NUMERICAL SIMULATION OF NONLINEAR WATER WAVES USING A PANEL METHOD; DOMAIN DECOMPOSITION AND APPLICATIONS

ISBN 90-3651038-4

Druk: FEBODRUK BV, Enschede
Omslagontwerp: Karbeel, Veldhoven

NUMERICAL SIMULATION OF NONLINEAR WATER WAVES USING A PANEL METHOD; DOMAIN DECOMPOSITION AND APPLICATIONS

PROEFSCHRIFT

ter verkrijging van
de graad van doctor aan de Universiteit Twente,
op gezag van de rector magnificus,
prof.dr. F.A. van Vught,
volgens besluit van het College voor Promoties
in het openbaar te verdedigen
op vrijdag 10 oktober 1997 te 13.15 uur.

door

Paulus Cornelis Antonius de Haas

geboren op 27 augustus 1966

te Someren

Dit proefschrift is goedgekeurd door de promotor
prof.dr.ir. P.J. Zandbergen.

Acknowledgements

This thesis is written as part of a research project, started in 1993, aimed at further developing a computer program for the numerical simulation of nonlinear water waves. It was financed by the Dutch Technology Foundation (STW) under grant TWI22.2724 and was supervised by prof. Zandbergen, Twente University, Faculty of Applied Mathematics. It was the continuation of earlier research projects by Romate, Broeze and van Daalen (see Chapter 1 and references therein) and was carried out simultaneously with the work by Berkvens which took place under the same grant. The research was carried out at Delft Hydraulics, first in 'de Voorst', near Marknesse and later in Delft. Computations were done on a Cray YMP and later on a Cray C-98/4256 supercomputer at the Academic Computing Services Amsterdam (SARA). Financial support for these computing systems was supported by the National Computing Facilities Foundation of the Netherlands (NCF). Computations were also done on a cluster of HP-workstations and a Silicon Graphics Power Challenge at Twente University.

During the years at Delft Hydraulics, I was supported by fine colleagues which took great interest in the research project. Especially Ad Reniers and Maarten Dingemans were very helpfull to me. I would like to address a special word of thank to Ashwini Otta with whom I had very stimulating conversations. I very much appreciate the presence of room mate Gert Klopman who helped me with all matters, ranging from software to fundamentals.

The Maritime Research Institute Netherlands (MARIN) functioned as counterpart of Delft Hydraulics in the research project. The representation by Rene Huijsmans and Hoite Raven was a great stimulant to the research. My own counterpart at MARIN was Patrick Berkvens who carried out the research on the numerical model especially in relation with floating bodies. I would like to thank him for the nice cooperation during the research and his devotion to our work.

There was close cooperation with the Applied Analysis group of the Faculty of Applied Mathematics of Twente University. On the topic of parallel computing I cooperated with Martin Streng and Patrick Strating who strongly contributed to the results presented in Chapter 8 of this thesis. Furthermore

I would like to thank Douwe Dijkstra and especially Bernhard Müller for the carefull review of the manuscript. I was lucky having the predecessors of the research project around; the support of Ed van Daalen and Jan Broeze was very welcome and I would like to thank them for their continuing interest in the work. The constant factor behind it all was the presence and guidance of prof. Zandbergen. I appreciate the way he encouraged me and kept me on the right track.

Among my personal friends I would like to thank all those who took interest in my work, especially my parents and Jeanette.

Delft, september 1997 *Paul de Haas*

Contents

Abstract

In studying the influence of water waves on constructions such as dikes, wave breakers and offshore constructions, on ships but also on natural processes such as sediment transport and changes in bottom topography, more and more use is made of numerical models. An important class of such models consists of models in which the flow is described by potential theory. On the one hand the assumptions made in potential theory are valid in many studies, on the other hand the description - its field equation is Laplace's equation for the velocity potential - offers many possibilities for finding solutions with numerical models.

The panel method is a numerical method which makes use of a boundary integral formulation for Laplace's equation, so that only the boundaries of the fluid domain have to be covered with grid points. Moreover this enables a natural description of the movement of the free surface in the time domain, which is determined by nonlinear dynamical and kinematical boundary conditions. Nonlinearity of the free-surface boundary conditions is often of importance in studying the influence of waves in coastal and ocean engineering.

In this thesis a two-dimensional and three-dimensional numerical model are studied, based on a panel method, for the description of nonlinear water waves. The focus is on two important aspects: firstly the dependence of the computational effort on the number of grid points and secondly some specific numerical difficulties which arise when the method is used in application-like computations. With respect to the former aspect, a domain decomposition technique is studied. The latter aspect is studied for some examples and the suitability and limitations of some parts of the method for these examples are investigated.

In more detail the contents of this thesis is as follows. For the domain decomposition technique, an iterative method is chosen in which the domain is divided in the horizontal direction. The length-to-height ratios of the subdomains, among other things, determine the convergence of the iterative method. Because the domains in problems involving water waves generally have large length-to-height ratios, relatively many subdomains can be chosen with a limited loss of convergence. As a consequence the panel method can be applied much more efficiently with domain decomposition. In the case of subdomains with fixed length-to-height ratios, the computational costs per time step depend

at most linearly on the length of the domain.

Furthermore the approriateness of the numerical model is studied for three types of problems. Firstly the use of the model for the simulation of waves generated by a translating or rotating wavemaker is studied. An important improvement for such computations is the use of descriptions in the physical domain rather than in the computational domain for extrapolations towards lateral boundaries such as the wavemaker boundary. Secondly the propagation of wave groups is simulated and the use of various formulations of the wave group signal is studied. The domain decomposition technique proves to be a succesful method for these computations. Thirdly the approriateness of the numerical model for the simulation of waves diffracting around a surface-piercing construction is studied. Because of the large differences in horizontal velocities on the free surface, the use of a mixed Eulerian-Lagrangian description is necessary.

Samenvatting

Bij de bestudering van de invloed van golven op constructies zoals dijken, golfbrekers en offshoreconstructies, op schepen maar ook op natuurlijke processen zoals sedimenttransport en bodemvervorming, wordt steeds meer gebruik gemaakt van numeriek wiskundige modellen. Een belangrijke klasse van zulke modellen wordt gevormd door modellen waarbij de waterbeweging onder invloed van golven wordt beschreven door een potentiaalstroming. Enerzijds zijn de aannamen die voor een potentiaalstroming gemaakt worden in veel situaties gerechtvaardigd, anderzijds biedt de beschrijving - een Laplace-vergelijking voor de snelheidspotentiaal in het vloeistofgebied - veel mogelijkheden tot het vinden van oplossingen met numerieke modellen.

De panelenmethode is een numerieke methode die gebruikt maakt van een randintegraalformulering voor de Laplace-vergelijking zodat alleen de rand van het vloeistofgebied belegd hoeft te worden met rekenpunten. Daardoor is bovendien een natuurlijke beschrijving van de beweging van het vrije oppervlak in het tijdsdomein mogelijk, welke bepaald is door niet-lineaire dynamische en kinematische randvoorwaarden. Deze niet-lineariteit is vaak belangrijk bij de bestudering van de invloed van golven.

In dit proefschrift worden een tweedimensionaal en een driedimensionaal numeriek model bestudeerd, gebaseerd op een panelenmethode, voor de beschrijving van niet-lineaire golven op water. Speciale aandacht is er voor twee aspecten: de relatie tussen aantal rekenpunten en benodigde rekeninspanning, en enkele specifieke numerieke moeilijkheden die optreden bij toepassingsgerichte berekeningen. Met betrekking tot het eerstgenoemde aspect wordt een domeindecompositie techniek bestudeerd. Het laatstgenoemde aspect wordt bestudeerd aan de hand van een aantal voorbeelden. Daarmee worden de geschiktheid en de beperkingen van een aantal onderdelen van het numerieke model aangetoond.

Meer concreet laat de inhoud van dit proefschrift zich als volgt beschrijven. Voor de domeindecompositietechniek is een iteratieve methode gekozen waarbij het rekengebied in de horizontale richting wordt opgedeeld. De convergentie van de iteratieve methode wordt onder meer bepaald door de lengte-hoogteverhouding van de subdomeinen. Aangezien het rekengebied bij golfproblemen in het algemeen een grote lengte-hoogteverhouding heeft kunnen

relatief veel subdomeinen genomen worden met slechts een gering verlies aan convergentie. Voor de panelenmethode betekent dit dat door het gebruik van domeindecompositie aanzienlijke rekenwinsten geboekt kunnen worden. Wanneer subdomeinen met een vaste lengte-hoogte verhouding gebruikt worden, hangen de rekenkosten per tijdsstap hoogstens lineair af van de lengte van het rekengebied.

In een aantal toepassingen wordt verder naar de geschiktheid van het numerieke model voor dit soort problemen gekeken. Ten eerste betreft dit het gebruik van het model voor de simulatie van golven opgewekt door een translerend of een roterend golfschot. Hierbij blijkt dat een beschrijving in het fysische domein meer geschikt is dan een beschrijving in het rekendomein voor extrapolaties naar zijranden, zoals de rand gevormd door een golfschot. Ten tweede wordt de voortplanting van golfgroepen gesimuleerd voor verschillende formuleringen van het golfgroepsignaal. Het gebruik van de domeindecompositietechniek is daarbij zeer voordelig. Ten derde wordt de geschiktheid voor de simulatie van golven rond een constructie bestudeerd, welke door het vrije oppervlak heen steekt. Door de grote verschillen in horizontale snelheden van het vrije oppervlak is het gebruik van een gemengd Eulers-Lagrangiaanse beschrijving noodzakelijk.

Chapter 1

Introduction

1.1 Context of the work

In the field of ocean and coastal engineering much attention is paid to the description and prediction of water waves. The water motion induced by waves is of importance for e.g. the run-up and forces on coastal structures, forces on and motion of offshore structures and ships and for the transport of sediment. Knowledge about the structure of the wave field and the associated flow field is often used as input for studies related to these problems.

Different approaches can be distinguished towards solving an engineering problem involving water waves. A first approach is the use of a physical model. The subject of interest is built (on scale) in the laboratory and examined by generating waves with machine-driven wavemakers for a number of different wave conditions. An advantage of physical modelling is that many aspects of the problem can be taken into account. Difficulties lie in the proper scaling of the problem and doing accurate measurements of certain properties. Only point measurements can be made and one has to avoid disturbances due to the instruments. Moreover, physical modelling has become relatively expensive.

A second approach is the use of mathematical models. Based on fundamental conservation laws of physics, equations can be derived that describe the motion of water for the problem under consideration. By simplifying the equations and/or approximating the solution of these equations, information can be obtained. The description of waves is a classical subject of mathematical physics and has been studied since the 17th century. Since then the mathematical theory has developed enormously but only for a small number of water wave problems satisfactory solutions can be obtained directly from mathematical formulations.

A third approach, which is applied often in engineering problems, is the use of numerical models to solve the equations describing the problem. Mathematical models can be solved numerically for some specific wave conditions and

configurations, providing answers which are very hard or impossible to obtain with analytical methods. Numerical models have become more and more popular because of the growth of computational power and the advance of numerical techniques.

An important aspect of the description of water waves is its nonlinear character. For high or steep waves nonlinear effects become important. Up to the seventies, these waves were mainly described using perturbation series. With the rise of powerful computers it has also become possible to choose a numerical approach for nonlinear models. An important class of numerical models for nonlinear waves is based on boundary integral equation methods. In these models, the fully nonlinear boundary conditions for the free surface are used and the field equation for a potential flow is solved using a method for a corresponding boundary integral equation. In these methods only grid points on the boundaries of the fluid domain are needed.

In this thesis a particular numerical model involving a boundary integral equation method is studied. It is able to describe three-dimensional potential flow using the fully nonlinear free-surface conditions in problems with varying bottom topography. At present its use is mainly limited by the large computational demands still involved with the model. Also the limited experience and verification of problems other than standard theoretical problems prevent its use in practical engineering. We will pay attention to both aspects in this thesis.

1.2 Models for nonlinear waves

We make a distinction between two different interests in the use of nonlinear models. Firstly, the interest in the prediction of wave propagation influenced by wave-wave, wave-bottom, wave-wind or wave-current interaction. Secondly, the interest in water motion in nonlinear waves locally near structures and ships. The use of numerical models in both fields of interest is discussed next.

In the field of models for wave propagation, a first classification can be made based on which area is studied: oceans, offshore or nearshore seas or harbours. Furthermore it is important to identify which physical processes are important. For an overview of this matter see Battjes [4]. Especially in the nearshore region and in harbours there is a large class of problems for which the assumption of potential flow can be made and attention is paid to the effect of the nonlinear free-surface boundary conditions on the characteristics of the wave field. Various models have been developed for this purpose. Here we mention the mild-slope equation and the Boussinesq models. There have been important developments in these methods during the last few decades, both in

the mathematical formulation and in the numerical implementation.

In the field of models for local water motion near structures or around ships, there is a large diversity of models with respect to the assumptions made about the flow. Viscosity or turbulence may be taken into account which has large implications on which numerical technique has to be used. If the characteristic dimension of an object is of the order of the wave length, viscous effects can be neglected for the bulk area of the domain and the oscillatory part of the motion. For this type of problems, boundary integral equation methods are often used. If viscosity is important, other methods have to be used such as finite volume methods or marker-and-cell methods. These methods demand much more computational effort and therefore very few three-dimensional methods have been reported so far, see e.g. Alessandrini [2]. Also the treatment of the moving free surface is more complicated in these methods.

The role of the present potential flow model for both types of problems can be indicated as follows. For wave propagation it can provide the verification of other models with more assumptions and approximations such as Boussinesq-type models because it uses the fully nonlinear free-surface conditions. For local water motion the model is valid for a certain class of problems. For problems outside this class it can provide boundary conditions for models which are more appropriate. Examples of such problems are shown in this thesis.

1.3 Developments

The model which is studied in this thesis has been developed by Romate [58] and Broeze [13] since the beginning of 1983. As mentioned before, it concerns a three-dimensional model in which also arbitrary bottom topographies can be used. Moreover the method is suitable for computing the motion of floating bodies as shown by van Daalen [18]. A large number of applications can be handled by the model of which a few are discussed in this thesis.

Comparisons with developments elsewhere show that the model has always been competitive on an international level. A short review of present day models is given by Kim [42]. To place the present work in relation to the preceding work on the model, a short overview is given of the work by Romate, Broeze and van Daalen.

Romate [58] investigated the choice of a numerical method for nonlinear free-surface waves and built the model on which the present model is based. Studies on accuracy and stability and on aspects such as absorbing boundary conditions were performed. The model was shown to perform well for mildly nonlinear waves.

Broeze [13] continued the studies on accuracy and stability and improved these up to a level at which highly nonlinear waves could be simulated over a

number of wave periods. Also computations in more complex domains with uneven bottoms were presented including the computation of a plunging breaking wave in three dimensions.

Van Daalen [18] constructed the model for the simulation of floating bodies in water and showed results of computations with circular and rectangular cylinders in two dimensions. The three-dimensional version has been developed by Berkvens who showed that it can simulate motions of an ellipsoid floating in a free surface [9].

Although considerable progress has been made, little attention has yet been paid to applications. An important reason for this is the large computation time associated with such simulations. Also the lack of acquaintance with the accuracy and stability for problems other than model problems is related to this (although Broeze [13] has made some comparisons with experiments). In this thesis the focus is on these two aspects and especially on domain decomposition as a means to reduce computational costs.

An important goal which is kept in sight, now and in the near future, is the possibility to simulate the motion of ships moored in a harbour or sailing at sea. It is obvious that, although the preceding work includes some important steps in that direction, still a lot of work remains to be done.

1.4 Outline of the thesis

This thesis can be divided into three parts. The first part describes the model developed by Romate and Broeze and features of the numerical method are investigated closer and improved. The second part describes the domain decomposition method which is shown to improve the efficiency of the model. The third part contains three studies in which the model is applied to relevant nonlinear wave problems. The division into chapters is as follows:

In Chapter 2 the mathematical formulation for nonlinear waves is described which is solved in the numerical model. First and second-order Stokes' wave theory is discussed to provide reference for later chapters. In Chapter 3 the numerical method is described and some important features are pointed out. The accuracy and stability of the method is examined in Chapter 4.

The second part, about domain decomposition, starts with an introduction in Chapter 5. In Chapter 6 convergence results are derived for the domain decomposition method chosen for solving Laplace's equation. Next, in Chapter 7, the implementation into the time-domain numerical method is described and effects on accuracy and stability are discussed. Results on efficiency are shown using approximative formulas and some examples. In Chapter 8 it is shown that the domain decomposition method can easily be implemented on a parallel system and results on parallel efficiency are presented for two systems.

Finally the extensions described previously are applied in typical water wave problems which show the present capabilities of the method. In Chapter 9 computations of the two-dimensional model are discussed which show the usefulness of the model as a verification tool. In Chapter 10 the simulation of propagating two-dimensional wave groups over large distances is presented. Emphasis is on the use of domain decomposition and on the stability of wave signals. Chapter 11 describes the three-dimensional simulation of waves around a surface piercing cylinder. Special attention is paid to stability. In Chapter 12 the main conclusions are given and recommendations for future investigations are stated.

Chapter 2

Mathematical formulation

2.1 Introduction

The mathematical formulation of the nonlinear water wave problem is derived from fundamental conservation laws. It consists of equations which are valid in the fluid domain and of equations which are valid on the boundaries only. Together they define the nonlinear wave problem in which the nonlinearity comes from the free-surface boundary condition.

Exact solutions of these equations for general wave problems are impossible to obtain and therefore approximative equations are often used. Many different wave theories have been developed in the past, each with its own range of validity. The solutions and properties of the equations used in these wave theories can provide valuable insight in the solution of the fully nonlinear wave problem. Especially the linearized wave problem is important because it forms the basis of many other wave theories.

The numerical solution of the nonlinear wave problem which is described in this thesis, is based on the use of an integral equation involving only the boundary of the fluid domain, instead of Laplace's equation for the total fluid domain. With this formulation it is possible to use grid points on the boundary only.

This chapter is divided as follows. First we formulate the equations for the nonlinear wave problem in Sections 2.2 and 2.3. Then some wave theories are discussed in Section 2.4 and solutions of these wave theories are described. The formulation in terms of an integral equation is described in Section 2.5.

2.2 Potential flow

Models for propagating nonlinear free-surface gravity waves are usually based on the potential-flow model. In these models the fluid is assumed to be incompressible and inviscid and the flow irrotational. In this section we briefly

show how the equations describing a potential flow are related to fundamental conservation laws.

In a Cartesian coordinate system fixed in space with coordinates (x, y, z), the local fluid velocity is denoted by $\mathbf{v} = (u, v, w)^T$. The continuity equation for an incompressible fluid expressing the conservation of mass is given by

$$\nabla \cdot \mathbf{v} = 0, \tag{2.1}$$

in which $\nabla = (\frac{\partial}{\partial x}, \frac{\partial}{\partial y}, \frac{\partial}{\partial z})^T$. The conservation of momentum is expressed by the Navier-Stokes equations. For a homogeneous inviscid fluid they simplify to the Euler equations which read

$$\frac{D\mathbf{v}}{Dt} = \frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v} \cdot \nabla)\mathbf{v} = -\frac{1}{\rho}\nabla p - g\mathbf{e}_z. \tag{2.2}$$

ρ is the fluid mass density, p the pressure, g the gravitational acceleration and $\frac{D}{Dt}$ denotes the material derivative. $\mathbf{e}_z$ denotes the unit vector in the positive vertical direction.

We now assume that the fluid flow is irrotational at some time $t = t_0$. i.e.

$$\nabla \times \mathbf{v} = \mathbf{0}. \tag{2.3}$$

From the Euler equations it can be deduced that, if no vorticity is transferred into the fluid domain through the boundaries, the flow will remain irrotational. Because $\nabla \times \mathbf{v} = \mathbf{0}$, the velocity $\mathbf{v}$ can be written as the gradient of a scalar potential ϕ:

$$\mathbf{v} = \nabla\phi. \tag{2.4}$$

The continuity equation (2.1) reduces to Laplace's equation for the velocity potential:

$$\Delta\phi := \nabla^2\phi = 0, \tag{2.5}$$

and the Euler equations can be rewritten and integrated giving

$$\frac{\partial \phi}{\partial t} + \frac{1}{2}\nabla\phi \cdot \nabla\phi = -\frac{p - p_0}{\rho} - gz + B(t). \tag{2.6}$$

p_0 is an arbitrary constant and $B(t)$ is a function of time t which are usually omitted by redefining ϕ, without affecting the velocity field. Equation (2.6) is called Bernoulli's equation.

We see that the potential flow is governed by the potential ϕ satisfying Laplace's equation. If ϕ is known throughout the fluid, the physical quantities $\mathbf{v}$ and p can be obtained from definition (2.4) and Bernoulli's equation (2.6).

To solve these equations for a specific problem, we need initial conditions as well as boundary conditions for the elliptic equation (2.5). The boundary conditions depend on the type of boundary under consideration. In the next section we describe some boundary types and formulate appropriate boundary conditions.

2.3 Boundary conditions

In nonlinear wave problems several boundaries can be distinguished, the free surface being the most obvious one. Depending on the type of problem also the influence of bottom and structures can be considered. Artificial lateral boundaries come in through the truncation of the fluid domain to the domain of interest for a specific problem. Figure 2.1 illustrates all these types of boundaries in a schematic illustration of a ship near a quay.

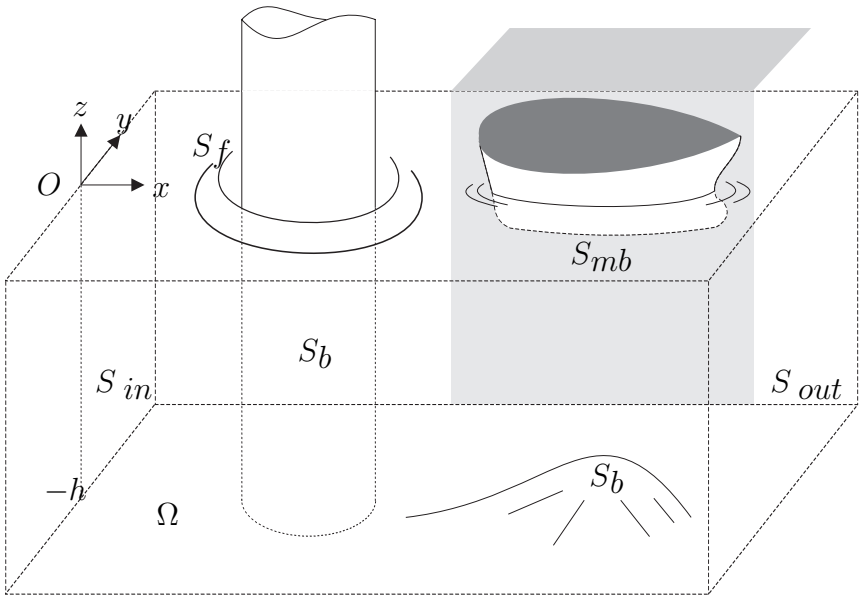

Figure 2.1 Schematic illustration of the fluid domain and some boundaries

2.3.1 Free surface

Because the free surface is a moving boundary, we need more than one condition to complete the potential flow model. The first condition is called the kinematic condition and states that fluid particles at the free surface remain at the free surface:

$$\frac{D\mathbf{x}_f}{Dt} = \mathbf{v} = \nabla\phi, \ \mathbf{x}_f \in S_f. \tag{2.7}$$

The second (dynamic) condition is derived from Bernoulli's equation. By assuming that the atmospheric pressure p at the free surface is zero, the following so called dynamic boundary condition is obtained:

$$\frac{D\phi}{Dt} = \frac{1}{2}\left(\nabla\phi\right)^2 - gz, \ \text{on } S_f. \tag{2.8}$$

2.3.2 Bottom and structures

The condition that is normally used on the bottom and on structures is that of impermeability. On solid fixed boundaries such as the bottom or a quay, the normal component of the velocity is zero:

$$\frac{\partial \phi}{\partial n} = \mathbf{v} \cdot \mathbf{n} = 0, \text{ on } S_b. \tag{2.9}$$

$\mathbf{n}$ is the unit vector normal to the boundary and is chosen to point inside the fluid domain here. On moving boundaries such as a wavemaker, the normal velocity of the fluid coincides with the velocity of the body $\mathbf{v}_{mb}$ in that direction:

$$\frac{\partial \phi}{\partial n} = \mathbf{v}_{mb} \cdot \mathbf{n}, \text{ on } S_{mb}. \tag{2.10}$$

For floating bodies, the same condition can be formulated, but now the body motion is a part of the problem itself. The motion of the body is governed by the forces exerted on the body. The equations of motion of the body have to be solved simultaneously with the potential flow equations. We refer to van Daalen [18] for a further description and a numerical approach to solve this problem. See also Tanizawa [68] and Berkvens [9].

2.3.3 Artificial lateral boundaries

For wave problems in which the fluid domain extends to infinity, an additional condition is required to make the problem well-posed. Moreover we need to truncate the fluid domain Ω in the present numerical model at some distance from the area of interest. Also, when comparing with experiments or in modelling practical problems, we want to consider only a finite part of the fluid domain. Hence, artificial lateral boundaries are needed and artificial boundary conditions have to be imposed there.

We can distinguish two types of boundary conditions for these artificial lateral boundaries. At inflow boundaries, conditions should be formulated corresponding to a desired wave field. An example is the velocity field due to the motion of a wavemaker but more general velocity profiles can be prescribed here. In Section 2.4 we discuss solutions of wave problems over horizontal bottoms which can be used to formulate the inflow boundary condition. In these cases we can formulate expressions for the potential itself:

$$\phi = F(\mathbf{x}(t), t), \ \mathbf{x}(t) \in S_{in}. \tag{2.11}$$

Note that the position of the inflow boundary itself in this formulation is allowed to vary, as in the case of real wavemakers.

At outflow boundaries, conditions should be formulated such that the excluded part of the fluid domain is simulated. The motion due to outgoing waves

(entering this excluded part) should be simulated here. Because the properties of the outgoing waves are in general unknown, finding an acceptable absorbing boundary condition can be very difficult. An example is Higdon's condition ([35], [36]),

$$\frac{\partial \phi}{\partial t} = \frac{c}{\cos(\alpha)} \frac{\partial \phi}{\partial n}, \ \mathbf{x} \in S_{out} \tag{2.12}$$

which reduces to the well-known Sommerfeld's radiation condition if $\alpha = 0$ is chosen. The values of c and α should be good estimates for the phase velocity and the angle between the normal direction of the boundary and the direction of propagation, respectively. In cases in which reflections occur, inflow boundaries should also be considered as outflow boundaries. A combined boundary condition should be formulated then.

Another possibility to simulate an infinite outer field near the outflow boundaries, is the use of artificial damping. In such methods an artificial dissipative term is added to the equations in a damping zone near the outflow boundaries of the truncated domain. An example of such a dissipative term was given by Betts and Mohamad [10] who only added it to the dynamic boundary condition:

$$\frac{D\phi}{Dt} = \frac{1}{2} (\nabla \phi)^2 - gz - \nu \phi, \text{ on } S_f. \tag{2.13}$$

The parameter ν is specified by increasing it from zero at the beginning of the damping zone to a given positive value at the end of the zone. Reflection properties depend on ν and on the length of the damping zone.

In the past much attention has been paid to the suitability of absorbing boundary conditions for free-surface wave problems and for the use in numerical methods. In Romate [60] and Broeze and Romate [12] a number of such conditions are studied. See also Givoli [29] for an extensive overview of non-reflecting boundary conditions. We do not pay much attention to the suitability of absorbing boundary conditions here although the proper use can be very profitable in practical simulations.

We return to the conditions for inflow boundaries later on. In Chapter 9 we will study the suitability of several types of wavemaker motion for the generation of a desired wave field.

2.4 Wave theories

At present no analytical solutions are known for the nonlinear wave problem. To obtain useful formulations, models that approximate the nonlinear model have been developed. The solutions of these approximative models give insight into the propagation of waves and can provide answers to certain problems. Good sources of information on wave theories are for instance Sarpkaya and Isaacson [62], Mei [52], Whitham [70], Lamb [44] and Dingemans [23].

The linear approach in which wave fields are split up into separate wave components is a good example of a useful approximative model. Spectral formulations, in which this splitting is used, are the basis for many studies on wave propagation.

In the numerical model described here, approximations to solutions of the fully nonlinear equations are of importance. In the simulation of propagating waves, the characteristics of the wave field come in through the inflow boundary, so a specification is needed there. In the following this is illustrated by considering Stokes' first and second-order theory to obtain approximate solutions of the nonlinear model which can be used as inflow boundary conditions in the numerical model. Furthermore it is shown that spurious waves may enter the domain if approximate solutions are used.

We only consider solutions of wave theories for a horizontal bottom. In general the inflow boundary of a model (either theoretical, numerical or experimental) marks the beginning of an area in which we are interested in the disturbance of a wave field as a result of a bottom topography, interaction with structures or interaction between waves. The wave conditions at the inflow boundary therefore have to correspond to the local depth and since we are interested in the changes of the wave field, we usually want to formulate conditions that correspond to waves which in a way propagate undisturbed over a horizontal bottom. In this section we derive some of these conditions.

It is to be noted that the numerical model does not necessarily requires the use of inflow conditions which are periodic or stationary in one or the other way. Because of the time-dependent simulation of the wave field, any time series can be used. For example in simulations in which the fluid is in rest initially, the inflow conditions can be increased gradually until the desired conditions have been reached. Inflow conditions composed of random signals can also be used.

2.4.1 Linear theory

As mentioned before, the linear approach is the basis of many studies on wave propagation. In this approach the free-surface conditions are linearized and solutions of the linearized problem are split into separate components which propagate independently. It is alternatively referred to as small amplitude wave theory, sinusoidal wave theory or as Airy theory. In this section we formulate the linear problem and derive its solution following the very readable account in Sarpkaya and Isaacson [62]. First we formulate the governing equations and the solution to these equations. Next the formulation of inflow boundary conditions based on the solution is considered.

Linearizing the free-surface conditions (2.7) and (2.8) around $z = 0$ we find

$$\frac{\partial \phi}{\partial z} - \frac{\partial \eta}{\partial t} = 0, \text{ at } z = 0 \tag{2.14}$$

and

$$\frac{\partial \phi}{\partial t} + g\eta = 0, \text{ at } z = 0, \tag{2.15}$$

which can also be combined as

$$\frac{\partial^2 \phi}{\partial t^2} + g\frac{\partial \phi}{\partial z} = 0, \text{ at } z = 0 \tag{2.16}$$

and

$$\eta = -\frac{1}{g}\left(\frac{\partial \phi}{\partial t}\right)_{z=0}. \tag{2.17}$$

We want to find the solution of the linearized problem for a propagating periodic wave in water of constant depth and we consider the two-dimensional situation with waves propagating in x-direction with phase velocity c, see Figure 2.2.

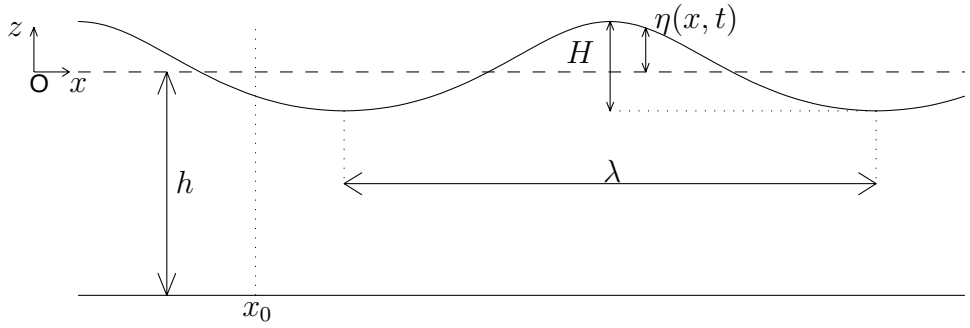

Figure 2.2 Definition of some characteristic quantities for a periodic wave.

A solution of the Laplace equation (2.5) can be found by writing ϕ as

$$\phi(x, z, t) = Z(z)\Phi(x - ct) \tag{2.18}$$

and using separation of variables to derive ordinary differential equations for both Z and Φ. By choosing the time level $t = 0$ as a time level at which a wave top is at the line $x = 0$ and by using (2.9) we find the periodic solution

$$\phi(x, z, t) = A\frac{g}{\omega}\frac{\cosh(k(z + h))}{\cosh(kh)}\sin(k(x - ct)) \tag{2.19}$$

with wave number $k = \frac{2\pi}{\lambda}$ and frequency $\omega = ck$. From condition (2.17) we find

$$\eta(x, t) = A\cos(k(x - ct)). \tag{2.20}$$

which defines A as the wave amplitude. Boundary condition (2.16) expresses ω (or $c = \omega/k$) in k and results in the dispersion relation for linearized free-surface waves

$$\omega^2 = gk \tanh(kh). \tag{2.21}$$

Using this relation, equation (2.19) can also be written as

$$\phi(x, z, t) = A\frac{\omega}{k}\frac{\cosh(k(z+h))}{\sinh(kh)}\sin(k(x - ct)). \tag{2.22}$$

From this solution of the linearized problem many interesting properties of water waves and of the motion of water itself can already be derived. The reader is referred to Sarpkaya and Isaacson [62] for an overview of this matter. In this section we concentrate on the mathematical formulation of solutions of the wave problem. In Chapters 9, 10 and 11 the physical meaning of these expressions will become more apparent.

For the formulation of inflow boundary conditions using equation (2.22) we now consider the semi-finite domain which is bounded in negative x-direction by an inflow boundary $x = x_0$. On this boundary we formulate as a boundary condition the evaluation of solution (2.22) at $x = x_0$:

$$\phi(x_0, z, t) = A\frac{\omega}{k}\frac{\cosh(k(z+h))}{\sinh(kh)}\sin(k(x_0 - ct)) =: F(z, t; \omega, A). \tag{2.23}$$

In the definition of F the value of k is implicitly determined by the dispersion relation (2.21).

The solution of this linear boundary value problem is obviously given by (2.22) and is as such not interesting. We will, however, use the above to consider solutions of the boundary value problem in cases in which the boundary condition on $x = x_0$ is not equal to $F(z, t; \omega, A)$.

As a first example we take as an inflow boundary condition

$$\phi(z, t) = F(z, t; \omega_1, A_1) + F(z, t; \omega_2, A_2). \tag{2.24}$$

Because of the linearity of the boundary value problem, the solution is given as the sum of the right-hand side of equation (2.22) with $(\omega, A) = (\omega_1, A_1)$ and with $(\omega, A) = (\omega_2, A_2)$. Both components propagate independently and their phase velocities are determined by the dispersion relation (2.21).

As a second example we now consider a boundary condition corresponding to the linearized motion of a translating wavemaker with position $X(t) = x_0 + X_0 \sin \omega t$. For this motion the most appropriate condition is in terms of the velocity $U(t) = \phi_x(x, z, t)|_{x=x_0}$:

$$U(t) = \frac{dX}{dt}(t) = \omega X_0 \cos \omega t. \tag{2.25}$$

Contrary to the function $F(z,t;\omega,A)$, this boundary condition is independent of z. The solution of this boundary value problem is found to be (taking $x_0 = 0$)

$$\begin{aligned}\phi(x,z,t) &= X_0 c_0 \frac{\omega}{k}\frac{\cosh(k(z+h))}{\sinh(kh)}\sin(k(x-ct)) \\ &\quad - X_0\frac{g}{\omega}\sum_{j=0}^{\infty} c_j \frac{\cos(k_j(z+h))}{\cos(k_j h)} e^{-k_j x}\cos(\omega t) \qquad (2.26)\end{aligned}$$

and

$$\eta(x,t) = X_0 c_0 \cos(k(x-ct)) - X_0\sum_{j=0}^{\infty} c_j \frac{\cos(k_j(z+h))}{\cos(k_j h)} e^{-k_j x}\sin(\omega t). \qquad (2.27)$$

with k_j the positive solutions of the dispersion relation $-\omega^2 = gk_j\tan(k_j h)$ and c_j certain coefficients. See Schäffer [63] for a discussion of this boundary value problem and a description of an efficient way to determine the coefficients c_j.

The first term in (2.27) describes a propagating wave and the second term describes the so-called evanescent modes for which the fluid motion away from the wavemaker decays exponentially. See Schäffer [63] for a discussion of this boundary value problem.

In the next section we will see that in nonlinear models spurious waves which do not vanish for large x can occur when solutions of the linear approach are prescribed on the inflow boundary.

2.4.2 Second-order theory

As an example of a nonlinear theory we discuss Stokes' second-order theory here, see e.g. Sarpkaya and Isaacson [62]. In Stokes' higher-order theory ϕ and η are written as perturbation series of the following form:

$$\phi = \epsilon\phi_1 + \epsilon^2\phi_2 + \epsilon^3\phi_3 + \ldots \qquad (2.28)$$

and similarly for η. The perturbation parameter ϵ expresses the nonlinearity, for example in terms of the wave slope. The perturbation series are substituted in Laplace's equation and the boundary conditions, after which terms of the same order in ϵ are collected. Laplace's equation and the bottom boundary condition result in:

$$\frac{\partial^2\phi_n}{\partial x^2} + \frac{\partial^2\phi_n}{\partial z^2} = 0, \text{ for } n = 1,2,\ldots \qquad (2.29)$$

and

$$\frac{\partial\phi_n}{\partial z} = 0, \text{ for } n = 1,2,\ldots, \text{ at } z = -h. \qquad (2.30)$$

In first and second order in ϵ we find for the free-surface boundary conditions:

$$\frac{\partial^2\phi_1}{\partial t^2} + g\frac{\partial\phi_1}{\partial z} = 0, \text{ at } z = 0 \tag{2.31}$$

$$\eta_1 = -\frac{1}{g}\left(\frac{\partial\phi_1}{\partial t}\right)_{z=0} \tag{2.32}$$

and

$$\begin{aligned}\frac{\partial^2\phi_2}{\partial t^2} + g\frac{\partial\phi_2}{\partial z} &= -\eta_1\frac{\partial}{\partial z}\left[\frac{\partial^2\phi_1}{\partial t^2} + g\frac{\partial\phi_1}{\partial z}\right] \\ &\quad - \frac{\partial}{\partial t}\left[\left(\frac{\partial\phi_1}{\partial x}\right)^2 + \left(\frac{\partial\phi_1}{\partial z}\right)^2\right], \text{ at } z = 0\end{aligned} \tag{2.33}$$

$$\eta_2 = -\frac{1}{g}\left(\frac{\partial\phi_2}{\partial t} + \eta_1\frac{\partial\phi_1}{\partial z\partial t} + \frac{1}{2}\left(\frac{\partial\phi_1}{\partial x}\right)^2 + \frac{1}{2}\left(\frac{\partial\phi_1}{\partial z}\right)^2\right)_{z=0} \tag{2.34}$$

respectively. The equations of the first-order problem are equal to those of the linearized problem. The equations of the second-order problem are similar to those of the first-order problem except for the extra terms in the right-hand sides of (2.33) and (2.34). These extra terms contain quadratic expressions in ϕ_1 and η_1. A solution up to second order can be found by considering a first-order solution and determining the second-order part by using equations (2.33) and (2.34).

Given the first-order solution (2.20) and (2.22), the right-hand side of equation (2.33) will only contain terms with phase function $2k(x - ct)$ and the second-order part of the solution can be found similarly to the determination of the first-order solution itself. The expressions for ϕ and η then are (see Sarpkaya and Isaacson [62]):

$$\phi(x,z,t) = \epsilon\phi_1(x,z,t) + \epsilon^2\phi_2(x,z,t) \tag{2.35}$$

$$\begin{aligned}&= A\frac{\omega}{k}\frac{\cosh(k(z+h))}{\sinh(kh)}\sin(k(x-ct)) \\ &\quad + \frac{3}{8}A^2\omega\frac{\cosh(2k(z+h))}{\sinh^4(kh)}\sin(2k(x-ct))\end{aligned} \tag{2.36}$$

and

$$\eta(x,t) = \epsilon\eta_1(x,t) + \epsilon^2\eta_2(x,t) \tag{2.37}$$

$$\begin{aligned}&= A\cos(k(x-ct)) \\ &\quad + \frac{1}{4}A^2k\frac{\cosh(kh)}{\sinh^3(kh)}[2+\cosh(2kh)]\cos(2k(x-ct)).\end{aligned} \tag{2.38}$$

The vertical structure and the phase function of the second-order solution do not correspond to those of a first-order solution. The phase function contains the double wave number $2k$ but propagates with the same phase velocity c, its angular frequency being equal to $2kc = 2\omega$. Therefore this second-order solution is called the bound second harmonic. Its frequency and wave number do not satisfy the dispersion relation (2.21). It differs from the so-called free second harmonic which is to be considered as a separate first-order solution and which propagates with its own phase velocity $c_f = \frac{2\omega}{k_f}$ with k_f given by the dispersion relation (2.21): $(2\omega)^2 = gk_f \tanh(k_f h)$. Free second harmonic waves occur as spurious waves when the second-order part of (2.36) is not taken into account in the inflow boundary condition. We will clarify this in the following.

We again consider the semi-finite problem with the inflow boundary at $x = x_0$ for both the first-order and the second-order problem. If we take the first term in equation (2.36) evaluated in $x = x_0$ (defined here as $F_1(z, t; \omega, A)$) for the first-order problem and the second term (defined as $F_2(z, t; \omega, A)$) for the second-order problem, we again find (2.36) as the solution of the complete problem for $\phi = \epsilon\phi_1 + \epsilon^2\phi_2$ and $\eta = \epsilon\eta_1 + \epsilon^2\eta_2$.

If the term $F_2(z, t; \omega, A)$ is omitted in the inflow boundary condition for the second-order problem, then the semi-finite problem formulated as above has a different solution. If we consider the corresponding situation in the physical system then the 2ω motion induced on the inflow boundary by the first-order signal has to be compensated by an opposite 2ω motion. In second-order theory this motion is expressed by a solution containing a free second harmonic and a series of evanescent modes, see Schäffer [63]. The propagating free second harmonic disturbs the first-order signal and the wave field is not periodic anymore.

In physical tests in which periodic waves are required, one wants to avoid the occurrence of free higher harmonics. This implies that higher-order harmonic motions of certain amplitude and phase have to be imposed on the first-order motion of the wavemaker. In the case of irregular waves, higher-order contributions are also needed to obtain a well-defined wave field. In Chapter 9 we will see that the numerical model is capable of simulating the water motion induced by a wavemaker. In this way it is possible to determine whether higher-order motions of wavemakers are able to suppress spurious waves.

Second-order theory can also provide insight into the development of free waves with smaller frequencies, so-called subharmonics. They can arise as spurious waves because of difference frequencies when using two or more different first-order signals. The importance of the identification of signals with low frequencies is indicated in Chapter 10.

We again consider the infinite problem of Figure 2.2 with as first-order com-

ponent

$$\begin{aligned}\phi_1(x,z,t) &= A_n\frac{\omega_n}{k_n}\frac{\cosh(k_n(z+h))}{\sinh(k_nh)}\sin(k_n(x-c_nt)) \\ &\quad + A_m\frac{\omega_m}{k_m}\frac{\cosh(k_m(z+h))}{\sinh(k_mh)}\sin(k_m(x-c_mt)).\end{aligned} \tag{2.39}$$

This bichromatic wave signal describes a wave group, an example of which is illustrated in Figure 2.3 by the full line.

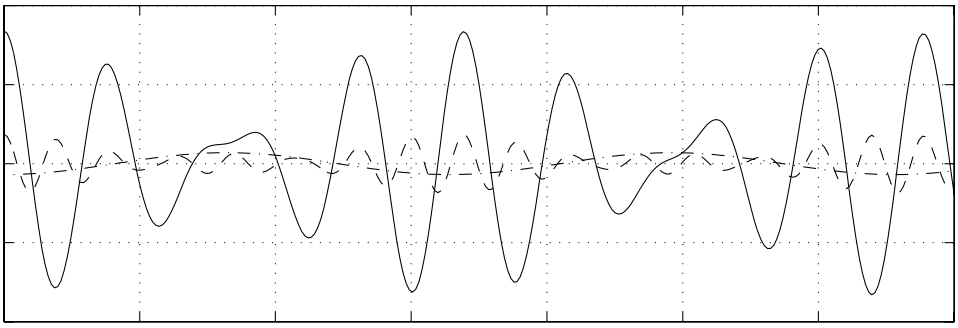

Figure 2.3 First-order wave group signal (−) with subharmonics (−·) and superharmonics (−−).

In equation (2.33) of the second-order problem, the right-hand side now contains terms with phase functions

$$\begin{cases} \theta^+_{nn} := 2k_nx - 2\omega_nt, \\ \theta^+_{mm} := 2k_mx - 2\omega_mt, \\ \theta^+_{nm} := (k_n+k_m)x - (\omega_n+\omega_m)t, \\ \theta^-_{nm} := (k_n-k_m)x - (\omega_n-\omega_m)t. \end{cases} \tag{2.40}$$

The solution of the second-order problem will therefore contain three superharmonic waves with frequencies $2\omega_n$, $2\omega_m$ and $\omega_n+\omega_m$ and one subharmonic wave with frequency $|\omega_n-\omega_m|$. Note that these four waves all have different phase velocities. Because they travel along with and are determined by the frequencies *and* by the wave lengths of the first-order components, these waves are also called 'bound waves'.

The second-order surface elevation η_2 can be expressed in terms of the first-order amplitudes A_n and A_m by means of a transfer function $G^{\pm}_{nm}$ as

$$\begin{aligned}\eta_2 &= G^+_{nn}A_nA_n\cos(2\theta_n) + G^+_{mm}A_mA_m\cos(2\theta_m) \\ &\quad + G^+_{nm}A_nA_m\cos(\theta_n+\theta_m) + G^-_{nm}A_nA_m\cos(\theta_n-\theta_m).\end{aligned} \tag{2.41}$$

Expressions for $G^{\pm}_{nm}$ can be found in Schäffer [63]. The function G^-_{nm} corresponding with the subharmonic is always negative. From this fact it can be

derived that wave groups formed by two first-order signals have a bound subharmonic which induces a suppression of the mean water level below the high waves and a rise below the small waves, see Figure 2.3. This is related to the variation of the short-wave averaged momentum flux along a wave group, also known as radiation stress, see Longuet-Higgins and Stewart ([47], [48]). Expressions for $G^{\pm}_{nm}$ have also been found for three-dimensional problems in which the two first-order solutions propagate in different directions (indicated by phase functions $\mathbf{k}_n \cdot \mathbf{x} - \omega_n t$ and $\mathbf{k}_m \cdot \mathbf{x} - \omega_m t$), see Sharma [65] or Dean and Sharma [21].

If we consider a situation of an inflow boundary $x = x_0$ on which only the bichromatic first-order signal (2.39) is given without the induced second-order signal, then again free super- and subharmonic waves will occur with frequencies $2\omega_n$, $2\omega_m$, $\omega_n + \omega_m$ and $|\omega_n - \omega_m|$. In the frame work of second-order theory, their wave lengths and phase velocities are determined by the dispersion relation (2.21).

2.4.3 Fully nonlinear theory for periodic waves

As mentioned in the introduction, the fully nonlinear theory has no known analytical solutions. Although many theories may yield very good approximations, care has to be taken with respect to the range of validity of these approximations. For example second-order Stokes' theory will show a bump in the trough of a periodic wave when the second-order terms become too large.

A numerical approach which is valid for all water depths and up to very high waves, is a Fourier approximation method after Rienecker and Fenton [57] based on the stream function theory by Dean [20]. The velocity potential and surface elevation of a periodic wave travelling over a horizontal bottom are expressed in Fourier series. The solution satisfies the nonlinear free-surface conditions exactly at a number of equidistantly distributed x-coordinates x_m over one wave length. In this way the error in the solution is determined by the truncation error of the Fourier series only. The wave elevation is written as

$$\eta(x,t) = \frac{a_0}{2} + \sum_{j=1}^{N} a_j \cos(jk(x - x_0 - ct)), \tag{2.42}$$

and the potential as

$$\begin{aligned} \phi(x,z,t) &= (c + B_0)(x - x_0) \\ &\quad + \sum_{j=1}^{N} B_j \frac{\cosh(jk(z+h))}{\cosh(jkh)} \sin(jk(x - x_0 - ct)) + b(t). \end{aligned} \tag{2.43}$$

for some coefficients a_j and B_j found by substitution of equations (2.42) and (2.43) into the nonlinear free-surface conditions. The latter term $b(t)$ comes in by choosing $B(t) = 0$ in the dynamical boundary condition (2.6) and is equal to

$-(g(R-h)-\frac{1}{2}c^2)t$ with R some constant. The wave crest is located at $x = x_0$ at $t = 0$ and the wave travels with speed c in positive x-direction.

A case which is frequently taken as a model problem in this thesis, is a wave on water of intermediate depth. The wave length λ for this case is 60 m on a depth h of 10 m. In contrast to waves in linear theory, the wave frequency ω depends on the waveheight H. From the solution of the Fourier approximation method, c and hence ω and the wave period T can be obtained. For wave heights $H = 2.5$ m and $H = 5.0$ m we find $c = 8.7058$ m/s and 9.1618 m/s and $T = 6.8920$ s and 6.5489 s respectively. These waves are shown in Figure 2.4.

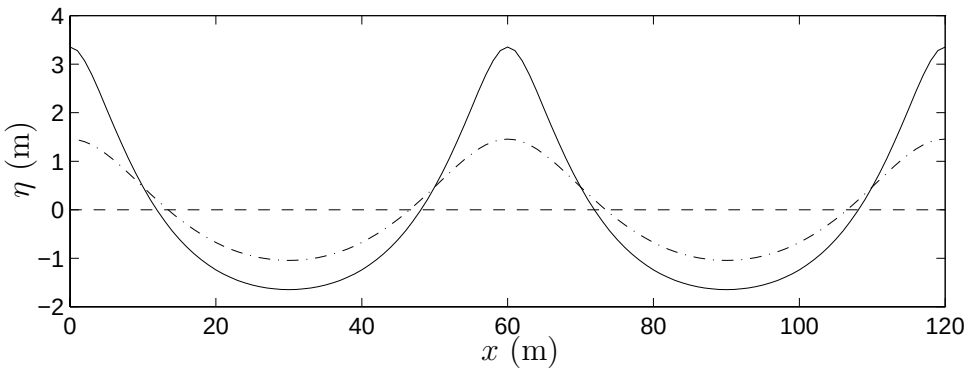

Figure 2.4 Nonlinear wave profiles. $H = 5.0$ m $(-)$ and $H = 2.5$ m $(-\cdot)$.

A typical feature of nonlinear waves (which can already be observed from second-order theory) is that wave tops are sharpened and wave troughs are flattened with increasing wave height. Another typical feature is that, contrary to linear waves, the particle paths are not closed in general. In the absence of a background flow, water particles experience a period-averaged drift in the propagation direction of the waves. This is of importance for the numerical method because in a Lagrangian description of the grid-point motion on the free surface, the grid points will drift as well.

2.5 Boundary integral equation methods

In this section we describe the approach that has been implemented numerically to solve the equations for nonlinear wave problems including problems with uneven bottoms and general inflow and outflow conditions. This approach is based on the possibility to transform Laplace's equation to a boundary integral equation.

In Section 2.5.1 we show how a boundary integral equation can be derived from Laplace's equation. Finally, in Section 2.5.2 we give a survey of numerical methods that can be used to solve this boundary integral equation.

2.5.1 Boundary integral equation for Laplace's equation

The derivation of the boundary integral equation is based on Green's theorem which states that for two arbitrary scalar fields u and w which are C^2-continuous throughout a domain Ω

$$\iiint_{\Omega} \left[u\Delta w - w\Delta u\right] dV = \iint_{\partial\Omega} \left[w\frac{\partial u}{\partial n_\xi} - u\frac{\partial w}{\partial n_\xi}\right] dS_\xi \tag{2.44}$$

With $\partial\Omega$ we indicate the boundary of Ω. $\boldsymbol{\xi}$ is the integration variable and $\mathbf{n}_\xi$ is the normal unit vector pointing inward at $\boldsymbol{\xi}$ on $\partial\Omega$.

For a fixed point $\mathbf{x} \in \Omega \cup \partial\Omega$ we now replace u by the unknown potential $\phi(\mathbf{y})$, satisfying $\Delta\phi = 0$ and w by the fundamental solution

$$G(\mathbf{x};\mathbf{y}) = \frac{-1}{4\pi|\mathbf{x}-\mathbf{y}|}. \tag{2.45}$$

Fundamental solutions are solutions of Laplace's equation outside the point $\mathbf{x}$. In the point $\mathbf{x}$ the fundamental solution is singular and the Laplace operator can be expressed in terms of the Dirac delta function δ:

$$\Delta_y G(\mathbf{x};\mathbf{y}) = \delta(\mathbf{x}-\mathbf{y}),\ \mathbf{x},\mathbf{y} \in \mathbb{R}^3. \tag{2.46}$$

In two-dimensions we use the fundamental solution $G(\mathbf{x};\mathbf{y}) = \ln(|\mathbf{x}-\mathbf{y}|)/2\pi$.

For field points $\mathbf{x} \in \Omega$ (i.e. in the interior of $\Omega \cup \partial\Omega$) we find

$$\phi(\mathbf{x}) = \iint_{\partial\Omega} \left[\frac{\partial\phi}{\partial n_\xi}(\boldsymbol{\xi})G(\mathbf{x};\boldsymbol{\xi}) - \phi(\boldsymbol{\xi})\frac{\partial G}{\partial n_\xi}(\mathbf{x};\boldsymbol{\xi})\right] dS_{\boldsymbol{\xi}}. \tag{2.47}$$

The potential is expressed in values on the boundary only.

For field points $\mathbf{x} \in \partial\Omega$ on the boundary, the right-hand side of (2.47) is a singular integral due to the presence of the fundamental solution. It can be evaluated using the concept of finite part of an integral. We then find

$$\frac{\vartheta(\mathbf{x})}{4\pi}\phi(\mathbf{x}) = \fint\!\!\!\fint_{\partial\Omega} \left[\frac{\partial\phi}{\partial n_\xi}(\boldsymbol{\xi})G(\mathbf{x};\boldsymbol{\xi}) - \phi(\boldsymbol{\xi})\frac{\partial G}{\partial n_\xi}(\mathbf{x};\boldsymbol{\xi})\right] dS_{\boldsymbol{\xi}}. \tag{2.48}$$

The symbol $\fint\!\!\!\fint$ denotes the finite part of the integral in the sense of Hadamard. $\vartheta(\mathbf{x})$ is the interior angle of $\partial\Omega$ at $\mathbf{x}$; if $\mathbf{x}$ is on a smooth part of $\partial\Omega$, $\vartheta(\mathbf{x}) = 2\pi$. See Jaswon and Symm [38] for an extensive treatment of integral equations and potential problems.

The boundary integral equation (2.48) is now expressed completely in terms of values of ϕ and $\frac{\partial\phi}{\partial n}$ on the boundary. It is this equation which is solved numerically in the panel method.

