Computational Mechanics (1993) 13, 12–28

Computational Mechanics

A three-dimensional panel method for nonlinear free surface waves on vector computers

Jan Broeze[1], **Edwin F. G. van Daalen**[2], **Pieter J. Zandbergen**[2]

[1] Delft Hydraulics, P.O. Box 152, 8300 AD Emmeloord, The Netherlands
[2] University of Twente, Faculty of Applied Mathematics, P.O. Box 217, 7500 AE Enschede, The Netherlands

Abstract. A number of recent improvements in a higher-order three-dimensional (3D) panel method for highly nonlinear free surface wave simulations is discussed. Special attention is paid to grid evolution techniques, and stability of the time-dependent problem. Due to the improvements, stable and accurate results can be obtained for linear and highly nonlinear wave problems. Although no artificial smoothing is applied, even extreme problems like the development of breaking waves in a 3D configuration can be simulated. The computer code has been specially developed for implementation on a vector computer. The program is highly vectorized, and use is made of mathematical libraries for acceptable CPU-times.

1 Introduction

The nonlinear interaction of steep gravity waves with non-flat bottom profiles plays an important role in many hydrodynamic problems. Also the interaction with fixed or freely floating bodies is a serious problem that deserves a lot of attention.

For the description of waves propagating over smooth bottom topologies or over large structures, viscous effects can be neglected, as well as compressibility effects. Assuming the flow to be irrotational then, the problem can be simplified by introducing a velocity potential. This potential satisfies Laplace's equation.

Different classes of solution methods for the nonlinear free surface potential flow problem exist. Analytical solutions can be obtained relatively easily in simple geometries, assuming the solution to have certain properties (such as periodicity in time and periodicity in space). For nonlinear problems with small wave heights, solutions to the linearized problems may still suffice, but the fully nonlinear problem is much more difficult to solve.

Numerical methods are needed for solving nonlinear problems in complex geometries. Boussinesq-type equations can still be used for weakly nonlinear waves in shallow water, but a model for the full fluid flow (without prior assumptions about the solution) is necessary for solving arbitrary nonlinear problems.

For the numerical solution of the fully nonlinear free surface wave problem, various numerical techniques can be used. Field discretization techniques (see e.g. [12]) have the property that the amount of grid points increases very rapidly for large problems, or that dense grids are desirable. Boundary element techniques (or panel methods) need less extra nodes for dense grids. These methods have the additional advantage that less complicated grid motion algorithms are needed in solving problems with changing boundary shapes.

Various boundary element methods and panel methods for nonlinear free surface wave problems have been developed. Some successful 2D methods are described in [3, 4]. Many authors claim that their method can be extended to three dimensions. However, large problems may occur then. These problems are mainly due to the description of the evolution of the grid, the geometric modelling, numerical instabilities and the CPU-time required for the computations.

Results of successful 3D computations have been reported in [2, 10, 11, 13]. However, in most of these methods, numerical instabilities have to be suppressed by numerical smoothing techniques, or the method is of low order (so that a very dense grid is needed; on the other hand, these dense grids cannot be used due to memory restrictions). Another drawback is the CPU-time needed for these methods.

In this paper we will discuss the further development of a higher order 3D panel method for nonlinear free surface waves. The technique that we apply for the motion of the grid appears to be of great importance for the success of the method. Additionally, we spent a lot of attention to the numerical stability of our approach. Due to accurate stability considerations, we managed to obtain a stable discretization, without numerical smoothing. Additionally, we found a useful condition for the time step in nonlinear computations.

Due to the improvements, the method provides stable results for nonlinear waves, up to breaking, without artificial smoothing. The method has been implemented on a supercomputer, and due to a high degree of vectorization, high performances are obtained.

The reader is referred to [1] for an extensive description and analysis of the improvements in the method.

The outline of this paper is as follows. In the following section we give the equations that govern the problem. In Sect. 3 a short overview is given of the panel method developed by Romate [7]. Next we discuss our new approach for the grid evolution (Sect. 4), some adaptations in the panel method (Sect. 5), and the stability of the time-dependent problem (Sect. 6). In the seventh section, the efficiency of the method on a supercomputer is discussed. Finally, numerical results of computations on a periodic propagating wave and a breaking wave are presented in Sect. 8.

2 Mathematical formulation

The fluid is assumed to be inviscid and incompressible. Under the assumption that the fluid flow is irrotational, a velocity potential ϕ (satisfying $\underline{v} = \nabla\phi$) can be introduced. The continuity equation reduces to Laplace's equation for the potential:

$$\nabla^2\phi = 0, \quad \underline{x} \in \Omega. \tag{1}$$

In order to have a well-posed elliptic problem, boundary conditions (ϕ or $\partial\phi/\partial n$ prescribed) should be given on all boundaries of the domain. Solving this problem provides the potential and its spatial derivatives.

Time-dependency comes into the problem by the time-dependent boundary conditions. These give expressions for the time derivative of ϕ or $\partial\phi/\partial n$ in terms of the position and spatial derivatives. By inserting the numerically evaluated values for these expressions, values for the time derivatives can be obtained, so that the problem can be integrated in time in each node on the boundary. Using the values obtained on the new time level as new boundary conditions for the elliptic problem, again we have a well-posed problem, and the solution of the spatial problem can be determined. This process can be repeated until the desired time level has been reached.

The boundary conditions that govern the evolution at the free surface are the dynamic free surface condition (expressing that the pressure is zero):

$$\frac{\partial\phi}{\partial t} + \frac{1}{2}(\nabla\phi)^2 + gz = 0, \quad (\underline{x} \in S_f) \tag{2}$$

and the kinematic condition (expressing that the node travels along with the free surface):

$$\frac{D\underline{x}}{Dt} \cdot \underline{n} = \frac{\partial\phi}{\partial n}, \quad (\underline{x} \in S_f) \tag{3}$$

($\underline{n}$ is the outward directed normal).

On the bottom of the domain the no-flux condition holds:

$$\frac{\partial\phi}{\partial n} = 0, \quad (\underline{x} \in S_b). \tag{4}$$

This condition can also be imposed on fixed lateral boundaries of the domain, or on fixed structures.

The domain can be truncated in horizontal direction by simulating a wave maker (on which the normal derivative of the potential is prescribed as a function of position and time), or by using absorbing boundary conditions (simulating artificial boundaries). A first order absorbing boundary condition that we implemented is Sommerfeld's condition, where c is an estimated value for the phase velocity:

$$\frac{\partial \phi}{\partial t} = -c\frac{\partial \phi}{\partial n}, \quad (\underline{x} \in S_a). \tag{5}$$

We refer the reader to [2, 7] for higher order absorbing boundary conditions, that can be used to radiate waves propagating at different angles of incidence with respect to the boundary or at different velocities.

3 Original numerical algorithm

Romate [7–9] has developed a panel method for simulating 3D free surface waves. His method provides stable and very accurate results for linear and mildly nonlinear wave problems, due to the higher order solution method for the spatial problem and the accurate time integration method.

However, the method provides instabilities in the simulation of highly nonlinear gravity waves. In a typical simulation, large errors in the elevation occur at the free surface near the inflow and outflow boundaries, which spoil the numerical solution within one wave period.

In this section we will give a brief description of Romate's method, so that alternations that we have made, can be more easily understood.

3.1 Solution method for the spatial problem

For the solution of the spatial problem—Eq. (1) with appropriate boundary conditions—Romate [7] applied a higher order panel method. In this method, the elliptic field equation is not solved by a direct discretization, but the field equation is replaced by a boundary integral equation, with the same solutions. Numerically solving the boundary integral equation—where only a discretization of the boundaries is needed—provides a solution to the elliptic field equation.

