^* = (\boldsymbol{\phi}_0, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_1, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_2, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_3)$$

is the stacked vector of known values. To make sure the system has a unique solution there must be as many equations as there are unknowns. The number of equations are still $N_0 + N_1 + N_2 + N_3$ and the number of unknowns are $N_0 + N_1 + N_2 + N_3$. This means that the number of discrete points on each section of C can be chosen freely. The goal is now to set up a system on the form $M\boldsymbol{v} = \boldsymbol{b}$ where $\boldsymbol{v}$ represents the unknown values. Choosing specific values for i and j the values A_{ij}^{lk} and B_{ij}^{lk} can be considered components of $N_j \times N_i$ matrices. In Appendix B a system on the form $M\boldsymbol{v} = \boldsymbol{b}$ is found to be

$$\begin{bmatrix} B_{00} & -A_{10} & -A_{20} & -A_{30} \\ B_{01} & \frac{1}{2}I_{11} - A_{11} & -A_{21} & -A_{31} \\ B_{02} & -A_{12} & \frac{1}{2}I_{22} - A_{22} & -A_{32} \\ B_{03} & -A_{13} & -A_{23} & \frac{1}{2}I_{33} - A_{33} \end{bmatrix} \boldsymbol{\Phi} = \begin{bmatrix} -\frac{1}{2}I_{00} + A_{00} & -B_{10} & -B_{20} & -B_{30} \\ A_{01} & -B_{11} & -B_{21} & -B_{31} \\ A_{02} & -B_{12} & -B_{22} & -B_{32} \\ A_{03} & -B_{13} & -B_{23} & -B_{33} \end{bmatrix} \boldsymbol{\Phi}^* \tag{3.7}$$

where I_{jj} is the $N_j \times N_j$ identity matrix. Note that the elements of B_{2j} and B_{3j} can be set to zero since $\partial_{\boldsymbol{n}}\phi_2$ and $\partial_{\boldsymbol{n}}\phi_3$ are always zero, however in testing scenarios they will remain non-zero. Solving system (3.7) now yields the

required values for $\partial_{\boldsymbol{n}}\phi_0$ and we are one step closer to be able to calculate ϕ_x and ϕ_z at the surface.

To solve system (3.7) the values of A_{ij}^{lk} and B_{ij}^{lk} must be calculated. It is again reminded that this has to be done every discrete time step. One could somewhat naively compute the integral for every combination of i, j, l and k however in a realistic setting this operation would be very computationally intensive. For this very reason most of the matrix elements is approximated using the midpoint rule. In cases where $\boldsymbol{x}\left(m^l\right)$ are "close" to $\boldsymbol{x}\left(m^k\right)$ the midpoint rule is assumed to be a bad approximation for the integral since the Green's function is close to its singularity, thus in these cases a Gaussian quadrature numerical integral is used. When $i = j$ and $l = k$ analytical expression for the integrals are found. In chapter 4 these methods are explained in more detail.

3.6 Solving the surface wave equations

Given the solution of system (3.7) the discrete values of $\partial_{\boldsymbol{n}}\phi_0(s)$ are now known. In addition the discrete values of $\phi_0(s)$ and $\eta(s)$ are known from the last time step. Values for η_x, ϕ_x and ϕ_z must now be derived using these known values. In the cases where $\frac{d}{ds}f$, where $f = f(s)$ is any function of s, needs to be evaluated numerically polynomial interpolation using neighboring points will used to approximate $\frac{d}{ds}f$.

3.6.1 Calculating η_x

Since the discrete values of $\eta(s)$ are known one can numerically calculate $\frac{d}{ds}\eta(s)$. By the chain rule it can also be determined that $\frac{d}{ds}\eta(x) = \eta_x(x)x_0'(s)$ thus

$$\eta_x(x) = \frac{1}{x_0'(s)}\frac{d}{ds}\eta(x)$$

3.6.2 Calculating ϕ_x and ϕ_z

Given the parametrization of C using the parameter s the unit tangential and normal vector can be written as

$$\boldsymbol{t}(s) = \frac{(x'(s), z'(s))}{\sqrt{(x'(s))^2 + (z'(s))^2}}$$
$$\boldsymbol{n}(s) = \frac{(z'(s), -x'(s))}{\sqrt{(x'(s))^2 + (z'(s))^2}}$$

Using the definition of the tangential and normal derivatives this gives the relation

$$\gamma \partial_{\boldsymbol{t}} \phi(s) = \phi_x(x,z)x'(s) + \phi_z(x,z)z'(s) \tag{3.8}$$
$$\gamma \partial_{\boldsymbol{n}} \phi(s) = \phi_x(x,z)z'(s) - \phi_z(x,z)x'(s) \tag{3.9}$$

where $\gamma := \sqrt{(x'(s))^2 + (z'(s))^2}$. In addition note that $\frac{d}{ds}\phi(s) = \phi_x(x,z)x'(s) + \phi_z(x,z)z'(s)$, which means that $\frac{d}{ds}\phi(s) = \gamma \partial_{\boldsymbol{t}} \phi(s)$. Using equations (3.8) and (3.9) the values for ϕ_x and ϕ_z can then be determined by

$$\phi_x(x,z) = \frac{z'(s)\gamma \partial_{\boldsymbol{n}} \phi(s) + x'(s)\partial_{\boldsymbol{s}} \phi(s)}{\gamma^2}$$
$$\phi_z(x,z) = \frac{-x'(s)\gamma \partial_{\boldsymbol{n}} \phi(s) + z'(s)\partial_{\boldsymbol{s}} \phi(s)}{\gamma^2}$$

using the values of $\partial_{\boldsymbol{n}}\phi_0(s)$ and by numerically calculating $\partial_{\boldsymbol{s}}\phi_0(s)$ from the discrete values of $\phi_0(s)$.

3.6.3 Calculating the fluid velocity field

The main focus of this thesis is the computation of surface waves. However, a small note should be done on the fluid velocity field inside the domain defined by the surface, edges and bottom. From the definition of potential flow it is known that the velocity components of the fluid velocity $\boldsymbol{v}$ is given by

$$\boldsymbol{v} = (\phi_x, \phi_z)$$

It is therefore possible to calculate the fluid velocity inside the domain if $\phi(x,z,t)$ is known inside the domain. There are several ways to compute the solution to ϕ using the boundary data, however we will stick to the boundary integral approach and use an integral identity to compute ϕ inside the domain. In Appendix A.2 an integral identity is found connecting points at boundary to points inside the domain. The integral identity is found to be

$$\phi(\boldsymbol{\xi}) = \oint_C \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}} \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) d\boldsymbol{x}$$

Since ϕ and $\partial_{\boldsymbol{n}}\phi$ is known at the boundary the value of $\phi(\boldsymbol{\xi})$ can be found for any point $\boldsymbol{\xi}$ inside the boundary by computing the above line integral. The partial derivatives of ϕ can then be found using a finite difference formula on the points inside the domain. The exact details of choices made in regards to discretization and normalization of the velocity field is explained in chapter 6

4

Calculation of the A_{ij} and B_{ij} matrix elements

The matrix elements are given by

$$A_{ij}^{lk} := \int_{I^l} \partial_{\boldsymbol{n}} \mathcal{G}_i \left(s; m^k\right) \left|\partial_s C_i^l \left(s,t\right)\right| ds$$
$$B_{ij}^{lk} := \int_{I^l} \mathcal{G}_i \left(s; m^k\right) \left|\partial_s C_i^l \left(s,t\right)\right| ds$$

Since $\eta(x_0(s),t)$ and in turn $\mathcal{G}_0(s;m^k)$ and $\partial_{\boldsymbol{n}}\mathcal{G}_0(s;m^k)$ change with time the matrix has to be generated again every discrete time step. To reduce run-time a numerical integral will not be used for every element. Instead most elements should be approximated using the mid-point rule. This is assumed to be a good approximation since the Green's function is assumed to vary little over I^l, given that there are enough discrete points in space. However, since the Green's function has a singularity the it will not vary little around this singularity. Therefore numerical integration will be used close to this singularity. At the singularity approximate analytical formulas will be derived.

Let ρ be a set threshold distance and let $\boldsymbol{m}_i^l$ be the point $(x_i(s), z_i(s))$ evaluated on the midpoint of I_l, that is $\boldsymbol{m}_i^l = \left(x_i\left(m^l\right), z_i\left(m^l\right)\right)$. When $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| \leq \rho$ the points $\boldsymbol{m}_i^l$ and $\boldsymbol{m}_j^k$ are said to be close. The three methods for computing the matrix elements will now be determined by the cases $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| > \rho$,

$\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| \leq \rho$ and $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| = \rho$. The exact value for ρ will be decided based on how accurate the solution is and how much ρ has an effect on runtime. Higher order approximations for the integral could also be used, but it was determined that the first order approximations was satisfactory (section 5.1.1).

4.1 $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| > \rho$

In these cases the integral is approximated using the mid point rule, that is

$$A_{ij}^{lk} \approx \partial_{\boldsymbol{n}} \mathcal{G}(\boldsymbol{m}_i^l; \boldsymbol{m}_j^k) |\partial_s C(m_i^l, t)| \hat{L}\left(I_i^l\right)$$
$$B_{ij}^{lk} \approx \mathcal{G}(\boldsymbol{m}_i^l; \boldsymbol{m}_j^k) |\partial_s C(m_i^l, t)| \hat{L}\left(I_i^l\right)$$

where $\hat{L}\left(I^l\right)$ is the length of the line I^l. The length of each small interval $\hat{L}\left(I_i^l\right)$ is simply given by

$$\hat{L}\left(I_i^l\right) = \frac{1}{N_i}$$

since $s \in [0, 1]$.

4.2 $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| \leq \rho$

For these matrix elements the Gaussian quadrature method is used to evaluate the integrals. A slight approximation is still be used though, namely that everything that is not singular when $\boldsymbol{m}_i^l = \boldsymbol{m}_j^k$ is taken out of the integral and evaluated using the midpoint rule. Note that when evaluating the matrix coefficients at the surface only discrete values of $z_0 = \eta$ is known. When evaluating A_{00}^{lk} and B_{00}^{lk} the points $\boldsymbol{m}_0^{l-1}$, $\boldsymbol{m}_0^l$ and $\boldsymbol{m}_0^{l+1}$ are used for polynomial interpolation of degree 2 as an approximation for η. When the polynomial expression is found the derivative with respect to s can easily be calculated so that all necessary expressions are known such that the integral over I^l can be performed. Observe that for the integral over I^0 and I^{N_i-1} the points $\boldsymbol{m}_0^{l-1}$ and $\boldsymbol{m}_0^{l+1}$ are not defined. In these cases the points $\boldsymbol{m}_0^l$, $\boldsymbol{m}_0^{l+1}$ and $\boldsymbol{m}_0^{l+2}$ for $l = 0$ and $\boldsymbol{m}_0^{l-2}$, $\boldsymbol{m}_0^{l-1}$ and $\boldsymbol{m}_0^l$ for $l = N_0 - 1$ are used instead. This causes a small extrapolation of the interpolated polynomials but it is assumed that with sufficiently large N_0 this will not pose a problem. This is the case for all the cases where polynomial interpolation is used is this paper.

4.3 $\left|\boldsymbol{m}_i^l - \boldsymbol{m}_j^k\right| = 0$

These matrix elements are calculated by simplifying the principal value integral. Again, all that is not singular will be taken outside the integral and be approximated using the midpoint rule. The Idea now is to express the x and z as polynomials of degree 2 with coefficients given by the polynomial interpolation using Newton's divided difference. Then, try to find simplified analytical solutions to the integral over the Green's function.

4.3.1 B_{jj}^{kk}

B_{jj}^{kk} is evaluated as

$$B_{jj}^{kk} = \frac{1}{2\pi} PV \int_{p^{k-1}}^{p^{k+1}} \ln\left(\left|(x_j(s,t), z_j(s,t)) - \boldsymbol{m}_j^k\right|\right) |\partial_s C(s,t)| ds$$

where p^{k+1} and p^{k-1} represents the end and start point of the interval I^k respectively. Since the following computations are time independent and independent of which part of C that are discussed the variable t and subscripts i and j are omitted. To make this integral possible and easier to evaluate analytically several simplifications are made. Firstly a change of variable is made. Remember that m^k is the mid point in the interval I^k. A new variable u is defined by

$$s = m^k + \epsilon u, \quad u \in [-1, 1]$$

Note that since $s \in [0, 1]$ the following expressions holds: $m^k = \Delta s(k + \frac{1}{2})$, $\Delta s = \frac{1}{N}$. ϵ is chosen such that $m^k + \epsilon = p^{k+1}$ and $m^k - \epsilon = p^k$. In general $x(s)$ and $z(s)$ are arbitrary functions of s, but $\eta(x(s))$ and $h(x(s))$ will be approximated using polynomial interpolation of degree 2 in s and since x is parametrized as a polynomial of maximum degree 1 they can be written using the change of variables as

$$\begin{aligned} x &= \alpha_0 + \alpha_1 \epsilon u \\ z &= \beta_0 + \beta_1 \epsilon u + \beta_2 \epsilon^2 u^2 \end{aligned}$$

where the polynomial coefficients depends on the parametrization. Since $x_i(s)$ is always a polynomial of degree 1 it has the general form

$$\begin{aligned} & x(s) = a_0 + a_1 s \\ \implies & x(u) = a_0 + a_1(m + \epsilon u) \\ \implies & \begin{cases} \alpha_0 &= a_0 + a_1 m \\ \alpha_1 &= a_1 \end{cases} \end{aligned}$$

Here a_0 and a_1 can be taken directly from the parametrization described in 3.3. $z(s)$ is in general a polynomial of degree 2 which is expressed by polynomial interpolation using newton divided difference, thus it is written as

$$\begin{aligned} & z(s) = b_0 + b_1(s - (m^k - \Delta s)) + b_2(s - (m^k - \Delta s))(s - m^k) \\ \implies & z(u) = b_0 + b_1(\epsilon u + \Delta s) + b_2(\epsilon u + \Delta s)(\epsilon u) \\ \implies & \begin{cases} \beta_0 &= b_0 + b_1\Delta s \\ \beta_1 &= b_1 + b_2\Delta s \\ \beta_2 &= b_2 \end{cases} \end{aligned}$$

As in section 4.2; since $C(s)$ is not singular it is assumed to vary little over the domain of integration and therefore approximated using the midpoint rule. Using that $\ln\left(\sqrt{f(x)}\right) = \frac{1}{2}\ln(f(x))$ and the change of variables the integral becomes

$$B_{jj}^{kk} = \frac{1}{4\pi}|\partial_s C(m^k)|PV\int_{-1}^{1} \ln\left((\alpha_0 + \alpha_1\epsilon u - \tau^k)^2 + (\beta_0 + \beta_1\epsilon u + \beta_2\epsilon^2 u^2 - \zeta^k)^2\right)\epsilon du$$

where $\boldsymbol{m}^k := (\tau^k, \zeta^k)$. Note that the polynomial coefficients are dependent on k. Given sufficiently many discrete points ϵ is assumed small enough to disregard terms of higher order of ϵ. Since the integral are known to be singular at $u = 0$ the coefficients $\alpha_0 = \tau^k$ and $\beta_0 = \zeta^k$. Multiplying out the parenthesis it was found that the leading order of epsilon inside the Green's function is ϵ^2. Disregarding terms of $O(\epsilon^3)$ the integral becomes

$$B_{jj}^{kk} = \frac{1}{4\pi}|\partial_s C(\boldsymbol{m}^k)|\epsilon PV\int_{-1}^{1} \ln\left(\left[(\alpha_1)^2 + (\beta_1)^2\right]\epsilon^2 u^2\right) du$$

Evaluating the integral gives

$$B_{jj}^{kk} = \frac{1}{4\pi}|\partial_s C(\boldsymbol{m}^l)|\epsilon\left[-2u + u\ln\left(\left[(\alpha_1)^2 + (\beta_1)^2\right]\epsilon^2 u^2\right)\right]_{-1}^{1}$$

4.3.2 A_{jj}^{kk}

A_{jj}^{kk} is evaluated as

$$A_{jj}^{kk} = \frac{1}{2\pi}PV\int_{p_j^k}^{p_j^{k+1}} \partial_{\boldsymbol{n}} \ln\left(\left|(x_j(s,t), z_j(s,t)) - \boldsymbol{m}_j^k\right|\right)|\partial_s C(s)|ds$$

Using the definition of the normal derivative with the unit normal vector given in section 3.6 gives that

$$\partial_{\boldsymbol{n}}\mathcal{G}(x_j(s), z_j(s); \boldsymbol{m}_j^k) = \frac{1}{2\pi}\frac{(x_j - \tau_j^k)z_j' - (z_j - \zeta_j^k)x_j'}{\left((x_j - \tau_j^k)^2 + (z_j - \zeta_j^k)^2\right)\sqrt{x'^2 + z'^2}}$$

As before everything that is not singular are taken out of the integral and evaluated with the mid point rule. Note that $(x - \tau^k)z' - (z - \zeta^k)x'$ evaluated at the mid point is zero, thus leading to

$$A_{jj}^{kk} = 0$$

if one disregard terms of $O(\epsilon^3)$, therefore it was decided that the leading order of ϵ should be used. Doing the same process as for B_{jj}^{kk}, but keeping terms of $O(\epsilon^3)$ it was found that A_{jj}^{kk} should be evaluated as

$$\begin{aligned} A_{jj}^{kk} &= \frac{1}{2\pi}|\partial_s C(\boldsymbol{m}^k)|\epsilon \int_{-1}^{1} \frac{\alpha_1\beta_2}{\alpha_1^2 + \beta_1^2} du \\ &= \frac{1}{\pi}|\partial_s C(\boldsymbol{m}^k)|\epsilon \frac{\alpha_1\beta_2}{\alpha_1^2 + \beta_1^2} \end{aligned}$$

5

Verification of the model

To verify that the boundary integral is implemented correctly, and that it gives the right solution two tests will now be made. These tests are a direct test of the solution to the linear system given in (3.7) and a comparison to an eigenfunction expansion. For the comparison to the eigenfunction expansion the model needs to be linearized.