Boundary integral equations (2.47) and (2.48) can be interpreted physically by considering the boundary to be covered with a source and a dipole distribution. Every point $\boldsymbol{\xi}$ induces a fluid flow by the presence of a source and a dipole. For example, the potential $\phi_{s,\xi}$ due to a source of strength q is given by $\phi_{s,\xi}(\mathbf{x}) = -q/(4\pi|\mathbf{x} - \boldsymbol{\xi}|)$. A similar expression can be given for the potential $\phi_{d,\xi}$ due to a dipole in $\boldsymbol{\xi}$. In a point $\mathbf{x} \in \Omega$ the flow due to all sources and dipoles on the boundary $\partial\Omega$ can now be expressed by the integral (2.47). In a point $\mathbf{x} \in \Omega$ on the boundary itself a factor $\vartheta(\mathbf{x})/4\pi$ has to be taken into account because of the presence of the singular integral, see for example [38].

It is also possible to derive integral equations which contain only a source or a dipole distribution or to use a formulation for $\nabla\phi$ instead of ϕ. See Romate [58] for a summary of possible integral equations and a discussion of the appropriateness of these integral equations for the numerical implementation for the simulation of nonlinear waves. Another approach leading to an integral equation, also briefly described by Romate, is to consider the free surface as a vortex sheet. Finally it is remarked that the use of Cauchy's integral equation is another way to arrive at an integral equation for the velocity potential ϕ. This approach has been succesfully used in a number of methods but since it is based on complex variables its use is restricted to two-dimensional domains. For an extensive treatment of the derivation of integral equations (and their numerical treatment), the reader is referred to Hackbusch [34].

2.5.2 Numerical approaches to the boundary integral equation

The panel method which is used to solve integral equation (2.48) is described in detail in Section 3.2. The description is an enumeration of the parts of the algorithm to find a numerical solution. Here we place the panel method in an overview of more general methods for integral equations.

In *boundary element methods* the approximate solution of the integral equation is formulated using the method of weighted residuals. For an operator equation

$$Lu = f, \ \mathbf{x} \in \partial\Omega \tag{2.49}$$

the solution u is approximated by

$$\tilde{u}(\mathbf{x}) = \sum_{i=1}^{N} c_i u_i(\mathbf{x}) \tag{2.50}$$

where the $u_i(\mathbf{x})$ are known local basis functions, e.g. lower order polynomials, and where the constants c_i are to be determined. In the weighted residual method, N weighting functions $w_j(\mathbf{x})$ $(j = 1, \ldots, N)$ are introduced and the

following N weighted integrals of the residual

$$R(\tilde{u}) = L\tilde{u} - f \tag{2.51}$$

are set equal to zero:

$$(w_j, R(\tilde{u})) = \int_{\partial\Omega} w_j R(\tilde{u}) dS = 0,\ j = 1, \ldots, N. \tag{2.52}$$

From these N algebraic equations the c_j's can be determined. Different choices for w_j lead to different methods. Most commonly functions with local support are chosen. This method is also applied in finite element methods (FEM) for the direct approximation of partial differential equations and therefore its counterpart in the approximation of integral equations is called boundary element methods (BEM).

Two important types of boundary element methods are point collocation and Galerkin. In point collocation Dirac delta functions are used for the weight functions w_j whereas in Galerkin the basis functions $u_i(\mathbf{x})$ are used.

In *panel methods* an approach is followed resembling the discretization used in finite difference methods. Based on the positions of a number of collocation points $\mathbf{x}_j$ a discretization of the boundary $\partial\Omega$ by panels is chosen and the collocation points are used as expansion points for Taylor series expansions for the functions (source and dipole distributions) and the panel geometry. The derivatives in the expansions are approximated by finite differences using the function values in neighbouring collocation points. A further distinction can be made based on the order of the finite difference schemes employed.

The lower order panel method, with constant sources, dipoles and flat panel approximations is the same as the lower order point collocation method in the weighted residual class mentioned above. See further Romate [58], Section 8.1, and Hackbusch [34].

Chapter 3

Numerical method

3.1 Introduction

The mathematical model for nonlinear water waves can be divided into two parts. On the one hand we have boundary conditions describing the evolution of the boundaries and the potential on the boundaries in time. On the other hand we have an elliptic problem (Laplace's equation) in a domain whose boundaries change in time. These two parts are handled alternately by the numerical model using a method of lines approach.

In this approach the boundary conditions (2.7) and (2.8) are integrated in time with a higher order Runge-Kutta method. To determine the time derivatives in these equations, the spatial derivatives in the right-hand sides have to be determined. This can be done with the solution of the Laplace problem. The Laplace problem is solved using a panel method for the boundary integral equation (2.48).

In more detail it can be described as follows. At $t = 0$ we have an initial configuration described by a domain Ω_0 and initial conditions for ϕ or $\frac{\partial \phi}{\partial n}$ on the boundary $\partial\Omega_0$ of the domain. For the free surface we specify ϕ and for the bottom we specify $\frac{\partial \phi}{\partial n}$ $(= 0)$. All other boundaries are also initialized.

Then Laplace's equation is solved in this domain in which all parts of the boundary can be specified as either a Dirichlet boundary (if ϕ is specified) or a Neumann boundary (if $\frac{\partial \phi}{\partial n}$ is specified). The solution of this equation (obtained with the panel method) provides $\frac{\partial \phi}{\partial n}$ on the free surface and therefore also the spatial derivatives which are needed in the free-surface boundary conditions (2.8) and (2.7). By integrating these boundary conditions in time, a new position is found for the free-surface boundary and values for the potential ϕ on this boundary on the next time level are determined. These are then again used to formulate a new Laplace problem and in this way the time marching scheme proceeds.

The time integration also applies to other boundaries but its use depends

on the type of boundary. The position of the grid on the boundaries adjacent to the free-surface boundary, is integrated such that it follows the motion of the grid on the free surface boundary. The values of ϕ or $\frac{\partial \phi}{\partial n}$ on these boundaries do not have to be integrated in time when they are specified explicitly as function of place and time on for example the bottom or an inflow boundary with specified velocities. On an outflow boundary where Sommerfeld's condition is used, equation (2.12) provides values for the potential ϕ on the next time level in the same way as for the free surface.

Summarizing, the numerical model for nonlinear water waves consists of a repetition of the following sequence of operations.
Panel method:

1. calculation of the variables describing the geometry based on positions of the collocation points on the boundary of the fluid domain

2. calculation of the source and dipole coefficients which represent the source and dipole distribution in the boundary integral equations

3. substitution of ϕ for Dirichlet boundaries and $\frac{\partial \phi}{\partial n}$ for Neumann boundaries at time level t_n

4. solution of the discretized boundary integral equations at time level t_n, giving $\frac{\partial \phi}{\partial n}$ for Dirichlet boundaries and ϕ for Neumann boundaries

5. calculation of the tangential velocities at time level t_n from ϕ using finite differences

6. calculation of the boundary velocity $\nabla \phi$ at time level t_n from the tangential and normal velocities

Time marching scheme:

7. integration of the time dependent boundary conditions, giving the collocation point positions, ϕ for Dirichlet boundaries and $\frac{\partial \phi}{\partial n}$ for Neumann boundaries at time level t_{n+1}

Because the choice of the grid is determined mainly by the use of the panel method, we describe the panel method first in Section 3.2. In Section 3.3 the time marching scheme based on the Runge-Kutta method is described. Some important features of the numerical model are discussed in Section 3.4. In Section 3.5 we summarize the computational results of Broeze [13]. They serve as a starting point for the investigations in this work.

3.2 Panel method

3.2.1 Geometric description

The panel method is used to solve the boundary integral equation (2.48). In the discretization of this equation, the boundary $\partial\Omega$ is divided into subsurfaces $\partial\Omega_p$ in such a way that they can be covered by a structured grid with M_p by N_p panels. Such a grid is called a network. Figure 3.1 shows some examples. If

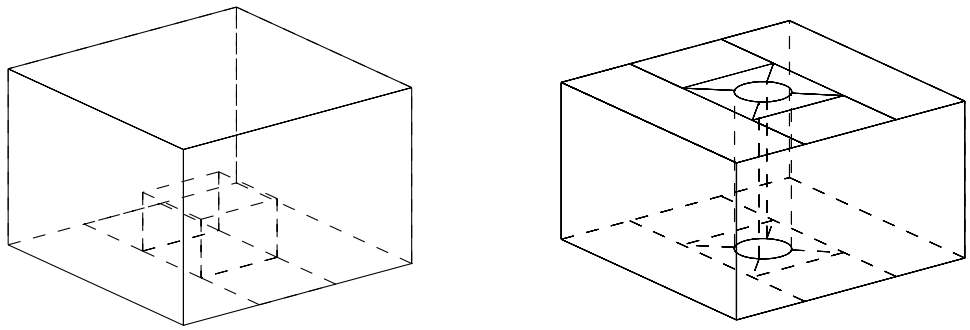

Figure 3.1 Domain boundaries divided into networks.

a network edge has more than one adjacent network, this edge is divided into edge parts accordingly. Figure 3.1 shows examples of configurations with some networks connecting to three other networks on one edge.

Every panel is represented by one collocation point which is located near the centre of the panel. The collocation points are also used as the computational points in the time marching scheme and therefore constitute the basic description of the numerical model.

The geometry of the panels is described by using the positions of the collocation points. For the description a parametrization is needed which corresponds to coordinates in a computational domain and a mapping from the computational to the physical domain. This mapping is also used to determine derivatives of the flow variables and the geometry.

The parametrization of a network is based on a rectangular grid in the computational domain. The collocation points correspond to the points with coordinates $(u, v) = (i - \frac{1}{2}, j - \frac{1}{2})$, where $(i, j) \in \{1, \ldots, M_p\} \times \{1, \ldots, N_p\}$, see Figure 3.2.

First- and second-order derivatives of the flow variables and the geometry in a collocation point are determined using nine-point molecules. The collocation point is located in the centre of the molecule if its corresponding panel is not located at an edge of the network. Central difference schemes can be used then. If the panel is located at an edge of the network, the collocation point is on the edge of the molecule as well, as shown in Figure 3.2. One-sided difference

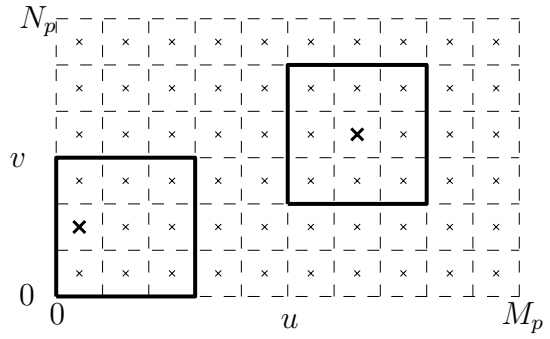

Figure 3.2 Computational molecules on a network.

schemes are used for these molecules. No use is made of information beyond the edges of the network.

The finite difference schemes are only applied to the collocation points and not to the panel vertices. Although the panel vertices can be associated with the points $(u, v) = (i, j) \in \{0, \ldots, M_p\} \times \{0, \ldots, N_p\}$ in the computational domain, they are not given by these coordinates in the parametrization. Therefore the edges of a network in the physical domain do not necessarilly correspond to the edges of the rectangle in the computational domain. The edges are determined by a special algorithm to which we come back in Section 4.2. In the following this is indicated by using the verb 'to associate' instead of the verb 'to correspond' for the panel vertices.

The relation between discrete derivatives in the computational domain (which are easy to determine) and the discrete derivatives in the physical domain is determined by Taylor expansions of functions in the computational domain using tangential coordinates in the physical domain. We clarify this with a two-dimensional example.

In the two-dimensional case a network corresponds to a straight line in the computational domain and three-point molecules are used for the discrete derivatives. For a collocation point with computational coordinate u_0, the value of a function f in the neighbouring points with coordinates $u_1 = u_0 + 1$ and $u_{-1} = u_0 - 1$ can be written as

$$f(u_1) = f(u_0) + (s_1 - s_0)f_s + \frac{1}{2}(s_1 - s_0)^2 f_{ss} + \ldots \tag{3.1}$$

$$f(u_{-1}) = f(u_0) - (s_0 - s_{-1})f_s + \frac{1}{2}(s_0 - s_{-1})^2 f_{ss} - \ldots \tag{3.2}$$

The variable s corresponds with a tangential coordinate along the surface in the physical domain and the subscripts 1, 0 and -1 correspond to the collocation points with coordinates u_1, u_0 and u_{-1} resp. in the computational domain.

By using the standard difference schemes for the derivatives f_u and f_{uu} in the computational domain (in this case a central scheme) we find, neglecting Δs^3, Δs^4, etc.

$$\begin{pmatrix} \tilde{f}_u \\ \tilde{f}_{uu} \end{pmatrix} = T \begin{pmatrix} \tilde{f}_s \\ \tilde{f}_{ss} \end{pmatrix} \tag{3.3}$$

with

$$T := \begin{pmatrix} \frac{1}{2}((s_1 - s_0) + (s_0 - s_{-1})) & \frac{1}{4}((s_1 - s_0)^2 - (s_0 - s_{-1})^2) \\ ((s_1 - s_0) - (s_0 - s_{-1})) & \frac{1}{2}((s_1 - s_0)^2 + (s_0 - s_{-1})^2) \end{pmatrix}. \tag{3.4}$$

For each collocation point, the discrete derivatives $\tilde{f}_u$ and $\tilde{f}_{uu}$ are expressed by difference weights in the computational domain. The inverse of the matrix T transforms these difference weights to the difference weights in the physical domain.

3.2.2 Intersections

The panel vertices at the edges of the networks describe the intersections between the networks. Panels on both sides of the intersection have these panel vertices in common which implies that the resolution of adjacent networks along their intersection has to be the same. Lines through the collocation points in the directions normal to the intersection therefore approximately intersect and this property is used to determine the so-called intersection points, see Figure 3.3.

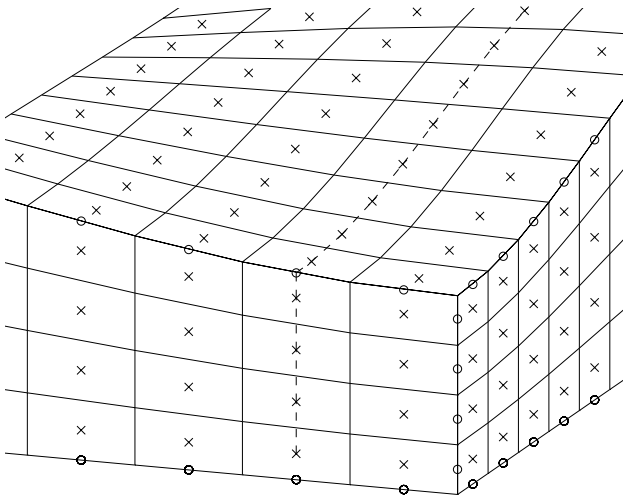

Figure 3.3 Location of intersection points (o) and collocation points (x).

The intersection points are obtained by an iterative algorithm which is discussed in Section 4.2. They can be associated with the points $(u, v) = (0, j - \frac{1}{2})$,

$(M_p, j-\frac{1}{2})$, $(i-\frac{1}{2}, 0)$ and $(i-\frac{1}{2}, N_p)$ where $i \in \{1, \ldots, M_p\}$ and $j \in \{1, \ldots, N_p\}$ in the computational domain. The panel vertices on the edge of a network are determined from the intersection points. Each edge consists of a number of edge parts equal to the number of networks adjacent to this edge. On each edge part there are two panel vertices at the end points of the edge part which are indicated as outer panel vertices. The other panel vertices on the edge part are referred to as inner panel vertices.

The outer panel vertices are determined with an iterative algorithm using the positions of the intersection points. This iterative algorithm is similar to that for the intersection points themselves. Note that outer panel vertices can be on the edge of more than two different networks. The inner panel vertices are determined by the evaluation of a cubic spline through the intersection points based on distances $\frac{1}{2}$ in the computational domain between intersection points and panel vertices.

Once again it is remarked that for each network the description of the physical domain as function of the coordinates in the computational domain is based on the coordinates of the collocation points and is independent from the description of other networks. So, intersections (in the physical domain) of networks in general do not correspond to sets of prescribed points in the computational domain of one network (such as the points on the edges of the rectangle of size M_p by N_p in Figure 3.2), since they are also dependent on the description of the other networks. In 3-D it may even be the case that the lines which are used for the determination of the intersection points do not intersect in the physical domain. Because the intersection points need to be uniquely defined, they are determined as close as possible to both lines.

3.2.3 Geometric variables

In the discretization of the boundary integral, geometric variables are needed such as the lengths of the panel sides and the surfaces of the panels. They are related to the positions of the panel vertices. For the stability of the method it is important that the panels mutually connect, which implies that their joint panel vertices are uniquely defined. See Broeze [13]. In the description of the determination of the intersections, we already saw that the panel vertices on the edges of the networks are determined such that on the intersections panels indeed connect. The panel vertices which are not on the edge of a network have to be vertices of four panels because of the rectangular structure of the networks.

These panel vertices are determined by the four collocation points of the panels involved. For each of these collocation points the quadratic description of the geometry is evaluated in the point (u, v) in the computational domain that is associated with the panel vertex. Because local descriptions are used, the four evaluations will be different in general for curved surfaces and the position

of the panel vertex is determined by averaging them. The geometric variables are determined from the positions of the panel vertices.

3.2.4 Discretization of the boundary integral equation

The next step in the discretization of the boundary integral equation (2.48) is the computation of the integral over every separate panel S_j for all collocation points $\mathbf{x}_i$. We write the discretized integrals as

$$\frac{1}{2}\phi(\mathbf{x}_i) = \sum_{j=1}^{N} \iint_{S_j} \left(\frac{\partial \phi}{\partial n}(\boldsymbol{\xi}) G(\mathbf{x}_i;\boldsymbol{\xi}) - \phi(\boldsymbol{\xi}) \frac{\partial G}{\partial n}(\mathbf{x}_i;\boldsymbol{\xi}) \right) dS_\xi, \ i = 1, \ldots, N. \quad (3.5)$$

Higher order approximations of $\phi(\boldsymbol{\xi})$ and $\frac{\partial \phi}{\partial n}(\boldsymbol{\xi})$ are obtained by using Taylor expansions in the collocation points $\mathbf{x}_j$ of the panels S_j. All derivatives of ϕ and $\frac{\partial \phi}{\partial n}$ in $\mathbf{x}_j$ can be expressed in terms of the values of ϕ and $\frac{\partial \phi}{\partial n}$ in the collocation points of the nine-point molecule of $\mathbf{x}_j$. Therefore the discretized equations (3.5) can be written as

$$\sum_{j=1}^{N} \left[C_s(i,j) \frac{\partial \phi}{\partial n}(\mathbf{x}_j) + C_d(i,j) \phi(\mathbf{x}_j) \right] = 0, \ i = 1, \ldots, N. \quad (3.6)$$

The coefficients $C_s(i,j)$ and $C_d(i,j)$ are called source and dipole coefficients respectively. Together they are called influence coefficients. Expressions for the influence coefficients can be found in Broeze [13].

The N discretized equations (3.6) contain $2N$ variables. For the solution of the Laplace problem, every network is identified as either a Dirichlet or a Neumann boundary. In every collocation point either ϕ or $\frac{\partial \phi}{\partial n}$ is known. The number of unknowns therefore equals N and equation (3.6) can be written as a system of N linear equations:

$$\mathcal{A}\mathbf{x} = \mathbf{b}. \quad (3.7)$$

3.2.5 Solution of the discretized problem

The above system of equations can be solved either by a direct method or by an iterative method. In general, the matrix $\mathcal{A}$ is full, and neither symmetric nor positive definite. Thus little can be said about the convergence of iterative methods. In our method we have used both Gaussian elimination and a Conjugate Gradients Squared method, as proposed by Sonneveld [66]. This method belongs to the class of conjugate gradient type solvers. We refer to Romate [61] for an investigation of several solvers in this class applied to boundary integral equation methods.

3.2.6 Order of the panel method

Romate [58] studied the accuracy of the panel method as a function of the order of approximation of the source and dipole distributions and of the geometry. In the present numerical method a linear approximation is used for the source distribution and a quadratic approximation for both the dipole distribution and the geometry, giving a local truncation error of order $O(\Delta x^3)$. Near the edges of the networks, the geometry is approximated one order lower. The global truncation error is determined by the occurrence of non-smooth boundaries. In the problems we consider, the boundary is composed of a number of networks so that the global truncation order equals $O(\Delta x^2)$.

3.3 Time marching scheme

3.3.1 Runge-Kutta method

The evolution of the free surface and of the potential on the free surface are computed with a Runge-Kutta method. The evolution equations (the boundary conditions (2.7) and (2.8)) can be written as

$$\mathbf{u}_t = \frac{d\mathbf{u}}{dt} = \mathbf{f}(\mathbf{u}, \nabla\phi)\,, \tag{3.8}$$

with $\mathbf{u} = (x_f, y_f, z_f, \phi)^T$. Also the Sommerfeld radiation condition can be written in this form.

Classical higher order Runge-Kutta methods require the use of a number of intermediate time levels at which the right-hand side of the differential equation (in this case (3.8)) has to be evaluated. These methods are based on the first-order approximation

$$\mathbf{u}(t + \Delta t) = \mathbf{u}(t) + \mathbf{u}_t(t)\Delta t + O(\Delta t^2). \tag{3.9}$$

The time stepping method that was used by Romate is the classical fourth-order Runge-Kutta method. Important features of the method are that it has a high-order accuracy and that the stability region is reasonably large. A drawback of the method in combination with the panel method is that at every time step the Laplace problem has to be solved on the intermediate time levels as well. This implies that the panel method is applied four times per time step. At each time level the geometry and therefore also the influence coefficients have to be computed which is relatively expensive.

Because of this drawback Broeze studied a fourth-order Runge-Kutta method that also uses second-order time derivatives of $\mathbf{u}$ and therefore needs only one intermediate time level to obtain the same order of accuracy. This so-called

two-stage two-derivative method is based on the approximation

$$\mathbf{u}(t+\Delta t) = \mathbf{u}(t) + \mathbf{u}_t(t)\Delta t + \frac{1}{2}\mathbf{u}_{tt}(t)\Delta t^2 + O(\Delta t^3). \tag{3.10}$$

For the evaluation of the term $\mathbf{u}_{tt}$ on the right-hand side, second-order evolution equations are required. For the free surface these can be obtained from the first-order dynamic and kinematic boundary conditions. We come back to these second-order time derivatives in Sections 4.4.2 and 4.4.3. The two-stage two-derivative method is described in more detail in Appendix A.

3.3.2 Grid motion

Two main possibilities can be distinguished when integrating the free-surface grid in time.

Firstly, using an Eulerian description in which the free-surface elevation $\eta(x, y, t)$ is considered a function of time and fixed horizontal coordinates only and is integrated according to

$$\frac{\partial \eta}{\partial t} = \frac{\partial \phi}{\partial z} - \frac{\partial \phi}{\partial x}\frac{\partial \eta}{\partial x} - \frac{\partial \phi}{\partial y}\frac{\partial \eta}{\partial y}. \tag{3.11}$$

Secondly, according to a Lagrangian description in which the position of a material fluid particle on the free surface is followed.

$$\frac{D\mathbf{x}_f}{Dt} = \nabla\phi. \tag{3.12}$$

It is not possible to describe breaking waves with the Eulerian approach because the surface elevation η is not single-valued at the position of the breaker. When adopting the Lagrangian approach on the other hand, a deformation of the grid will occur when drift velocities vary in the horizontal directions due to diffraction, refraction or reflection. For example when waves approach a surface piercing object, the water particles in front of the object stagnate whereas the water particles beside the object drift along the object.

In the present method the Lagrangian description has been adopted because, besides permitting the simulation of breaking waves, it was also noted that instabilities develop near the inflow boundaries when the Eulerian description is used, see Broeze [13]. The disadvantages of a Lagrangian description can be removed up to a certain level by using a grid correction algorithm.

In this grid correction algorithm, not the flow velocity $\mathbf{v} = \frac{D\mathbf{x}_f}{Dt} = \nabla\phi$ and acceleration $\mathbf{a} = \frac{D^2\mathbf{x}_f}{Dt^2}$ are used for the time integration of the collocation points $\mathbf{x}_i$, but a grid velocity $\mathbf{v}_g$ and a grid acceleration $\mathbf{a}_g$. The determination of the new position of a collocation point is based on

$$\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t) + \mathbf{v}_g\Delta t + \frac{1}{2}\mathbf{a}_g\Delta t^2 + O(\Delta t^3). \tag{3.13}$$

The determination of the potential on the new time level uses the material derivative

$$\frac{\tilde{D}\phi}{\tilde{D}t} := \frac{\partial\phi}{\partial t} + \mathbf{v}_g \cdot \nabla\phi \tag{3.14}$$

following the grid point, instead of the material derivative $\frac{D\phi}{Dt}$ following the material point. The grid velocity $\mathbf{v}_g$ is determined as follows.

For every collocation point $\mathbf{x}_i$, the grid velocity $\mathbf{v}_g$ consists of the material velocity $\mathbf{v}$, an alignment velocity $\mathbf{v}_a$ and a correction velocity $\mathbf{v}_c$:

$$\mathbf{v}_g = \mathbf{v} + \mathbf{v}_a + \mathbf{v}_c. \tag{3.15}$$

The alignment velocity $\mathbf{v}_a$ is only used for the collocation points on the lateral boundaries and on the bottom. It is applied in such a way that the networks on these boundaries follow the motion of the networks on the free surface. In Section 4.3 we consider the determination and application of $\mathbf{v}_a$ in more detail.

The correction velocity $\mathbf{v}_c$ is a velocity tangential to the boundary which is used to avoid a too large distortion of the grid on the individual networks. This algorithm is studied in Section 4.4.

The grid correction algorithm is capable of controlling the grid sufficiently. A nice example was shown in the simulation of a plunging breaking wave in the work of Broeze [13]. The velocity field in this situation varies enormously. Still it was possible to continue this simulation up to an advanced stage of breaking without a large deformation of the grid. However, the use of correction velocities that are not related to the velocities of the fluid particles has some disadvantages which is discussed in Section 4.4.

3.4 Features and properties of the method

The numerical method has some features and properties which become important in complex computations. In this section we consider some of them and point out some difficulties which may arise from them. Such considerations are important in the process of developing the method.

3.4.1 Network structure

An important characteristic of the method is the structuring of the grid into rectangular subgrids called networks. In problems with a structure or body intersecting the free surface, the free surface has to be built up of multiple networks in order to keep a sufficient resolution away from the structure or body. It may also be required that multiple networks have to be used on the

free surface to be able to cover it with the rectangular structured networks. An example of such a case is shown in Figure 3.1.

3.4.2 Grid motion

As described in Section 3.3.2, the Lagrangian description is used for the movement of the collocation points on the free surface together with a tangential velocity correction to maintain a good grid quality. Because the resolution on adjacent networks has to be the same near the intersections, algorithms are used to move the networks on lateral boundaries along with the free-surface networks. The networks can be moved in a normal direction and in a tangential direction. Depending on the type of boundary condition on the lateral boundaries either the use of regridding or the use of correction velocities as described in Section 3.3.2 can be considered.

For an outflow boundary where Sommerfeld's radiation condition is imposed, the formulation in terms of the material derivative has to be used and correction velocities which move this boundary along with the free surface can be employed. If however a regridding algorithm would be used instead of correction velocities, grid velocities $\mathbf{v}_g$ would have to be determined from the grid position on the old and the new time level. So far only the first approach has been used.

For lateral boundaries where the boundary conditions are *not* integrated in time but are prescribed explicitly at every time level, regridding algorithms can be used. Examples are boundaries with the impermeability condition or inflow boundaries with an explicit formulation for ϕ or $\frac{\partial \phi}{\partial n}$ as e.g. in equation (2.43).

3.4.3 Midpoint description

The collocation points, which are used for both the panel method and the time integration method, are located near the panel centers. Towards the panel vertices, values of ϕ, $\frac{\partial \phi}{\partial n}$, velocities and geometry are obtained from the information in the collocation points to which we refer as the midpoint description. These values will generally be less accurate near the panel vertices and there are some specific parts of the numerical method which are strongly affected by the midpoint description and that become important in applying the method.

1. The intersection between two adjacent networks is defined by the panel vertices at the edges of the networks. The geometry description there is obtained from collocation points located away from the intersection and is more inaccurate when the networks connect under an angle. When networks connect smoothly, information from both sides can be combined to obtain the same accuracy as inside the networks.

2. The nine-point molecule for discretizing derivatives in a collocation point $\mathbf{x}_i$ lies completely in the network to which $\mathbf{x}_i$ belongs to. Therefore the de-

scription of the derivatives becomes less accurate near the edges of the networks. Also here accuracy can be improved for smoothly connected networks if information is taken from both networks and the computational molecules are allowed to cross the intersection.

3. Another consequence of the molecules being restricted to one network only, is that in the outer collocation points one-sided discretisations are used for the derivatives. As shown in Broeze [13], Section 6.3.2, this may cause instabilities in the time marching scheme near intersections of free-surface networks with lateral networks. They can be avoided by extrapolating information from the lateral network towards the intersection. If networks connect smoothly, similar to the previous remark, two-sided discretisations can be computed from computational molecules crossing the network intersection.

3.5 Computational results obtained with the method

In the work of Broeze special attention was paid to the improvement of the accuracy and stability of the method developed by Romate. In this section we discuss the most important computational results which were achieved with Broeze's method.

3.5.1 Verification on periodic waves

To test the numerical model on a consistent and stable approximation of the fully non-linear free-surface problem the results from the Fourier approach by Rienecker and Fenton [57] (see Section 2.4.3) can be used to compare with. To exclude starting effects, the periodic solution is imposed as initial condition in the numerical method. This is done by putting the collocation points at time level $t = t_0$ on the prescribed surface $z = \eta(x, y, t_0)$ and by initializing the potential in those points with the values of the solution of the Fourier approach. Also on the lateral boundaries these values are imposed, either as Dirichlet condition or as Neumann condition.

In the computation the solution is prescribed on the lateral boundaries and the bottom at every time level and the solution on the free surface is obtained by the panel method and the time marching scheme. At every time step the numerical solution obtained in this way can be compared with the Fourier approach.

Broeze verified his results using configurations with different time and spatial resolutions, different domain sizes, different algorithms for the grid motion and with different directions of propagation of the solution with regard to the

orientation of the grid. Also the use of Higdon's radiation condition was tested. We summarize the result of the verification as follows:

- The numerical method gives stable and accurate results for both mildly and highly nonlinear waves. The resolution was taken the same in both cases, so for mildly nonlinear waves the geometry description was more accurate than for highly nonlinear waves. Therefore, errors were smaller in the former case.

- Higdon's radiation condition gives smaller errors than prescribing the solution from the Fourier approach on the outflow boundary. This is due to the fact that numerical errors are reflected more in the latter case. An important source of numerical errors can be the numerical dispersion of the finite difference scheme. This topic has been investigated by Romate in [58].

- The lateral boundaries in the direction of the propagating wave have a negligible effect on the accuracy. Changing the waves' direction of propagation relative to the grid lines hardly influences the accuracy.

- Computations with highly nonlinear waves generate numerical cross waves at the outflow boundaries. This instability can be removed by adding an extra (consistent) term to the dynamic free-surface condition for the collocation points near the outflow edge of the free-surface network.

3.5.2 Verification on experiments

Results obtained with the numerical model have also been compared with experiments carried out in a wave flume. The two-dimensional model was applied to a configuration including a bar on an even bottom. The computations showed good agreement with the measured data. The three-dimensional model was applied to a bottom topography with an elliptic shoal on a slope. It was concluded that the computational results agreed fairly well with the measured data. Errors were observed due to poorly absorbed waves near the outflow boundaries and due to the coarse mesh which had to be used. Both computations will be shown as examples for the application of domain decomposition in Chapter 7.

Chapter 4

Further study and verification

4.1 Introduction

In the work of Broeze [13], the numerical method for nonlinear water waves has been verified for a number of test problems. Most of the computations were performed over a limited number of wave periods, at most six, and it was shown that the test problems could be solved in a stable way. In the studies described here, attention is shifted towards applicability and therefore we have investigated the robustness of the method in problems with large simulation times. It appears that in such simulations, some parts of the method are important for stability of the computations and these parts are studied in this chapter. On the one hand it concerns the grid correction algorithm that has to prevent large grid deformations. On the other hand it concerns algorithms which are sensitive to deformed grids and these are studied in Sections 4.2 and 4.3.

An approach for the problem of grid deformation, is the use of a mixed Eulerian-Lagrangian description for the grid motion on the free surface. When the deformation of the grid would be large (for example due to large local spatial variations in velocities), a relatively large Eulerian contribution is desired, thus counteracting the deformation. When flexibility of the grid is required (for example in breaking waves or near moving objects), the Lagrangian contribution should constitute the largest part. The Eulerian description is related to the use of tangential velocities. This description and the determination of second-order time derivatives are described in Section 4.4.

At the end of this chapter in Section 4.5 the solutions for the stability problems are verified with numerical experiments. It is shown that, with the improvements, it is possible to do accurate and stable simulations over many wave periods.

4.2 Intersection algorithm

For the description of the intersection algorithm developed by Broeze, we consider the two-dimensional case of two networks connecting under an angle as illustrated in Figure 4.1. The networks in 2-D in this illustration correspond to

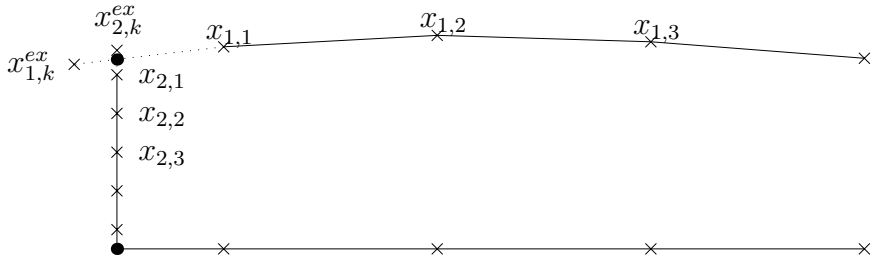

Figure 4.1 Collocation points used to determine the intersection of two networks in the 2-D-method.

the grid lines in 3-D through the collocation points in the direction normal to the edges although in general these grid lines will not be in one plane.

The iterative procedure for the determination of the intersection point consists of the following steps:

0. Choose distances $\xi_{1,k} = \xi_{2,k} = 1/2,\ (k = 0)$ in the computational domain for extrapolation from network 1 and 2.

1. Determine higher order accurate extrapolations $\mathbf{x}^{ex}_{1,k}$ and $\mathbf{x}^{ex}_{2,k}$ beyond the networks, at distances $\xi_{1,k}$ and $\xi_{2,k}$ in the computational domain from the outmost collocation points $\mathbf{x}_{1,1}$ and $\mathbf{x}_{2,1}$ respectively.

2. Approximate the surface near the extrapolated point by a straight line through the outmost collocation point and the extrapolated point for both networks.

3. Determine the intersection point $\mathbf{x}_{I,k}$ of both lines.

4. Determine a better approximation for the extrapolation distances $\xi_{1,k}$ and $\xi_{2,k}$ from the ratio of the distances between $\mathbf{x}_{I,k}$ and the outmost collocation point, and the distance between $\mathbf{x}^{ex}_{1,k}$ resp. $\mathbf{x}^{ex}_{2,k}$ and the outmost collocation point.

   Repeat steps 1 to 4 ($k := k+1$) until an accurate approximation has been found.

We analyze the convergence of this algorithm for a configuration in which the second network is vertical. This simplifies the analysis considerably because the

extrapolation from the second network is of no further importance anymore and the iterative algorithm can be described solely in terms of the iterands $\xi_{1,k}$. For convenience of notation the second network is described by the vertical $x = 0$.

Given an extrapolation distance $\xi_{1,k}$ and using an extrapolation formula expressed by the functions f_1, f_2 and f_3, we find:

$$\mathbf{x}_{1,k}^{ex} = f_1(\xi_{1,k})\mathbf{x}_{1,1} + f_2(\xi_{1,k})\mathbf{x}_{1,2} + f_3(\xi_{1,k})\mathbf{x}_{1,3}. \tag{4.1}$$

The z-coordinate of the intersection point $\mathbf{x}_{I,k}$ is found by intersecting the second network $x = 0$ with the line

$$l_k : \ \mathbf{x}(\alpha) = \mathbf{x}_{1,1} + \frac{\alpha}{\xi_{1,k}}(\mathbf{x}_{1,k}^{ex} - \mathbf{x}_{1,1}), \ \alpha \in \mathbb{R}. \tag{4.2}$$

The parameter α is normalized such that it corresponds to the distance in the computational domain. Intersection of l_k with $x = 0$ then yields:

$$\xi_{1,k+1} = \xi_{1,k}\frac{x_{1,1}}{x_{1,1} - x_{1,k}^{ex}}. \tag{4.3}$$

Using Lagrangian interpolation polynomials for f_1, f_2 and f_3 we find

$$\begin{aligned} \xi_{1,k+1} &= \frac{x_{1,1}}{\xi_{1,k}(-\frac{1}{2}x_{1,1} + x_{1,2} - \frac{1}{2}x_{1,3}) + (-\frac{3}{2}x_{1,1} + 2x_{1,2} - \frac{1}{2}x_{1,3})} \\ &=: \frac{a}{b\xi_{1,k} + c} =: \varphi(\xi_{1,k}) \end{aligned} \tag{4.4}$$

Note that this formula is independent from the z-coordinates and depends only on the distribution of the collocation points along the x-direction. It can be shown that for some irregular distributions of the grid with $|x_3 - x_2| > |x_2 - x_1|$ the description in the computational domain describes a curve which does not intersect the line $x = 0$ and the iterative algorithm does not converge in these cases. For grids with $|x_3 - x_2| < |x_2 - x_1|$ it can be shown that the iterative algorithm does converge.

This iterative algorithm has been replaced by the direct evaluation of a Lagrangian interpolation polynomial defined by the positions of the free-surface collocation points in the physical domain. It was shown that simulations using this evaluation can be continued over a longer simulation time than simulations using the iterative algorithm. However, the original iterative algorithm by Broeze was developed to deal with cases when both boundaries are curved and may still prove to be valuable, for example when simulating (curved) objects moving in waves. It is clear that adjustments have to be made then.

4.3 Grid alignment algorithm

In the time-marching scheme the (vertical) lateral boundaries follow the free-surface boundary. This is not done by constructing the networks of these boundaries anew every time step, but by using an alignment velocity that corresponds to the velocity of the intersection of the free surface with the lateral boundary. This enables the use of boundary conditions on the lateral boundaries involving an integration in time, as for example Sommerfeld's radiation condition. For the integration the material derivative $\frac{\tilde{D}}{\tilde{D}t}$ as defined in Section 3.3.2 is used, see Figure 4.2.

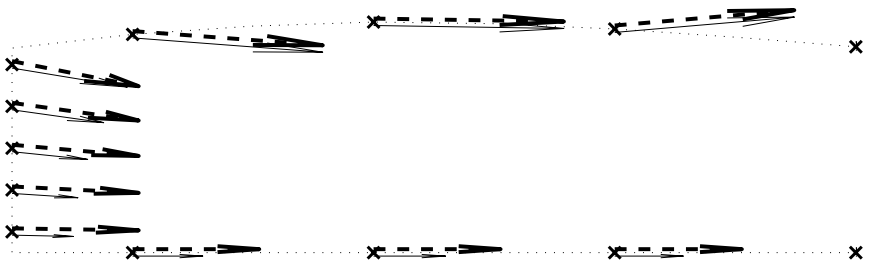

Figure 4.2 The grid alignment algorithm applied to the inflow boundary. The solid arrows indicate $\mathbf{v}$ and the dashed arrows indicate $\mathbf{v}_g$. Note that the horizontal component of $\mathbf{v}_g$ is equal for all collocation points on the inflow boundary.

Broeze developed an algorithm in which the horizontal component of the alignment velocity is uniform over the lateral boundary. In this way the lateral boundaries are kept vertical and drift along with the edges of the free surface. A tangential alignment velocity is used to move the bottom networks along with the movement of the lateral boundaries. Because the boundary condition for the bottom networks involves no time integration, an explicit regridding is also an option for these networks.

The alignment velocity of the lateral boundaries is found by extrapolating the material velocities of the collocation points of these boundaries to the intersection of the free-surface network with the lateral boundary. The extrapolation is based on distances in the computational domain with the intersection point at distance $\frac{1}{2}$ from the most nearby collocation point. Because the intersection point does not correspond to the 'computational' point at distance $\frac{1}{2}$ in general, a certain error is introduced. See also Section 4.2 where extrapolation is applied to the free-surface network. It was observed in time-domain simulations that the extrapolated value of the material velocity for the inflow boundary, averaged over a wave period, was too large. In these simulations the inflow boundary overtook the first collocation point on the free surface despite the use of the grid correction algorithm and these computations broke down.

The use of a Lagrangian interpolation polynomial based on distances in the physical domain gives better values for the alignment velocity. It was shown that simulations can be prolonged using this approach and that the grid distribution near the inflow and outflow boundaries is stabilized.

4.4 Grid correction algorithm

The grid correction algorithm as developed by Broeze has proved to be essential for the stable computation of highly nonlinear waves where large distortions of the grid may occur. It is based on the use of tangential correction velocities which are determined from:

- local variations in resolutions
- local variations in velocities
- local curvature of the boundary
- curvature of the grid lines along the boundary
- connection with the grid lines on adjacent boundaries

Closer examination of this algorithm shows that its use in combination with the two-stage two-derivative Runge-Kutta scheme introduces some additional difficulties in determining second-order quantities. These are particularly important for the free-surface boundary because the integration of the Dirichlet boundary condition and the integration of the position of the collocation point, using the dynamic and the kinematic boundary condition respectively, have to be consistent with each other. On boundaries where the boundary condition is a function of the position of the collocation point only, this consistency is automatically fulfilled because the new boundary condition automatically corresponds to the new position of the collocation point.

We discuss some of these difficulties in the following and indicate their effect on the accuracy of the method. First the relation with the Eulerian description is explained in Section 4.4.1. In Sections 4.4.2 and 4.4.3 the determination of the second-order time derivatives is described.

4.4.1 Use of tangential velocities

The use of the tangential correction velocities introduces a certain error which can be indicated in 2-D as follows. Consider a point $\mathbf{x}_0$ on the free surface at a certain time level $t = t_0$ and let l be the tangent at $\mathbf{x}_0$, see Figure 4.3. Let s be the distance along l to $\mathbf{x}_0$ and $\mathbf{s}$ and $\mathbf{n}$ the tangential and normal unit-vectors at $\mathbf{x}_0$ respectively. Locally the surface can be approximated as

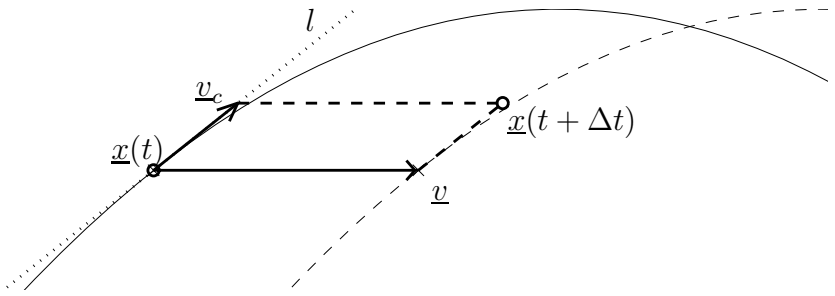

Figure 4.3 Error in time integration for a curved free surface using a grid correction velocity. Indicated are the free-surface at time levels t (solid line) and $t+\Delta t$ (dashed line), the material velocity $\mathbf{v}$ and the tangential correction velocity $\mathbf{v}_c$.

$\mathbf{x} = \mathbf{x}_0 + s \cdot \mathbf{s} + \frac{1}{2}\eta_{ss} \cdot s^2\mathbf{n} + O(s^3)$. Using a tangential correction velocity $\mathbf{v}_c$ the error in approximating the free-surface location at time level $t_0 + \Delta t$ then is of order $O(s^2) = O((|\mathbf{v}_c|\Delta t)^2)$. If $\mathbf{x}_0(t)$ is integrated using formula (3.13) up to and including terms of order Δt, i.e. $\mathbf{x}_0(t+\Delta t) = \mathbf{x}_0(t) + \Delta t(\mathbf{v} + \mathbf{v}_c)$, the error is still of the same order. The order of time stepping methods which only use the first-order time derivative is therefore not affected, although the truncation error can be expected to be larger.

If the point is integrated using formula (3.13) up to and including terms of order $(\Delta t)^2$, i.e. $\mathbf{x}_0(t+\Delta t) = \mathbf{x}_0(t) + \Delta t(\mathbf{v} + \mathbf{v}_c) + \frac{1}{2}(\Delta t)^2(\mathbf{a} + \mathbf{a}_c)$, the order of the error is reduced to $(\Delta t)^2$. Only if $\mathbf{a}_c$ is determined such that it accounts for the curvature of the boundary and for the gradient in velocity along the boundary, the $(\Delta t)^3$-order of the error is maintained. We discuss this matter in Section 4.4.2 and 4.4.3.

The effect of an error in the determination of the free surface can be interpreted as follows. The dynamic boundary condition is applied to a small deviation from the true free surface. Because the dynamic boundary condition assumes the atmospheric pressure to be zero on the surface on which it is applied, it consequently imposes a (small non-zero) pressure field on the true free surface.

Although the truncation error will remain of the same order when used in the classical fourth-order Runge-Kutta scheme, it is mentioned that for large time steps its effect may become important when the tangential correction velocity is relatively large. This is the case when an Eulerian description is used. The relation between the use of tangential velocities and the Eulerian description is described next.

The Eulerian description is usually derived using the function

$$F(x,y,z,t) = z - \eta(x,y,t),\ \mathbf{x} = (x,y,z) \in \mathbb{R}^3. \tag{4.5}$$

This defines the free-surface elevation η implicitly by $F(x,y,z,t) = 0$. By taking the material derivative we find

$$\frac{DF}{Dt} = \frac{\partial F}{\partial t} + \mathbf{v} \cdot \nabla F = -\frac{\partial \eta}{\partial t} + \mathbf{v} \cdot \begin{pmatrix} -\frac{\partial \eta}{\partial x} \\ -\frac{\partial \eta}{\partial y} \\ 1 \end{pmatrix}. \tag{4.6}$$

For material points on the free surface we have $\frac{DF}{Dt} = 0$ and we find

$$\frac{\partial \eta}{\partial t} = w - u\frac{\partial \eta}{\partial x} - v\frac{\partial \eta}{\partial y},\ \mathbf{v} = (u,v,w). \tag{4.7}$$

The relation with the tangential velocity can be understood more easily by considering the material derivative $\frac{\hat{D}\eta}{\hat{D}t}$ using a displacement velocity $(\hat{u},\hat{v})$ in the horizontal plane instead of the flow velocity (u,v). For $(\hat{u},\hat{v})$ the projection of the grid velocity $\mathbf{v}_g$ on the horizontal plane can be taken. We find

$$\frac{\hat{D}\eta}{\hat{D}t} = \frac{\partial \eta}{\partial t} + \begin{pmatrix} \hat{u} \\ \hat{v} \end{pmatrix} \cdot \begin{pmatrix} \eta_x \\ \eta_y \end{pmatrix} = w - (u-\hat{u})\frac{\partial \eta}{\partial x} - (v-\hat{v})\frac{\partial \eta}{\partial y}. \tag{4.8}$$

If $(\hat{u},\hat{v}) = (0,0)$, we obtain the Eulerian description. If $(\hat{u},\hat{v}) = (u,v)$, we obtain the Lagrangian description and $\frac{\hat{D}\eta}{\hat{D}t} = w$. Expression (4.8) can be referred to as a mixed Eulerian-Lagrangian description as it (linearly) interpolates between the Lagrangian and the Eulerian description. [1] Such a description was also used by Kim et al. [43] in order to describe the run-up and run-down along a shoreline. They used a formulation in which the Eulerian description was adapted by rotating the coordinate system. Rotation was increased towards the shoreline such that close to the shoreline the rotated z_*-axis was almost parallel to the shoreline.

In the study on wave diffraction by a surface-piercing cylinder, presented in Chapter 11, we will use the mixed Eulerian-Lagrangian description to control the grid around the cylinder.

4.4.2 Determination of the grid acceleration

In the integration of the collocation points in the two-stage two-derivative Runge-Kutta scheme, a grid velocity $\mathbf{v}_g$ and a grid acceleration $\mathbf{a}_g$ are used

[1] The term mixed Eulerian-Lagrangian description (MEL) is often used in literature when higher-order time integration formulae are used as in equation (3.10). So in these cases its meaning is different from the meaning given to mixed Eulerian-Lagrangian here.

instead of the material velocities $\mathbf{v}$ and $\mathbf{a}$. Using a material derivative $\frac{\tilde{D}}{\tilde{D}t}$ (see definition (3.14)) they are related as

$$\begin{aligned}
\mathbf{a}_g &= \frac{\tilde{D}}{\tilde{D}t}\{\mathbf{v}_g\} = \frac{\partial \mathbf{v}_g}{\partial t} + \mathbf{v}_g \cdot \nabla \mathbf{v}_g \\
&= \frac{\partial \mathbf{v}}{\partial t} + \frac{\partial (\mathbf{v}_g - \mathbf{v})}{\partial t} + \mathbf{v} \cdot \nabla \mathbf{v} + \mathbf{v} \cdot \nabla (\mathbf{v}_g - \mathbf{v}) \\
&\quad + (\mathbf{v}_g - \mathbf{v}) \cdot \nabla \mathbf{v} + (\mathbf{v}_g - \mathbf{v}) \cdot \nabla (\mathbf{v}_g - \mathbf{v}) \\
&= \mathbf{a} + \frac{\partial (\mathbf{v}_a + \mathbf{v}_c)}{\partial t} + \mathbf{v}_g \cdot \nabla (\mathbf{v}_a + \mathbf{v}_c) + (\mathbf{v}_a + \mathbf{v}_c) \cdot \nabla \mathbf{v}, \qquad (4.9)
\end{aligned}$$

assuming a differentiable flow field $\mathbf{v}_g = \mathbf{v} + \mathbf{v}_a + \mathbf{v}_c$. The contribution $\mathbf{v}_a + \mathbf{v}_c$ to the grid velocity is, however, determined by the grid distribution using extrapolations and grid dependent variations and approximate methods have to be used to determine its derivatives. This can be rather complicated, especially for the time derivative $\frac{\partial (\mathbf{v}_a + \mathbf{v}_c)}{\partial t}$. Instead of using equation (4.9) directly, Broeze chose to determine $\mathbf{a}_g$ using the algorithm for $\mathbf{v}_g$. This approach and its suitability are indicated next.

The grid alignment velocity $\mathbf{v}_a$ is determined from extrapolations. It is expressed as a linear combination (determined by the grid distribution) of material velocities $\mathbf{v}$ in three collocation points. For the acceleration $\mathbf{a}_a$ the same extrapolation is used of accelerations $\mathbf{a}$ in the same collocation points. The error in $\mathbf{a}_a$ determined in this way can be expected to be of the same order as the error made in $\mathbf{v}_a$ as an approximation to the velocity in the intersection point.

The grid correction velocity $\mathbf{v}_c$ is determined as a function of grid distribution, geometry etc. which can be indicated as $\mathbf{v}_c = F(\{\mathbf{x}_i | i = 1 \ldots N\}, \partial\Omega, \ldots)$. The acceleration $\mathbf{a}_c$ is determined in exactly the same way (i.e. $\mathbf{a}_c = F(\{\mathbf{x}_i | i = 1 \ldots N\}, \partial\Omega, \ldots)$) instead of being determined from the acceleration of a point moving along the surface and therefore an error in the motion of the free surface can be expected to arise from this. This error is related to the $O((|\mathbf{v}_c|\Delta t)^2)$ term mentioned before.

4.4.3 Determination of the second-order material derivative

In the two-stage two-derivative Runge-Kutta scheme also a second-order time derivative of the potential is used. An expression for this derivative has already been given by Broeze [13]. A new expression is derived here and compared with Broeze's expression. We start the derivation from Bernoulli's equation

$$\frac{D\phi}{Dt} - \frac{1}{2}\mathbf{v} \cdot \nabla \phi + gz + \frac{p}{\rho} = 0, \ \mathbf{x} \in \Omega. \qquad (4.10)$$

Because $\frac{\tilde{D}\phi}{\tilde{D}t} = \frac{D\phi}{Dt} + (\mathbf{v}_g - \mathbf{v}) \cdot \nabla\phi$, we have

$$-\frac{p}{\rho} = \frac{\tilde{D}\phi}{\tilde{D}t} + (\tfrac{1}{2}\mathbf{v} - \mathbf{v}_g)\nabla\phi + gz, \ \mathbf{x} \in \Omega. \tag{4.11}$$

The grid velocity $\mathbf{v}_g$ is chosen such that grid points remain on the free surface. Therefore the material derivative of the pressure term equals zero and we have

$$\frac{\tilde{D}}{\tilde{D}t}\left\{-\frac{p}{\rho}\right\} = \frac{\tilde{D}}{\tilde{D}t}\left\{\frac{\tilde{D}\phi}{\tilde{D}t} + (\tfrac{1}{2}\mathbf{v} - \mathbf{v}_g)\nabla\phi + gz\right\} = 0, \ \mathbf{x} \in \partial\Omega_{FS}. \tag{4.12}$$

The second-order material derivative of ϕ can thus be written as

$$\frac{\tilde{D}^2\phi}{\tilde{D}t^2} = \frac{\tilde{D}}{\tilde{D}t}\left\{(\mathbf{v}_g - \tfrac{1}{2}\mathbf{v})\nabla\phi - gz\right\} = \frac{\tilde{D}}{\tilde{D}t}\left\{-\tfrac{1}{2}\nabla\phi \cdot \nabla\phi - gz + \mathbf{v}_g \cdot \nabla\phi\right\}. \tag{4.13}$$

In Broeze [13] this equation is worked out further by applying the usual product rule for differentiation. Here we analyze under which conditions the use of the product rule is valid for $\frac{\tilde{D}}{\tilde{D}t}$ in this equation. We again assume that $\mathbf{v}_g$ is a differentiable function in time and space.

We consider $\frac{\tilde{D}}{\tilde{D}t}\{\mathbf{v}_g \cdot \nabla\phi\}$ and apply the definition of material derivative first and then apply the product rule which is valid for $\frac{\partial}{\partial t}$ and the spatial derivatives. Working out these derivatives we find

$$\begin{aligned}\frac{\tilde{D}}{\tilde{D}t}\{\mathbf{v}_g \cdot \nabla\phi\} \ &= \ \frac{\tilde{D}}{\tilde{D}t}\{\mathbf{v}_g\} \cdot \nabla\phi + \mathbf{v}_g \cdot \frac{\tilde{D}}{\tilde{D}t}\{\nabla\phi\} \\ &\quad + (\mathbf{v}_g \cdot (\nabla\mathbf{v}_g \cdot \nabla\phi) - (\mathbf{v}_g \cdot \nabla\mathbf{v}_g) \cdot \nabla\phi)\end{aligned} \tag{4.14}$$

in which $\nabla\mathbf{v}_g = (\nabla u_g, \nabla v_g, \nabla w_g)^T$, and the inner product with $\mathbf{v}_g$ or $\nabla\phi$ is taken on all components separately. So the product rule is valid when the term $(\mathbf{v}_g \cdot (\nabla\mathbf{v}_g \cdot \nabla\phi) - (\mathbf{v}_g \cdot \nabla\mathbf{v}_g) \cdot \nabla\phi)$ equals zero. In general

$$\mathbf{a} \cdot ((\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)^T \cdot \mathbf{c}) = (\mathbf{a} \cdot (\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)^T) \cdot \mathbf{c}) \tag{4.15}$$

if $b_{1,y} = b_{2,x}$, $b_{1,z} = b_{3,x}$ and $b_{2,z} = b_{3,y}$, i.e. interchanging the index of $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)^T$ and the index of the coordinate gives the same value. Because $(\mathbf{b}_1, \mathbf{b}_2, \mathbf{b}_3)^T = (\nabla u_g, \nabla v_g, \nabla w_g)^T$ this condition is equivalent with $\frac{\partial u_g}{\partial y} = \frac{\partial v_g}{\partial x}$, $\frac{\partial u_g}{\partial z} = \frac{\partial w_g}{\partial x}$ and $\frac{\partial v_g}{\partial z} = \frac{\partial w_g}{\partial y}$.

