UNIVERSITY OF OSLO
Department of informatics

A finite element method for fully nonlinear water waves

X. Cai
H. P. Langtangen
B. F. Nielsen
A. Tveito

Preprint 1996-3

March 26, 1996

A finite element method for fully nonlinear water waves*

Xing Cai[1], Hans Petter Langtangen[2], Bjørn Fredrik Nielsen[3] and Aslak Tveito[4]

[1,3,4] *Department of Informatics, University of Oslo, P.O. Box 1080 Blindern, N-0316 Oslo, Norway. email: xingca@ifi.uio.no, bjornn@ifi.uio.no, aslak@ifi.uio.no*

[2] *Mechanics Division, Department of Mathematics, University of Oslo, P.O. Box 1053 Blindern, N-0316 Oslo, Norway. email: hpl@math.uio.no*

Abstract

We introduce a numerical method for a mathematical model of fully nonlinear water waves. The method is based on a transformation of the dynamic water volume onto a fixed domain, thus regridding at each time step is avoided. The transformation introduces an elliptic boundary value problem which is solved by a preconditioned conjugate gradient method. Numerical experiments are presented and they show nice convergence properties of the iterative solver as well as convergence of the entire solution towards a reference solution computed by another scheme.

1 Introduction

The purpose of this paper is to introduce a new numerical method for solving fully nonlinear free surface waves modelled by standard potential theory. The main application area of this method is typically the calculation of wave forces on marine installations.

These types of wave problems have traditionally been addressed by linear theory or second order perturbation theory to estimate weak nonlinear effects. The governing Laplace equation has almost exclusively been solved by boundary element methods (BEM), usually with piecewise constant elements. Important contributions to solving the fully nonlinear wave problems using BEM are briefly reviewed by Ferrant [11]. In linear or second order theory one can derive relevant Green functions for the integral equation such that it is only necessary to discretize the surface of the marine installations, and at most a smaller part of the mathematically flat water surface in the vicinity of the installations. Hence, the advantage of integral equations and BEM over solving the Laplace equation directly in the water volume is obvious. Nevertheless, in the fully nonlinear case

*This work has been supported by the Research Council of Norway (NFR) under grant no. 463.95/002 and 110673/420 (*Numerical Computations in Applied Mathematics*).

the integral equation approach requires discretization of the complete, moving boundary of the water volume. In this case we will later in this paper give heuristic arguments which show that finite element or finite difference discretization of the Laplace equation in the water volume may be asymptotically more efficient than BEM. New preconditioning techniques for Laplace equation solvers are the main reason for this result.

Solution of the Laplace equation arising in the model of fully nonlinear water waves around marine installations by the finite element method (FEM) is not very common. Actually, these types of volume methods have received surprisingly little attention in the literature. To our knowledge, Eatock Taylor [19] is the pioneer of applying finite element methods (FEM) in this respect. Equally novel is the spline method of Even Mehlum described in [15] in which smooth spline functions in connection with spectral expansion are used to solve the Laplace equation.

The method presented here is based on a finite element discretization of the Laplace equation in the water volume, while the free surface boundary conditions are discretized by standard finite difference techniques. The choice of finite elements instead of, e.g., finite differences for the Laplace equation is mainly motivated by the inherent flexibility of finite elements with respect to adaptive grids and higher order approximations, that is, p and $h-p$ extensions of FEM, although these extensions will not be discussed herein. Use of finite elements to treat complicated geometries, due to the bottom topography and, e.g., marine installations, is of less importance since we will present a domain imbedding approach that handles the geometry aspect of the problem without the need for sophisticated gridding techniques. The possibly complicated geometry of the free surface normally requires re-gridding of the water volume at each time step, but we avoid this by a time dependent mapping of the water volume onto a simple stationary solution domain for the Laplace equation.

Most of the computational effort in this numerical method is devoted to the solution of the Laplace equation. We use efficient preconditioners in combination with the conjugate-gradient (CG) method to achieve an optimal solution method, that is, the total cost of a simulation is proportional to the product of the number of grid points and the number of time steps. The implementation of efficient preconditioners is significantly simplified since the mapping of the water volume and the domain imbedding technique allow the solution domain for the Laplace equation to be, e.g., a box with a uniform grid.

In the following sections, we first outline the well known mathematical model of nonlinear water waves. Then we present a brief discussion of the choice of finite element methods versus boundary element methods for solving the Laplace equation. This discussion forms the background for deriving a modified system of governing equations. Thereafter a new, efficient method for solving the Laplace equation is described. Based on this, we formulate the numerical algorithm for the dynamic problem. Finally, we evaluate the performance of the algorithm in several two- and three-dimensional applications. The associated simulation programs have been developed using `Diffpack` (see [9, 17]).

2 The governing equations

2.1 Outline of the mathematical model

Let $(\bar{x}, \bar{y}, \bar{z})$ be the spatial coordinates and let t denote time. We make the standard assumption that the wave induced velocity field is divergence free and irrotational. Introducing the velocity potential $\varphi(\bar{x}, \bar{y}, \bar{z}, t)$, the governing partial differential equation, arising from mass conservation in the water volume, is the Laplace equation $\nabla^2\varphi = 0$. Besides the velocity potential, the free surface $\bar{z} = \eta(\bar{x}, \bar{y}, t)$ is the other primary unknown of the problem. The motion of the free surface is governed by the kinematic and dynamic boundary conditions:

$$\eta_t + \varphi_{\bar{x}}\eta_{\bar{x}} + \varphi_{\bar{y}}\eta_{\bar{y}} - \varphi_{\bar{z}} = 0, \quad (1)$$

$$\varphi_t + \frac{1}{2}(\varphi_{\bar{x}}^2 + \varphi_{\bar{y}}^2 + \varphi_{\bar{z}}^2) + g\eta = 0. \quad (2)$$

Here g is the gravitational acceleration in the negative $\bar{z}$-direction, and subscripts denote derivatives, for example, $\eta_t \equiv \partial\eta/\partial t$. Equation (1) guarantees that no mass transfers through the free surface, while equation (2) is a force (pressure) balance at the surface. At solid boundaries, the normal derivative of φ must equal the normal velocity of the boundary. However, in this paper we will study problems with fixed solid boundaries, i.e. $\partial\varphi/\partial n = 0$ at these boundaries. We refer to, e.g. Whitham [20] for a derivation of the equations above.

The water volumes of interest in this paper can be written on the form

$$\overline{\Omega}(t) = \{(\bar{x}, \bar{y}, \bar{z}) \,|\, (\bar{x}, \bar{y}) \in \Omega_{\bar{x}\bar{y}} \backslash \Omega_S,\ -H \le \bar{z} \le \eta(\bar{x}, \bar{y}, t)\}. \quad (3)$$

Here, $\Omega_{\bar{x}\bar{y}}$ is a two-dimensional computational domain containing a solid obstacle with $\bar{x}\bar{y}$ cross section Ω_S. Both $\Omega_{\bar{x}\bar{y}}$ and Ω_S are stationary with respect to time. An example of a solution domain of this type is shown in Figure 4 on page 19. The solid obstacle may represent an oil platform leg. To avoid the influence of radiation conditions on the evaluation of the proposed numerical algorithm, we will restrict the numerical examples to a water tank such that the boundaries of $\Omega_{\bar{x}\bar{y}}$ are solid. For simplicity, the depth H is considered as constant, but the treatment of a space-time dependent depth function will follow directly from our treatment of the free surface.

2.2 Boundary elements versus finite elements

In modelling water waves, most numerical methods are based on a certain operator splitting. At a specific time level, equations (1) and (2) are solved at the two-dimensional free surface to determine the new shape of the three-dimensional water volume $\overline{\Omega}$, then the Laplace equation is solved in $\overline{\Omega}$. Of course, the most CPU-time consuming steps are the discretization of the time dependent solution domain $\overline{\Omega}$ and the solution of the Laplace equation therein.

