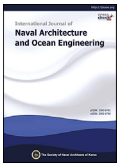
Available online at www.sciencedirect.com

ScienceDirect

Publishing Services by Elsevier

International Journal of Naval Architecture and Ocean Engineering xx (2017) 1—15
http://www.journals.elsevier.com/international-journal-of-naval-architecture-and-ocean-engineering/



Improved HPC method for nonlinear wave tank

Wenbo Zhu [a], Marilena Greco [b,c,*], Yanlin Shao [d]

[a] Marine Design and Research Institute of China, Shanghai, China
[b] Centre for Autonomous Marine Operations and Systems (AMOS), Dept. of Marine Technology, NTNU, Trondheim, Norway
[c] CNR-INSEAN, Marine Technology Institute, Via di Vallerano 139, 00128, Roma, Italy
[d] Department of Mechanical Engineering, Technical University of Denmark, 2800, Lyngby, Denmark

Received 7 October 2016; revised 4 March 2017; accepted 9 March 2017
Available online ■ ■ ■

Abstract

The recently developed Harmonic Polynomial Cell (HPC) method has been proved to be a promising choice for solving potential-flow Boundary Value Problem (BVP). In this paper, a flux method is proposed to consistently deal with the Neumann boundary condition of the original HPC method and enhance the accuracy. Moreover, fixed mesh algorithm with free surface immersed is developed to improve the computational efficiency. Finally, a two dimensional (2D) multi-block strategy coupling boundary-fitted mesh and fixed mesh is proposed. It limits the computational costs and preserves the accuracy. A fully nonlinear 2D numerical wave tank is developed using the improved HPC method as a verification.

Keywords: Harmonic polynomial cell method; Potential-flow theory; Flux method; Fixed mesh; Multi-block strategy; Nonlinear numerical wave tank

1. Introduction

A proper prediction for ships and offshore-structures behavior at sea is vitally important for motion control and safety guarantee. Meanwhile, the wave—body interactions and resulting consequences on the response of ships and platforms depend on the features of the incident-wave systems, e.g. the nonlinearities. The latter are affected by the water depth and, from finite to shallow depth regions, by the sea—bottom topography that can support wave steepening and breaking. When such waves meet floating marine structures diffraction and radiation waves are caused; viscous effects may matter depending on the dimensions and geometry of the structure and on the occurrence of flow separation. A Computational Fluid Dynamics (CFD) method based on Navier—Stokes (NS) equations for generally viscous and turbulent flows is apparently closest to the physical phenomena and most comprehensive among all the numerical methods because it has the least assumptions. However, the application of such CFD methods is still limited due to the requirement for strong computational ability and substantial increase of computer power. Therefore, Domain Decomposition (DD) strategies, within which potential flow theory is applied in regions where viscous effects can be neglected, will be of great benefits.

The Harmonic Polynomial Cell (HPC) method developed by Shao and Faltinsen, (2012) demonstrates high accuracy and efficiency as a potential flow field solver. It has been extended to 3D by Shao and Faltinsen (2014a) and applied to study interior and exterior flow problems relevant for marine applications such as sloshing in tanks, nonlinear waves over different sea-bottom topographies, and nonlinear wave diffraction by a bottom-mounted vertical circular cylinder. Lately, current effects have been included by Shao and Faltinsen (2014b). However, strictly speaking inconsistency exists in the original HPC method for Neumann boundaries

* Corresponding author. Marine Technology Department and NTNU AMOS, NTNU, Otto Nielsens Veg 10, 7491, Trondheim, Norway.
E-mail addresses: marilena.greco@cnr.it, marilena.greco@ntnu.no (M. Greco).

Peer review under responsibility of Society of Naval Architects of Korea.

http://dx.doi.org/10.1016/j.ijnaoe.2017.03.009

where the normal derivatives are discontinuous (Shao and Faltinsen, 2012).