5.1 Verification of the linear system provided by the boundary integral equation

To verify that solving system (3.7) gives the right solution to the Laplace equation on C it can be tested against a case where the solution is known. To find a function that satisfy the Laplace equation consider a complex function $f : \mathbb{C} \to \mathbb{C}$. If f is holomorphic on a open interval Ω then its real part $\Re\{f\}$ (and imaginary part $\Im\{f\}$) will satisfy the Laplace equation[6]. There are many such functions, however it is preferable to choose one that might represent the the dynamic in the x- and z-direction. A function that is assumed to replicate the dynamic somewhat is a functions that is periodic in the x-direction and hyperbolic in the z-direction. To get a solution ϕ that satisfy

these conditions consider the the complex function

$$\begin{aligned} f(\zeta) &= \cos(\zeta) \\ &= \frac{e^{i\zeta} + e^{-i\zeta}}{2} \end{aligned}$$

Writing ζ as $\zeta = x + iz$ where $x, z \in \mathbb{R}$ the function can be written as

$$\begin{aligned} f(x + iz) &= \frac{1}{2}\left(e^{ix}e^{-z} + e^{-ix}e^{z}\right) \\ &= \frac{1}{2}\left[(\cos(x) + i\sin(x))e^{-z} + (\cos(x) - i\sin(x))e^{z}\right] \end{aligned}$$

Choosing $\phi(x, z) = \Re\{f(x + iz)\}$ gives

$$\begin{aligned} \phi(x, z) &= \cos(x)\left(e^{-z} + e^{z}\right) \\ &= 2\cos(x)\cosh(z) \end{aligned}$$

Observe that $\phi(x, z)$ is not guarantied to satisfy the boundary conditions $\partial_{\boldsymbol{n}}\phi_2 = 0$ and $\partial_{\boldsymbol{n}}\phi_3 = 0$, however this is not a problem since the test function only has the purpose to check if the linear system gives the right solution given some boundary conditions, hence this is the reason why the matrix elements B_{2j} and B_{3j} are not set to zero.

When choosing ϕ the value of both the stacked vectors $\boldsymbol{\Phi}$ and $\boldsymbol{\Phi}^*$ will be known. By choosing $h(x)$ and $\eta(x)$ the matrices can be generated and the system can be solved. When the system is solved an approximated value for $\boldsymbol{\Phi}$, called $\hat{\boldsymbol{\Phi}}$, can be compared to the real solution. Let $\boldsymbol{\Phi}_i$ and $\hat{\boldsymbol{\Phi}}_i$ be the component of the stacked vector. The mean difference (MD) defined as

$$\text{MD} := \frac{1}{N-1}\sum_{i=0}^{N-1}\left|\boldsymbol{\Phi}_i - \hat{\boldsymbol{\Phi}}_i\right|$$

is then used to determine the accuracy of the approximate solution where N is the total number of elements in the stacked vectors.

5.1.1 Result of linear system verification

Mainly two cases was done to check if the BIEs gave the right solution to the Laplace equation on C. The parameters used in these two runs are listed in table 5.1 with the function h for the second case chosen as

$$h_2(x) = \begin{cases} H, & x < 10.6 \\ -0.2625(x - 10.6) + H, & 10.6 \leq x < 12.2 \\ 0.11, & 12.2 \leq x < 13.8 \\ 0.2625(x - 15.4) + H, & 13.8 \leq x < 15.4 \\ H, & x \geq 15.4 \end{cases}$$

Table 5.1: Parameters for the two main tests performed to verify that linear system gives the right solution.

Parameter	L	H	$\eta(x)$	$h(x)$
Case 1	30	1	$0.1\sin\left(x\left(1+e^{-4(x-10)^2}\right)+1\right)$	$H-0.25e^{-0.2(x-10)^2}$
Case 2	30	0.53	$0.1\sin\left(x\left(1+e^{-4(x-10)^2}\right)+1\right)$	See $h_2(x)$

Both cases are illustrated in figure 5.1. The particular choice of surface profile

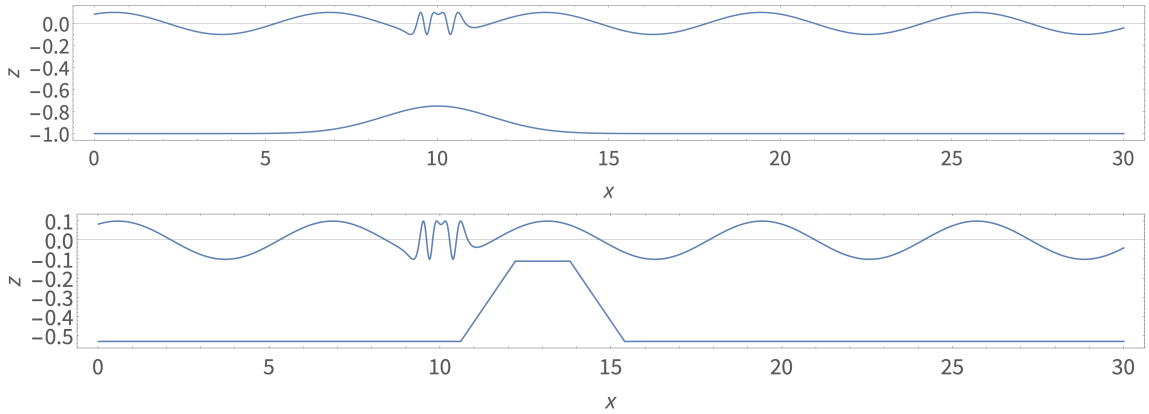


Figure 5.1: Shows the functions $\eta(x)$ and $-h(x)$ for the first and second case respectively. Note that the axis are not entirely to scale.

was made to represent both high and low frequencies and the wave amplitude expected. It is also very important that the surface profile doesn't overlap with the bottom profile since these cases are not handled by the model. This is also a good opportunity to test for which ρ the approximations described in chapter 4 are satisfactory.

The two solutions for case 2 with $N_0 = N_2 = 2000$, $N_1 = N_3 = 65$ and $\rho = \infty$ is plotted in figure 5.2. From the results seen in table 5.2 it is clear that case 2 need higher resolution in space and a higher value for the threshold ρ. This is reasonable since the distance from the surface to the bottom is smaller in case 2 which means that there are more cases where the distance between two points are close to ρ. It is also expected that the sharp edges on the bottom profile should effect the numerical result. Since ρ can be chosen quite small ($\rho = 2$) without effecting the result to much (MD $< 10^{-3}$) given enough discrete points it was decided that higher order approximation for the matrix elements was not necessary. The time required to construct and solve the matrix was also measured. It was observed that for case 2 with $N_0 = N_2 = 2000$, $N_1 = N_3 = 65$ and $\rho = \infty$ the runtime for constructing and solving the system was $T = 8.548$s.

Table 5.2: Results for the two main tests performed to verify that linear system gives the right solution. Here $\rho = \infty$ practically means that Gaussian integration is used for every element except for the singularities. The difference in the number of discrete points along the x- and z-axis is chosen such there are about the same number of discrete points per meter.

	N_0	N_1	N_2	N_3	ρ	MD
Case 1	100	5	100	5	∞ / 1.5 / 0.5 / 0.1	6.7×10^{-3} / 6.3×10^{-3} / 9.8×10^{-3} / 1.8×10^{-2}
	300	10	300	10	∞ / 1.5 / 0.5 / 0.1	8.9×10^{-4} / 8.4×10^{-4} / 1.2×10^{-3} / 2.5×10^{-3}
	800	25	800	25	∞ / 10 / 2 / 0.5	1.7×10^{-4} / 1.7×10^{-4} / 1.6×10^{-4} / 2.2×10^{-4}
	2000	65	2000	65	∞ / 10 / 2 / 0.5	4.2×10^{-5} / 4.2×10^{-5} / 4.2×10^{-5} / 5.0×10^{-5}
Case 2	100	5	100	5	∞ / 1.5 / 0.5 / 0.1	8.6×10^{-3} / 9.8×10^{-3} / 1.4×10^{-2} / 2.2×10^{-2}
	300	10	300	10	∞ / 1.5 / 0.5 / 0.1	1.9×10^{-3} / 1.9×10^{-3} / 2.2×10^{-3} / 2.8×10^{-3}
	800	25	800	25	∞ / 15 / 10 / 2 / 0.5	5.7×10^{-4} / 5.8×10^{-4} / 6.2×10^{-4} / 8.1×10^{-4} / 1.3×10^{-3}
	2000	65	2000	65	∞ / 15 / 10 / 2 / 0.5	2.0×10^{-4} / 2.1×10^{-4} / 2.4×10^{-4} / 3.2×10^{-4} / 5.2×10^{-4}

Given 20000[1] discrete points in time this will approximately equate to 48 hours for the whole solution process. Going to $\rho = 2$ the runtime for constructing and solving was lowered to $T = 7.417$s which equate to approximately 41.2 hours which was considered a significant improvement in runtime. Further, going to $\rho = 0.5$ reduced to runtime $T = 7.394$s $\rightarrow$ 41.1 hours and was not considered significant improvement in runtime such that improving the accuracy with higher order approximations was again not considered necessary.

An important note is that the MD does not represent the error perfectly. The parametrized curve C has some sharp edges where the tangent of the curve is not well defined, such as the corners of the domain and in case 2 the bottom profile. Since the BIEs relies on the derivative along the curve it should be expected that the solution to the Laplace equation close to these points has higher error than the rest of C. This is illustrated for case 2 with $N_0 = N_2 = 2000$, $N_1 = N_3 = 65$ and $\rho = \infty$ in figure 5.3 where the peaks in the difference is the points in the arrays that corresponds to points on C that are close to the sharp edged. Note that the four peaks at around $i = 1000$ is the part of the surface that is directly above the sharp edges on the bottom and it is clear that the solution at these points are also effected. Furthermore, in figure 5.4 it is clear that the error stays around 2.5×10^{-5} most of the time except for close to the sharp edges which means that the method is more accurate than the MD suggests at the points where C is not sharp. These results suggest that the implementation of the BIEs give an mostly accurate solution to the Laplace equation on C, but looses accuracy on points on C that are close to non-smooth points.

The fact that the difference in the numerical solution and analytical solution is almost as high as 5×10^{-2} around non-smooth parts of C can be seen as a weakness in the model. If this model were to be improved one could investigate

1. The number of discrete points in time is an educated guess. This guess is somewhat based on the discussion in chapter 8

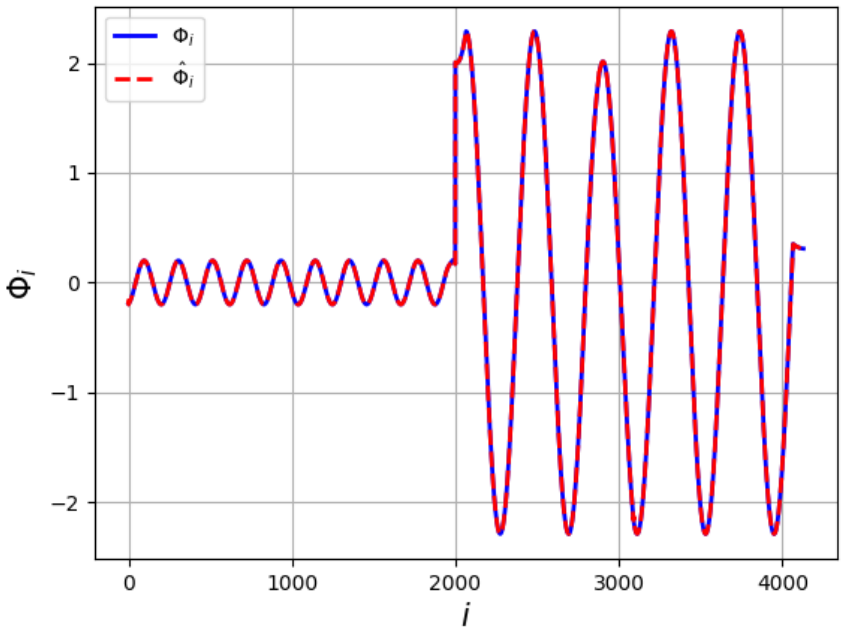


Figure 5.2: Shows the analytical and numerical solution to the Laplace equation on C plotted together.

the possibility of using a finite difference rule similar to Jenssen in [7] to find a value for $\frac{d}{ds}C(s)$. In his thesis, Jenssen uses these finite difference rules to find a value for the derivate at sharp points on a complex contour. However, these difference rules require a grid point on the sharp point. In the current implementation of the BIEs the grid points are on the mid points of intervals such that the grid points will never be on the corners of C. In addition, since the discrete points on C are not chosen directly but instead decided by the parametrization parameter s it is difficult to ensure that discrete points of x and z is on the sharp points on C. Further, Jenssen uses these finite difference rules to implement the "perfectly matched layer" to avoid reflections at the edges of his boundary. Using this method together with the boundary integral method one could avoid the reflection caused by the boundary condition on the right hand side. It is important to note that to combine the themes in this thesis and the perfectly matched layer one would have to reformulate the descriptions of the discretization.

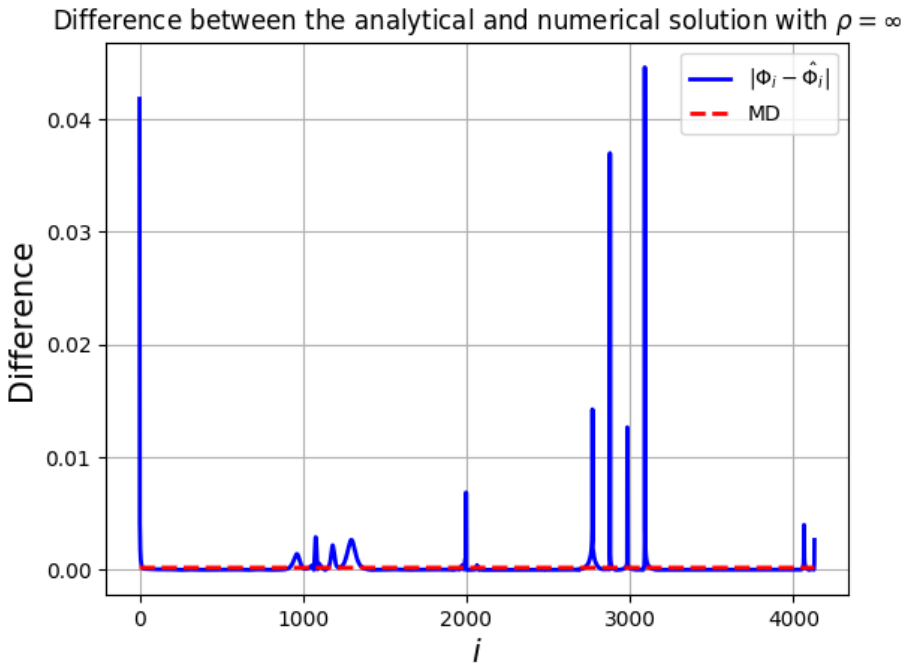


Figure 5.3: Shows the difference between the analytical solution and the numerical solution for the Laplace equation on C.

5.2 Linearization

To make sure that the boundary integral method gives the correct solution it should be compared to another solution method. If the boundary integral method is linearlized it can be compared to a solution found using eigenfunction expansion. Another benefit of linearization is that if there are any instabilities in the linear scheme it suggest that the linear terms themselves is a cause of instabilities.

5.2.1 Linearization of the surface waves equations

The surface wave equations derived earlier are

$$\begin{aligned} \eta_t &= \phi_z - \eta_x \phi_x, && z = \eta \\ \phi_t &= -\frac{1}{2}\left(\phi_x^2 + \phi_z^2\right) - g\eta, && z = \eta \end{aligned}$$

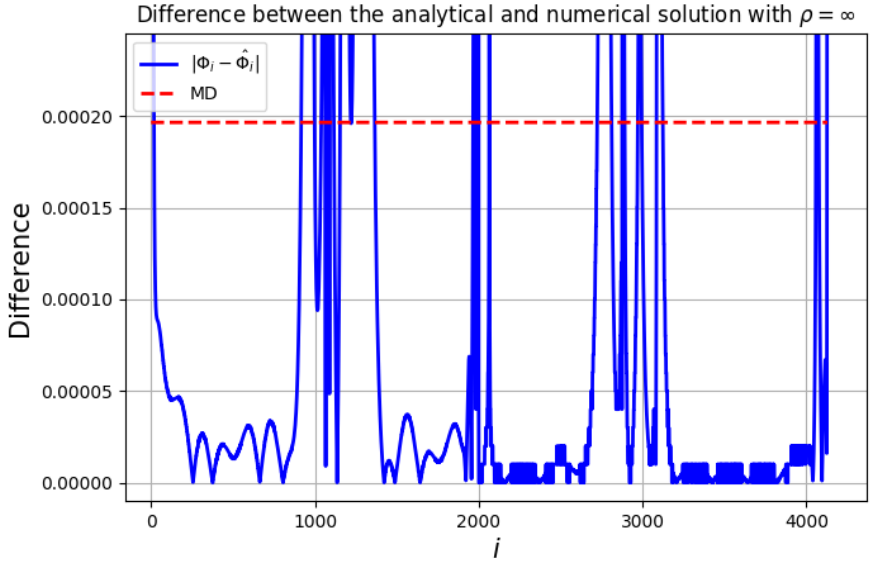


Figure 5.4: Shows the zoomed in difference between the analytical solution and the numerical solution for the Laplace equation on C.