This is true for an irrotational velocity field in which $\nabla \times \mathbf{v} = \mathbf{0}$. But for the grid velocity $\mathbf{v}_g$ it is not clear whether it is irrotational. Because it has no relation with the physics of the problem but is determined on the basis of the discretization it can be expected that $\nabla \times \mathbf{v}_g \neq \mathbf{0}$. This implies that the product rule as applied by Broeze in equation (4.13) is valid for the term $\nabla\phi \cdot \nabla\phi$ but not for the term $\mathbf{v}_g \cdot \nabla\phi$.

4.5 Verification

The accuracy and stability of the numerical method in simulations over many wave periods can be tested by simulating the nonlinear periodic wave given by the method of Rienecker and Fenton [57] as already explained in Section 3.5.1. Broeze [13] found that if the solution according to this method is used as boundary condition on the outflow boundary, reflections arise which spoil the solution within a few wave periods. Instead Higdon's radiation condition was used. For the model problem of a 5 m high wave of length 60 m on 10 m waterdepth (see Section 2.4.3), it was shown that both in two and in three dimensions this wave can be simulated over about 5 wave periods in a stable way.

The same wave conditions are used in the present verification to test the accuracy and stability when simulating over many more wave periods. Because of the large computation time involved with the three-dimensional model and the length of the simulation we only tested the two-dimensional model.

The more accurate algorithms described in Sections 4.2 and 4.3, are used in the computations presented here. The second-order time derivatives are determined using the algorithms developed by Broeze. By comparing computations with and without the use of these time derivatives, its effects are investigated. To that purpose we first consider the use of Sommerfeld's radiation condition.

4.5.1 Sommerfeld's radiation condition

A difficulty associated with the use of Higdon's radiation condition is that in its traditional formulation, derived from Sommerfeld's condition, it is not consistent with the dynamic free-surface boundary condition (2.8). The solution given in equation (2.43) satisfies the dynamic boundary condition but does not satisfy Sommerfeld's condition because of the term $-(g(R-h)-\frac{1}{2}c^2)t$. For Sommerfeld's condition to be consistent with the dynamic boundary condition (2.8) and solution (2.43) an extra term has to be added:

$$\frac{\partial \phi}{\partial t} = c\frac{\partial \phi}{\partial n} - \left(g(R-h) - \frac{1}{2}c^2 \right), \ \mathbf{x} \in S_{out}. \tag{4.16}$$

The omission of this term causes the generation of an extra current on the outflow boundary. In the simulation with the numerical method this is noticed by a stretching of the domain and consequently a decrease of the mean water level, see Figure 4.4. Averaged over the domain the computed surface elevation η_c without the extra term is approximately 0.04 m below the surface elevation η_e as given by equation (2.42). Because of the decrease of the mean water-level in the simulation, the phase velocity of the simulated wave becomes smaller causing a phase error, which explains the large error in surface elevation near the second wave top.

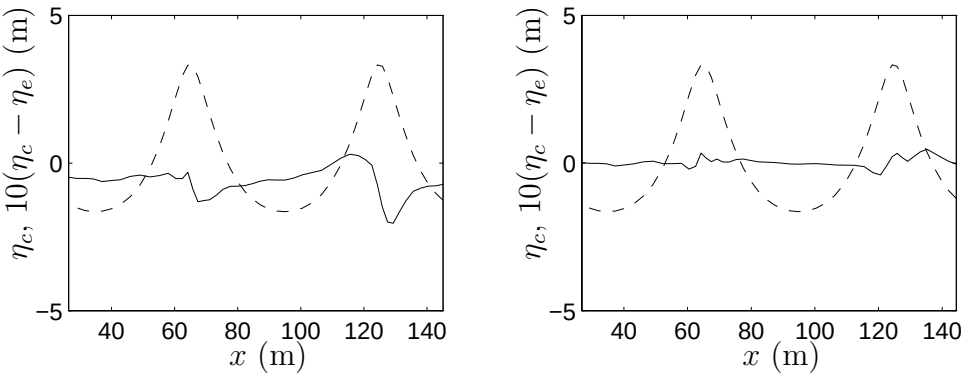

Figure 4.4 Computed free-surface elevation η_c $(--)$ and the error $10(\eta_c - \eta_e)$ $(-)$ at $t = 30$ s in the computation without (left figure) and with (right figure) the extra term $g(R-h) - \frac{1}{2}c^2$ in Sommerfeld's radiation condition.

This error cannot be noticed in the results shown by Broeze because these results are presented in a way which is too coarse to identify the error. In the verification studies to follow we use the corrected Sommerfeld's condition on the outflow boundary. It is remarked that for simulations of waves in more practical problems it is important to have good indications of phase speed and direction of propagation. Reflections generated by omitting the extra term are probably negligible compared with reflections due to the use of incorrect values for the latter quantities.

4.5.2 Second-order time derivatives

To examine the effect of the second-order time derivatives we have used three variants, a, b and c, of the 2-stage-2-derivative method (RK-2-2) with decreasing contributions of the second-order time derivative. The coefficients used in these methods are given by Broeze [13], Section 6.1.3, and are repeated in Appendix A. Also the classical fourth-order Runge-Kutta method (RK-4) has been used.

The domain in this test is 120 m long and consists of 50 panels on the free surface, 30 on the bottom and 10 panels on the inflow and outflow boundaries. Computations have been done for waveheights $H = 5.0$ m ($T \approx 6.55$ s) and $H = 5.5$ m ($T \approx 6.46$ s) which is almost 92 % of the highest wave that is physically stable.

In Table 4.1 at page 50, the duration of the computation is given in terms of the number of wave periods that can be simulated before the computation breaks down.

The breakdown of the computations is caused by slowly growing $2\Delta x$-instabilities. Figure 4.5 shows the maximum absolute error in the computed

Table 4.1 Stability of the method for two wave conditions and for different Runge-Kutta time stepping methods expressed in terms of numbers of wave periods that can be run.

	$H = 5.0$ m	$H = 5.5$ m
RK-2-2-a	134.05 s $\approx 20.5\ T$	40.85 s $\approx 6.3\ T$
RK-2-2-b	192.00 s $\approx 29.3\ T$	85.70 s $\approx 13.3\ T$
RK-2-2-c	228.85 s $\approx 34.9\ T$	85.85 s $\approx 13.3\ T$
RK-4	1074.60 s $\approx 164.1\ T$	638.80 s $\approx 98.8\ T$

free-surface elevation for the computations using RK-2-2-c and RK-4. In Figure 4.6 the error in the computed free-surface elevation is shown at 4 time levels for the computation using the classical Runge-Kutta method RK-4.

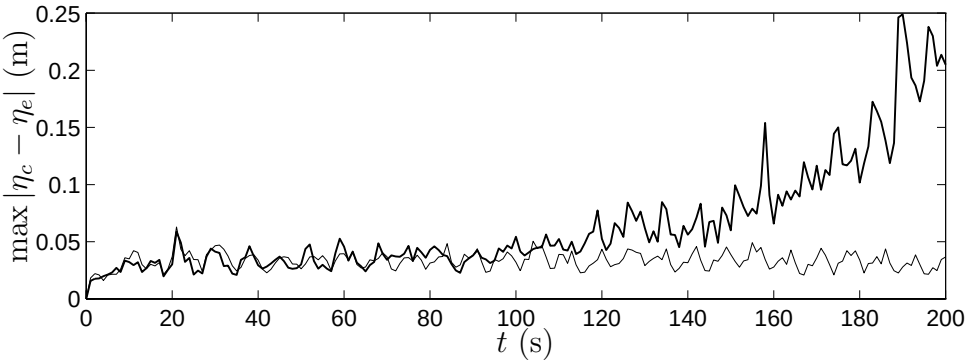

Figure 4.5 Maximum absolute error in free-surface elevation during the time interval $[0, 200]$ s, for the computations using RK-2-2-c (thick line) and RK-4 (thin line).

4.6 Conclusions

In this chapter we have examined parts of the numerical method and considered accuracy and stability of these parts in time-domain computations.

The description of position and movement of intersections between networks is improved with the use of distances in the physical domain instead of distances in the computational domain.

The use of a grid velocity different from the material velocity and the material derivative related to this grid velocity complicates the determination of second-order quantities as used in the two-stage two-derivative Runge-Kutta

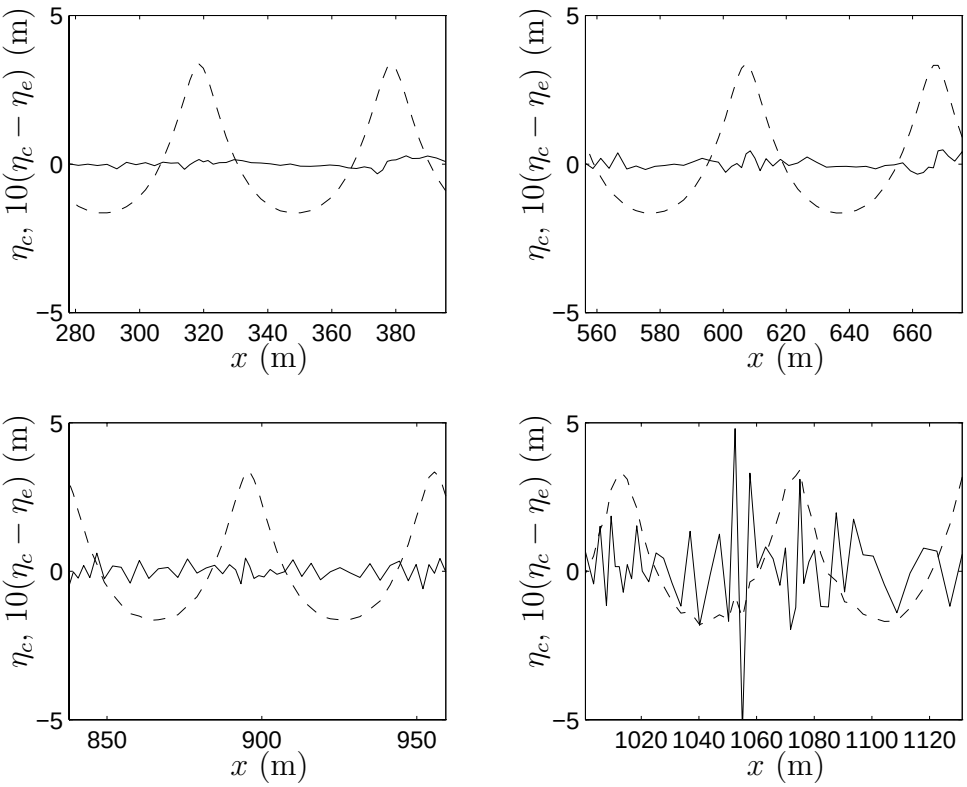

Figure 4.6 Computed free-surface elevation η_c $(--)$ and the error $10(\eta_c - \eta_e)$ $(-)$ at $t = 300$ s (upper left), $t = 600$ s (upper right), $t = 900$ s (lower left) and $t = 1070$ s (lower right).

method. It is shown that the present determination introduces errors in the grid acceleration and the second-order material derivative of ϕ. Numerical experiments using highly nonlinear waves over many wave periods reveal that computations using the two-stage two-derivative Runge-Kutta method are less stable than computations using the classical Runge-Kutta method. This may be related to the loss of accuracy in the former method.

The standard Sommerfeld's radiation condition does not apply to the potential in the case of nonlinear waves. It imposes an additional current on the outflow boundary. In computations with the numerical model this manifests itself by stretching of the domain and a decrease of the mean water-level. By adding an extra term to the standard Sommerfeld's condition the accuracy of the computations can largely be improved.

It is shown that stable computations over many wave periods can be carried out. Eventually computations are spoiled by slowly growing $2\Delta x$-instabilities. Closer study is needed to identify the cause of this instability.

Chapter 5

Domain decomposition

5.1 Introduction

As we have seen in the previous chapters, computations are often restricted to small domains and short simulation times due to limited computational power. But even though computer capacity is growing extremely fast during the last decades, there is another aspect which needs to be paid attention to if one wants to enlarge simulations with the present numerical method. This aspect is the superlinear dependence of the computational costs on the number of collocation points used.

The superlinear dependence of the computational costs is obtained when the boundary integral equation method is applied without using any information from the problem on hand, which can be used to decrease the computational costs. For example, for pairs of panels and field points that are far enough apart, the corresponding influence coefficients can be neglected and do not have to be calculated. The question is, however, how and to which degree such refinements of the boundary integral equation method can be applied in order to obtain a more favourable relation between computational costs and number of collocation points.

A promising approach towards this problem is the use of a domain decomposition method. The domain is divided into subdomains and the boundary integral equation method is applied to the separate subdomains. The computational effort per subdomain is much smaller. The efficiency of the domain decomposition method is determined by the extra computational effort which is required to reach equivalence of the decomposed problem with the original problem.

Since the early 1980's domain decomposition methods have received much attention, especially from people working with field discretization techniques. The most important motive in these studies is to increase the efficiency of

solvers, which is one of the aspects of importance in boundary integral equation methods as well. Other important motives, also valid for boundary integral equation methods, are the use of parallel computers and the wish to decompose problems in mathematical physics into subproblems of simplified geometry and/or simplified mathematical description. This interest resulted in a series of yearly conferences on domain decomposition methods reported of in [30], [15] and following.

In the field of boundary integral equation methods, theory of domain decomposition techniques has been developed by [37]. Application to time-domain methods for nonlinear water waves in two dimensions was presented by Wang et al. [69] and by Wu and Eatock Taylor [71]. However, no details about the efficiency of the methods are presented there. A somewhat different succesful approach using subdomains in both two and three dimensions has been proposed by Fenton and Kennedy [25] who employ a local polynomial approximation in all subdomains.

In this thesis we focus on a particular domain decomposition method and we outline the abilities of this method for present and future applications. The domain decomposition method that we describe here consists of a division of the computational domain into subdomains and of an iterative (coupling) procedure which generates a sequence of boundary conditions on the interfaces between the subdomains. This sequence converges to conditions corresponding to the solution of the original problem. The solution of the decomposed problem then converges to the solution of the original problem.

The remainder of this chapter is divided as follows. In Section 5.2 the domain decomposition method for Laplace's equation is described. In Section 5.3 the relation of the method with previous work in the field of domain decomposition is described.

5.2 Description

We consider Laplace's equation on a domain Ω with prescribed boundary conditions. As illustrated in Figure 5.1, we decompose Ω into the subdomains Ω_1 and Ω_2 which are separated by an interface Γ. To obtain a unique definition of the normal derivative on the interface we choose

$$\mathbf{n}(\mathbf{x}) = \mathbf{n}_2(\mathbf{x}) = -\mathbf{n}_1(\mathbf{x}), \ \mathbf{x} \in \Gamma. \tag{5.1}$$

Boundary conditions on the interface for Laplace's equation in the separate subdomains are not known beforehand, and as an initial guess a Dirichlet condition φ on Γ is imposed. If φ equals the exact solution of Laplace's equation on the original domain at the interface location ($\varphi = \phi|_\Gamma$), then the solutions ϕ_1 and ϕ_2 in the respective subdomains equal the solution ϕ of the original domain.

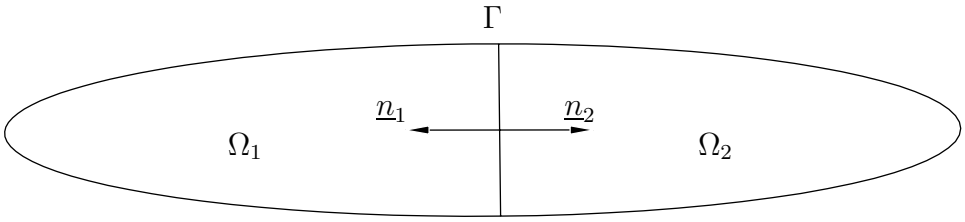

Figure 5.1 Decomposed domain.

Also the gradients $\nabla\phi_1$ and $\nabla\phi_2$ will then be continuous over the interface. Because of the uniqueness of the solution of Laplace's equation any other Dirichlet condition on the interface leads to a discontinuity in the gradient on the interface. In the same way will any Neumann condition imposed on the interface and different from $\psi = \nabla\phi|_\Gamma \cdot \mathbf{n}$, lead to a discontinuity in the potential itself over the interface.

This criterium can be formulated using the Steklov-Poincaré operator T defined as

$$T : \varphi \rightarrow \nabla\phi_1|_\Gamma \cdot \mathbf{n} - \nabla\phi_2|_\Gamma \cdot \mathbf{n}. \tag{5.2}$$

The boundary condition φ of the solution ϕ of the original, one-domain problem, satisfies the equation

$$T\varphi = 0. \tag{5.3}$$

See Agoshkov [1] and the references therein for more information about this operator.

The correct boundary conditions can be found by generating a sequence of boundary conditions such that the discontinuity in the computed variable ($\nabla\phi_1|_\Gamma \cdot \mathbf{n} - \nabla\phi_2|_\Gamma \cdot \mathbf{n}$ if a Dirichlet condition is imposed and $\phi_1|_\Gamma - \phi_2|_\Gamma$ if a Neumann condition is imposed) converges to zero. There are different ways to construct such a sequence and the procedure which will be used in this thesis is the following scheme, referred to as the D/D-N/N-scheme. For shorthand we define $\frac{\partial\phi_i^{(k)}}{\partial n} = \nabla\phi_i^{(k)}|_\Gamma \cdot \mathbf{n}$.

0. Choose an initial Dirichlet condition $\varphi^{(k)}$, $(k = 0)$.

1. Solve $\Delta\phi_1^{(k)} = 0$ in Ω_1 and $\Delta\phi_2^{(k)} = 0$ in Ω_2 with Dirichlet condition $\phi_1^{(k)} = \phi_2^{(k)} = \varphi^{(k)}$ on Γ (D/D). This yields $\frac{\partial\phi_1^{(k)}}{\partial n}$ and $\frac{\partial\phi_2^{(k)}}{\partial n}$.

2. Formulate a Neumann condition $\psi^{(k+1)}$ by taking a weighted average of the computed solutions for some weighting factor $\omega_N^{(k)}$:

$$\psi^{(k+1)} = \omega_N^{(k)} \frac{\partial\phi_1^{(k)}}{\partial n} + (1 - \omega_N^{(k)}) \frac{\partial\phi_2^{(k)}}{\partial n}. \tag{5.4}$$

3. Solve $\Delta\phi_1^{(k+1)} = 0$ in Ω_1 and $\Delta\phi_2^{(k+1)} = 0$ in Ω_2 with Neumann condition $\frac{\partial\phi_1^{(k+1)}}{\partial n} = \frac{\partial\phi_2^{(k+1)}}{\partial n} = \psi^{(k+1)}$ on Γ (N/N). This yields $\phi_1^{(k+1)}$ and $\phi_2^{(k+1)}$.

4. Formulate a Dirichlet condition $\varphi^{(k+2)}$ by taking a weighted average of the computed solutions for some weighting factor $\omega_D^{(k)}$:

$$\varphi^{(k+2)} = \omega_D^{(k+1)}\phi_1^{(k+1)}|_\Gamma + (1 - \omega_D^{(k+1)})\phi_2^{(k+1)}|_\Gamma. \tag{5.5}$$

5. Repeat procedure steps 1 to 4 ($k := k + 2$) till convergence is reached.

In the case of more subdomains (and thus more interfaces) the scheme can be generalized by performing each step in the scheme on all subdomains and for all interfaces simultaneously respectively. The exchange of information on every interface concerns only the neighbouring subdomains.

In the numerical approach the Laplace problems are solved with the panel method in all subdomains separately. Anticipating the quantative description in Chapter 7, the following implications with respect to efficiency can be given:

- By introducing the interfaces, extra panels are needed in the panel method. However, because the memory required per subdomain depends quadratically on the number of panels in the considered subdomain, the total required memory can be reduced considerably, depending on the number of interfaces.

- A similar reasoning can be given for the required computation time to determine the source and dipole coefficients. These coefficients have to be determined only at the beginning of the iterative procedure, since the geometry does not change during the iterative process. Because of the quadratic dependence of the required computation time on the number of collocation points per subdomain, the total costs over all subdomains can be considerably smaller.

- There is a similar reduction of required computation time for solving the linear system of equations for one iteration step. However, because now this has to be done every step of the iterative procedure, the total costs for this part of the method depend also on the convergence of the iterative procedure.

We have formulated the scheme for Laplace's equation with inhomogeneous boundary conditions on the outer boundaries. It is, however, important to realize that we only generate a sequence of boundary conditions on the interface and that the conditions on the outer boundaries are fixed during the iteration process. We can therefore simplify the description in the following way.

Because of the linearity of Laplace's equation, the convergence of the solution of the Laplace problems with inhomogeneous boundary conditions is equivalent with the convergence of the solution of the Laplace problems for the error functions $\tilde{\phi}_i^{(k)} = \phi|_{\Omega_i} - \phi_i^{(k)},\ i = 1, 2$, with homogeneous boundary conditions on the outer boundaries:

$$\begin{cases} \Delta\tilde{\phi}_i^{(k)} = 0 & \text{in } \Omega_i \\ \tilde{\phi}_i^{(k)} = 0 & \text{on } \partial\Omega_i \backslash \Gamma \\ \tilde{\phi}_i^{(k)} = \tilde{\varphi}^{(k)} & \text{on } \Gamma \end{cases}, \ i = 1, 2. \tag{5.6}$$

For these problems the interface boundary conditions have to converge to 0. The convergence of the iterative procedure is the subject of Chapter 6.

5.3 Relation to previous work

The earliest known iterative domain decomposition technique was proposed in the pioneering work of H.A. Schwarz in 1870 [64] to prove the existence of harmonic functions on irregular regions which are the union of overlapping subregions. Variants of Schwarz's method were later studied by Sobolev (1936), Morgenstern (1956) and Babuška (1957). See also Courant and Hilbert (1962). In Schwarz' approach, the technique is applied directly to the partial differential equation, like the description in Section 5.2.

The domain decomposition technique can also be applied to discretizations of the differential equations. In both approaches suitable continuity requirements between adjacent subproblems have to be enforced. The resulting matrix of the system of linear equations can be written as a block-structured matrix. In most studies on domain decomposition techniques for these problems, the focus is on finding suitable iterative methods, particularly on the formulation of proper preconditioners for these methods. The idea behind this approach is that if discretized equations for grid points which are close together, are strongly related, it is better to solve these equations (to a large degree) locally, i.e. in the blocks.

Contrary to the resulting matrix in a field discretization method, the submatrices corresponding to the subdomain problems are full. In fact, the approach described in Section 5.2 can be represented by a block-structured matrix after discretization. In our approach, however, we do not consider the matrix formulation for the total problem. Instead, the subdomain problems are formulated on the continuous level and the numerical solution is sought for the subdomains separately. It is open to determine which solver is used for which subdomain problem. The use of a block-structured matrix to represent the total system of linear equations for all subdomains, in connection with a boundary integral equation method, is described by e.g. Wang et al. [69].

An important distinction which is made in field discretization techniques is the use of either overlapping or non-overlapping subdomains. The use of overlapping subdomains does not seem to be very useful for boundary integral equation methods because the boundaries of the overlap between two subdomains are not shared by both subdomains in the discretization. See Figure 5.2.

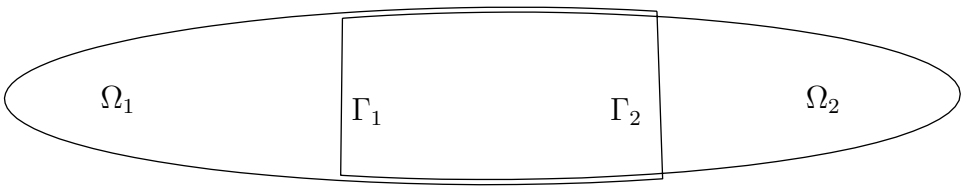

Figure 5.2 Overlapping subdomains. Note that Γ_1 is not a boundary of subdomain Ω_1 and Γ_2 is not a boundary of subdomain Ω_2.

The implementation of the D/D-N/N scheme into a field discretization technique is also known as a Neumann-Neumann preconditioner, see e.g. [45]. Other coupling schemes are possible as well. The following scheme (referred to as D/*-*/N) is known as the Neumann-Dirichlet preconditioner.

0. Choose an initial Dirichlet condition $\varphi^{(k)}$, $(k = 0)$.

1. Solve $\Delta\phi_1^{(k)} = 0$ in Ω_1 with Dirichlet condition $\phi_1^{(k)} = \varphi^{(k)}$ on Γ (D/*). This yields $\frac{\partial\phi_1^{(k)}}{\partial n}$.

2. Formulate this solution as a Neumann condition $\psi^{(k+1)}$:

$$\psi^{(k+1)} = \frac{\partial\phi_1^{(k)}}{\partial n}. \tag{5.7}$$

3. Solve $\Delta\phi_2^{(k+1)} = 0$ in Ω_2 with Neumann condition $\frac{\partial\phi_2^{(k+1)}}{\partial n} = \psi^{(k+1)}$ on Γ (*/N). This yields $\phi_2^{(k+1)}$.

4. Formulate a Dirichlet condition $\varphi^{(k+2)}$ by taking a weighted average of the computed solutions for some weighting factor $\omega_D^{(k+1)}$:

$$\varphi^{(k+2)} = \omega_D^{(k+1)}\phi_2^{(k+1)}|_\Gamma + (1 - \omega_D^{(k+1)})\varphi^{(k)}. \tag{5.8}$$

5. Repeat procedure steps 1 to 4 $(k := k + 2)$ till convergence is reached.

The generalization of this scheme to the case of more than two subdomains is not as straightforward as in the D/D-N/N-scheme. Interfaces may be characterized as either Dirichlet boundaries or Neumann boundaries for the various steps

in the iteration procedure. In both schemes care has to be taken that problems may become indeterminate when only Neumann conditions are imposed in a subdomain. Convergence properties as well as computational implementations are different for both schemes. We will discuss these aspects in Chapters 6 and 7.

In the schemes mentioned, the coupling between subdomains is performed locally. An important topic in domain decomposition is the use of global coupling which can be essential to obtain convergence in the case of many non-overlapping subdomains. The need to have a global coupling in the application to Laplace's equation and with the subdivisions used will be discussed in Chapter 6.

The investigations on domain decomposition in this thesis are especially aimed at application of the method in time-domain computations of nonlinear water waves. Therefore the domains which are studied are generally domains with large length-to-height ratios and only subdivisions in the horizontal direction with vertical interfaces are considered. For rectangular subdomains, the iterative process is studied using a Fourier mode analysis. Such an analysis has been given by Funaro et al. [27] for the D/*-*/N scheme for two subdomains in 2-D. Emphasis in this study was on the optimal choice of the weighting factor ω. In this thesis, in Chapter 6, the D/D-N/N scheme is studied and formulations are developed for the multi-subdomain case and for 3-D problems.

Furthermore the iterative scheme is studied for non-rectangular subdomains which occur in time-domain computations with wave disturbances at the free surface and uneven bottoms. Large parts of the studies presented in Chapters 6 and 7 have been reported in [32].

Chapter 6

Convergence studies

6.1 Introduction

As the efficiency of the domain decomposition method is determined by the convergence of the iterative procedure, knowledge about the convergence is of great importance. In this chapter the convergence of the iterative procedure for Laplace's equation as it occurs in our wave problems is analyzed for a number of simplified geometries. The results are presented as a function of the geometry of the domain.

The subdivisions of the domain considered here are restricted to subdivisions in horizontal directions. In general the water-wave problems of interest to us have horizontal scales which are much larger than the vertical scale. Also in deep water, the fluid motion due to waves is restricted to the upper part. From the criterium for deep water ($h/\lambda > \frac{1}{2}$) the relevant vertical scale can be approximated as $\frac{1}{2}\lambda$. Subdivisions in the horizontal directions lead to subdomains with smaller length-to-height ratios and the introduction of vertical interfaces which are small relative to the horizontal scale. It is remarked here that in three-dimensional problems involving objects and complex geometries, one may want to subdivide the domain in a more general way. An example is the area between a ship and a quay where the distance between ship and quay can be smaller than the water depth.

This chapter is organized as follows. First, in Section 6.2, we consider rectangular subdomains. These geometries allow an analytical description of the convergence behaviour and the notions used to describe convergence are introduced here. In Section 6.3 results of numerical experiments are presented for subdomains representing a horizontal bottom and a free surface with a wave-like disturbance. In Section 6.4 the geometry related to more general water wave problems with uneven bottoms is considered.

6.2 Results for rectangular domains

As a first simplification of the domain, we consider a rectangular domain divided into a number of subdomains in the horizontal direction. In the case of rectangular subdomains we can solve the Laplace problems analytically and the convergence of the iterative process can be study fairly simply through a Fourier mode analysis. See also Funaro et al. [27] who applied the Fourier mode analysis to the D/*-*/N scheme for the two-subdomain problem.

To simplify notation we consider only problems with (homogeneous) Dirichlet conditions on the outer boundaries. The Laplace problems originating from water wave problems have a Neumann condition on the bottom. For problems with a horizontal bottom we can, however, reflect the homogeneous Laplace problem in the bottom to obtain a Laplace problem with only Dirichlet conditions. The height of the domain is of course twice that of the original one then.

6.2.1 Two subdomains

We investigate the convergence of the D/D-N/N scheme for a Dirichlet problem with two subdomains characterized by the dimensions given in Figure 6.1:

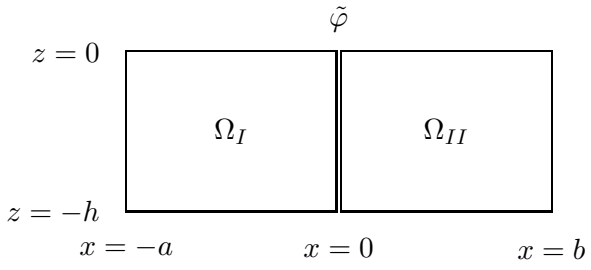

Figure 6.1 Definition of the domain.

In the iterative process according to the D/D-N/N-scheme, two types of Laplace problems are defined in the subdomains. One with Dirichlet conditions on the interface (procedure step 1) and one with Neumann conditions on the interface (procedure step 3). The solutions of these two problems can easily be obtained by separation of variables. The convergence of the iterative procedure can then be expressed in terms of the consecutive solutions.

We first formulate the solutions on both subdomains with a Dirichlet condition on the interface $\Gamma = \{(x, z) \in \Omega | x = 0\}$. The original homogeneous

Dirichlet problem

$$\begin{cases} \Delta\tilde{\phi} = 0 & \text{in } \Omega \\ \tilde{\phi} = 0 & \text{on } \partial\Omega \end{cases} \tag{6.1}$$

is decomposed into problems with the initial Dirichlet boundary condition $\tilde{\varphi}$ on the interface Γ:

$$\begin{cases} \Delta\tilde{\phi}_i = 0 & \text{in } \Omega_i \\ \tilde{\phi}_i = 0 & \text{on } \partial\Omega_i \backslash \Gamma \\ \tilde{\phi}_i = \tilde{\varphi} & \text{on } \Gamma \end{cases} \quad , \; i = 1, 2. \tag{6.2}$$

We represent the boundary condition on the interface by the Fourier series

$$\tilde{\varphi}(z) = \sum_{n=1}^{\infty} c_n \sin\left(\frac{n\pi}{h} z\right). \tag{6.3}$$

The solution in Ω_I then equals

$$\tilde{\phi}_1(x, z) = \sum_{n=1}^{\infty} c_n \frac{\sinh\left(\frac{n\pi}{h}(x+a)\right)}{\sinh\left(\frac{n\pi}{h}a\right)} \sin\left(\frac{n\pi}{h} z\right), \tag{6.4}$$

and we have a similar expression for the solution in Ω_{II}. This leads to the following expression for $\frac{\partial\tilde{\phi}_1}{\partial x}$ on the interface:

$$\left.\frac{\partial\tilde{\phi}_1}{\partial x}\right|_{\Gamma} = \sum_{n=1}^{\infty} c_n \frac{n\pi}{h} \left(\tanh\left(\frac{n\pi}{h}a\right)\right)^{-1} \sin\left(\frac{n\pi}{h} z\right) =: \sum_{n=1}^{\infty} \alpha_{1,n} c_n \sin\left(\frac{n\pi}{h} z\right). \tag{6.5}$$

and a similar one for $\frac{\partial\tilde{\phi}_2}{\partial x}$ on the interface. In these expressions each separate Fourier mode $c_n \sin\left(\frac{n\pi}{h} z\right)$ is multiplied by a factor $\alpha_{1,n}$ and $\alpha_{2,n}$ respectively, which is independent of the Fourier coefficients c_n.

To arrive at a Neumann boundary condition $\tilde{\psi}$ we take a weighted average $\omega_N \frac{\partial\tilde{\phi}_1}{\partial x}|_\Gamma + (1-\omega_N)\frac{\partial\tilde{\phi}_2}{\partial x}|_\Gamma$ of both solutions. An important observation is that the averaging formula can be applied to the separate Fourier modes of $\frac{\partial\tilde{\phi}_i}{\partial x}|_\Gamma$. Therefore, we can represent this step in the iterative procedure for each Fourier mode $\tilde{\varphi}_n$ separately by the following diagram:

$$\begin{array}{c} \tilde{\varphi}_n := c_n \sin\left(\frac{n\pi}{h} z\right) \\ \swarrow \qquad\qquad \searrow \\ \frac{\partial\tilde{\phi}_1}{\partial x}|_\Gamma = \alpha_{1,n}\tilde{\varphi}_n \qquad \frac{\partial\tilde{\phi}_2}{\partial x}|_\Gamma = \alpha_{2,n}\tilde{\varphi}_n \\ \searrow \qquad\qquad \swarrow \\ \tilde{\psi}_n = \omega_N \frac{\partial\tilde{\phi}_1}{\partial x}|_\Gamma + (1-\omega_N)\frac{\partial\tilde{\phi}_2}{\partial x}|_\Gamma \\ = (\omega_N \alpha_{1,n} + (1-\omega_N)\alpha_{2,n})\tilde{\varphi}_n \end{array}$$

In the odd steps of the iterative procedure a Neumann condition is imposed on the interface. The solution process of these steps can be represented by a similar diagram as above:

$$
\begin{array}{c}
\tilde{\psi}_n := c'_n \sin\left(\frac{n\pi}{h} z\right) \\
\swarrow \qquad\qquad \searrow \\
\tilde{\phi}_1|_\Gamma = \beta_{1,n}\tilde{\psi}_n \quad \| \quad \tilde{\phi}_2|_\Gamma = \beta_{2,n}\tilde{\psi}_n \\
\searrow \qquad\qquad \swarrow \\
\tilde{\varphi}_n = \omega_D \tilde{\phi}_1|_\Gamma + (1-\omega_D)\tilde{\phi}_2|_\Gamma \\
= (\omega_D \beta_{1,n} + (1-\omega_D)\beta_{2,n})\tilde{\psi}_n
\end{array}
$$

The coefficients $\beta_{1,n}$ and $\beta_{2,n}$ are equal to $\alpha_{1,n}^{-1}$ and $\alpha_{2,n}^{-1}$ respectively: calculating $\frac{\partial \tilde{\phi}_i}{\partial x}|_\Gamma$ for a given $\tilde{\phi}_i|_\Gamma$ is the inverse operation of calculating $\tilde{\phi}_i|_\Gamma$ for a given $\frac{\partial \tilde{\phi}_i}{\partial x}|_\Gamma$.

Now that we have formulated the solutions of all the Laplace problems that occur during the iterative process, we can express the rate of convergence in terms of the aforementioned parameters. The D/D-N/N scheme can be represented as:

$$\tilde{\varphi}^{(0)} \to \tilde{\psi}^{(1)} \to \tilde{\varphi}^{(2)} \to \tilde{\psi}^{(3)} \to \tilde{\varphi}^{(4)} \to \tilde{\psi}^{(5)} \to \dots \tag{6.6}$$

The change in the interface values of the potential from step k to step $k+2$ can be expressed per Fourier mode $\tilde{\phi}_n^{(k)} = c_n^{(k)} \sin\left(\frac{n\pi}{h} z\right)$ through

$$c_n^{(k+2)} = \epsilon_n^{(k)} c_n^{(k)},\ k \geq 0,\ k \text{ even} \tag{6.7}$$

with

$$\epsilon_n^{(k)} := (\omega_D^{(k+1)}\beta_{1,n} + (1-\omega_D^{(k+1)})\beta_{2,n})(\omega_N^{(k)}\alpha_{1,n} + (1-\omega_N^{(k)})\alpha_{2,n}),\ k \geq 0,\ k \text{ even}. \tag{6.8}$$

If we take $\omega_N^{(k)} = \omega_D^{(k+1)} = \omega$ for all iteration steps k, k even, then $\epsilon_n = \epsilon_n^{(k)}$ is independent of k and the convergence process can be characterized by this parameter. We call ϵ_n the amplification factor of the n-th Fourier mode. In this case the Fourier coefficients $c_n^{(k)}$ can be expressed in terms of the Fourier coefficients $c_n^{(0)}$ of the starting value $\tilde{\varphi}^{(0)}$:

$$c_n^{(k)} = \epsilon_n c_n^{(k-2)} = \epsilon_n^{k/2} c_n^{(0)},\ k \geq 2,\ k \text{ even}. \tag{6.9}$$

If $|\epsilon_n| < 1$ then the iterative procedure is convergent for the n-th Fourier mode.

Working out equation (6.8) we find for the D/D-N/N scheme

$$\epsilon_n = (2+\gamma_n)\omega^2 - (2+\gamma_n)\omega + 1 = 1 - (2+\gamma_n)\omega(1-\omega), \tag{6.10}$$

where

$$\gamma_n = \frac{\tanh\left(\frac{n\pi}{h}b\right)}{\tanh\left(\frac{n\pi}{h}a\right)} + \frac{\tanh\left(\frac{n\pi}{h}a\right)}{\tanh\left(\frac{n\pi}{h}b\right)}. \tag{6.11}$$

In the analysis of this expression we make the distinction between the case of subdomains of equal length ($a = b$) and the case of subdomains of unequal length ($a \neq b$).

If $a = b$ we have $\gamma_n = 2$ and $\epsilon_n = 4(\omega - \frac{1}{2})^2$. So the amplification factor ϵ_n is independent of n. If we choose $\omega = \frac{1}{2}$ we find $\epsilon_n = 0$. Thus the exact result is found after one iteration step. This can be explained by considering the geometry of the domain; because of the symmetry we find opposed horizontal derivatives on the interface, which by averaging ($\omega = \frac{1}{2}$) directly leads to the correct Neumann boundary condition $\tilde{\psi}^{(0)} = 0$.

If $a \neq b$ then different Fourier modes have different amplification factors. For a given geometry the range of γ_n as function of n is limited. Therefore also the range of the amplification factors ϵ_n is limited: ϵ_n is an increasing function of n with lower bound $\epsilon_1 = 1-(2+\gamma_1)\omega(1-\omega)$ and upper bound $\epsilon_\infty = 4(\omega-\frac{1}{2})^2$. Notice that if the values of a and b are exchanged, γ_n and consequently ϵ_n do not change. This contrary to the amplification factor for the D/*-*/N worked out in Appendix B.

More information about the relation between convergence and geometry can be obtained when we express γ_n in terms of the length-to-height ratios of the subdomains. We take

$$\hat{a} = \frac{a}{h} \text{ and } \hat{b} = \frac{b}{h}. \tag{6.12}$$

Then we can express γ_n as

$$\gamma_n = \frac{\tanh n\pi\hat{b}}{\tanh n\pi\hat{a}} + \frac{\tanh n\pi\hat{a}}{\tanh n\pi\hat{b}}. \tag{6.13}$$

For a fixed value of ω and varying values of $\hat{a}$ and $\hat{b}$ the contour plots in Figure 6.2 indicate the amplification factor ϵ_1 for different geometrical configurations. Notice that by taking the multiples $(n\hat{a}, n\hat{b})$, we can find the amplification factors ϵ_n of the consecutive Fourier modes, i.e. $\epsilon_n(\hat{a}, \hat{b}) = \epsilon_1(n\hat{a}, n\hat{b})$.

Figure 6.2 indicates the following. For a large set of geometrical configurations the iterative procedure is convergent for all Fourier modes if $\omega = 0.50$. For very small length-to-height ratios in one of the subdomains the D/D-N/N scheme does not converge for the first modes. When the length-to-height ratio is larger than 2 for both subdomains, the iterative procedure converges very fast for all Fourier modes. The jump $|\tilde{\phi}_1^{(k)} - \tilde{\phi}_2^{(k)}|$ is reduced by a factor smaller than 10^{-10} going from iteration step k to $k + 2$.

We can also see that the set of geometrical configurations for which the iterative procedure is convergent for $\omega = 0.49$ is approximately the same as for $\omega = 0.5$. However, for large length-to-height ratios the amplification factors of all Fourier modes are approximately equal to the upper bound $\epsilon_\infty = 0.0004$. The 0-contour line indicates the set of geometrical configurations for which the amplification factor of the first Fourier mode equals 0. Nevertheless, the amplification factors of the higher modes in the same configuration have a positive value and converge much more slowly.

These results can be interpreted as follows. For subdomains with *equal* length, the solutions generated in the iterative procedure are symmetric with

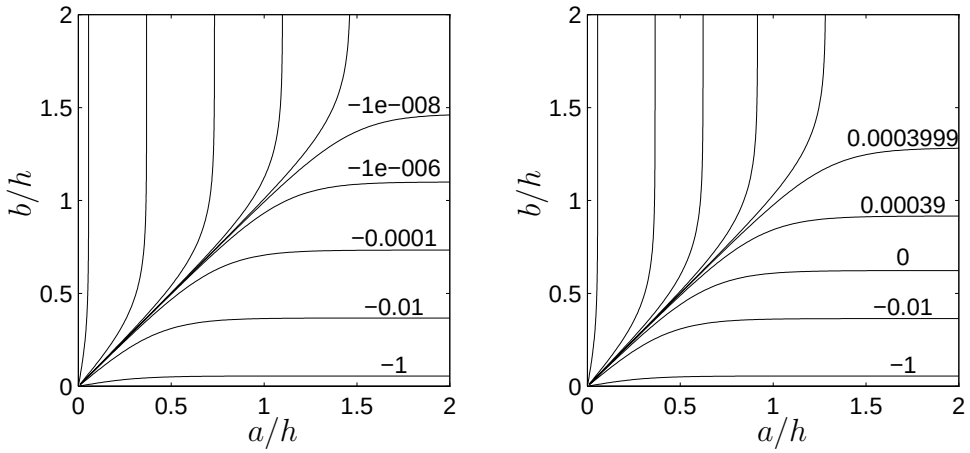

Figure 6.2 Contour plots for ϵ_1; left graph: $\omega = 0.50$, right graph: $\omega = 0.49$. Because of the symmetry of the plot - see the previous remark about interchanging the values of a and b - the values of the amplification factors for the contourlines in the upper left parts have not been indicated.

respect to the boundary condition on the interface. A symmetric averaging ($\omega = \frac{1}{2}$) annihilates the error made in the initial boundary condition.

For subdomains with *unequal* lengths, the solutions generated in the iterative procedure are asymmetric. However, this asymmetry is hardly noticeable when the length-to-height ratios $\hat{a}$ and $\hat{b}$ are large, especially for the higher Fourier modes. Taking $\omega = \frac{1}{2}$ then also leads to very fast convergence.

So far the two-subdomain problem has been analyzed for problems with homogeneous Dirichlet conditions, also on the vertical boundaries $z = -a$ and $z = b$. In the case of Neumann conditions on either one or both of these boundaries, the subdomain solutions have a slightly different form which results in a different expression for γ_n. In the case of a Neumann boundary at $x = -a$ and a Dirichlet boundary at $x = b$ for example, we have $\gamma_n = \tanh(n\pi\hat{a})\tanh(n\pi\hat{b}) + (\tanh(n\pi\hat{a})\tanh(n\pi\hat{b}))^{-1}$. Because of the fast convergence of the tanh-function to 1, the contour plots for ϵ_1 as shown in Figure 6.2 for this case are only different for very small length-to-height ratios.

6.2.2 Three or more subdomains

The iterative process in the multi-subdomain problem can be described similarly to the iterative process of the 2-subdomain problem in the previous subsection. We formulate and analyze it for the 3-subdomain problem first. Consider the domain shown in Figure 6.3.

In procedure step 1 (see Section 5.2) Dirichlet boundary conditions are im-

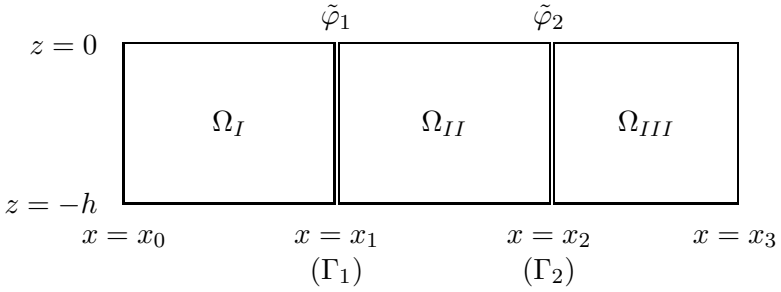

Figure 6.3 Definition of a domain with three subdomains.

posed on the interfaces and Neumann conditions are imposed in procedure step 3. We can again compute the solution of both types of problems in all subdomains. A difference with the 2-subdomain problem is now that the inner subdomain has two inhomogeneous boundary conditions instead of one. The solution there is the sum of the Laplace problem with $\tilde{\varphi}_1$ (or $\tilde{\psi}_1$) as boundary condition on Γ_1 and homogenous boundary conditions elsewhere plus the Laplace problem with $\tilde{\varphi}_2$ (or $\tilde{\psi}_2$) as boundary condition on Γ_2 with otherwise homogenous boundary conditions. Laplace problems with only one inhomogeneous boundary condition have already been discussed in Section 6.2.1 and expressions for the solutions in all subdomains can be expressed in terms of the solutions given there. A more detailed description is given in Appendix B.

Again, the iterative process for each Fourier mode can be described separately. The transformation of the n-th Fourier-mode $\tilde{\varphi}_{1,n}^{(k)} = c_{1,n}^{(k)} \sin(\frac{n\pi}{h} z)$ on interface Γ_1 and $\tilde{\varphi}_{2,n}^{(k)} = c_{2,n}^{(k)} \sin(\frac{n\pi}{h} z)$ on interface Γ_2 during the iterative process can be represented by the following transformation matrix $A_n^{(k)}$:

$$\begin{pmatrix} c_{1,n}^{(k+2)} \\ c_{2,n}^{(k+2)} \end{pmatrix} = \begin{pmatrix} A_{11,n}^{(k)} & A_{12,n}^{(k)} \\ A_{21,n}^{(k)} & A_{22,n}^{(k)} \end{pmatrix} \begin{pmatrix} c_{1,n}^{(k)} \\ c_{2,n}^{(k)} \end{pmatrix} =: A_n^{(k)} \begin{pmatrix} c_{1,n}^{(k)} \\ c_{2,n}^{(k)} \end{pmatrix} \tag{6.14}$$

The matrix $A_n^{(k)}$ can be expressed in terms of coefficients similar to $\alpha_{i,n}$ and $\beta_{i,n}$ in the 2-subdomain problem and the weighting factors $\omega_{i,D}^{(k)}$ and $\omega_{i,N}^{(k)}$, see Appendix B. In this way the iterative process is expressed by a system of homogeneous linear difference equations. For an initial condition $\mathbf{c}_n^{(0)}$ and fixed weighting factors, the solution of this difference equation can be expressed as

$$\mathbf{c}_n^{(k)} = A_n^k \mathbf{c}_n^{(o)}. \tag{6.15}$$

If and only if all eigenvalues lie inside the unit-circle, then the solution converges to zero, see e.g. Kelley and Peterson [39]. So if both eigenvalues $\epsilon_{1,n}$ and $\epsilon_{2,n}$ lay inside the unit-circle then the iterative process converges for the n-th Fourier mode.

For a general geometric configuration, the expressions for the eigenvalues are very complicated and involve too many parameters to be analyzed easily. Therefore we restrict ourselves to the case of subdomains of equal length l and $\omega = \frac{1}{2}$. The matrix A_n then has two eigenvalues $\epsilon_{1,n}$ and $\epsilon_{2,n}$ equal to:

$$\epsilon_{1,n} = -\frac{1}{4C(C+1)}, \tag{6.16}$$

and

$$\epsilon_{2,n} = -\frac{1}{4C(C-1)}, \tag{6.17}$$

with $C = \cosh(\frac{n\pi}{h}l)$, see Appendix B. The first Fourier-mode ($n = 1$) exhibits the slowest convergence. The condition $|\epsilon_1| < 1$ gives the following condition on the ratio $\frac{l}{h}$:

$$\frac{l}{h} > \pi^{-1}\text{arccosh}\left(\frac{1}{2} + \frac{1}{2}\sqrt{2}\right) \approx 0.2. \tag{6.18}$$

Contrary to the 2-subdomain problem with subdomains of equal length, the amplification factor is not equal to zero. The condition for convergence is somewhat stronger than the condition for the 2-subdomain problem with subdomains of unequal lengths.

When the subdomains are not of equal length, the eigenvalues can be determined for some given parameters. For large length-to-height ratio of all subdomains the choice of $\omega = \frac{1}{2}$ leads to very fast convergence on both interfaces for all Fourier modes. The symmetry argument used in the 2-subdomain case can be used here as well: Because of the large length-to-height ratio the solutions that are generated during the iterative process are determined mainly locally, i.e. they are hardly influenced by the boundary conditions on other interfaces. The solutions on either side of each interface are almost symmetric which explains the fast convergence.

The matrix representation can also be used in the case of a decomposition into four or more subdomains. For a decomposition into M subdomains, $M-1$ interfaces are present and $M-1$ independent initial Dirichlet conditions must be formulated. During the iteration process they are transformed and the transformation can be represented by a $M-1$ by $M-1$ matrix A_n for each Fourier mode.

The parameters in this matrix involve the M subdomain lengths and the weighting factor ω. To be able to say something about the convergence, we again consider the case of subdomains of equal length l and $\omega = \frac{1}{2}$. The matrix A_n is a five-diagonal matrix and upper bounds for its eigenvalues $\epsilon_{i,n},\ i = 1, \ldots M-1$ can be found using Gershgorin's theorem, see Appendix B. We have

$$|\epsilon_{i,n}| < \frac{1}{4}S^{-2},\ i = 1, \ldots, M-1. \tag{6.19}$$

We have $\frac{1}{4}S^{-2} < 1$ for all n if

$$\frac{l}{h} > \pi^{-1}\text{arcsinh}\left(\frac{1}{2}\right) \approx 0.15. \tag{6.20}$$

These results again show very fast convergence for a large length-to-height ratio.

6.2.3 Three-dimensional problems

The analysis and results are also applicable to Laplace problems in three dimensions. If we consider a block-shaped three-dimensional domain with height h and width w with interfaces parallel to the y-z-plane, then we have to deal with Fourier series of the form

$$\tilde{\varphi}(y,z) = \sum_{m=1}^{\infty}\sum_{n=1}^{\infty} c_{m,n}\sin\left(\frac{m\pi}{w}y\right)\cdot\sin\left(\frac{n\pi}{h}z\right). \tag{6.21}$$

The expressions describing the iterative process in three dimensions are the same as those in two dimensions with

$$\frac{n\pi}{h} \text{ replaced by } \left(\left(\frac{m}{w}\right)^2 + \left(\frac{n}{h}\right)^2\right)^{\frac{1}{2}}\pi. \tag{6.22}$$

This applies to equation (6.11) and to the expressions for the eigenvalues in the multi-subdomain case. Therefore the convergence results for the 2-subdomain problem can be translated directly to three-dimensional problems: convergence occurs when *either* the length-to-height *or* the length-to-width ratio is large. The convergence is determined by the largest ratio.

This result is somewhat suprising: Consider a two-dimensional problem which does not converge due to small length-to-height ratios and a three-dimensional problem with the same length-to-height ratios. If the width of the three-dimensional problem is taken small enough, then it converges, although it resembles the two-dimensional problem then more than in the case of large width.

These convergence properties give the opportunity to handle large three-dimensional problems. Assuming that the length-to-height ratios are not too small, the convergence is not affected by a small length-to-width ratio (large width) . Still the subdomains in this kind of problems can be very large due to a large width. In this case the domain decomposition method can be applied again in the separate subdomains, which are then subdivided further in the y-direction, see Figure 6.4, page 70.

In this way a checkerboard of subdomains is created in which the actual solution of the Laplace problems is generated in the cells of this checkerboard.

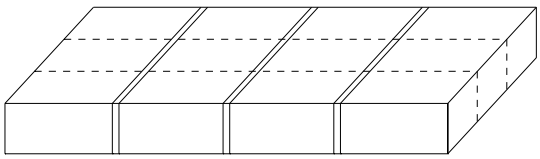

Figure 6.4 Two-dimensional subdivision of a 3-D domain. First the domain is divided by the interfaces indicated by double solid lines. Each subdomain is then divided further by interfaces indicated by the dashed lines.

The required number of iterations is at most the maximum number for the division in one direction times the maximum number for the other direction.

In this thesis we do not study the two-dimensional subdivision mentioned above any further. We only remark here that the iteration process which alternately iterates in x- and in y-direction is not efficient. There is no need to have a converged solution in the y-direction if the boundary conditions generated by the iteration in the x-direction are still far from the solution. A more general coupling in which no distinction is made between the two directions of the subdivision seems to be more efficient. (For example the D/D-N/N-scheme on all subdomains of the checkerboard simultaneously.) Convergence in such a two-dimensional subdivision has to be studied further.

6.2.4 Global coupling

An important result of the analysis in this section is that for a fixed length-to-height ratio of the subdomains which is not too small, the iteration process is convergent independent of the number of subdomains. This result seems to contradict the fact that in elliptic problems the domain of dependence is global. In the field of domain decomposition techniques, global dependence is an important issue. We discuss the aforementioned result by considering the following Laplace problem.