Fixed mesh strategy can always save significant computational time compared to dynamic mesh algorithm in numerical modeling. However it may have difficulty in getting sufficient accuracy near the boundaries of moving bodies. DD is frequently used to combine the advantages of different methods. By DD is meant a domain subdivision in zones solved with different methods. A DD can be applied in time (i.e. temporal subdivision) and/or in space and is chosen to ensure a good compromise between reliability of the solution and numerical efficiency. Mathematically, the DD algorithms are used for solving partial differential equations. An overview of DD approaches in this framework can be found in Quarteroni and Valli (1999). Cai (2003) introduces overlapping DD methods and implements them for solving partial differential equations. DD strategy has already been used in hydrodynamics. It is mostly applied for studying fluid—structure interaction problems. Near structures with sharp corners or anyway with relevant flow separation, as well as, in fluid regions with important breaking and fragmentation of the air—water interface, NS solvers are necessary. However they require heavy computational time. So the DD strategy is used for combining viscous and potential flow solvers. Colicchio et al. (2006) developed a DD algorithm to deal with two-phase flows and applied it to a dam breaking followed by a water impact against a vertical wall. Near the downstream wall a NS solver with Level-Set technique for the air—water interface was adopted, while the upstream flow evolution was modeled with a fully-nonlinear Boundary Element Method (BEM). In this case, the common boundary of the two subdomains contains the air—water interface and the use of an overlapping was chosen so to ensure a more robust coupling algorithm though allowing a smoother transition from the BEM to the NS solution. Colicchio et al. (2010) applied a 3D DD strategy for studying the water-on-deck phenomena. Here a domain subdivision is proposed using a multi-block strategy to combine the advantage of fixed mesh and boundary-fitted mesh algorithms for problems dominated by potential-flow effects. Therefore the same solution method, the HPC method, is used in the whole fluid domain.

The main objective of this paper is to contribute in improving the original HPC method in terms of efficiency and accuracy. Here we limit ourselves to 2D flow problems. However, the proposed solutions strategies would suit for 3D cases as well.

Firstly, we present in Section 2 a brief introduction of the basic principles and equations of the original HPC method. In Section 3, we propose a flux method to enforce Neumann boundary conditions. Different ways of applying the flux method are compared by studying a Boundary Value Problem (BVP) in both shallow and deep water conditions. In Section 4, we describe the boundary-fitted mesh strategy which is used in the original HPC method. Then we develop two specific treatments of fixed mesh algorithm. They are tested by solving one time step of a regular sinusoidal propagating wave. In Section 5, a fully nonlinear wave tank is developed. The fixed mesh strategy is verified by generating a solitary wave. Predicted wave profile and corresponding CPU time are compared against reference solutions. Furthermore, a multi-block algorithm combining boundary-fitted mesh and fixed mesh is proposed in Section 6. The former is used for the domain near Neumann boundaries like wave maker and floating body. The fixed mesh is applied for fluid regions away from that kind of boundaries. The multi-block mesh strategy is adopted to the numerical wave tank in Section 5 and verified by comparison of the wave elevation and computational time with the results from the single mesh approach. In the last section, the main conclusions and future steps are given.

2. Original harmonic polynomial cell method

This section gives a brief introduction of the HPC method. For detailed description, see Shao and Faltinsen, (2012). In 2D cases, an Earth-fixed (x, y) reference frame is considered and the water domain is divided into quadrilateral cells. Each cell contains four quadrilateral elements and 9 grid nodes. Globally, each grid node is associated with a double index, say (i, j), with i the index in x direction, ranging between 1 and $nx + 1$, and j the index in y direction, ranging between 1 and $ny + 1$. Locally in each cell, the boundary nodes are numbered from 1 to 8. Meanwhile, the 9th node is the center of the cell, as shown in Fig. 1.

The velocity potential ϕ within the cell is approximated by the following interpolation function, in terms of the velocity potential ϕ_i at the eight boundary nodes

$$\phi(x,y) = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\, f_j(x,y)\right]\phi_i. \tag{1}$$

Here $f_j(x, y)$ are the eight harmonic polynomials adopted in 1, i.e. $f_1(x,y) = 1$; $f_2(x,y) = x$; $f_3(x,y) = y$; $f_4(x,y) = x^2 - y^2$;

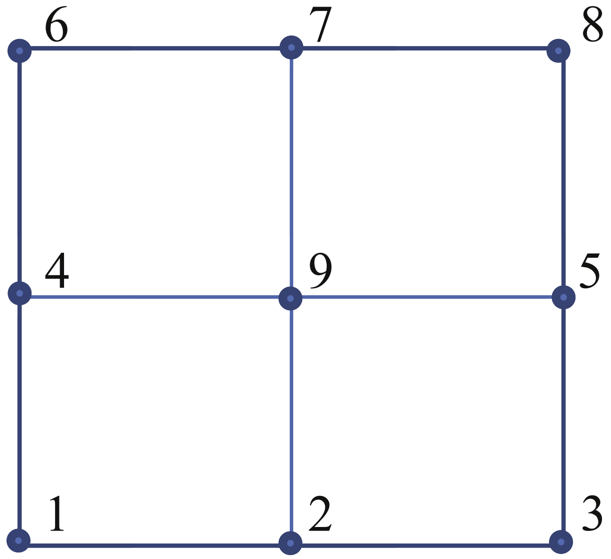


Fig. 1. Definition of local index for cells.