In the panel method, first a higher order approximation of the boundary is obtained by approximating it by curved quadrilateral panels (geometric description). Next, the boundary integral equation is applied in one collocation point per panel, where higher order expansions of the functions along the panels are included. By inserting the known boundary conditions (ϕ or its normal derivative) for each collocation point into the discretized boundary integral equations, a square matrix problem is obtained for the unknowns. Solving this matrix equation provides the solution to the problem on the boundary.

Geometric modelling. For the approximation of the boundary integral equations, the boundary S is first divided into a number of smooth subsurfaces. These subsurfaces must each be mapped onto a rectangle in the computational domain. This mapping is defined by mapping a given set of points onto the integer points in the computational domain (using cubic splines for the x, y and z-values). By dividing the rectangle into squares with sides of unit length, a panel division is obtained from the back transformation to the physical domain.

From the spline approximation for the surfaces, for each panel some data are determined from the spline approximation, such as:

- one collocation point (in the centre of the panel in the computational domain);
- a local coordinate system for the panel, with origin in the collocation point, two axes in tangential direction along the boundary and one axis in inward normal direction;
- quadratic curvature terms for a higher order approximation of the local panel shape;

- weights for finite difference approximations for first and second order tangential derivatives, using values in this panel collocation point and in the collocation points of adjacent panels.

Influence coefficients computation. Influence coefficients are obtained from a discretization of the boundary integral equation that is solved instead of the field equation.

The elliptic field equation is solved by imposing Green's third identity on the boundary:

$$-\frac{1}{2}\phi(\underline{x}) = \iint_{\partial\Omega} \left(\frac{\partial\phi}{\partial n_\xi}(\underline{\xi})\cdot G(\underline{\xi},\underline{x}) - \phi(\underline{\xi})\cdot\frac{\partial G}{\partial n_\xi}(\underline{\xi},\underline{x}) \right) \mathrm{d}S_\xi \qquad (\underline{x} \text{ on } \partial\Omega) \tag{6}$$

where G is a fundamental solution of Laplace's equation, with $\underline{x}$ on a smooth part of the boundary. This integral equation is imposed in all N collocation points, where the boundary integral in the right-hand side expression is split up into contributions from each panel:

$$-\frac{1}{2}\phi(\underline{x}_j) = \sum_{i=1}^{N} \iint_{\Delta S_i} \left(\frac{\partial\phi}{\partial n_\xi}(\underline{\xi})\cdot G(\underline{\xi},\underline{x}_j) - \phi(\underline{\xi})\cdot\frac{\partial G}{\partial n_\xi}(\underline{\xi},\underline{x}_j) \right) \mathrm{d}S_\xi \qquad (j = 1,\ldots,N). \tag{7}$$

The following higher order approximation of Eq. (7) is used:

$$\begin{aligned} -\frac{1}{2}\phi(\underline{x}_i) = \sum_{j=1}^{N} \Bigg[& c_s^0(i,j)\frac{\partial\phi}{\partial n}(\underline{x}_j) + c_s^{s_1}(i,j)\frac{\partial}{\partial s_1}\frac{\partial\phi}{\partial n}(\underline{x}_j) + c_s^{s_2}(i,j)\frac{\partial}{\partial s_2}\frac{\partial\phi}{\partial n}(\underline{x}_j) \\ & - c_d^0(i,j)\phi(\underline{x}_j) - c_d^{s_1}(i,j)\frac{\partial}{\partial s_1}\phi(\underline{x}_j) - \cdots - c_d^{s_2 s_2}(i,j)\frac{\partial^2}{\partial s_2^2}\phi(\underline{x}_j) \Bigg] \end{aligned} \tag{8}$$

where $\{\underline{s}_1, \underline{s}_2, \underline{n}\}$ is a local coordinate system in 3D space, with $\underline{s}_1$ and $\underline{s}_2$ in tangential direction and $\underline{n}$ in normal direction with respect to the panel.

The values of the c_s- and c_d-terms follow from a higher order discretization of the variation of $G(\underline{\xi},\underline{x})$ resp. $\partial G/\partial n$ along the panels in Eq. (7).

The surface integrals along the panels in the right-hand side expression of this equation are approximated by using truncated Taylor expansions of the functions in physical space, with the collocation point as expansion point for the functions. The (first and second order) tangential variations are expressed in the values in neighbouring collocation points, using the finite difference weights determined at the stage of the geometric modelling.

The surface integrals are approximated by first assuming the panel to be flat, and adding contributions due to quadratic curvature. They are computed by reducing most surface integrals to line integrals along the panel edges (see [8]).

Solution of the spatial problem. The discretized boundary integral equations provide a set of N equations for $2N$ values ϕ and ϕ_n:

$$C_d\underline{\phi} + C_s\underline{\phi}_n = \underline{0} \tag{9}$$

where $\underline{\phi}$ and $\underline{\phi}_n$ are the values of ϕ and ϕ_n in the N collocation points. C_d is the matrix constituted from the dipole coefficients, and C_s contains the source coefficients.

In order to have a well-posed problem for the elliptic equation, on all boundaries either ϕ or ϕ_n must be given. Inserting the known value of ϕ or ϕ_n in each collocation point into (9) provides the matrix equation

$$A\underline{x} = \underline{b} \tag{10}$$

where $\underline{b}$ is constituted from inserting the known values into (9), and $\underline{x}$ stands for the unknown values in (9). Solving this matrix equation provides a solution in the collocation points for the boundary integral equation problem.

Tangential first and second order spatial derivatives are obtained by using the weights for finite difference approximations, determined at the geometric modelling stage. Values for the derivatives in the global coordinate system are calculated by using the local transformation matrix from the local coordinate system to the global system.

3.2 *Simulation of time-dependent problems*

In the free surface wave problem, time-dependence comes into the problem by the time-dependence in the boundary conditions.

In the linear method, the linearized equations govern the evolution of the boundary conditions in time (imposed on the undisturbed surface), where Laplace's equation must be satisfied on each time level.

When solving nonlinear problems, also the nonlinear terms are included in the dynamic free surface condition, and the positions of the collocation points change in time (governed by the dynamic free surface condition). Romate [7] chose an Eulerian domain (fixed horizontal positions for the vertical boundaries that truncate the free surface). This implies that after each time step, a grid redistribution technique had to be used for determining a new grid along the free surface.

The classical fourth order Runge-Kutta method was chosen, because it has a relatively favourable stability region, and it can be easily implemented.

4 Grid evolution

In nonlinear free surface wave computations, the time-dependent shape of the free surface should be taken into account, because the free surface conditions are imposed at the disturbed surface position. Additionally, it is desirable to be able to include other moving boundary types, such as a wave maker or a moving body in the fluid domain.

The development of the boundary shape in time is governed by the grid motion. The grid motion can be divided into two parts:

- the motion in normal direction, that governs the evolution of the boundary shape in time, and
- the tangential motion of the collocation points, that governs the grid distribution along the boundary for the following time steps.

The motion of the domain boundaries in normal direction is part of the solution. The values for the motion velocities in normal direction at the free surface are determined by the kinematic boundary condition, Eq. (3). On the other boundaries, we will suggest additional conditions.

The tangential motion is not governed by given equations, but the choice of the tangential velocities is part of the numerical technique for solving the time-dependent problem.

In this section, first we explain our technique for the evolution of the grid in time, i.e. how the grid on a new time level is obtained. Next, we will discuss the motion of the grid in normal direction, because the equations that govern the normal motion are independent of the numerical solution method. After that, we will consider the tangential grid motion.