These equations will be linearized by assuming that the functions are small perturbation to a constant mean quantity, that is

$$\eta := \bar{\eta} + \eta'$$
$$\phi := \bar{\phi} + \phi'$$

where bar over the variables signify the mean constant value and the prime signify the small perturbation. Plugging these definition into the non-linear system gives

$$\begin{aligned} \eta'_t &= \phi'_z - \eta'_x \phi'_x, && z = \bar{\eta} + \eta' \\ \phi'_t &= -\frac{1}{2}\left(\phi'^2_x + \phi'^2_z\right) - g(\bar{\eta} + \eta'), && z = \bar{\eta} + \eta' \end{aligned}$$

The small perturbations is now assumed to be small enough that their products are negligible, and since the water surface starts at rest at $z = 0$ it make sense to set $\bar{\eta} = 0$. The primes are now longer necessary and the linearized equations are now written as

$$\begin{aligned} \eta_t &= \phi_z, && z = 0 \\ \phi_t &= -g\eta, && z = 0 \end{aligned}$$

Note that the linear system are evaluated at $z = 0$. This is because ϕ in general depends on z in a non-linear way. This can be seen for example with the Taylor

expansion of ϕ_z in z around the mean value $z = \bar{\eta} = 0$

$$\phi_z(x, z, t) = \phi_z(x, 0, t) + \phi_{zz}(x, 0, t)\eta + O(\eta^2)$$

It is now clear that all but the first term is a product of small quantities and can be disregarded. In this linear case the bottom profile will be assumed flat, that is the value for z at the bottom will be given by

$$z_2 = -H$$

where H is a constant.

5.2.2 Linearization of the boundary integral method

Since ϕ and its partial derivatives now should be evaluated at $z_0 = 0$ and $z_2 = -H$ the boundary integral is now performed around a rectangle with width L and height H. This also means that the computation of the matrix elements of the system derived in sections 3.5 only needs to be done once. This is because the only thing that changes in these computation through the numerical integration process is where the Green's function should be evaluated and that does not change since the evaluation of z_0 are no longer dependent on time. This also has the effect of speeding up the computation process.

5.3 Eigenfunction expansion

The method of eigenfunction expansion works by expressing the solution of a initial boundary value problem as a series of eigenfunctions. The coefficients in this series will be determined by from the differential equation (DE), boundary conditions and initial conditions by plugging the series into the PDE.

To use eigenfunction expansion methods the linear problem needs to be reformulated as an initial boundary value problem for the Laplace equation. The initial conditions will simply be $\phi(x, z, 0) = \eta(x, 0) = 0$. The task is then to reformulate the boundary conditions.

5.3.1 Surface boundary condition

The linear system of equations for the surface profile will now be transformed into a boundary condition at $z = 0$. The equations where derived in section

5.2.1 and were found to be

$$\begin{cases} \eta_t &= \phi_z \\ \phi_t &= -g\eta \end{cases}, \quad z = 0$$

One equation can eliminated by differentiating the second equation with respect to time and plugging into the first one giving

$$\begin{aligned} \phi_{tt} &= -g\phi_z \\ \implies \phi_{tt} + g\phi_z &= 0, \quad z = 0 \end{aligned}$$

which concludes the boundary condition at the surface.

5.3.2 Bottom boundary condition

The boundary condition on the bottom was given in section 3.4 and is given as

$$\nabla\phi \cdot \boldsymbol{n} = 0, \quad z = -h(x)$$

Let $F : \mathbb{R}^2 \to \mathbb{R}$ be defined as

$$F(x, z) = z + h(x)$$

It is clear that for if $F(x, z) = 0$ it implies that $z = -h(x)$. Since $z = -h(x)$ is a contour line of F it implies that ∇F is normal to $z = -h(x)$. Since $\nabla F = (h_x, 1)$ is invariant in z the gradient of F will always be normal to the bottom. Therefore the unit normal on the bottom can be written as

$$\begin{aligned} \boldsymbol{n} &= \frac{\nabla F}{|\nabla F|} \\ &= \frac{(h_x, 1)}{\sqrt{h_x^2 + 1}} \end{aligned}$$

The boundary condition on the bottom can then be written as

$$\begin{aligned} (\phi_x, \phi_z) \cdot \frac{(h_x, 1)}{\sqrt{h_x^2 + 1}} &= 0 \\ \implies \phi_x h_x + \phi_z &= 0 \end{aligned}$$

For simplicity the bottom will be assumes flat, thus

$$\phi_z = 0, \quad z = -H$$

5.3.3 Edge boundary conditions

On the right hand side of the domain it will be assumed that all the incoming water will be reflected. This means that no water can penetrate the boundary, thus

$$\phi_x = 0, \quad x = L$$

On the left hand side there will be a wave pulse that is invariant in the z-direction, thus

$$\phi_x = f(t), \quad x = 0$$

where $f(t)$ is the incoming wave pulse. Note that ϕ_x is the x-component of the velocity, thus the function f describes the incoming fluid velocity from the left-hand side.

5.3.4 Solution to the initial boundary value problem

The final linear system is described by the Laplace equation on a rectangle with the boundary conditions described above. To summarize, the linear system is

$$\begin{aligned}
\nabla^2\phi &= 0 \\
\phi &= 0, && t = 0 \\
\eta &= 0, && t = 0 \\
\phi_{tt} + g\phi_z &= 0, && z = 0 \\
\phi_z &= 0, && z = -H \\
\phi_x &= 0, && x = L \\
\phi_x &= f(t), && x = 0
\end{aligned}$$

This is a linear PDE with an inhomogeneous boundary condition. The idea is to do an eigenvalue expansion in x and solve for the coefficients. However, the inhomogeneous boundary condition at $x = 0$ must first be made homogeneous. This is done by transforming ϕ into an inhomogeneous PDE with homogeneous boundary conditions in x. To find such transformation a simple function is found that satisfy the boundary condition at $x = 0$ and $x = L$. A simple such function is

$$\phi_{bd} = \left(x - \frac{1}{2L}x^2\right) f(t)$$

The following definition is now made

$$u(x, z, t) = \phi(x, z, t) - \phi_{bd}(x, z, t)$$

The transformed problem then becomes

$$\begin{aligned} \nabla^2 u &= \frac{1}{L} f(t) && (5.1)\\ u &= 0, && t = 0\\ u_{tt} + g u_z &= -\left(x - \frac{1}{2L}x^2\right) f''(t), && z = 0\\ u_z &= 0, && z = -H\\ u_x &= 0, && x = L\\ u_x &= 0, && x = 0 \end{aligned}$$

By solving this problem a solution for ϕ can be determined by $\phi = u + \phi_{bd}$.

To solve the initial boundary value problem the solution $u(x, z, t)$ is expressed as an eigenvalue expansion

$$u(x, z, t) = \sum_{k=0}^{\infty} N_k(z, t) M_k(x)$$

where M_k are the eigenfunctions and N_k are the z and time dependent Fourier coefficients. The eigenfunctions should be orthonormal with respect to the inner product defined as

$$\langle f(x), g(x) \rangle = \int_0^L f(x) g(x) dx$$

That is; the eigenfunction should satisfy the equation

$$\langle M_k, M_{k'} \rangle = \delta_{kk'}$$

where $\delta_{kk'}$ is the Kronecker delta defined as

$$\delta_{kk'} = \begin{cases} 1 & , \quad k = k' \\ 0 & , \quad k \neq k' \end{cases}$$

To find an expression for the coefficients N_k note that

$$\begin{aligned} \langle u, M_k \rangle &= \int_0^L \sum_{k'} \left(N_{k'} M_{k'}\right) M_k dx \\ &= \sum_{k'} N_{k'} \int_0^L M_{k'} M_k dx \\ &= \sum_{k'} N_{k'} \langle M_{k'}, M_k \rangle \\ &= N_k \end{aligned}$$

This also implies that for two eigenfunction expansion for the same function the coefficients are equal since doing the same derivation with $u = \sum_{k'} N'_{k'} M_{k'}$ would result in $N'_k = N_k$.

Since the differential operator in this problem is ∇^2 the eigenfunctions M_k should satisfy the eigenfunction equation

$$\frac{d^2}{dx^2} M_k(x) = -\lambda_k^2 M_k(x)$$

where $-\lambda_k^2$ are the eigenvalues for the operator $\frac{d^2}{dx^2}$. Solving the eigenfunction equation gives

$$M_k(x) = C_1 \cos(\lambda_k x) + C_2 \sin(\lambda_k x)$$

Having that $\frac{d}{dx} M_k(x) = -\lambda_k C_1 \sin(\lambda_k x) + \lambda_k C_2 \cos(\lambda_k x)$ gives the following system of equation for the boundary conditions at $x = 0$ and $x = L$

$$\begin{bmatrix} 0 & \lambda_k \\ -\lambda_k \sin(\lambda_k L) & \lambda_k \cos(\lambda_k L) \end{bmatrix} \begin{bmatrix} C_1 \\ C_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

For non trivial solutions to C_1 and C_2 it must be required that the determinant of the coefficient matrix must be zero, thus

$$\begin{aligned} \lambda_k^2 \sin(\lambda_k L) &= 0 \\ \implies \lambda_k &= \frac{k\pi}{L}, \quad n = 0, 1, 2, \ldots \end{aligned}$$

It is then clear that for the above system to be consistent it must be required that $C_2 = 0$. The basis eigenfunctions is therefore

$$M_{k,\text{basis}}(x) = \cos(\lambda_k x), \quad k = 0, 1, 2, \ldots$$

The normalization constant, C_n, can now be found using

$$\begin{aligned} \langle M_k, M_k \rangle &= 1 \\ \implies C_n^2 \int_0^L M_k^2(x) dx &= 1 \\ \implies C_n^2 \int_0^L \cos^2(\lambda_k x) dx &= 1 \\ \implies C_n^2 \frac{L(2\pi k + \sin(2\pi k))}{4\pi k} &= 1 \end{aligned}$$

Notice that $k \neq 0$ and must be treated separately. Using that $k = 1, 2, 3, \ldots$ the above equation reduces to

$$\begin{aligned} C_n^2 \frac{L}{2} &= 1 \\ \implies C_n &= \sqrt{\frac{2}{L}}, \quad k = 1, 2, 3, \ldots \end{aligned}$$

For $k = 0$ the normalization gives

$$\begin{aligned} &\langle M_0, M_0 \rangle = 1 \\ \implies & C_n^2 \int_0^L 1 dx = 1 \\ \implies & C_n^2 L = 1 \\ \implies & C_n = \sqrt{\frac{1}{L}}, \quad k = 0 \end{aligned}$$

This gives the orthonormal eigenfunctions

$$M_k(x) = \begin{cases} \sqrt{\frac{1}{L}}, & k = 0 \\ \sqrt{\frac{2}{L}} \cos(\lambda_k x), & k = 1, 2, 3, \dots \end{cases}$$

Given the eigenfunctions and their normalization factors the expansion is now written as

$$u(x, z, t) = \sqrt{\frac{1}{L}} N_0(z, t) + \sum_{k=1}^{\infty} N_k(z, t) M_k(x)$$

This is now a Fourier-cosine series with z and time dependent coefficients. Plugging this expansion into the inhomogeneous boundary value problem (5.1) gives

$$\sqrt{\frac{1}{L}} \partial_{zz} N_0(z, t) + \sum_{k=1}^{\infty} \left(\partial_{zz} N_k(z, t) M_k(x) + N_k(z, t) \partial_{xx} M_k(x) \right) = \frac{1}{L} f(t)$$

By the eigenfunction equation defining M_k this can be written as

$$\sqrt{\frac{1}{L}} \partial_{zz} N_0(z, t) + \sum_{k=1}^{\infty} \left(\partial_{zz} N_k(z, t) - \lambda_k^2 N_k(z, t) \right) M_k(x) = \frac{1}{L} f(t)$$

Since there is an eigenfunction expansion on the left the right hand side can also be written as an eigenfunction expansion. That is, the right hand side can be written as

$$\frac{1}{L} f(t) = b_0(t) M_0 + \sum_{k=1}^{\infty} b_k(t) M_k$$

where $b_k(t)$ must be the Fourier-cosine coefficients of the function $\frac{1}{L} f(t)$. The coefficients can therefore be written as

$$\begin{aligned} & b_k(t) = \frac{1}{L} f(t) \int_0^L M_k dx, \quad k = 1, 2, 3, \dots \\ \implies & b_k(t) = \frac{1}{L} f(t) \sqrt{\frac{2}{L}} \int_0^L \cos\left(\frac{k\pi}{L} x \right) dx \\ \implies & b_k(t) = 0, \quad k = 1, 2, 3, \dots \end{aligned}$$

Since $b_k(t) = 0$ for $k = 1, 2, 3, \ldots$ it implies then that $b_0(t)M_0 = \frac{1}{L}f(t)$. Matching the coefficients for each k then gives

$$\begin{aligned} \sqrt{\frac{1}{L}}\partial_{zz}\, N_0(z,t) &= \frac{1}{L}f(t) \\ \partial_{zz}N_k(z,t) - \lambda_k^2 N_k(z,t) &= 0, \qquad k = 1, 2, 3, \ldots \end{aligned}$$

These DEs can now be solved to determine the Fourier-coefficients N_k. For $k = 0$ a trial solution

$$N_0(z,t) = A(t)z^2 + B(t)z + C(t)$$

is made. Plugging the trial solution into the DE gives

$$\begin{aligned} \sqrt{\frac{1}{L}}2A(t) &= \frac{1}{L}f(t) \\ \implies A(t) &= \frac{1}{2\sqrt{L}}f(t) \end{aligned}$$

For $k \neq 0$ the solution is

$$N_k(z,t) = D(t)e^{\lambda_k z} + E(t)e^{-\lambda_k z}, \quad k = 1, 2, 3, \ldots$$

The solution to u therefore becomes

$$u(x,z,t) = \sqrt{\frac{1}{L}}\left(\frac{1}{2\sqrt{L}}f(t)z^2 + B(t)z + C(t)\right) + \sum_{k=1}^{\infty}\left(D_k(t)e^{\lambda_k z} + E_k(t)e^{-\lambda_k z}\right)M_k(x)$$

The time dependent constants must now be found by matching the solution with the boundary conditions. First the necessary derivatives must be computed. These are

$$\begin{aligned} u_z(x,z,t) &= \sqrt{\frac{1}{L}}\left(\frac{1}{\sqrt{L}}f(t)z + B(t)\right) \\ &\quad + \sum_{k=1}^{\infty}\lambda_k\left(D(t)e^{\lambda_k z} - E(t)e^{-\lambda_k z}\right)M_k(x) \\ u_{tt}(x,z,t) &= \sqrt{\frac{1}{L}}\left(\frac{1}{2\sqrt{L}}f''(t)z^2 + B''(t)z + C''(t)\right) \\ &\quad + \sum_{k=1}^{\infty}\left(D_k''(t)e^{\lambda_k z} + E_k''(t)e^{-\lambda_k z}\right)M_k(x) \end{aligned}$$

Boundary condition: $u_z = 0, \quad z = -H$

Have that

$$u_z(x,-H,t) = \sqrt{\frac{1}{L}}\left(-\frac{1}{\sqrt{L}}f(t)H + B(t)\right) + \sum_{k=1}^{\infty}\lambda_k\left(D(t)e^{-\lambda_k H} - E(t)e^{\lambda_k H}\right)M_k(x) = 0$$

Since the first two terms does not depend on x the above expression implies

$$\sqrt{\frac{1}{L}}\left(-\frac{1}{\sqrt{L}}f(t)H + B(t)\right) = 0$$
$$\implies B(t) = \frac{1}{\sqrt{L}}f(t)H$$

and that

$$D_k(t)e^{-\lambda_k H} - E_k(t)e^{\lambda_k H} = 0$$
$$\implies E_k(t) = e^{-2\lambda_k H}D_k(t)$$

Boundary condition: $u_{tt} + gu_z = -\left(x - \frac{1}{2L}x^2\right)f''(t), \quad z = 0$

$$\sqrt{\frac{1}{L}}C''(t) + \sum_{k=1}^{\infty}\left(1 + e^{-2\lambda_k H}\right)D_k''(t)M_k(x) + g\left[\sqrt{\frac{1}{L}}\left(\frac{H}{\sqrt{L}}f(t)\right) + \sum_{k=1}^{\infty}\lambda_k\left(1 - e^{-2\lambda_k H}\right)D_k(t)M_k(x)\right]$$

$$= -\left(x - \frac{1}{2L}x^2\right)f''(t)$$

$$\implies \sqrt{\frac{1}{L}}\left(C''(t) + g\frac{H}{\sqrt{L}}f(t)\right) + \sum_{k=1}^{\infty}\left(\left(1 + e^{-2\lambda_k H}\right)D_k''(t) + g\lambda_k\left(1 - e^{-2\lambda_k H}\right)D_k(t)\right)M_k(x)$$

$$= -\left(x - \frac{1}{2L}x^2\right)f''(t)$$

The function on the right hand side can be expanded as an eigenfunction expansion so that the Fourier coefficients can be matched. The function on the right hand side will be expanded as

$$-\left(x - \frac{1}{2L}x^2\right)f''(t) = \sqrt{\frac{1}{L}}c_0(t) + \sum_{k=1}^{\infty}c_k(t)M_k$$

where the coefficients for $k \neq 0$ will be given by

$$\begin{aligned} c_k(t) &= -f''(t)\sqrt{\frac{2}{L}}\int_0^L \left(x - \frac{1}{2L}x^2\right)\cos\left(\frac{k\pi}{L}x\right)dx \\ &= f''(t)\sqrt{\frac{2}{L}}\frac{L^2}{k^2\pi^2} \\ &= f''(t)\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2} \end{aligned}$$

and for $k = 0$

$$\begin{aligned} c_0(t) &= -f''(t)\sqrt{\frac{1}{L}}\int_0^L \left(x - \frac{1}{2L}x^2\right)dx \\ &= -f''(t)\sqrt{\frac{1}{L}}\frac{L^2}{3} \\ &= -f''(t)\frac{L^{\frac{3}{2}}}{3} \end{aligned}$$

The coefficients can now be matched to give for $k = 0$

$$\begin{aligned} C''(t) + g\frac{H}{\sqrt{L}}f(t) &= -f''(t)\frac{L^{\frac{3}{2}}}{3} \\ \implies C''(t) &= -g\frac{H}{\sqrt{L}}f(t) - \frac{L^{\frac{3}{2}}}{3}f''(t) \end{aligned}$$

and for $k = 1, 2, 3, \ldots$

$$\begin{aligned} f''(t)\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2} &= \left(1 + e^{-2\lambda_k H}\right)D_k''(t) + g\lambda_k\left(1 - e^{-2\lambda_k H}\right)D_k(t) \\ \implies D_k''(t) &= -g\lambda_k\frac{\left(1 - e^{-2\lambda_k H}\right)}{\left(1 + e^{-2\lambda_k H}\right)}D_k(t) + \frac{1}{1 + e^{-2\lambda_k H}}\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2}f''(t) \end{aligned}$$