$f_5(x,y) = xy$; $f_6(x,y) = x^3 - 3xy^2$; $f_7(x,y) = 3x^2y - y^3$; $f_8(x,y) = x^4 - 6x^2y^2 + y^4$. $c_{j,i}$ are elements of the inverse of the matrix $[D]$, whose elements $d_{i,j} = f_j(x_i, y_i)$. Since the harmonic polynomials satisfy the Laplace equation everywhere in the space, the velocity potential also satisfies the Laplace equation automatically.

Considering that the origin of the local coordinate system is set at the position of the 9th node, the velocity potential of the 9th node in each cell can be simplified as

$$\phi_9 = \phi(x = x_9 = 0, y = y_9 = 0) = \sum_{i=1}^{8} c_{1,i}\phi_i. \tag{2}$$

Given that the cells are overlapped, Eq. (2) connects all the grid nodes. The Dirichlet boundary conditions are satisfied by enforcing the velocity potential at the corresponding boundary nodes. The Neumann boundary conditions are fulfilled by taking the normal derivate of Eq. (1) at the boundary nodes, i.e.

$$\frac{\partial\phi}{\partial n}(x,y) = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\nabla f_j(x,y)\cdot\boldsymbol{n}(x,y)\right]\phi_i, \tag{3}$$

where $\boldsymbol{n}(x,y)$ is the unit normal vector of the Neumann boundary at the node (x,y).

3. Flux method for Neumann boundary condition

3.1. Basis of flux method

Here an alternative way of enforcing the Neumann boundary condition is proposed. This uses a distributed flux. The main idea of flux method is to consider the flux passing through a certain length of Neumann boundary as the boundary condition. Assuming a generic geometry of the Neumann boundary, a simple way to calculate the flux is obtained approximating locally the boundary as piece-wise straight-line segments. Fig. 2 provides an example of the simplified piece-wise straight-line subdivision of a Neumann boundary portion. Let us assume that we want to enforce the boundary condition at the node B, then the two closest nodes A and C along the Neumann boundary are used to define an integration line $(A - B - C)$ for the flux estimation.

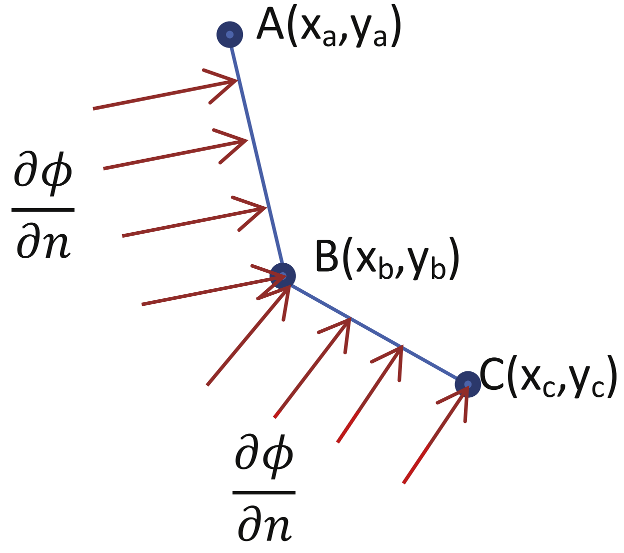


Fig. 2. Simplified piece-wise straight line subdivision of Neumann boundary.