4.1 *Grid motion technique*

In the panel method, the geometric description of the boundaries is determined from the collocation point positions.

A frequently applied technique for the development of the grid along the boundary in simulations with time-dependent shape of the boundaries, is grid distributions along the boundary after the Lagrangian displacements (see e.g. [7]). Such a redistribution is necessary in order to maintain a well-connected grid at the intersections of the surface with the lateral boundaries. However, such redistributions may introduce additional inaccuracies.

In our method, each collocation point j travels at a motion velocity $\underline{v}_j$. This velocity is integrated in time with the same time integration method as the boundary conditions. At a new time level, again the geometric modelling algorithm is applied, which provides a new approximation of the boundary shape on the basis of the new collocation points positions. By choosing appropriate values for the tangential motion velocities of the collocation points, a desirable panel distribution on the new time level is obtained.

In this method, no redistributions are applied after the Lagrangian movement of the collocation points. Here, large deformations of the panel distributions can occur, which may lead to inaccuracies in the panel method. However, the deformations are necessary in the computation of highly nonlinear waves and breaking waves. These deformations occur in a natural way in our method. Additionally, the new approach can handle non-equidistant grids, which are desirable in problems with locally large gradients in the solution.

4.2 Domain boundaries evolution

At the free surface, the kinematic condition, Eq. (3), should be satisfied.

At natural fixed boundaries (such as the bottom or fixed structures), the normal velocity should be zero.

Various options can be used for the motion of (artificial) lateral inflow and outflow boundaries. We have tested the following options for the motion of these boundaries:

- an Eulerian approach: fixed horizontal positions for the inflow and outflow boundaries ($v_n = 0$),
- a Lagrangian approach: the boundary travels along with the fluid particles in normal direction ($v_n = \partial\phi/\partial_n$),
- an "advanced" approach: the lateral boundaries travel in horizontal direction at the horizontal motion velocity of the fluid particles at the intersection with the free surface, without dependency on the vertical coordinate.

Notice that in the third option, the normal components of the boundary velocity and the fluid particles velocities do not necessarily coincide.

From a variational formulation we have concluded that, for well-posedness, the velocity of the computational domain boundary and the fluid particles should coincide at the intersection of the free surface with the lateral boundaries (see [1, 13]). Hence, the first (Eulerian) approach provides an ill-posed problem, and will lead to numerical instabilities at the intersection of the free surface with the lateral boundaries.

The fully Lagrangian description provides a well-posed problem. However, due to variations of the horizontal Stokes drift along the vertical, after a few wave periods, highly curved inflow and outflow boundaries occur (see [13]). In order to prevent numerical instabilities then, a very dense grid is required on these boundaries, which is undesirable.

The third option is the most suitable one, and in the numerical results we will show that stable and long computations are possible with this approach.

4.3 Grid distribution evolution

The equations that govern the development of the potential and the boundary shape in time provide a unique solution for the continuous problem. However, the motion of the collocation points in tangential direction along the boundaries is not prescribed. This degree of freedom can be utilized to keep a desirable grid distribution in time, without the need of grid redistributions.

In the numerical method, the boundaries are approximated by a number of panels with a finite number of node points (the collocation points). In a certain sense these collocation points should be regularly distributed along each smooth boundary part. However, there are no restrictions on variations in grid sizes. Hence, variations may exist in the tangential motion velocity of the collocation points.

We have developed a tangential grid motion control algorithm for two purposes. The first aim is to move the nodes such that well-connected grids (of adjacent networks) are maintained. The second purpose is to have an adaptive grid motion algorithm that concentrates the grid at certain locations when desired, and on the other hand prevents the grid from becoming too dense anywhere.

Basic tangential grid motion. In order to make large deformations of the grid develop in a natural way, for example in computations of the development of breaking waves, a basically Lagrangian description for the position $\underline{x}$ of each node point is chosen:

$$\frac{D\underline{x}}{Dt} = \nabla\phi. \tag{11}$$

This choice has the additional advantage that in nonlinear computations the grid clusters near a wave top (where the largest gradients occur in the solution), and generally the grid is least dense in a wave bore (with smaller gradients), see [1].

Due to the partially non-Lagrangian motion of the inflow and outflow boundaries, the tangential motion velocities of the collocation points on the lateral boundaries and bottom should next be corrected to match well with the velocities of each adjacent boundary. In our method, these corrections are determined from a comparison of the *tangential* velocity of collocation points with the *normal* velocity of the adjacent boundary. Next, these corrections are added to the tangential motion velocities of the collocation points.

For more details of this algorithm, we refer the reader to [1].

Adaptive grid motion. Finally, small corrections are applied to the tangential velocities to obtain a desirable grid distribution along the subsurfaces at the new time level.

Corrections of the local tangential motion velocities of the collocation points are determined from:

- local variations in distances between the subsequent collocation points,
- variations in velocities,
- local curvature of the boundary,
- curvature of the grid lines along the boundary,
- connection with the grid lines on adjacent boundaries.

This adaptive grid motion algorithm concentrates the grid at certain locations when desired, and prevents the grid from becoming too dense anywhere.

5 Adaptations of the panel method for solving the spatial problem

The panel method used for solving the spatial problem is a very accurate higher order method. It provides very accurate solutions for elliptic problems. Up to quadratic curvature terms are included in the method. Hence, it provides accurate results for slightly curved panels. However, special attention is needed for highly curved surface shapes, because curvature terms of third degree are not included.

At the stage of geometric modelling, Romate [7] used bicubic splines, i.e. the surfaces are represented by piecewise polynomials of third degree. However, only up to quadratic curvature terms are included in the further analysis. We have found (see [13]) that this approach may lead to a singular Jacobian occurring for unequal grid spacings. Using a quadratic spline prevents the occurrence of these possible causes for numerical instabilities, without loss of accuracy. These theoretical results were confirmed by numerical computations on a breaking wave. The first approach provides numerical instabilities during the development of the plunging wave (and spoils the solution before the wave front has become vertical). Using a quadratic spline representation gives stable results, almost until the tip of the jet hits the free surface.

We have also considered the accuracy of the values of the influence coefficients for highly curved surfaces. Using quadratically curved panels for the approximation of the surface may introduce gaps between the panels. These gaps have influence on the numerical solution for polynomial solutions of low degree of Laplace's equation. We have experienced numerical instabilities due to these inaccuracies in time-dependent computations. Therefore, some coefficients, such as the constant dipole coefficient (solid angle) should be evaluated by using the exact positions of the corners of the panels.

6 Time integration and numerical stability

For the time integration we implemented the classical fourth order Runge-Kutta method and a fourth order 2-stage 2-derivative generalized Runge-Kutta method. The first method was chosen because it can be easily implemented, but it requires large computational efforts (the geometric data and influence coefficients must also be redetermined at the intermediate time levels due to the time dependency of the positions of the boundaries). The other method is cheaper, but values for the second time derivative are needed.

These two time integration methods have equal stability regions. They contain a considerable part of the imaginary axis (from $-2\sqrt{2}i$ to $2\sqrt{2}i$), which is favourable for the computation of wave problems.

In this section we will pay more attention to the stability of the numerical solution method for the free surface wave problem.

6.1 Numerical stability: semi-discretizations

In this subsection we perform some stability analyses of the nonlinear free surface problem, assuming the time integration to be exact, i.e. we only consider (in)stabilities due to the spatial discretizations.

First, we analyze the stability of the unbounded free surface problem, and afterward we will consider the influence of edges and corners on the stability.