Using the hyperbolic identity $\tanh(\theta) = \frac{1-e^{-2\theta}}{1+e^{-2\theta}}$ this can finally be written as

$$D_k''(t) = -g\lambda_k \tanh(\lambda_k H)D_k(t) + \frac{1}{1 + e^{-2\lambda_k H}}\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2}f''(t)$$

Final solution

The final solution for ϕ will finally be

$$\begin{aligned}
&\phi(x,z,t) = u(x,z,t) + \phi_{bd}(x,z,t) \\
\implies &\phi(x,z,t) = \left(x - \frac{1}{2L}x^2\right) f(t) + \sqrt{\frac{1}{L}}\left[\left(\frac{1}{2\sqrt{L}}z^2 + \frac{H}{\sqrt{L}}z\right) f(t) + C(t)\right] \\
&\qquad + \sum_{k=1}^{\infty} D_k(t)\left(e^{\lambda_k z} + e^{-2\lambda_k H} e^{-\lambda_k z}\right)\sqrt{\frac{2}{L}}\cos(\lambda_k x)
\end{aligned}$$

where $D_k(t)$ must satisfy

$$D_k''(t) = -g\lambda_k \tanh(\lambda_k H) D_k(t) + \frac{1}{1 + e^{-2\lambda_k H}}\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2} f''(t) \tag{5.2}$$

and $C(t)$ must satisfy

$$C''(t) = -g\frac{H}{\sqrt{L}} f(t) - \frac{L^{\frac{3}{2}}}{3} f''(t)$$

To compute the surface profile the equation

$$\eta(x,t) = -\frac{1}{g}\phi_t(x,z,t), \quad z = 0$$

can be used. Using this equation the surface profile will be given by

$$\begin{aligned}
\eta(x,t) = -\frac{1}{g}&\left[\left(x - \frac{1}{2L}x^2\right) f'(t) + \sqrt{\frac{1}{L}}\left[\left(\frac{1}{2\sqrt{L}}z^2 + \frac{H}{\sqrt{L}}z\right) f'(t) + C'(t)\right]\right. \\
&\left. + \sum_{k=1}^{\infty} D_k'(t)\left(e^{\lambda_k z} + e^{-2\lambda_k H} e^{-\lambda_k z}\right)\sqrt{\frac{2}{L}}\cos(\lambda_k x)\right], \quad z = 0 \\
\implies \eta(x,t) = -\frac{1}{g}&\left[\left(x - \frac{1}{2L}x^2\right) f'(t) + \sqrt{\frac{1}{L}} C'(t)\right. \\
&\left. + \sum_{k=1}^{\infty} D_k'(t)\left(1 + e^{-2\lambda_k H}\right)\sqrt{\frac{2}{L}}\cos(\lambda_k x)\right]
\end{aligned}$$

Now let $F(t)$ be the anti-derivative of $f(t)$. The function C' can then be written as

$$C'(t) = -g\frac{H}{\sqrt{L}} F(t) - \frac{L^{\frac{3}{2}}}{3} f'(t) + \kappa$$

where κ is an arbitrary constant. From the initial conditions $\phi = \eta = 0,\ t = 0$ it must be required that $f(0) = f'(0) = C(0) = C'(0) = E(0) = E'(0) = 0$. These conditions restricts the incoming fluid velocity $f(t)$. For D this is easily solved by using this condition as initial condition for solving equation (5.2). To achieve the condition on C the constants of integrations must be set accordingly. For C' the requirement is that

$$\kappa = g\frac{H}{\sqrt{L}}F(0) + \frac{L^{\frac{3}{2}}}{3}f'(0)$$

In the actual numerical computation it will only be required that these functions are approximately zero.

The DE for $D_k(t)$ will be solved numerically, but note that depending on how many terms in the Fourier series determine for how many k's $D_k(t)$ must be solved for. Equation (5.2) can be solved by reducing it to a system of first order equation by writing

$$D_k^1(t) := D_k(t)$$
$$D_k^2(t) := D_k'(t)$$

Using this notation the second order equation can be written as

$$\frac{d}{dt}D_k^1(t) = D_k^2$$
$$\frac{d}{dt}D_k^2(t) = -g\lambda_k \tanh(\lambda_k H)D_k^1(t) + \frac{1}{1+e^{-2\lambda_k H}}\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2}f''(t)$$

or as

$$\frac{d}{dt}\begin{bmatrix} D_k^1 \\ D_k^2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -g\lambda_k \tanh(\lambda_k H) & 0 \end{bmatrix}\begin{bmatrix} D_k^1 \\ D_k^2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{1}{1+e^{-2\lambda_k H}}\sqrt{\frac{2}{L}}\frac{1}{\lambda_k^2}f''(t) \end{bmatrix}$$

Since this equation should be solved for multiple k's one can write this as an uncoupled system of ODEs. Let K be the number of terms included in the

Fourier expansion. The full system will then be written as

$$\frac{d}{dt}\begin{bmatrix} D_1^1 \\ D_1^2 \\ D_2^1 \\ D_2^2 \\ \vdots \\ D_K^1 \\ D_K^2 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & 0 & & 0 & 0 \\ -g\lambda_1 \tanh(\lambda_1 H) & 0 & 0 & 0 & \cdots & 0 & 0 \\ 0 & 0 & 0 & 1 & & 0 & 0 \\ 0 & 0 & -g\lambda_2 \tanh(\lambda_2 H) & 0 & & 0 & 0 \\ & \vdots & & & \ddots & & \vdots \\ 0 & 0 & 0 & 0 & & 0 & 1 \\ 0 & 0 & 0 & 0 & \cdots & -g\lambda_K \tanh(\lambda_K H) & 0 \end{bmatrix} \begin{bmatrix} D_1^1 \\ D_1^2 \\ D_2^1 \\ D_2^2 \\ \vdots \\ D_K^1 \\ D_K^2 \end{bmatrix}$$

$$+ f''(t) \begin{bmatrix} 0 \\ \frac{1}{1+e^{-2\lambda_1 H}} \sqrt{\frac{2}{L}} \frac{1}{\lambda_1^2} \\ 0 \\ \frac{1}{1+e^{-2\lambda_2 H}} \sqrt{\frac{2}{L}} \frac{1}{\lambda_2^2} \\ \vdots \\ 0 \\ \frac{1}{1+e^{-2\lambda_K H}} \sqrt{\frac{2}{L}} \frac{1}{\lambda_K^2} \end{bmatrix}$$

With the numerical solution of $D(t)$ and $D'(t)$ the surface profile can be computed.

5.4 Comparison of the linear models

Table 5.3: Parameters for the linear boundary integral method in the linear test. The same parameters was used for the Fourier methods solution in addition that the number of terms was chosen to be $K = 300$.

Parameter	g	L	H	t_{final}	$\eta(x,0)$	$\phi(x,z,0)$	$\partial_{\boldsymbol{n}}\phi_1(z,t)$	$h(x)$	N_0	N_1	N_2	N_3	N_t
Value	9.81	20	1	8	0	0	$-0.4e^{-6(t-2)^2}$	H	500	25	500	25	800

Now that two solution methods to the linear surface waves have been derived they can be compared to see if they yield the same result. Using the parameters seen in table 5.3 a test was ran to compare the two solution methods. In this test the number of terms in the Fourier expansion was 300. Figure 5.5 shows a time evolution of the two solutions plotted together. The time evolution of the MD is seen 5.6 and it is clear that the max MD $\approx 4 \times 10^{-4}$. It is therefore concluded based on this test and the test performed in section 5.1 that the boundary integral method is implemented correctly and that it gives the right solution to the surface wave equations.

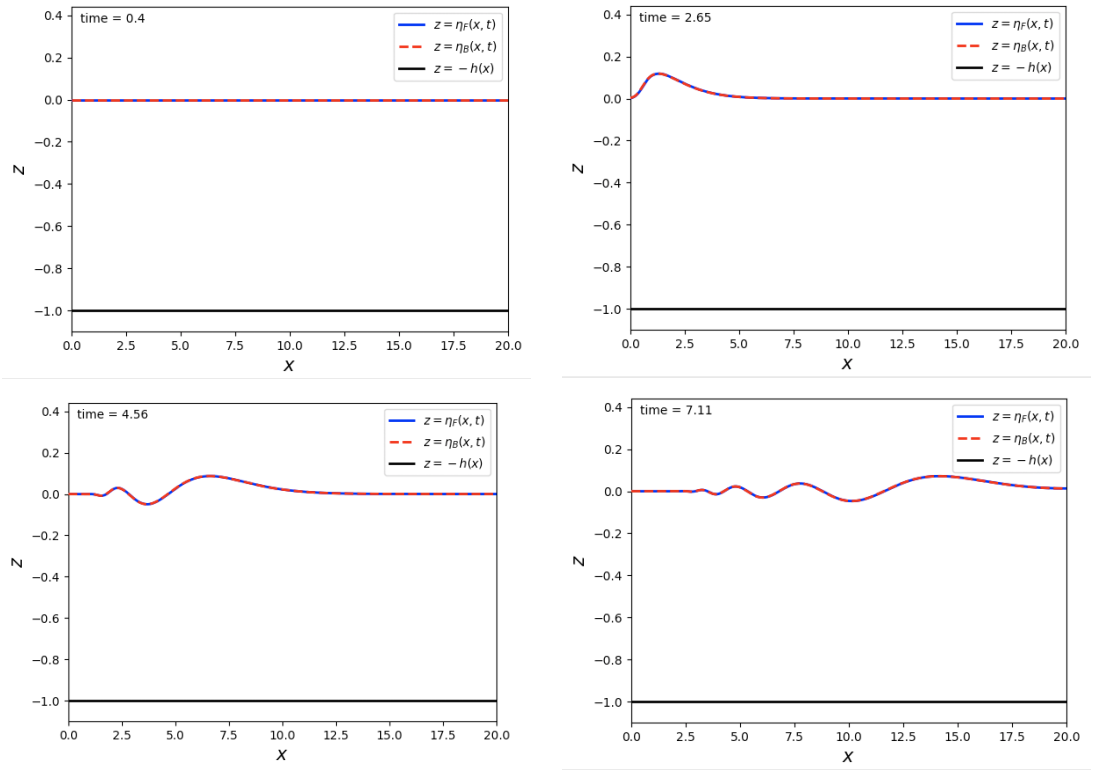


Figure 5.5: Shows the two solutions to the linear surface wave equation. Here η_F is the solution found with eigenfunctions expansion and η_B is the solution found with the boundary integral method.

No instabilities were observed for the linear boundary integral method, thus any instabilities that might occur is most likely due to the non-linear formulation of the problem.

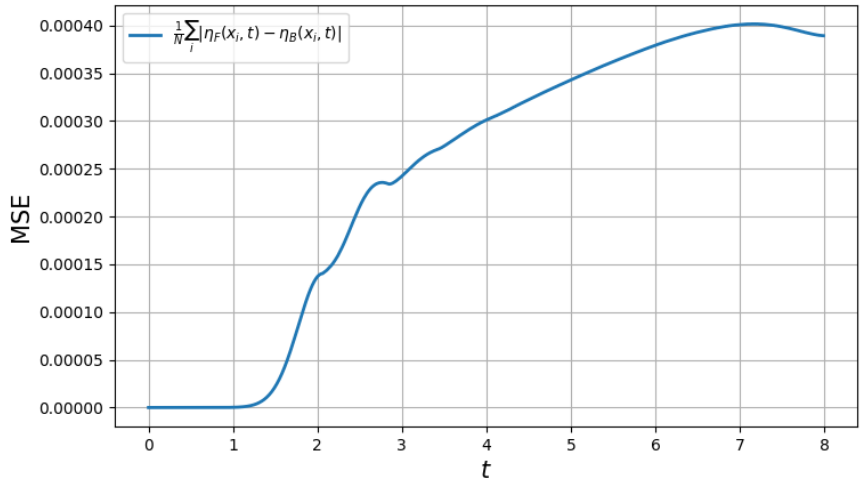


Figure 5.6: Shows time evolution of the MD where N is the number of discrete points in the x-direction.

6 Design and implementation

The source code will eventually be available at `https://github.com/Tyggesnusk/MasterThesisCode-SurfaceWaves.git`

The numerical implementation was done in C++ using the external libraries "Boost" [8] for numerical solutions of ODE-systems and "Intel Math Kernel Library" (MKL) [9] for necessary linear algebra routines. For plotting the computing data was imported into a python script that used the libraries "numpy" and "matplotlib".

6.1 Numerical integration

To solve the system of ODEs described in the beginning of chapter 3 numerically the "Odeint" part of the Boost library was used. The library lets the user solve problems of the form

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0$$

where y is potentially a vector of functions. The library routines can solves the system using a number of implemented steppers including "Explicit Euler" and

"Runge-Kutta 4". In this case the Runge-Kutta 4 method was chosen as it is a generally accurate scheme [10].

6.2 Discretization choices

As discussed in section 3.2 the number of discrete points on each section of C can be chosen independently. Since the surface and bottom will in all cases be longer that the edges the choice will usually be that $N_0 = N_2 > N_1 = N_3$. The number of discrete time steps will be a discussion later, but is should be noted here that this choice has an effect on the stability of the system.

When computing the fluid velocity field a grid was chosen in terms of the number of points in the x- and z-direction called N_x and N_z respectively. Note that there are no necessity to chose N_x and N_z in relation to the number of discrete points on the boundary. In fact there are several reasons not do it: 1) the code should be general enough that for example $N_0 \neq N_2$. The choice of N_x would now be ambiguous. 2) choosing N_x equal the number of points on the surface would potentially leave the vector field very cluttered when plotting. 3) choosing to many points on the grid would be computationally intensive since for each point in the grid a numerical integral will be computed.

When the grids of points of ϕ are computed the partials ϕ_x and ϕ_z can be computed. To do so the central finite difference formula was used given by

$$\frac{\partial}{\partial x}\phi(x,z) \approx \frac{\phi(x+\Delta x,z)-\phi(x-\Delta x,z)}{2\Delta x}$$
$$\frac{\partial}{\partial z}\phi(x,z) \approx \frac{\phi(x,z+\Delta z)-\phi(x,z-\Delta z)}{2\Delta z}$$

where Δx and Δz is the distance between between the grid points in the x- and z-direction respectively. This choice has the consequence that the differentiated values (and in extension the vector field) is not known on the grid points closest the the boundary. Note that the points on the boundary can not in general be used to compute the partial derivatives since a point on the boundary is not necessarily aligned with the grid points. It can be argued that for this reason the number of grid points should be in direct relation the the number of points on the boundary, but the points above was deemed more important. Another approach might have been that say N_0 could be an integer multiple of N_x. That way not all, but some points at a regular interval would be aligned with the grid. The problem with this approach, and perhaps the biggest problem choosing N_x and N_z in relation to N_i, is that the boundary is not a rectangle. This has the effect that for a point $\boldsymbol{x} = (x, z)$ immediately close to the boundary

need a point $\boldsymbol{x}' = (x + \Delta x, z)$ to compute the derivative. Since the boundary in general has more points than the grid the point $\boldsymbol{x}'$ might lie outside the boundary. This would not be the case if the domain was a rectangle, but for example if the boundary has a steep slope this is bound to happen. Taking the above discussion into account N_x and N_z was chosen independently.

To further avoid clutter when plotting the fluid velocity field the vectors are normalized. To avoid numerical overflow for vectors that has size close to zero the normalization

$$\boldsymbol{v} \mapsto \frac{\boldsymbol{v}}{\sqrt{v_x^2 + v_z^2 + 1}}$$

was chosen.

6.3 Parallelization and hardware

Each time step the system matrices described in section 3.5 must be filled. The system that must the be solved is

$$\begin{bmatrix} B_{00} & -A_{10} & -A_{20} & -A_{30} \\ B_{01} & \frac{1}{2}I_{11} - A_{11} & -A_{21} & -A_{31} \\ B_{02} & -A_{12} & \frac{1}{2}I_{22} - A_{22} & -A_{32} \\ B_{03} & -A_{13} & -A_{23} & \frac{1}{2}I_{33} - A_{33} \end{bmatrix} \boldsymbol{\Phi} = \begin{bmatrix} -\frac{1}{2}I_{00} + A_{00} & -B_{10} & -B_{20} & -B_{30} \\ A_{01} & -B_{11} & -B_{21} & -B_{31} \\ A_{02} & -B_{12} & -B_{22} & -B_{32} \\ A_{03} & -B_{13} & -B_{23} & -B_{33} \end{bmatrix} \boldsymbol{\Phi}^*$$

After filling the system matrices a matrix-vector product must be computed at the right-hand side for the linear system to be solved. Let $N = N_0 + N_1 + N_2 + N_3$ be the total number if discrete points on the boundary. Since the number of elements in the system matrices are N^2 the time complexity for filling the matrices will be $O(N^2)$ while solving the linear system of equation with Gaussian elimination will have time complexity $O(N^3)$ [11]. It is therefore clear that the computation time will increase extensively when increasing the system size. In addition the solution might also require fine details in time, which will further increase computation time. It will therefore be great help if the code can be parallelized. Since the computation of one matrix element is independent of the computation of any of the other elements this process can easily be parallelized. In addition there exists algorithms for parallelizing the matrix-vector product and for solving a system of linear equations.

There were mainly two parallelization techniques considered in this implementation: OpenMP [12] and MPI [13]. Generally OpenMP is chosen on shared memory computers with several CPU-cores while MPI is chosen clusters of computers where communication between nodes is essential. MKL has routines for both cases so the choice comes down to hardware. For this thesis

the main computations was done on a computer owned by the Department of Mathematics and Statistics. The computer is running Linux (Contos) and has 2 Intel Xeon Platinum 8168 with 24 cores with 768GB memory with the addition of hyper-threading which allows two threads on each core[14]. Since each processor has shared memory OpenMP was chosen.

Amdahl's law is states that the speedup factor $S(p)$ as a function of the number of processors p is

$$S(p) = \frac{p}{1 + (p-1)f}$$

where f is the fraction of the code that cannot be divided into concurrent tasks[11]. To get an idea of how much of the code that was parallelize the time

Table 6.1: Parameters used to test the speedup.