There is a strong tradition in marine hydrodynamics for solving the Laplace equation by boundary element type techniques instead of a straightforward finite element discretization in the volume. The reason for this is that the number of unknowns in the resulting linear system is reduced. However, modern preconditioning techniques for elliptic boundary value problems have led us to reconsider this point of view. For a general description and an analysis of boundary element methods and finite element methods we would like to refer to the books of Becker [2] and Ciarlet [8].

Let us give a very heuristic comparison between FEM and BEM for solving the Laplace equation in $\overline{\Omega}$. For simplicity, we consider the case of a square domain without any internal obstacle and an associated uniform grid with n grid points in each direction. Thus the number of unknowns is roughly as given by Table 1.

	FEM	BEM
2D	n^2	n
3D	n^3	n^2

Table 1: The number of unknowns for FEM and BEM discretizations of the Laplace equation.

Here, the system matrix associated with the finite element discretization is sparse. Using linear elements, we get five nonzero diagonals in 2D and seven in 3D. Higher order elements introduce more nonzero diagonals, but it is always a fixed number independent of n. On the other hand, the matrices arising from a boundary element discretization are dense.

The matrix-systems can be solved by both direct methods and iterative methods. For the problems under consideration, iterative methods are known to be much faster so we assume that both the BEM and the FEM systems are solved by a conjugate-gradient-type iterative method. It is well known that BEM matrices are well conditioned, hence an iterative method is expected to converge in a finite number of iterations I_B independent of n. Since the computational cost of one matrix-vector multiplication for BEM is the square of the number of unknowns, the computational cost of solving the BEM-system iteratively is $\mathcal{O}(n^2)$ in 2D and $\mathcal{O}(n^4)$ in 3D.

The cost of one matrix-vector multiplication in FEM is $\mathcal{O}(N_F)$ where N_F is the number of nonzeros of the FEM-matrix (recall that N_F is of the same order as the number of unknowns). Without any preconditioning, the FEM-matrix is well known to have a spectral condition number $\kappa_F \sim \mathcal{O}(h^{-2})$, where $h \approx 1/n$ denotes the global mesh size. Since the number of CG iterations is of order $\mathcal{O}(\kappa_F^{1/2})$ (see Axelsson [1]), this leads to $\mathcal{O}(n)$ iterations and a computational cost of $\mathcal{O}(n^3)$ for 2D problems and $\mathcal{O}(n^4)$ for 3D problems.

For the last 20 years there has been a lot of research activity connected to the development of preconditioners for elliptic boundary value problems. A recent survey of this field is provided by Bruaset [5]. One particularly simple scheme is known as "The modified incomplete LU-factorization" (MILU). It was introduced by Gustafsson [13] and will be further discussed below. It is well known that this preconditioning technique reduces the spectral condition number of the FEM matrix to $\mathcal{O}(h^{-1})$, thus requiring $\mathcal{O}(n^{1/2})$ iterations. This leads to a computational cost of $\mathcal{O}(n^{2.5})$ in 2D and of $\mathcal{O}(n^{3.5})$ in 3D.

However, it is also possible to derive preconditioners which result in uniform condition numbers. Hence, the number of CG iterations needed to solve the discretized Laplace equation become independent of n. With these optimal preconditioners it only requires a computational cost of $\mathcal{O}(n^2)$ in 2D and $\mathcal{O}(n^3)$ in 3D for FEM. A summary of these heuristic arguments is given in Table 2.

Discretization method	FEM				BEM	
Preconditioning	MILU		Opt. precond.			
Space dimensions	2D	3D	2D	3D	2D	3D
Number of unknowns	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n)$	$\mathcal{O}(n^2)$
Iteration number	$\mathcal{O}(n^{0.5})$	$\mathcal{O}(n^{0.5})$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$	$\mathcal{O}(1)$
Computational cost	$\mathcal{O}(n^{2.5})$	$\mathcal{O}(n^{3.5})$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^3)$	$\mathcal{O}(n^2)$	$\mathcal{O}(n^4)$

Table 2: The table shows heuristic estimates of the number of iterations and of the computational cost for BEM and for FEM.

Based on these observations, we find it interesting to consider the application of FEM for solving the Laplace equation in the three-dimensional water volume using a preconditioned conjugate gradient (PCG) method.

2.3 Transformation of the physical domain

As discussed above, the solution of the Laplace equation is the most time-consuming part when modelling fully nonlinear water waves. Notice that the Laplace equation must be solved at each time step in a time dependent geometry $\overline{\Omega}(t)$ as defined in equation (3). Our approach to an efficient solution of the Laplace equation is founded on two basic demands:

1. We do not want to re-grid the water volume $\overline{\Omega}$ at each time step since this is a complicated, time-consuming process.

2. We want to solve the linear system arising from the finite element discretization by a preconditioned conjugate gradient (PCG) algorithm where the associated linear operator is uniformly well conditioned with respect to the mesh partition parameter n.

The first demand can be met by introducing a simple transformation of the water volume, in which we map $\overline{\Omega}$ onto a stationary domain $\widetilde{\Omega}$ on the form

$$\widetilde{\Omega} = \{(x,y,z) \,|\, (x,y) \in \Omega_{xy} \backslash \Omega_S,\ -H \leq z \leq 0\}$$

with boundaries fixed in time. In this way we avoid the re-gridding process at each time step. The transformation $\rho : (\bar{x},\bar{y},\bar{z}) \rightarrow (x,y,z)$ can be defined as

$$x = \bar{x}, \quad y = \bar{y}, \quad z = \left(\frac{\bar{z}+H}{f} - 1\right) H, \tag{4}$$

where $f(\bar{x}, \bar{y}, t) = \eta(\bar{x}, \bar{y}, t) + H$ and $f(\bar{x}, \bar{y}, t) \geq 0$. However, the case of $f = 0$ means a dry spot in the physical domain and will not be considered further. Hence, we assume that $|\eta(\bar{x}, \bar{y}, t)| < H$ throughout this paper. Note also that the $\bar{x}\bar{y}$-coordinates are the same as the xy-coordinates in the transformation, we thus drop notations $\bar{x}$,$\bar{y}$ and use x,y instead throughout the remaining text.

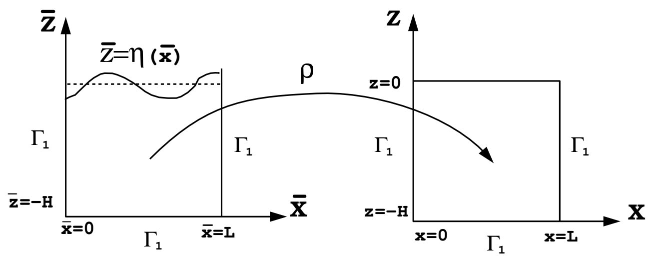

Figure 1: Transformation between physical and stationary solution domains in a two-dimensional example.