The unbounded free surface. Here, Von Neumann analysis is used to verify the numerical Cauchy stability (see [5]) of the nonlinear free surface problem.

The Von Neumann condition requires that no eigensolution to a (*linear*) numerical initial value problem exists that is periodic in space and has an absolute growth factor greater than 1 in time.

In order to analyze the stability of the numerical approximations for the nonlinear free surface problem, the problem is linearized around a nonlinear solution. The growth of a disturbance (ε, ψ) in (ϕ, η) is investigated. In the continuous problem, the equations that govern the evolution of this combined solution in a Lagrangian system are:

$$\frac{D(\phi+\varepsilon)}{Dt} = -\frac{1}{2}(\nabla\phi)^2 - \nabla\phi\cdot\nabla\varepsilon - \frac{1}{2}(\nabla\varepsilon)^2 - g\eta - g\psi + \underline{v}\cdot\nabla\phi + \underline{v}\cdot\nabla\varepsilon; \quad \frac{D(\eta+\psi)}{Dt} = \frac{\partial\phi}{\partial z} + \frac{\partial\varepsilon}{\partial z}. \tag{12}$$

Here, a 2D disturbance is considered that is harmonic along the surface (and decays in vertical direction), of the form: ((ξ, ζ) are the coordinates in a local coordinate system $(\underline{s}, \underline{n}\}$, with $\underline{s}$ in tangential direction and $\underline{n}$ in outward normal direction)

$$\begin{bmatrix} \varepsilon \\ \psi \end{bmatrix} = e^{\omega t} \begin{bmatrix} e^{ik\xi + |k|\zeta}\varepsilon_0 \\ e^{ik\xi}\psi_0 \end{bmatrix} \quad (k \in \mathbb{R}, |k| \leqq \pi/(\Delta\xi)). \tag{13}$$

The evolution of the disturbance in time is determined by the value of ω in Eq. (13). A stable problem is obtained if for the eigensolutions of the initial-value problems, this parameter has a non-positive ($\leqq 0$) real part.

Equation (12) governs the evolution of the solution and the disturbance in time. For determining the value of ω in Eq. (13), it is most convenient to consider the growth in a system with zero tangential motion velocity v_s. Notice that the free surface condition is imposed at the exact surface position, so the velocity of the system in normal direction v_n should be equal to the normal derivative of the solution: $v_n = \partial(\phi + \varepsilon)/\partial n$. Hence, the derivatives d./dt in the following equations denote time derivatives in a system travelling at velocity $\partial(\phi + \varepsilon)/\partial n$ in normal direction. The evolution of the errors at the surface is governed by:

$$\frac{d\varepsilon}{dt} = \frac{\partial\varepsilon}{\partial t} + v_n\frac{\partial\varepsilon}{\partial n} \approx -\phi_s\frac{\partial\varepsilon}{\partial s} - g\psi; \quad \frac{d\psi}{dt} = \underline{n}\cdot\underline{e}_3\frac{\partial\varepsilon}{\partial n} = \cos\alpha\frac{\partial\varepsilon}{\partial n} \tag{14}$$

where α is the local angle of inclination of the free surface, ϕ_s denotes the tangential derivative of the velocity potential ϕ, and v_n stands for the velocity in normal direction.

Now, consider the evolution of a numerical error in the solution. For that aim, the discrete counterparts of the spatial derivatives of ε and ψ in the right-hand side of (14) should be inserted. In the numerical approximations, central discretizations for the tangential derivatives are used. Because of the high order of accuracy of the panel method (see [7, 8]), here the numerically evaluated normal derivative is assumed to be exact.

The numerical discretizations provide:

$$\omega\begin{bmatrix}\varepsilon_0\\ \psi_0\end{bmatrix}=\begin{bmatrix}-i\phi_s\sin(k\Delta\xi)/\Delta\xi & -g\\ |k|\cos\alpha & 0\end{bmatrix}\begin{bmatrix}\varepsilon_0\\ \psi_0\end{bmatrix}. \tag{15}$$

The matrix in this equation has eigenvalues

$$\lambda=\frac{i}{2}\left(-\phi_s\sin(k\Delta\xi)/\Delta\xi\pm\sqrt{[\phi_s\sin(k\Delta\xi)/\Delta\xi]^2+4g|k|\cos\alpha}\right). \tag{16}$$

These eigenvalues are purely imaginary (in other words, the eigensolutions have $\mathrm{Re}(\omega)=0$), so the problem is marginally stable, and the Von Neumann condition is satisfied.

Stability of the free surface treatment near the boundaries. Here we consider the possible occurrence of instabilities at the edges of the free surface due to the numerical approach. We will consider possible instabilities due to one-sided discretization of terms in the time-dependent boundary conditions.

Stable results have been obtained by a number of researchers for the nonlinear free surface problem in 2D boundary element methods (with collocation points on the corners of the boundary elements), see e.g. [4]. They obtain these stable results only by applying the free surface conditions as well as a boundary condition (usually the same condition as the boundary condition used on the whole lateral boundary) in the edge points of the free surface.

Because of the complex structure of the discretizations for the fully nonlinear free surface problem near the boundaries, we restrict the analysis of the bounded free surface wave problem to necessary conditions for stability.

Above, it was derived that in the interior collocation points of the free surface, solutions that are periodic in space will not grow in time, i.e. the Von Neumann condition for stability is satisfied. Hence, the numerical discretizations in the interior free surface points provide a marginally stable problem.

Here we consider whether such periodic solutions will not grow due to the discretizations in the outmost collocation points.

For the solution at the free surface, Eq. (12) governs the development of the sum of the undisturbed solution and the disturbance in time. Near the intersection with a vertical boundary, the collocation points travel at a horizontal motion velocity that is equal to the horizontal derivative of the undisturbed potential (simulating a wave maker that generates or absorbs the wave solution). In a Lagrangian system, the evolution of the disturbance in the collocation points in time can be written as follows then:

$$\frac{D\varepsilon}{Dt}\approx(v_{s_1}-\phi_{s_1})\frac{\partial\varepsilon}{\partial s_1}+(v_{s_2}-\phi_{s_2})\frac{\partial\varepsilon}{\partial s_2}-g\psi,\quad \frac{D\psi}{Dt}=\cos\alpha\frac{\partial\varepsilon}{\partial n}+v_{s_1}\frac{\partial\psi}{\partial s_1}+v_{s_2}\frac{\partial\psi}{\partial s_2} \tag{17}$$

where $\underline{s}_1$ is the tangential unit vector along the free surface normal to the intersection line with the lateral boundary (outward directed) and $\underline{s}_2$ is the tangential unit vector along the free surface in the direction of the edge line. Subscripts of $\underline{v}$ represent components of the motion velocity of the collocation point into that direction, and subscripts of ϕ denote partial derivatives.

Central discretizations for the tangential derivatives in s_2-direction are used. One-sided discretizations are used for the tangential derivatives in s_1-direction.

Now, consider the effects for a periodic solution in the tangential directions, i.e. ε and ψ are assumed to have the form $\exp(i\kappa_1\xi+i\kappa_2\vartheta+(|\kappa_1|+|\kappa_2|)\zeta)$ with $\kappa_1,\kappa_2\in\mathbb{R}$. (ξ,ϑ,ζ) are coordinates in 3D space, with basis $\{\underline{s}_1,\underline{s}_2,\underline{n}\}$.