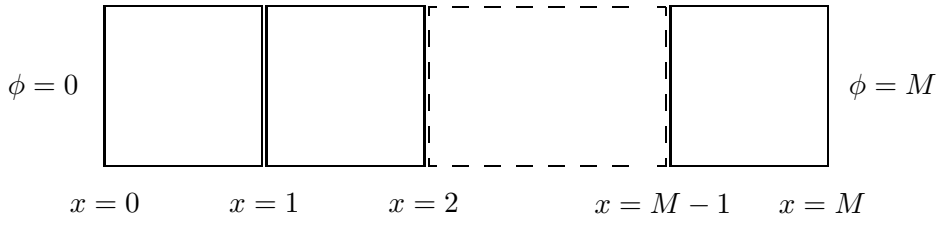

Figure 6.5 Domain subdivided in M subdomains.

If the problem has a homogeneous Neumann condition (i.e. $\frac{\partial \phi}{\partial n} = 0$) on the two horizontal boundaries then this problem is essentially one-dimensional

and its solution is given by $\phi = x$. Convergence is dependent on the initial boundary condition on the interfaces. For example with $\phi = 0$ on all interfaces the information on the right outer boundary has to travel all the way to the left thereby travelling only one interface per iteration step. If the problem is solved with less or with no subdomains at all, i.e. global coupling, then information is passed on much faster.

If the problem has Dirichlet conditions on the two horizontal boundaries corresponding to the same solution, i.e. $\phi = x$, then information about the solution is located near all interfaces and convergence is independent from the number of subdomains, under the assumption that length-to-height ratios are not too small. Global coupling is not needed in this case.

The convergence for problems with a fixed length-to-height ratio of the original domain is of course not independent of the number of subdomains M. Increasing M leads to smaller length-to-height ratios of the subdomains. Another way of considering the convergence process for subdomains with small length-to-height ratios is that for these problems the convergence on one interface is not independent of the boundary conditions on the nearest interfaces and thus of the convergence on these interfaces.

We do not address the problem of global coupling to solve the latter problem any further. We remark that global coupling is obtained by using coarser grids extending over several subdomains. We refer to Hackbusch [34] in which techniques as panel clustering and multigrid for boundary integral equation methods are described.

6.3 Results for domains with even bottoms

In the simulation of nonlinear water waves over an even bottom, the domain in which Laplace's equation has to be solved deviates from the rectangular form because of the disturbed free-surface boundary. Therefore, we introduce one complication to the rectangular domain, namely a localized disturbance of the free surface near an interface. We suppose that the length-to height ratios are large enough to have no noticable influence of the lengths of the subdomains on the solutions on the interface.

An analysis in which exact or approximate expressions for solutions are used to characterize the iterative process is very difficult now. It is however clear that the asymmetry near the interface leads to an asymmetry in the solutions on both sides of the interfaces. This can be explained by expressing the solutions in terms of Fourier modes again.

Consider a Laplace problem on a domain with a wave-like disturbance of the upper horizontal boundary and divided by a vertical interface, see Figure 6.6.

Homogeneous boundary conditions are imposed at the outer boundaries and

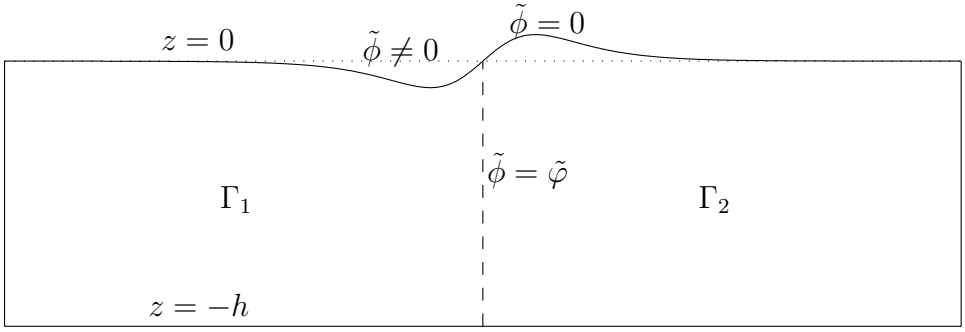

Figure 6.6 Computational domain with a local disturbance near the interface.

an inhomogenous one, say $\tilde{\varphi}_n = c_n \sin(\frac{n\pi}{h}z)$, at the interface. If we consider the rectangular subdomains with the undisturbed horizontal boundary $z = 0$ then the equivalent boundary value problem in these subdomains must have an inhomogeneous boundary condition, say $\tilde{\chi} = \sum_{m=1}^{\infty} d_m \sin(\frac{m\pi}{l}x)$, on $z = 0$. For this problem, the solution in both subdomains is not only expressed in terms of the Fourier mode $\sin(\frac{n\pi}{h}z)$ in the vertical direction, but also in terms of Fourier modes $\sin(\frac{m\pi}{l}x)$, $(m = 1, 2, \ldots)$ in the horizontal direction. Necessarily the derivatives $\frac{\partial\tilde{\phi}_1}{\partial x}$ and $\frac{\partial\tilde{\phi}_2}{\partial x}$ on the interface contain contributions of other Fourier modes than just $\sin(\frac{n\pi}{h}z)$. Therefore the Fourier modes $\sin(\frac{n\pi}{h}z)$, $(n = 1, 2, \ldots)$ can not be treated separately.

Furthermore, because the inhomogeneous boundary condition $\tilde{\chi}$ at $z = 0$ is given implicitly as a function of the boundary condition $\tilde{\varphi}$ at the interface, amplification factors can not be found as easily as in the analysis of Section 6.2. The previous consideration shows that because of the asymmetry of the domain the subdomain problems can be expected to have solutions which are differently. The coefficients $\alpha_{1,n}$ and $\alpha_{2,n}$ which express the Fourier modes of $\frac{\partial\tilde{\phi}_1}{\partial x}$ and $\frac{\partial\tilde{\phi}_2}{\partial x}$ respectively, in terms of the Fourier modes of φ are not equal and the use of the weighting factor $\omega = \frac{1}{2}$ does not annihilate the error made in the initial boundary condition. In this case a different value of ω, which takes the asymmetry in the solutions into account, may lead to a larger reduction of the error.

Because of the difficulty of describing the convergence analytically, we have performed some numerical experiments with the panel method to get an indication of the sensitivity of the iterative process for this asymmetry. These experiments are described next. The implementation of the domain decomposition method into the complete numerical method will be described in Chapter 7.

6.3.1 Two subdomains

In Section 6.2 the amplification factor ϵ_n is defined as the ratio of the Fourier coefficients of two consecutive Fourier modes in the iterative process, that is $c_n^{(k+2)} = \epsilon_n c_n^{(k)}$, see equation (6.7). In the numerical experiments to follow we do not consider single Fourier modes and we define the amplification factors $\epsilon_D^{(k)}$ and $\epsilon_N^{(k)}$ as the amplification of the jump over the interface in ϕ and $\frac{\partial \phi}{\partial n}$ respectively going from in iteration step k to iteration step $k+2$. In the numerical method this is translated into

$$\epsilon_D^{(k)} = \frac{\max_i |\phi_1^{(k+2)}(\mathbf{x}_i) - \phi_2^{(k+2)}(\mathbf{x}_i)|}{\max_i |\phi_1^{(k)}(\mathbf{x}_i) - \phi_2^{(k)}(\mathbf{x}_i)|}, \ \mathbf{x}_i \in \Gamma, \ k \geq 0, \ k \text{ even} \tag{6.23}$$

and similarly for $\epsilon_N^{(k)}$ with ϕ replaced by $\frac{\partial \phi}{\partial n}$ and $k \geq 1$, k odd. The subindices 1 and 2 of ϕ refer to the subdomains. These amplification factors indicate the convergence of the iteration process. For rectangular domains, $\epsilon_D^{(k)}$ and $\epsilon_N^{(k)}$ are equivalent to the amplification factors defined in Section 6.2.1 for single Fourier modes.

The iterative process in the numerical experiments is stopped if the jump over the interface in either ϕ (which is $\max_i |\phi_1^{(k)}(\mathbf{x}_i) - \phi_2^{(k)}(\mathbf{x}_i)|$), or $\frac{\partial \phi}{\partial n}$ (similarly with ϕ replaced by $\frac{\partial \phi}{\partial n}$) is below a certain stop criterion.

In the following we look at Laplace problems originating from one specific nonlinear wave problem and investigate the relationship between the angle between free surface and interface and an average of the amplification factors for this geometry.

We consider the domain given in Figure 6.7. The free-surface shape corre-

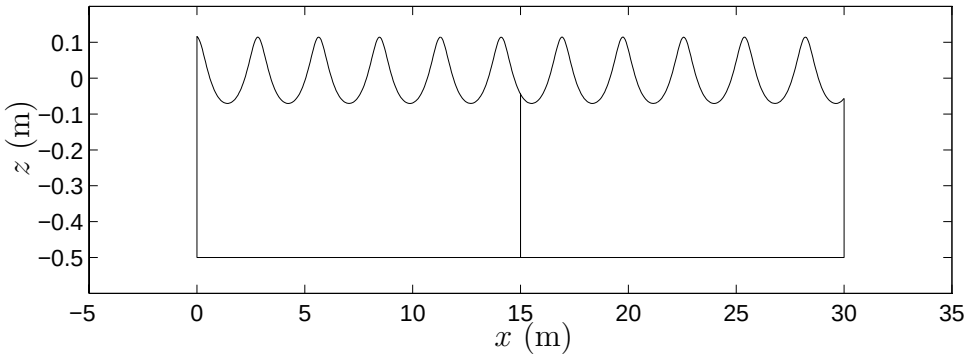

Figure 6.7 Domain with a disturbance near the interface.

sponds to a waveprofile of a nonlinear propagating periodic wave and is calculated according to the method of Rienecker and Fenton [57]. The wave conditions for this wave are: depth $h = 0.50$ m, wave period $T = 1.44$ s, wave height

$H = 0.185$ m and wave length $\lambda = 2.82$ m. The wave is mildly nonlinear; its height is equal to 60 % of the maximum wave height.

In the numerical experiments the interface has been placed at different locations compared with the positions of the wavetops in order to obtain different values of the slope of the free surface at the interface. For these wave conditions the maximum slope is obtained when the wavetop is shifted approximately $\frac{1}{8}\lambda$ from the position of the interface. The starting boundary condition $\varphi^{(0)}$ for the iteration process is taken uniform and equal to 2.0 which is much larger than the absolute value of the solution ϕ at the interface position in all computations. Therefore the error in $\varphi^{(0)}$, i.e. $\tilde{\varphi}^{(0)}$ contains contributions of all Fourier modes $\sin(n\pi/h)$, so far as they can be represented on the grid.

Two different grid distributions have been taken on the interface, both with 5 collocation points. An equidistant grid (I) and a non-equidistant grid (II) clustering near the free surface. In Figure 6.8 the results of the computations are plotted. The average amplification factor $\bar{\epsilon}$ is defined as $(\epsilon_D^{(1)} + \epsilon_N^{(2)})/2$.

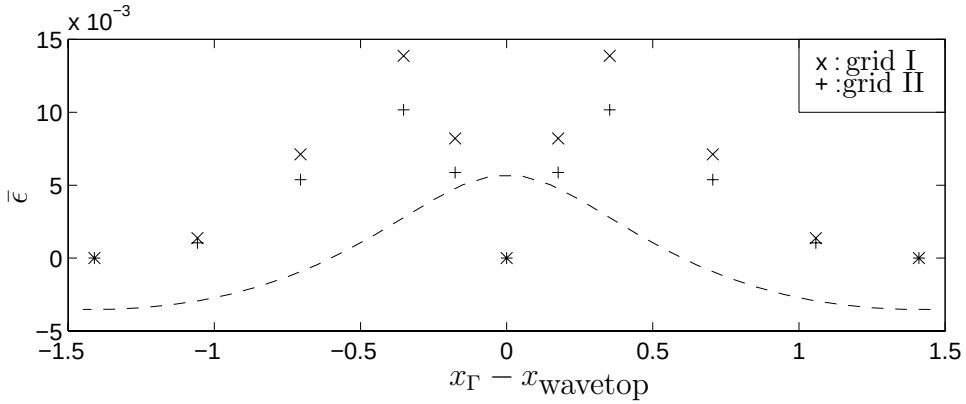

Figure 6.8 Average amplification factor as a function of the position of the interface. The dashed line indicates the elevation of the upper boundary near the interface at the specified position.

The following observations are made:

- The amplification factor $\bar{\epsilon}$ increases when the slope α of the free surface at the interface increases. In fact the amplification factor $\bar{\epsilon}$ is approximately quadratic in α.

- The amplification factor $\bar{\epsilon}$ is smaller for the non-equidistant grid. Examination of both subdomain solutions indicates that it is determined by the jump in the computed unknown (either ϕ or $\frac{\partial \phi}{\partial n}$) in the upper collocation point.

- The amplification factor $\bar{\epsilon}$ and the individual amplification factors $\epsilon_D^{(k)}$

and $\epsilon_N^{(k)}$ are very small. Only a few iterations are needed to achieve the discretization accuracy.

We conclude that the iterative process in the numerical method converges well for this geometry. Its convergence rate depends on the discretization. The optimal convergence rate is achieved with the most accurate solution near the local asymmetry of the interface. For a given domain corresponding to a nonlinear wave problem, the best choice of a vertical interface with respect to convergence rate is, as can be expected, perpendicular to the free surface.

The influence of the weighting factor ω has been studied with the same configuration. It was seen that convergence can be improved if the solution obtained in the subdomain in which the upper boundary makes a sharp angle with the interface (the left subdomain in Figure 6.6) is given a larger weight than the solution obtained in the subdomain on the other side of the interface. The largest amplification factor shown in Figure 6.8 was reduced with a factor of order 7 with the use of $\omega = 0.45$. The relation between the optimal weighting factor ω and the slope of the upper boundary has not been studied further. The use of a more appropriate weighting factor may especially be profitable in cases where the iteration process does not converge for $\omega = \frac{1}{2}$.

Finally it is noted that a way to find a better weighting factor is to relate the solutions generated on either side of the interface to the boundary condition on the upper boundary which is fixed during the iteration process. For example the solutions $\phi_1^{(k+1)}$ and $\phi_2^{(k+1)}$ at the $(k+1)$-th iteration step should be averaged such that the new boundary condition $\varphi^{(k+2)}$ matches the boundary condition ϕ at the intersection $\mathbf{x}_{I,FS}$ of the interface with the upper boundary. That is, determine ω such that $\phi = \omega\phi_1 + (1-\omega)\phi_2$ at $\mathbf{x} = \mathbf{x}_{I,FS}$. Similarly the solutions $\frac{\partial\phi_1^{(k)}}{\partial x}$ and $\frac{\partial\phi_2^{(k)}}{\partial x}$ should be averaged such that the new boundary condition $\psi^{(k+1)}$ matches the horizontal derivative $\frac{\partial\phi}{\partial x}$ (which can be approximated by using the tangential derivative of ϕ) at the same point $\mathbf{x}_{I,FS}$.

Using such a procedure the notation introduced at the beginning of Section 6.2, i.e. $\omega_D^{(k)}$ and $\omega_N^{(k)}$ becomes appropriate again. Numerically these weighting factors can be found by extrapolating the values of the solutions in the interface collocation points to the intersection point $\mathbf{x}_{I,FS}$. Preliminary computations with this algorithm did not give appropriate values for ω in all configurations. This is probably due to the large gradients near $\mathbf{x}_{I,FS}$ which makes the extrapolation rather sensitive to the truncation error of the panel method. This aspect should be studied further if a more robust algorithm is to be obtained. It is furthermore remarked that also the use of derivatives tangential to the interface can be employed, as advocated by Tan [67].

6.3.2 Three or more subdomains

Similar to the discussion in Section 6.2.4 we make a distinction between problems which have fixed length-to-height ratios of the subdomains and problems with a fixed length-to-height ratio of the original domain and study the effect of increasing the number of subdomains on the convergence.

Convergence at each interface is influenced by the local asymmetry and is a function of local asymmetry only if the length-to-height ratios are not too small. In the latter case we saw that for rectangular subdomains of fixed length-to-height ratio and of equal size, the convergence rate is independent of the number of subdomains. To check if the same is true for a general wave problem with an even bottom we considered the geometry of a nonlinear periodic wave as in the previous section. For computations with 12, 20 and 100 subdomains of fixed size, it is seen that with the use of more subdomains, more iterations are required, but that the number of required iterations does not exceed a certain limit. This limit is determined by the iterative process on the interface with the slowest convergence which corresponds with the maximum slope for a given wave problem at the interface.

From this observation we conclude that indeed convergence at each interface is a function of local asymmetry only. In fact, in subdomains bounded by interfaces on which the iterative process has already converged, the iteration process can be stopped and there is no need to solve Laplace's equation there any more. This is especially profitable when doing a computation in which large parts of the domain are not disturbed by waves (yet), for example when starting from rest.

In the case of a domain of fixed size, increasing the number of subdomains decreases the length-to-height ratios of the individual subdomains. For length-to-height ratios smaller than a certain threshold, convergence on different interfaces is not independent any more. This is illustrated with a time-dependent computation in Chapter 7, Section 7.5.1.

6.3.3 Three-dimensional problems

In wave problems in three dimensions the slope of the free surface at the intersection with the interface varies along the interface in general. Also for regular waves not travelling in the direction in which the domain has been subdivided, i.e. normal to the interface, this slope varies. The question arises how the iterative process is influenced by the presence of this geometrical variation.

To get an indication of the influence on the convergence rate, we have performed some numerical experiments with regular waves in which the propagation direction of the waves was varied relative to the normal direction of the

interface. See Figure 6.9 for an example of the configuration used. The wave conditions are the same as for the model problem also used in Chapter 4 with wave height $H = 5.0$ m, water depth $h = 10.0$ m and wave length $\lambda = 60.0$ m.

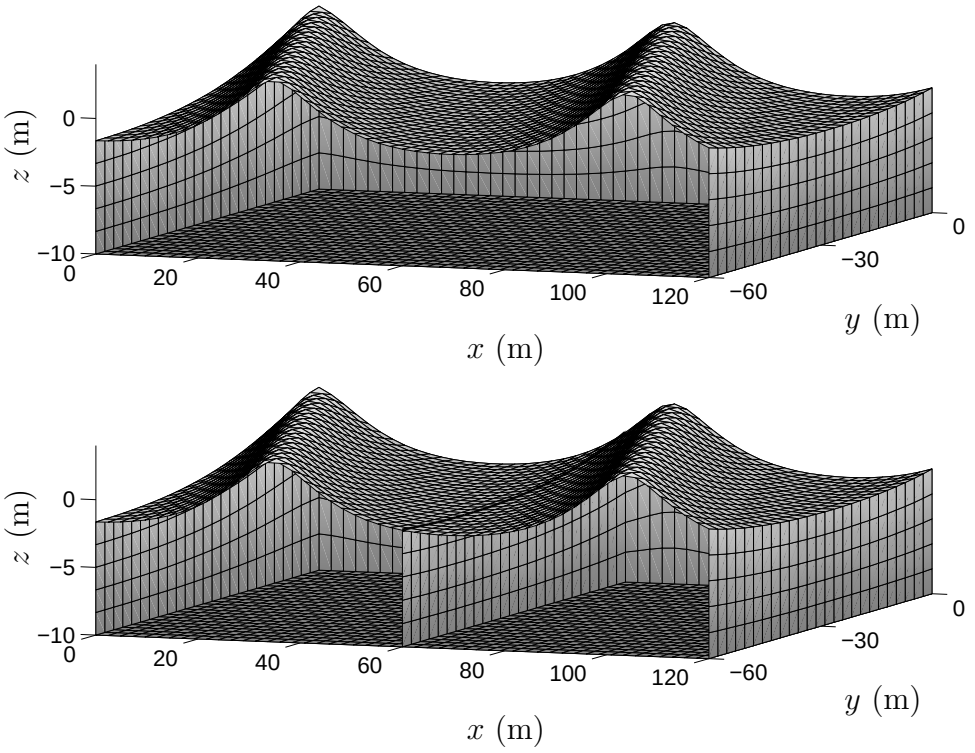

Figure 6.9 Domain of the configuration used in the numerical experiments (lower figure) and the corresponding undivided one (upper figure) for $\beta = 30^o$ and $x_0 = 60$ m.

Because of the limited width of 60 meter of the domain the variation of the slope along the interface is restricted to a certain range which depends on the location of the wave crest relative to the interface. This is indicated by the coordinate x_0 which corresponds to the x-coordinate of the wave crest in the plane $y = 0$ and by the coordinate $x_0' = x_0 + 60 \tan\beta$ which corresponds to the x-coordinate of the wave crest in the plane $y = -60$. In the numerical experiments these coordinates have been varied as well.

The results of the numerical experiments are given in Table 6.1. The average amplification factor $\bar{\epsilon}$ is again defined as $(\epsilon_D^{(1)} + \epsilon_N^{(1)})/2$. The columns with the headings $\eta_{\max}$, $\eta_{\min}$, $\frac{\partial \eta}{\partial x}_{\max}$ and $\frac{\partial \eta}{\partial x}_{\min}$ indicate the maximum and minimum values of the elevation and the slope in the positive x-direction on the intersection with the interface.

From these results we conclude the following:

Table 6.1 Average amplification factor $\bar{\epsilon}$ for a number of configurations indicated by the wave crest coordinate $(x_0, 0)$ and the angle β between the propagation direction of the wave and the normal vector on the interface.

x_0	x_0'	$\eta_{\max}$	$\eta_{\min}$	$\frac{\partial \eta}{\partial x}_{\max}$	$\frac{\partial \eta}{\partial x}_{\min}$	$\bar{\epsilon}$ $(\cdot 10^{-3})$
$\beta = 0^o$						
60.0	60.0	3.354	3.354	0.000	0.000	0.0
62.5	62.5	2.933	2.933	0.296	0.296	16.6
65.0	65.0	2.081	2.081	0.358	0.358	24.9
70.0	70.0	0.471	0.471	0.271	0.271	19.1
80.0	80.0	-1.238	1.238	0.090	0.090	2.46
90.0	90.0	-1.646	1.646	0.000	0.000	0.0
$\beta = 10^o$						
60.0	70.58	3.354	0.360	0.353	0.000	21.3
65.0	75.58	2.109	-0.666	0.353	0.157	17.7
70.0	80.58	0.513	-1.262	0.270	0.085	9.48
80.0	90.58	-1.210	-1.646	0.092	-0.001	1.20
90.0	100.58	-1.242	-1.646	0.003	-0.088	1.09
$\beta = 30^o$						
60.0	90.0	3.354	-1.646	0.310	0.000	9.38
70.0	100.0	0.854	-1.646	0.261	-0.064	5.79
80.0	110.0	-0.174	-1.646	0.110	-0.183	2.79
90.0	120.0	2.431	-1.646	0.026	-0.310	9.49
$\beta = 63.4^o$						
60.0	180.0	3.354	-1.646	0.160	-0.155	2.26
80.0	200.0	3.354	-1.646	0.135	-0.160	1.89
$\beta = 90.0^o$						
$-\infty$	∞	3.354	-1.646	0.000	0.000	0.0

- The configurations which are symmetrical in the interface, that is with $\beta = 0^o$ and $\beta = 90^0$, again show very fast convergence.

- For each angle β the rate of convergence decreases with an increase of the most extreme value $|\frac{\partial \eta}{\partial x}|_{\max}$ of the slope of the upper boundary in the direction of the normal to the interface.

- These results do not show a one-to-one relation between $|\frac{\partial \eta}{\partial x}|_{\max}$ and $\bar{\epsilon}$ which is valid for all configurations.

In comparison with the two-dimensional experiments, the corresponding results for $\beta = 0$ show faster convergence. At this point it is not clear whether in general three-dimensional computations in which the slope varies along the interface and reaches the maximum value $|\eta_x|_{\max}$, convergence is also better than in the two-dimensional one with slope $|\eta_x|_{\max}$ at the intersection of the upper boundary with the interface.

6.4 Results for domains with uneven bottoms

Finally we consider the convergence of the iterative procedure for geometries related to wave problems which include a bottom topography. The difference with the problems studied in the previous section, is that the geometry has an additional asymmetry near the intersection of the interface with the bottom. Because the bottom is a Neumann boundary in the formulation of the Laplace problem, the interface boundary conditions can not be represented with Fourier modes in the same way as in the previous sections, as is explained next.

Consider Figure 6.10 in which the (two-dimensional) geometries are illustrated that have been discussed so far. The geometry with the uneven bottom is illustrated with no wave disturbance at the free surface. For the problems

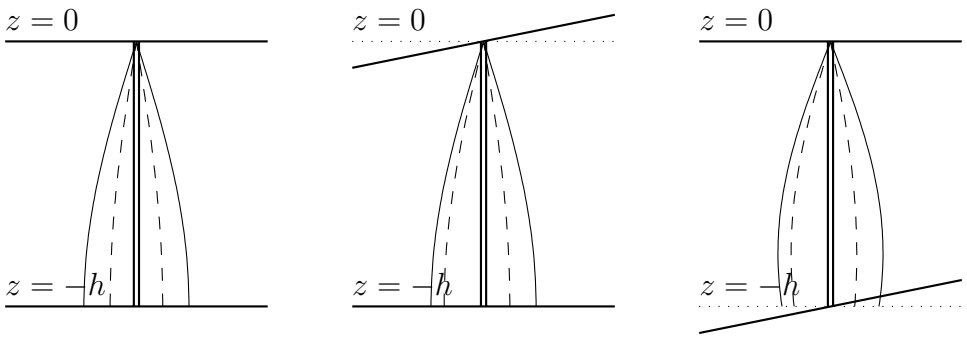

Figure 6.10 Illustration of the geometries discussed in Sections 6.2, 6.3 and 6.4 respectively. The solid curved lines in each figure represent the largest Fourier mode of the boundary condition $\tilde{\varphi}$ on the interface and the dashed lines indicate possible corresponding solutions $\frac{\partial \tilde{\phi}_i}{\partial x}$ in the respective subdomains.

with even bottoms, the largest Fourier mode of the boundary condition $\tilde{\varphi}$ on the interface $[-h, 0]$ is represented by $c_1 \sin(\frac{\pi}{2h}z)$. Note that due to the Neumann condition $\frac{\partial \tilde{\phi}}{\partial n} = 0$, the z-coordinate is scaled with $\frac{\pi}{2h}$ instead of $\frac{\pi}{h}$. The corresponding curves in Figure 6.10, left two figures, connect perpendicular to the bottom illustrating the property $\frac{\partial \tilde{\varphi}}{\partial z} = 0$ at $z = -h$.

For the problem with the uneven bottom, any boundary condition $\tilde{\varphi}$ imposed on the interface in order to solve the Laplace problems in both subdomains, can not satisfy $\frac{\partial \tilde{\varphi}}{\partial z} = 0$ at $z = -h$ because the normal direction does *not* coincide with the z-direction. It is therefore not possible to represent $\tilde{\varphi}$ in terms of Fourier modes $c_n \sin(\frac{n\pi}{2h}z)$. In Figure 6.10, right figure, this has been illustrated by curves that do not connect perpendicular to the bottom. Maybe a representation in terms of a sum of Fourier modes with a vertical scale different from $2h$ is possible, but it is not clear how the solution of the Laplace problems then

can be expressed in terms of these modes.

With numerical experiments it may still be possible to find a relation between slope of the bottom near the interface with the amplification factor of the iterative procedure. For a clear analysis such experiments should be carried out first for geometries without geometrical wave disturbances at the free surface (such as in linear wave problems). At present, however, it is not clear what the validity of the results would be, since it is unknown what the relevant vertical scale is and how the sequence of boundary conditions on the interface should be represented. We have not performed such experiments.

The effect of uneven bottoms on the iterative process is illustrated with some time-domain computations in the next chapter. Although not quantified, it is seen that the presence of the uneven bottom deteriorates the convergence of the iterative process.

6.5 Conclusions

We have examined the convergence behaviour of the D/D-N/N scheme for Laplace's equation in a number of geometrically simplified domains related to the simulation of water waves. Convergence is expressed in terms of an amplification factor which indicates the amplification of the jump in the potential made across an interface.

For rectangular domains it was found that convergence is obtained almost immediately if the length-to-height ratio of all subdomains is not too small. This implies that in order to have convergence, the number in which the domain can be divided is limited. Moreover, it is seen that in this case convergence is independent of the number of subdomains if the length-to-height ratio of the subdomains is fixed.

For domains in which the upper boundary is formed by a wave-like disturbance as in the simulation of nonlinear waves over an even bottom, it was found that the asymmetry of the upper boundary in the intersection with the interfaces decreases the rate of convergence. For geometries related to nonlinear wave problems, convergence is always obtained since the slope of the surface of (non-breaking) water waves is limited. In the case of more than two subdomains and for length-to-height ratios which are not too small, the number of required iterations and the related amplification factors are small.

Chapter 7

Application of domain decomposition to time-domain computations

7.1 Introduction

In the application of the domain decomposition method to the numerical method for nonlinear water waves, a number of aspects related to the discretization have to be considered.

On the smallest scale this involves choices about how to construct computational molecules near and on interfaces. On the largest scale this involves the splitting of the method into the solution of the time-independent problem (Laplace's equation) and the time-integration. One can either apply the domain decomposition method only to the time-independent part and treat the time-integration as before, *or* extend the domain decomposition also to the time-integration so that subdomains, apart from the coupling procedure, are treated as separate one-domain problems. These and other choices affect the performance of the method as a whole. Furthermore of course the efficiency of the application of the domain decomposition method itself is of importance. These aspects will be discussed in this chapter.

In Section 7.2 we discuss the implementation of the domain decomposition method into the numerical method. In Section 7.3 the effect of the domain decomposition method on accuracy and stability in time-domain simulations is studied. In Section 7.4 models are presented which can be used to predict required memory and required CPU-time for a given subdivision. Several aspects of efficiency are discussed here. These results are illustrated with examples in Section 7.5. In Section 7.6 the results are discussed in relation to more general computations and with respect to the usefulness of the domain decomposition technique. Finally, in Section 7.7 conclusions are given.

7.2 Implementation

So far we have investigated the domain decomposition method for Laplace problems for a fixed geometrical configuration. In the time-domain computation of propagating nonlinear water waves, the domain changes in time and Laplace's equation for ϕ has to be solved at every time level. The choice of the subdivision of the domain at every time level is of importance for the convergence rate. Furthermore the subdivision has some important implications for the numerical properties of the algorithm. Both aspects are discussed in this section.

7.2.1 Subdivisions

For the solution of Laplace's equation one is free to choose the position of the interfaces. With respect to convergence rates, the analysis of the previous chapter implies that for vertical interfaces the optimum choice is perpendicular to the free surface. In 3-D computations it is almost impossible to meet this condition. But also in 2-D computations this condition has some serious drawbacks. The choice of perpendicularly connecting interfaces necessarily leads to a reorganization of data from one subdomain to the other as the time-marching scheme proceeds. We clarify this in the following.

In an Eulerian description of the movement of the grid of the free surface, the collocation points only move in the vertical direction. To have an organization of data into subdomains which is fixed in time, the interfaces then have to be located at fixed horizontal positions. The propagating water waves will then "move through" the interfaces and the free surface will not be perpendicular to the interface but its slope will vary from a minimum value to a maximum value.

In a Lagrangian description the collocation points move along with the water particles. In this case no reorganization of data is required if (and only if) the interfaces drift along with the free-surface collocation points representing the water particles on the free surface. Still, the waves will propagate through the interfaces because the phase speed of the waves is larger than the velocity of the water particles. Also with the mixed Eulerian-Lagrangian method the phase speed is larger than the velocity of the free-surface collocation points.

In the present implementation of the domain decomposition method this choice (interfaces that drift along with the free-surface collocation points) has been made. It implies a loss of efficiency with respect to the convergence rate of the domain decomposition method compared with the optimum subdivision. On the other hand, this implementation in which data do not have to be reorganized is relatively easy and furthermore it has two important advantages with respect to parallel computing. First of all the computational load can be divided equally over all subdomains and secondly the transfer of data between different processors from one time-level to the other is minimal. The efficiency

of the domain decomposition method in combination with parallel computing will be studied further in Chapter 8.

7.2.2 Numerical aspects

Every subdivision has to satisfy the following demand on the numerical implementation. The outer boundaries such as the free surface and the bottom boundaries have to be split up into separate networks such that every network is assigned to only one subdomain. These networks are not allowed to extend beyond an interface because the computational molecules near the interface then would be divided over two different subdomains. The computation of influence coefficients in one subdomain would then depend on the geometry of the other subdomain and the subdomain problems would not be solved independently anymore. Necessarily one-sided discretizations are used on the free-surface and bottom networks near the interfaces in the numerical solution of Laplace's equation.

The actual implementation of the domain decomposition method in the time-domain method is based on the use of the original implementation described in Chapter 3, for all subdomains separately. The interfaces are treated as inflow and outflow boundaries with respect to grid organization and grid motion and therefore move along with the free-surface collocation points as already discussed in Section 7.2.1. Each interface is represented by two networks, one for each connecting subdomain. It is ensured that the coordinates of corresponding collocation points and grid points on the two networks are identical. Furthermore every subdomain is built up from a separate set of networks and subdomains are separated by network edges, see Figure 7.1. This satisfies the demand mentioned above. Details and numerical properties of this implementation are described next.

Intersections with interfaces

For each interface, the intersections of the connecting networks with the two interface networks are determined for both interface networks separately first. The interface network assigned to the 'left' connecting subdomain is intersected with the connecting networks of the 'left' subdomain and the interface network assigned to the 'right' connecting subdomain is intersected with the connecting networks of the 'right' subdomain. These intersections are determined in the same way as the intersections with the networks of the inflow and outflow boundary. Because both interface networks have to represent the same set of panels, the answers for each intersection point are averaged to obtain unique coordinates for the panel vertices.

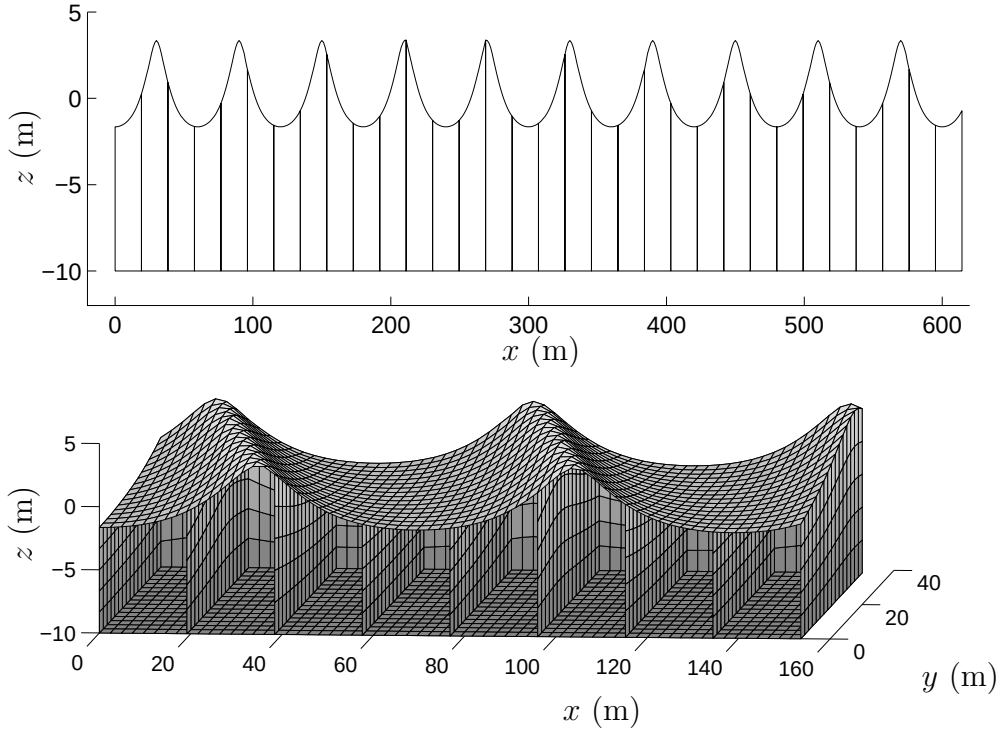

Figure 7.1 Subdivision of the domain into separate networks for the model problem in 2-D (upper plot) and in 3-D (lower plot). In this example every subdomain of the 2-D problem consists of 4 networks. Every subdomain of the 3-D problem consists of 6 networks. The networks in the front lateral boundary are not shown.

It is remarked that the intersection points can also be determined from the networks on the outer boundaries only. These networks connect smoothly and only an interpolation at the horizontal coordinates of the interface is needed. In both ways information from both sides of the interface is used.

Grid motion

The treatment of the interfaces with respect to grid motion is also the same as the treatment of the inflow and outflow boundaries. As already mentioned in Section 7.2.1, the interfaces drift along with the free-surface collocation points by using velocities extrapolated to the free surface. A difficulty associated with this approach is that at $t = 0$, the interface positions have to be chosen at a certain distance from network intersections connected with objects and changes in bottom geometry, see e.g. Figure 7.7. The bar consists of three bottom networks. This difficulty also applies to the outer lateral boundaries but as interfaces are usually chosen closer to the middle of the domain, special

attention has to be paid to this.

Actually, the grid motion of the interfaces is not necessarily required in the time marching scheme since the interfaces are only used for the solution of Laplace's equation. But then interfaces have to be constructed anew every time level based on the positions of the (moving) collocation points of the outer boundaries. This is not an easy task because it requires more control over these collocation points. In the present implementation the motion of the grid is controled on the subdomain level.

Related to this, is the discretization of the tangential derivatives needed in the time integration of the free-surface collocation points. Because of the network structure, one-sided discretizations are used for the collocation points near the interfaces. For these points two-sided information for the tangential derivatives is available from the collocation points at the other side of the interfaces, similar to the information required for the determination of the intersection points. The use of two-sided discretizations is not necessarily better in this case, because other computational molecules would be used for the time integration than for the solution of Laplace's equation. It is believed that consistency in discretization of the various parts of the numerical method is important for the stability of the method. Closer study of this aspect is recommended since, generally, two-sided discretizations are prefered over one-sided discretizations.

A last remark is made about the boundary conditions on the interfaces. In the present implementation the solution from the previous time level is saved and used as an initial guess for the boundary conditions on the interfaces. It may be useful to integrate ϕ on the interfaces with for example a Sommerfeld condition to obtain a better approximation on the next time level. The material derivative based on the grid velocity for the interface collocation points can be used.

Stop criterium

The stop criterium for the iterative process in the domain decomposition method is based on the jump in the boundary conditions over the interfaces. The iterative process is stopped if the differences of ϕ and $\frac{\partial \phi}{\partial n}$ over each interface are smaller than certain values δ_D and δ_N respectively. (Because at each iteration step either ϕ or $\frac{\partial \phi}{\partial n}$ is given the same value at both sides of the interfaces, only the jump in the variable to be solved needs inspection.) The values of δ_D and δ_N have to be chosen of the same order of accuracy as the local discretization error of the numerical method. If they are chosen too small, even the numerical solution of the *converged* boundary value problem will not satisfy the stop criterium. The most critical part of the numerical method with respect to the stop criterium is the accuracy of the solver of the system of linear equations. It was found that in all computations considered in this thesis a residual of order 10^{-7} for the solution of the system of linear equations is sufficient when using a

maximum jump of 10^{-7} in the boundary conditions over all interfaces. Better guide-lines should be formulated taking into account spatial resolution and the order of magnitude of ϕ and $\frac{\partial \phi}{\partial n}$.

Due to the treatment of the interface networks similar to the treatment of lateral networks in a one-domain problem, the actual coding of this implementation is rather easy. It resembles the original code with the following differences:

- Subroutines called for a problem on the subdomain level, such as the determination of all influence coefficients of a subdomain or the solution of the system of linear equation, are called in a loop over all subdomains.
- Extra information is needed to determine how subdomains connect.
- Synchronization points are introduced for the updating of the boundary conditions in the domain decomposition process and to make sure that geometrical data of the two networks of every interface are identical.

The coding of the program will be described in more detail in Chapter 8 in relation with implementations for parallel systems.

7.3 Accuracy and stability

As was already shown by Romate [58], the accuracy of the panel method is lower near the network edges. By introducing subdomains, new network edges are introduced and the question is how they affect local as well as global accuracy. Another property which is sensible to the use of network edges is the stability of the method. Broeze has paid attention to stability properties of the numerical method. In Chapter 4 we have extended these studies by considering two-dimensional computations over large simulation times. Attention was paid to the intersection with and the motion of the lateral boundaries.

With the use of the domain decomposition method extra vertical boundaries are introduced and accuracy is decreased near the interfaces. Therefore the stability of the method is reconsidered here for computations with the use of domain decomposition. The stability of three-dimensional computations using domain decomposition is not studied in this section, but will be demonstrated with some examples in Section 7.5. As also mentioned in Chapter 4, similar computations in three dimensions at present require too much CPU-time to be analyzed easily.

For examination of the stability of the two-dimensional method using domain decomposition we use the same configuration as in Section 4.5 and again

examine the simulation time that can be reached. The results therefore correspond to the ones tabulated in Table 4.1. In the computations presented here the classical fourth-order Runge-Kutta method is used.

In Table 7.1 results on the following configurations are tabulated. The rows labelled by '1 SD' and '2 SD' contain the results for one and two subdomains respectively. The row labelled by '2 SD, FG' contains the results of the two-subdomain computation with a resolution on the free surface twice as high. Finally in row '1 SD, 5 NW' the result for a configuration consisting of one (sub)domain and with the free surface consisting of two networks is given. With this configuration the effect of using one-sided discretizations in the absence of interfaces is examined.

Table 7.1 Stability of the method for configurations consisting of one and two subdomains

	$H = 5.0$ m	$H = 5.5$ m
1 SD	1074.60 s $\approx$ 164.1 T	638.80 s $\approx$ 98.8 T
2 SD	222.05 s $\approx$ 33.9 T	107.95 s $\approx$ 16.7 T
2 SD, FG	863.75 s $\approx$ 131.9 T	248.55 s $\approx$ 38.4 T
1 SD, 5 NW	41.00 s $\approx$ 6.26 T	40.95 s $\approx$ 6.33 T

From Table 7.1 it is clear that the stability of the computation is affected by using domain decomposition. Still the number of wave periods that can be simulated is reasonably large for this highly nonlinear wave. The break-down of the computation using two subdomains is caused by the irregular geometry which occurs as the error in computed free-surface elevation slowly grows in time. This growth is illustrated in Figure 7.2, page 88. Using a finer grid the computation is more accurate and much more stable.

Remarkably the computation in one domain using two networks at the free surface is much less stable. Closer observation shows that the Laplace problem with domain decomposition is solved more accurately than the Laplace problem using two networks at the free surface. Therefore the latter computation is less stable as is argued next.

The conclusiveness of the results of Table 7.1 with respect to accuracy and stability for more general computations is rather small because the results only involves two-dimensional computations with only two subdomains. It should be noted though, that the investigated model problem is highly nonlinear which puts high demands on robustness. Still the number of wave periods over which the computation can be continued accurately is reasonably large. The meaning of this result for more general computations strongly depends on the cause of the reduction of accuracy and stability when using subdomains. The following

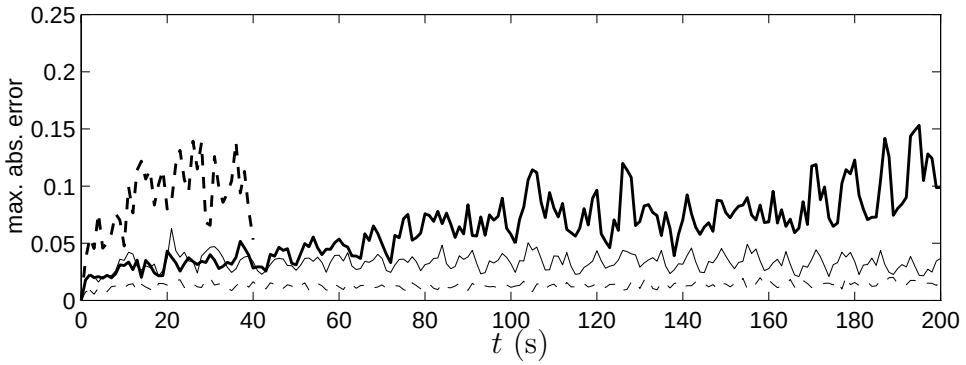

Figure 7.2 Maximum absolute error in computed free-surface elevation for $H = 5.0$ m during the time interval $[0, 200]$ for the computations 1 SD (thin line), 2 SD (thick line), 2 SD, FG (thin dashed line) and 1 SD, 5 NW (thick dashed line).

hypothesis is put forward.

The use of subdomains introduces relatively large truncation errors near the interfaces in both the solution of the Laplace problem and the time-marching scheme due to the one-sided discretizations. Also an error due to the (finite) stop criterium of the iterative process is introduced. This error is negligible compared with the truncation error for the problem discussed here. The truncation errors are generated at every time level, propagate through the domain and are only partially absorbed at the lateral boundaries. The maximum absolute error therefore slowly grows in time until the free surface is disturbed to a level where no stable computation is possible.

When using many subdomains, every interface is a source for the generation of errors. The maximum absolute error in a domain of fixed length can therefore be expected to be approximately proportional to the number of subdomains. If this is the case, then there is a maximum to the number of subdomains which can be used if one has to obtain a certain specified accuracy. This number also depends on the length of the simulation. An example of the dependence between generated error and number of subdomains will be shown in Chapter 10.

The hypothesis that errors are mainly generated locally near the interfaces can be investigated by increasing the resolution locally around the interfaces, so that the truncation error is equally large over the total domain. A problem with the present implementation is that due to the Lagrangian movement of the free-surface collocation point, this higher resolution is smeared out in the time-domain through the use of the grid correction algorithm. A mixed Eulerian-Lagrangian description is needed then to control the grid near the interfaces. An indication that locally increasing the resolution stabilizes computations with domain decomposition can be deduced from the computation

with an overall finer grid.

In three-dimensional computations an additional difficulty lies in the movement of the interfaces. In regions where the velocity varies in the horizontal directions, the free-surface grid distorts. Examples are shown in Figures 7.6 and 7.10. The interfaces, which are connected with the free-surface grid, can be chosen to move with a uniform velocity over the total interface or with velocities uniform over each vertical row of panels. In the former case the proper connection with the free-surface grid is lost. In the latter case the interface itself distorts as well, which influences the iterative process. Examples will be shown in Section 7.5.

The grid on the outer boundaries distorts in the same way as the grid on the interfaces. But since the largest variations normally occur in the middle of the domain, the distortion will be larger on the interfaces.

7.4 Efficiency

The efficiency of domain decomposition depends on many factors and is therefore not easy to predict in advance. From one point of view the relation between the number of subdomains and computational costs is rather simple and that is when all subdomains have a fixed constant length and a fixed number of panels per subdomain. The length of the total domain is proportional to the number of subdomains then, so that also the required memory is proportional to the length of the domain. If the interfaces are not too close together, the iterative processes on the various interfaces are independent. The number of required iterations then increases with increasing number of subdomains but converges to a fixed value. This implies that the computational effort for the solution of Laplace's equation per subdomain has an upper limit independent from the number of subdomains used.

The computational effort per subdomain for other algorithms, such as the time integration, is independent from the convergence of the iterative process. So the total required CPU-time per time step per subdomain also has an upper limit independent from the number of subdomains used. Therefore the required CPU-time per time step is proportional to the number of subdomains and computational costs of the numerical method for nonlinear water waves increase only linearly with an increase of the length of the domain. Note that on larger domains a computation is continued over larger time intervals generally.

From a more practical point of view, domain decomposition is applied to a domain of given size and the question is what the relation is between computational costs and number of subdomains. If this is clear then an optimum subdivision can be chosen, with respect to some relevant criteria. In this section

we follow this approach.

As already mentioned in Chapter 5 several aspects are of importance for the efficiency of the domain decomposition method. On the one hand extra panels are required and Laplace's equation has to be solved more than once. On the other hand the computational costs of some important parts of the method depend superlinearly on the number of panels per subdomain, which favours the use of small subdomains. Computational costs in terms of required CPU-time and required memory are described by appropriate models.

The model described here involves only those parts of the method for which simple dependencies of the computational costs on the number of panels can be determined. For example for the solution of the system of linear equations with Gaussian elimination there is a cubic dependence between the number of floating point operations and the number of unknowns. With the use of iterative solvers the required number of floating-point operations strongly depends on the problem, the solver, stop criteria, etc., see Romate [61]. Thus iterative solvers are therefore not considered in the model. Also other parts of the method which have been optimized with respect to efficiency but for which the dependence is very hard to determine are not considered in the model. The effect of these optimizations and of the use of iterative solvers is described qualitatively in Section 7.4.3.

Furthermore do CPU-times of course depend on the type of computing system and the performance of the system. With the models presented here it is still possible to obtain some insight into the efficiency of the domain decomposition method.

7.4.1 Model for required memory

The required memory is mainly determined by two-dimensional arrays containing the influence coefficients. In our numerical method we use three such arrays; one for the dipole coefficients, one for the source coefficients and one to store the system matrix $\mathcal{A}$. The size of these arrays for subdomain m is equal to N_m^2 in which N_m is the number of panels in subdomain m. Also $K = 225$ one-dimensional arrays with size N_m are needed per subdomain in the three-dimensional method to store other variables and workspace. For the two-dimensional method this number can be reduced to a value $K = 117$. Besides the arrays of size N_m there are a number of variables and small arrays which only form a small contribution and which are not taken into account. The total number of required real variables in subdomain m is therefore estimated as $3N_m^2 + KN_m$.

In the following we start by considering a one-domain problem and we derive a formula for the required memory as a function of the number of subdomains assuming that the subdomains are of the same size. The geometry of the do-

main is similar to those of the model problems considered so far. We make the distinction between networks which are directed along the direction of the subdivision (free-surface and bottom networks in 2-D and additionally some lateral boundaries in 3-D) and networks which are perpendicular to the direction of the subdivision (inflow and outflow networks in 2-D respectively remaining lateral boundaries in 3-D and of course interface networks).

Let N_s be the total number of panels on the networks along the direction of the subdivision and N_n the total number of panels in the other direction for the one-domain problem. Let M be the number of subdomains. Under the assumption that N_s can be divided by M we find that the number of panels per subdomain $N_{SD} = N_m,\ m = 1 \ldots M$, equals

$$N_{SD} = N_s/M + N_n. \tag{7.1}$$

The number of required real variables in the arrays mentioned above per subdomain then equals $3N_{SD}^2 + KN_{SD}$ and the total number of real variables equals

$$M(3N_{SD}^2 + KN_{SD}) = 3N_s^2/M + (6N_n + K)N_s + (3N_n + K)N_nM. \tag{7.2}$$

Because in general $N_s \gg N_n$, the number of subdomains giving a minimum number of required real variables is rather large.

7.4.2 Model for required CPU-time

We model the required CPU-time for the three most time-consuming parts of the numerical method.

1. Computation of influence coefficients.

   The number of influence coefficients is quadratic in the number of panels. We assume that the computation of each coefficient requires the same amount of CPU time. The total required CPU-time per subdomain then equals $\alpha_I N_{SD}^2$ for some coefficient α_I.

2. Solution of the matrix problem.

   The required CPU-time to solve a system of linear equations strongly depends on the solver used. Here we model the use of Gaussian elimination which requires an amount of CPU-time that is cubic in the number of panels. Therefore it equals $\alpha_S N_{SD}^3$ for some coefficient α_S.

3. Parts of the method for which the number of floating-point operations depend linearly on the number of panels.

   Most parts of the method depend only linearly on the number of panels such as the determination of geometric quantities and the time integration. The required CPU-time per time step of these parts can be gathered in

one expression. Again we assume that the computation of such a part of the method requires the same amount of CPU-time for each panel. The total required CPU-time per subdomain for these parts then equals $\alpha_L N_{SD}$ for some coefficient α_L.

The coefficients α_I, α_S and α_L depend on the hardware that is used and have to be estimated from measurements. The coefficients α_I and α_L are much larger for the three-dimensional method than for the two-dimensional method because more floating-point operations are required per panel.

The required CPU-time furthermore depends on the number of time levels p per time step and of course on the number of iterations k of the iterative process, added over all time levels. The total required CPU-time per time step then equals

$$\begin{aligned} M(p \cdot \alpha_I N_{SD}^2 + k \cdot \alpha_S N_{SD}^3 + \alpha_L N_{SD}) &= \frac{1}{M^2}(k\alpha_S N_s^3) \\ &+ \frac{1}{M}(p\alpha_I N_s^2 + 3k\alpha_S N_s^2 N_n) + (2p\alpha_I N_s N_n + 3k\alpha_S N_s N_n^2 + \alpha_L N_s) \\ &+ M(p\alpha_I N_n^2 + k\alpha_S N_n^3 + \alpha_L N_n). \end{aligned} \tag{7.3}$$

7.4.3 Remarks on efficiency

Some parts of the method need a more elaborate modelling than in the model described above. Because the required CPU-time for these parts strongly depends on the configuration used it is not very useful to try to extend the model for required CPU-time. But since it mostly involves parts which are more efficient than the ones described above a closer observation of the effect of using domain decomposition is needed. In the following only a qualitative description is given for these parts.

Number of iterations

As the studies in Chapter 6 show, the number of required iterations for the iterative process increases with the number of subdomains. The variable k in equation (7.3) therefore depends on the number of subdomains. It also depends on the boundary conditions which are used as initial guess for the iterative process. As initial guess the boundary conditions on the last time level can be taken or boundary conditions which have been integrated according to, for example, Sommerfelds condition.

Inverse matrix

Another direct way of solving which may be more efficient than Gaussian elemination is the use of inverse matrices to solve the equations. For a single iteration

process we denote the subsequent matrix systems as $\mathcal{A}^{(l)}\mathbf{x}^{(l)} = \mathbf{b}^{(l)}$, $l = 1 \ldots k$ with k the number of required iterations. Because the influence coefficients only depend on geometry only two type of matrices occur in the iteration process of the D/D-N/N-scheme: a matrix $\mathcal{A}_D$ when on all interfaces Dirichlet conditions are imposed and a matrix $\mathcal{A}_N$ when on all interfaces Neumann conditions are imposed. The inverse matrices $\mathcal{A}_D^{-1}$ and $\mathcal{A}_N^{-1}$ can be used on all odd and even steps of the iteration process respectively and therefore need to be computed just once. The solution $\mathbf{x}^{(l)}$ can be found by just performing a simple matrix-vector multiplication. Similarly a LU-decomposition of the matrices $\mathcal{A}_D^{-1}$ and $\mathcal{A}_N^{-1}$ can be employed. Especially if many iterations are needed this may be profitable.

A drawback is that at least two extra two-dimensional arrays are needed to store the inverse matrices. But as we shall see, memory requirements when using domain decomposition are generally not so high so that the use of inverse matrices is a feasible option.

Iterative solvers

For the solution of a system of linear equations many iterative solvers can be used. Their efficiency is generally much higher than the efficiency of Gaussian elimination, especially for large problems. For the systems of linear equations occurring in water wave problems and solved with the panel method using the Conjugate Gradient Squared method by Sonneveld [66], we find that it becomes more efficient than Gaussian elimination for problems larger than about 400 panels. If the matrix system is badly conditioned, however, convergence is not always ensured and special techniques, such as multigrid, may be required. Domain decomposition methods also are able to improve the condition number. In our application of domain decomposition we actually avoid the problem of badly conditioned matrices for problems with large length-to-height ratios, by subdividing the domain and solving the matrix system to a large extent locally in the subdomains.