The Jacobian matrix $\boldsymbol{J}$ associated with the transformation ρ has the form

$$\boldsymbol{J}(x, y, z, t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ -\dfrac{(z+H)f_x}{f} & -\dfrac{(z+H)f_y}{f} & \dfrac{H}{f} \end{bmatrix}. \tag{5}$$

Hence, the transformation ρ is well defined and if f is positive then $\det \boldsymbol{J} > 0$. Introducing the transformation ρ in the Laplace equation, the solution domain becomes much simpler at the cost of an unknown, variable coefficient in the governing partial differential equation. The transformed equation reads

$$\nabla \cdot (\boldsymbol{K} \nabla \varphi) = 0 \quad \text{in } \widetilde{\Omega}, \tag{6}$$

where the coefficient matrix $\boldsymbol{K}(x, y, z, t)$ is given by

$$\begin{aligned} \boldsymbol{K}(x, y, z, t) &= \frac{1}{\det \boldsymbol{J}} \boldsymbol{J}\boldsymbol{J}^T \\ &= \frac{1}{H} \begin{bmatrix} f & 0 & -(z+H)f_x \\ 0 & f & -(z+H)f_y \\ -(z+H)f_x & -(z+H)f_y & \dfrac{H^2 + (z+H)^2(f_x^2 + f_y^2)}{f} \end{bmatrix}. \end{aligned} \tag{7}$$

Normally, we consider the two-dimensional computational domain Ω_{xy} on the form $\Omega_{xy} = [0, L_1] \times [0, L_2]$. Hence the new three-dimensional solution domain $\widetilde{\Omega}$ has a simple box shape. However, complicated geometries occur when the water volume contains a solid obstacle. This may demand sophisticated gridding techniques. In these situations, a domain imbedding approach (also referred to as the method of fictitious domains) may be useful in overcoming this difficulty. Roughly speaking, we introduce an artificial, non-physical permeability that is unity in water and that equals a small value ϵ inside the solid obstacle. More precisely, we replace $\boldsymbol{K}$ in the governing partial differential equation (6) by $\boldsymbol{K}_\epsilon$, where

$$\boldsymbol{K}_\epsilon = \begin{cases} \epsilon \boldsymbol{I} & (x,y) \in \Omega_S, \\ \boldsymbol{K} & (x,y) \in \Omega_{xy} \backslash \Omega_S. \end{cases}$$

Here, $\boldsymbol{I}$ denotes the identity matrix.

It has been shown, see [16], that the errors in the discrete finite element approximations to φ and $\nabla\varphi$ are of order ϵ in proper norms regardless of the mesh size, as long as the element boundaries coincide with the boundaries of the obstacle. To treat complicated geometries accurately, one can hence choose ϵ small enough so that the errors due to "water flow" through the obstacle are negligible in comparison with other discretization errors.

Introducing domain imbedding, the governing partial differential equation can always be solved in the simple box-shaped computational domain

$$\Omega = [0, L_1] \times [0, L_2] \times [-H, 0]. \tag{8}$$

Hence we can choose, e.g., a uniform partition of Ω. More important, it makes the implementation of efficient preconditioners easy. This will be addressed in Section 3.

2.4 The system of governing equations

We will now list the complete initial-boundary value problem to be solved in this paper. It is convenient to introduce two new variables; the potential and the $\bar{z}$-component of particle velocity evaluated at the surface η, (see Zakharov [21])

$$F(x,y,t) = \varphi(x,y,\eta(x,y,t),t), \tag{9}$$
$$G(x,y,t) = \varphi_{\bar{z}}(x,y,\eta(x,y,t),t). \tag{10}$$

Moreover, we split the boundary of Ω into three non-overlapping components: $\partial\Omega = \Gamma_1 \cup \Gamma_2 \cup \Gamma_3$. Here, Γ_1 represents the solid walls: $x = 0, L_1$, $y = 0, L_2$, $z = -H$; Γ_2 is the free surface $z = 0$ outside the obstacle ($(x,y) \notin \Omega_S$), and Γ_3 is the remaining part (the free surface "inside" the obstacle). With these new variables, we can write the complete system of partial differential equations and boundary conditions on the following form.

$$\nabla \cdot (\boldsymbol{K}_\epsilon \nabla \varphi) = 0 \quad \text{in } \Omega, \tag{11}$$

$$\frac{\partial \varphi}{\partial n} = 0 \quad \text{on } \Gamma_1, \tag{12}$$

$$\varphi = F \quad \text{on } \Gamma_2, \tag{13}$$

$$\frac{\partial \varphi}{\partial z} = 0 \quad \text{on } \Gamma_3, \tag{14}$$

$$\eta_t + F_x\eta_x + F_y\eta_y - (1+\eta_x^2+\eta_y^2)G = 0 \quad \text{on } \Gamma_2, \tag{15}$$

$$F_t + \frac{1}{2}(F_x^2+F_y^2) - \frac{1}{2}(1+\eta_x^2+\eta_y^2)G^2 + g\eta = 0 \quad \text{on } \Gamma_2, \tag{16}$$

$$\eta(x,y,0) = \eta^0 \quad \text{on } \Gamma_2, \tag{17}$$

$$F(x,y,0) = 0 \quad \text{on } \Gamma_2, \tag{18}$$

$$G(x,y,0) = 0 \quad \text{on } \Gamma_2. \tag{19}$$

The latter three conditions reflect that the surface is initially at rest with a prescribed shape $\bar{z} = \eta^0(x,y)$. Similar equations for F and G have been introduced and used by Mehlum in [15].

3 Preconditioning

It is readily seen that the system (11)-(14) is a standard, variable-coefficient Laplace-type equation with Neumann and Dirichlet boundary conditions. Its finite element formulation is straightforward and gives rise to a linear system on the form

$$\boldsymbol{A}\boldsymbol{\xi} = \boldsymbol{b}, \tag{20}$$

where $\boldsymbol{A}$ is a sparse, symmetric and positive definite matrix, and $\boldsymbol{\xi}$ is a vector of the unknown φ values at the grid points of Ω. Such linear systems can be efficiently solved by the PCG method, see e.g. Axelsson and Barker [1]. That is, instead of solving (20) explicitly, we solve the equivalent system

$$\boldsymbol{M}^{-1}\boldsymbol{A}\boldsymbol{\xi} = \boldsymbol{M}^{-1}\boldsymbol{b}, \tag{21}$$

where $\boldsymbol{M}$ is also a sparse, symmetric and positive definite matrix. The preconditioner $\boldsymbol{M}$ should be constructed such that $\boldsymbol{M}$ is spectrally close to $\boldsymbol{A}$ and such that problems on the form $\boldsymbol{M}\boldsymbol{x} = \boldsymbol{g}$ can be efficiently solved.

As mentioned above, over the last 20 years preconditioners for the efficient numerical solution of discretized second order elliptic problems have been extensively studied. The most popular methods are based on domain decomposition techniques, multigrid methods and incomplete factorizations, see e.g. Hackbusch [14] and Bruaset [5]. In this paper we will consider so-called fast solvers as preconditioners for systems on the form (20). It is proved in [7] that a very small domain imbedding parameter ϵ does not destroy the optimal convergence property of the PCG method. In fact, it is shown that the number of CG-iterations is bounded independently of ϵ and the mesh parameter n.

Recall that we have mapped the dynamic, physical domain $\overline{\Omega}(t)$ onto a stationary computational domain Ω. Hence, we get an elliptic boundary value problem posed on a rectangular domain in 2D or a box in 3D. In this case, we can use fast solvers for the Laplacian as preconditioners for the system (20), see e.g. Greenbaum [12]. That is, the preconditioner $\boldsymbol{M}$ is defined as the matrix associated with a finite element discretization of the following problem

$$\begin{gathered} \nabla^2\psi = 0 \quad \text{in } \Omega, \\ \frac{\partial\psi}{\partial n} = 0 \ \text{ on } \Gamma_1 \cup \Gamma_3, \quad \psi = 0 \ \text{ on } \Gamma_2, \end{gathered} \tag{22}$$

where we use the same discretization method and the same element type as for the system (11)-(14). In this case, it is well known that the number of CG-iterations needed to solve (21) is bounded independently of the mesh partition parameter n.