Inserting this form into Eq. (17), with a one-sided two-point discretization for the tangential derivatives in s_1-direction, gives:

$$\frac{D}{Dt}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix}=\begin{bmatrix}M_{11} & -g\\ \cos\alpha(|\kappa_1|+|\kappa_2|) & M_{22}\end{bmatrix}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix} \tag{18}$$

with

$$M_{11}=(v_{s_1}-\phi_{s_1})\frac{1-\mathrm{e}^{-i\kappa_1\Delta\xi}}{\Delta\xi}+(v_{s_2}-\phi_{s_2})\frac{\mathrm{e}^{i\kappa_2\Delta\vartheta}-\mathrm{e}^{-i\kappa_2\Delta\vartheta}}{2\Delta\vartheta},$$

$$M_{22}=v_{s_1}\frac{1-\mathrm{e}^{-i\kappa_1\Delta\xi}}{\Delta\xi}+v_{s_2}\frac{\mathrm{e}^{i\kappa_2\Delta\vartheta}-\mathrm{e}^{-i\kappa_2\Delta\vartheta}}{2\Delta\vartheta} \tag{19}$$

where $\Delta\xi$ and $\Delta\vartheta$ are the mesh-widths in s_1 and s_2-direction respectively. M_{11} and M_{22} have small real parts, where the signs of these parts depend on $(v_{s1}-\phi_{s1})$ and on ϑ_{s1}. The matrix in (18) has eigenvalues

$$\lambda=\tfrac{1}{2}(M_{11}+M_{22}\pm\sqrt{(M_{11}-M_{22})^2-4g\cos\alpha(|\kappa_1|+|\kappa_2|)}). \tag{20}$$

An eigenvalue with small positive real part occurs if $(v_{s1}-\phi_{s1})>0$ or if $v_{s1}>0$. Instabilities will arise then.

Notice that the instabilities are due to the occurrence of numbers with small positive real parts on the diagonal of the matrix in Eq. (18). Therefore, in order to eliminate the instabilities we have studied the modification of the dynamic free surface condition in the outmost collocation points as follows:

$$\frac{\partial\phi}{\partial t}=a((\phi_{s_1})_{\text{lat. bound.}}-(\phi_{s_1})_{\text{free surf.}})-\frac{1}{2}(\nabla\phi)^2-gz \tag{21}$$

where the derivatives are extrapolated values toward the intersection of the free surface and the vertical boundary. The parameter a should be positive, but small to prevent inaccuracies. The first term in the right-hand side provides a negative contribution to the first diagonal element M_{11}. Because, for solutions of the continuous problem, both derivatives in s_1-direction in Eq. (21) are equal, using Eq. (21) in the outmost collocation points does not influence the exact solution, but possible numerical derivations are suppressed.

Adding a contribution as in Eq. (21) to the system provides a contribution to the first diagonal element in the matrix in Eq. (18), due to the one-sided discretization of the tangential derivative at the free surface. If a two-point discretization for this derivative is used, and the approximation of the derivative on the lateral boundary is exact, Eq. (18) becomes:

$$\frac{D}{Dt}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix}=\begin{bmatrix}M_{11}-ad & -g\\ \cos\alpha(|\kappa_1|+|\kappa_2|) & M_{22}\end{bmatrix}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix} \tag{22}$$

with $d=(1-\exp\{-i\kappa_1\Delta\xi\})/\Delta\xi-i\kappa_1$, which has a (small) positive real part. Thus, the eigenvalues of the matrix become

$$\lambda=\tfrac{1}{2}(M_{11}+M_{22}-ad\pm\sqrt{(M_{11}-M_{22}+ad)^2-4g\cos\alpha(|\kappa_1|+|\kappa_2|)}). \tag{23}$$

These eigenvalues only have non-positive real parts if a is large enough.

In corners of the surface, contributions as in Eq. (21) should be used for derivatives in both tangential directions, because one-sided discretizations are used there for the tangential derivatives in both directions.

Interpretation: We conclude from the eigenvalues in Eq. (23) that both types of numerical instabilities are suppressed by adding a contribution as in Eq. (21) to the dynamic free surface condition at the edges of the free surface. Numerical results presented in [1] illustrate that Eq. (21) is essential for numerical stability.

Notice that in linear computations the diagonal elements M_{11} and M_{22} are equal to 0. Hence, the numerical instabilities described above will not occur then.

6.2 *Time step restrictions due to the free surface conditions*

Due to the bounded stability domain of the time integration methods that we apply, the time steps used in the computations are subjected to some limitations. In this subsection we consider the restrictions due to the free surface conditions.

Stability requirements for the unbounded linearized free surface wave problem are well known. From inserting a Fourier-mode into the linearized equations, one can easily conclude that for stability of the problem we need

$$i\sqrt{gk}\,\Delta t \in R_{\text{stab}} \tag{24}$$

where R_{stab} is the stability region of the time integration method, and k is the wave number for the shortest wave that can be represented on the grid ($k = \pi/(\Delta x)$).

More complex conditions can be derived if also the nonlinear terms are included in the analyses.

From Eq. (14) we obtain the following equations for the evolution of an error ε and ψ in ϕ resp. z:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix} = \begin{bmatrix}-i\phi_s k & -g\\ k\cos\alpha & 0\end{bmatrix}\begin{bmatrix}\varepsilon\\ \psi\end{bmatrix} \tag{25}$$

where we have assumed that the disturbance is harmonic, with wave number, k, and α is the local angle of inclination of the free surface ($\cos\alpha = \mathrm{d}x/\mathrm{d}s)_{\text{surf}}$).

For numerical stability of the time integration, the product of the time step and the eigenvalues of the matrix in the right-hand side of Eq. (25) should be in the stability region of the time integration method. The eigenvalues of the matrix are:

$$\lambda = \frac{i}{2}(-\phi_s k \pm \sqrt{(\phi_s k)^2 + 4gk\cos\alpha}). \tag{26}$$

We can conclude that for stability of the nonlinear method, the following condition should be satisfied:

$$\frac{i}{2}(-\phi_s k \pm \sqrt{(\phi_s k)^2 + 4gk\cos\alpha})\Delta t \in R_{\text{stab}}, \quad \forall k \in [0, \pi/\Delta x]. \tag{27}$$

The time integration methods that we apply in our method, the classical fourth order Runge-Kutta method or the 2-stage 2-derivative Runge-Kutta method, have absolute stability regions that include the part of the imaginary axis $\langle -i2\sqrt{2}, i2\sqrt{2}\rangle$.

We conclude that a necessary condition for the time step, when solving the unbounded nonlinear free surface problem with these time integration methods, is:

$$\tfrac{1}{2}(-\phi_s k \pm \sqrt{(\phi_s k)^2 + 4gk\cos\alpha})\Delta t < 2\sqrt{2}, \quad k = \pi/\Delta x, \quad \cos\alpha = (\mathrm{d}x/\mathrm{d}s)_{\text{surf}}. \tag{28}$$

7 Efficient computer implementation

The numerical solution procedure discussed in the previous sections has been applied in a computer code (using FORTRAN 77), and was implemented on a CRAY Y-MP4/464. In this section we consider the CPU-time usage of the most expensive parts of the method, exploiting the possibilities of the supercomputer.

7.1 *Efficiency*

The major part of the CPU-time required in the computations is due to the determination of the influence coefficients and the solution of the matrix problems. The other routines together typically require less than 5% of the CPU-usage. We will pay some special attention to the most expensive parts.

Influence coefficients. For accuracy reasons, analytic expressions are used for the influence coefficients in the near-field. Gauss quadrature may be used for the panels in the far-field without influence on the accuracy.

The use of combined analytic expressions and Gauss quadrature has been vectorized by first gathering all field points that require the same integration method for a panel into one vector. Next the influence coefficients are calculated per vector, and the results are scattered afterwards.