Parameter	g	L	H	$\eta(x)$	$\phi(x,z)$	ρ	N_0	N_1	N_2	N_3
Value	9.81	30	0.53	$0.1\sin\left(x\left(1+e^{-4(x-10)^2}\right)+1\right)$	$2\cos(x)\cosh(z)$	2	2000	65	2000	65

it took to generate the matrix and to solve the system using the parameters seen in table 6.1 using

$$h(x) = \begin{cases} H, & x < 10.6 \\ -0.2625(x-10.6) + H, & 10.6 \leq x < 12.2 \\ 0.11, & 12.2 \leq x < 13.8 \\ 0.2625(x-15.4) + H, & 13.8 \leq x < 15.4 \\ H, & x \geq 15.4 \end{cases}$$

was measured and is plotted as a function of number of threads in figure 6.1 As seen in the figure it seems reasonable that about about 37.5% of the matrix generation and system solving process can not be parallelized. Both the two CPUs could be taken advantage of using MPI, however parallelizing the system solution through MPI is quite tedious work and was not prioritized. In addition the speedup achieved by utilizing more than 48 threads was not considered worth it. According the estimated percent of non-parallizable code the potential speedup going from 48 to 96 thread would only be $S(48) = 2.58$ to $S(96) = 2.62$. It might also be that the overhead caused by MPI could increase run time, however this point was not explored further.

Two methods for solving the linear system was implemented. The first one using LU-decomposition using the MKL routine "dgetrf" and then solving the reduced system with the MKL routine "dgetrs". The second method was using MKLs implementation[15] of the iterative solver GMRES. In both cases the matrix-vector product had to be computed which was done using the MKL routine "dgemv". According to the MKL documentation [16] a lot the linear

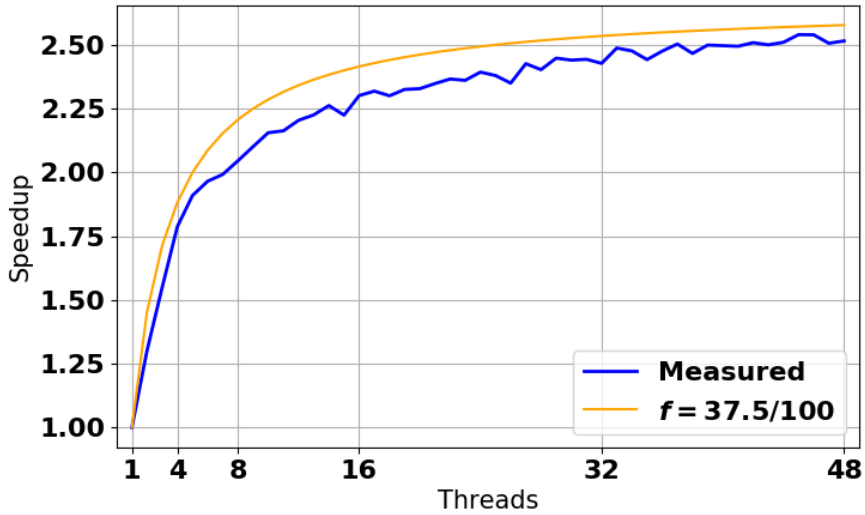


Figure 6.1: Shows the speedup as a function of the number of threads.

algebra routines in MKL are threaded for OpenMP. In this case the relevant MKL functions that are threaded with OpenMP are “dgetrf” for computing the LU factorization of the system matrix and “dgemv” for computing the matrix-vector product.

7 Simulations

Table 7.1: Parameters for the small test.

Parameter	g	L	H	t_{final}	$\eta(x,0)$	$\phi(x,z,0)$	$\partial_{\boldsymbol{n}}\phi_1(z,t)$	$h(x)$	N_0	N_1	N_2	N_3	N_t
Value	9.81	10	1	5	0	0	$-0.4e^{-6(t-2)^2}$	$H-0.25e^{-4(x-3)^2}$	200	50	200	50	2500

As general small test of the method a run was done using the parameters listed in table 7.1 which yielded the result seen in figure 7.1. Here the orange line represent the bottom while the blue represent the surface. Note that the figures are not to scale. The horizontal axis shows 10 meters while the vertical only shows 1.5 meters. The vector field inside the domain represent the fluid velocity field.

7.1 Matching the wave tank at the University of Oslo

In this section parameters for the bottom and tank-length are taken from a master thesis by Anne Raustøl[5]. The bottom has an upward linear ramp followed by a shallow flat top followed again by a downward ramp. This type of ramp is shown in figure 7.2 and was first proposed by [17]. To achieve the

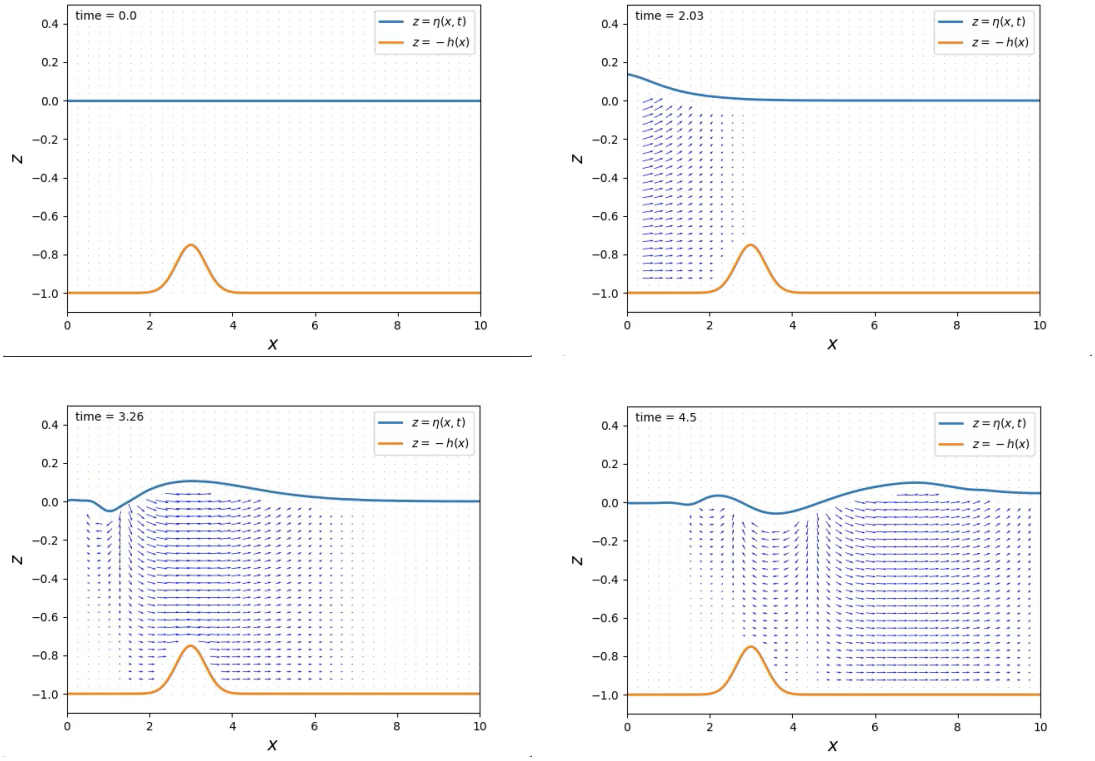


Figure 7.1: Shows a small test for the non-linear boundary integral method together with the computed vector field.

specific bottom used by Raustøl $h(x)$ was chosen as

$$h(x) = \begin{cases} H, & x < 10.6 \\ -0.2625(x - 10.6) + H, & 10.6 \leq x < 12.2 \\ 0.11, & 12.2 \leq x < 13.8 \\ 0.2625(x - 15.4) + H, & 13.8 \leq x < 15.4 \\ H, & x \geq 15.4 \end{cases}$$

where $H = 0.53$. The wave tank is 24.5 meters long and 0.5 meters wide and it assumed that the tank is wide enough that the waves can be modeled as 1-dimensional horizontal motion. At the end of the tank there is a dampening shore[4], however in this thesis no methods for avoiding reflection were explored. Because of this the simulated tank was simply extended to 30 meters to allow the wave interaction over the shallow bottom to happen before reflected waves interfere. The incoming fluid velocity was chosen as Gaussian in time and invariant in space. This means that the fluid velocity is equal at all points at the right hand edge, but start of as approximately zero at $t = 0$ and stays approximately zero after the incoming wave has passed. Table 7.2 shows all

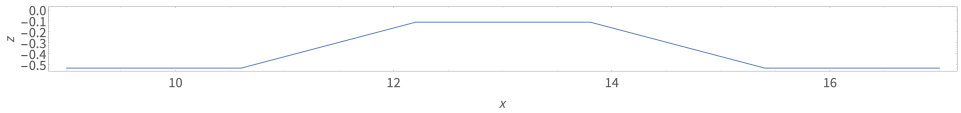


Figure 7.2: Shows the type of bottom proposed by [17].

Table 7.2: Parameters for a run simulating the wave tank at UiO.

Parameter	g	L	H	t_{final}	$\eta(x,0)$	$\phi(x,z,0)$	$\partial_{\boldsymbol{n}}\phi_1(z,t)$	ρ	N_0	N_1	N_2	N_3	N_t
Value	9.81	30	0.53	15	0	0	$-0.4e^{-6(t-2)^2}$	2	500	25	500	25	5000

the parameters used for the simulated run. In figure 7.3 the time evolution of the simulated run is shown. Note that the wave does not hit the right edge before the end of the simulation this was also confirmed by manually looking at the data produces by the simulation. In Figure 7.4 a smaller section of the

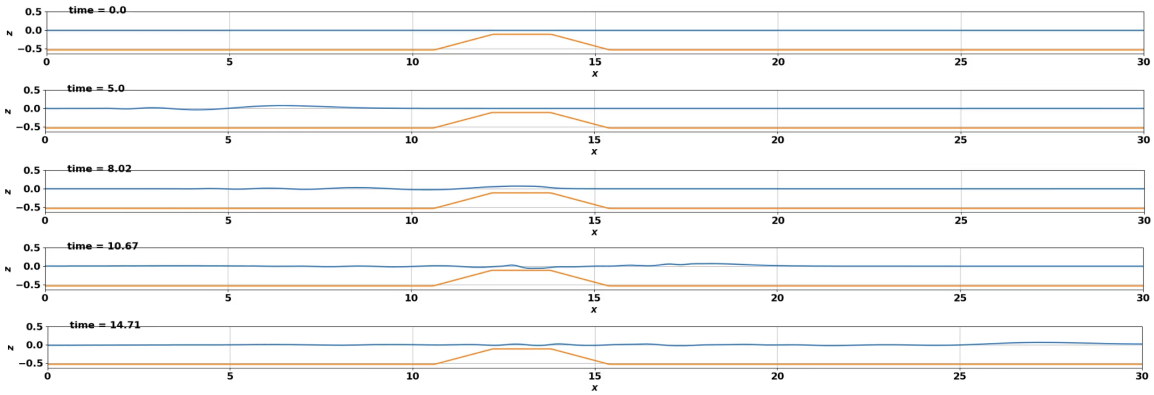


Figure 7.3: Shows simulation based on the wave tank at UiO.

tank is shown to better show the wave interaction. Even though the tank is made longer it may be that waves can be reflected from the start of the first ramp on the bottom. If water is reflected here it will reach the left hand side of the tank. This can be a potential problem if the returning water velocity is inconsistent with the boundary condition on the left hand side. In the case of $\partial_{\boldsymbol{n}}\phi_1(z,t) = -0.4e^{-6(t-2)^2}$ it might be that the incoming wave has passed before the returning fluid reach the left edge. In that case the boundary condition at the left edge will be approximately zero, which will mean that the returning fluid will reflect again from the left hand side. This doubly reflected water will after a time reach the interacting waves at the shallow bottom. Depending on the experimental setup this might be representable of the experiment if reflected water reflect of a wave paddle. However, if the model is used to model a tank where the wave paddle continuously creates waves it might lead to inconsistencies in the model. In figure 7.5 it can be seen in the fluid velocity field that some water returns from the bottom profile and is reflected from the

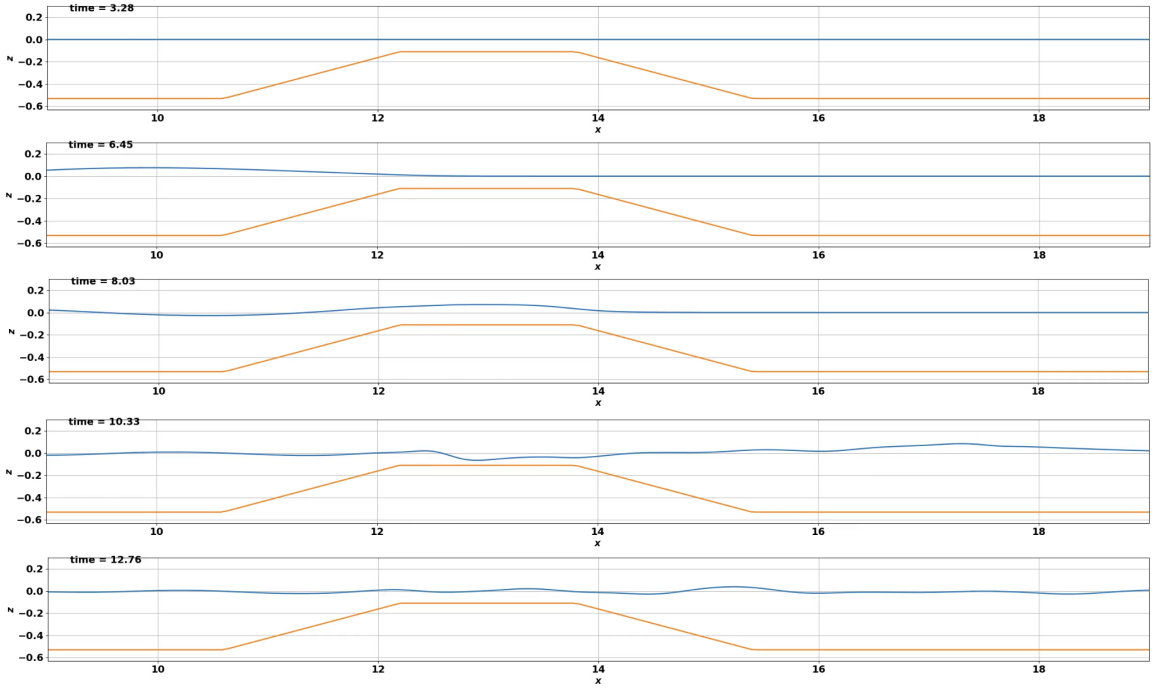


Figure 7.4: Shows the time evolution of a section of the wave tank.

left hand side.

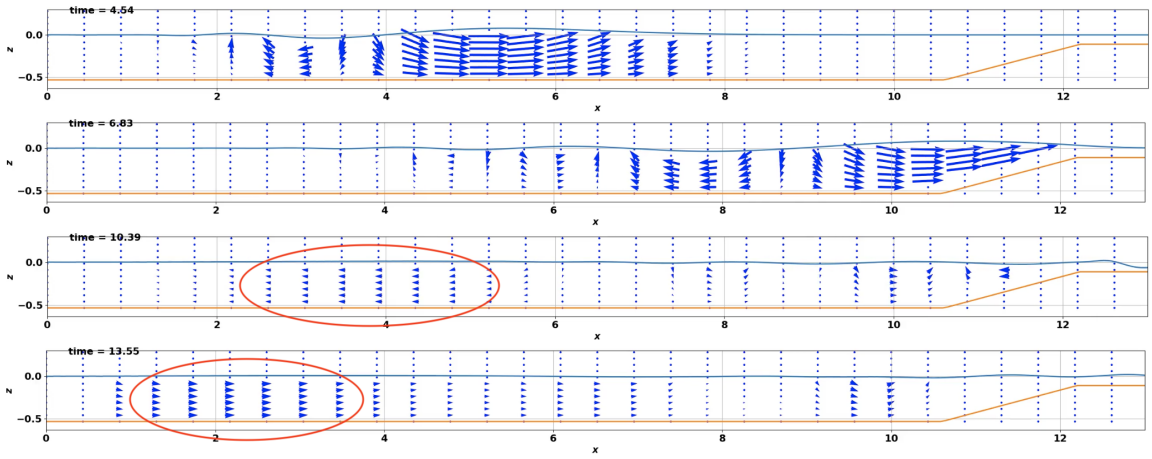


Figure 7.5: Shows the time evolution the first section of the tank. The red circle indicate water that may have been reflected from the bottom profile. The vectors are artificially scaled to better show the reflected fluid.

8 Instabilities

There are mainly two cases of known instabilities: when the number of discrete points in time are to small compared to the number of discrete points on space, and when the wave amplitude gets to large. These cases will now be discussed and will be followed by a discussion on how to possibly deal with said instabilities

8.1 Relationship between step sizes in space and time

Table 8.1: Parameters for an unstable run caused by to few discrete points in time.