3.0.1 Convergence properties of the PCG method

Let $\boldsymbol{\xi}_m$ be the m'th approximation to the solution $\boldsymbol{\xi}$ of (20) in the CG-method applied to (21). Then the relative error satisfies

$$\frac{||\boldsymbol{\xi} - \boldsymbol{\xi}_m||_{\boldsymbol{A}}}{||\boldsymbol{\xi} - \boldsymbol{\xi}_0||_{\boldsymbol{A}}} \leq \sigma$$

in at most

$$m \sim \operatorname{int}\left(\frac{1}{2}\sqrt{\kappa(\boldsymbol{M}^{-1}\boldsymbol{A})}\ln\frac{2}{\sigma}\right) \tag{23}$$

CG-iterations, see e.g. Axelsson [1]. Here, $\sigma > 0$ is the error level, $\kappa(\boldsymbol{M}^{-1}\boldsymbol{A})$ denotes the spectral condition number of $\boldsymbol{M}^{-1}\boldsymbol{A}$ and $||\cdot||_{\boldsymbol{A}}$ denotes the so-called energy norm associated with the matrix $\boldsymbol{A}$. Since $\boldsymbol{M}$ is the matrix associated with a finite element discretization of the problem (22), it is well known that the spectral condition number of $\boldsymbol{M}^{-1}\boldsymbol{A}$ satisfies

$$\kappa(\boldsymbol{M}^{-1}\boldsymbol{A}) \leq \frac{\max\limits_{(x,y,z)}(\lambda_{\max}(\boldsymbol{K}(x,y,z,t)))}{\min\limits_{(x,y,z)}(\lambda_{\min}(\boldsymbol{K}(x,y,z,t)))},$$

see e.g. [3] and references therein. Here, $\boldsymbol{K}$ is the matrix defined in (7) and $\lambda_{\max}$ and $\lambda_{\min}$ denote the maximum and minimum eigenvalue of $\boldsymbol{K}$ at a specific time level, respectively.

For simplicity we consider this estimate closer in the 2D case, i.e. $(x,z) \in \Omega_2 = [0,L] \times [-H,0]$. The generalization to 3D is straightforward. We find for

2D cases that

$$\kappa(\boldsymbol{M}^{-1}\boldsymbol{A}) \leq \frac{\max\limits_{(x,z)}\left(r+\sqrt{r^2-4}\right)}{\min\limits_{(x,z)}\left(r-\sqrt{r^2-4}\right)},$$

where $f(x,t) = \eta(x,t) + H$ and

$$r(x,z,t) \;=\; \frac{f}{H} + \frac{H}{f} + \frac{(z+H)^2|f_x|^2}{Hf}$$

Next, if $r_{\max} = \max_{(x,z)}\ r(x,z,t) = \max_x r(x,0,t)$ denotes the maximum value of r in Ω_2 at a specific time level then

$$\kappa(\boldsymbol{M}^{-1}\boldsymbol{A}) \leq \frac{r_{\max}+\sqrt{r_{\max}^2-4}}{r_{\max}-\sqrt{r_{\max}^2-4}}. \tag{24}$$

It is readily seen that $r(x,z,t) \geq 2$ and since $|\eta(x,t)| < H$ it can be shown that

$$r_{\max} \leq \max\left(\frac{\max\limits_x f}{H} + \frac{H}{\max\limits_x f}, \frac{\min\limits_x f}{H} + \frac{H}{\min\limits_x f}\right) + \frac{H}{\min\limits_x f}\left(\max_x |f_x|\right)^2. \tag{25}$$

Hence, if $|f_x|$ is bounded then it follows from (23)-(25) that the number of CG-iterations needed to solve the preconditioned system associated with a 2D case is bounded independently of n.

In order to apply the preconditioner, $\boldsymbol{M}$ we must be able to solve systems on the form

$$\boldsymbol{M}\boldsymbol{x} = \boldsymbol{g} \tag{26}$$

efficiently. It is well known that problems on this form can be solved in $\mathcal{O}(N)$ operations, where N denotes the number of unknowns, using domain decomposition techniques or multigrid methods.

However, applying the domain imbedding technique, described in the previous section, the problem (22) is posed on a rectangular domain in 2D and a box in 3D. For such problems FFT-based fast solvers for the efficient numerical solution of the discretized Laplacian (26) have been thoroughly studied by several authors, see e.g. [6] and [10]. In general, these solution methods require $\mathcal{O}(N \log N)$ arithmetic operations. Moreover, in [7] we proved that the number of CG-iterations needed to solve (21) is bounded indepently of the domain imbedding parameter ϵ.

3.1 Numerical experiments; Simplified cases

In this section we present some examples illustrating the behavior of the domain transformation technique and the FFT-based preconditioner described above. Simulations based on the full model (11)-(19) in 2D and 3D are presented below.

We solve a typical variable-coefficient Laplace-type problem on the form (11)-(14) that arises at each time step in the numerical algorithm for the full wave problem. For simplicity, we restrict our attention to a two-dimensional computational domain $[0, L] \times [-H, 0]$ and prescribe three stationary water surface functions: $\eta(x, t) = \eta_i(x)$, $i = 1, 2, 3$, where

$$\begin{aligned}
\eta_1(x) &= 0.2x^2(\pi - x)^2, \\
\eta_2(x) &= -0.7|x - \pi/2|, \\
\eta_3(x) &= 0.7|x - \pi/2|.
\end{aligned}$$

We solve equation (11) with $\partial\varphi/\partial n = 0$ on $x = 0, L$ and $z = -H$, while at $z = 0$ we use the Dirichlet condition $\varphi(x, 0) = \varphi^{(e)}(x, \bar{z}(x, 0))$, where

$$\begin{aligned}
\varphi^{(e)}(x, \bar{z}(x, z)) &= \frac{1}{2}\Big[\sin(e^{-(\bar{z}+2\pi)}\cos x)\cosh(e^{-(\bar{z}+2\pi)}\sin x) \\
&\quad + \sin(e^{\bar{z}}\cos x)\cosh(e^{\bar{z}}\sin x)\Big].
\end{aligned}$$

One can then verify that $\varphi(x, z) = \varphi^{(e)}(x, \bar{z}(x, z))$ is the exact solution of the problem.

3.1.1 Convergence of the numerical solutions

First we investigate whether the convergence rate of the finite element method is affected by the domain transformation or the smoothness of the surface. The errors of the finite element solutions are measured in the discrete L_∞- and L_2-norms, defined by

$$||g||_{\Delta,\infty} = \max_{j=1,\ldots,N} |g_j|, \quad ||g||_{\Delta,2} = (\frac{1}{N}\sum_{j=1}^{N} |g_j|^2)^{1/2},$$

where g is a discrete finite element function, and g_j, $j = 1, \ldots, N$, denote its nodal values. Assuming an error function $E(h) = Ch^\alpha$, we can estimate the constants C and α from computer experiments. In the following, only α is estimated by comparison of subsequent experiments. Gaussian elimination is used as the equation solver, and linear elements are used for the discretization. The numerical results are shown in Table 3, where finite element solutions are denoted by $\widehat{\varphi}$. Clearly, second order convergence is obtained for all test problems. Hence, it seems that the expected order of the error is preserved by the domain transformation technique.