The efficiency of iterative solvers for the subdomain problems is influenced by the use of domain decomposition in several ways. When using an iterative solver for the system equation $\mathcal{A}^{(l)}\mathbf{x}^{(l)} = \mathbf{b}^{(l)}$, the number of iterations which is needed by the solver depends on the begin estimate for the solution $\mathbf{x}^{(l)}$. This begin estimate becomes better and better as the iterative process of the domain decomposition method proceeds, i.e. as l increases. Therefore the number of iterations required in the iterative solver decreases every step of the iterative process of the domain decomposition method. The average number of iterations over all steps of the iterative process is much smaller than the number of iterations at the first step. When using Gaussian elimination the required CPU-time for solving the system of linear equations remains constant during the iterative process.

There is another way in which efficiency can be gained by the use of iterative solvers. In the first few steps of the iterative process there is no need to have a fully converged solution of the system equation before one moves to the next step of the iterative process. In this way a considerable number of iterations of the iterative solver may be saved although the iterative process itself may take more iterations. Such an approach has been studied by Tan [67] where it is referred to as 'inexact domain decomposition'.

Computation influence coefficients

The computation of the influence coefficients itself is a relatively expensive operation because it involves the integration of complex quantities over curved panels. If the field point $\mathbf{x}_i$ and the panel S_j are close together it is very important to have an accurate computation of the (nearly) singular integrals and analytic expressions are used for the influence coefficients. If $\mathbf{x}_i$ and S_j are further apart, these analytic expressions can be approximated sufficiently by Gaussian quadrature formulas which require much less CPU-time. The assumption that the computation of each influence coefficient requires the same amount of time is not valid anymore then. Broeze [13] found that the required CPU-time for computing influence coefficients, thereby also using Gaussian quadrature, is approximately proportional to $N_{SD}^{1.5}$. This of course depends on the criteria for the use of Gaussian quadrature and on matters as for example geometry and resolution.

Undisturbed subdomains

An important way of obtaining a higher efficiency, is the possibility to treat subdomains where no or only small wave disturbances are present, separately. This can be especially important in problems where the simulation is started from water in rest. There are several ways to reduce CPU-times of computations in such subdomains. We only mention two:

1. In subdomains where the geometry does not change or only changes slowly, it is possible to use so-called 'frozen coefficients'. With this term is meant that the influence coefficients are not re-evaluated over a certain time interval. The technique was used by Romate [58] by only evaluating the influence coefficients once per time step in the classical fourth-order Runge-Kutta method. Broeze [13] showed that this leads to a loss of volume and consequently to a loss of accuracy. However, if the geometry of a subdomain only changes slowly, frozen coefficients can be used over some specific time interval with a negligible effect on accuracy.

2. 'Freezing' information of course also applies to other parts of the algorithm but is especially profitable for the solution of the system of linear

equations. Profits are obtained automatically if an iterative solver is used: In slowly changing subdomains, the solution from the previously solved Laplace problem almost satisfies the present Laplace problem and very few iterations are needed. Direct solvers, however, require the same amount of CPU-time regardless of the quality of the begin estimate. A way of reducing CPU-time then, is to pause the computation of the solution of the system of linear equations during the iterative process, if the iterative process has converged on the adjacent interfaces of the subdomain and boundary conditions have not changed.

Hardware dependencies

In the model described above it is assumed that the required CPU-time for a certain algorithm is proportional to the number of floating-point operations required for this algorithm. In general CPU-times are influenced by other matters as for example the efficiency of adressing memory or the efficiency of vector processing which may again depend on the problem size. In terms of the model formulations given above this has to be accounted for by a dependence of the coefficients α_I, α_S and α_L, not only on the type of hardware but additionally on the size of N_{SD} itself. We have not done such detailed modelling here but one has to keep in mind the dependence of the efficiency of a computing system on the problem size.

7.5 Examples

In this section, four examples are given in which the aspects mentioned above are studied. The first two examples show the efficiency of the domain decomposition method, both in 2-D and in 3-D, for the model problem. Because we want to make a comparison with the model described in Section 7.4.2 we use Gaussian elimination and exact computation of the influence coefficients for these two problems. Additionally the effect of the use of an iterative solver on required CPU-time is considered.

The other two examples show the application of domain decomposition to problems for which experimental results are available, again one in 2-D and one in 3-D. These problems involve a specified bottom topography and therefore also complement Chapter 6 with respect to convergence of the iterative process in geometries with an uneven bottom.

7.5.1 Model problem in 2-D

The model problem which we consider here is the same one as discussed in Section 7.3. But as we are now interested in applying subdivisons with many

subdomains on the same problem, we take a domain with 256 panels on the free surface and 128 panels on the bottom boundary allowing divisions into 2, 4, 8, 16 and 32 subdomains of equal size. Taking the same resolution of 25 panels per wave length we obtain a domain of length 614.4 m = 10.24λ. See Figure 7.1, upper plot, which shows the configuration with 32 subdomains.

For the number of subdomains given above, computations are performed over 200 timesteps up to the time level t=10.0 s. The total number of iterations per time step (including those needed on the 3 intermediate time levels) for these computations is shown in Figure 7.3. The figure shows some remarkable

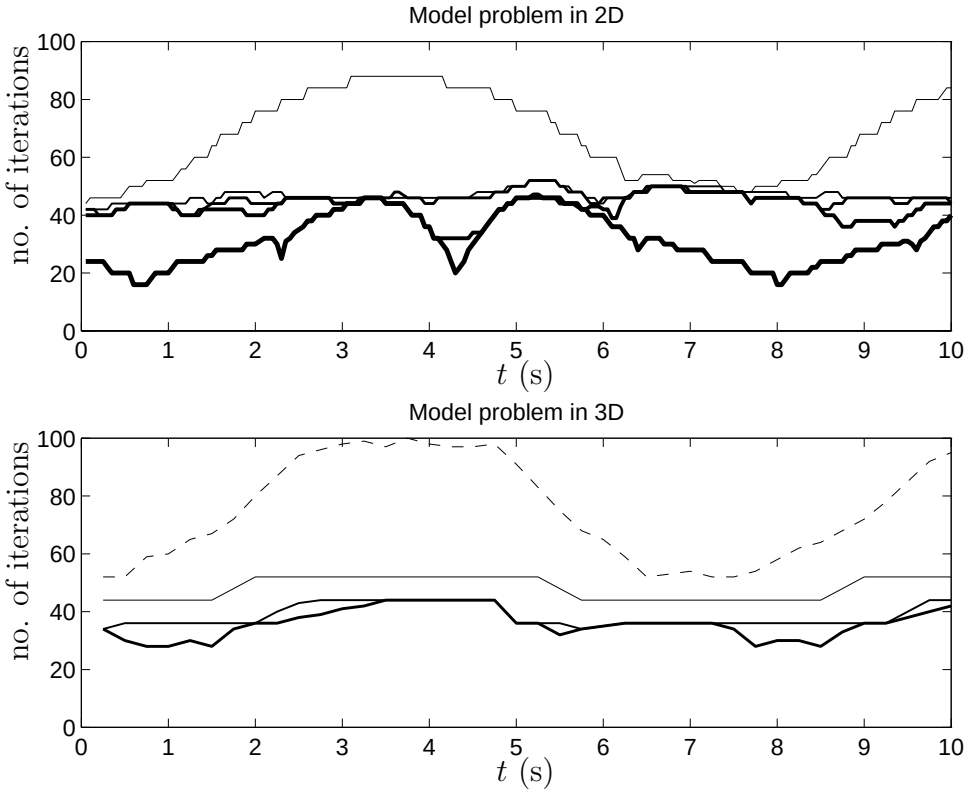

Figure 7.3 Number of iterations k required per timestep during the computation over the time interval $[0, 10]$ for the model problem in 2-D (upper plot) and the model problem in 3-D (lower plot) using different number of subdomains. The thickness of the lines decreases as more subdomains are used, that is 2, 4, 8, 16 and 32 subdomains in 2-D and 2, 4 and 8 subdomains in 3-D. The solid lines in the lower plot correspond to waveheight $H = 2.5$ m and the dashed line corresponds to the 8-subdomain computation with $H = 5.0$ m.

features. Firstly the required number of iterations k shows a variation in time which is seen most clearly for the problem using 2 subdomains. The variation shows a minimum around the time levels $t = 0.7$, $t = 4.3$ and $t = 8.0$ s. At

these time instances the interface is almost perpendicular to the free surface giving the fastest convergence. In between maximum values are reached when the free surface has a maximum slope at the intersection with the interface. The periodicity in the time levels at which these extremes are reached and in which the geometry of the domain is repeated again, is not equal to the wave period T because the domain drifts in the propagation direction of the waves. It therefore takes a time interval $\tilde{T} \approx 7.3$ s $> T$ to obtain the same geometry of the domain again. The variation in k is smaller for the subdivisions into more than two subdomains because in these cases more often there are interfaces with large slopes at their intersections with the free surface. The overall convergence is determined by the interface with the slowest local convergence.

Secondly the figure shows an increase of the required number of iterations at each time step with an increasing number of subdomains. This number converges to a level of around 46 iterations using 16 subdomains but then shows a large increase for the 32-subdomain problem. This is most probably due to the very small length-to-height ratios for this problem. At time instances at which the interfaces are evenly divided (for example at the initial time level $t = 0$) the overall convergence is comparable with the 16, 8 and 4 subdomain division. In between the interfaces group together because the free-surface collocation points cluster near the wave tops. This leads to very small length-to-height ratios - see Figure 7.4 - which shows the domain at time level $t = 10$ s. For these small length-to-height ratios, convergence on the interfaces is not independent anymore.

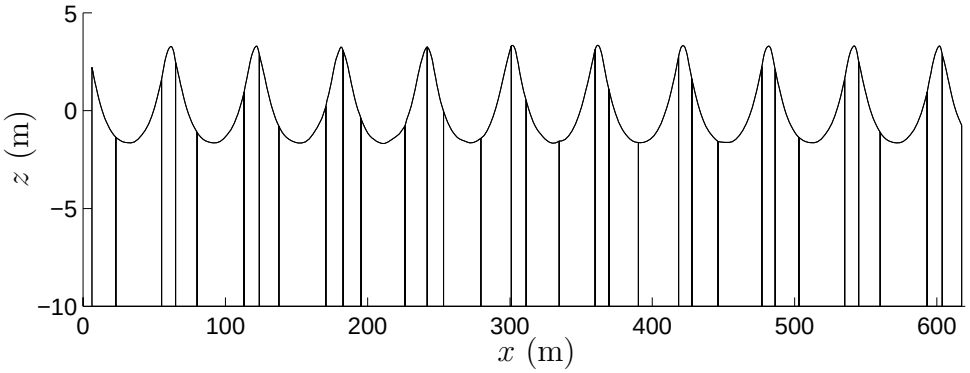

Figure 7.4 Domain of the model problem in 2-D at time level $t = 10$ s using 32 subdomains. The smallest subdomains have a length of around 10 m instead of the initial length of 19.2 m.

The required CPU-time for the computation as a function of the number of subdomains is shown in Figure 7.5. Measured CPU-times are compared with predictions from the model described in Section 7.4.2. The coefficients α_I, α_S and α_L are determined from measurements on required CPU times of the

corresponding parts of the method for the one-domain problem. The parameter k is taken equal to the average number of iterations over the time interval $[0, \tilde{T}]$. Because the performance of the computer system is strongly influenced by the number of subdomains, also CPU-times are indicated which have been corrected for the (average) performance of the system. These corrected CPU-times are determined such that they correspond to the measured performance of the computer system for the one-subdomain problem using the same solver.

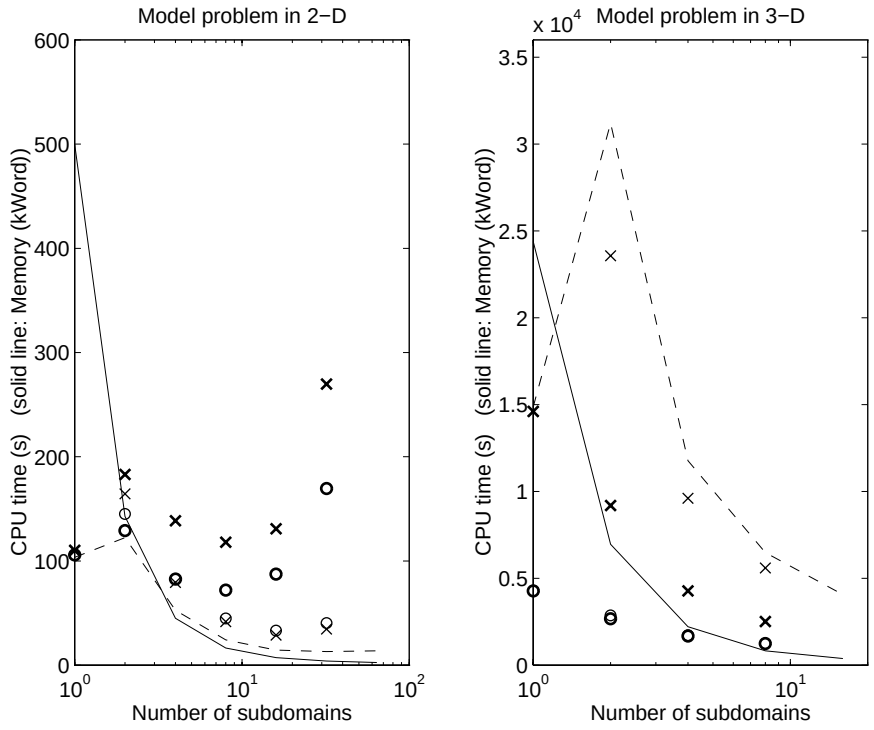

Figure 7.5 Measured CPU-times over the time interval $[0, \tilde{T}]$ using 1, 2, 4, 8, 16 and 32 subdomains in the model problem in 2-D and 1, 2, 4 and 8 subdomains in the model problem in 3-D. Crosses indicate computations using Gaussian elimination and circles indicate computations using CGS. Symbols which are printed thinly indicate CPU-times that are corrected for the performance of the computing system. The solid lines indicate the required memory and the dashed lines indicate the prediction for required CPU-time based on the model. In cases in which the thin symbols can not be observed, the thin and the thick symbols coincide.

The following observations are made:

- The CPU-times for the one-domain problem are almost the same when using Gaussian elimination and CGS. Closer observation of the measured CPU-times shows that both solvers take the same amount of CPU-time

and that the computation using CGS reaches a higher performance than the computation using Gaussian elimination for this number of panels.

- For the computations using Gaussian elimination, the CPU-times for the subdivisions are larger than the CPU-time for the original one-domain problem. This is partially due to the relatively poor performance of the computer system when using domain decomposition. If the results are corrected for the performance of the system, CPU-times are found which are smaller than for the one-domain problem, except for the two-domain problem. The corrected results are close to the predicted CPU-times.

  The relatively poor performance is related to the many memory references which have to be made in the two-dimensional arrays representing the source and dipole coefficients when using domain decomposition. In the present implementation these arrays are read every iteration step.

- For the computations using CGS, reductions in CPU-time are obtained when using 4, 8 and 16 subdomains. For all subdivisions, except the two-subdomain problem, the performance of the system is again smaller than for the one-domain problem.

- The computations using CGS require less CPU-time than the corresponding computations using Gaussian elimination with the same number of subdomains. The corrected results, however, only show small differences. Because the corrected results for the computations using CGS are related to a higher performance than the corrected results for the computations using Gaussian elimination, this may imply that less floating point operations are required for the computations using Gaussian elimination.

- The memory required for these computations is largely decreased when using domain decomposition.

This model problem is not representative. for more general wave problems. In a way it is a worst-case example with respect to required CPU-time because the domain is relatively small. In Chapter 10 a problem will be presented in which the domain is about one hundred wave lengths long. Also the computed wave is highly nonlinear which requires many iterations. Furthermore waves are present in all subdomains. In subdomains with only small disturbances less iterations of an iterative solver are required. Finally it is remarked that the resolution is moderately low. With a higher resolution reductions in required CPU-time when using subdomains are larger.

7.5.2 Model problem in 3-D

The model problem in 3-D that is treated next is similar to the model problem in 2-D described in Section 7.5.1. Except for the waveheight the same wave

conditions are used. Because the three-dimensional model requires much more CPU-time than the two-dimensional one, a domain with a smaller length has been chosen which is only one quarter of the length of the domain of the model problem in two dimensions, allowing up to 8 subdomains. The configuration used here furthermore differs on the following points.

- The same resolution of 25 panels per wave length has been chosen on the free-surface network in both directions. Because grid lines have to connect in the three-dimensional model the resolution on the connecting lateral boundaries and on the bottom is the same as on the free surface. The resolution in the z-direction has been reduced from 10 to 5 panels.

- The time over which is simulated is again taken equal to 10 s but the time step is increased to 0.25 s in order to reduce computational costs. On the lateral boundaries the analytical solution is imposed because it allows a larger timestep than Sommerfeld's radiation condition.

- The waveheight has been taken equal to 2.5 m instead of 5.0 m in order to reduce the total computational costs. For the investigation of the number of iterations the waveheight $H = 5.0$ m is only used for the 8-subdomain problem. The direction of propagation is under an angle of 30^o with the x-direction.

The number of required iterations is shown in the lower plot of Figure 7.3. It shows the same characteristics as the upper plot. For the 2- and 4-subdomain problem the number of required iterations is approximately equal to the corresponding numbers in 2-D although the minimum numbers for the 2-subdomain problem are smaller in 2-D.

The number of required iterations is significantly larger for the 8-subdomain problem with $H = 2.5$ m compared with the 2- and 4-subdomain problem. For $H = 5.0$ m almost the same number of iterations are needed as in the 2-D computation with 32 subdomains. This seems to contradict the observation made in Section 6.3.3 that convergence is better if waves propagate more oblique to the interface. But since the interfaces move along with the free-surface grid, the interfaces distort which influences the convergence negatively. See Figure 7.6 in which the domain at $t = 10.0$ s is shown.

Figure 7.5, right plot, shows required CPU-time and required memory together with predictions based on the models of Sections 7.4.1 and 7.4.2 for the present three-dimensional computations. The following observations are made:

- The CPU-time for the one-domain problem is much smaller when using CGS then when using Gaussian elimination due to the large number of panels (4090) for this three-dimensional problem even though the performance of the Gaussian elimination solver is much higher than the performance of CGS. This latter statement is also true for the number of panels used in the subdivisions although differences are smaller there.

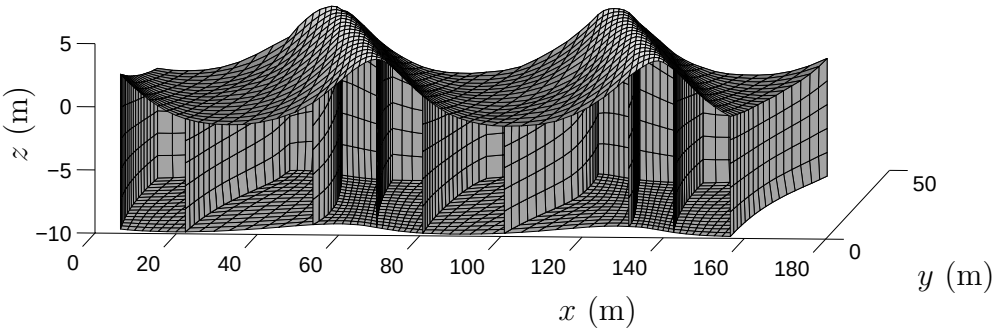

Figure 7.6 Domain of the model problem in 3-D at time level $t = 10$ s using 8 subdomains. Again the networks of the front lateral boundary are not shown.

- For the computations using Gaussian elimination, the CPU-times for the subdivisions are smaller than the CPU-time for the original one-domain problem. Contrary to the two-dimensional results the performance for these computations *increases*, because for the large number of panels used here, the performance of the Gaussian elimination solver is better than of the other operations. Using domain decomposition it is called many times. If the results are corrected for the performance of the system, CPU-times are found which are more in line with the prediction. Still less CPU-time is required except for the two-subdomain problem.

- For the computations using CGS, reductions in CPU-time are obtained in all cases up to a factor 3 when using 8 subdomains. The performance is hardly affected so that it can be concluded that the number of floating point operations is decreased.

- The memory required for these computations is largely decreased when using domain decomposition.

The remarks made in Section 7.5.1 on the implications of the results for this model problem for more general problems in two dimensions also apply to the results presented here.

7.5.3 Waves propagating over a bar in 2-D

As a first example of applying domain decomposition in a time-domain computation with an *uneven* bottom we consider the simulation represented in Figure 7.7 which includes an underwater bar. The point of interest with respect to wave propagation is the generation of higher frequency components of the incoming wave by the underwater bar. This configuration was also used by Broeze [13]

who compared his results with experimental results reported by Beji and Battjes in [5]. Here we first discuss the application of domain decomposition. Only a small remark about the comparison with experiments, repeated by Luth [49], is made thereafter. In the present study computations were done using 1, 2, 4

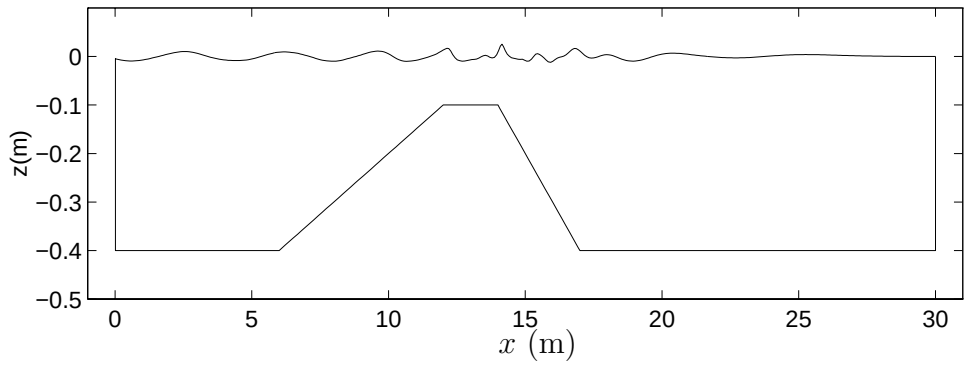

Figure 7.7 Domain for the simulation of waves over an underwater bar at $t = 16.0$ s.

and 8 subdomains. At the start of the computation interfaces are distributed evenly over the domain which implies that the 2 subdomain problem has an interface located at $x = 15.0$ m on the downstream slope (1:10) and the problems using more subdomains additionally have interfaces on the upstream slope (1:20). The drift in the simulated wave is small so that during the computation the interfaces do not drift far away from their initial position.

The 1-domain problem has a total of 593 panels. The subdivisions have approximately the same amount of panels on the outer boundaries and 8 panels for each interface network. The computations have been done using the CGS-solver and the algorithm using both analytical expressions and Gaussian quadrature to compute the influence coefficients. Therefore no comparison is made with the model for required CPU-time. The computations were also done on a Pentium/90 PC under DOS.

We first consider the number of required iterations. For the divisions into 2, 4 and 8 subdomains we have plotted the index of the time step against the number of required iterations per time step, see Figure 7.8.

It can be seen that as waves enter the domain, the number of iterations increases, which is due to the increased asymmetry near the interfaces. The increase in the case of 8 subdomains becomes so large that it exceeds the imposed maximum of 20 iterations for a single Laplace problem after 234 time steps. The variation in the case of 2 subdomains is larger than in the case of 4 subdomains.

Results on required CPU-time and memory are given in Table 7.2.

A considerable speed-up can be noticed when using a PC in the case of 4 subdomains. For the Cray-computer the speed-up is not so spectacular. Again this

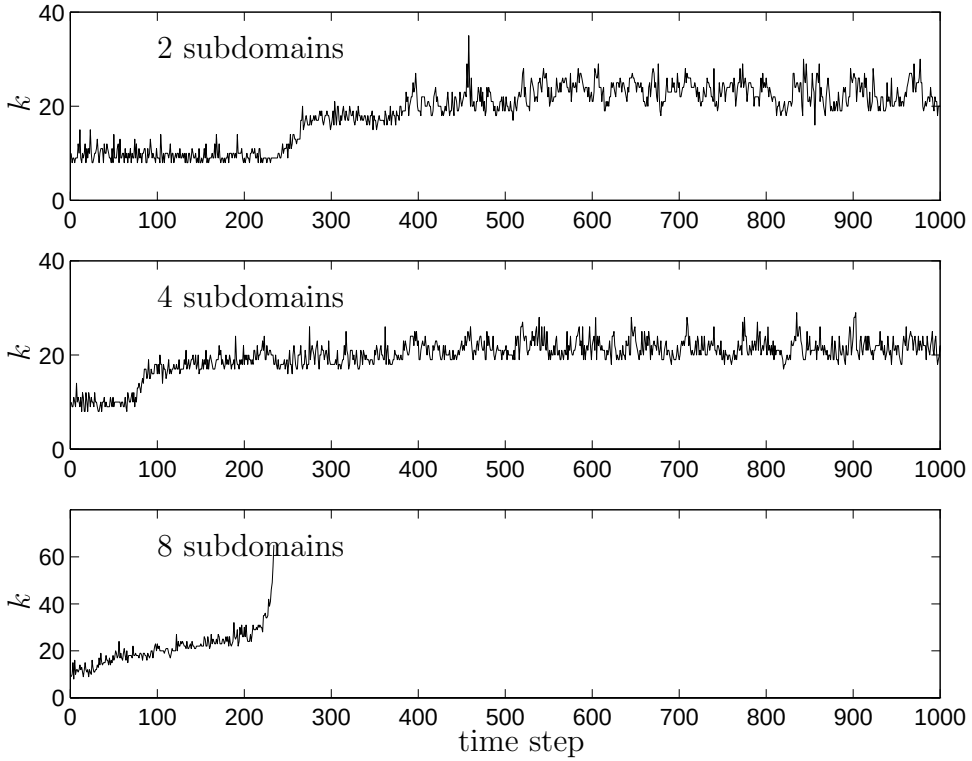

Figure 7.8 Number of required iterations k during the simulation using 2, 4 and 8 subdomains.

is related to the relatively slower performance of the Cray on memory references in two-dimensional arrays representing the source and dipole coefficients.

Finally some remarks can be made on the accuracy of the results. Firstly it is remarked that the computed wave heights at the positions were measurements were taken, up to $x = 23.0$ m, are hardly affected by the application of domain decomposition. The results of the 4-subdomain problem show only a small phase error compared with the results of the 1-domain problem. The corresponding time signals of both computations are therefore hardly discernible.

In the experiment by Beji and Battjes [5] the depth decreases from $x = 18.95$ m onwards with a 1:25 slope, whereas in his computations, Broeze [13] used a flat bottom there. Therefore no comparisons were made by Broeze for stations beyond $x = 17.3$ m. In the experiment by Luth [49] this part of the bottom is flat and comparisons can be made there as well. Results of these comparisons are reported by Dingemans [23], Section 5.9. It was found that the agreement between computation and experiment as shown by Broeze for the stations $x =$ 5.7, 12.5, 14.5 m and 17.3 m, is comparable with the agreement found in the present computations for the stations $x = 19.0$, 21.0 and 23.0 m. In comparison with a number of Boussinesq models and a Hamiltonian model it was found

Table 7.2 Computational requirements for different number of subdomains.

No. of subdomains	CPU-time on PC (h)	CPU-time on C98 (min)	Required memory (MWord)
1	21.51	22.00	1.729
2	19.51	20.16	1.289
4	7.84	13.60	1.088
8	–	–	1.049

that the present results are closer to the experiment.

7.5.4 Waves propagating over a shoal in 3-D

In the last example of this chapter the domain decomposition method is applied to a three-dimensional configuration already used by Broeze [13] (Section 11.2), with an uneven bottom. The configuration consists of a 1:50 slope on which a bank is situated, see Figure 7.9. Waves fall in under an angle of 20^o with the slope. The wave period is 1 s and the wave height near the wavemaker is 4.64 cm. The configuration originates from experiments performed at Delft

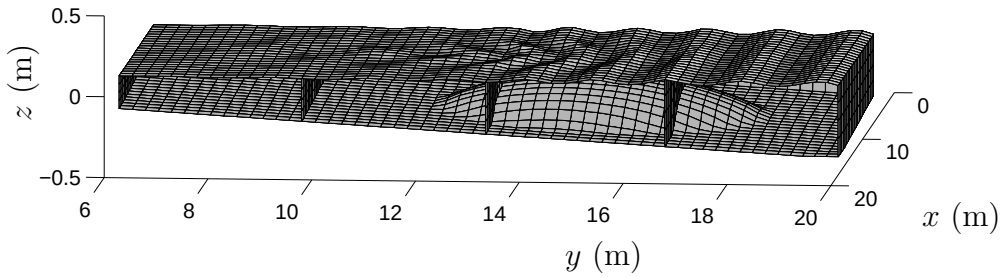

Figure 7.9 Domain of the shoal-problem at $t = 9.26$ s in the computation using 4 subdomains. The networks in the front lateral boundary are not shown. Waves enter the domain from the right.

Hydraulics described by Berkhoff et al. [7], see also Dingemans [23]. As waves propagate over the shoal, refraction effects emerge and waves focus behind the shoal.

In the computations presented here we only consider subdivisions into 4 subdomains in the y-direction (as shown above) and the x-direction. The relation between number of subdomains and convergence of the iterative process or

efficiency is not studied here. The computations are only discussed with respect to stability.

The computation with the subdivision in the y-direction and the interfaces perpendicular to the initial propagation direction of the waves breaks down after 9.26 s. At this time level the iterative process diverges on the interfaces located near $y = 13.1$ m and $y = 16.5$ m. The increase in jump in the computed value of ϕ over the interface per iteration step is largest on the interface behind the shoal (amplification factor $\epsilon_D \approx 1.28$).

It is not clear what causes the divergence of the Laplace problem at this time level. Most probably it is related to the distortion of the grid on the interfaces, especially on the interface near $y = 13.1$ m. Due to the large variation of wave height in the region behind the shoal, the variation in horizontal velocities of the free-surface grid is large which causes the grid to distort. As a consequence the interface located behind the shoal distorts as well.

The distortion of the grid on the interfaces is much smaller in the computation with the subdivision in the x-direction and the interfaces *parallel* to the initial propagation direction of the waves. This computation can be continued over the total time interval $[0, 16]$. The number of required iterations varies only slightly and does not exceed 8. The free-surface grid of this computation at $t = 16.0$ s is shown in Figure 7.10.

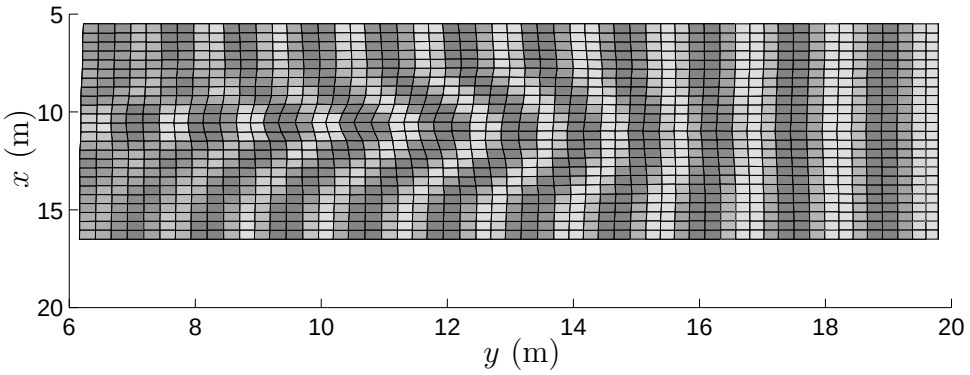

Figure 7.10 Free-surface grid of the shoal-problem at $t = 16.0$ s in the computation using a subdivision in the x-direction into 4 subdomains. Light coloring indicates positive elevation, dark coloring indicates negative elevation.

In conclusion it can be said that the iterative process of the domain decomposition method needs further study for cases with uneven bottoms. An appropriate approach to these problems is the use of an Eulerian description to avoid the distortion of the grid on the interfaces. For the case presented here it is nevertheless possible to successfully apply the domain decomposition method.

7.6 Discussion

So far in this chapter, we have studied different aspects of applying domain decomposition in the time-domain numerical method for nonlinear water waves. The example presented in Section 7.5.3, shows that domain decomposition can be succesfully applied because it reduces computational costs without affecting accuracy. A more general statement about the usefulness of the implemented domain decomposition technique is however required. Therefore we discuss the several aspects in their mutual connection and in relation to more practical computations in this section.

The usefulness of applying domain decomposition can be considered in several ways. One can either have the purpose of minimizing computational costs for a given water wave problem or have the purpose of enlarging the range of water wave problems that can be handled. Another purpose may be to be able to use a larger resolution for a given water wave problem. These different approaches are discussed in the following.

7.6.1 Minimizing computational costs

In all the examples presented so far computational costs were considered as a function of the number of subdomains for a given water wave problem. It was shown in these examples that an optimum subdivision exists for which the required CPU-time is minimal. Another example of minimizing computational costs for a given water wave problem is shown in Chapter 9. We formulate some criteria that can be followed consecutively on which to determine the optimum number of subdomains for a general water wave problem.

1. The relation between the number of subdomains and required memory can be determined fairly easily. It indicates the range of possible subdivisions for a given available memory capacity.

2. The relation with required CPU-time is much harder to determine. First based on the geometry of the domain an estimation has to be made on the number of required iterations per time step as a function of the number of subdomains. There is a minimum length-to-height ratio of the subdomains for which convergence on the interfaces is mutually independent. Subdivisions for which convergence on separate interfaces is not independent may lead to a diverging iterative process and should be avoided.

3. The minimum length-to-height ratio of the subdomains is related to the maximum drift of the interfaces and to the degree of asymmetry around the interfaces and thus to the wave conditions of the simulated wave field. For waves propagating over an even bottom only the maximum slope of the free surface is of importance. The example in Section 7.5.1 shows

that the minimal length-to-height ratio should be around 2 for the almost highest wave. The dependence of this minimum on other wave conditions and bottom geometry has to be investigated further.

4. Next the relation between number of subdomains and the mean value of the required number of iterations has to be determined. Again this is related to wave conditions and bottom geometry.

5. Based on an estimation of the required number of iterations and on an estimation of the parameters α_I, α_S and α_G predictions can be made on required CPU-time using the model in Section 7.4.2. This does not, however, give a prediction when using parts of the algorithm which may be more efficient than the modelled ones.

6. A number of parameters mentioned so far, required for this prediction are quite uncertain. Better estimates can be obtained by executing computations over only a few time steps.

7. Next it has to be determined which subdivisions do not lead to an unacceptable loss of accuracy. Loss of accuracy is related to the number of subdomains and the duration of the simulation.

8. These subdivisions may become acceptable after increasing the resolution. This may be an option if the loss of accuracy can be avoided by locally increasing the resolution. Predictions on required CPU-time then have to be corrected for the increase of the number of panels per subdomain.

From these criteria we see that many aspects of the efficiency of domain decomposition are not clear yet. Still it may be helpfull to determine an efficient subdivision based on the criteria mentioned above. Comparison with similar computations done before may further improve the predicted CPU-times.

7.6.2 Enlarging the domain

Another way of considering the usefulness of domain decomposition is from the viewpoint of enlarging the domain. This way of applying the domain decomposition technique has already been discussed in Section 7.4 consedering subdomains of fixed length-to-height ratio. It was argued that in this case the computational costs for many subdomains increase at most linearly as a function of the number of subdomains. determined for moderately small numbers

Enlarging the domain, by adding subdomains of fixed length-to-height ratio, we obtain the linear dependence between computational costs and number of collocation points which was sought for in the problem statement in Section 5.1. From a more general perspective the domain decomposition method can be viewed as a way to use the observation that in water wave problems with

domains with large length-to-height ratios, the information on one side of the domain is hardly related to the information on the other side of the domain. This observation applies to both the computation of the influence coefficients and the solution of the system of linear equations for the one-domain problem. With respect to the computation of the influence coefficients, the domain decomposition method automatically avoids the computation of those coefficients which can be neglected. With respect to the solution of the system of linear equations, the domain decomposition method implicitly rewrites the system of linear equations into a block-structured matrix with a much better condition number.

7.6.3 Increasing the resolution

If computations for a given water wave problem with a fixed subdivision are enhanced by increasing the resolution then convergence characteristics of the iterative process are hardly influenced. But as the computational costs of required memory and CPU-time per subdomain depend superlinearly on the number of panels *per subdomain*, the total of these computational costs will also increase superlinearly. Other techniques are needed then to efficiently solve the subdomain problems, especially the part in which the system of linear equations is solved.

7.6.4 Cutting up irregular geometries

An important objective of domain decomposition methods frequently stressed in literature, is to reduce the geometrical complexity of a domain by cutting it up in regular geometries. The subdomain problems may be easier to solve or specific numerical methods can be applied like spectral methods. This aspect has not been discussed in the present application so far because the domains which have been discussed up to now, only involve singly connected domains with relatively large length-to-height ratio. An interesting example of a configuration in which the domain is different to this respect, is a configuration with a ship near a quay. The motion of the water between ship and quay has great influence on the motion of the ship. This area, however, only forms a small part of the domain and has a small length-to-height ratio. Such areas require special treatment when applying domain decomposition.

7.7 Conclusions

In this chapter we have studied the application of the domain decomposition method for Laplace's equation to the numerical method for nonlinear water waves. Because we require that Laplace's equation be solved independently

in each subdomain, the networks are not allowed to belong to more than one subdomain in the time-independent part. By adopting the original one-domain implementation for each subdomain in both the time-independent and the time-dependent part, this constraint is met and the organization into networks is the same throughout the total numerical method.

As a consequence one-sided discretizations are used near the interfaces, which affects accuracy and stability. With the use of domain decomposition the maximum error in free-surface elevation slowly grows in time which is believed to be caused by the relatively large local truncation error near the interface. Numerical tests have been performed to examine this hypothesis. In three-dimensional problems an additional difficulty lies in the distortion of interfaces when the velocity has a large variation along the interfaces.

The efficiency of the domain decomposition method for a given problem is easily determined for the required memory but it is hard to predict required CPU-times. By considering some examples in both 2-D as in 3-D it was shown that large reductions in computational requirements can be achieved when using domain decomposition. Although clear guide-lines are not given it is expected that reductions can be substantial, especially for large problems.

With the domain decomposition method, the computational costs of the panel method and consequently of the numerical method for nonlinear water waves per time step, can be made to depend linearly on the number of collocation points by using subdomains of fixed size.

Chapter 8

Parallel computing

8.1 Introduction

The domain decomposition method as described in Chapter 7 introduces a subdivision of the computational work which is very suitable for parallel computing. The computation of the influence coefficients in each subdomain is independent of the computation of the influence coefficients in other subdomains. Also the solution of the systems of linear equations in each subdomain can be determined independently at each iteration step. These algorithms require almost all of the computational effort and therefore a high efficiency can be expected from the use of a parallel system.

The problem, however, with employing a system of parallel processors lies in those parts of the computational model which can not be done independently. This, in general, decreases the efficiency of the system. An obvious example here is the updating of the boundary conditions on the interface for which information from neighbouring subdomains is needed. Also some algorithms of the time-dependent part of the numerical method, such as the movement of the interfaces along with the free-surface collocation points, require data from neighbouring subdomains.

In this chapter we study some ways in which the numerical method can be parallelized and we show results of several parallel systems for some nonlinear wave problems. The aspects of parallelizability and efficiency are studied in relation to the number of subdomains. The parallel computation of the subdomain problems is a rather coarse grained parallelization in contrast to for example the parallel computation of the influence coefficients or the parallel solution of the system of linear equations for one subdomain. Such a parallelization may very well lead to a speed-up but since emphasis is on the use of domain decomposition we only consider the parallelization based on the subdomain division.

This chapter is divided as follows. In the continuation of this section first a small outline is given of the purpose of parallel computing in general and

some types of parallel systems. In Section 8.2 we present some parallel implementations of the numerical method. In Section 8.3 results are presented of computations on a SGI Power Challenge with 8 processors and a cluster of 5 HP workstations. Finally in Section 8.4 the results are summarized.

8.1.1 Purposes of parallel computing

In contrast to the domain decomposition method itself, parallel computing does not in general reduce the number of required floating point operations. Using a parallel system of processors the floating point operations are divided over the processors. For a given problem the number of floating point operations may even be increased to obtain an efficient use of the parallel system. But then what are the purposes of parallel computing?

The purpose of parallel computing is related to the aspect of computational costs and these again can be defined in several ways depending on the wishes of the user or the provider of the available computing system. We distinguish the following purposes:

- From the viewpoint of the user the purpose can be to have results as quickly as possible. This is generally indicated by the wall-clock time which is the duration of the computation from the beginning to the end. In our present study this boils down to finding the optimal number of subdomains and the optimal number of processors to have a minimum wall-clock time for a given water wave problem.

- From the viewpoint of the provider of a parallel system the purpose can be to have a minimum load of the system such that it is available to as many users as possible. This can be expressed, for example, in terms of the user time which is the time a processor is not available to other users. For our problems the purpose would then be to find the optimal number of subdomains and the optimal number of processors to achieve a minimum total user time over all processors.

- From the viewpoint of both the user and the provider the purpose can be to have a minimal memory load of the system.

- Other purposes can be expressed in minimizing a cost function which involves both wall-clock time and total user time and memory load. Also other costs can be incorporated as for example costs for hard disk storage.

The study in this chapter is mainly aimed at determining the user time overhead and finding the minimum wall-clock time for a given water wave problem.

8.1.2 Parallel systems

Parallel systems of processors can be divided into two important classes of systems: 'shared memory systems' and 'distributed memory systems'. In a shared memory system all processors address the same physical memory whereas in a distributed memory system every processor has its own memory banks. It is obvious that in a shared memory system synchronization of processes is especially important because the (shared) memory can be adressed by multiple processors at the same time. On the other hand it is clear that communication between processors is an important issue for distributed memory systems. Therefore these systems are sometimes also referred to as 'message passing systems'.

Another characterization of parallel systems of processors can be made in terms of the number of processors. In shared memory systems the number of processors is limited due to the physical restrictions in connecting the processors to one central memory. In distributed memory systems the number of processors is unlimited in principle. But because for a practical system the communication lines between processors have to be limited as well, there is a certain limit to the number of processors of these systems as well. At present the largest shared memory sytems suitable for computational fluid dynamics consist of 64 processors, whereas the largest distributed memory systems hold up to 16384 processors.

The programming of a code for a parallel system can also be quite different for both systems. For a shared memory system it is most natural to have one program in which parts that can be performed in parallel are indicated explicitly. For a distributed memory system it is most natural to have (copies of) programs running on all processors, with built-in message passing instructions. Both types of parallel programming are strongly in development and become more and more easy to use. Next we consider the strategy which is normally followed to parallelize a code and we discuss some tools for actually doing this. In Section 8.2.2 we then discuss some parallel implementations of the numerical model for nonlinear water waves.

8.1.3 Parallelization strategy

A natural approach towards parallelization is to search for those parts of the algorithm which can be performed independently from each other and which are performed sequentially. Most often these parts are coded within loops and for an efficient parallelization the largest work loops are searched for first. If not all of the computational work is parallelized then the efficiency of the parallel system with respect to wall-clock time is strongly affected. We quantify this in the following.

With T_p we indicate the wall-clock time which is needed to run a computa-

tion on p processors. The speed-up S_p is then defined as

$$S_p = \frac{T_1}{T_p} \tag{8.1}$$

and the efficiency E_p as the speed-up per processor

$$E_p = \frac{S_p}{p}. \tag{8.2}$$

If f is the fraction of the program that can be performed in parallel then $T_p = fT_1/p + (1-f)T_1$ and speed-up and efficiency can be expressed as

$$S_p = \frac{p}{f + (1-f)p} \text{ and } E_p = \frac{1}{f + (1-f)p}. \tag{8.3}$$

This can be considered as the parallel version of Amdahl's law. It indicates the maximum speed-up which is to be expected for a given number of processors. For a given fraction f the maximum speed-up equals $1/(1-f)$.

With the definitions given in Section 7.4.2 it is possible to express f in terms of total number of panels and number of subdomains under the assumptions made there. In Section 8.3.1 measured required numbers of floating-point operations will be used to determine f for the example presented there and the corresponding maximum attainable speed-up will be computed for this case.

The actual parallelization of the loops can be done in different ways. Next we consider some common approaches towards parallelization.

Shared memory systems; Compiler directives

Most shared memory systems are provided with compilers that support compiler directives which indicate which parts of a code can be executed in parallel. Three different levels for parallel execution can be distinguished: Parallellism on job-level, on subroutine-level and on DO-loop level. On all these levels compiler directives can be used. The presence of compiler directives forces the compiler to write additional code to handle the parallel execution. The efficiency of this extra code strongly depends on the ability of the compiler to reduce the overhead to a minimum. By writing more explicit compiler directives the code can be optimized in this respect.

Distributed and shared memory systems; Message passing

Since the 80's message-passing libraries have been developed which contain instructions to transfer data from one processor to another. Two popular message-passing libraries are PVM (Parallel Virtual Machine, [28]) and MPI (Message Passing Interface, [53]). Both libraries can be used on many different systems.

This is an important advantage because coding based on these libraries can easily be transfered from one system to the other.

Message-passing instructions are used whenever processes have to be synchronized. In general the use of message passing requires a careful analysis of the data flow and is considered to be less easy than the use of compiler directives.

8.2 Present implementations

As mentioned in the beginning of this chapter some important parts of the numerical method for nonlinear waves are performed independently in the subdomains and therefore a logical approach is to parallelize the method based on the subdomain division. The implementation of the domain decomposition method itself has already been discussed in Section 7.2.2. It was mentioned there that numerically all subdomains can be considered as separate one-domain problems, apart from the coupling over the interfaces and the positioning of the interfaces. First we describe the coding of the sequential implementation in more detail. Next we describe some possible parallel implementations.

8.2.1 Sequential implementation

The data is organized in subdomains which consist of separate sets of networks. The subdomains are ordered following the direction of the subdomain division and the networks are ordered per subdomain so that the networks of subdomain 1 are treated first, then those of subdomain 2 and so on. For some algorithms only the interface networks are needed which are then treated in the same order. The structuring of the networks themselves into panels is not relevant for the subdomain division and is not discussed here.

Each time step of the numerical method consists of a number of intermediate time steps which all contain the same algorithms. The order in which they are executed is the following:

1. Determination of network intersections.

   Based on the positions of the collocation points the intersection of all networks with adjacent networks (in the same subdomain) is determined. (DO-loop over all networks)

2. Synchronization of network intersections.

   Intersections between two adjacent subdomains are represented by intersections between networks in both subdomains. Because they are computed independently in both subdomains the results from both network intersections have to be averaged to obtain unique coordinates for the intersection points. (DO-loop over all interface networks)

3. Determination of geometric quantities.

   Based on the positions of the collocation points and of the intersection points, geometric quantities such as panel size, direction cosines and discretization weights are determined. (DO-loop over all networks)

4. Computation of influence coefficients.

   (DO-loop over all subdomains with per subdomain a DO-loop over the subdomain networks)

5. The iterative procedure of the domain decomposition.

   Over a number of iteration steps the Laplace problems in the separate subdomains are solved and coupled:

   (a) Solution of the matrix problems.
   Each subdomain has its own Laplace's equation to be solved. (DO-loop over all subdomains)

   (b) Coupling of the boundary conditions over the interfaces.
   (DO-loop over all interface networks)

6. Time stepping.

   The time integration of the time-dependent variables consists of several parts, but are treated as one here. (DO-loops over all networks)

This implementation is illustrated for both a one-domain problem and a subdivided problem in Figure 8.1.

8.2.2 Parallel implementations

In this section we describe four parallel implementations. In the first two only the computation of the influence coefficients and the solution of the system of linear equations are parallelized. Both compiler directives and message passing have been used so that a comparison with respect to introduced overhead is possible. In the other two the complete numerical method is parallelized using message passing.

Using compiler directives

As is clear from the overview of the sequential implementation given in Section 8.2.1 there are many DO-loops in which the computations in the separate subdomains are performed independently and can be executed in parallel. By simply adding compiler directives at the call of the DO-loop the compiler takes care of the parallel execution of the particular DO-loop.

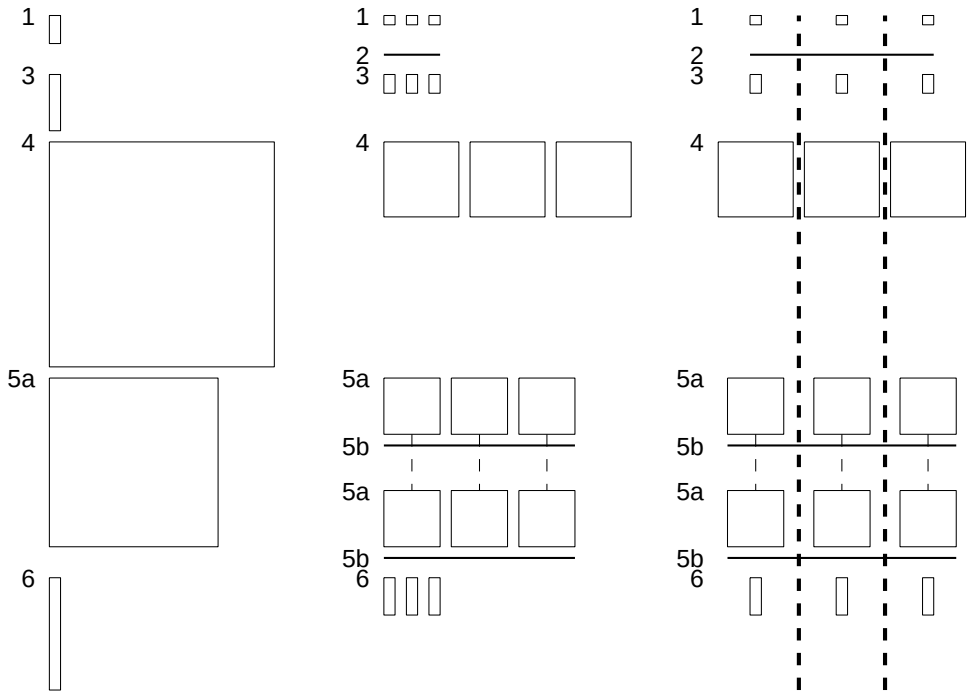

Figure 8.1 Illustration of the algorithms called consecutively in one intermediate time step for a undivided problem (left figure) and a 3-subdomain problem (middle and right figure). In the right figure the use of 3 processors, one for each subdomain, is indicated by separating the computations by dashed lines. The size of the rectangels indicate the amount of computational work.

In the implementation presented here we have only parallelized the computation of the influence coefficients and the solution of the system of linear equations as these are the parts in which most of the computational work is done. The degree of parallelization can be increased by using compiler directives in more parts of the coding.

Because overhead is introduced by the use of compiler directives, it is interesting to compare this implementation with an implementation using message passing in which the same two DO-loops are parallelized. Implementations using message passing are described next.

Using message passing

In the message-passing implementation a single program-multiple data approach has been employed, where each processor performs the computations needed in the subdomains that have been assigned to it. A master process is started which then starts a number of slave processes on the processors in use. Message-

passing instructions are used only at the synchronization points. Implementations differ on which processors handle which subdomains and on which (local) synchronization points have been coded with message-passing instructions.

We repeat here that global synchronization points occur in monitoring the convergence of the overall iterative process of the domain decomposition method and the determination of the (adaptive) time-step size. Local synchronization points occur for updating the interface boundary conditions in the domain decomposition process. Because time-stepping is performed separately in all subdomains another local synchronization point is needed to make sure that geometrical data of corresponding panels of the two networks of every interface are identical.

Three different implementations have been developed.

1. Partial parallelization.

   In order to compare the use of message passing with the use of compiler directives described above, a message-passing implementation has been developed in which the same two DO-loops are parallelized as in the approach using compiler directives. In this implementation the program is run on each processor but these two DO-loops are performed only for the subdomains assigned to it. The computations in the other parts are performed for all subdomains on all processors.

2. Full parallelization.

   In this implementation all computations required in a subdomain are executed only on the processor to which this subdomain is assigned. Within this approach two different implementations can be distinguished.

   (a) Message-passing instructions for the local synchronization points are only called for subdomains that connect to subdomains assigned to other processors (data per processor). For local synchronization of data of adjacent subdomains assigned to the same processor, the coding of the sequential program is used. Therefore the coding of this approach is essentially the same as the coding of the sequential implementation. This implementation has been used in the computations presented in Section 8.3.1.

   (b) Message-passing instructions for the local synchronization points are called for all subdomains, also when they are assigned to the same processor (data per subdomain). The coding of this approach is essentially the same as the coding of the original sequential implementation without the domain decomposition technique. Compared with the data-per-processor implementation this involves more message-passing instructions especially when using many subdomains and few processors. This implementation has been developed first and has

been used in the computations presented in Section 8.3.2 where the number of subdomains is quite modest.

Finally we remark that the implementation of the three-dimensional code is as straightforward as the two-dimensional code because the data structure of subdomains and networks is the same. Only the bookkeeping of the panels on each network is more complicated. In this thesis only examples of the parallel implementation of the two-dimensional model will be presented. But, when size and convergence behaviour of three-dimensional problems are comparable, we expect the results of the two-dimensional examples to be transferable to the three-dimensional problems.

8.3 Results

In this section we present results of the use of two different parallel systems. The wave problems which are used to study the parallel performance are chosen in accordance with the speed of the parallel system. In Section 8.3.1 we will study the performance of a SGI Power Challenge computer with 8 processors. The wave problem studied involves the propagation of a wavegroup in a domain of length 5 kilometer and with the use of up to 200 subdomains. In Section 8.3.2 a heterogeneous cluster of 5 HP workstations is studied. The wave problem under consideration involves the simulation of waves generated by a translating wavemaker. This problem is much smaller and involves at most 16 subdomains.

Results on parallel performance are given mainly in terms of required wall-clock time. Its constituents like required CPU-time, user time, synchronization time and communication time are hard to measure and are therefore not identified much further. On the other hand, the origins of the results on parallel performance are obscured when the system is heterogeneous, not dedicated or when the computational problem has a load unbalance. The examples are presented in order of increasing heterogeneousness of the parallel system and the computations.

8.3.1 Homogeneous shared memory system: SGI Power Challenge

The wave problem which is used in this section involves a propating wave group over an even bottom and is described as an application of the numerical model in Chapter 10. It is described in more detail there. The most important characteristics with respect to the study in this chapter are the following:

- The length of the domain is large (5000 m, built up of 1600 panels on the free surface and 800 panels on the bottom in the one-domain problem)

and has a very small length-to-height ratio (the waterdepth is 12 m with 6 panels on the vertical boundaries).

- The resolution is chosen such that subdivisions are allowed into 25, 32, 40, 50, 80, 100, 160 and 200 equally sized subdomains.

- An initial disturbance is imposed on the free surface and the convergence of the iterative process does not change much during the simulation. For the studies on parallel performance the first 20 timesteps are taken which corresponds to one wave period of the carrier wave of the wave group.