Parameter	g	L	H	t_{final}	$\eta(x,0)$	$\phi(x,z,0)$	$\partial_{\boldsymbol{n}}\phi_1(z,t)$	$h(x)$	N_0	N_1	N_2	N_3	N_t
Value	9.81	10	1	5	0	0	$-0.4e^{-6(t-2)^2}$	$H-0.25e^{-4(x-3)^2}$	200	50	200	50	800

Running a test with the parameters seen in 8.1 yields the numerical instability seen in figure 8.1. These parameters are the same as in the test performed in section 7 with fewer discrete points in time. It is therefore reason to believe that there exist some condition on the relation between discrete points in time and space which determine numerical stability or rather a relationship between the step size in space Δx and the step size in time Δt. To further test existence of such relationship the same parameters as the stable run in section 7 was again used, but with $N_0 = N_2 = 400$ and it was then observed that instabilities

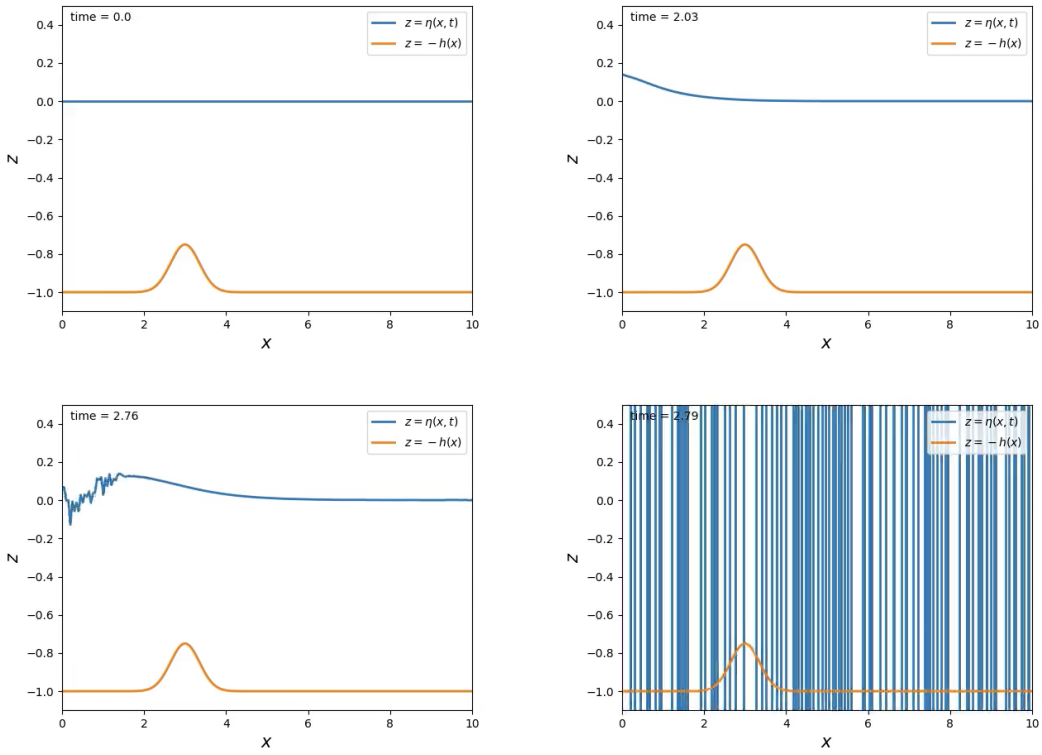


Figure 8.1: Shows the evolution of the numerical instabilities from a test run with few discrete points in time.

occurred. Furthermore, increasing L from 10m to 20m with $N_0 = N_2 = 400$ yielded a stable solution. This clearly support the idea that the instabilities come from not satisfying a specific relation between Δx and Δt. This relation has consequences for the runtime of the numerical implementation. For a specific length L increasing accuracy for the BIEs requiring smaller Δx which in itself slows down the computation process. In addition, smaller Δx require smaller Δt to keep stability, thus slowing down the computation process even more.

8.1.1 A note on initial condition

When there was no incoming wave, but instead an initial surface elevation

$$\eta(x, 0) = 0.1e^{-5(x-L)^2}$$

it was observed that fewer discrete points in time was needed to ensure stability. Using the same number of discrete points in space as for the stable test run in section 7 it was observed that $N_t = 800$ was enough to keep stability, which

was not enough for an incoming wave. This scenario is shown in figure 8.2. Lowering to $N_t = 200$ however broke stability. There indeed seem to be a

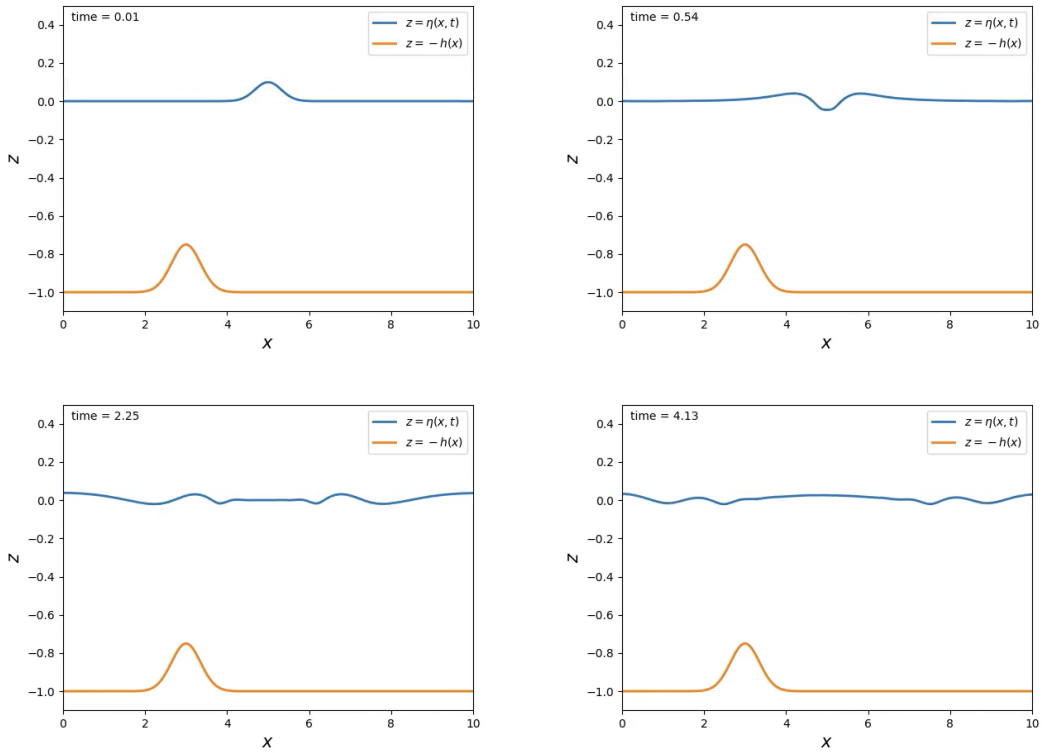


Figure 8.2: Shows the evolution of the test run with an initial surface elevation.

relation between Δx and Δt in this case as well, however not as strict as for incoming waves. Note that this scenario is not necessarily physical as it's not entirely clear how such surface elevation could occur without any initial fluid velocity present, but it serves to illustrate that the condition on the relation between Δx and Δt seem to depend on how the waves are formed.

8.2 Wave amplitudes effect on stability

Using the same parameters as the stable test run in section 7 but with the magnitude of the incoming fluid velocity increased from 0.4 to 0.8, specifically setting $\partial_{\boldsymbol{n}}\phi_1(z,t) = -0.8e^{-6(t-2)^2}$, leads to the instabilities seen in figure 8.3. A higher magnitude in incoming fluid velocity over the same period in time will naturally increase the wave amplitude and it seem as if this higher amplitude wave causes an unstable solution. It might be, however that it is not the surface

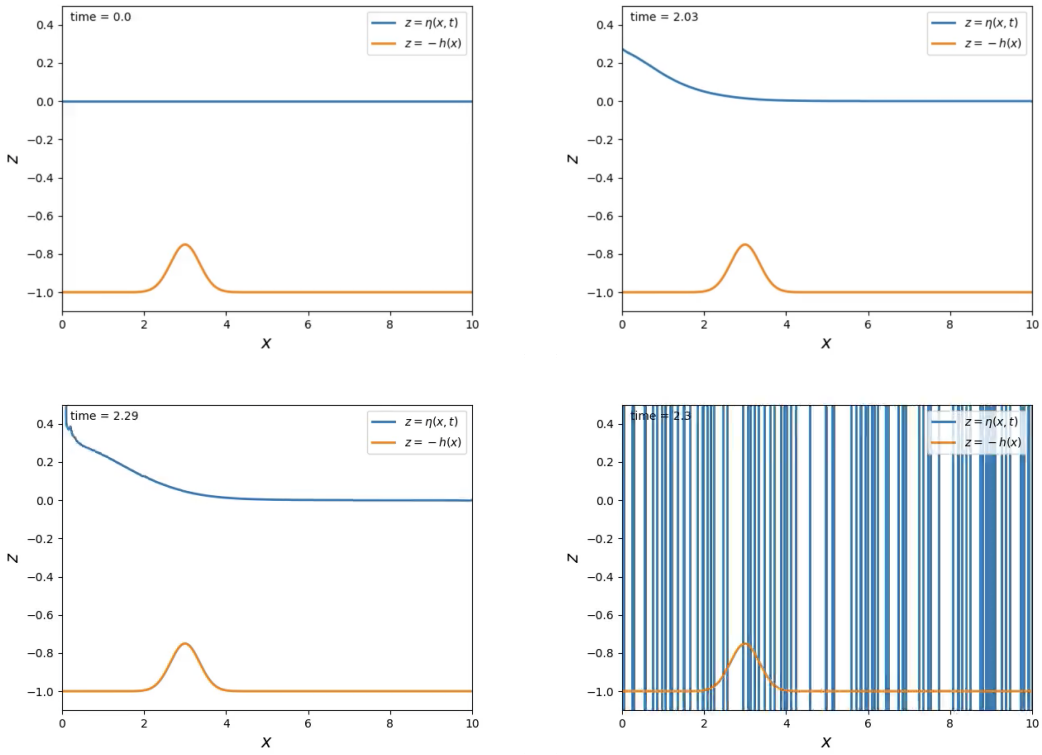


Figure 8.3: Shows the evolution of the numerical instabilities from a test run with higher incoming fluid velocity.

elevation but the higher fluid velocity itself that causes the instability, or a combination of the fluid velocity and the amplitude. To test what causes the stability the incoming velocity was changed to

$$\partial_{\boldsymbol{n}}\phi_1(z,t) = -0.8e^{-6(t-2)^2}e^{-75(z+0.5)^2}$$

That way, the middle of the left edge will still have the high fluid velocity as in the last test, but since the fluid velocity is lower near the edges the surface elevation will not be as large. If the higher fluid velocity is responsible for the instabilities this test run should still be unstable. If it's stable it suggest that the surface elevation is responsible for the instability. From the result seen in figure 8.4 it is clear that this scenario results in no instabilities suggesting that the instabilities arise from the surface elevation. Instabilities was also observed for the case with an initial surface elevation without incoming fluid velocity when the initial surface elevation was chosen as

$$\eta(x,0) = 0.2e^{-5(x-L)^2}$$

which is twice the amplitude of the stable one. It was observed that increasing N_t did not remove, nor delayed the instabilities. The fact that increasing N_t

did not effect the stability for “higher” amplitude waves suggest another cause for these kind of instabilities and it might be the case that these must be treated differently.

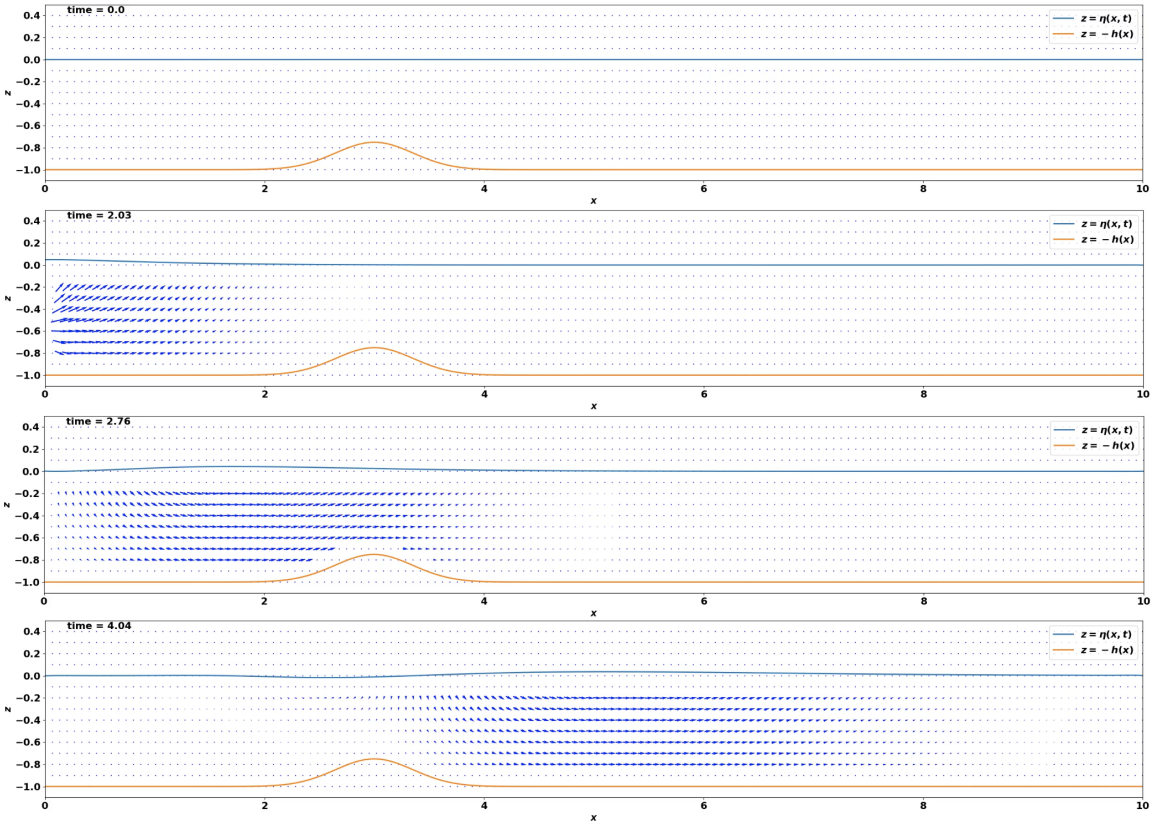


Figure 8.4: Shows the stable run with higher incoming fluid velocity in the middle for the left edge. The solution is plotted to scale and with the fluid velocity field to better visualize the difference in incoming fluid velocity along the left edge.

Again an initial surface elevation case was tested and it was indeed seen that a larger magnitude initial surface elevation caused an instability. As with the incoming wave increasing N_t had no effect on stability.

8.3 Smoothing using artificial diffusion

It is now hypothesized that the instabilities manifest them selfs as perturbations of high frequencies in the spatial frequency domain. To illustrate the perturbation of high frequencies the discrete spatial Fourier transform is performed on

the unstable numerical solution of $\eta(x,t)$. Note that a numerical implementation of the Fourier transform (such as the "Fast Fourier Transform") requires periodic boundary conditions. Since $\eta(0,t) = f(t) \neq \eta(L,t) = 0$ the solution must be transformed into a functions with periodic boundary conditions. This can be done by subtracting the solution by the function

$$g(x,t) = \frac{\eta(L,t) - \eta(0,t)}{L}x + \eta(0,t)$$

The function μ defined as

$$\mu(x,t) = \eta(x,t) - g(x,t)$$

will now satisfy the periodic boundary conditions $\mu(0,t) = \mu(L,t) = 0$. The Fourier transform

$$\mu(x,t) \xrightarrow{\mathcal{F}} \hat{\mu}(k,t)$$

where k is the wave number in space, will now give the frequency spectrum $\hat{\mu}(k,t)$ of $\mu(x,t)$ and can be used to see the perturbation of high frequencies. Figure 8.5 shows the discrete Fourier transform of the unstable run shown in figure 8.1 and one can see that high frequencies starts to form at around 2.73 seconds. In some cases kinds these high frequencies can be canceled by adding

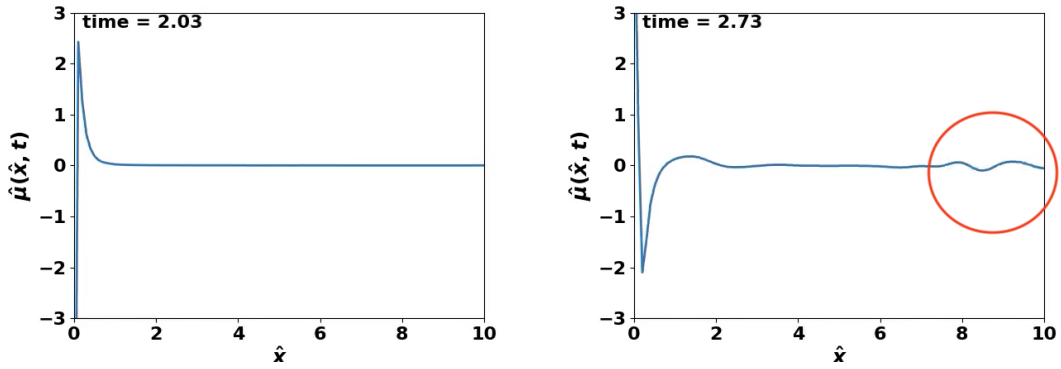


Figure 8.5: Shows the time evolution of the spatial-frequency domain. Note that the variable $\hat{x}$ is connected to the wave number k through $\hat{x} = 2\pi k$. The circle emphasizes the perturbation of high frequencies around the 2.73 second mark.

diffusive terms to the original PDE[18]. To illustrate this consider the following PDE

$$\partial_t \eta(x,t) = \epsilon \partial_{xx} \eta(x,t)$$

where ϵ is a real positive number. This is now of course the one-dimensional heat equation. The idea now is to show that high frequencies in space for the

solution to the heat equation are damped heavily if ϵ is large enough. Consider the Fourier transform in space given by

$$\mathcal{F}(\eta(x,t)) = \hat{\eta}(k,t) = \int_{-\infty}^{\infty} e^{-ikx}\eta(x,t)dx$$

where k is the wave number in space. Doing the Fourier transform on the heat equation gives

$$\partial_t \hat{\eta}(k,t) = \epsilon \int_{-\infty}^{\infty} e^{-ikx}\partial_{xx}\eta(x,t)dx$$

which after doing integration by parts two times gives

$$\partial_t \hat{\eta}(k,t) = -\epsilon k^2 \hat{\eta}(k,t)$$

This is now a differential equation for the frequency spectrum as a function of time and the wave number k. This DE can be solved by integrating in time yielding

$$\hat{\eta}(k,t) = \hat{\eta}_0 e^{-\epsilon k^2 t}$$

where $\hat{\eta}_0$ is a constant. It is now clear that high frequencies (large k) will be small, especially when t also get large, thus $\epsilon\partial_{xx}\eta(x,t)$ can be considered a diffusive term. The instabilities presented in this chapter is now attempted diffused by modifying the surface wave equation into

$$\eta_t = \phi_z - \eta_x\phi_x + \epsilon\eta_{xx}$$

The goal then is to adjust ϵ such that ϵk^2 is large for large k as to dampen high frequencies but small for low frequencies such that it does not dampen the physical frequencies in the solution.