3.1.2 Numerical results for the PCG method

Now we want to study the number of CG-iterations needed to solve the linear system associated with the two-dimensional test problem described above. A vector $\boldsymbol{\xi}_0$ with zeroes is used as initial guess for the iterations, and as stopping criterion we have used $||\boldsymbol{b} - \boldsymbol{A}\boldsymbol{\xi}_k||_2/||\boldsymbol{b} - \boldsymbol{A}\boldsymbol{\xi}_0||_2 < \sigma$. From (23) and (24) we

η	N	$\frac{\|\|\varphi^{(e)} - \widehat{\varphi}\|\|_{\Delta,\infty}}{\|\|\varphi^{(e)}\|\|_{\Delta,\infty}}$	α	$\frac{\|\|\varphi^{(e)} - \widehat{\varphi}\|\|_{\Delta,2}}{\|\|\varphi^{(e)}\|\|_{\Delta,2}}$	α
$\eta_1(x)$	20×20	2.19495e-02		1.46557e-02	
"	40×40	4.64268e-03	2.241	3.95751e-03	1.889
"	60×60	2.07548e-03	1.986	1.81234e-03	1.926
"	80×80	1.16071e-03	2.020	1.03679e-03	1.941
$\eta_2(x)$	20×20	6.78302e-04		6.55385e-04	
"	40×40	1.67616e-04	2.017	1.66366e-04	1.978
"	60×60	7.57982e-05	1.957	7.43733e-05	1.986
"	80×80	4.35374e-05	1.927	4.20004e-05	1.986
$\eta_3(x)$	20×20	3.75848e-03		3.05628e-03	
"	40×40	9.60091e-04	1.969	7.64445e-04	1.999
"	60×60	4.24015e-04	2.016	3.40584e-04	1.994
"	80×80	2.40499e-04	1.971	1.92002e-04	1.992

Table 3: Numerical results for the domain transformation technique for solving the 2D Laplace equation with $\eta = \eta_1, \eta_2, \eta_3$.

can estimate the number of iterations m_{est} needed to solve our model problem. The actual number of iterations used by the PCG method is denoted by m_O. Tables 4-6 show m_{est}, m_O and m_M, where m_M denotes the number of iterations needed by the PCG method when the MILU preconditioner [13] is applied. We have also estimated α in an assumed relation between m_M and the number of unknowns N, $m_M \sim \mathcal{O}(N^{\alpha})$, by comparing subsequent experiments.

	σ=1.0e-04			σ=1.0e-06			σ=1.0e-08			σ=1.0e-10		
N	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α
33^2	8	15		13	20		19	25		24	31	
65^2	8	20	0.212	13	28	0.248	19	36	0.269	24	45	0.275
129^2	7	26	0.191	13	39	0.242	18	53	0.282	24	66	0.279
257^2	7	35	0.216	12	55	0.249	18	74	0.242	23	95	0.264
513^2	7	46	0.198	11	76	0.234	17	105	0.253	23	136	0.260
m_{est}	14			20			26			33		

Table 4: The number of CG-iterations needed to solve our model problem with surface elevation $\eta = \eta_1$.

From Tables 4-6 we observe that the preconditioner based on (22) is preferable to MILU preconditioning. Clearly, m_O is independent of the number of unknowns N, whereas m_M grows with N at the rate $\alpha \approx 0.25$.

4 The dynamic problem

The previous section dealt with the numerical solution of the Laplace equation at a specific time level. Now we describe the method for solving the evolution

	σ=1.0e-04			σ=1.0e-06			σ=1.0e-08			σ=1.0e-10		
N	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α
33^2	8	13		12	18		16	23		21	28	
65^2	7	16	0.153	12	25	0.242	17	32	0.244	22	40	0.263
127^2	7	22	0.232	12	33	0.203	17	45	0.249	22	57	0.258
257^2	7	28	0.231	12	46	0.241	17	64	0.256	22	83	0.273
513^2	7	35	0.161	12	63	0.227	17	90	0.247	22	118	0.255
m_{est}	13			19			25			31		

Table 5: The number of CG-iterations needed to solve our model problem with surface elevation $\eta = \eta_2$.

	σ=1.0e-04			σ=1.0e-06			σ=1.0e-08			σ=1.0e-10		
N	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α	m_O	m_M	α
33^2	7	13		10	19		14	25		18	31	
65^2	7	19	0.280	10	27	0.259	14	37	0.289	18	46	0.291
129^2	6	25	0.200	10	40	0.287	14	54	0.276	18	67	0.274
257^2	6	34	0.223	10	56	0.244	14	78	0.267	17	99	0.283
513^2	6	45	0.203	10	79	0.249	13	112	0.262	17	146	0.281
m_{est}	10			14			19			23		

Table 6: The number of CG-iterations needed to solve our model problem with surface elevation $\eta = \eta_3$.

equations for η and F, i.e. equations (15) and (16). Inside the time interval $[0, T]$, the numerical solution is sought at a finite number of time levels t_k such that $0 = t_0 < t_1 < t_2 \ldots < t_S = T$. We introduce Δt_k such that $\Delta t_k = t_k - t_{k-1}$. When the computational domain is of rectangular shape, it is convenient to solve (15)-(19) by a finite difference method. The details of this method will now be listed for the two-dimensional case, with straightforward extensions to the fully three-dimensional problem. Let the computational domain $[0, L] \times [-H, 0]$ be partitioned into $(n_x - 1)$ and $(n_z - 1)$ subintervals in the x- and z-directions, respectively. We want to compute the approximations of $\varphi(x, \bar{z}, t)$ and $\eta(x, t)$ together with $F(x, t)$ and $G(x, t)$ at discrete time instants at points in the dynamic, physical domain which are mapped from the corresponding grid points in the computational domain,

$$
\begin{aligned}
\eta_i^k &\approx \eta(x_i, t_k), \quad (i, k) \in [1, n_x] \times [0, S], \\
F_i^k &\approx F(x_i, t_k), \quad (i, k) \in [1, n_x] \times [0, S], \\
G_i^k &\approx G(x_i, t_k), \quad (i, k) \in [1, n_x] \times [0, S], \\
\varphi_{i,j}^k &\approx \varphi(x_i, \bar{z}_{i,j}^k, t_k), \quad (i, j, k) \in [1, n_x] \times [1, n_z] \times [0, S], \\
& \qquad x_i = \frac{i-1}{n_x - 1} L, \quad \bar{z}_{i,j}^k = \frac{j-1}{n_z - 1}(\eta_i^k + H) - H.
\end{aligned}
$$

4.1 Discretization of surface conditions

For the equations (15) and (16), we apply centered differences of the Leap-frog type in time. Using a discrete spatial difference operator $\mathcal{D}$ to be defined below, the discrete surface conditions can be written

$$\begin{aligned}
\eta_i^{k+1} &= \eta_i^{k-1} - (\Delta t_{k+1} + \Delta t_k)\left[\mathcal{D}F_i^k \mathcal{D}\eta_i^k - (1 + (\mathcal{D}\eta_i^k)^2)G_i^k\right], \\
F_i^{k+1} &= F_i^{k-1} - \frac{\Delta t_{k+1} + \Delta t_k}{2}\left[(\mathcal{D}F_i^k)^2 - (1 + (\mathcal{D}\eta_i^k)^2)(G_i^k)^2 + 2g\eta_i^k\right],
\end{aligned}$$

for $k = 1, \dots, S$. Here, $\mathcal{D}\eta_i^k \approx \eta_x(x_i, t_k)$ and $\mathcal{D}F_i^k \approx F_x(x_i, t_k)$. The values η_i^1 and F_i^1 are found using the initial conditions (18)-(19)

$$\begin{aligned}
\frac{\eta_i^1 - \eta_i^0}{\Delta t_1} &= 0 \quad \Rightarrow \quad \eta_i^1 = \eta_i^0, \\
\frac{F_i^1 - F_i^0}{\Delta t_1} &= -g\eta_i^0 \quad \Rightarrow \quad F_i^1 = -\Delta t_1\, g\eta_i^0.
\end{aligned}$$

The algorithm for a two-dimensional fully nonlinear wave simulation can then be written as follows.

1. Evaluate the actual initial conditions ($1 \le i \le n_x$): $\eta_i^0 = \eta^0(x_i)$, $F_i^0 = G_i^0 = 0$.

2. While $t_k < T$ carry out the following steps for $k = 0, 1, \dots$:

   (a) Compute $\mathcal{D}\eta_i^k$ and $\mathcal{D}F_i^k$ as the approximations of $\eta_x(x_i, t_k)$ and $F_x(x_i, t_k)$ for $2 \le i \le n_x - 1$.