We have managed to vectorize the whole process of computation of the analytic expressions for the influence coefficients by using very long loops. In this loop, the influence of one panel due to each collocation point for which analytic expressions are demanded, is computed. As a consequence, data such as the solid angle need to be computed once for determining the constant, linear and quadratic source and dipole contributions. Because the evaluation of these data is very expensive, this process requires much less CPU-time than if the coefficients of different order would be evaluated in different loops. However, the resulting loop is very long (it contains about 500 FORTRAN source lines), and the "-o aggress" option is needed in order to vectorize the loop.

The following table gives the CPU-time needed for the evaluation of influence coefficients for three typical problems on the CRAY Y-MP, using one processor.

Table 7.1. CPU-times required for the computation of influence coefficients on the CRAY Y-MP for a some typical configurations

Vector-mode	Approximation	Number of panels		
		450	650	1000
Non-vectorized	all analytic	17.5s	36.5s	
	analytic/Gauss	7.5s	12.6s	
Vectorized	all analytic	2.3s	4.5s	
	analytic/Gauss	1.1s	1.7s	10s

The number of coefficients that has to be determined is quadratic in the number of panels. However, if Gauss quadrature is used for larger problems, the relative number of coefficients to be evaluated numerically will increase. Thus, the CPU-time required for the evaluation of the influence coefficients is of a lower power than 2 in time. In fact, from test computations we found that approximately $T_{\text{inf. coeff.}} \sim N^{1.5}$ (in the range $100 \leqq N \leqq 2500$).

Solution of the matrix problem. Direct as well as iterative methods for solving the matrix problem $A\underset{\sim}{x} = \underset{\sim}{b}$ have been applied. We have compared the required CPU-time using Gaussian elimination, successive overrelaxation and a Preconditioned Conjugate Gradients Squared method (PCGS).

Although the third method has been specially developed for sparse matrix problems, we found that it is much faster than the other methods (for a typical problem with 650 unknowns, 0.1s are required for the latter method, whereas 0.6s are needed for the Gaussian elimination). In this method, preconditioning is performed with the diagonal elements.

The most expensive part of the PCGS method is the matrix-vector multiplication. In order to exploit the performance of the supercomputer, we have considered the efficiency of three approaches for this part: (the values between brackets indicate the required CPU-time for one matrix-vector multiplication with 2040 rows and columns):

- own implementation; treating each row after the previous one. (0.054s)
- own implementation; treating each column after the previous one. (0.037s)
- BLAS routine MXV (equal to NAG-routines MURRV and SGEMV). (0.027s)

Although for problems with different vector-lengths, the required CPU-times for the matrix vector multiplication are different, the ratios are approximately equal. Obviously the BLAS routine is the fastest one, as could be expected.

The row by row approach requires more CPU-time than the column by column approach, because the columns of the matrix are stored sequentially in the memory.

The number of operations for performing one iteration with PCGS is about quadratic in the number of panels. However, for larger problems, the matrix problem will be worse conditioned, so that more iterations are needed. Therefore, the increase of CPU-time required for solving the matrix problem with PCGS will be worse than quadratic in the number of unknowns. However, this also occurs for other iterative schemes, and direct methods require CPU-times that are of third order in the number of unknowns. For that reason, we prefer PCGS in our computations.

Performance. The Cray has a theoretical maximum performance of approximately $1.3*10^9$ floating operations per second (utilizing the 4 processors). This implies that theoretically the maximum number of floating operations per CPU second is about $325*10^6$.

The performance in computations with our method depends on the problem to be solved. In computations with a linearized approach, where geometric data and influence coefficients need to be determined once, most CPU time is spent by the PCGS matrix-vector solution algorithm. Most CPU time in this algorithm is spent on the matrix-vector multiplication. A very high performance is achieved for this operation because of the efficient usage of the vector pipes by these instructions, so that the performance of the computer is near the maximum attainable. In computations with the nonlinear approach, the geometric data and influence coefficients are redetermined each (intermediate) time level. Although this part is fully vectorized, the vectors are considerably shorter, due to the separation into analytical and numerical expressions. Moreover, the instructions are less straightforward, so that the pipes of the vector machine have a lower filling degree. In typical computations with the nonlinear method, performances of about $150*10^6$ floating operations per CPU second have been measured.

7.2 Further improvements of the efficiency

In the previous subsection we considered the most efficient implementation of algorithms for the panel method. In this subsection we consider how the efficiency can be further increased by an optimum usage of the available data.

Frozen coefficients. Above we have seen that the computation of the influence coefficients contributes largely to the required CPU time in nonlinear computations.

In linear computations, the influence coefficients for the panel method only need to be determined at the initial time level. In nonlinear computations, the shape of the boundary of the domain changes. Therefore, new geometric data, and new values for the influence coefficients should be determined every (intermediate) time level. This makes the computations much more expensive, even if nonlinearities play a minor role in the solution.

Romate [7] reduced the extra amount of work needed by not evaluating new values for the influence coefficients on the intermediate time levels of the four-stage Runge-Kutta method. However, this approach may reduce the accuracy of the method (and possibly introduce numerical instabilities) in highly nonlinear wave computations.

In numerical computations on mildly nonlinear wave problems, very often small time steps are needed for the stability of the solution, but not for accuracy. In such situations 'frozen coefficients' —approaches are still desirable. For that aim we have introduced a flexible version of this approach: the geometric data and influence coefficients are kept fixed during a given time interval. This makes it possible to have an accurate approach for highly nonlinear wave problems, whereas we can save a large amount of CPU-time in nearly linear computations.

One should be careful with utilizing this option. Due to the basically Lagrangian approach at the surface, the grid cluster near a wave top. In [1] it is shown that due to using frozen coefficients, a gradual loss of volume occurs. No loss of volume occurs if the coefficients are re-evaluated at all time levels.

An alternative is to use frozen coefficients in the far field—where the variations of the integrands in Eq. (7) are small—and to update the coefficients in the near field. However, such an approach would only provide a small reduction of the CPU time, since (expensive) analytical expressions are used for the influence coefficients in the near field, and the Gauss quadrature approximations for the panels in the far field are relatively inexpensive.

Time step. In the previous section we found a condition for the maximum allowable time step.

Because of the high order of accuracy of the time integration methods, mostly small time steps are only needed for the stability of the method. For accuracy reasons, larger time steps might suffice. Therefore, it is useful to use time steps that approximate the maximum allowable time step for stability.

The maximum allowable time step depends on the solution (see the condition derived in the previous section). Thus, it is useful to determine the maximum allowable time step at every time level. Using this maximum value, the existing time step may be reduced or increased. This dynamic time stepping approach provides the solution at a certain time level with minimal effort.

7.3 Memory usage

In the panel method, three large full matrices (with sizes N^2) occur. The first two matrices contain the source and dipole coefficients of the matrix equation, and the third matrix is used to store the system matrix (of the matrix problem to be solved). Additionally some (relatively little) memory space is needed for other variables.

In principle, only storing the resulting system matrix would be sufficient. However, in our method also the matrices containing the source and dipole coefficients are stored. These matrices are needed afterward, because problems with different boundary conditions (e.g. for the potential problem, and for the partial time derivatives) need to be solved with the same influence coefficients. The coefficients are needed then to generate the different right-hand side expressions. Another reason for conserving the coefficient matrices is that some solution methods for the matrix-vector problem alter the system matrix. For that reason we have decided to store the three matrices.