Integrating along the two elements $(A - B - C)$, the flux Q can be expressed as

$$Q = \int\limits_{\substack{Neumann\\ boundary\ segments}} \frac{\partial\phi}{\partial n}(x,y)\mathrm{d}l = \int\limits_{A-B-C} \frac{\partial\phi}{\partial n}(x,y)\mathrm{d}l. \tag{4}$$

Substituting the normal velocity given by Eq. (3) into Eq. (4), gives

$$\begin{aligned} Q &= \int\limits_{A-B-C} \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\nabla f_j(x,y)\cdot\boldsymbol{n}(x,y)\right]\phi_i\mathrm{d}l \\ &= \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\int\limits_{A-B-C}\nabla f_j(x,y)\cdot\boldsymbol{n}(x,y)\mathrm{d}l\right]\phi_\iota. \end{aligned} \tag{5}$$

Note that Eq. (5) is formulated for general shape of the elements, i.e. the flux method is even suitable for body surfaces with discontinuous normal vectors. To cope with singular potential flow characteristics at sharp corners, Liang et al. (2015) have proposed a local potential flow solution, which was coupled with the original HPC method based on domain decomposition strategy. This has not been implemented in the present work. Compared with the way Neumann boundary conditions are satisfied in the original HPC method, the flux method is more consistent, i.e. it leads to a weak formulation (see e.g., Atkinson and Han, 2009) of the local boundary condition and therefore has less stringent smoothness requirements. In the original HPC method, the Neumann boundary conditions are satisfied on either side of the discontinuous surface at the sharp corners. In a general case when the Neumann boundaries are not straight lines, numerical integration method, for instance, Gaussian quadrature may also be applied to calculate the flux in Eq. (5). The flux through a straight line as both cases shown in Fig. 3 can be simplified as

$$\begin{aligned} Q_{AB} &= \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\int\limits_{A-B}\nabla f_j(x,y)\cdot\boldsymbol{n}(x,y)\mathrm{d}l\right] \\ \phi_i &= \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}F_j\right]\phi_i, \end{aligned} \tag{6}$$

where $F_j = \int\limits_{A-B}\nabla f_j(x,y)\cdot\boldsymbol{n}(x,y)\mathrm{d}l$, i.e. $F_1(x_a,x_b,y_a,y_b) = 0$; $F_2(x_a,x_b,y_a,y_b) = y_b - y_a$; $F_3(x_a,x_b,y_a,y_b) = x_a - x_b$; $F_4(x_a, x_b,y_a,y_b) = 2x_by_b - 2x_ay_a$; $F_5(x_a,x_b,y_a,y_b) = \frac{1}{2}(x_a^2 - x_b^2 + y_a^2 - y_b^2)$; $F_6(x_a,x_b,y_a,y_b) = -3x_a^2y_a + 3x_b^2y_b + y_a^3 - y_b^3$; $F_7(x_a, x_b,$

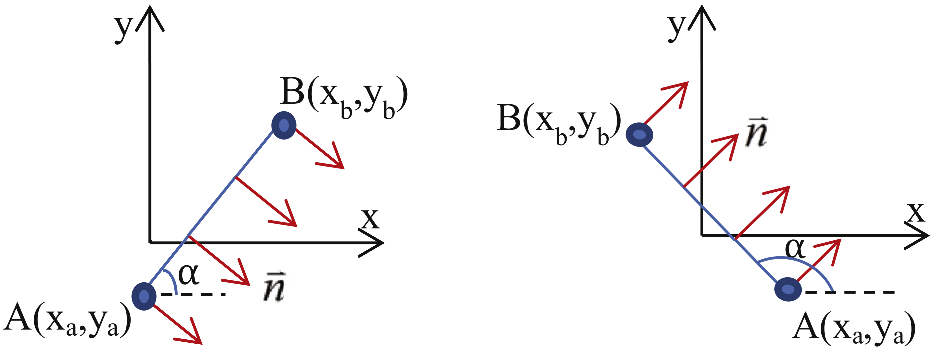


Fig. 3. Definition of flux through straight segments. Left: Positive direction of flux for $0 \le \alpha \le \frac{\pi}{2}$. Right: Positive direction of flux for $\frac{\pi}{2} < \alpha < \pi$.

$y_a, y_b) = x_a^3 - x_b^3 - 3x_a^2 y_a + 3x_b y_b^2$; $F_8(x_a, x_b, y_a, y_b) = 4(x_a y_a^3 - x_a^3 y_a - x_b y_b^3 + x_b^3 y_b)$. For flux opposites to cases given in Fig. 3, $-F_j$ is adopted instead. Then Eq. (5) can be solved by calculating the flux through two elements separately.