   (b) If ($k = 0$) then set $\eta_i^1 = \eta_i^0$, $1 \le i \le n_x$, else compute η_i^{k+1} according to the relation ($1 \le i \le n_x$):

   $$\eta_i^{k+1} = \eta_i^{k-1} - (\Delta t_{k+1} + \Delta t_k)\left[\mathcal{D}F_i^k \mathcal{D}\eta_i^k - (1 + (\mathcal{D}\eta_i^k)^2)G_i^k\right].$$

   (c) If ($k = 0$) then set $F_i^1 = -\Delta t_1 g\eta_i^0$, $1 \le i \le n_x$, else compute F_i^{k+1} according to the relation ($1 \le i \le n_x$):

   $$F_i^{k+1} = F_i^{k-1} - \frac{\Delta t_{k+1} + \Delta t_k}{2}\left[(\mathcal{D}F_i^k)^2 - (1 + (\mathcal{D}\eta_i^k)^2)(G_i^k)^2 + 2g\eta_i^k\right].$$

   (d) Solve the Laplace equation in the water volume by the proposed domain transformation technique and the domain imbedding method.

   (e) Compute G_i^{k+1} ($1 \le i \le n_x$) by

   $$G_i^{k+1} = \frac{n_z - 1}{2(\eta_i^{k+1} + H)}(3\varphi_{i,n_z}^{k+1} - 4\varphi_{i,n_z-1}^{k+1} + \varphi_{i,n_z-2}^{k+1}).$$

Remarks.

1. The proposed algorithm is explicit. Formally, the order of accuracy is $\mathcal{O}(\Delta x^2, \Delta t^2)$ for constant Δt and a second order spatial difference operator $\mathcal{D}$.

2. At each time step, during the solution of the Laplace equation, the velocity potential from the previous time step is used as start vector for the PCG method.

3. The Neumann boundary conditions at the solid boundaries $x = 0$ and $x = L$ imply that $\mathcal{D}\eta_1^k = \mathcal{D}\eta_{n_x}^k = \mathcal{D}F_1^k = \mathcal{D}F_{n_x}^k = 0$.

4. $\mathcal{D}\eta_i^k$ and $\mathcal{D}F_i^k$ $(2 \leq i \leq n_x - 1)$ should be the approximation of the first-order derivatives with respect to x on interior grid points at the free surface. The simplest way of calculating them is by the centered finite difference like e.g.

$$\mathcal{D}F_i^k = (F_{i+1}^k - F_{i-1}^k)/2\Delta x, \tag{27}$$

   with the exception of grid points which are at the boundary of the obstacle. However, if $\Omega_S = \emptyset$ and the computational domain is of rectangular shape, cubic spline interpolations of discrete F_i^k and η_i^k can be used to calculate $\mathcal{D}\eta_i^k$ and $\mathcal{D}F_i^k$. Numerical experiments indicate that both accuracy and stability can be enhanced compared with the centered difference (27). In this way, we may achieve better accuracy at a negligible cost since the matrices associated with the spline interpolations can be computed in a factored form once and for all.

5. It is possible to compute G_i^k by the simple form,

$$G_i^k = \frac{n_z - 1}{(\eta_i^k + H)}(\varphi_{i,n_z}^k - \varphi_{i,n_z-1}^k),$$

   but in order to achieve second-order accuracy, we employ the formula given in step 2e of the numerical algorithm.

6. The numerical algorithm is only conditionally stable. In the fully nonlinear case, it is difficult to derive a theoretical stability condition, and Δt must be determined on an experimental basis. In simpler problems a constant Δt is sufficient, while in more challenging problems we apply a variable Δt, where the number of PCG iterations is used as an indicator whether Δt is too large or too small. If the number of iterations is below a lower critical limit, say two, the size of Δt is doubled, while Δt is halved if the number of iterations is larger than an upper critical limit, say ten. Smoothing of F_i^k and η_i^k by standard algorithms can also be necessary.

5 Numerical experiments; 2D waves

The proposed numerical algorithm is first evaluated in the case of a two-dimensional wave motion. The length of the domain in the x-direction is L. Linear elements are used in the finite element discretization. Cubic spline interpolations are applied to calculate $\mathcal{D}\eta_i^k$ and $\mathcal{D}F_i^k$. As stopping criterion for the PCG method with FFT-based preconditioner, we employ $||\boldsymbol{r}_m||_2 < 10^{-6}$, where $\boldsymbol{r}_m = \boldsymbol{A}\boldsymbol{\xi}_m - b$ is the residual in iteration number m. This stopping criterion has shown to be appropriate for the current and later numerical experiments.

5.1 Shallow water

Consider long waves in shallow water with the following parameters,

$$L = 100, \quad H = 5, \quad \eta_0(x) = \frac{729}{16}\left[\left(\frac{x}{L}\right)^2\left(\frac{x-L}{L}\right)^4 - \frac{1}{105}\right].$$

Due to the absence of an analytical solution for the current example, we compare our numerical solution with the results of a spline method developed by Mehlum [15]. The spline computations which define a reference solution were performed on a very fine mesh. We study the convergence of the numerical solution with respect to this reference solution under refinement of the grid. In particular, the surface elevation $\eta(x,t)$ and velocity potential at the surface $F(x,t)$ are considered. Numerical solutions obtained by the new algorithm are denoted by $\widehat{\eta}$ and $\widehat{F}$, while solutions by Mehlum's spline method are denoted by $\eta^{(s)}$ and $F^{(s)}$. Both the discrete L_2-norm and the discrete L_1-norm are utilized, where

$$||g||_{\Delta,2} = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(g_j)^2}, \quad ||g||_{\Delta,1} = \frac{1}{n}\sum_{j=1}^{n}|g_j|,$$

with g being a discrete function with function values g_j, $j = 1, \ldots, n$.

N	Δt	$\frac{\|\|\widehat{\eta}-\eta^{(s)}\|\|_{\Delta,1}}{\|\|\eta^{(s)}\|\|_{\Delta,1}}$	$\frac{\|\|\widehat{\eta}-\eta^{(s)}\|\|_{\Delta,2}}{\|\|\eta^{(s)}\|\|_{\Delta,2}}$	$\frac{\|\|\widehat{F}-F^{(s)}\|\|_{\Delta,1}}{\|\|F^{(s)}\|\|_{\Delta,1}}$	$\frac{\|\|\widehat{F}-F^{(s)}\|\|_{\Delta,2}}{\|\|F^{(s)}\|\|_{\Delta,2}}$
9×5	0.50	1.181e-01	1.351e-01	5.087e-02	5.660e-02
17×9	0.25	2.526e-02	3.082e-02	1.223e-02	1.315e-02
33×17	0.10	7.884e-03	9.170e-03	3.163e-03	3.440e-03
65×33	0.05	2.109e-03	2.395e-03	8.134e-04	8.836e-04
129×65	0.025	5.704e-04	6.459e-04	2.142e-04	2.352e-04

Table 7: The 2D shallow water experiment; a comparison of η and F computed by the new algorithm and Mehlum's spline method at $t = 5$.

For moderate sizes of constant time steps ($0.05 \le \Delta t \le 0.25$), the number of PCG iterations required for solving the corresponding Laplace equation is at most 5 regardless of the grid size. The number decreases for smaller time steps. From Table 7 we observe rapid convergence of the solution with our new algorithm towards the reference solution computed by Mehlum's spline method. In fact, errors less than the line thickness in the plot are easily achieved even on a coarse grid, see Figures 2 and 3.