7.4 Domain decomposition techniques

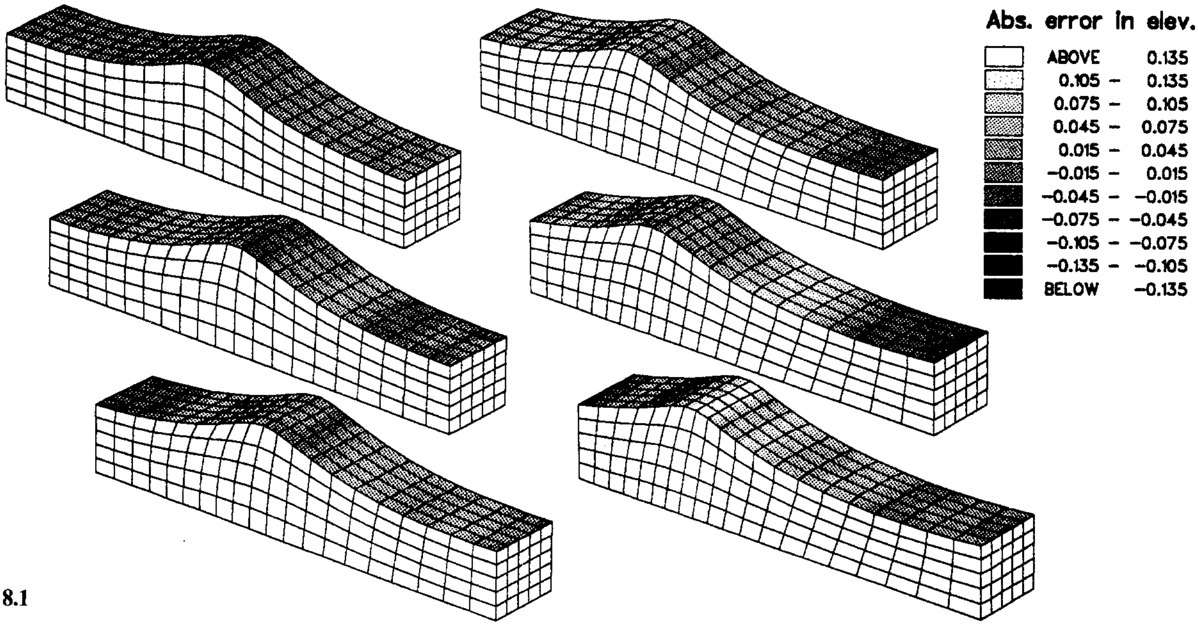
Due to the above mentioned memory requirements, we can use at most approximately 3200 panels on the CRAY Y-MP (with job-size limit 32 MWord). However, in order to be able to handle realistic problems with a large number of wave lengths in the domain, more panels are needed. For that reason, domain decomposition techniques (see e.g. Kane [6]) should be applied to further extend the applicability of the method. Domain-decomposition techniques reduce the memory requirements of the problems, because of the quadratic relation between the number of panels and the memory requirement. Dividing the problem into a number of subdomains provides a large reduction of the number of coefficients to be computed and stored, especially in shallow domains. Domain decomposition techniques have the additional advantage that more efficient algorithms may be applied for the solution of the matrix problem, as shown in [6].

8 Numerical results

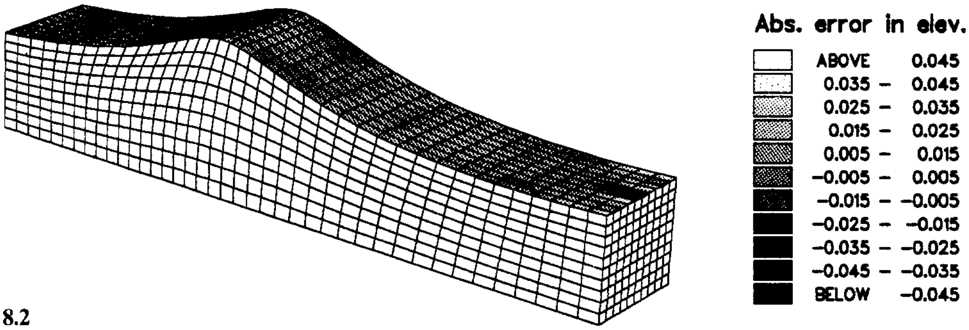
Due to the improvements, the method provides stable results, not only for linear problems, but also for (highly) nonlinear wave problems.

First, we will show results of a computation on a periodic, propagating wave solution with height 5 m on 10 m water depth (over 80% of the maximum height), and wave length 60 m. This wave solution has an Eulerian wave period of 6.5 s.

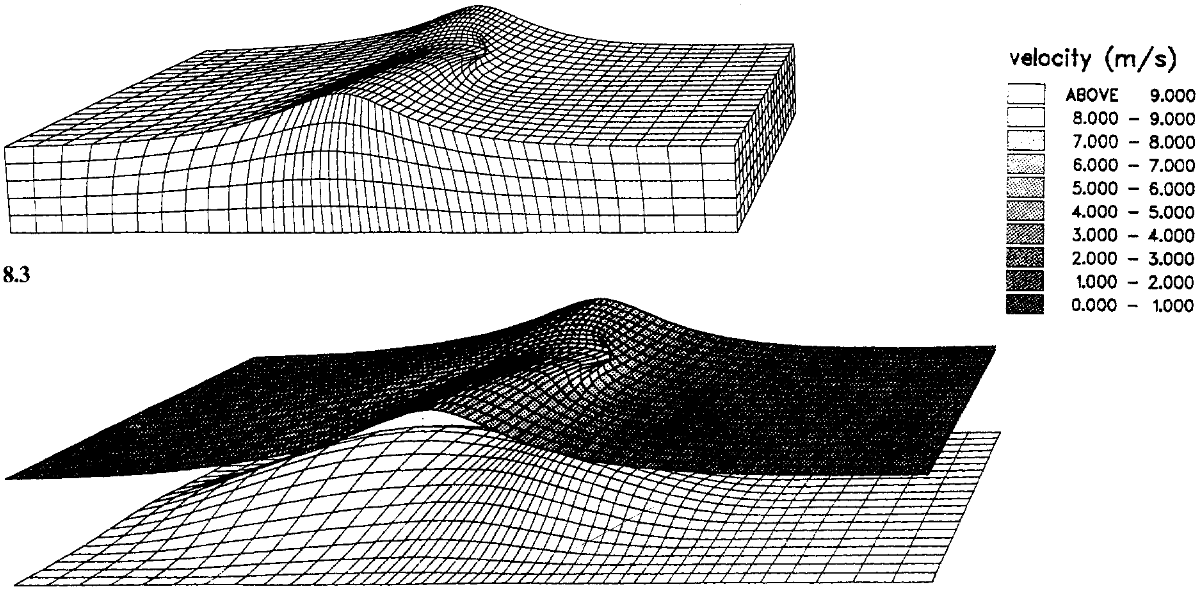
With 20 rows of panels in the wave propagation direction, 5 rows in cross direction, and 5 in vertical direction on the vertical boundaries, the total number of panels is 450. The total memory



8.1



8.2



8.3

8.4

Figs. 8.1–8.4. 8.1 Results of computations on a highly nonlinear periodic wave, propagating to the right. Shape of the grid and errors in elevation of the solution after 0 to 5 wave periods (at 1 wave period intervals). **8.2** Results of computations on a highly nonlinear periodic wave, propagating to the right, with a dense grid. Shape of the grid and errors in elevation of the solution after 2 wave periods. **8.3** Shape of free surface and lateral boundary grids after interaction of a solitary wave with a sand bank. **8.4** Perspective view of free surface and bottom after interaction of a solitary wave with a sand bank. Shadings at the free surface represent velocities of fluid particles

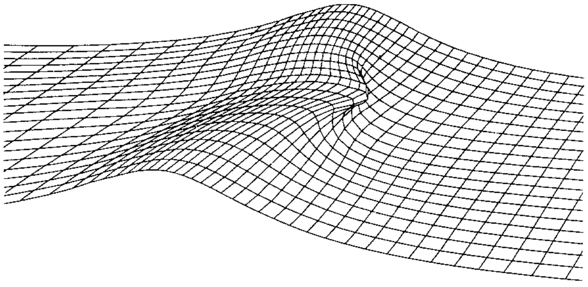


Fig. 8.5. Detailed view of the free surface grid shape during wave breaking

requirement in this simulation is about 6 Mword. For each time step, approximately 3 CPU seconds are required. With time steps 0.2 s—i.e. 33 time steps per wave period—about 90 CPU seconds are needed for one wave period.