For flux through vertical and horizontal boundaries and corner with right angle as shown in Fig. 4, Eq. (5) can be simplified as follows.

- Flux through vertical boundary extending from $-\Delta y$ to Δy:

$$Q_V = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i} \int_{-\Delta y}^{\Delta y} \frac{\partial f_j}{\partial x} \mathrm{d}y\right]\phi_i = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i} F_j^V\right]\phi_i, \quad (7)$$

where $F_j^V = \int_{-\Delta y}^{\Delta y} \frac{\partial f_j}{\partial x}\mathrm{d}y$, i.e. $F_1^V(x) = 0$; $F_2^V(x) = 2\Delta y$; $F_3^V(x) = 0$; $F_4^V(x) = 4x\Delta y$; $F_5^V(x) = 0$; $F_6^V(x) = 6x^2\Delta y - 2\Delta y^3$; $F_7^V(x) = 0$; $F_8^V(x) = 8x^3\Delta y - 8x\Delta y^3$.

- Flux through horizontal boundary extending from $-\Delta x$ to Δx:

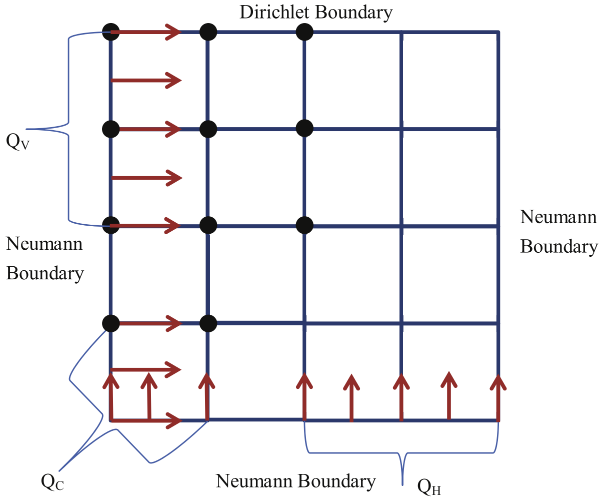


Fig. 4. Sample of the flux through vertical, horizontal and corner Neumann boundary.

$$Q_H = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i} \int_{-\Delta x}^{\Delta x} \frac{\partial f_j}{\partial y} \mathrm{d}x\right]\phi_i = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i} F_j^H\right]\phi_i, \quad (8)$$

where $F_j^H = \int_{-\Delta x}^{\Delta x} \frac{\partial f_j}{\partial y}\mathrm{d}x$, i.e. $F_1^H(y) = 0$; $F_2^H(y) = 0$; $F_3^H(y) = 2\Delta x$; $F_4^H(y) = -4y\Delta x$; $F_5^H(y) = 0$; $F_6^H(y) = 0$; $F_7^H(y) = 2\Delta x^3 - 6y^2\Delta x$; $F_8^H(y) = 8y^3\Delta x - 8y\Delta x^3$.

- Flux through corner boundary which consists of a vertical and a horizontal segment extending from $-\Delta y$ to 0 and 0 to Δx respectively:

$$Q_C = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i}\left(\int_{-\Delta y}^{0} \frac{\partial f_j}{\partial x}\mathrm{d}y + \int_{-\Delta x}^{0} \frac{\partial f_j}{\partial y}\mathrm{d}x\right)\right]$$
$$\phi_i = \sum_{i=1}^{8}\left[\sum_{j=1}^{8} c_{j,i} F_j^C\right]\phi_i, \quad (9)$$

where $F_j^C = \int_{-\Delta y}^{0} \frac{\partial f_j}{\partial x}\mathrm{d}y + \int_{-\Delta x}^{0} \frac{\partial f_j}{\partial y}\mathrm{d}x$, i.e. $F_1^C(y) = 0$; $F_2^C(y) = \Delta y$; $F_3^C(y) = \Delta x$; $F_4^C(y) = 2x\Delta y - 2y\Delta x$; $F_5^C(y) = -\frac{1}{2}\Delta y^2 - \frac{1}{2}\Delta x^2$; $F_6^C(y) = 3x^2\Delta y - \Delta y^3 + 3y\Delta x^2$; $F_7^C(y) = -3x\Delta y^2 + \Delta x^3 - 3y^2\Delta x F_8^C(y) = 4x^3\Delta y - 4x\Delta y^3 - 4y\Delta x^3 + 4y^3\Delta x$.