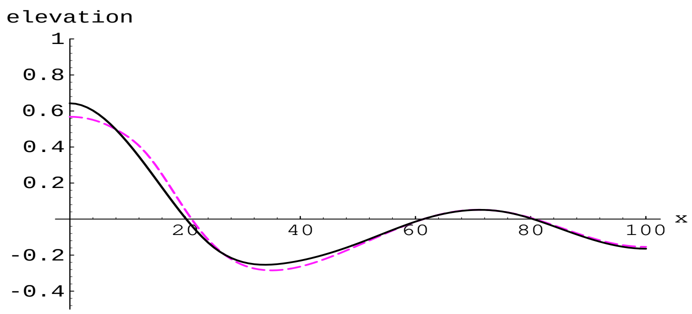

Figure 2: The 2D shallow water experiment; surface elevation at $t = 5$, using $N = 9 \times 5$ and $\Delta t = 0.5$. The solid line is the result from the new algorithm and the dashed line is the result from Mehlum's spline method.

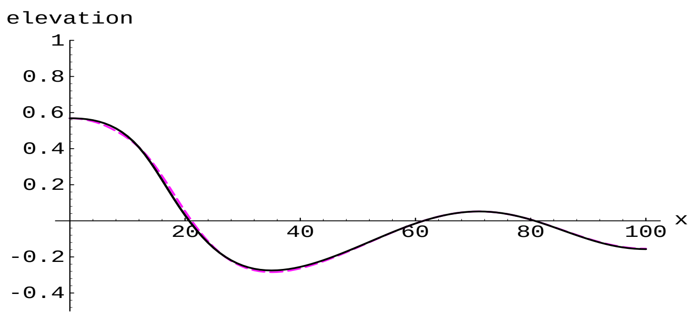

Figure 3: The 2D shallow water experiment; surface elevation at $t = 5$, using $N = 17 \times 9$ and $\Delta t = 0.25$. The solid line is the result from the new algorithm and the dashed line is the result from Mehlum's spline method.

5.2 Deep water

We now consider waves whose typical wavelength is much less than the depth of the water tank. In the experiments, the parameters are as follows.

$$L = 160, \quad H = 70, \quad \eta_0(x) = 6.5 \cos\left(\frac{\pi}{L}x\right) + 5.5 \cos\left(\frac{2\pi}{L}x\right).$$

Again we note that the solution computed by the proposed algorithm converges towards the solution obtained by Mehlums spline method, cf. Table 8. The solution of the Laplace equation at each time level requires 4-6 iterations of the PCG method independent of the number of spatial grid points.

N	Δt	$\frac{\|\widehat{\eta}-\eta^{(s)}\|_{\Delta,1}}{\|\eta^{(s)}\|_{\Delta,1}}$	$\frac{\|\widehat{\eta}-\eta^{(s)}\|_{\Delta,2}}{\|\eta^{(s)}\|_{\Delta,2}}$	$\frac{\|\widehat{F}-F^{(s)}\|_{\Delta,1}}{\|F^{(s)}\|_{\Delta,1}}$	$\frac{\|\widehat{F}-F^{(s)}\|_{\Delta,2}}{\|F^{(s)}\|_{\Delta,2}}$
9×11	0.50	9.607e-02	7.862e-02	5.269e-02	4.969e-02
17×21	0.25	2.578e-02	2.233e-02	1.627e-02	1.493e-02
33×41	0.10	7.436e-03	6.817e-03	5.677e-03	5.186e-03
65×81	0.05	2.601e-03	2.485e-03	2.981e-03	2.917e-03

Table 8: The 2D deep water experiment; a comparison of η and F computed by the new algorithm and Mehlum's spline method at $t = 10$.

6 Numerical experiments; 3D waves

6.1 The wave motion in a water tank

We consider waves in a 3D water tank without any internal obstacle. The three-dimensional computational domain is $\Omega = [0, 80] \times [0, 80] \times [-50, 0]$ and the initial condition is

$$\eta^0(x, y) = \left(-0.9\cos\left(\frac{\pi x}{L_1}\right) + \cos\left(\frac{2\pi x}{L_1}\right)\right)\left(1 - 0.9\cos\left(\frac{\pi y}{L_2}\right) + \cos\left(\frac{2\pi y}{L_2}\right)\right).$$

Trilinear elements are used in the finite element discretization. Again we compare the numerical solution from the new algorithm with the solution computed by Mehlum's spline method on a very fine grid. The information about the grid sizes and the errors can be found in Table 9. The PCG method converges within 5-7 iterations at each time step for the FFT-based preconditioner and the same stopping criterion as in the 2D experiments.

N	Δt	$\frac{\|\widehat{\eta}-\eta^{(s)}\|_{\Delta,1}}{\|\eta^{(s)}\|_{\Delta,1}}$	$\frac{\|\widehat{\eta}-\eta^{(s)}\|_{\Delta,2}}{\|\eta^{(s)}\|_{\Delta,2}}$	$\frac{\|\widehat{F}-F^{(s)}\|_{\Delta,1}}{\|F^{(s)}\|_{\Delta,1}}$	$\frac{\|\widehat{F}-F^{(s)}\|_{\Delta,2}}{\|F^{(s)}\|_{\Delta,2}}$
$5 \times 5 \times 6$	0.50	2.626e-01	2.458e-01	3.427e-01	3.872e-01
$9 \times 9 \times 11$	0.25	9.761e-02	8.744e-02	1.027e-01	1.226e-01
$17 \times 17 \times 21$	0.10	3.409e-02	2.998e-02	3.221e-02	4.361e-02
$33 \times 33 \times 41$	0.05	1.100e-02	9.561e-03	1.256e-02	2.696e-02

Table 9: The wave motion in a water tank; a comparison of η and F computed by the new algorithm and Mehlum's spline method at time $t = 4$.

6.2 A water tank containing an obstacle

We end this presentation by a numerical experiment with water waves in a tank containing a vertical cylinder with square base, i.e. the physical domain is on the form (3) with Ω_S being a square located close to the center of the tank. The horizontal section of the geometry to be considered is depicted in Figure 4. This experiment is partially representative for simulating wave forces on a marine installation. However, real-world applications usually involve more complicated geometries. As mentioned earlier in this paper, a domain imbedding approach is introduced to treat the internal obstacle. In the general case, the geometry of the obstacle will not intersect the underlying uniform grid of the computational domain along the grid lines. This is of no principal difficulty, but in the numerical experiments we have decided to work with a geometry that coincides with the grid lines such that errors from intersections are eliminated.

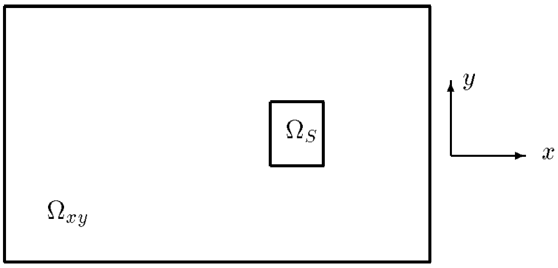

Figure 4: Top view of a water tank containing a square obstacle Ω_S.

We set $\Omega = [0, 200] \times [-60, 60] \times [-70, 0]$ and $\Omega_S = [125, 150] \times [-15, 15]$. The initial condition is on the form

$$\eta^0(x, y) = \begin{cases} 9\left(\frac{x-35}{35}\right)^4 \left(\frac{4(x-35)}{35} + 5\right) - 0.54564 & 0 \leq x \leq 35, \\ -0.54564 & \text{otherwise.} \end{cases}$$

Because of the symmetry in the y-direction, it suffices to solve the water wave system in half of the original region, i.e. for $y \geq 0$. At each time step, equations (15) and (16) are solved in the domain $\Omega_{xy} \backslash \Omega_S$.