The diffusive terms did not work on the instability shown in figure 8.1. This is somewhat expected since adding a diffusing term (or any other type of term for that matter) will introduce another stability condition on Δx and Δt. This idea was further supported when the diffusive terms was added to the already stable scenario with an initial surface elevation. In that case the solution went from stable to unstable by adding the diffusive term, which suggest a stability condition regarding Δx and Δt. However, when increasing N_t both the initial surface elevation case just mentioned and the test run shown in figure 8.1 became stable when adding the diffusive term. This result is rather unhelpful since the instability shown in 8.1 was already fixed by increasing N_t, therefore it is no point adding the diffusive term since it also requires to increase N_t to remain stable.

For instabilities caused by higher amplitude waves for incoming the diffusive term had little to no effect on the stability, unless N_t was to low in which case

made the instability occur earlier. For example running with the incoming fluid velocity

$$\partial_{\boldsymbol{n}}\phi_1(z,t) = -2.8e^{-6(t-2)^2}e^{-75(z+0.5)^2}$$

and otherwise the same parameters as the stable test run in section 7 that an instability occurred at $t = 1.92$. This is an instability occurring on an incoming wave where the wave amplitude gets to large. When adding adding the diffusive term $0.25\eta_{xx}$ the instability occurred at $t = 1.86$. When N_t was increased to $N_t = 5000$ the instability again occurred at $t = 2.01$. Increasing to $N_t = 8000$ delayed the instability to $t = 2.14$. With the substantial increase in runtime by increasing N_t adding the diffusive terms on unstable incoming waves was considered non-successful. The diffusive term did, however, have a successful effect on the case with initial surface elevation. This is curious and there seem to be a big difference between incoming waves and initial surface elevation which might be connected to the discovery in section 5.1.1 that there are higher inaccuracies around the non-smooth parts of C. It is therefore conjectured that improving the accuracies at the non-smooth parts of C will improve the stability for incoming waves.

An alternative view on surface elevation stability is explored in Appendix C

9

Concluding remarks

In this thesis surface waves in a wave tank was modeled using a boundary integral approach. It was discovered that the BIEs gave a mostly correct solution to the Laplace equation on the boundary, however having rather large error ($\sim 5 \times 10^{-2}$) close to non-smooth parts of the boundary. This was somewhat expected since the boundary integral relies on the directional derivative of the parametrized boundary, which is undefined at the sharp edged. Nonetheless the model was used to run with parameters based on a wave tank at the University of Oslo and it was observed that the model predicts reflections from the bottom profile used.

Two types of instabilities was explored. It was found that to keep the solutions stable it was required to satisfy a relationship between the discrete step size around the boundary and the discrete step size in time. It was concluded removing there instabilities using artificial diffusion was not helpful since by discretizing the diffusive term one is required to satisfy yet another relationship between the discrete step size around the boundary and the discrete step size in time. The second kind of instability was thought to come from to high surface elevations. It was observed that artificial diffusion had little to no effect on incoming waves with to large surface elevation.

For further work improving the models regarding inaccuracies around non-smooth boundary points should be a priority, and in that regard investigate the possibility for using perfectly matched layer to avoid reflecting waves from the left hand side of the channel. Further, exploring ways to optimize the numerical

implementation could prove beneficial. E.g. looking at alternative libraries for linear algebra, other parallelization techniques or general optimizing.

A

Green's functions

A.1 Green's function for Laplace operator

In this section a particular Green's function for the Laplace operator is found. With $\mathcal{L} = \nabla^2$ the Green's function, $\mathcal{G}(\boldsymbol{x} - \boldsymbol{\xi})$, must satisfy

$$\nabla^2 \mathcal{G}(\boldsymbol{x} - \boldsymbol{\xi}) = \delta(\boldsymbol{x} - \boldsymbol{\xi}) \tag{A.1}$$

To make notation easier the substitution $\mu := \boldsymbol{x} - \boldsymbol{\xi}$ is made. Thus if $\mathcal{G}(\mu)$ is a solution to the equation

$$\nabla^2 \mathcal{G}(\mu) = \delta(\mu) \tag{A.2}$$

then $\mathcal{G}(\boldsymbol{x} - \boldsymbol{\xi})$ is a solution to equation (A.1). The following computation can be justified using the theory of generalized functions[19], but the derivation here are done heuristically. Hence the Dirac delta function is assumed to have the following properties

$$\delta(\boldsymbol{x}) = \begin{cases} +\infty & , \quad \boldsymbol{x} = 0 \\ 0 & , \quad \boldsymbol{x} \neq 0 \end{cases}$$

$$\int_V \delta(\boldsymbol{x}) dV = 1, \quad 0 \in V$$

A condition for the Green's function in two dimensions will now be found. Consider a circular disk S_ϵ with radius ϵ centered at $\eta = 0$. Integrating equation

(A.2) over S_ϵ then gives

$$\begin{aligned}\int_{S_\epsilon} \nabla^2 \mathcal{G}(\mu) dV &= \int_{S_\epsilon} \delta(\mu) dV \\ \implies \int_{S_\epsilon} \nabla \cdot (\nabla \mathcal{G}(\mu)) dV &= 1 \\ \implies \oint_{C_\epsilon} \nabla \mathcal{G}(\mu) \boldsymbol{n} \cdot dS &= 1\end{aligned}$$

where $C_\epsilon = \partial S_\epsilon$ and $\boldsymbol{n}$ is a normal vector out of the circle C_ϵ. Taking the limit as $\epsilon \to 0$ gives the following constraint on $\mathcal{G}$

$$\lim_{\epsilon \to 0} \oint_{C_\epsilon} \nabla \mathcal{G}(\mu) \boldsymbol{n} \cdot dS = 1 \tag{A.3}$$

Now consider polar coordinate system with r and θ being the radial and angular coordinates respectively with solutions of $\mathcal{G}$ being rotationally invariant. That is $\mathcal{G}(r, \theta) = \mathcal{G}(r)$. For such solution equation (A.3) simplifies to

$$\begin{aligned}\lim_{\epsilon \to 0} \int_0^{2\pi} \epsilon \partial_r \mathcal{G}(\epsilon) d\theta &= 1 \\ \implies \lim_{\epsilon \to 0} \epsilon \partial_r \mathcal{G}(\epsilon) &= \frac{1}{2\pi}\end{aligned} \tag{A.4}$$

Now note that any solution to equation (A.2) also satisfy the equation

$$\nabla^2 \mathcal{G}(\mu) = 0 \quad \mu \neq 0 \tag{A.5}$$

Writing this equation in polar coordinates and constraining $\mathcal{G}$ to be rotational invariant equation (A.5) becomes

$$\begin{aligned}\frac{1}{r} \partial_r (r \partial_r \mathcal{G}(r)) &= 0, \quad r = 0 \\ \implies r \partial_r \mathcal{G}(r) &= C \\ \implies \partial_r \mathcal{G}(r) &= \frac{C}{r}\end{aligned} \tag{A.6}$$

where C is a constant. A solution for (A.6) is

$$\mathcal{G}(r) = C \ln(r)$$

Condition (A.4) now gives

$$\begin{aligned}\lim_{\epsilon \to 0} \epsilon \partial_r \left(C \ln(r) \right) |_{r=\epsilon} &= \frac{1}{2\pi} \\ \implies \lim_{\epsilon \to 0} \epsilon \left(\frac{C}{\epsilon} \right) &= \frac{1}{2\pi} \\ \implies C &= \frac{1}{2\pi}\end{aligned}$$

Hence a Green's function to the Laplace operator is

$$\mathcal{G}(r) = \frac{1}{2\pi}\ln(r)$$

which in case of Cartesian coordinates becomes

$$\mathcal{G}(\mu) = \frac{1}{2\pi}\ln(|\mu|)$$

Reverting back to the original problem then gives

$$\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) = \frac{1}{2\pi}\ln(|\boldsymbol{x}-\boldsymbol{\xi}|)$$

A.2 Boundary integral equation for the Laplace operator

To find a boundary integral equation an integral identity must first be found. With $\mathcal{L} = \nabla^2$ consider the following integral over space

$$\int_V \phi\nabla^2\psi dV$$

Using the identity $\nabla\cdot(\phi\nabla\psi) = \nabla\phi\cdot\nabla\psi + \phi\nabla^2\psi$ the above integral can be written as

$$\begin{aligned}
\int_V \phi\nabla^2\psi dV &= \int_V \left(\nabla\cdot(\phi\nabla\psi) - \nabla\phi\cdot\nabla\psi\right) dV \\
&= \oint_S \phi\nabla\psi\cdot\boldsymbol{n}dA - \int_V \nabla\phi\cdot\nabla\psi dV \\
&= \oint_S \phi\nabla\psi\cdot\boldsymbol{n}dA - \int_V \left(\nabla\cdot(\psi\nabla\phi) - \phi\nabla^2\psi\right) dV \\
&= \oint_S \phi\nabla\psi\cdot\boldsymbol{n}dA - \oint_S \psi\nabla\phi\cdot\boldsymbol{n}dA + \int_V \psi\nabla^2\phi dV
\end{aligned}$$

Now let $\phi = \phi(\boldsymbol{x})$ be a function of interest and $\psi = \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})$ be a Green's function of the Laplace operator the integral becomes

$$\int_V \phi(\boldsymbol{x})\nabla^2\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})dV = \oint_S \left(\phi(\boldsymbol{x})\nabla\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\cdot\boldsymbol{n} - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\nabla\phi(\boldsymbol{x})\cdot\boldsymbol{n}\right) dA + \int_V \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\nabla^2\phi(\boldsymbol{x})dV$$

The notation that $\nabla f\cdot\boldsymbol{n} = \partial_{\boldsymbol{n}} f$ will now be used. In our case the conservation of mass equation $\nabla^2\phi = 0$ can be used together with the fundamental property of

the Green's function $\mathcal{L}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) = \delta(\boldsymbol{x}-\boldsymbol{\xi})$ and the property $\int_V \phi(\boldsymbol{x})\delta(\boldsymbol{x}-\boldsymbol{\xi})dV = \phi(\boldsymbol{\xi})$ to write the integral as

$$\phi(\boldsymbol{\xi}) = \oint_S \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA \tag{A.7}$$

This is now an integral identity connecting points inside a domain to the points on the boundary. Since the goal is to get an identity for the connection between points on the boundary the point $\boldsymbol{\xi}$ is evaluated on the boundary $S = \partial V$ using equation (A.7). There is a complication with this approach however, and it's that when integrating over the boundary for a point $\boldsymbol{\xi} \in S$ the value $\mathcal{G}(\boldsymbol{\xi};\boldsymbol{\xi})$ must at some point be evaluated. Using the Green's function found in Appendix A.1 this means evaluating $\frac{1}{2\pi}\ln(|\boldsymbol{\xi}-\boldsymbol{\xi}|) = \frac{1}{2\pi}\ln(0)$ which is nonsense. To deal with this a *Cauchy principal value* integral is used. This is done in the following way: Let $\boldsymbol{\xi}$ a point on S that should be evaluated. The surface S is than deformed to form a small semi circle with radius ϵ around $\boldsymbol{\xi}$ called C_ϵ. The part of S with the semi circle removed is called S_ϵ, that is $\lim_{\epsilon\to 0} S_\epsilon \cup C_\epsilon = S$. This is all shown in figure A.1. The point $\phi(\boldsymbol{\xi})$ is then evaluated through the limiting

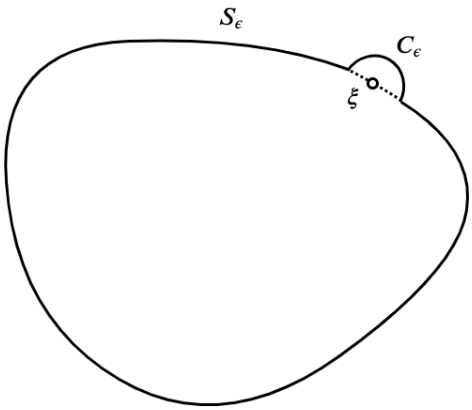


Figure A.1: Shows the deformed surface S_ϵ.

process

$$\begin{aligned}\phi(\boldsymbol{\xi}) &= \lim_{\epsilon\to 0}\int_{S_\epsilon\cup C_\epsilon} \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA \\ &= \text{PV}\oint_S \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA + \lim_{\epsilon\to 0} R_\epsilon\end{aligned} \tag{A.8}$$

where PV means that a Cauchy principal value integral should be used and R_ϵ is

$$R_\epsilon = \int_{C_\epsilon} \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA$$

To evaluate $\lim_{\epsilon \to 0} R_\epsilon$ the normal derivatives must now be analyzed. Since C_ϵ is a semi circle with radius ϵ centered at $\boldsymbol{\xi}$ the unit normal vector will be

$$\boldsymbol{n} = \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|}$$

The normal derivative of the Green's function will then be

$$\begin{aligned}
\partial_{\boldsymbol{n}} \mathcal{G}(\boldsymbol{x}; \boldsymbol{\xi}) &= \left(\nabla \frac{1}{2\pi} \ln(|\boldsymbol{x} - \boldsymbol{\xi}|) \right) \cdot \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|} \\
&= \left(\frac{1}{2\pi} \frac{1}{|\boldsymbol{x} - \boldsymbol{\xi}|} \nabla |\boldsymbol{x} - \boldsymbol{\xi}| \right) \cdot \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|} \\
&= \left(\frac{1}{2\pi} \frac{1}{|\boldsymbol{x} - \boldsymbol{\xi}|} \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|} \right) \cdot \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|} \\
&= \left(\frac{1}{2\pi} \frac{\boldsymbol{x} - \boldsymbol{\xi}}{|\boldsymbol{x} - \boldsymbol{\xi}|^3} \right) \cdot (\boldsymbol{x} - \boldsymbol{\xi})
\end{aligned}$$

Now note that for $\boldsymbol{x}$ on a circle with radius ϵ centered at $\boldsymbol{\xi}$ the radius can be expressed as

$$\epsilon = |\boldsymbol{x} - \boldsymbol{\xi}|$$

Also noting that $(\boldsymbol{x} - \boldsymbol{\xi}) \cdot (\boldsymbol{x} - \boldsymbol{\xi}) = |\boldsymbol{x} - \boldsymbol{\xi}|^2$ the normal derivate of the Green's function then becomes

$$\partial_{\boldsymbol{n}} \mathcal{G}(\boldsymbol{x}; \boldsymbol{\xi}) = \frac{1}{2\pi\epsilon}$$

Since C_ϵ is semi circle it is parametrized as

$$C_\epsilon(\theta) = \boldsymbol{\xi} + \epsilon(\cos(\theta), \sin(\theta)), \quad \theta \in [\theta_0, \theta_0 + \pi]$$

where θ_0 is chosen such that the semi circle starts a the point connection it to the rest of S_ϵ. The integral then becomes

$$R_\epsilon = \int_{\theta_0}^{\theta_0 + \pi} \left(\phi(C\epsilon(\theta)) \frac{1}{2\pi\epsilon} - \frac{1}{2\pi} \ln(\epsilon) \partial_{\boldsymbol{n}} \phi\left(C_\epsilon(\theta)\right) \right) \left| \frac{d}{d\theta} (C_\epsilon(\theta)) \right| d\theta$$

From the parametrization it is clear that

$$\begin{aligned}
\left| \frac{d}{d\theta} (C_\epsilon(\theta)) \right| &= |\epsilon(-\sin(\theta), \cos(\theta))| \\
&= \epsilon
\end{aligned}$$

thus the integral becomes

$$R_\epsilon = \int_{\theta_0}^{\theta_0 + \pi} \left(\phi(\boldsymbol{\xi} + \epsilon[\cos(\theta), \sin(\theta)]) \frac{1}{2\pi} - \epsilon \frac{1}{2\pi} \ln(\epsilon) \partial_{\boldsymbol{n}} \phi\left(C_\epsilon(\theta)\right) \right) d\theta$$

When $\epsilon \to 0$ the integral can be written as

$$R_\epsilon \sim \phi(\boldsymbol{\xi})\frac{1}{2\pi}\int_{\theta_0}^{\theta_0+\pi} d\theta - \epsilon\frac{1}{2\pi}\ln(\epsilon)\partial_{\boldsymbol{n}}\phi(\boldsymbol{\xi})\int_{\theta_0}^{\theta_0+\pi} d\theta, \quad \epsilon \to 0$$

$$\implies \lim_{\epsilon\to 0} R_\epsilon = \frac{1}{2}\phi(\boldsymbol{\xi})$$

since

$$\phi(\boldsymbol{\xi})\frac{1}{2\pi}\int_{\theta_0}^{\theta_0+\pi} d\theta = \phi(\boldsymbol{\xi})\frac{1}{2}$$

and

$$\lim_{\epsilon\to 0}\epsilon\frac{1}{2\pi}\ln(\epsilon)\partial_{\boldsymbol{n}}\phi(\boldsymbol{\xi})\int_{\theta_0}^{\theta_0+\pi} d\theta = \frac{1}{2}\partial_{\boldsymbol{n}}\phi(\boldsymbol{\xi})\lim_{\epsilon\to 0}\epsilon\ln(\epsilon) = 0$$

Thus equation (A.8) becomes

$$\phi(\boldsymbol{\xi}) = \text{PV}\oint_S \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA + \frac{1}{2}\phi(\boldsymbol{\xi})$$

which gives the final boundary integral equation

$$\frac{1}{2}\phi(\boldsymbol{\xi}) = \text{PV}\oint_S \left(\phi(\boldsymbol{x})\partial_{\boldsymbol{n}}\mathcal{G}(\boldsymbol{x};\boldsymbol{\xi}) - \mathcal{G}(\boldsymbol{x};\boldsymbol{\xi})\partial_{\boldsymbol{n}}\phi(\boldsymbol{x})\right) dA$$