3.2. Case study on a rectangular box

Here we study a mixed Dirichlet–Neumann BVP in a 2D rectangular box by the original HPC method and the HPC method combined with flux method. This test case is relevant for numerical wave-tanks based on potential-flow theory because it is representative of the type of BVP to be solved for the velocity potential within the Eulerian step of the Mixed Eulerian–Lagrangian (MEL) solution algorithm (see Ogilvie, 1967). The box length L is 40 times the box height h which is the same as what was studied by Shao and Faltinsen (2012), i.e. $h = 2$m. The origin of the coordinate system is located at the mid point of the Dirichlet boundary. The BVP is shown in Fig. 5.

The Dirichlet boundary condition on the top surface and the Neumann boundary condition on the other three surfaces are given, respectively, by the velocity potential function

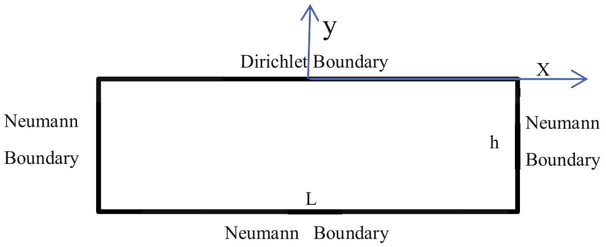


Fig. 5. BVP of a 2D rectangular box.

$$\phi = \cosh[k(y+h)]\cos(kx), \tag{10}$$

and its local normal derivative. Expression (10) represents the spatial behavior of a Airy wave with wave number k, in a water region with depth h here representing the vertical size of the box. The coordinate system is shown in Fig. 5. x and y are the horizontal and vertical coordinates, respectively. Here squared meshes are used for the HPC solutions.

In the following, we examine the effect of applying the flux method (1) for all Neumann boundaries or (2) only at the corners between Neumann boundaries with the original HPC method for the other boundary points. Approach (2) is indicated as ‘corner only’. Within strategy (1), both overlapped flux and non-overlapped flux are studied. Fig. 6 demonstrates three samples of applying the flux as boundary condition.

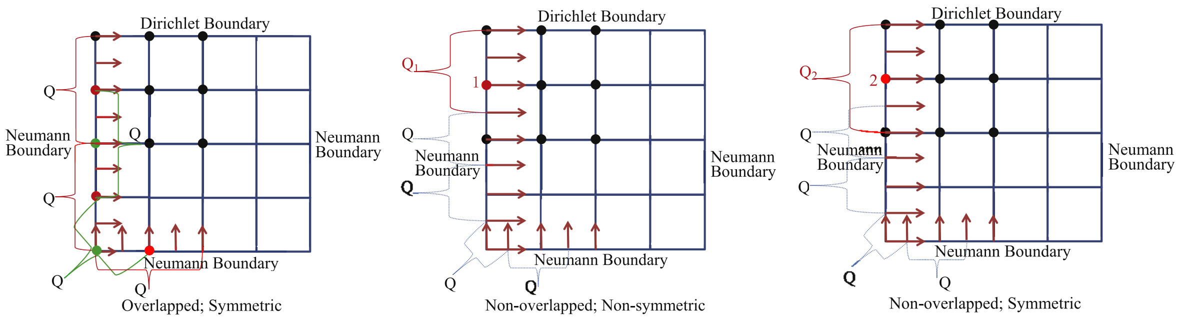
3.3. Results

The accuracy of the flux method is verified by comparison of the L_2 errors with the original HPC method, in terms of a convergence study. The L_2 errors are defined as

$$e_{L_2} = \sqrt{\sum_{i=1}^{N}\left(f_i^{num} - f_i^{an}\right)^2 \Big/ \sum_{i=1}^{N}\left(f_i^{an}\right)^2}. \tag{11}$$

Here N is the number of the total grid points, f_i^{num} and f_i^{an} are the numerical and analytical solutions for the velocity potential of the grid points, respectively.