- In order to make the differences in computational load between different subdomains as small as possible, Gaussian elimination and exact computation of the influence coefficients is used so that the computational effort for these two algorithms is independent from the presence of the disturbance in a subdomain. The problem can therefore considered to be load balanced over all subdomains.

First we consider the partially parallelized implementations and compare the introduced overhead using compiler directives and message passing. Figures 8.2 and 8.3 show total user time and speed-up for all subdomain divisions using 1, 2, 4, 6 and 8 processors for these two implementations. Before discussing these results we first describe how the presented quantities are obtained.

In the computations with the implementation using *compiler directives*, the UNIX command 'time' is used to register user time and wall-clock time. The user time indicates the sum of the user times of all processors. The user time of a single processor includes time passed by during non-parallelized parts in which the considered processor is not used and time passed by in which the considered processor is not used due to a load inbalance. This is of importance when during these times the processors are not available to other users. The wall-clock time indicates the real time from the start to the end of the computation and does not significantly differ from the user time divided by the number of processors.

In the computations with the implementation using *message passing*, also the UNIX command 'time' is used for which the wall-clock time contains the same information as mentioned above. This command, however, only registers the user time of the master process and not the user times of the slave processes. Therefore, user time is measured instead by explicit FORTRAN commands which indicate user times between start point and end point for each process, and then adding them. The user time obtained this way includes the user times of non-parallelized parts which are performed on all used processors. Notice that the same times are included in the user time of the implementation using compiler directives. But it does not include wall-clock times of processors which are not used in parallelized parts, when a load inbalance occurs. In the implementation using message passing these processors are then available to

other users. In this respect the parallel system is used more efficiently using message passing than using compiler directives.

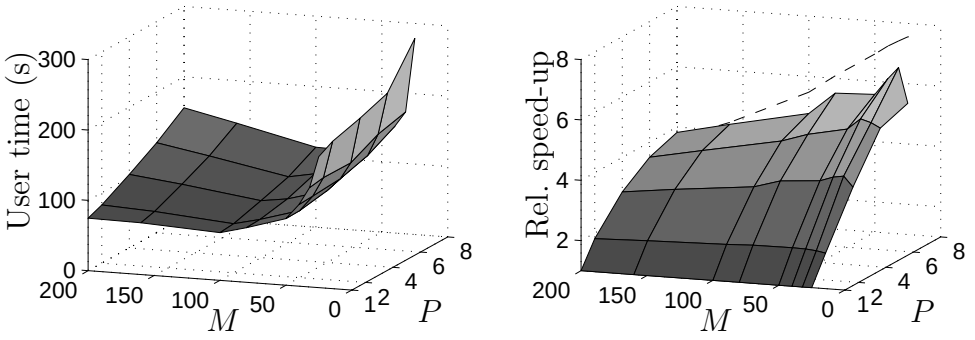

Figure 8.2 User time and speed-up using compiler directives with P processors and M subdomains. Left figure: Total required user time. Right figure: Corresponding speed-up on wall-clock time relative to the one-processor run. The dashed line indicates the maximum attainable speed-up for the 8-processor runs based on equation (8.3) with measured values for f.

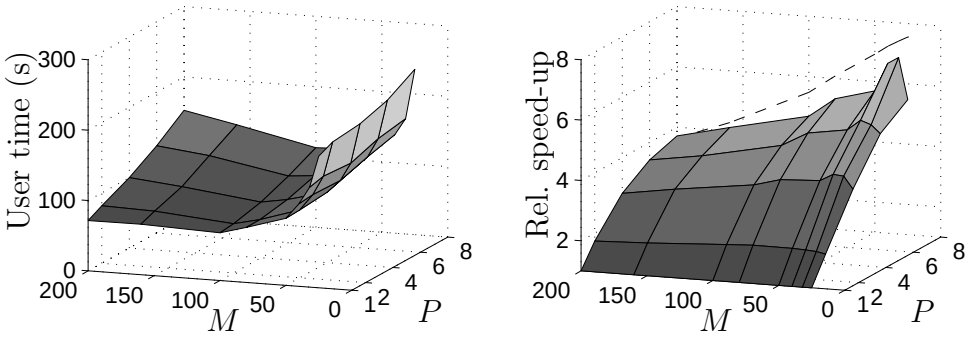

Figure 8.3 User time and speed-up using message passing. Legend as in Figure 8.2.

Firstly it can be seen that differences between the results of both implementations are small. Using 1 processor the required user times are almost identical and show an optimum for 100 subdomains. For both implementations the total required user time increases when using more processors and the increase is largest for the 200-subdomain problem. The largest difference between the implementation using compiler directives and the one using message passing occurs for the 25-subdomain problem using 8 processors. This computation has the largest relative load inbalance (7 processors handling 3 subdomains and 1 processor handling 4 subdomains) and therefore the introduced overhead is larger for the former implementation.

Secondly we consider the results on speed-up. For the runs which have no load inbalance the speed-up using message passing is larger than the speed-up using compiler directives. Apparently more overhead is introduced using the latter implementation. The runs *with* a load inbalance clearly show a loss of speed-up compared with the load balanced runs.

With respect to the indicated line of maximum attainable speed-up we remark the following.
The maximum attainable speed-up is determined from the fraction f which is determined by counting the number of floating-point operations in the routines that compute the influence coefficients and solve the system of linear equations. The parallelized part, however, not only contains those two routines, but also contains a search algorithm that identifies which networks belong to which subdomains. When using many subdomains this search algorithm has a noticable contribution to the total number of floating point operations and for these cases the true fraction f of the 8-processor runs presented here is larger than the one used to determine the maximum attainable speed-up. This line should therefore be positioned a bit higher in the plot, especially for the computations with many subdomains. This explains why for the 200-subdomain problem the presented results exceed the predicted maximum attainable speed-up.

Finally we show the results for the full parallelization using message-passing in Figure 8.4.

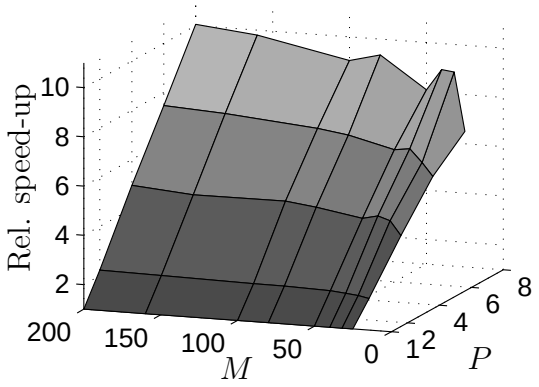

Figure 8.4 Speed-up on wall-clock time relative to the use of 1 processor using full parallelization.

Superlinear speed-up is achieved for all subdomain divisions except for the 25-subdomain problem and the 32-subdomain problem run on 2 and 6 processors. Closer observation of the coding shows that the superlinear speed-up originates from the use of search algorithms, such as the one mentioned above, which for each processor run over all networks of the subdomains assigned to this processor only, instead of over all subdomains. The required CPU-time

for this search algorithm is quadratic in the number of networks and therefore savings are obtained when more processors are used and fewer subdomains are assigned per processor. The search algorithm can be coded in such a way that the required CPU-time only depends linearly on the number of networks. Considering this it can be argued that the results presented in Figure 8.4 do not show the true parallel speed-up. On the other hand it shows that parallelization is able to reveal inefficient algorithms and at the same time that parallel computations may not suffer much from such inefficiencies. Furthermore it can again be observed that speed-up is lost when a run is load inbalanced. This is seen most clearly for the 25- and the 32-subdomain and the 6-processor and 8-processor runs.

8.3.2 Hetereogeneous distributed memory system: HP cluster

In this section we consider the efficiency of a heterogeneous cluster of 5 HP workstations which are connected by ethernet. Typically, the efficiency of a parallel implementation is defined as the ratio of the measured speed-up (as compared with single-CPU performance) to the theoretical, optimal speed-up, as in (8.2). In order to calculate the latter quantity, considering a full parallelization, we now have to take the inhomogeneity of the cluster into account.

Suppose we have a cluster of P workstations, and a decomposition of the domain into M (equal) subdomains. For processor $p = 1, \ldots, P$, define c_p as the relative computational speed of processor p, as compared with processor 1. Throughout this section we assume that this is also the fastest workstation in the cluster. The cluster is used optimally if processor p performs the calculations for a number of subdomains M_p equal to

$$M_p = \frac{c_p}{\sum_{p=1}^{P} c_p} M. \tag{8.4}$$

Here we have assumed that the single-CPU wall-clock time is proportional to the number of subdomains assigned to that processor. For a given distribution $\mathcal{D} = \{M_1, \ldots, M_p\}$ of the subdomains over the workstations we denote by $T_{\mathcal{D}}$ the wall-clock time of the parallel execution. By T_1 we again denote the wall-clock time for the case that all subdomains are assigned to processor 1. Ideally, for the optimal distribution $\mathcal{D}_o$ as given by (8.4),

$$T_{\mathcal{D}_o} = \frac{1}{\sum c_p} T_1. \tag{8.5}$$

In general however, if $\mathcal{D}$ is not the optimal distribution, $T_{\mathcal{D}}$ will be larger. The

efficiency $\eta_{\mathcal{D}}$ is therefore defined as

$$\eta_{\mathcal{D}} := T_{\mathcal{D}_o}/T_{\mathcal{D}} = \left(\frac{T_1}{\sum c_p}\right)/T_{\mathcal{D}}. \tag{8.6}$$

Notice that $\eta_{\mathcal{D}} \leq 1$. In Table 8.1 we give the relative speeds c_p of the separate processors used in the computations. The wave problem which is used here

Table 8.1 Relative processing speeds of the workstations in the cluster. The results are accurate up to about 5%.

processor	c_p
p_1	1.00
p_2	0.78
p_3	0.82
p_4	0.63
p_5	0.62

is discussed in Chapter 9. It involves the simulation of waves generated by a translating wavemaker. The domain is much smaller than the one in the previous section and contains only 256 panels at the free surface and 128 at the bottom. Subdivisions into 2, 4, 8, 12 and 16 subdomains have been chosen, but since the latter subdomain division does not converge at a certain time level, it has not been used in the parallel computations presented here.

For the measurements we have chosen to simulate only the last 20 time steps. During this part of the simulation the wave is present in almost the entire computational domain and the required number of iterations has reached a level that may be considered representative for simulations over larger time intervals. We remark that in contrary to the computations presented in Section 8.3.1 the variation of the number of iterations during the considered time interval is reasonably large for the 2- and 4-subdomain problem and that the average number of iterations is different for the different subdivisions.

To represent practical simulations, the iterative solver CGS has been chosen for these computations. This implies that the number of floating point operations is not equal in all subdomains and a (small) load inbalance between different subdomains is introduced.

The aim of the investigations presented here is to assess the parallel efficiency of different distributions $\mathcal{D}$ of processors for a *given* domain decomposition. As mentioned in Section 8.2.2 these computations were done using the full parallelization with the data organized per subdomain. In Table 8.2 we give the speed-up and efficiency for various choices of the distribution $\mathcal{D}$ of the

subdomains over the processors. This speed-up is measured with respect to the wall-clock time for the case that all subdomains are allocated on the fastest processor. During the tests the cluster was not in use by other users.

Table 8.2 Speed-up and efficiency for various numbers of subdomains and some choices of the distribution of the subdomains over the processors.

N	processors	$\mathcal{D}$	Speed-up	$\eta_{\mathcal{D}}$
2	p_1, p_2	$\{1,1\}$	1.54	0.87
	p_2, p_3	$\{1,1\}$	1.54	0.96
4	p_1, p_2, p_3, p_4	$\{1,1,1,1\}$	2.35	0.74
	p_1, p_2, p_3	$\{2,1,1\}$	1.95	0.88
8	p_1, p_2, p_3, p_4, p_5	$\{2,2,2,1,1\}$	2.40	0.63
	p_1, p_2, p_3, p_4, p_5	$\{3,2,1,1,1\}$	2.24	0.59
12	p_1, p_2, p_3, p_4, p_5	$\{3,3,2,2,2\}$	2.61	0.66
	p_1, p_2, p_3, p_4, p_5	$\{4,2,2,2,2\}$	2.36	0.60

From the results in Table 8.2 we see that the achieved efficiencies are not close to 1 (with one exception). The main reason is that in the definition of efficiency (i.e. (8.4)), in fact the domain is allowed to be partitioned in a non-integer number of subdomains. It is very unlikely that the integer partitions as considered in the experiments are optimal. It is better to allow the subdomains to be of unequal size as will be shown next for the 2-subdomain problem. The test with only the processors p_2 and p_3 achieves an efficiency of almost 1, due to the small difference in computational speed.

In studying the effect of choosing the decomposition of the domain in relation to the computational speed of the processors we only consider the two-subdomain problem run on the processors p_1 and p_2. Even for such a small number of subdomains an optimal subdivision can not be found because the computational work of different algorithms of the method depends differently on the number of panels per subdomain. If a subdivision of the domain is chosen for which one algorithm requires equal CPU-times in both subdomains, then most probably other algorithms require CPU-times that are *not* equal in both subdomains. As an example we consider a subdivision in which the number of panels in the two subdomains is given by $p_1^2/p_2^2 = c_1/c_2$. In this case the speed-up increases to 1.64 and the efficiency equals 0.92.

8.4 Conclusions

In this chapter we have considered some parallel implementations of the numerical method for nonlinear water waves based on the division into subdomains. Both the use of compiler directives for parallel execution as the use of

message-passing libraries can be implemented easily due to the subdivision of the computational work per subdomain.

The parallel computation of two DO-loops containing the largest part of the computational work has been implemented by using compiler directives and message passing and the efficiency of both implementations has been compared on a homogeneous shared memory system. The former implementation introduces a larger overhead in user time for runs which are load unbalanced. For load balanced runs hardly no difference is observed in overhead but speed-up with respect to wall-clock time is larger for the implementation using message passing.

A full parallelization has been developed using message passing and almost all runs with this implementation show a superlinear speed-up on the homogeneous system. Especially for many subdomains the speed-up is influenced by a search algorithm which is executed more efficient when more processors are used. Load unbalanced runs again show some loss in speed-up relative to the load balanced runs.

Computations with the fully parallelized implementation have also been performed on a heterogeneous distributed memory system. Due to the heterogeneity of the system the efficiency of the system is relatively small. It can be increased by using subdomains of unequal size for which the amounts of computational work correspond more to the computational speed of the processors to which these subdomains are assigned.

Chapter 9

Validation and design studies involving wavemakers

9.1 Introduction

In studies carried out in a experimental wave flume, the considered problem is usually investigated for a number of wave conditions. This often involves periodic propagating waves, but also irregular waves with certain prescribed characteristics are used. Because wavemakers are used to generate these wave fields, design and use of wavemakers is an important aspect of using such model facilities.

For the generation of a certain wave field, the velocity of the wavemaker should resemble the velocity field of the desired wave field locally as much as possible. The velocity field can be characterized in terms of the vertical velocity profile and in accordance with this profile a translational or a rotational wavemaker is preferred. Another characterization is based on the wave-induced frequency of the water motion. This is especially important in nonlinear waves because, besides the carrier-wave frequencies, also super- and subharmonic frequencies are present.

The design and use of wavemakers is usually studied with potential flow models using Laplace's equation in the fluid domain and with the velocity of the wavemaker as boundary condition on the inflow boundary. Analytical solutions of these boundary value problems are hard to find because the velocity field has to be imposed on the actual position of the wavemaker which varies in time. Therefore higher-order theories are developed in which the inflow boundary condition is approximated as a perturbation series around the mean wavemaker position. An example of such a boundary value problem is the one discussed in Section 2.4.1 in which the linearized inflow condition (2.25) is used together with the linearized free-surface condition. Wavemaker theories can be distinguished on the basis of the approximation technique and on the basis of the order of

approximation.

In the numerical model for nonlinear water waves, the free surface is modelled by collocation points which move along with the free surface. The free-surface boundary conditions are imposed at the exact position of these collocation points. The inflow boundary condition of a moving wavemaker can be modelled similarly by collocation points following the position of the wavemaker in time and imposing the velocity of the wavemaker at its position. In this way, the boundary value problems associated with a certain wavemaker can be investigated numerically. It offers the possibility to verify wavemaker theory without having to resort to a physical wave flume.

In this chapter we examine the suitability of the numerical model to simulate waves generated by wavemakers. For the examination of accuracy and stability of such computations, we discussed computations of the highly nonlinear model problem in Chapter 4. It was shown that to a large extent this problem can be computed accurately and stable with the two-dimensional model. This gives great confidence in computations of more general problems. But since accuracy and stability are strongly related to the motion of the grid and because no special conditions are imposed on the motion of the lateral boundaries and the bottom in the model problem, some validation is needed to prove the usefulness of the present model.

The difficulty of grid motion in combination with lateral boundaries is associated with the Lagrangian motion of the grid. This is most apparent when doing computations with fixed lateral boundaries such as surface piercing structures. On the other hand the Lagrangian approach is necessary when structures move along with the water or in the case of run-up on a shoreline. A mixed Eulerian-Lagrangian description such as the one discussed in Section 4.4.1, then is the most flexible approach to general wave problems.

The use of a mixed Eulerian-Lagrangian description will not be studied until Chapter 11. In this chapter we describe two studies with the two-dimensional model in which the motion of a wavemaker is simulated on the inflow boundary. In both studies the wavemaker has a fixed mean position and the Lagrangian description is used for the grid on the free surface. Therefore these studies show how Lagrangian motion and fixed lateral boundaries can be combined. In Section 9.2 computations are described in which the motion of a translating wavemaker, as measured in experiments in a physical wave flume, is imposed as inflow boundary condition in the model. In this way direct comparison is possible between experiment and computation. Additionally an example is shown in which theoretically developed wave signals for a translating wavemaker are tested on their correctness. In Section 9.3 some computations are presented which are performed to test the suitability of a design of a rotating wavemaker. No comparison is made with experiments. The results are only discussed in relation to stability and qualitative characteristics of the generated wave field.

9.2 Waves generated by a translating wavemaker

In this section we discuss two different studies involving the simulation of waves generated by a translating wavemaker. In the first study, described in Section 9.2.2, the results of computations are described in which the motion of a translating wavemaker, as measured in experiments in a physical wave flume, is imposed as inflow boundary condition in the model. The wave height measured at several positions in the wave flume is compared with the computed wave height at the same positions and is shown to agree very well. This gives great confidence in the use of the model to verify wavemaker theory and an example of such a verification is shown in the second study described in Section 9.2.3.

The implementation of the inflow boundary conditions is the same for the computations of both studies and is shortly described next.

9.2.1 Implementation

For the simulation of a translating wavemaker in the numerical model for nonlinear water waves, its position is required to prescribe the positions of the collocation points on the wavemaker in time and its velocity is required to prescribe the inflow boundary condition. If the position of a material point on the wavemaker is described by $\mathbf{X}(t) = (X(t), Z)$ then its velocity $\mathbf{V}(t) = (V(t), 0)$ and acceleration $\mathbf{A}(t) = (A(t), 0)$ are given by $V(t) = \frac{DX(t)}{Dt}$ and $A(t) = \frac{DV(t)}{Dt} = \frac{D^2X(t)}{Dt^2}$ respectively, see Figure 9.1.

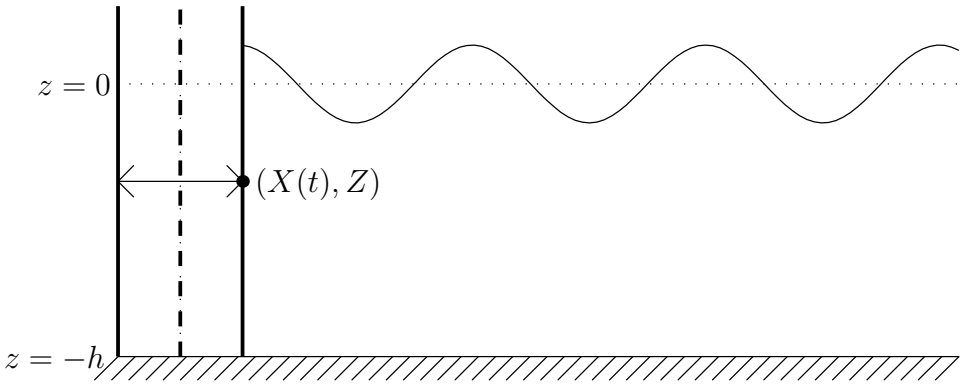

Figure 9.1 Configuration for the simulation of waves generated by a translating wavemaker.

The boundary conditions for the Laplace problems $\Delta\phi = 0$ and $\Delta\phi_t = 0$ are given by

$$\frac{\partial\phi}{\partial n} = \mathbf{V} \cdot \mathbf{n} = \dot{X}, \ \mathbf{x} = \mathbf{X}(t) \tag{9.1}$$

and

$$\begin{aligned} \frac{\partial \phi_t}{\partial n} &= \frac{\partial}{\partial t}\{\nabla\phi\}\cdot\mathbf{n} = \Big(\frac{D}{Dt}\{\nabla\phi\} - \mathbf{V}\cdot\nabla(\nabla\phi)\Big)\cdot\mathbf{n} \\ &= A - V\phi_{xx},\ \mathbf{x} = \mathbf{X}(t) \end{aligned} \tag{9.2}$$

respectively. The term ϕ_{xx} is related to ϕ_{zz} through Laplace's equation, i.e. $\phi_{xx} = -\phi_{zz}$ and ϕ_{zz} is determined from discrete derivatives of ϕ_z along the wavemaker.

With respect to grid motion, most of the algorithms described in Chapter 4 are applied to this problem as well. This includes the Lagrangian motion of the free-surface collocation points, the determination of the intersection point of the free surface with the wavemaker and the (vertical) motion of the collocation points on the (wetted part of the) wavemaker. Because the horizontal motion is prescribed explicitly, no use is made of an alignment velocity to move the inflow boundary along with the free-surface collocation points. In the computations the free-surface collocation points drift away from the inflow boundary as time proceeds. Therefore extrapolations from the free surface to the inflow boundary are taken over increasing physical distances but no problems were observed in the computations in the simulated time interval.

9.2.2 Validation study

In order to validate the numerical model, comparison is made with experiments in a wave flume. These experiments were carried out to test an active wave absorption system and are reported by Luth [50]. To exclude differences due to different incoming wave signals, the measured position $X(t)$ of the wavemaker is taken and the corresponding velocity $V(t)$ and acceleration $A(t)$ are determined. These data are used to formulate time series of values for the boundary conditions $\frac{\partial\phi}{\partial n}$ and $\frac{\partial\phi_t}{\partial n}$ on the (exact position of the) wavemaker.

The waterdepth equals 0.50 m and comparison is made for four experiments. Three experiments involve periodic waves with wave period $T = 0.90$, 1.37 and 2.40 s respectively and one experiment involves bichromatic waves with carrier wave periods $33/16 \approx 2.06$ s and $33/11 = 3.00$ s. In the experiments the waveheight is measured at several locations in the waveflume with a sampling rate of 25 Hz. Comparison with the computations is made at locations $x = 5.0$, 10.0 and 14.2 m behind the position of the wavemaker in rest.

We first discuss the results for the periodic wave with wave period $T = 1.37$ s. Figure 9.2 shows measured and computed elevation and the difference between both signals.

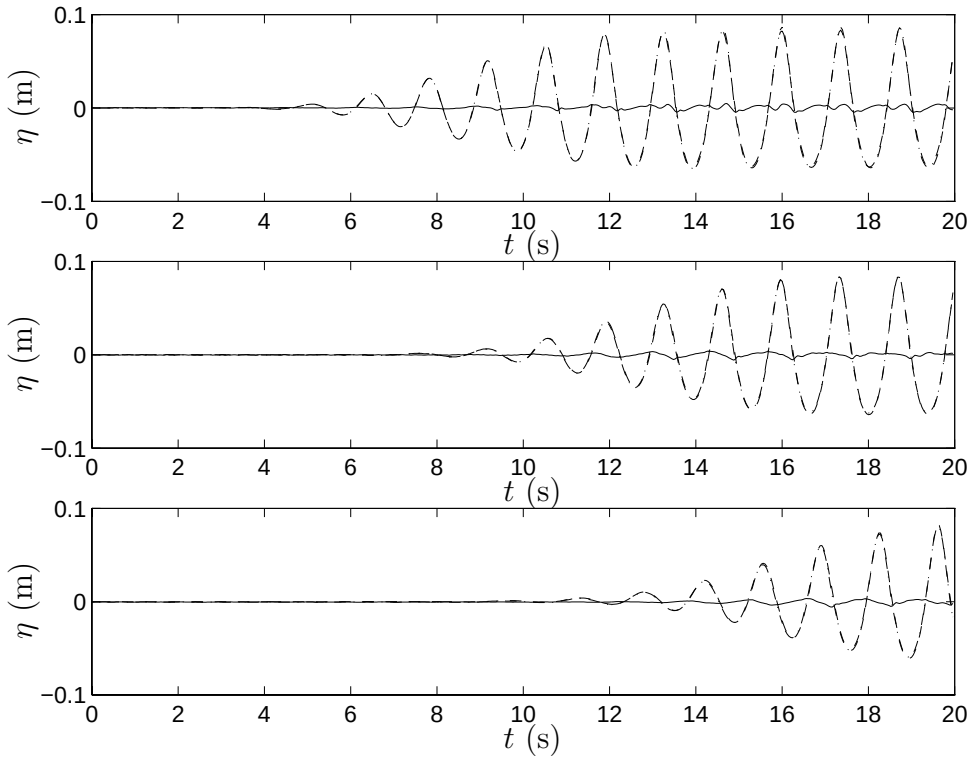

Figure 9.2 Measured (−−) and computed (−·) free-surface elevation at locations $x = 5.0$, 10.0 and 14.2 m for $T = 1.37$ s. The difference between both signals is indicated with a solid line.

The maximum difference between measured and computed free-surface elevation over the time interval [0,20] s at the three locations is equal to 0.0048, 0.0056 and 0.0058 m respectively, which is approximately 6 % of the crest surface elevation. Note that the difference signal is quite regular at locations $x = 10.0$ and 14.2 m but is more irregular at $x = 5.0$ m. To investigate if the regular difference signals are due to phase differences caused by numerical dispersion, the spatial resolution is doubled. As a result the maximum difference between measured and computed elevation at locations $x = 10.0$ and 14.2 m decreases to 0.0032 and 0.0044 m respectively. The maximum difference at $x = 5.0$ m increases to 0.0055 m. This indicates that the observed difference at $x = 5.0$ m is not caused by numerical dispersion.

Since the irregularity for location $x = 5.0$ m is most apparent after $t = 10$ s, it may be the case that the difference is due to spurious waves with smaller propagation velocity. At location $x = 10.0$ m some irregularities can be observed around $t = 19.0$ s. Maybe spurious waves not modelled in the computations, like cross waves or currents due to leakage at the wavemaker, are the cause of these differences, but no explanation has been found yet.

Comparisons between measured and computed free-surface elevation for the

other wave conditions show differences similar to the difference observed at $x = 5.0$ m for $T = 1.37$ s for *all* locations. The differences can not be decreased by increasing the resolution. As an example we show the measured and computed elevation for the bichromatic wave in Figure 9.3.

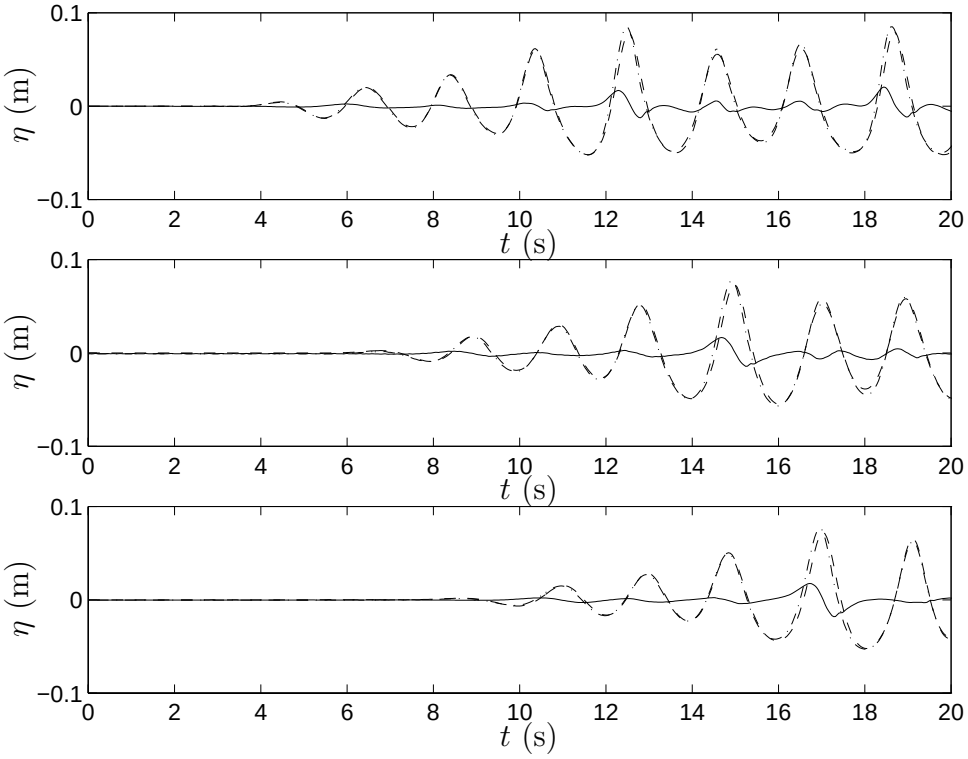

Figure 9.3 Measured ($--$) and computed ($-\cdot$) free-surface elevation at locations $x = 5.0$, 10.0 and 14.2 m for the bichromatic wave. The difference between both signals is indicated with a solid line.

9.2.3 Verification study

For the generation of nonlinear periodic waves, the wavemaker motion has to consist of superharmonic contributions to the carrier wave frequencies as explained in Chapter 2. Also for bichromatic waves and other irregular wave fields, the addition of subharmonic and superharmonic contributions is required when one wants to suppress the generation of spurious waves. The required wavemaker motion is usually studied with the use of perturbation techniques. The nonlinearity of the free-surface boundary conditions and of the wavemaker boundary conditions are accounted for by formulating boundary value problems of different order in a perturbation parameter ϵ. This, in general, requires much algebra and verification is desired to check the results and the range of validity.

Because in the present numerical model the exact positions of the boundaries are followed and boundary conditions are imposed on these positions, the model can be used for the verification of wavemaker signals.

In this section we present an example of such a verification. The theory that is verified is a second-order theory for regular and irregular waves based on a multiple scales method for a translating wavemaker. It is reported by Petit e.a. [56] and is developed to find an alternative for the usual frequency domain approach. In the latter technique, the required computing time for the generation of irregular waves (correct up to second order) is proportional to the square of the number of components compared to the time necessary for the first order signal. For the technique of multiple scales we refer to e.g. Mei [52] and Nayfeh [54].

As a test-case for the theory we consider a domain with waterdepth $h = 0.5$ m and a carrier frequency with period $T = 1.87$ s for the translating wavemaker. The position $X(t)$ of the wavemaker is represented as

$$X(t) = a_1 \sin(\omega t) + a_2 \sin(2\omega t + \phi_2) \tag{9.3}$$

with $\omega = 2\pi/T$ and the following values for the parameters a_1, a_2 and φ_2:

Table 9.1 Values of the parameters used in the verification study.

	a_1	a_2	φ_2
first-order	0.0902	0.0	0.0
second-order	0.0902	0.0061	-0.0607

The wavemaker motion is gradually started by applying a transition coefficient, which increases linearly from 0 to 1 over the time interval $[0, 2T]$, to all relevant signals. The results for these wavemaker signals are compared by considering the free-surface elevation at $t = 10T$, see Figure 9.4. It is clear that no periodic wave field is generated with only a first-order signal. At this time level the wave field in the spatial interval $[0, 15]$ m is disturbed by spurious 2ω-waves. Because these waves travel slower than the waves with frequency ω, no disturbances are noted beyond $x = 15.0$ m. The second-order signal provides the proper wavemaker motion.

9.3 Waves generated by a rotating wavemaker

In this section we discuss computations in which waves are generated by the motion of a hinged rotating wavemaker. The configuration originates from a design study in which an important demand of the design was to be able to

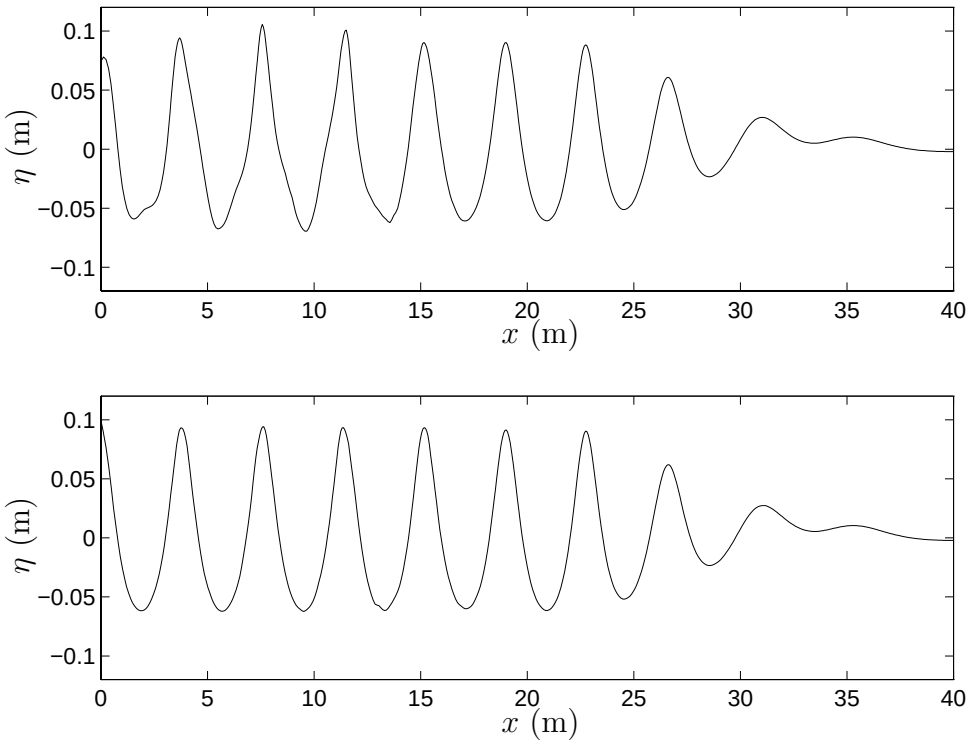

Figure 9.4 Computed free-surface elevation at $t = 10.0$ s for a first-order and a second-order wavemaker signal.

generate shallow as well as deep water waves without too much spurious waves. Because the vertical velocity profile is quite different for both types of waves the position of the hinge is crucial to the design. Other important questions posed in the study, concerned the forces on the wavemaker and required power. The latter two aspects can be considered with the present model due to the possibility to compute the pressure distribution on the wetted part of the wavemaker with Bernoulli's equation.

In the studies presented here the hinge is chosen at a distance $d = 1.20$ m below the still water level and a waterdepth h of 3.50 m, see Figure 9.5. Computations for three different frequencies $\omega = \frac{2\pi}{T}$ of the wavemaker motion have been performed with periods $T = 1.8$ s, 2.4 s, and 3.0 s. The position of the wavemaker is described by its angle α with the vertical. The maximum angular stroke is given by α_0:

$$\alpha(t) = \alpha_0 \sin(\omega t). \tag{9.4}$$

Usually the wavemaker motion is described in terms of a horizontal stroke $S(z)$. Its horizontal displacement is then described by $X(z,t) = S(z)\sin(\omega t)$. See for example Schäffer [63] and Dean [20]. For the same maximum angular stroke α_0,

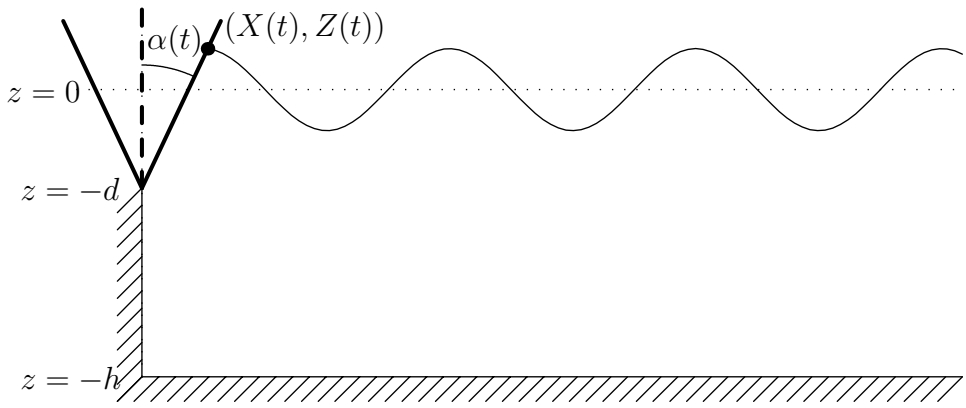

Figure 9.5 Configuration for the simulation of waves generated by a rotating wavemaker.

this description and the description given in equation (9.4), have a difference in maximum horizontal velocity equal to $L\omega\frac{\alpha_0^3}{3} + O(\alpha_0^5)$.

In the present study the amplitude α_0 of the wavemaker motion has been varied in order to show the effect of nonlinearity and to determine the maximum wavemaker motion for which waves start to break. In the following we first describe the computational configuration and its demand on the numerical method. In Section 9.3.2 results of computations are shown.

9.3.1 Implementation

As for the translating wavemaker, the motion of the fluid near the wavemaker can be described by expressing the normal derivative of the potential ϕ in terms of the velocity of the wavemaker itself. The position $\mathbf{X}(t)$ of a material point on the wavemaker is given by

$$\mathbf{X}(t) = \mathbf{X}_0 + L \begin{pmatrix} \sin\alpha(t) \\ \cos\alpha(t) \end{pmatrix} = \mathbf{X}_0 + L \cdot \mathbf{s}(t). \tag{9.5}$$

in which $\mathbf{X}_0$ denotes the position of the hinge, L the distance to the hinge and $\mathbf{s}(t)$ and $\mathbf{n}(t)$ the tangential and normal unit vector on the wavemaker respectively, see Figure 9.5. Its velocity $\mathbf{V}(t)$ and acceleration $\mathbf{A}(t)$ are given by

$$\mathbf{V}(t) = \dot{\alpha}L\mathbf{n}(t),\ \mathbf{A}(t) = \ddot{\alpha}L\mathbf{n}(t) + \dot{\alpha}L\dot{\mathbf{n}}(t) = \ddot{\alpha}L\mathbf{n}(t) - \dot{\alpha}^2L\mathbf{s}(t). \tag{9.6}$$

The potential flow problem in the fluid domain then has the following accompanying boundary conditions for the Laplace problems $\Delta\phi = 0$ and $\Delta\phi_t = 0$:

$$\frac{\partial\phi}{\partial n} = \mathbf{V}\cdot\mathbf{n} = \dot{\alpha}L,\ \mathbf{x} = \mathbf{X}(t) \tag{9.7}$$

and

$$\begin{aligned} \frac{\partial \phi_t}{\partial n} &= \frac{\partial}{\partial t}\{\nabla\phi\}\cdot\mathbf{n} = \left(\frac{D}{Dt}\{\nabla\phi\} - \mathbf{V}\cdot\nabla(\nabla\phi)\right)\cdot\mathbf{n} \\ &= (\mathbf{A} - \mathbf{V}\cdot\nabla(\nabla\phi))\cdot\mathbf{n},\ \mathbf{x} = \mathbf{X}(t). \end{aligned} \tag{9.8}$$

respectively. The latter term involves the second-order spatial derivatives ϕ_{xx}, ϕ_{xz}, ϕ_{zx} and ϕ_{zz} which can be determined similar to the translating case by using Laplace's equation for ϕ to obtain ϕ_{xx} from ϕ_{zz} and taking discrete derivatives of the first-order spatial derivatives.

There is, however, an additional difficulty because these boundary conditions are imposed in the collocation points and not in fixed material points. Because the collocation points on the wavemaker are moved tangentially, an extra acceleration in the normal direction is introduced due to the rotational movement of the wavemaker. This acceleration enters boundary condition (9.8) if it is used for a collocation point moving with velocity $\mathbf{V}_g$. Because at present no proper formulations for the grid acceleration are used, this boundary condition is not used for the solution of $\Delta\phi_t = 0$ and the time-marching is performed using the classical fourth-order Runge-Kutta method. For the computation of the pressure p, however, ϕ_t is required and the above boundary condition is used. The computed pressure therefore contains errors due to the determination of the grid acceleration, but since they are small relative to the material acceleration and enter the computation of the pressure only through the boundary condition, we expect their contributions to be negligible.

Due to the Lagrangian motion, the distance between the first collocation point on the free surface and the wavemaker increases during the computation. The extrapolation which is normally used in the algorithm to determine the tangential component of the grid alignment velocity, described in Section 4.3, therefore gives unrealistic values when this distance becomes very large. It is replaced here by an explicit regridding algorithm. The extrapolation used to determine the intersection point between free surface and wavemaker however does not give any problems. This extrapolation is done in the coordinate system fixed to the wavemaker given by the vectors $\mathbf{n}(t)$ and $\mathbf{s}(t)$. It is clear that problems may arise from the Lagrangian motion and the midpoint description when longer simulation times are required.

The accurate integration of the pressure on the wavemaker with the present Gaussian quadrature formula to obtain the forces and the power on the wavemaker requires a relative high resolution there. The connecting grid on the free surface near the wavemaker has been given a high resolution as well: Computations with low resolution show that as the collocation points on the free surface drift away from the wavemaker, the length of the connecting panel on the free surface become much larger than those on the wavemaker, leading to an ill-conditioned matrix and an unstable computation.

9.3.2 Results

For three frequencies the amplitude α_0 has been varied to examine the effect of nonlinearity. Table 9.2 shows the variation in simulated conditions.

Table 9.2 Used values of α_0 in the simulations.

$T = 1.8$ s	$T = 2.4$ s	$T = 3.0$ s
2.86	2.86	5.73
5.73	5.73	11.46
8.59	8.59	17.19
11.46	11.46	22.92
	14.33	*28.65*

Values printed italized indicate simulations for which the corresponding computation breaks down within 10 wave periods. The break-down in all these cases is caused by the occurrence of extreme steep waves leading to breaking waves or to a free surface near the wavemaker almost parallel to the wavemaker itself.

Results are shown for $T = 1.8$ s, $\alpha_0 = 8.59^o$ and for $T = 3.0$ s, $\alpha_0 = 22.92^o$. Figure 9.6 shows the free-surface elevation after 10 wave periods. In both cases an irregular surface can be observed. A higher wave at the front of the wave is present due to the gradual start of the wavemaker motion (over two wave periods) but also spurious waves can be seen. These waves are due to the first-order motion of the wavemaker, generating free higher harmonics and due to the velocity profile itself which is represented rather badly due to the restricted depth of the wavemaker, especially for the more shallow wave. The spurious waves can also be observed from Figure 9.7 in which free-surface elevations at positions $x = 4.0$, 8.0 and 12.0 m have been plotted for the computation with $T = 3.0$ s, $\alpha_0 = 22.92^o$.

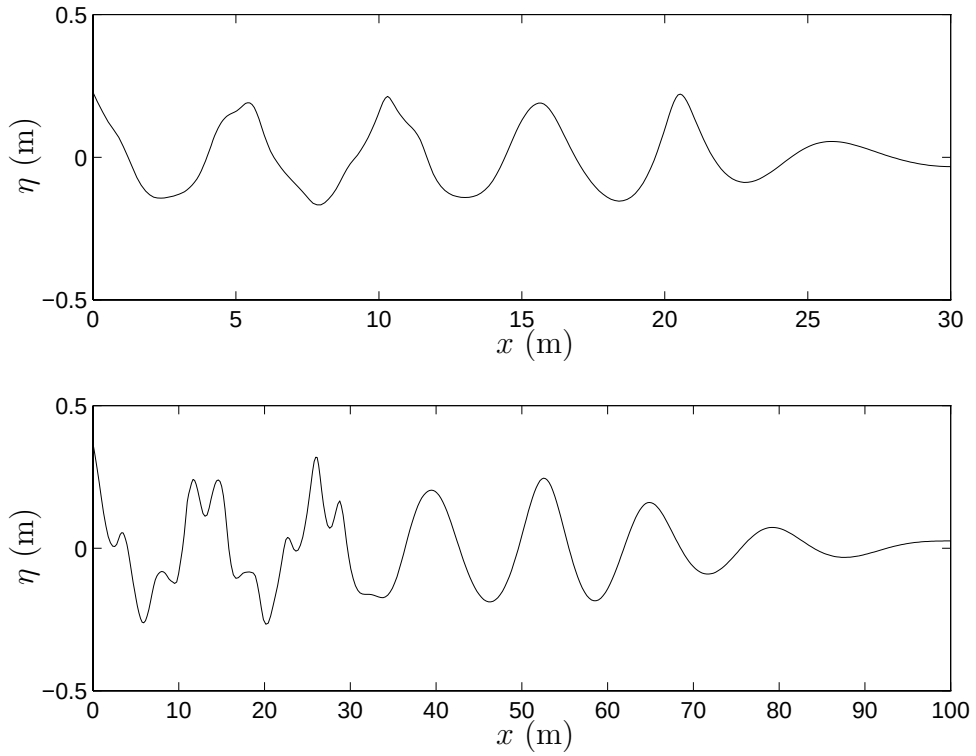

Figure 9.6 Free-surface elevation at $t = 10\ T$. Upper plot for $T = 1.8$ s, $\alpha_0 = 8.59^o$, lower plot for $T = 3.0$ s, $\alpha_0 = 22.92^o$.

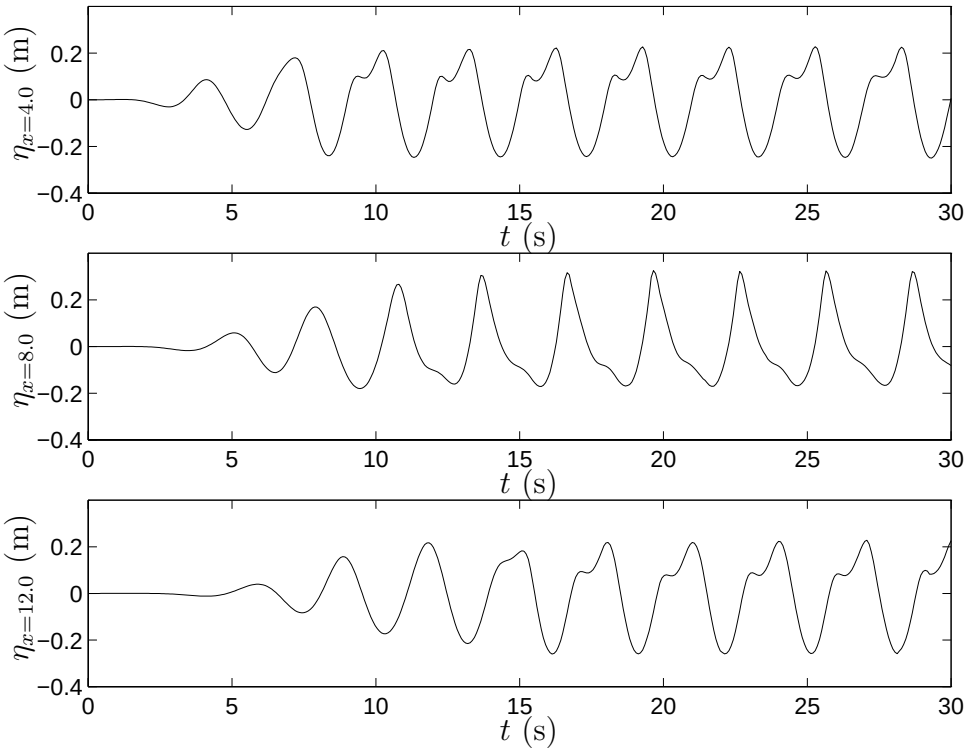

Figure 9.7 Free-surface elevation at positions $x = 4.0$, 8.0 and 12.0 m for $T = 3.0$ s, $\alpha_0 = 22.92^o$.

Figure 9.8 shows time-series of the horizontal force F_x and the moment $M_{y,H}$ around the hinge exerted by the water on the wavemaker, and the hydrodynamic power P and the run-up z_r on the wavemaker.

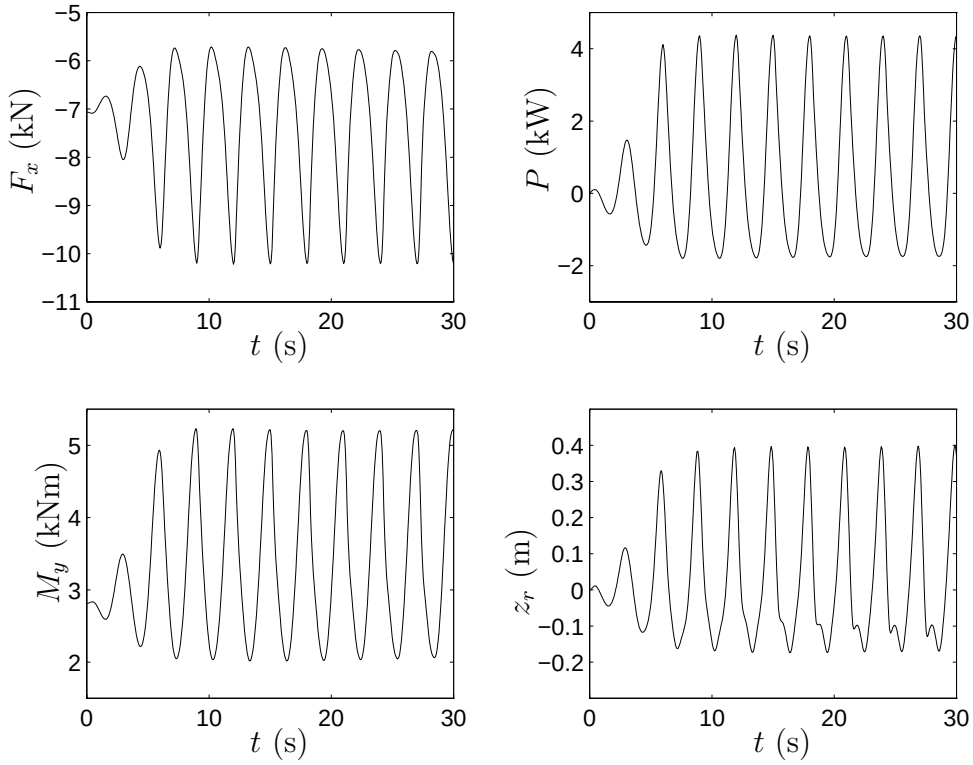

Figure 9.8 Time-series of some quantities for $T = 3.0$ s, $\alpha_0 = 22.92^o$.

These figures clearly show nonlinear behaviour. The graphs for F_x, $M_{y,H}$ and run-up are not symmetrically around the starting values which correspond to the still-water level. The hydronamic power P shows negative values when the wavemaker moves backwards through $x = 0$ m. Note the higher order variation in the run-up which grows towards the end of the computation. This was also seen for $T = 1.8$ s. This is probably due to inaccuracies in the determination of the intersection between wavemaker and free surface towards the end of the computation.

For design considerations the time-series for F_x, $M_{y,H}$ and P have to be corrected for the motion of water at the rear side of the wavemaker and for the momentum of the wavemaker itself. An assumption which can be made to the first purpose, is that waves at the rear side can propagate away undisturbed from the wavemaker. In that case the time-series can be corrected by adding (for P) respectively subtracting (for F_x and $M_{y,H}$) the same time-series but shifted over half a waveperiod. It is noted here that this is only correct for a

symmetric motion of the wavemaker and not when a second-order 2ω-motion is added.

The mass of the wavemaker can be accounted for by simply adding time-series of F_x, $M_{y,H}$ and P for the motion of the wavemaker itself. In Figure 9.9, maximum and minimum values of time-series corrected for the above two purposes (for F_x the mass of the wavemaker has not been taken into account), are plotted for different values of α_0 and compared with results of linear theory. The moment of inertia $I_{y,H}$ around the hinge has been taken equal to 425 kg m^2/s^2 here.

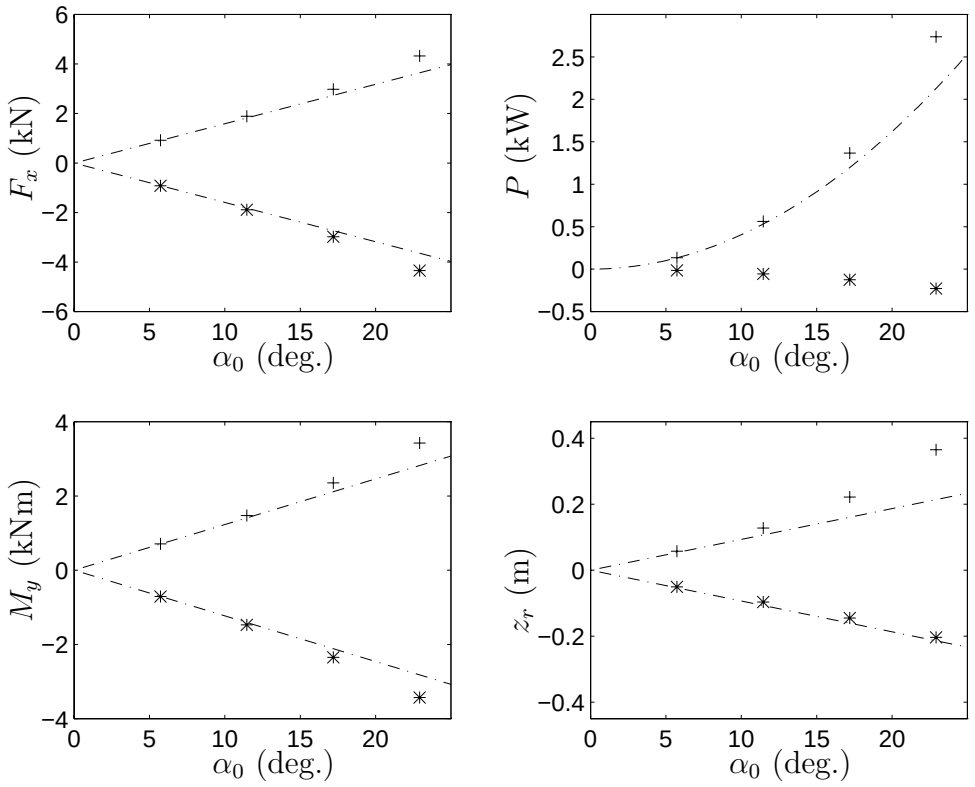

Figure 9.9 Maximum and minimum values of some quantities as function of wavemaker amplitude for $T = 3.0$ s. The dashed lines indicate predictions from linear theory.

A deviation from the results of linear theory can be observed whereas the results for $T = 1.8$ s (not shown here) are identical to those of linear theory.

9.4 Conclusions

With the numerical model for nonlinear water waves it is possible to simulate the generation of waves due to the motion of a wavemaker. Both for a translational

and a rotational moving wavemaker, computations have been performed.