The domain imbedding parameter was chosen to be $\epsilon = 10^{-12}$. In this 3D case the discretized Laplacian (26) was solved by a multigrid method. Thus providing a suitable implementation of the preconditioner $\boldsymbol{M}$ for the problem (11)-(14). Figure 5 shows that the numerical solutions converge under refinement of the grid. Moreover, figure 6 indicates that the proposed numerical algorithm models the water flow around the obstacle properly. For suitable time steps (e.g. $\Delta t = 0.05, 0.1$) we observed that the PCG method gained expected accuracy in less than 10 iterations per time step. The number of CG-iterations needed to solve the problem was independent of the domain imbedding parameter ϵ and the number of spatial grid points. Further details of the domain imbedding procedure and the influence of ϵ can be found in [16] and [7].

7 Conclusions

We have developed a new numerical scheme for the fully nonlinear equations modelling water waves. The key feature of the method is that regridding of the computational domain at each time step is avoided and optimal convergence of

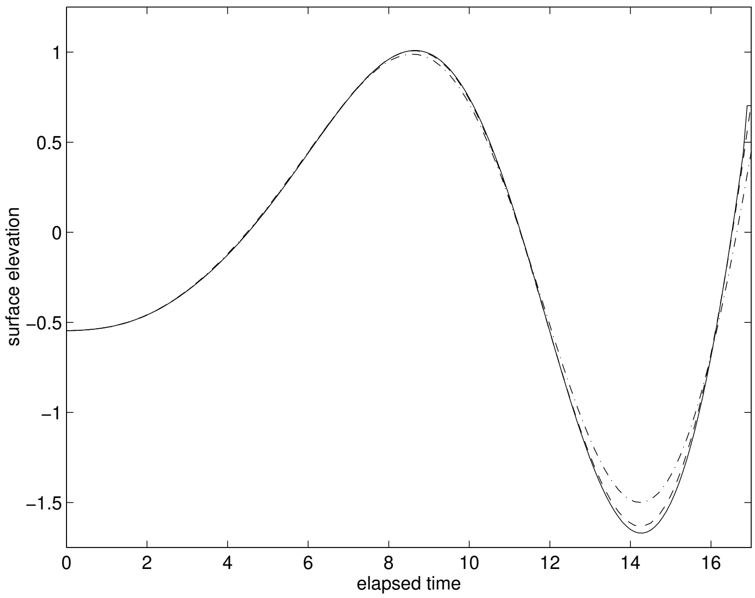

Figure 5: A 3D water tank containing an obstacle; Numerical solutions of the surface elevation at $(x = 125, y = 22.5)$ for $0 \le t \le 17$. The solid line is the result from the numerical discretization $n_x = 129$, $n_y = 33$, $n_z = 49$ and $\Delta t = 0.05$; The dashed line is the result from the numerical discretization $n_x = 65$, $n_y = 17$, $n_z = 25$ and $\Delta t = 0.1$; The dash-dotted line is the result from the numerical discretization $n_x = 33$, $n_y = 9$, $n_z = 13$ and $\Delta t = 0.1$.

the conjugate gradient method is achieved. Numerical comparisons with another carefully tested scheme show that the solutions generated by the two schemes seem to converge towards the same solution as the mesh sizes are reduced.

Acknowledgement

The authors want to thank Dr. Mehlum for numerous interesting discussions and for providing test results using his novel spline-based method. Finally we want to thank Dr. Bruaset, Dr. Rusten and Professor Winther for useful discussions concerning the numerical solution of elliptic boundary value problems.

References

[1] O. Axelsson, V. A. Barker, *Finite Element Solution of Boundary Value Problems, Theory and Computation*, Academic Press, Orlando, FL, 1984.

[2] A. A. Becker, *The Boundary Element Method in Engineering, A complete course*, McGraw-Hill, 1992.

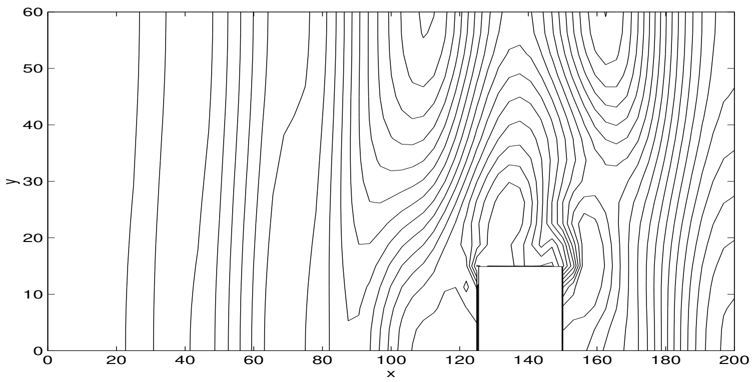

Figure 6: A 3D water tank containing an obstacle; Contour plot of the surface elevation at $t = 30$; The parameters for the numerical discretization are $n_x = 65$, $n_y = 17$, $n_z = 25$ and $\Delta t = 0.1$.

[3] J. H. Bramble, J. E. Pasciak and A. H. Schatz, *An Iterative Method for Elliptic Problems on Regions Partitioned into Substructures*, Math. of Comp., v. 46, 1986, pp. 361-369.

[4] E. O. Brigham, *The Fast Fourier Transform*, Prentice-Hall, Englewood Cliffs, N.J., 1974.

[5] A. M. Bruaset, *A Survey of Preconditioned Iterative Methods*, Pitman Research Notes In Mathematics Series, Longman Scientific & Technical, 1995.

[6] B. L. Buzbee, G. H. Golub, C. W. Nielson, *On direct methods for solving Poisson's equations*, SIAM J. Numer. Anal., Vol. 8, No. 4, (1971), pp. 722–736.

[7] X. Cai, B. F. Nielsen and A. Tveito, *An analysis of a preconditioner for the discretized pressure equation arising in reservoir simulation*, Preprint 1995-4 at The Department of Informatics, University of Oslo.

[8] P. G. Ciarlet, *The finite element method for elliptic problems*, North-Holland Publishing Company, 1978.

[9] Diffpack Home Page, *http://www.oslo.sintef.no/diffpack*.

[10] F. W. Dorr, *The direct solution of the discrete Poisson equation on a rectangle*, SIAM Rev., 12 (1970), pp. 248–263.

[11] P. Ferrant, *Time domain computation of nonlinear diffraction loads upon three dimensional floating bodies*, ISOPE 95 Conference, 1995.

[12] A. Greenbaum, *Diagonal scalings of the Laplacian as preconditioners for other elliptic differential operators*, SIAM J. Matrix Anal. Appl., Vol. 13, No. 3 (1992), pp. 826–846.

[13] I. Gustafsson, *A class of first order factorization methods*, BIT, 12 (1978), pp. 142–156.

[14] W. Hackbusch, *Multi-Grid Methods and application*, Springer-Verlag, 1985.

[15] E. Mehlum, *Splines and ocean wave modelling*, Report STF A93008, SINTEF Oslo, (1993).

[16] B. F. Nielsen and A. Tveito, *On the approximation of the solution of the pressure equation by changing the domain*, to appear in SIAM J. Appl. Math.

[17] Nonlinear Water Wave Simulator Home page, *http://www.oslo.sintef.no/diffpack/projects/xic/Nonlinear-Wave/www/index.html*.

[18] P. N. Swarztrauber, *The methods of cyclic reduction, Fourier analysis and the FACR algorithm for the discrete solution of Poisson's equation on a rectangle*, SIAM Review, 19 (1977), pp. 490–501.

[19] R. E. Taylor, *Analysis of Non-Linear Wave-Body Interaction Using Finite Elements.* In J. Grue, B. Gjevik and J. E. Weber (eds): *Waves and Nonlinear Processes in Hydrodynamics*, Kluwer Academic Publishers, 1996.

[20] G. B. Whitham, *Linear and nonlinear waves*, John Wiley & Sons, Inc., New York, 1974.

[21] V. E. Zakharov, *Stability of periodic waves of finite amplitude on the surface of a deep fluid*, J. Appl. Mech. Tech. Phys., 9 (1968), pp. 190–194. (English tansl.)