In the numerical results on this problem which were shown in [13], instabilities were observed in the corners of the free surface. These numerical instabilities are eliminated by applying (21) to the dynamic free surface condition. The following figure shows the shape of the boundary grid, and errors in the elevation (compared with an exact Fourier solution) of the initial solution, and up to 5 wave periods (at 1 wave period intervals).

Figure 8.1 shows that the numerical results only contain small errors in the elevation. During the first 5 wave periods, the errors in the elevation are within 2% of the wave height. No large increase of the errors occurs during this time interval and after longer computations. From this, we conclude that the method provides stable and accurate results for such highly nonlinear wave problems. The computations on this solution can be continued, without growing errors or instabilities.

Figure 8.2 shows results from computations on the same problem with a finer grid (with the double number of rows of panels in each direction) after two wave periods. From comparing Figs. 8.1 and 8.2 it is clear that a denser grid provides more accurate results.

Also extreme phenomena like the development of breaking waves can be computed with the method. A plunging breaker was obtained by computing the interaction of a high solitary wave (with height 3.5 m on 5 m water) with a "sand bank" on the bottom. The sand bank has maximum height 4 m, and a radius of 20 m.

In these computations, the fluid domain boundaries are discretized with 2500 panels, so that 25 Mword memory space is used. For determining the influence coefficients, about 25 CPU seconds are needed. If the coefficients are not frozen, 1 CPU minute is needed for one time step. In the computations on this problem, during the steepening of the wave, a concentration of the panels occurs near the jet of the wave, where large velocities occur. Equation (28) shows that very small time steps are required then to prevent numerical instabilities. In our computations, a typical value for the time step during the final stages of computation is 0.005 s. It took about 150 time steps to attain the level of Fig. 8.3, i.e. $2\frac{1}{2}$ CPU hours were needed.

During the computations, the value of the time step was dynamically controlled, demanding Eq. (28) to be satisfied in every surface point. Without using any numerical smoothing technique, the following solution is obtained (see Figs. 8.3–8.5). It can be seen from those figures that the wave has started to break.

The grid used in these computations is too coarse for further continuing the computations (the large curvature near the tip cannot be represented with this grid very well). Using a coarser grid provides the computations to break down sooner. Due to restrictions on the memory requirements of the jobs, a much finer grid cannot be used. Therefore, we have not yet repeated this computation with a finer grid.

It should be noted that computations, wherein the time steps sometimes violate condition (28), clearly provide numerical instabilities, independent of the steepness of the wave. This indicates that (28) is very useful for the determination of the maximum allowable time step.

9 Conclusions

In this paper we have discussed some essential approaches for obtaining stable and accurate results for (highly nonlinear) free surface wave problems without applying numerical smoothing techniques.

The control of the grid motion is very important for an advanced approach for the evolution of the domain boundaries. This tool also prevents the necessity of using grid redistribution techniques after every time step.

Some parts of the numerical solution technique for the spatial problem (degree of the spline approximation for the boundaries and the accuracy of the evaluations for the influence coefficients) do not only have influence on the accuracy of the spatial solution, but they also are important for the stability of the time-dependent results.

We have considered the stability of the discretization technique for the problem. A special contribution to the dynamic free surface condition is needed at the edges of the free surface, to prevent numerical instabilities. These possible instabilities are due to the fact that one-sided discretizations for the tangential derivatives along the free surface are used. From the stability analyses we also derived a very useful condition for the time step. This condition can be used for a dynamic control of the time step during the computations.

Numerical results indicate that the method provides stable and very accurate results. Highly nonlinear periodic waves, as well as the development of breaking waves can be simulated without numerical instabilities, although no artificial smoothing is used. The computations on the breaking wave are truncated because a finer mesh is needed for a good representation. Due to memory restrictions we have not yet been able to repeat those computations with a finer grid.

Acknowledgements

These investigations were supported by the Netherlands Technology Foundation (STW). Funds for using the Cray Y-MP supercomputer were provided by the National Computing Facilities Foundation of the Netherlands (NCF/WGS).

References

1. Broeze, J. (1993): Numerical modelling of nonlinear free surface waves with a 3D panel method. Ph.D. thesis, Enschede, The Netherlands (to appear)
2. Broeze, J.; Romate, J. E. (1992): Absorbing boundary conditions for free surface wave simulations with a panel method. J. Comp. Phys. 99, 146–158
3. Dold, J. W.; Peregrıne, D. H. (1986): An efficient boundary-integral method for steep unsteady water waves. In: K. W. Morton et al. (eds.): Numerical methods for fluid dynamics II. Oxford University Press, Oxford, pp. 671–679
4. Grilli, S. T.; Skourup, J.; Svendsen, I. A. (1989): An efficient boundary element method for nonlinear water waves. Eng. Anal. Boundary Elements 97–107
5. Gustafsson, B.; Kreiss, H.-O.; Sundström, A. (1972): Stability theory of dıfference approximations for mixed ınitial boundary value problems, II. Math. Comp. 26, 649–686
6. Kane, J. H.; Keyes, D. E.; Guru Prasad, K. (1991): Iterative solution techniques ın boundary element analysis. Int. J. Num. Meth. Eng. 31, 1511–1536
7. Romate, J. E. (1989): The numerical simulation of nonlınear gravity waves in three dimensions using a hıgher order panel method. Ph.D. thesis, Enschede
8. Romate, J. E. (1988): Local error analysis in 3D panel methods. J. Eng. Math. 22, 123–142
9. Romate, J. E.; Zandbergen, P. J. (1989): Boundary integral equation formulations for free-surface flow problems ın two and three dimensions. Computational Mechanics 4, 276–282
10. Skourup, J.; Sterndorff, M. J.; Hansen, E. A. (1991): A 3D nonlinear boundary element method for wave-structure interaction. In: A. S. Arcilla et al. (eds.): Computer modelling in ocean engineering '91, Balkema, Rotterdam, 57–68
11. Xü, H.; Yue, D. K. P. (1992): Numerical study of three-dimensional overturning water waves. To appear in: Revised abstracts of the Seventh International Workshop on water waves and floating bodies, Val de Reuil, France
12. Yeung, R. W.; Vaidhyanathan, M. (1992): Non-linear ınteraction of water waves with submerged obstacles, Int. J. for Numer. Methods in Fluıds 14, 1111–1130
13. Zandbergen, P. J.; Broeze, J.; van Daalen, E. F. G. (1992): A panel method for the simulation of nonlınear gravity waves and ship motions. To appear in: J. H. Kane et al. (Eds.): Advances in boundary element techniques. Berlin, Heidelberg, New York: Springer

Communicated by T. A. Couse, Aprıl 19, 1993