The generation of waves in laboratory experiments is simulated by using the measured motion of a translating wavemaker as inflow boundary condition in the numerical model. Very good agreement can be found with one experiment, but not at all measuring locations. In all other cases differences between measured and computed elevation are small but systematic and show an irregular behaviour. It is expected that this is due to the occurrence of spurious waves in the experiments. Furthermore it is shown that the correctness of signals for higher-order wavemaker motions can be verified for nonlinear periodic waves.

The design of a rotating hinged wavemaker is tested for its suitability to generate nonlinear periodic waves with a first-order wavemaker signal. The generation of waves is simulated for a number of frequencies and it is shown that for this particular case spurious waves are generated, especially for the lower frequencies.

Chapter 10

Propagation of wave groups

10.1 Introduction

In coastal areas long waves, with typical periods of minutes, are of importance in several processes such as sediment transport and vessel motion in harbours. Long-wave motion is usually split up between a bound part which is due to nonlinear difference interactions between short sea and swell waves and a free part which are waves that move with their own phase velocity according to an appropriate dispersion relation. See also Chapter 2. The distinction between bound and free long waves is of importance in many studies on processes such as the ones mentioned above, because the kinematics can be quite different in both kinds of wave motion.

Bound long waves can be transformed into free long waves by the presence of an uneven bottom. This process has been studied by many people. Because of the different time- and space-scales involved, often a multiple-scales technique is used. See for example Nayfeh [54] for an introduction. In this approach an evolution equation for the wave-envelope of a wave-group signal is derived, together with a wave equation for the long-wave motion. For horizontal depth, this was given first by Benney and Newell [6] and later by Davey and Stewartson [19], and for uneven bottom by Chu and Mei [16].

In 1989, Liu and Dingemans [46] reconsidered the problem anew and resolved some arbitrariness in Chu and Mei's approach. The third-order evolution equations derived were later evaluated numerically by Dingemans et al. [22] in a 1D-formulation (2DV fluid motion). An example was shown of a soliton-like wave group over an underwater bar. This example is used here to illustrate the possibility to do large-scale computations with the panel method using domain decomposition.

There is, however, a difficulty in applying the fully nonlinear boundary conditions on signals derived from perturbation techniques such as multiple-scales. The wave group signal in Dingemans et al. [22] is a solution of fixed form in

their formulation but it is not clear to what extent it is of fixed form in the fully nonlinear formulation. Furthermore the influence of applying the domain decomposition technique on accuracy on large scales is yet unknown. These aspects are also studied in this chapter.

Before summarizing the contents of this chapter we would like to refer to another numerical study on propagating wave groups over a bottom topography performed by Barnes [3]. In part of his study he employs an efficient boundary element method described in Cooker et al. [17]. See also Dold [24] for a detailed analysis and optimalization of this method.

In the remainder of this chapter the following is discussed. First, in Section 10.2, the formulations for the wave group signal used in the computations with the fully nonlinear model are presented. Next, in Section 10.3 the results of these computations are shown and discussed with respect to the suitability of the formulations. In Section 10.4 the sensitivity of these results to the use of domain decomposition is analyzed and its efficiency is discussed. Finally we conclude in Section 10.5.

10.2 Formulations for a wave group signal

In Liu and Dingemans [46] and Dingemans et al. [22] a mathematical model is described for the wave envelope A of a carrier wave signal. In this model third-order equations are derived with a multiple-scales technique for a first-order carrier wave signal given in complex notation by

$$\eta_1(x,t) = \frac{1}{2}(Ae^{i\chi_0} + *) \tag{10.1}$$

and

$$\phi_1(x,z,t) = \frac{1}{2}\left(-\frac{g\cosh(k_0(z+h))}{\omega_0\cosh(k_0h)}iAe^{i\chi_0} + *\right) \tag{10.2}$$

with $\chi_0 = k_0x - \omega_0t$, being the phase function of the carrier wave. The $*$-symbol denotes the complex conjugate of the preceding term.

From solvability conditions of the third-order equations, evolution equations are derived for the envelope function A. For a horizontal bottom these equations simplify to a nonlinear Schrödinger (NLS) equation, see also Mei [52]. This equation admits several steady solutions for A which can be used to create an initial signal for a simulation.

In our computations we have chosen a soliton-solution described by the envelope function

$$A(x,t) = a\ \mathrm{sech}\left(\sqrt{\frac{-\nu_1}{\frac{\partial c_g}{\partial k_0}}}\,a\cdot(x-c_gt)\right)\exp\left\{-i\frac{\nu_1a^2}{2}t\right\}, \tag{10.3}$$

in which a and c_g are the amplitude and the group velocity of the carrier wave. ν_1 equals

$$\nu_1 = \left(\frac{\omega_0 k_\infty c_g}{2g \sinh^2 q} + k_0 \right) \frac{\frac{k_0 g^2}{2\omega_0} + \frac{\omega_0^2 c_g}{4 \sinh^2 q}}{c_g^2 - gh} + k_0^2 \omega_0 \kappa \tag{10.4}$$

with

$$\kappa = \frac{1}{16 \sinh^4 q} \left(\cosh 4q + 8 - 2 \tanh^2 q \right), \tag{10.5}$$

$$k_\infty = \frac{\omega_0^2}{g} \text{ and } q = k_0 h, \tag{10.6}$$

see Dingemans et al. [22], p. 364. The parameters have been evaluated for $a = 1$ m, $\omega_0 = 2\pi/6$ rad/s and $h = 12$ m. The corresponding wave length L_0 and group velocity c_g according to linear theory are equal to 50.73 m and 5.52 m/s respectively. Figure 10.1 shows the free surface elevation for these values.

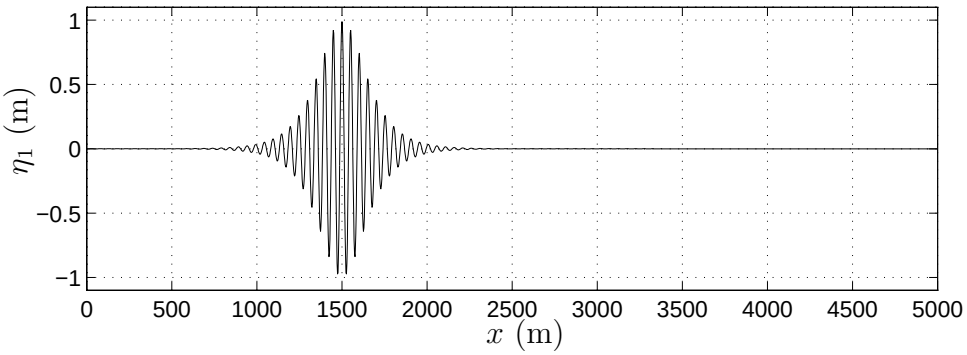

Figure 10.1 Free-surface elevation η_1 of the soliton solution A.

Changes in $|A(x,t)|$ describe amplitude modulations of the waves. In equation (10.3) they are determined by the parameters a, ν_1 and $\partial c_g / \partial k_0$. Based on an elevation $1 \cdot 10^{-3}$ times the maximum elevation, the wave group has a length of approximately 1850 m.

Changes in $\arg\{A(x,t)\}$ describe frequency modulations. For the solution given by equation (10.3) it is quadratic in a and furthermore it is determined by ν_1. For the values of the parameters given above, $\nu_1 \approx 1.2 \cdot 10^{-3}$ so that the frequency modulation is negligible.

The wave signal can be used to provide boundary conditions for the numerical method on the inflow boundary as in Chapter 9, but it can also provide an initial boundary condition for the free-surface boundary. In order to save CPU-time we have used the latter option. A known elevation $\eta(x,t)$ is required as the initial disturbance of the free surface. The panel method furthermore requires

an initial value of the potential ϕ on the free surface which imposes the initial velocity field on the free surface. In the third-order model it is, just as η, given in terms of the third-order perturbation serie. Because of the large and complex expressions associated with the series, we have tried a number of alternatives and have studied the degree in which they describe a signal that propagates undisturbed over a horizontal bottom. These alternatives will be described next. The most appropriate signal can then be used as signal for simulation of a propagating wave group over an uneven bottom in order to study the effect of the bottom topography. The influence of uneven bottoms is not studied here.

A difficulty of simulating nonlinear wave signals consists of the imposition of the boundary conditions at the unknown free-surface elevation. In perturbation techniques, one usually expands free-surface elevation and potential around the still-water level $z = 0$ and the potential is evaluated at the still-water level. In the numerical approach, the grid points are located at $z = \eta$ so that evaluations there deviate from those of the perturbation approach. A Taylor expansion for ϕ can be used to account for the location of the free surface at $z = \eta$:

$$\phi(x,z,t)|_{z=\eta} = \phi(x,0,t) + \eta(x,t)\frac{\partial\phi}{\partial z}(x,0,t) + O(\eta^2). \tag{10.7}$$

Besides the first order expressions given in equations (10.1) and (10.2) we have used a Stokes' second-order contribution given by

$$\eta_2(x,t) = \frac{1}{2}\left(\frac{1}{4}\frac{k_0\cosh(k_0h)}{\sinh^3(k_0h)}[2+\cosh(2k_0h)]A^2e^{2i\chi_0} + *\right) \tag{10.8}$$

and

$$\phi_2(x,z,t) = \frac{1}{2}\left(\frac{3}{8}\frac{\omega\cosh(2k_0(z+h))}{\sinh^4(k_0h)}iA^2e^{2i\chi_0} + *\right). \tag{10.9}$$

and a bound long-wave contribution based on the linearized depth-integrated mean-flow equations, see Longuet-Higgins and Stewart [47]. The free-surface elevation ζ and potential ϕ_{bl} of the latter wave are given by

$$\zeta(x,t) = \frac{2c_g/c - \frac{1}{2}}{2(c_g^2 - gh)}g\left(|A(x-c_gt)|^2 - \left\langle|A|^2\right\rangle\right) \tag{10.10}$$

and

$$\begin{aligned}\phi_{bl}(x,t) &= \int_{-\infty}^{x}\frac{\partial\phi}{\partial x'}dx' = \int_{-\infty}^{x}u\,dx' = \int_{-\infty}^{x}\frac{c_g}{h}\zeta\,dx' &(10.11)\\ &= \frac{2c_g/c-\frac{1}{2}}{2(c_g^2-gh)}\frac{c_gg}{h}\int_{-\infty}^{x}\left(|A|^2 - \left\langle|A|^2\right\rangle\right)dx'. &(10.12)\end{aligned}$$

$\langle|A|^2\rangle$ denotes the mean value of $|A|^2$ over a time interval much longer than the wave group period. For the soliton solution (10.3), $\langle|A|^2\rangle = 0$. The contributions of these higher order terms for the wave conditions used here are illustrated in Figures 10.2 and 10.3.

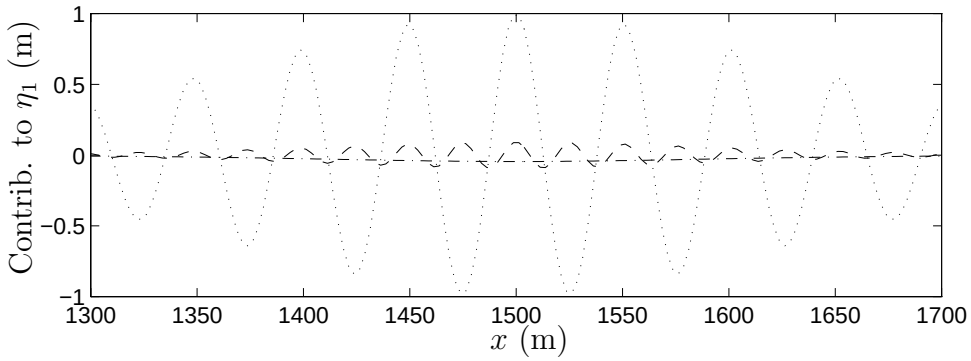

Figure 10.2 Contributions to the first-order free-surface elevation η_1 $(\cdots)$ for the initial signal:
η_2 $(--)$, ζ $(-\cdot)$.

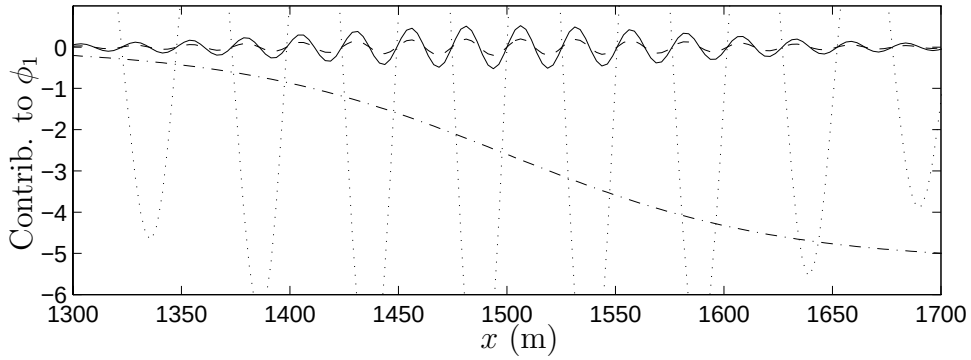

Figure 10.3 Contributions to the first-order potential ϕ_1 $(\cdots)$ for $t = 0$:
$\eta_1 \frac{\partial \phi_1}{\partial z}$ $(-)$, ϕ_2 $(--)$, ϕ_{bl} $(-\cdot)$. Note that the contribution of $\eta_1 \frac{\partial \phi_1}{\partial z}$ is in phase with the second-order contribution ϕ_2.

10.3 Results

For our computations we have selected a number of formulations which are tabulated in Table 1, see page 148.

These initial signals are used in simulations over 60 wave periods in a domain with length 5000 m. Free-surface collocation points are distributed over $z = \eta(x)$ with equal horizontal distances. The resolution of the computational configuration is taken the same for all simulations and is given by $\Delta x = 2.5$ m $\approx L_0/20$ m on the free surface, $\Delta x = 5.0$ m on the bottom and $\Delta t = T_0/20$ s. The computations presented in this section are done using 50 subdomains. Every interface network is covered with 6 collocation points. In Section 10.4 we consider the sensitivity to and the efficiency of the use of a different number of subdomains.

Table 10.1 Initial signal for the various computations.

	η	ϕ	using formula (10.7)
run 1	η_1	ϕ_1	no, ϕ evaluated at $z = \eta$
run 2	η_1	ϕ_1	no, ϕ evaluated at $z = 0$
run 3	η_1	ϕ_1	yes
run 4	$\eta_1 + \eta_2$	$\phi_1 + \phi_2$	no, ϕ evaluated at $z = 0$
run 5	$\eta_1 + \zeta$	$\phi_1 + \phi_{bl}$	no, ϕ evaluated at $z = 0$
run 6	$\eta_1 + \eta_2 + \zeta$	$\phi_1 + \phi_2 + \phi_{bl}$	no, ϕ evaluated at $z = 0$

The results are illustrated best by showing the free-surface elevation at $t = 45T$ for the different computations. In Figure 10.4 it is plotted for runs 2, 4, 5 and 6. Run 1 has been shown separately in a larger plot in Figure 10.5 in order to show the details better. The result of run 3 is similar to that of run 1, because the vertical profile of ϕ_1 is almost linear in the range $-\eta \leq z \leq \eta$. Therefore the differences between the evaluation of ϕ_1 at $z = \eta$ (run 1) and the use of equation (10.7) (run 3) are hardly discernible. Results of run 3 are therefore not shown here. Only if larger amplitudes are used, the difference between both contributions will become noticable.

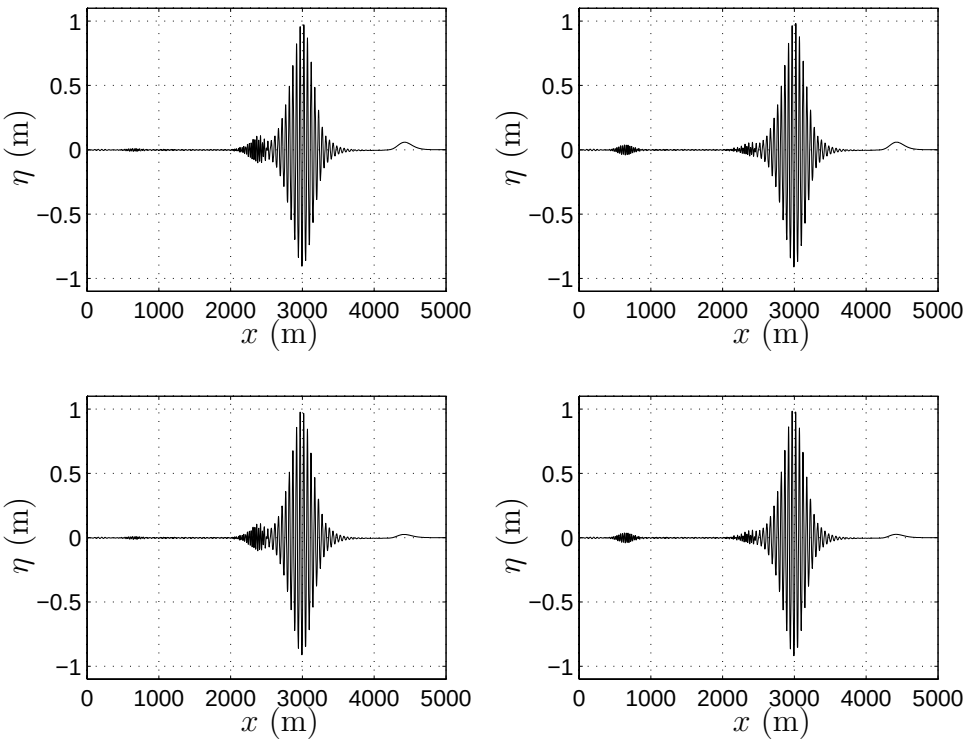

Figure 10.4 Free-surface elevation at $t = 45\ T$ for: run 2 (upper left), run 4 (upper right), run 5 (down left) and run 6 (down right).

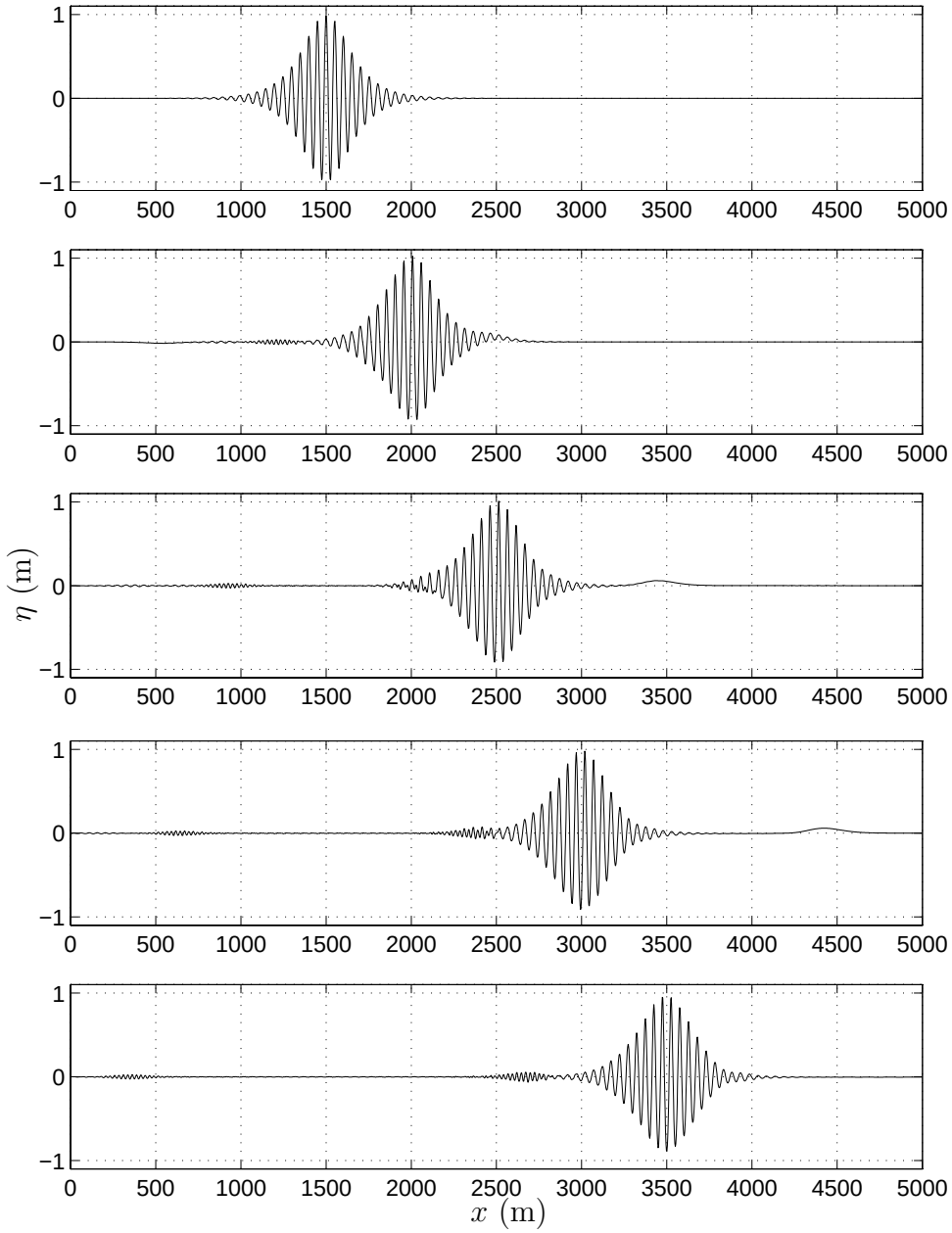

Figure 10.5 Free-surface elevation at $t = 0,\ 15\ T,\ 30\ T,\ 45\ T$ and $60\ T$ for run 1.

A typical feature common to all computations is the generation of small left-going signals. In Figure 10.5 and in Figure 10.4 wave groups with carrier waves with a wave period of approximately 6.0 s (around $x = 200$ m) and 3.8 s (around $x = 700$ m) can be seen. Their group velocities are equal to 4.4 and 3.1 m/s respectively. The left-going signal is smallest in runs 2 and 5. Not

visible in Figures 10.4 is a small left-going *long* wave ($c = 11.1$ m/s) which at this point of the computation has already left the domain. It can be observed in Figure 10.5 for $t = 15T$ around $x = 550$ m. Its amplitude is about 1.5 cm.

There is also a right-going free-long wave (around $x = 4400$ m) in all computations. The computed phase velocity of this wave equals 11.1 m/s. The computations with a contribution of ζ and ϕ_{bl} show a reduction of the amplitude of this wave from 6.0 to 2.5 cm.

At the back of the wave group a smaller right-going wave group evolves consisting of carrier waves with wave period approximately 4.2 s. Its amplitude is smallest in run 4 and 6 which contain the second-order contribution η_2 and ϕ_2.

In summary it can be said that the differences between the computations presented here can be explained satisfactory by relating them to the contributions to the initial wave signal. However, the second-order contributions in runs 4, 5 and 6 do not prevent the generation of free waves nor do they prevent the generation of a left-going wave signal. At this point it is not clear whether this is due to the imposition of the boundary condition at the actual free surface or to the restriction to only second-order contributions. The use of formula (10.7) on the second-order part of the wave signal and the use of more higher-order contributions may improve the stationary character of the signal. By varying the amplitude of the carrier wave signal, indications can be obtained to which order the signal is of fixed form.

10.4 Use of domain decomposition

10.4.1 Effect of domain decomposition on accuracy

In Chapter 7 it was shown for the modelproblem in 2-D that the use of domain decomposition decreases the stability of the computation. It was argued that this is due to the larger errors in the solution of Laplace's equation near each interface because of the one-sided discretizations used there. In the computations presented in this chapter, the nonlinearity is not that high but because many subdomains are used the effect of such errors may become noticable. As the computations can be continued over at least 60 wave periods, stability seems not to be affected much. However, a closer observation of the accuracy is required to determine the effect of using subdomains.

To that purpose, and to the purpose of studying efficiency, we have also performed computations with different numbers of subdomains upto 200. Results on efficiency have been presented in Chapter 8 and only an additional small remark will be made about it in the next subsection. First we consider the effect on accuracy by comparing results of the 50-subdomain problem with the 200-subdomain problem. Again this is illustrated best by showing the free-surface

elevation at $t = 45T$, see Figure 10.6.

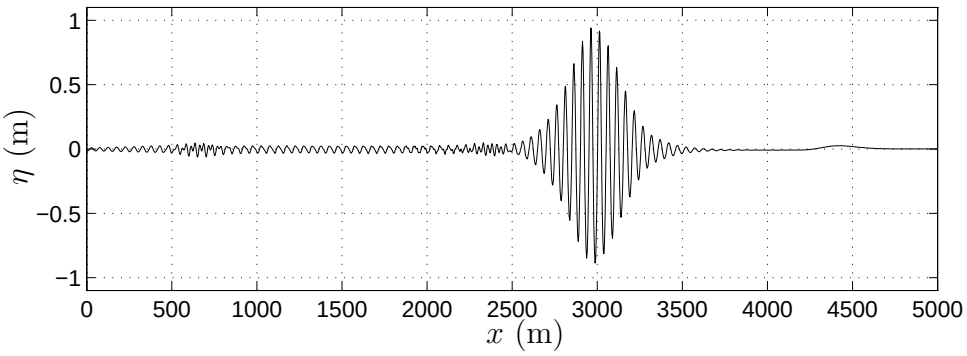

Figure 10.6 Surface elevation at $t = 45T$ for the 200-subdomain problem using the initial signal of run 6.

Most clear from Figure 10.6 in comparison with Figure 10.5 is the occurrence of small periodic waves in the area between $x = 0$ and $x = 2000$ m. The wave length is approximately equal to 50 m which is almost the same as both the length of the subdomains in this case and the wave length of the carrier wave. The amplitude of these waves is approximately 2 cm. They do not grow in time. Waves with the same wave length but an amplitude of approximately 0.4 cm can be observed in the 50-subdomain computation.

Comparison of time series of free-surface elevations at locations at every 500 m show only minor differences between the 50-subdomain and the 200-subdomain computation. Wave speed and wave height of all left-going and right-going waves are the same. The most noticable difference is found in a set-down of the free surface near the right outflow boundary after the free long wave has passed. The set-down in the 200-subdomain computation is 0.9 cm instead of 0.5 cm in the 50-subdomain computation.

From these observations we deduce that at each interface the local truncation error made there, contributes to the generation of small waves. The wave length of these waves is not determined by the subdomain length but by the wave length of the carrier wave because the local truncation error varies in the same rate as the free surface passing the interface. If more interfaces are present then also more waves are generated. The generation of these disturbances is not related to the generation of the free second-order signals. In the 50-subdomain computation they are small in comparison to the relevant signals. In the 200-subdomain computation the disturbances are of the same order which makes this computation not suitable for an analysis of the occurence of spurious waves.

Clearly one should be careful with using many subdomains as the waves generated near the interfaces may disturb the signals in which one is interested

in, or even interact with them. The number of subdomains but also the spatial and temporal resolution limits the degree in which signals can be identified. Variation of these parameters then has to ensure that analysis is not influenced by the use of the numerical method. With respect to such a study, it would be very profitable if the accuracy of the numerical model is not sensitive to the use of domain decomposition. We believe that this can be achieved by locally increasing the resolution near the interfaces.

10.4.2 Remark on efficiency

In Chapter 8 we have already studied the efficiency of the domain decomposition method as function of the number of subdomains. The computations were confined to the first 20 timesteps corresponding to 1 wave period of the carrier wave. To allow many different subdivisions into equally sized subdomains, the resolution was chosen to consist of 1600 panels at the free surface and 800 panels at the bottom. The numerical configuration used for the computations presented in Section 10.3 has a resolution of 2000 panels at the free surface and 1000 at the bottom. Because required CPU-time per time step and required memory by approximation depend quadratically on the number of panels per subdomain, it can be expected that these computational costs are approximately a factor $(3000/2400)^2 = 25/16$ larger than the ones of the corresponding computations presented in Chapter 8.

These figures are given to indicate the computational costs of a complete simulation for different numbers of subdomains. The simulations over 60 wave periods took about 1.5 hours on a Cray C98 computer at a computational speed of about 125 Mflop/s. The required memory was approximately 56 MByte. The use of a single domain for this simulation would have exceeded the capacity of the Cray computer. Moreover it is questionable whether the system of linear equations in this case is numerically solvable within the required accuracy. Therefore the use of the domain decomposition technique is inevitable.

With an eye to larger problems involving a bottom topography it is remarked again that the computational costs per time step depend at most linearly with the size of the computational domain. For comparison with the results of Dingemans et al. [22] on a domain with a length of 15 km, this implies three times as much computational costs per time step. However, a longer simulation time is required for this domain. The computational costs per simulation will then be an additional factor larger than those presented in this thesis. On the other hand we remark that with respect to required CPU-time we expect the use of an iterative solver to be much more efficient, because there are large areas where variables change slowly and only a few iterations are needed.

10.5 Conclusions

By using the domain decomposition method in the numerical model for nonlinear waves, it is possible to simulate the propagation of wave groups over large simulation times. For the formulation of a propagating wave group of fixed form it is important to include higher-order contributions. The release of free waves from the wave group can be related to second-order contributions to the first-order signal. The question remains, however, how to impose an initial signal to obtain a propagating wave group of fixed form over a horizontal bottom.

With respect to accuracy it is observed that near the interfaces disturbances are generated which may disturb the signals. For the computations used here with the higher resolution, these disturbances are not significant for the identification of the spurious wave signals.

Chapter 11

Diffraction by a vertical circular cylinder

11.1 Introduction

A subject which has received considerable attention in recent years, is the so-called 'ringing' effect on offshore structures like gravity base structures (GBS) and tension leg platforms (TLP). Ringing manifests itself in high frequency structural response in the form of transient energetic bursts. The hydrodynamic load mechanism is not yet fully understood but it is known that nonlinear wave forces highly contribute to ringing.

Several simplified approaches have been attempted to predict the nonlinear wave force exerted on a structural member by a steep non-breaking wave such as slender body theories including nonlinear effects due to the free surface and quasi-analytic second- and third order perturbation solutions. There is, however, still a need for more accurate numerical predictions based on a fully nonlinear wave-body interaction formulation. With this objective a comparative study was initiated in 1994 as part of a Joint Industry Project in order to obtain an overview of existing numerical methods for simulations of nonlinear waves. A part of this study involved a test-case for diffraction by a surface-piercing circular cylinder. This test-case is studied in the present chapter.

The results presented in this chapter mainly concern the appropriateness of the numerical method to handle simulations with surface-piercing objects. Because part of the present research project is aimed at developing the numerical method for the simulation of ship motion due to waves, the results can provide valuable insight for the latter problem. No attention is paid here to studies involving ringing although such problems are interesting on its own.

The chapter is divided as follows. First, in Section 11.2, we describe the test-case as it was formulated in the comparative study and the one which

is used in the computations. In Section 11.3 some possibilities of using the numerical method are considered and studied on their appropriateness for the test-case. The most suitable numerical configuration is then used in Section 11.4 for the simulation of the diffraction problem and results of the computations are discussed. In Section 11.5 some concluding remarks about this study are made.

11.2 Description of the diffraction problem

The test-case as it was formulated in the comparative study was chosen the same as the diffraction problem reported by Yang and Ertekin [72]. It involves the diffraction of a modulated finite depth Stokes' wave by a bottom-mounted circular cylinder, see Figure 11.1.

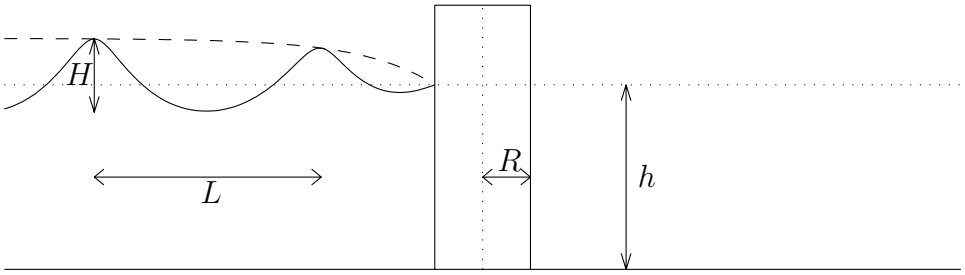

Figure 11.1 Modulated finite depth nonlinear wave incident on a bottom mounted circular cylinder. Indicated are the wave height H, wave length λ, water depth h and radius of the cylinder R.

The values of these parameters are given by

$$H/h = 0.4,\ h/R = 1.16 \text{ and } kR = 1.324\ (\lambda/R = 4.746) \qquad (11.1)$$

In the computations the radius R is taken equal to 1.0 m.

The parameters are such that the problem can be classified as belonging to the so-called diffraction regime, for which the size of the object is of the same order as the wave length. In this regime the contribution of viscosity to the wave forces on the object can be neglected (if there is no background current) and potential theory can be used to accurately determine these forces, see e.g. [62].

A Stokes' second-order wave signal was used by Yang and Ertekin [72] together with a modulation function α. This modulation function is formulated such that the front of the wave signal propagates with a certain (group) velocity c_g and a smooth transition is made near the front of the wave signal, see Figure 11.1. The modulated free-surface elevation η_α and velocity potential ϕ_α

are expressed by

$$\eta_\alpha(x,t) = \alpha(\psi)\eta(x,t) \text{ and } \phi_\alpha(x,z,t) = \alpha(\psi)\phi(x,z,t), \tag{11.2}$$

with $\psi = \psi(x,t)$ specified below and η and ϕ given by equations (2.36) and (2.38) respectively. The modulation function α is chosen as

$$\alpha(\psi) = \begin{cases} 1 - e^{\psi + \overline{k}R} & \text{if } \psi + \overline{k}R < 0 \\ 0 & \text{if } \psi + \overline{k}R \geq 0 \end{cases} \tag{11.3}$$

where

$$\psi(x,t) = \overline{k}(x - c_g t) \tag{11.4}$$

and

$$c_g = \frac{\omega}{2k}\left(1 + \frac{2kh}{\sinh(2kh)}\right). \tag{11.5}$$

The parameters ω and $k = 2\pi/\lambda$ are related by the dispersion relation (2.21) for linear theory and c_g expresses the group velocity according to linear theory. The parameter $\overline{k} = k/2$ is only used for defining the spatial variation of the modulation function α. In the comparative study the maximum non-dimensional horizontal force $\tilde{F} = \frac{F}{\rho g R^2 A}$ on the cylinder and the maximum non-dimensional run-up $\tilde{\eta}_{\max} = \eta_{\max}/A$ on the front of the cylinder were requested, with $A = \frac{1}{2}H$.

For these wave conditions it is found from the Fourier approximation method by Rienecker and Fenton [57] that for a periodic steady wave the third and fourth-order terms significantly contribute to the signal. In terms of the coefficients a_j in expression (2.42) this can be seen from the ratios a_3/a_1 and a_4/a_1 which equal 0.0776 and 0.0305 respectively for a wave with waveheight $H = 2A$. In the computations therefore the signal described by equations (2.42) and (2.43) is used instead of the second-order quantities (2.36) and (2.38).

In the computations presented in this chapter, also use is made of wave conditions with the same wave length but with smaller wave heights. These wave heights are indicated by H_ϵ with ϵ indicating the ratio of used wave height and the wave height given in equation (11.1), (i.e. $H_{100\%}$).

11.3 Use of the numerical method

The diffraction problem with the surface-piercing cylinder requires some specific numerical treatment within the numerical model for nonlinear water waves. This especially concerns the use of smoothly connecting networks on the free surface in 3-D and the use of a mixed Eulerian-Lagrangian description. Another subject which requires special consideration is the use of the boundary conditions on the inflow and outflow boundary. These elements of the numerical model are discussed in this section. See also Section 3.4 where some important features of the model have already been discussed.

11.3.1 Grid motion

When the free-surface collocation points are moved in a Lagrangian way, the collocation points will drift in the direction of wave propagation and it can be expected that especially near the cylinder the grid will distort. An Eulerian description would therefore be appropriate. But as already shown by Broeze [13], the use of the Eulerian description leads to an instability near the inflow boundary where the midpoint description is not able to accurately describe the incoming wave field. A solution for these problems is the use of a mixed Eulerian-Lagrangian description.

In the mixed description used here, the contribution of the Eulerian and the Lagrangian displacement to the grid velocity varies over the grid. The collocation points near the cylinder are moved almost Eulerian and the collocation points near the outer boundaries are moved almost Lagrangian. In between a mixed Eulerian-Lagrangian description is used in which the contribution of the Eulerian part to the grid motion decreases in favour of the Lagrangian part: The grid velocity $\mathbf{v}_g = (\hat{u}, \hat{v}, \hat{w})$ increases from $(0, 0, \frac{\partial \eta}{\partial t})$ to the material velocity (u, v, w). The transition used here is expressed in terms of the position of the collocation points relative to the cylinder: If a collocation point is on the j-th grid line in the outward radial direction, then the grid velocity is given by

$$\mathbf{v}_g = \omega \mathbf{v} + (1 - \omega) \begin{pmatrix} 0 \\ 0 \\ \frac{\partial \eta}{\partial t} \end{pmatrix} \tag{11.6}$$

with

$$\frac{\partial \eta}{\partial t} = w - u \frac{\partial \eta}{\partial x} - v \frac{\partial \eta}{\partial y} \tag{11.7}$$

and

$$\omega = 1 - e^{-\frac{1}{10} j^2}. \tag{11.8}$$

As already pointed out in Section 4.4 the difficulty associated with using grid velocities different from the material velocity is the determination of the second-order time derivatives. Especially near the cylinder where the Lagrangian contribution is small, relatively large errors are made. Therefore it is expected that the use of two-stage two-derivative Runge-Kutta methods introduces errors. We compare results for this method with the classical fourth-order Runge-Kutta method in Section 11.4.

11.3.2 Grid structure

Due to the presence of a surface-piercing object the boundaries of the fluid domain can not be covered with single networks as for example in the subdomains in the problems presented in Chapter 8. The grid is allowed to be curvilinear

but must have a rectangular structure in each network. Therefore the boundary of the fluid domain has to be built up from networks with smooth edges and each edge connecting under an angle with the (two) adjacent edges. For the circular cylinder this can be done by a division such as the one in Figure 3.1, right figure. A topview of the networks around the cylinder is shown in Figure 11.2, left plot.

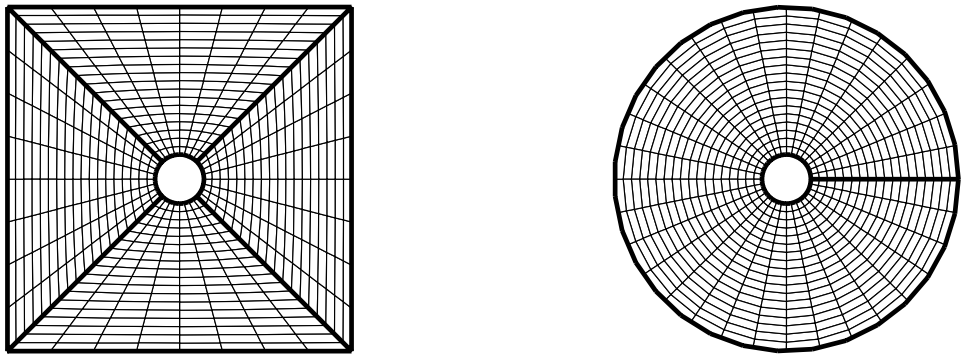

Figure 11.2 Topview of the free-surface networks. Left plot: Cylinder imbedded in four networks. Right plot: Cylinder imbedded in one network.

For this division the resolution on the four free-surface networks adjacent to the cylinder coarsens in the direction of the gridlines parallel to the intersection line. If one wants to enlarge the domain, other networks can be attached to the free-surface networks with the same resolution at the straight edges opposite to the edges of the cylinder. An example of such a grid is shown in Figure 12.1. In the present implementation one-sided discretizations are used near the intersection lines between smoothly connecting networks.

An alternative boundary division which has fewer smoothly connecting networks is illustrated in Figure 11.2, right plot. In this division there is only one free-surface network. The four edges for this network are the intersection with the cylinder, the intersection with the outer boundary and two edges on both sides of the (straight) intersection of the network with itself. The outer boundary, the bottom and the cylinder can be built up from one network in a similar way. Clearly this division does not allow outward connections of the free-surface networks with other free-surface networks without using a further coarsening grid.

In Section 11.4 computations using these two boundary divisions are presented and it appears that the boundary division illustrated in Figure 11.2, left plot, is not suitable due to the generation of large errors near the intersections of the free-surface networks. The generation of errors due to one-sided discretizations can be overcome by constructing computational molecules which are allowed to cross the intersection lines. This is presently not implemented in

the numerical method.

11.3.3 Lateral boundaries

In the treatment of the lateral boundaries, the movement and the type of boundary condition are important. With respect to the latter aspect, the lateral boundaries should ideally function as both inflow and outflow boundaries. The inflow conditions are specified as function of time and space in Section 11.2 and propose no problem. For the outflow conditions the use of an efficient absorbing boundary condition has to be considered. Because the wave field in diffraction problems can be quite complex, this is not an easy task.

For the use of Sommerfeld's radiation condition, a certain phase speed c and an angle of incident α have to be specified for the outgoing waves. Especially a specification of the latter quantity beforehand is quite uncertain. A disadvantage from using Sommerfelds' radiation condition is furthermore the restriction on the size of the time step due to stability requirements. Broeze showed that for stability the time step Δt has to satisfy $c\frac{\pi}{\Delta z}\Delta t < 2\sqrt{2}$ with Δz the panel size on the outflow boundary.

Another possible absorbing boundary condition, already mentioned in Chapter 2, is the use of sponge layers. This boundary condition has been used by, for example, Ferrant [26] in the same diffraction problem. In his approach the damping term is only applied to that part of the potential different from the potential defined by the incoming wave field.

In the computations presented here we have followed the quite crude approach in which the incoming wave field is used to define the boundary conditions on all lateral boundaries and no absorbing boundary condition is imposed. Therefore the lateral boundaries act as reflecting boundaries for waves different from the incoming wave. In order to generate periodic signals for wave runup and horizontal force on the cylinder over a sufficiently long time interval, the lateral boundaries have to be placed at a sufficiently large distance from the cylinder. In the computations we have varied the size of the domain to determine the influence of the lateral boundaries.

With respect to the movement of the lateral boundaries, the usual approach can be employed in which alignment velocities are determined such that the lateral boundaries follow the edge of the free surface. In the computations presented in this chapter, a different approach is followed using explicit regridding of the lateral boundaries based on the integrated position of the free-surface collocation points. Also the positions of the collocation points on the bottom and the cylinder are determined in this way.

11.4 Results

In this section we present the results of computations in which the options mentioned in the previous section are tested. This involves the use of the mixed Eulerian-Lagrangian description and the choice for the square or the circular free-surface grid. The circular free-surface grid is the most successful one and is used in computations in which the influence of the lateral boundaries is investigated and which are used to generate time signals for horizontal force and run-up.

First we consider the use of the mixed Eulerian-Lagrangian description. In order to study the effect of the second-order time derivatives in the two-stage two-derivative method, this method and the classical Runge-Kutta method are applied both and results are compared. The configuration with the circular free-surface grid in a domain with radius $R_D = 7.0$ m is taken with the resolution indicated in Figure 11.2. The waveheight $H = H_{50\%} = 0.232$ m is chosen for these computations.

The differences in positions of the collocation points of the free surface between both computations is considered at $t = 7.35$ s $\approx 4.08\ T$. It can be indicated as follows. For all collocation points, the x and y coordinates differ at most 8 cm and 4 cm respectively. The difference in the z coordinate at this time level is not larger than 2 cm, but is related to comparing it for different horizontal coordinates. The difference in computed horizontal force on the cylinder and the run-up at the front of the cylinder slowly increases in time and equals approximately 4% of the maximum value at $t = 7.35$ s $\approx 4.08T$. These differences are smaller than the variation due to the use of the *size* of the domain as will be shown later.

Next we consider the suitability of the square and circular free-surface grid. For different wave heights we have examined the stability of the numerical method in a domain with radius $R_D = 7.0$ m and with a resolution as indicated in Figure 11.2. In these computations we use the mixed Eulerian-Lagrangian description in combination with the two-stage two-derivative method.

It appears that even for small wave heights ($H_{10\%}$), the computations with the square free-surface grid break down within half a wave period. Near the inflow boundary x = -7.0 m, the resolution is low and large truncation errors are generated near the edges of the free-surface networks.

The computations with the circular free-surface grid can be maintained much longer. For $H = H_{50\%}$ it can be continued over more than 10.050 s $\approx 5.59\ T$ where as for $H = H_{75\%}$ and $H = H_{100\%}$ it can be continued over 5.325 s $\approx 3.01\ T$ and 2.775 s $\approx 1.61\ T$ respectively. The latter computations break down due to large irregularities near the inflow boundary. For the computation with $H = H_{100\%}$, the free-surface grid and the free-surface elevation at the final time level are indicated in Figure 11.3 in the upper and down plot respectively.

It can be seen that the run-up on the front of the cylinder is very large. It equals 0.70 m at this time level whereas the initial crest-surface elevation equals 0.29 m.

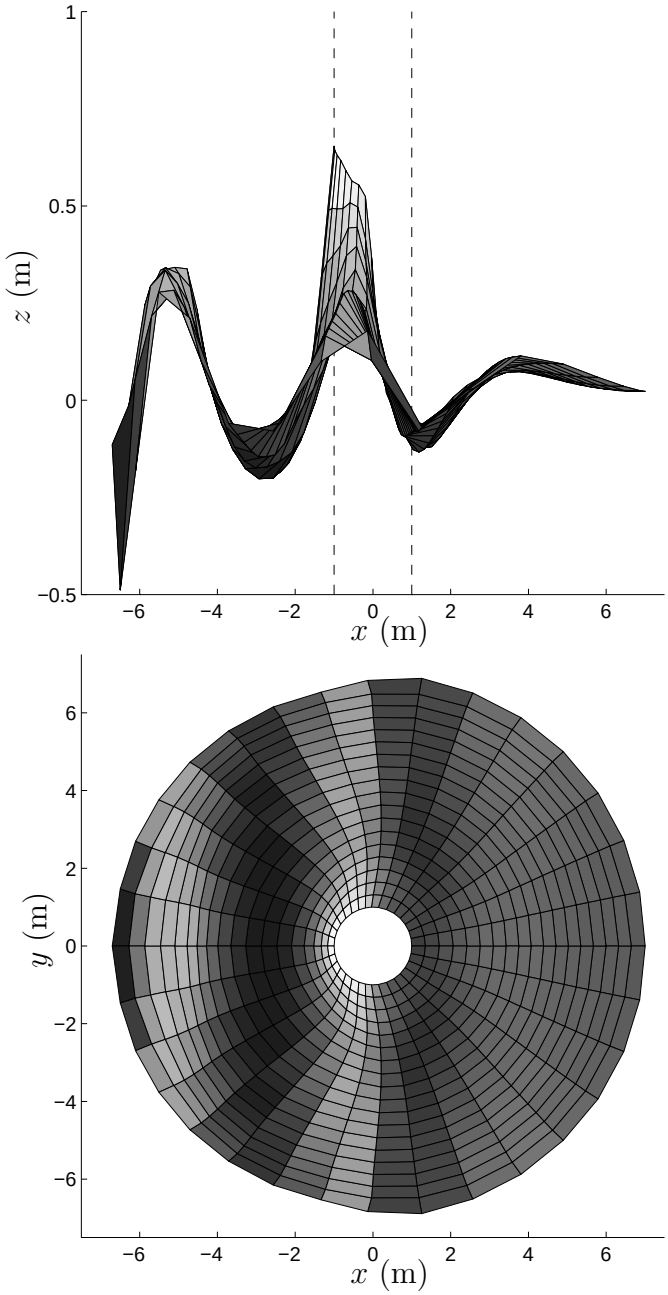

Figure 11.3 Free-surface grid at $t = 2.775$ s for $H = H_{100\%}$. Light coloring indicates positive free-surface elevation and dark coloring indicates negative free-surface elevation.

For $H = H_{50\%}$ the computations can be continued over more than 5 wave periods using the circular free-surface grid. These computations are used to analyze the horizontal force on the cylinder and the run-up at the front of the cylinder as function of time and to determine their maximum values. To investigate the influence of the lateral boundaries on these time series, computations are done using domains with radius $R_D = 7.0$, 9.0, 11.0 and 13.0 m. For increasing radius of the domain, the number of panels in the azimuthal direction is increased from 32 to 40, 52 and 64 respectively in order to maintain the same resolution on the lateral boundaries. Consequently the computational effort increases superlinearly. For $R_D = 13.0$ m it is more than 8 times larger than for $R_D = 7.0$ m. Figure 11.4 shows the time series for horizontal force and run-up for the computations mentioned above.

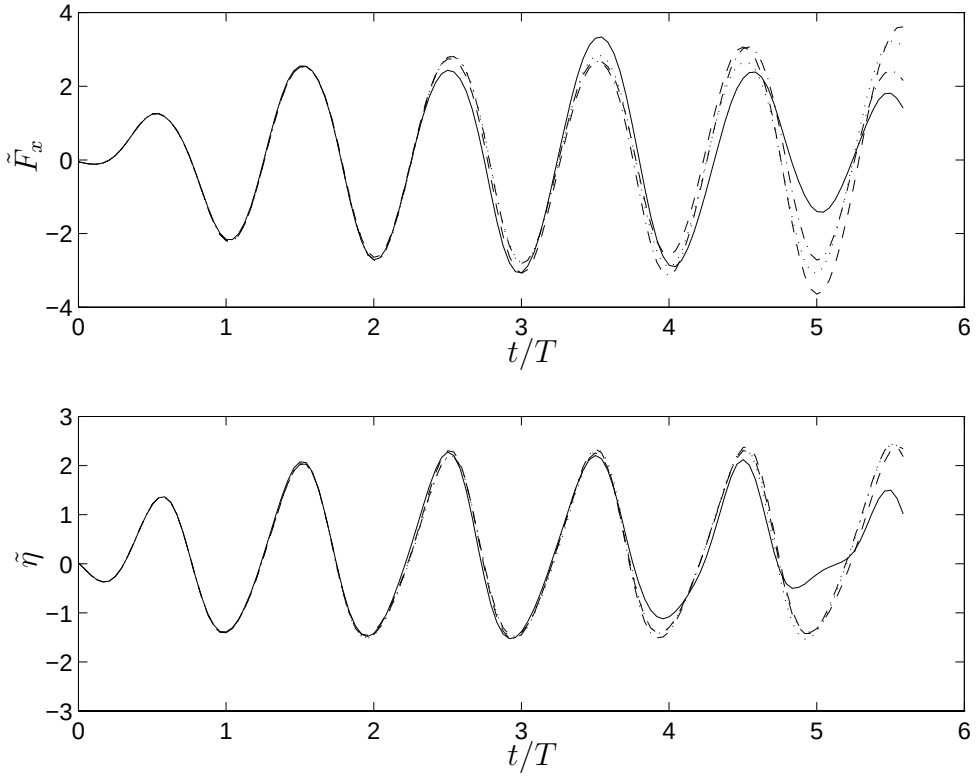

Figure 11.4 Time series of horizontal force on the cylinder and the run-up at the front of the cylinder for $R_D = 7.0$ m $(-)$, $R_D = 9.0$ m $(--)$, $R_D = 11.0$ m $(-\cdot)$ and $R_D = 13.0$ m $(\cdots)$.

The influence of the lateral boundary can clearly be seen for the smallest domain. Already after 2.5 T the force signal for $R_D = 7.0$ m deviates from the other force signals and it looses its periodic character. The differences can not be explained from the effect of reflected waves which again reflect against the

lateral boundaries: For $R_D = 11.0$ m a distance of 20 m has to be travelled. The largest group velocity possible for this water depth can be estimated from linear theory and equals $\sqrt{gh} = 3.37$ m/s which implies that reflected waves need at least 5.93 T to reach the cylinder again.

The differences may be explained from the presence of the lateral boundaries influencing the solution of Laplace's equation. The differences may also be caused by the use of a modulated wave signal; the modulated signal does not completely satisfy the nonlinear free-surface conditions and therefore changes when it enters the domain. In a larger domain the signal changes more before it reaches the cylinder. It is not clear to which degree these two aspects contribute to the differences. It is remarked that the computation of the horizontal force involves the integration of the pressure over the wetted surface of the cylinder and that resolution is different for domains of different size.

The most periodic signal is found for the largest domain $R_D = 13.0$ m. From this signal we estimate the maximum dimensionalized horizontal force to be equal to 2.8 and the maximum dimensionalized run-up to be equal to 2.3. The free-surface elevation at $t = 9.0$ s for this domain is shown in Figure 11.5.

11.4.1 Note on the comparitive study

The computations which were performed with the numerical model for the comparative study in 1994, were quite limited because only the circular domain with radius $R_D = 7.0$ m was used over a smaller simulation time. For a wave height $H = H_{100\%}$ the computation broke down after 2 wave periods due to a sawtooth instability at the intersection of the free surface with the cylinder. The requested quantities, i.e. maximum non-dimensional horizontal force $\tilde{F} = \frac{F}{\rho g R^2 A}$ on the cylinder and the maximum non-dimensional run-up $\tilde{\eta}_{\max} = \eta_{\max}/A$ on the front of the cylinder were found for the computation with wave height $H = H_{50\%}$.

The computations presented in this chapter were performed recently but showed large differences with the 1994-computation for the same configuration. The results of 1994 could not be reproduced and it is not clear why. Differences were observed for the requested quantities but also the stability of the method was different. As already mentioned in Section 11.4 the computation with wave height $H = H_{100\%}$ broke down due to large irregularities near the inflow boundary and not because of a sawtooth instability.