B

Linear system for the the values of $\phi|_C$

Using the notation that the elements of $\mathbf{\Phi} = (\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0, \boldsymbol{\phi}_1, \boldsymbol{\phi}_2, \boldsymbol{\phi}_3)$ and $\mathbf{\Phi}^* = (\boldsymbol{\phi}_0, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_1, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_2, \partial_{\boldsymbol{n}}\boldsymbol{\phi}_3)$ are vectors and that A_{ij} and B_{ij} are matrices containing the values A_{ij}^{lk} and B_{ij}^{lk} respectively, equation (3.6) for all j can be written as

$$\frac{1}{2}I_{00}\boldsymbol{\phi}_0 = A_{00}\boldsymbol{\phi}_0 - B_{00}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 + A_{10}\boldsymbol{\phi}_1 - B_{10}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 + A_{20}\boldsymbol{\phi}_2 - B_{20}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 + A_{30}\boldsymbol{\phi}_3 - B_{30}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$\frac{1}{2}I_{11}\boldsymbol{\phi}_1 = A_{01}\boldsymbol{\phi}_0 - B_{01}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 + A_{11}\boldsymbol{\phi}_1 - B_{11}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 + A_{21}\boldsymbol{\phi}_2 - B_{21}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 + A_{31}\boldsymbol{\phi}_3 - B_{31}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$\frac{1}{2}I_{22}\boldsymbol{\phi}_2 = A_{02}\boldsymbol{\phi}_0 - B_{02}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 + A_{12}\boldsymbol{\phi}_1 - B_{12}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 + A_{22}\boldsymbol{\phi}_2 - B_{22}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 + A_{32}\boldsymbol{\phi}_3 - B_{32}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$\frac{1}{2}I_{33}\boldsymbol{\phi}_3 = A_{03}\boldsymbol{\phi}_0 - B_{03}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 + A_{13}\boldsymbol{\phi}_1 - B_{13}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 + A_{23}\boldsymbol{\phi}_2 - B_{23}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 + A_{33}\boldsymbol{\phi}_3 - B_{33}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

where I_{jj} is the $N_j \times N_j$ identity matrix. Putting all the unknown values in the left hand side, and the known values on the right hand side gives

$$B_{00}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 - A_{10}\boldsymbol{\phi}_1 - A_{20}\boldsymbol{\phi}_2 - A_{30}\boldsymbol{\phi}_3 = \left(-\frac{1}{2}I_{00} + A_{00}\right)\boldsymbol{\phi}_0 - B_{10}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 - B_{20}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 - B_{30}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$B_{01}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 + \left(\frac{1}{2}I_{11} - A_{11}\right)\boldsymbol{\phi}_1 - A_{21}\boldsymbol{\phi}_2 - A_{31}\boldsymbol{\phi}_3 = -A_{01}\boldsymbol{\phi}_0 - B_{11}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 - B_{21}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 - B_{31}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$B_{02}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 - A_{12}\boldsymbol{\phi}_1 + \left(\frac{1}{2}I_{22} - A_{22}\right)\boldsymbol{\phi}_2 - A_{32}\boldsymbol{\phi}_3 = -A_{02}\boldsymbol{\phi}_0 - B_{12}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 - B_{22}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 - B_{32}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

$$B_{03}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_0 - A_{13}\boldsymbol{\phi}_1 - A_{23}\boldsymbol{\phi}_2 + \left(\frac{1}{2}I_{33} - A_{33}\right)\boldsymbol{\phi}_3 = -A_{03}\boldsymbol{\phi}_0 - B_{13}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_1 - B_{23}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_2 - B_{33}\partial_{\boldsymbol{n}}\boldsymbol{\phi}_3$$

This can now be written as a matrix system where A_{ij} and B_{ij} are block matrices

$$\begin{bmatrix} B_{00} & -A_{10} & -A_{20} & -A_{30} \\ B_{01} & \frac{1}{2}I_{11} - A_{11} & -A_{21} & -A_{31} \\ B_{02} & -A_{12} & \frac{1}{2}I_{22} - A_{22} & -A_{32} \\ B_{03} & -A_{13} & -A_{23} & \frac{1}{2}I_{33} - A_{33} \end{bmatrix} \boldsymbol{\Phi} = \begin{bmatrix} -\frac{1}{2}I_{00} + A_{00} & -B_{10} & -B_{20} & -B_{30} \\ A_{01} & -B_{11} & -B_{21} & -B_{31} \\ A_{02} & -B_{12} & -B_{22} & -B_{32} \\ A_{03} & -B_{13} & -B_{23} & -B_{33} \end{bmatrix} \boldsymbol{\Phi}^*$$

C

A somewhat related system of ODEs

In section 8.2 it is shown that the numerical solution is unstable for cases where the wave amplitude gets lo large. In this chapter an unconventional view on the source of the instabilities is taken.

C.1 Motivation

Consider the surface wave equations

$$\eta_t = \phi_z - \eta_x \phi_x$$
$$\phi_t = -\frac{1}{2}\left(\phi_x^2 + \phi_z^2\right) - g\eta$$

When these equations are discretized they can be viewed as a system of coupled ODEs. Given that the partial derivatives are approximated using some finite difference formula using neighboring points this system will be in the

form

$$\begin{aligned}
&\vdots \\
(\eta_i)_t &= f(\eta_{i-1}, \eta_i, \eta_{i+1}, \phi_{i-1}, \phi_i, \phi_{i+1}) \\
(\phi_i)_t &= h(\eta_{i-1}, \eta_i, \eta_{i+1}, \phi_{i-1}, \phi_i, \phi_{i+1}) \\
&\vdots
\end{aligned}$$

where f and h are functions based on the original system of PDEs and the specific approximation used to calculate the partial derivatives. It is now clear that the partial derivatives in space will couple the ODEs together. Now consider the somewhat strange modification to the original system of PDEs

$$\eta_t = \phi - \eta\phi \tag{C.1}$$
$$\phi_t = -\phi^2 - g\eta \tag{C.2}$$

Here η and ϕ are functions of a single variable t. This system of ODEs are related to the original system of PDEs in that the partial derivatives are replaced by the function that was differentiated, so for example $\phi_x \longrightarrow \phi$. This system of ODEs are related also related to the discretization of the PDE system in that if one assumes that $\Delta x = \Delta z$ and for example the partial derivatives are discretized using central difference then f and h will be the form

$$\begin{aligned}
f(\eta_{i-1}, \eta_i, \eta_{i+1}, \phi_{i-1}, \phi_i, \phi_{i+1}) &= \phi_i - \eta_i\phi_i + C_1 \\
h(\eta_{i-1}, \eta_i, \eta_{i+1}, \phi_{i-1}, \phi_i, \phi_{i+1}) &= -\phi_i^2 - g\eta_i + C_2
\end{aligned}$$

where C_1 and C_2 are the coupling terms, thus equation (C.1) and (C.2) will exist in the discretization of the PDE system, in fact it will be as many as discrete points of η and ϕ. Now if the coupling terms are small equation (C.1) and (C.2) might give some insight into the stability of the discretized. There are of course a lot of assumptions leading to this analysis, but it was considered an academic curiosity and put here in the appendix.

C.2 Analytical solution

Through some luck it was found that it was actually possible to find an analytical solution to equation (C.1) and (C.2). This was done through the following process: First we want to remove the parameter g from the system of ODEs.

Differentiating equation (C.2) in time gives

$$\begin{aligned}
\phi_{tt} &= -2\phi\phi_t - g\eta_t \\
&= -2\phi\phi_t - g(\phi - \eta\phi) \\
&= -2\phi\phi_t - g\left(\phi - \frac{1}{g}(-\phi^2 - \phi_t)\phi\right) \\
&= -2\phi\phi_t - g\phi - \phi^3 - \phi\phi_t \\
\implies \phi_{tt} &= -3\phi\phi_t - g\phi - \phi^3
\end{aligned} \tag{C.3}$$

A change of variables is now chosen as

$$\phi(t) = \sqrt{g}\tilde{\phi}[\sqrt{g}t]$$

Inserting into equation (C.3) gives

$$\left(\sqrt{g}\frac{d^2}{dt^2}\tilde{\phi}[\sqrt{g}t]\right) = -3\sqrt{g}\tilde{\phi}[\sqrt{g}t]\left(\sqrt{g}\frac{d}{dt}\tilde{\phi}[\sqrt{g}t]\right) - g\sqrt{g}\tilde{\phi}[\sqrt{g}t] - (\sqrt{g}\tilde{\phi}[\sqrt{g}t])^3$$

$$\tilde{\phi}''(\sqrt{g}t)g^{\frac{3}{2}} = -3\tilde{\phi}[\sqrt{g}t]\tilde{\phi}'[\sqrt{g}t]g^{\frac{3}{2}} - g^{\frac{3}{2}}\tilde{\phi}[\sqrt{g}t] - g^{\frac{3}{2}}\tilde{\phi}[\sqrt{g}t]^3$$

Multiplying with $g^{\frac{2}{3}}$ on both sides and scaling t as $\tilde{t} = \sqrt{g}\,t$ finally gives

$$\tilde{\phi}_{\tilde{t}\tilde{t}} = -3\tilde{\phi}\tilde{\phi}_{\tilde{t}} - \tilde{\phi} - \tilde{\phi}^3 \tag{C.4}$$

In a conversation with Dennis The[20] it was found through a computation in Maple that equation (C.4) had the same number of symmetries as the equation

$$y''(t) = 0$$

which motivated to try using Maple to compute an analytical solution. The analytical solution was then found to be

$$\tilde{\phi}(\tilde{t}) = \frac{b\cos\left(\tilde{t}\right) - \sin\left(\tilde{t}\right)}{a + b\sin\left(\tilde{t}\right) + \cos\left(\tilde{t}\right)}$$

where a and b are arbitrary constants. Using the definition of $\tilde{\phi}$ and $\tilde{t}$ gives

$$\phi(t) = \sqrt{g}\frac{b\cos\left(\sqrt{g}t\right) - \sin\left(\sqrt{g}t\right)}{a + b\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right)}$$

Plugging this solution into equation (C.1) a solution for $\eta(t)$ can be found. The solution was again found using Maple and was found to be

$$\eta(t) = \frac{c + \cos\left(\sqrt{g}t\right) + b\sin\left(\sqrt{g}t\right)}{a + b\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right)}$$

where c is an arbitrary constant. The goal now i to match the constants with the initial conditions such that one can find a constraint on the initial conditions that gives bounded solutions of the ODE system. Notice, however, that only two equations can be made using the initial conditions, but there are three constants. One of these constants are introduced through differentiating equation (C.2). By this step one might introduce false solutions. To see this consider the equation

$$\frac{d}{dt}\left(f(t)\right) = 0$$

where $f(t)$ is some differentiable function of t. It is clear that by adding any constants to the function f the equation will still hold. Therefore one must make sure that any additional constant introduced through this step satisfy the equation. The equation

$$\frac{d}{dt}\left(\phi_t + \phi^2 + g\eta\right) = 0$$

will now be used to determine the third constant c. By substituting the solution into this equation we get

$$-cg^{\frac{3}{2}} \frac{b\cos\left(\sqrt{g}t\right) - \sin\left(\sqrt{g}t\right)}{\left(a + b\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right)\right)^2} = 0$$

It is clear that if this equation should hold for all t it must be required that $c = 0$. The constants can now be matched with the initial conditions as follows

$$\phi(0) = \sqrt{g}\frac{b}{a+1}$$
$$\eta(0) = \frac{1}{a+1}$$

With the notation $\eta_0 := \eta(0)$ and $\phi_0 := \phi(0)$ it gives

$$a = \frac{1}{\eta_0} - 1$$
$$b = \frac{1}{\sqrt{g}}\frac{\phi_0}{\eta_0}$$

From the solutions of η and ϕ it is clear that they are unbounded if the denominator becomes 0 for any $t \geq 0$. The condition for unbounded solution can therefore be found by finding for which initial condition the equation

$$\left(\frac{1}{\eta_0} - 1\right) + \frac{1}{\sqrt{g}}\frac{\phi_0}{\eta_0}\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right) = 0$$

has a solution in t. By defining $\frac{1}{\sqrt{g}}\frac{\phi_0}{\eta_0} = \tan(\alpha) = \frac{\sin(\alpha)}{\cos(\alpha)}$ the equation can be written as

$$\frac{\sin(\alpha)}{\cos(\alpha)}\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right) = \left(1 - \frac{1}{\eta_0}\right)$$
$$\implies \frac{1}{\cos(\alpha)}\left[\sin(\alpha)\sin\left(\sqrt{g}t\right) + \cos\left(\sqrt{g}t\right)\cos(\alpha)\right] = \left(1 - \frac{1}{\eta_0}\right)$$
$$\implies \cos\left(\sqrt{g}t - \alpha\right) = \left(1 - \frac{1}{\eta_0}\right)\cos(\alpha)$$

It is now clear the this equation only has a solution for t if

$$\left|\left(1 - \frac{1}{\eta_0}\right)\cos(\alpha)\right| \leq 1$$

By using the definition $\alpha = \arctan\left(\frac{\phi_0}{\sqrt{g}\eta_0}\right)$ and the fact that $\cos(\arctan(\alpha)) = \frac{1}{\sqrt{1+\alpha^2}}$ the condition becomes

$$\left|\left(1 - \frac{1}{\eta_0}\right)\frac{1}{\sqrt{1 + \left(\frac{\phi_0}{\sqrt{g}\eta_0}\right)^2}}\right| \leq 1$$

In figure C.1 the phase space of the ODE system is plotted together with its stream plot. The stability condition was used to determine the stable and unstable regions of the phase space. From figure C.1 it is clear that there are

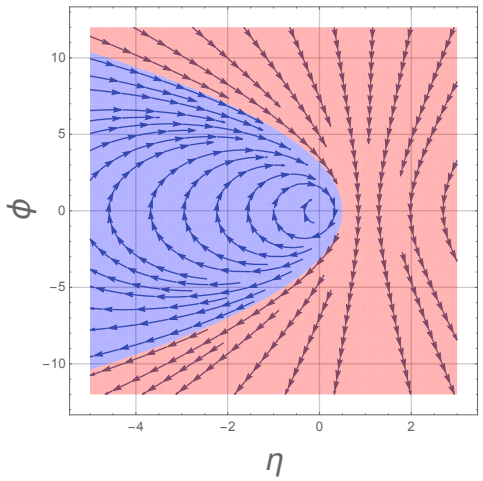


Figure C.1: Shows the stable (blue) and unstable (red) regions in the phase space of ODE system.

exist some η and ϕ that leads to unbounded solutions. Note that positive values

for η corresponds to positive surface elevation, therefore this could indicate that the discrete coupling terms that comes from the PDE system could be the source of instabilities when η gets to large. However, no direct connection was found between the unstable threshold for η found here and the actual surface elevation that lead to instabilities. This is of course no surprise since one would expect the coupling terms in the discretized PDE system to effect the exact surface elevation that will cause on instability. Even though this analysis is far from general for the instabilities there are a case where it is somewhat valid. If there is an initial surface elevation which is very flat on the top such that the spatial derivatives are very small it can be argued that the coupling terms are indeed small.

Bibliography

[1] M. S. Longuet-Higgins and E. Cokelet, "The deformation of steep surface waves on water-i. a numerical method of computation," *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, vol. 350, no. 1660, pp. 1–26, 1976.

[2] T. Vinje and P. Brevig, "Numerical simulation of breaking waves," *Advances in Water resources*, vol. 4, no. 2, pp. 77–82, 1981.

[3] S. T. Grilli and R. Subramanya, "Quasi-singular integrals in the modeling of nonlinear water waves in shallow water," *Engineering Analysis with Boundary Elements*, vol. 13, no. 2, pp. 181 – 191, 1994. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0955799794900205

[4] S. Jorde, "Kinematikken i bølger over en grunne," 2018.

[5] A. Raustøl, "Freake bølger over variabelt dyp," 2014.

[6] E. Saff and A. Snider, *Fundamentals of Complex Analysis with Applications to Engineering and Science*, 3rd ed. Pearson Education, Inc, 2003, p. 79.

[7] A. Janssen, "A complex contour based perfectly matched layer applied to a pattern generating model equation," 2017.

[8] "Boost c++ libraries," https://www.boost.org/, accessed: 25. May, 2020.

[9] "Intel math kernel libraries," https://software.intel.com/content/www/us/en/develop/tools/math-kernel-library.html, accessed: 25. May, 2020.

[10] T. Sauer, *Numerical analysis, 2nd edn. George Mason University*. Pearson Education Inc, Upper Saddle River. ISBN-10, 2012, p. 316.

[11] W. Barry, *Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers*, 2nd ed. Pearson Education India, 2006, pp. 354, 8.

[12] "Openmp," https://www.openmp.org/, accessed: 25. May, 2020.

[13] "Open mpi: Open source high performance computing," https://www.open-mpi.org/, accessed: 25. May, 2020.

[14] "Intel® xeon® platinum 8168 processor," https://ark.intel.com/content/www/us/en/ark/products/120504/intel-xeon-platinum-8168-processor-33m-cache-2-70-ghz.html, accessed: 25. May, 2020.

[15] "Iterative sparse solvers based on reverse communication interface (rci iss)," https://software.intel.com/content/www/us/en/develop/documentation/mkl-developer-reference-c/top/sparse-solver-routines/iterative-sparse-solvers-based-on-reverse-communication-interface-rci-iss.html, accessed: 29. May, 2020.

[16] "Openmp threaded functions and problems," https://software.intel.com/content/www/us/en/develop/documentation/mkl-linux-developer-guide/top/managing-performance-and-memory/improving-performance-with-threading/openmp-threaded-functions-and-problems.html, accessed: 25. May, 2020.

[17] O. Gramstad, H. Zeng, K. Trulsen, and G. Pedersen, "Freak waves in weakly nonlinear unidirectional wave trains over a sloping bottom in shallow water," *Physics of Fluids*, vol. 25, no. 12, p. 122103, 2013.

[18] N. D. Katopodes, *Free-Surface Flow: Computational Methods*. Butterworth-Heinemann, 2018.

[19] E. Zauderer, *Partial differential equations of applied mathematics*. John Wiley & Sons, 2011, vol. 71, pp. 425–429.

[20] D. The, "Associate professor at the university of tromsø," personal communication.