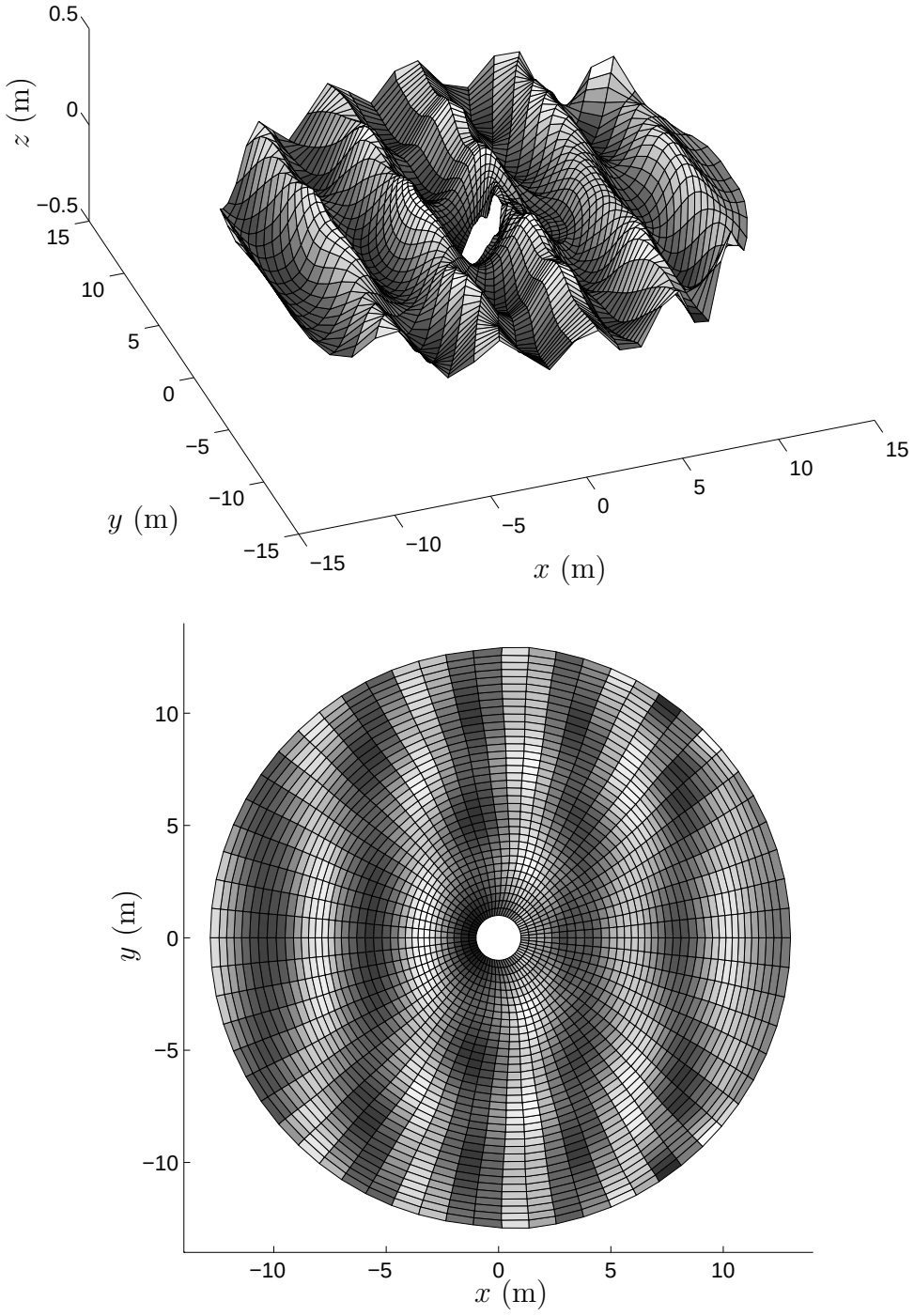

Figure 11.5 Free-surface elevation at $t = 9.0$ s for $R_D = 13.0$ m.

With respect to the differences in the requested quantities mentioned above, we discuss the old and new results in relation to the results reported in the comparative study next. It is remarked that the results of the 1994-computation were obtained for the small domain with $R_D = 7.0$ m and includes the effect of the presence of the lateral boundaries discussed in Section 11.4. The comparative study is reported by Nestegård [55]. It shows results of computations of five different numerical models including the present one from 1994. These results are plotted in Figure 11.6 together with the recent results presented in this chapter.

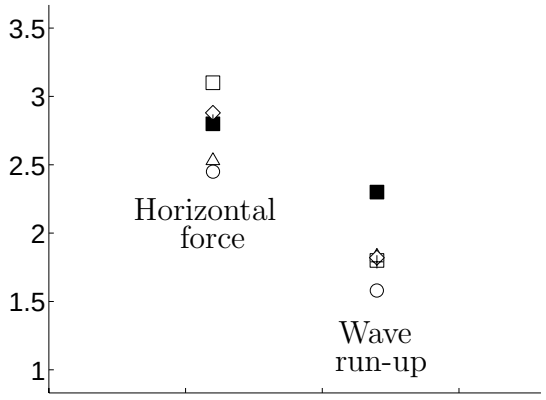

Figure 11.6 Results of five different numerical models for the diffraction problem as reported in Nestegård [55]. The results of the numerical model studied in this thesis are indicated by squares, the recent ones with filled squares.

From this figure it is clear that the results are quite different. Only for the run-up a reasonable agreement is found between four models. With respect to the results of the present numerical model, the maximum horizontal force computed recently is more in line with the results from other models, but the maximum run-up is much larger than those of other models where as the result from the 1994-computation shows much better agreement. Moreover the present result is for waveheight $H = H_{50\%}$. The computation with waveheight $H = H_{100\%}$ at the time level at which it breaks down, shows an even larger run-up of 3.0.

From the comparison of the present results with the results reported in the comparative study, including our 1994-computation, it is hard to conclude what the most accurate results are. Closer study is needed to validate the results presented in this chapter.

11.5 Conclusions

In this chapter we have examined the suitability of the numerical model to simulate waves diffracting around a circular cylinder. Parts of the method which need extra attention for this problem are the mixed Eulerian-Lagrangian description, the grid structure and the lateral boundaries.

With respect to the mixed Eulerian-Lagrangian description it was found that it can sufficiently control the motion of the grid around the cylinder. When using the two-stage two-derivative Runge-Kutta method, the determination of the second-order time derivatives contributes to the generation of errors but for this problem they are small relative to the influence of the lateral boundaries.

For the structure of the free-surface grid, two possibilities are examined. The use of a square free-surface grid is not suitable due to the generation of large errors near the intersection lines of the free-surface networks. With the use of a circular free-surface grid it is possible to simulate the diffracted wave field over more than 5 wave periods for a mildly nonlinear wave. For higher waves the computations break down due to the low resolution near the lateral boundaries.

Time series of the horizontal force on the cylinder and the run-up at the front of the cylinder are compared for computations with circular domains of different radius. For the largest domains the differences in the time series become very small. The time series become almost periodic and maximum values can be determined.

Chapter 12

Conclusions and recommendations

The subject of nonlinear water waves is a subject of both great interest and great attraction. In the classical mathematical sense, problems involving nonlinear water waves are hard to solve. Therefore other ways are sought such as the numerical approach. Nevertheless it is important to take notice of theories in which the nonlinear water wave problem is simplified because they provide tools to understand and describe the nonlinear behaviour. The concept of free and bound waves is an example of a useful description of nonlinear waves and has been described in Chapter 2.

The numerical solution which is described and studied in this thesis has been developed by Romate, Broeze, van Daalen and Zandbergen and is based on a panel method for Laplace's equation. Furthermore it consists of a time-dependent part in which the grid points move along with the free surface. These two parts are suited very well for the task of numerically solving the problem of nonlinear water waves. The details of the method are described in Chapter 3 and previous results concerning accuracy and stability of the method are summarized.

In Chapter 4, some specific algorithms of the numerical model are studied more closely and some inaccuracies are pointed out. These algorithms involve the intersection algorithm and the grid alignment algorithm. Furthermore the grid correction algorithm was studied, especially when used with second-order time derivatives. The latter study included a numerical test for a two-dimensional model problem. In the test an improved Sommerfeld radiation condition was used. The test shows that the second-order time derivatives in the present implementation decrease the accuracy. Nevertheless the numerical method is still accurate and stable over large simulation times for highly nonlinear waves.

Because the computational costs of the numerical model depend superlin-

early on the number of panels used in the discretization of the domain, the use of domain decomposition methods is a promising approach for the reduction of computational costs. In Chapter 5 the D/D-N/N scheme for the coupling of subdomain problems is explained and its relation to other methods known from literature is described. For the efficiency of the method, the convergence of the iterative process of the scheme is of essential importance. It is studied in Chapter 6 and it is shown there that the convergence is determined by the degree of asymmetry in the geometry of the domain near the interfaces. If the interfaces are not too close together, then the convergence on each interface is determined by the local asymmetry only. For the horizontal subdivision considered in our wave problems, the global convergence is determined by the interface with the worst convergence.

In Chapter 7 the use of the domain decomposition method in time-domain computations of nonlinear wave problems is studied. The method is implemented easily by applying the numerical method for one-domain problems to the subdomains separately, extended with coupling algorithms for the iterative process and for the determination of joint intersections. Consequently, the accuracy of solving Laplace's equation is decreased near the interfaces which affects the global accuracy. The efficiency of applying the domain decomposition method depends on many parameters, especially on the use of iterative solvers for the solution of the system of linear equations. Models are formulated to predict computational costs as function of number of subdomains. Some examples show that computational costs can be reduced considerably, especially for large problems.

Because the computations in the separate subdomains are independent to a large extent, parallel computing can be used easily. An implementation using compiler directives and an implementation using message-passing are compared in Chapter 8 and it is shown that the latter implementation is (slightly) more efficient. A completely distributed implementation using message-passing shows a superlinear speed-up for large numbers of subdomains on a homogeneous shared memory system. The efficiency of parallel computing is smaller on heterogeneous systems because it is impossible, in general, to choose the subdivision of the computational load (determined by the subdomain division and the assignment of subdomains to processors) in accordance with the heterogeneity of the system. Still speed-up can be found for heterogeneous distributed memory systems.

In Chapters 9, 10 and 11 the suitability of the numerical model is studied for nonlinear wave problems more related to problems originating from engineering studies. First, in Chapter 9, the use of the method is considered in computations in which waves generated by translating and rotating wavemakers are simulated. For both types of wavemakers, computations can be performed stably over sufficiently long simulation times. For a translating wavemaker, comparison

is made with experimental results and agreement between computation and experiment is found to be very good. It shows that nonlinear wave motion can be simulated accurately for this type of inflow boundary. With an example it is shown that the method can be used to verify the correctness of higher-order wavemaker signals. For a rotating wavemaker, the waves generated by a first-order wavemaker signal are simulated. The computations show the degree in which spurious waves are generated and give extreme values for forces on the wavemaker and required power for the wavemaker motion to generate the waves.

The simulation of a propagating wave group over an even bottom is the subject of Chapter 10. The effect of adding second-order contributions to the initial wave signal is studied. It is seen that groups of free waves separate from the main wave group and that these free waves have smaller waveheights when second-order contributions are added. For these computations, the use of domain decomposition is inevitable as already shown in Chapter 9. The decrease in accuracy because of the use of domain decomposition appears through the generation of spurious waves at the interfaces with the same wave period as the passing waves.

In Chapter 11 the possibility of simulating waves in a domain with a surface-piercing structure is illustrated for a vertical bottom-mounted cylinder. The use of a mixed Eulerian-Lagrangian description is necessary and provides stable computations for moderately high nonlinear waves in a circular domain. The effect of the presence of the lateral boundaries on measured signals on the cylinder is investigated for increasing radius of the domain. It is shown that almost periodic force and run-up signals can be obtained.

Recommendations

Recommendations for further study with respect to the investigations described in this thesis are the following:

- The algorithm for the determination of the intersection of the free surface with connecting boundaries needs further improvement. The original iterative method, described in Section 4.2, should be stabilized or the proposed extrapolation should be extended to deal with *two* curved boundaries. The latter remark also applies to the algorithm for the determination of the grid alignment velocities.

- The use of tangential grid correction velocities is important for the simulation of wave problems with large variations in horizontal velocity. Correct expressions for the associated second-order time derivatives should be found in order to improve the accuracy of the two-stage two-derivative

Runge-Kutta method and of the determination of the pressure. Its effect on the stability of the two-stage two-derivative Runge-Kutta method should be studied.

- Related to the use of tangential grid correction velocities, is the use of mixed Eulerian-Lagrangian methods. They should be used when the free-surface grid distorts due to large variations in horizontal velocities, for example in the vicinity of wavemakers. A mixed Eulerian-Lagrangian method which provides sufficient control over the free-surface grid should be developed.

- The suitability of absorbing boundary conditions for nonlinear waves should be investigated further. The relation between reflection coefficients and phase velocity and propagation direction of outgoing waves should be determined. Also the use of absorbing boundary conditions in combination with incoming wave signals should be investigated closer.

With respect to the use of domain decomposition, the main recommendations are the following:

- In this thesis we have only considered subdivisions of the domain in one direction. For large three-dimensional problems, savings may be obtained if a bi-directional subdivision is used. An iterative scheme for such a subdivision has been proposed in Section 6.2.3 but probably more efficient schemes are possible. Related to this is the use of subdomains with horizontal interfaces, for example for the area between a ship and a quay.

- The relation between the asymmetry of the geometry near interfaces due to wave disturbances, and the convergence rate of the iterative process, should be studied further. It may provide insight how to obtain better coupling of the boundary conditions in order to improve the convergence rate. Also the effect of bottom geometry on the convergence rate should be studied further for the same reason.

- With respect to the numerical implementation of the iterative process, the stop criteria should be formulated in relation to the accuracy of the panel method and the resolution used. Furthermore it should be investigated how the domain decomposition method can be applied in time-domain computations without affecting the global accuracy of the computations. A possible solution is the use of a locally refined grid near the interfaces. A way to increase resolution near the interfaces, is to use the grid points at the intersections of free-surface networks with interface networks as extra collocation points in the panel method and the time integration.

- The latter recommendation applies even more strongly to intersections of smoothly connecting free-surface networks within one subdomain. An

alternative approach for these intersections is to construct computational molecules which cross these intersection lines, thus avoiding one-sided discretizations.

In Chapters 9, 10 and 11, many observations were made how the numerical model for nonlinear water waves should be improved in order to apply it better in studying the water wave problems discussed in these chapters. The most important observations have been mentioned in the recommendations given above. With respect to the studies themselves, it is clear that many interesting questions are still unanswered. Since these questions are not related to the numerical model itself, they are not mentioned in this list of recommendations. It is, however, believed that the numerical model for nonlinear water waves can be a valuable tool in such studies.

Finally we relate the recommendations formulated above to the particular case of the simulation of ships moving in waves. This is one of the most important aims of the ongoing research project and therefore we devote some special attention to such computations. An example of a computation with the linearized free-surface conditions is used to illustrate this and is described next.

The motion of ships due to the presence of waves can be considered for various circumstances. With an eye to the possibilities of the numerical model for nonlinear water waves studied in this thesis, we consider a ship moored at a quay in a harbour with varying bottom topography. Waves entering the harbour, propagate towards the ship and cause the ship to move.

This is simulated with the numerical method in a very rough way with the computation illustrated in Figure 12.1. It shows a configuration in which the ship is modelled by a fixed sphere and the propagation of waves is computed using the linearized free-surface conditions. Waves enter at the left and lower boundary and reflect at the right and upper boundary which are modelled as solid boundaries.

Important features not modelled in this computation are the nonlinearity of the free-surface conditions and the motion of the ship due to the exciting wave forces on the ship's hull. The numerical method, based on the panel method described in this thesis, which takes both these features into account will be described by Berkvens [9]. Furthermore the forces exerted by a mooring system and the shape of a realistic ship are not modelled here.

The recommendations apply to the simulation of ships moving in *nonlinear* waves in the following way. The wetted part of the ship is a boundary of the fluid domain and is covered by networks. In the time-domain simulation, these networks have to move along with the changing waterline. This complicates the determination of the intersections with the free-surface networks. The present intersection algorithm has to be extended for this purpose.

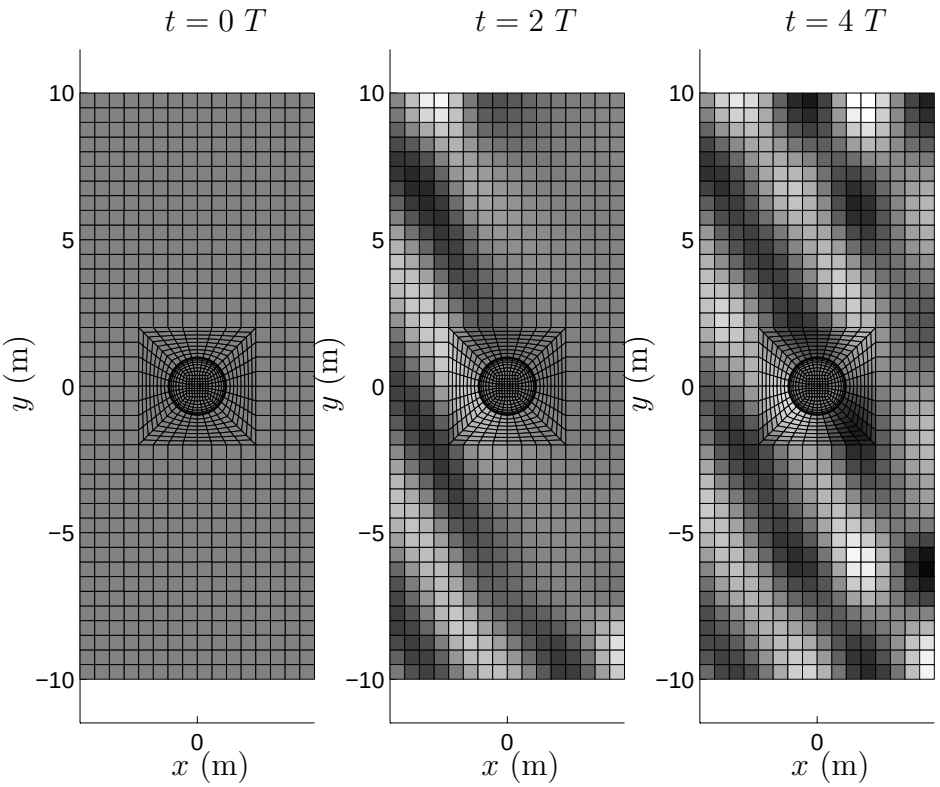

Figure 12.1 Domain used in a computation simulating the propagation of waves towards a sphere. Light coloring indicates positive free-surface elevation and dark coloring indicates negative free-surface elevation.

Similarly to the simulation of waves diffracting around a surface-piercing cylinder, a mixed Eulerian-Lagrangian method is needed to control the grid around the ship. In this method it has to be accounted for that the ship itself moves as well. With respect to the use of absorbing boundary conditions, it is important to know to which degree waves reflecting from the ship affect the accuracy of the computation and what the required size of the domain is.

To be able to model ships moored in a harbour, obviously a large domain is required. Moreover a relatively high resolution on the ship itself is required to describe the hull accurately. The domain decomposition technique is therefore necessary to perform such simulations, but alternative subdivisions than the ones described in this thesis are needed.

Finally we remark that in simulations with ships or other surface-piercing objects, the free-surface grid has to be built up from separate networks. From the experiences with such grids described in this thesis, we infer that developments like the ones described above are necessary for stable computations.

Appendix A

Two-stage two-derivative Runge-Kutta method

In this appendix the coefficients used in the two-stage two-derivative Runge-Kutta method, refered to in Section 4.5.2 are given. Runge-Kutta methods are integration methods for the solution of the initial value problem

$$u_t := \frac{du}{dt} = f\left(t, u\right), \; u(0) = u_0. \tag{A.1}$$

See also equation (3.8). The higher order derivatives are expressed as

$$u^{(l)} := \frac{d^l u}{dt^l} == \frac{d^{l-1}}{dt^{l-1}}\{f\left(t, u\right)\}, \; l \in \mathbb{N}. \tag{A.2}$$

In Runge-Kutta methods the variable u is integrated at a number of intermediate levels and these integrated values are combined to give u on the next level. For the classical fourth-order Runge-Kutta method $u(t+\Delta t)$ is found from $u(t)$ through

$$u(t + \Delta t) = u(t) + \frac{1}{6}\Delta t(k_1 + 2k_2 + 2k_3 + k_4) \tag{A.3}$$

where

$$\begin{cases} k_1 = f(t, u(t)) \\ k_2 = f(t + \frac{1}{2}\Delta t, u(t) + \frac{1}{2}\Delta t \; k_1) \\ k_3 = f(t + \frac{1}{2}\Delta t, u(t) + \frac{1}{2}\Delta t \; k_2) \\ k_4 = f(t + \Delta t, u(t) + \Delta t \; k_3) \end{cases} \tag{A.4}$$

In the two-stage two-derivative Runge-Kutta method, the evaluation of the function f at intermediate time levels is combined with the use of higher-order time derivatives as in Taylor methods. In Taylor methods a truncated Taylor expansion for u is used. The n-th order method can be written as

$$u(t + \Delta t) = u(t) + \sum_{l=1}^{n} \frac{(\Delta t)^l}{l!} \frac{d^l u}{dt^l}(t). \tag{A.5}$$

The Runge-Kutta and Taylor methods are combined to the two-stage two-derivative Runge-Kutta method as follows:

$$u(t+\Delta t) = u(t) + (\gamma_{11}\Delta t\ k_{11} + \gamma_{12}\frac{(\Delta t)^2}{2}k_{12} + \gamma_{21}\Delta t\ k_{21} + \gamma_{22}\frac{(\Delta t)^2}{2}k_{22}) \quad \text{(A.6)}$$

where

$$\begin{cases} k_{11} = f(t, u(t)) \\ k_{12} = \frac{df}{dt}(t, u(t)) \\ k_{21} = f(t+\beta\Delta t, u(t) + \beta\Delta t\ k_{11} + \frac{(\beta\Delta t)^2}{2}k_{12}) \\ k_{22} = \frac{df}{dt}(t+\beta\Delta t, u(t) + \beta\Delta t\ k_{11} + \frac{(\beta\Delta t)^2}{2}k_{12}) \end{cases} \quad \text{(A.7)}$$

The two-stage two-derivative Runge-Kutta methods considered in Chapter 4 and referred to as RK-2-2-a, RK-2-2-b and RK-2-2-c, have their coefficients $(\beta, \gamma_{11}, \gamma_{21}, \gamma_{12}, \gamma_{22})$ equal to

$$\begin{array}{ll} \text{a:} & (\frac{1}{2}, 1, 0, \frac{1}{3}, \frac{2}{3}) \\ \text{b:} & (\frac{2}{3}, \frac{13}{16}, \frac{3}{16}, \frac{6}{16}, \frac{6}{16}) \\ \text{c:} & (1, \frac{5}{6}, \frac{1}{6}, \frac{3}{6}, \frac{1}{6}) \end{array} \quad \text{(A.8)}$$

respectively. The latter method has the smallest contribution of the second-order derivatives.

Appendix B

Convergence analysis for rectangular subdomains

In Section 6.2 we have analyzed the convergence of the domain decomposition method for the D/D-N/N scheme for rectangular domains. For the case of two subdomains it was shown that the convergence can be described in terms of amplification factors which express the growth of the Fourier modes of the error in the interface boundary condition. For the case of three or more subdomains the resulting difference equation for the system of Fourier coefficients on the set of interfaces was given, without specification of the coefficients involved. In Section B.1 of this appendix, expressions for these coefficients are derived. In Section B.2 the eigenvalues determining the convergence of the iterative process in the multi-subdomain case for subdomains of equal length are derived. The expressions for the D/*-*/N scheme are worked out in Section B.3.

B.1 Three-subdomain problems

In this section we analyze the D/D-N/N scheme for a 3-subdomain problem. We refer to Figure 6.3 for the definition of the geometrical quantities for this problem. The solutions of the Laplace problems are determined by boundary conditions $\tilde{\varphi}_1$ and $\tilde{\varphi}_2$ on the interfaces at the even steps and by boundary conditions $\tilde{\psi}_1$ and $\tilde{\psi}_2$ at the odd steps. For all subdomains they can be expressed in the solutions occurring in the 2-subdomain problem as already mentioned in Section 6.2.2. So for boundary conditions

$$\tilde{\varphi}_1(z) = \sum_{n=1}^{\infty} c_{1,n} \sin\left(\frac{n\pi}{h} z\right) \tag{B.1}$$

and

$$\tilde{\varphi}_2(z) = \sum_{n=1}^{\infty} c_{2,n} \sin\left(\frac{n\pi}{h} z\right) \tag{B.2}$$

the solution $\tilde{\phi}_2$ in subdomain Ω_{II} equals

$$\begin{aligned}\tilde{\phi}_2(x,z) &= \sum_{n=1}^{\infty} c_{1,n}\frac{\sinh\left(\frac{n\pi}{h}(x_2-x)\right)}{\sinh\left(\frac{n\pi}{h}(x_2-x_1)\right)}\sin\left(\frac{n\pi}{h}z\right) \\ &\quad + \sum_{n=1}^{\infty} c_{2,n}\frac{\sinh\left(\frac{n\pi}{h}(x-x_1)\right)}{\sinh\left(\frac{n\pi}{h}(x_2-x_1)\right)}\sin\left(\frac{n\pi}{h}z\right). \end{aligned} \tag{B.3}$$

Similar expressions can be given for the solutions $\tilde{\phi}_1$ and $\tilde{\phi}_3$ in subdomains Ω_I and Ω_{III} respectively but consisting of only one serie. The horizontal derivatives of these solutions evaluated on the interfaces can, similarly to Section 6.2.1, be expressed using transformation coefficients $\alpha_{i,n}$ for the separate Fourier modes. Because the solution of Laplace's equation in the inner subdomain is determined by boundary conditions on two interfaces, the subindex-notation of $\alpha_{i,n}$ needs further specification for this subdomain as illustrated for $\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_1}$ and $\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_2}$:

$$\left.\frac{\partial\tilde{\phi}_2}{\partial x}\right|_{\Gamma_1} = \sum_{n=1}^{\infty}\alpha_{11,n}c_{1,n}\sin\left(\frac{n\pi}{h}z\right) + \sum_{n=1}^{\infty}\alpha_{12,n}c_{2,n}\sin\left(\frac{n\pi}{h}z\right) \tag{B.4}$$

and

$$\left.\frac{\partial\tilde{\phi}_2}{\partial x}\right|_{\Gamma_2} = \sum_{n=1}^{\infty}\alpha_{21,n}c_{1,n}\sin\left(\frac{n\pi}{h}z\right) + \sum_{n=1}^{\infty}\alpha_{22,n}c_{2,n}\sin\left(\frac{n\pi}{h}z\right). \tag{B.5}$$

The coefficient $\alpha_{ij,n}$ refers to the dependence of the horizontal derivative of the solution on interface i of the n-th Fourier mode of the (Dirichlet) boundary condition on interface j. Using equation (B.3) we find for the coefficients

$$\begin{cases} \alpha_{11,n} = -\frac{n\pi}{h}\tanh^{-1}\left(\frac{n\pi}{h}(x_2-x_1)\right), \\ \alpha_{12,n} = \frac{n\pi}{h}\sinh^{-1}\left(\frac{n\pi}{h}(x_2-x_1)\right), \\ \alpha_{21,n} = -\frac{n\pi}{h}\sinh^{-1}\left(\frac{n\pi}{h}(x_2-x_1)\right), \\ \alpha_{22,n} = \frac{n\pi}{h}\tanh^{-1}\left(\frac{n\pi}{h}(x_2-x_1)\right). \end{cases} \tag{B.6}$$

The solutions in the outer subdomains are determined by only one interface boundary condition just as in the two-subdomain problems. Therefore the corresponding transformation coefficients $\alpha_{i,n}$ are equiped with only two indices and follow from the analysis given in Section 6.2.1:

$$\begin{cases} \alpha_{1,n} = \frac{n\pi}{h}\tanh^{-1}\left(\frac{n\pi}{h}(x_1-x_0)\right), \\ \alpha_{2,n} = -\frac{n\pi}{h}\tanh^{-1}\left(\frac{n\pi}{h}(x_3-x_2)\right). \end{cases} \tag{B.7}$$

The iterative procedure can be considered for each Fourier mode separately and the formulation of new boundary conditions can be represented in the diagram shown below.

$$\begin{array}{cccc}
\tilde{\varphi}_{1,n} := c_{1,n}\sin\left(\frac{n\pi}{h}z\right) & & \tilde{\varphi}_{2,n} := c_{2,n}\sin\left(\frac{n\pi}{h}z\right) & \\
\swarrow & & \swarrow\quad\searrow & \\
\frac{\partial\tilde{\phi}_1}{\partial x}|_{\Gamma_1} = \alpha_{1,n}\tilde{\varphi}_{1,n} & \frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_1} = \alpha_{11,n}\tilde{\varphi}_{1,n} & \frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_2} = \alpha_{21,n}\tilde{\varphi}_{1,n} & \frac{\partial\tilde{\phi}_3}{\partial x}|_{\Gamma_2} = \alpha_{2,n}\tilde{\varphi}_{2,n} \\
 & +\alpha_{12,n}\tilde{\varphi}_{2,n} & +\alpha_{22,n}\tilde{\varphi}_{2,n} & \\
\searrow & \swarrow & \searrow & \swarrow \\
\tilde{\psi}_{1,n} = \omega_{1,N}\frac{\partial\tilde{\phi}_1}{\partial x}|_{\Gamma_1} + (1-\omega_{1,N})\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_1} & & \tilde{\psi}_{2,n} = \omega_{2,N}\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_2} + (1-\omega_{2,N})\frac{\partial\tilde{\phi}_3}{\partial x}|_{\Gamma_2} & \\
= c'_{1,n}\sin\left(\frac{n\pi}{h}z\right) & & = c'_{2,n}\sin\left(\frac{n\pi}{h}z\right) &
\end{array}$$

The transformation of the Fourier coefficients $c_{i,n}$ to the coefficients $c'_{i,n}$ can be expressed in terms of the matrix equation

$$\begin{pmatrix} c'_{1,n} \\ c'_{2,n} \end{pmatrix} = W_N D_n \begin{pmatrix} c_{1,n} \\ c_{2,n} \end{pmatrix} \tag{B.8}$$

with

$$D_n = \begin{pmatrix} \alpha_{1,n} & 0 \\ \alpha_{11,n} & \alpha_{12,n} \\ \alpha_{21,n} & \alpha_{22,n} \\ 0 & \alpha_{2,n} \end{pmatrix} \tag{B.9}$$

and

$$W_N = \begin{pmatrix} \omega_{1,N} & 1-\omega_{1,N} & 0 & 0 \\ 0 & 0 & \omega_{2,N} & 1-\omega_{2,N} \end{pmatrix}. \tag{B.10}$$

In the odd steps of the procedure, Neumann conditions decomposed in Fourier modes $\tilde{\psi}_{1,n}$ and $\tilde{\psi}_{2,n}$, are imposed on the interfaces and the potential on the interfaces is determined. This part of the iterative procedure can be represented with a similar diagram

$$\begin{array}{cccc}
\tilde{\psi}_{1,n} := c'_{1,n}\sin\left(\frac{n\pi}{h}z\right) & & \tilde{\psi}_{2,n} := c'_{2,n}\sin\left(\frac{n\pi}{h}z\right)\left(\frac{n\pi}{h}z\right) & \\
\swarrow & & \swarrow\quad\searrow & \\
\tilde{\phi}_1|_{\Gamma_1} = \beta_{1,n}\tilde{\psi}_{1,n} & \tilde{\phi}_2|_{\Gamma_1} = \beta_{11,n}\tilde{\psi}_{1,n} & \tilde{\phi}_2|_{\Gamma_2} = \beta_{21,n}\tilde{\psi}_{1,n} & \tilde{\phi}_3|_{\Gamma_2} = \beta_{2,n}\tilde{\psi}_{2,n} \\
 & +\beta_{12,n}\tilde{\psi}_{2,n} & +\beta_{22,n}\tilde{\psi}_{2,n} & \\
\searrow & \swarrow & \searrow & \swarrow \\
\tilde{\varphi}_{1,n} = \omega_{1,D}\tilde{\phi}_1|_{\Gamma_1} + (1-\omega_{1,D})\tilde{\phi}_2|_{\Gamma_1} & & \tilde{\phi}_{2,n} = \omega_{2,D}\tilde{\phi}_2|_{\Gamma_2} + (1-\omega_{2,D})\tilde{\phi}_3|_{\Gamma_2} &
\end{array}$$

Similar to the definition of the matrices D_n and W_N we now define the matrices N_n and W_D consisting of the coefficients $\beta_{ij,n}$ and $\omega_{i,D}$ respectively. The transformation can be written as

$$\begin{pmatrix} c_{1,n} \\ c_{2,n} \end{pmatrix} = W_D N_n \begin{pmatrix} c'_{1,n} \\ c'_{2,n} \end{pmatrix}. \tag{B.11}$$

The coefficients $\beta_{ij,n}$ are related by the coefficients $\alpha_{ij,n}$ through

$$\begin{pmatrix} \beta_{11,n} & \beta_{12,n} \\ \beta_{21,n} & \beta_{22,n} \end{pmatrix} = \begin{pmatrix} \alpha_{11,n} & \alpha_{12,n} \\ \alpha_{21,n} & \alpha_{22,n} \end{pmatrix}^{-1} \tag{B.12}$$

because calculating the set of solutions $(\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_1}, \frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_2})$ from the set $(\tilde{\phi}_2|_{\Gamma_1}, \tilde{\phi}_2|_{\Gamma_2})$ is the inverse operation of calculating $(\tilde{\phi}_2|_{\Gamma_1}, \tilde{\phi}_2|_{\Gamma_2})$ from $(\frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_1}, \frac{\partial\tilde{\phi}_2}{\partial x}|_{\Gamma_2}.)$ Working out the inverse matrix operation yields

$$\begin{pmatrix} \beta_{11,n} & \beta_{12,n} \\ \beta_{21,n} & \beta_{22,n} \end{pmatrix} = \left(\frac{n\pi}{h}\right)^{-2} \begin{pmatrix} -\alpha_{22,n} & \alpha_{12,n} \\ \alpha_{21,n} & -\alpha_{11,n} \end{pmatrix}. \tag{B.13}$$

The interface boundary conditions $\tilde{\phi}_{1,n}^{(k)}$ and $\tilde{\phi}_{2,n}^{(k)}$ at the k-th step in the iterative procedure are transformed by the operations illustrated in the above two diagrams to new interface boundary conditions $\tilde{\phi}_{1,n}^{(k+2)}$ and $\tilde{\phi}_{2,n}^{(k+2)}$. This is expressed by the transformation matrix $A_n^{(k)}$ defined in equation (6.14) which can be found by writing

$$A_n^{(k)} = W_D^{(k)} \cdot N_n \cdot W_N^{(k)} \cdot D_n. \tag{B.14}$$

If the weighting factors are kept fixed during the iterative process, then the matrix $A_n^{(k)} = A_n$ is independent of k and convergence is determined by the eigenvalues of the matrix A_n as explained in Section 6.2.2. Expressions for the eigenvalues can be found by working out the matrix equation (B.14) and include the subdomain lengths $x_1 - x_0$, $x_2 - x_1$ and $x_3 - x_2$, and the weighting factors $\omega_{1,N}$, $\omega_{1,D}$, $\omega_{2,N}$ and $\omega_{2,D}$. We illustrate this next for the special case of subdomains of equal length l and weighting factors $\omega_{1,N} = \omega_{1,D} = \omega_{1,N} = \omega_{1,D} = \frac{1}{2}$.

All coefficients $A_{ij,n}$ of the matrix A_n contain products $\alpha_*\beta_*$ in which the factors $\frac{n\pi}{h}$ and $\frac{n\pi}{h}^{-1}$ cancel. For the case mentioned above we then find

$$A_n = \begin{pmatrix} -\frac{1}{4}S^{-2} & -\frac{1}{4}S^{-1}T^{-1} + \frac{1}{4}S^{-1}T \\ -\frac{1}{4}S^{-1}T^{-1} + \frac{1}{4}S^{-1}T & -\frac{1}{4}S^{-2} \end{pmatrix} = -\frac{1}{4}S^{-2} \begin{pmatrix} 1 & C^{-1} \\ C^{-1} & 1 \end{pmatrix} \tag{B.15}$$

in which $S = \sinh(\frac{n\pi}{h}l)$, $T = \tanh(\frac{n\pi}{h}l)$ and $C = \cosh(\frac{n\pi}{h}l)$. Two different eigenvalues $\epsilon_{1,n}$, $\epsilon_{2,n}$ can be found, equal to

$$\epsilon_{1,n} = -\frac{1}{4C(C+1)} \tag{B.16}$$

and

$$\epsilon_{2,n} = -\frac{1}{4C(C-1)}. \tag{B.17}$$

In order to satisfy both $|\epsilon_{1,n}| < 1$ and $|\epsilon_{2,n}| < 1$, the parameter C has to satisfy

$$C > \frac{1}{2} + \frac{1}{2}\sqrt{2} \tag{B.18}$$

and consequently

$$\frac{l}{h} > \frac{1}{n\pi}\text{arccosh}\left(\frac{1}{2}+\frac{1}{2}\sqrt{2}\right). \tag{B.19}$$

In order to have convergence for all Fourier modes, l/h has to satisfy

$$\frac{l}{h} > \pi^{-1}\text{arccosh}\left(\frac{1}{2}+\frac{1}{2}\sqrt{2}\right) \approx 0.2. \tag{B.20}$$

This condition is given in equation (6.18).

B.2 Multi-subdomain problems

For the case of M subdomains, $M > 3$, the iterative process can be formulated similarly to the three-subdomain case. The matrices D_n and N_n have size $2(M-1)$ by $M-1$ and the matrices W_N and W_D have size $M-1$ by $2(M-1)$. In the general case M subdomains lengths and $2M$ weighting factors are involved. To simplify the analysis we again consider subdomains of equal length l and take all weighting factors equal to $\frac{1}{2}$.

We split up the matrix A_n in $\Phi_n = W_N D_n$ and $\Psi_n = W_D N_n$. These matrices are three-diagonal matrices which can be written as

$$\Phi_n = \frac{1}{2}S^{-1}\left(\frac{n\pi}{h}\right)\begin{pmatrix} 0 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 0 \end{pmatrix} \tag{B.21}$$

and

$$\Psi_n = -\frac{1}{2}S^{-1}\left(\frac{n\pi}{h}\right)^{-1}\begin{pmatrix} -C^{-1} & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & -C^{-1} \end{pmatrix} \tag{B.22}$$

Therefore the matrix A_n is a five-diagonal matrix. Upper bounds for its eigenvalues can be found using Gerschgorin's theorem for the eigenvalues of Φ_n and Ψ_n. All the eigenvalues of Φ_n are contained in the circle in the complex plane with center $(0,0)$ and radius $\frac{1}{2}S^{-1}\left(\frac{n\pi}{h}\right)$. Because $C^{-1} \leq 1$ we find the same circle for the eigenvalues of Ψ_n but with radius $\frac{1}{2}S^{-1}\left(\frac{n\pi}{h}\right)^{-1}$. So, for the eigenvalues $\epsilon_{i,n}$ of A_n we find

$$|\epsilon_{i,n}| < \frac{1}{4}S^{-2},\ i = 1,\ldots,M-1. \tag{B.23}$$

We have $\frac{1}{4}S^{-2} < 1$ if

$$\frac{nl}{h} > \pi^{-1}\text{arcsinh}\left(\frac{1}{2}\right) \approx 0.15. \tag{B.24}$$

This condition is satisfied for all Fourier modes if it is satisified for the largest Fourier mode ($n = 1$).

B.3 D/*-*/N scheme

In Chapter 5 the D/*-*/N scheme was mentioned, also known as the Neumann-Dirichlet preconditioner. The iterative process of this scheme can be analyzed in the same way as the D/D-N/N scheme.

The Laplace problems in the subdomains are the same as in the D/D-N/N scheme and therefore the coefficients α and β can be used to describe the D/*-*/N scheme as well. Only the order in which the Laplace problems are solved and the way new boundary conditions are formulated is different. We represent the D/*-*/N scheme by the following diagram for a single Fourier mode:

$$\begin{array}{c|c|c|}
 & \tilde{\varphi}_n^{(k)} = c_n \sin\left(\frac{n\pi}{h}z\right) & \\
 & \swarrow & \\
\left|\frac{\partial\tilde{\phi}_1}{\partial x}|_\Gamma = \alpha_1\tilde{\varphi}_n^{(k)}\right. & \| & * \\
 & \searrow & \\
 & \tilde{\psi}_n^{(k+1)} = \alpha_1\tilde{\varphi}_n^{(k)} & \\
 & \| & \searrow \\
* & \| & \tilde{\phi}_2|_\Gamma = \beta_2\tilde{\psi}_n^{(k+1)} \\
 & \| & \swarrow \\
\multicolumn{3}{c}{\tilde{\varphi}_n^{(k+2)} = \omega\beta_2\tilde{\psi}_n^{(k+1)} + (1-\omega)\tilde{\varphi}_n^{(k)}} \\
\multicolumn{3}{c}{= (1-\omega(1-\beta_2\alpha_1))\tilde{\varphi}_n^{(k)}}
\end{array}$$

The amplification factor ϵ_n is found to be

$$\epsilon_n = 1 - \omega\left(1 + \frac{\tanh\left(\frac{n\pi}{h}b\right)}{\tanh\left(\frac{n\pi}{h}a\right)}\right). \tag{B.25}$$

If $a = b$ and $\omega = \frac{1}{2}$ we have $\epsilon_n = 0$ for all modes and Laplace's equation is solved immediately. If $a \neq b$, ω can be chosen such that for one Fourier mode $\hat{n}$, the amplification factor $\epsilon_{\hat{n}} = 0$. Other modes then have a non-zero amplification factor. For the convergence properties for various configurations see Funaro et al. [27].

Bibliography

[1] Agoshkov, V.I., (1988), Poincaré-Steklov operators and domain decomposition methods in finite-dimensional spaces, in: *First Int. Symp. Domain decomposition methods for partial differential equations, edited by R. Glowinski e.a., G.H. Golub, G.A. Meurant and J. Périaux, Eds.)*, SIAM (Philadelphia, PA), pp. 73-112.

[2] Alessandrini, B., and Delhommeau, G., (1994), Simulation of three-dimensional unsteady viscous free surface flow around a ship model, *Int. J. for Num. Meth. in Fluids*, **19**, pp. 321-342.

[3] Barnes, T., (1996), *The generation of low-frequency water waves on beaches*, Ph.D. thesis, University of Bristol, United Kingdom.

[4] Battjes, J.A., (1994), Shallow water wave modelling, in *Proc. Symp. on Waves - Physical and numerical modelling*, **1**, pp. 1-23.

[5] Beji, S., and Battjes, J.A., (1993), Experimental investigation of wave propagation over a bar, *Coast. Engng.*, **19**, pp. 151-162.

[6] Benney, D.J., and Newell, A.C., (1967), The propagation of nonlinear wave envelopes, *J. Math. Phys.*, **46**, pp. 133-139.

[7] Berkhoff, J.W.C., (1982), Refraction and diffraction of water waves, wave deformation by a shoal; comparison between computations and measurements, Delft Hydraulics, report W154-VIII, The Netherlands.

[8] Berkvens, P.J.F., and Zandbergen, P.J.,(1996), Nonlinear reaction forces on oscillating bodies by a time-domain panel method, *J. of Ship Research*, **40**, No. 4, pp. 288-302.

[9] Berkvens, P.J.F., to appear, *Floating bodies interacting with water waves; simulations by a time-domain panel method*, Ph.D. thesis, University of Twente, Enschede, The Netherlands.

[10] Betts, P.L., and Mohamad, T.T., (1982), Water waves: a time-varying unlinearized boundary element approach, *Proc. 4th Int. Symp. on Finite Element Methods in Flow Problems*, Tokyo, pp. 923-929.

[11] Bourgat, J.F., Glowinski, R., Le Tallec, P., and Vidrascu, M., Variational formulation and algorithm for trace operator in domain decomposition calculations, in: *Proc. 2nd Int. Conf. on Domain decomposition methods for partial differential equations, SIAM, Philadelphia, 1989*, edited by Chan, T.F. e.a., pp. 3-16.

[12] Broeze, J., and Romate, J.E., (1992), Absorbing boundary conditions for free surface wave simulations with a panel method, *J. Comp. Phys.*, **99**, No. 1, pp. 146-158.

[13] Broeze, J., (1993), *Numerical modelling of nonlinear free surface waves with a 3D panel method*, Ph.D. thesis, University of Twente, Enschede, The Netherlands.

[14] Broeze, J., van Daalen, E.F.G., and Zandbergen, P.J., (1993), A three-dimensional panel method for nonlinear free surface waves on vector computers, *Computational Mechanics*, **13**, pp. 12-28.

[15] Chan, T.F., Glowinski, R., Périaux, J., and Widlund, O.B., eds., *Proc. 2nd Int. Conf. on Domain decomposition methods for partial differential equations, SIAM, Philadelphia, 1989.*

[16] Chu, V.H., and Mei, C.C., (1970), On slowly-varying Stokes waves, *J. Fluid Mech.*, **41**, pp. 873-887.

[17] Cooker, M.J., Peregrine, D.H., Vidal, C., and Dold, J.W., (1990), The interaction between a solitary wave and a submerged semicircular cylinder, *J. Fluid Mech.*, **215**, pp. 1-22.

[18] van Daalen, E.F.G., (1993), *Numerical and theoretical studies of water waves and floating bodies*, Ph.D. thesis, University of Twente, Enschede, The Netherlands.

[19] Davey, A., and Stewartson, K.,(1974), On the development of packets of surface gravity waves moving over an uneven bottom, *Proc. Royal Soc. London*, **A338**, pp 101-110.

[20] Dean, R.G., (1965), Stream function representation of nonlinear ocean waves, *J. Geophys. Res.*, **70**, No. 18, pp. 4561-4572.

[21] Dean, R.G., and Sharma, J.N., (1981), Simulation of wave systems due to nonlinear directional spectra, *Proc. Int. Symp. Hydrodynamics in Ocean Engng., The Norwegian Inst. of Technol.*, **2**, pp. 1211-1222.

[22] Dingemans, M.W., Petit, H.A.H., Meijer, Th.J.G.P., and Kostense, J.K., (1991), Numerical evaluation of the third-order evolution equations for

weakly nonlinear water waves propagating over uneven bottoms, in: *Computer Modelling in Ocean Engineering 91*, Barcelona, pp. 361-370.

[23] Dingemans, M.W., (1997), *Water wave propagation over uneven bottoms*, World Scientific Publ. Singapore. (1016 pp.)

[24] Dold, J.W., (1992), An efficient surface-integral algorithm applied to unsteady gravity waves, *J. Comp. Phys.*, **103**, pp. 90-115.

[25] Fenton, J.D., and Kennedy, A.B., (1996), Fast methods for computing the shoaling of nonlinear waves, in: *Proc. 25th Int. Conf. on Coastal Engng.*, Orlando, **1**, pp. 1130-1143.

[26] Ferrant, P., (1995), Time domain computation of nonlinear diffraction loads upon three dimensional floating bodies, *Proc. 5th Intl. Offshore and Polar Engng. Conf.*, The Hague, The Netherlands, **3**, pp. 280-288.

[27] Funaro, D., Quarteroni, A., and Zanolli, P., (1988), An iterative procedure with interface relaxation for domain decomposition methods, *SIAM J. Numer. Anal.*, **25**, pp. 1213-1236.

[28] Geist, A., *et al*, *PVM: Parallel Virtual Machine - A Users' Guide and Tutorial for Networked Parallel Computing*, The MIT Press, London.

[29] Givoli, D., (1991), Non-reflecting boundary conditions, *J. Comp. Phys.*, **94**, pp. 1-29.

[30] Glowinski, R., Golub, G.H., Meurant, G.A., and Périaux, J., eds., (1988), *Proc. First Int. Conf. Domain Decomposition Methods for Partial Differential Equations, SIAM, Philadelphia.*

[31] de Haas, P.C.A., Broeze, J., and Streng, M., (1996), Domain decomposition and parallel computing in a numerical method for nonlinear water waves, *Proc. of the EUROSIM 1996 Int. Conf.*, eds. Dekker, L., Smit, W., and Zuidervaart, J.C., Delft, The Netherlands, pp. 437-444.

[32] de Haas, P.C.A., and Zandbergen, P.J., (1996), The application of domain decomposition to time-domain computations of nonlinear water waves with a panel method, *J. Comp. Phys.*, **129**, No. 2, pp. 332-344.

[33] de Haas, P.C.A., Dingemans, M.W., and Klopman, G., (1996), Simulation of free long-wave generation due to uneven bottoms, in: *Proc. 25th Int. Conf. on Coastal Engng.*, Orlando, **1**, pp. 165-174.

[34] Hackbusch, W., (1995), *Integral equations. Theory and Numerical treatment*, Birkhäuser.

[35] Higdon, R.L., (1986), Absorbing boundary conditions for difference approximations to the multi-dimensional wave equation, *Math. Comp.*, **47**, pp. 437-459.

[36] Higdon, R.L., (1987), Numerical absorbing boundary conditions for the wave equation, *Math. Comp.*, **49**, pp. 65-90.

[37] Hsiao, G.C., and Wendland, W.L., (1991), Domain decomposition in boundary element methods, in: *Proc. 4th Int. Symp. Domain Decomposition Methods for Partial Differential Equations (edited by Glowinski, R., et al.)*, SIAM, Philadelphia, pp. 41-49.

[38] Jaswon, M.A., and Symm, G.T., (1977), *Integral equation methods in potential theory and elastostatics*, Academic Press, London.

[39] Kelley, W.G., and Peterson, A.C., (1991), *Difference Equations, An Introduction with Applications*, Academic Press, Boston.

[40] Kennedy, A.B., and Fenton, J.D., (1996), A fully nonlinear 3-D method for the computation of wave propagation, in: *Proc. 25th Int. Conf. on Coastal Engng.*, Orlando, **1**, pp. 1102-1115.

[41] Kennedy, A.B., to appear, *The propagation of water waves over varying topography*, Ph.D. thesis, Monash University, Clayton, Australia.

[42] Kim, C.H., (1995), Recent progress in numerical wave tank research: a review, *Proc. 5th Int. Offshore and Polar Engng. Conf.*, The Hague, The Netherlands, **3**, pp. 1-9.

[43] Kim, S.K., Liu, P.L.F., and Liggett, J.A., (1983), Boundary integral equation solutions for solitary wave generation, propagation and run-up, *Coast. Engng.*, **7**, pp. 299-317.

[44] Lamb, H., (1945), *Hydrodynamics*, Dover, New York.

[45] Le Tallec, P., (1994), Domain decomposition methods in computational mechanics, *Comp. Mech. Adv.*, **1**, pp. 121-220.

[46] Liu, P.L.F., and Dingemans, M.W., (1989), Derivation of the third-order evolution equations for weakly nonlinear water waves propagating over uneven bottoms, *Wave Motion*, **11**, pp. 41-64.

[47] Longuet-Higgins, M.S., and Stewart, R.W., (1962), Radiation stress and mass transport in gravity waves with applications to "surf beats", *J. Fluid Mech.*, **13**, pp. 481-504.

[48] Longuet-Higgins, M.S. and Stewart, R.W., (1964), Radiation stresses in water waves: a physical discussion with applications, *Deep Sea Res.*, **11**, pp. 529-562.

[49] Luth, H.R., Klopman, G., and Kitou, N., (1994), Project 13G: Kinematics of waves breaking partially on an offshore bar; LDV measurements for waves with and without a net onshore current. *Delft Hydraulics* Report H1573, 40 pp.

[50] Luth, H.R., (1994), Active wave absorption. Experimental verification in "Scheldegoot" facility. *Delft Hydraulics* Report H1222, part I.

[51] Marini, L.D., and Quarteroni, A., (1989), A relaxation procedure for domain decomposition methods using finite elements, *Numer. Math.*, **55**, pp. 575-598.

[52] Mei, C.C., (1983), *The applied dynamics of ocean surface waves*, Wiley, New York.

[53] Message Passing Interface Forum. Mpi: A message-passing interface standard. Computer Science Dept. Technical Report CS-94-230, University of Tennessee, Knoxville, TN, April 1994.

[54] Nayfeh, A.H., (1981), *Introduction to perturbation techniques*, J. Wiley and Sons, New York.

[55] Nestegård, A., (1994), Comparative study of fully nonlinear wave simulation programs, Det Norske Veritas Research AS Technical Report No. 94-2041.

[56] Petit, H.A.H., Klopman, G., and Otta, A.K., (1994), Laboratory wave generation. A second-order theory for regular and irregular waves in wave channels. *Delft Hydraulics* Report H1222.

[57] Rienecker, M.M., and Fenton, J.D., (1981), A Fourier approximation method for steady water waves, *J. of Fluid Mech.*, **104**, pp. 119-137.

[58] Romate, J.E., (1989), The numerical simulation of nonlinear gravity waves in three dimensions using a higher order panel method, Ph.D. thesis, University of Twente, Enschede, The Netherlands.

[59] Romate, J.E., and Zandbergen, P.J., (1989), Boundary integral equation formulations for free-surface flow problems in two and three dimensions, *Comp. Mech.*, **4**, pp. 276-282.

[60] Romate, J.E., (1992), Absorbing boundary conditions for free surface waves, *J. Comp. Phys.*, **99**, No. 1, pp. 135-145.

[61] Romate, J.E., (1993), On the use of conjugate gradient-type methods for boundary integral equations, *Comp. Mech.*, **13**, pp. 1-19.

[62] Sarpkaya, T., and Isaacson, M., (1981), *Mechanics of wave forces on offshore structures*, Van Nostrand Reinhold, New York.

[63] Schäffer, H.A., (1996), Second-order wavemaker theory for irregular waves, *Ocean Engng.*, **23**, No. 1, pp. 47-88.

[64] Schwarz, H.A., (1870), *Gesammelte Mathematische Abhandlungen*, Volume 2, Springer, Berlin (1890). First published in Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich, **15**, pp. 272-286.

[65] Sharma, J.N., (1979), *Development and evaluation of a procedure for simulating a random directional second-order sea surface and associated wave forces*, Ph.D. thesis, University of Delaware.

[66] Sonneveld, P., (1984), CGS, a fast Lanzcos solver for nonsymmetric linear systems, Delft University of Technology, Rep. 84-16, Dep. of Math. and Informatics, Delft.

[67] Tan, K.H., (1995), *Local coupling in domain decomposition*, Ph.D. thesis, University of Utrecht, The Netherlands.

[68] Tanizawa, K., (1995), A nonlinear simulation method of 3-D body motions in waves, *Journ. of The Soc. of Naval Architects of Japan*, **178**, pp. 179-191.

[69] Wang, P., Yao, Y., and Tulin, M., (1995), An efficient numerical tank for nonlinear water waves, based on the multi-subdomain approach with BEM, *Int. J. Numer. Meth. Fluids*, **20**, pp. 1315-1336.

[70] Whitham, G.B., (1974), *Linear and nonlinear waves*, Wiley-Interscience, New York.

[71] Wu, G.X., and Eatock Taylor, E., (1995), Time stepping solutions of the two-dimensional nonlinear wave radiation problem, *Ocean Engng.*, **22**, No.8, pp. 785-798.

[72] Yang, C., and Ertekin, R.C., (1992), Numerical simulation of nonlinear wave diffraction by a vertical cylinder, *Journ. of Offshore Mech. and Arctic Engng.*, **114**, pp. 36-44.

Index

Curriculum vitae

De schrijver van dit proefschrift werd geboren op 27 augustus 1966 te Someren. In 1984 behaalde hij het atheneum-diploma aan het Peelland College te Deurne. Daaropvolgend studeerde hij wiskunde aan de Technische Universiteit Eindhoven. Deze studie sloot hij in 1990 af met een afstudeerwerk met de titel 'Fraktionele integratie - Fraktionele differentiatie - Snelafnemende funkties'. Aansluitend volgde hij op dezelfde universiteit de opleiding voor eerstegraads leraar wiskunde en behaalde de onderwijsbevoegdheid hiervoor in 1992. In deze tijd werkte hij als docent wiskunde aan Gymnasium Beekvliet, te Sint Michielsgestel en aan het Christelijk Gymnasium Utrecht. In februari 1993 begon hij zijn werkzaamheden als onderzoeker in opleiding in dienst van NWO bij de Stichting voor Technische Wetenschappen aan het in dit proefschrift beschreven onderzoek. Het werk stond onder supervisie van prof. Zandbergen, Universiteit Twente en werd uitgevoerd bij het Waterloopkundig Laboratorium.