Fig. 7 shows the results of convergence study for the original HPC and the present implementation with flux method as overlapped, non overlapped-1, non-overlapped-2 and at the corner only in terms of L_2 errors versus number of unknowns for both shallow and deep water condition. The left figure corresponds to a shallow water case with $kh = 1.0$, while the right one is for deep water condition with $kh = 6.28$. All five solutions give similar rate of convergence, yet have different error level. The ‘corner only’ solution gives the lowest error, namely, applying flux only for the bottom corner nodes can improve the accuracy. Moreover as it is expected ‘non-overlapped-2’ shows better convergence than ‘non-overlapped-1’, since ‘non-overlapped-2’ avoids the asymmetry. Comparison between ‘non-overlapped-2’ and ‘overlapped’ implies that the use of non-overlapped flux leads to higher order of accuracy. This is probably because the flux is the integration of normal velocity, i.e. it is like expressing the local condition as an average over a certain length with weighting function equal to one. With the decreasing of integration interval, the constraint becomes stronger. This is also supported by the ‘original HPC’, in which the velocity conditions can be regarded as flux conditions with integration interval reducing to one point or with the same integration interval but with a weighting Dirac-Delta function in the collocation point. This leads to the strongest constraint and to high accuracy compared with the overlapped and non-overlapped cases. There is an obvious inconsistency between the two plots of Fig. 7 concerning the fact that the ‘original HPC’ shows higher error level than the ‘non-overlapped-2’ in shallow water, while ‘non-overlapped-2’ gives higher error level in deep water. Mainly, there are two reasons that may cause the inconsistency. One is that for $kh = 6.28$ the velocity potential has much larger gradients in vertical direction which makes the L_2 error more sensitive to the numerical approximation of the Neumann boundary condition. More in detail, the error will enlarge when using an averaged flux instead of enforcing the local value of the normal velocity because the averaged flux would be less consistent with the actual wave-velocity distribution in vertical direction. The other reason is that the positive effect of



Overlapped; Symmetric Non-overlapped; Non-symmetric Non-overlapped; Symmetric

Fig. 6. Neumann boundary condition: Left: Sample of overlapped flux, i.e. flux through an entire boundary of a cell (indicated as ‘overlapped’). Middle and right: samples of non-overlapped flux indicated, respectively, as ‘non overlapped-1’ and ‘non overlapped-2’. They differ for the way the flux is treated near the intersection with Dirichlet Boundary, the middle one counts the flux through an asymmetric part (Q_1) in order to cover all the Neumann boundary and the right one counts the flux through a symmetric part (Q_2). Please be aware that Eq. (7) integrates from $-\Delta y$ to Δy, which does not apply directly for all the non-overlapping cases.

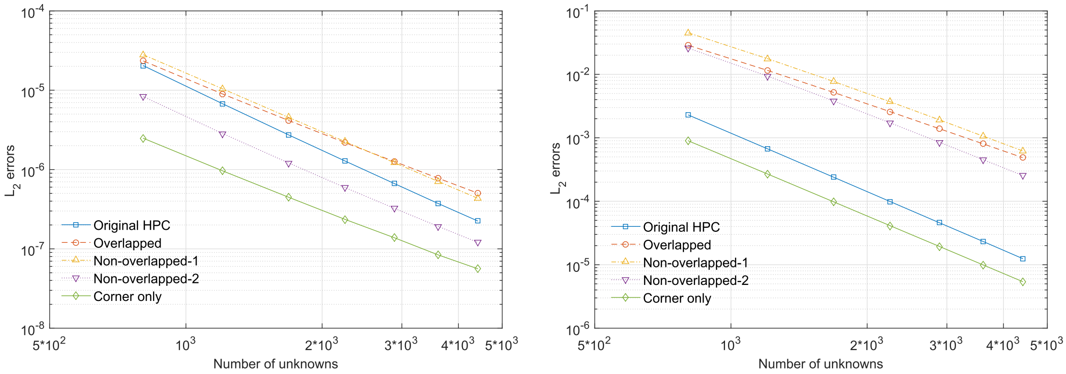


Fig. 7. Comparison of L_2 errors. Left: $kh = 1.0$. Right: $kh = 6.28$. ‘overlapped’ represents the results with overlapped flux as given in Fig. 6 (left). ‘non-overlapped-1’ represents the results with non-overlapped flux as given in Fig. 6 (middle). ‘non-overlapped-2’ represents the results with the modified non-overlapped flux as given in Fig. 6 (right). ‘corner only’ represents the results with the flux method applied only at the corners between Neumann boundaries.

using the bottom nodes within the flux method has much larger influence on the global error L_2 in shallow water than in deep water.

Local errors are also of interest and the absolute local errors are defined as:

$$error = \frac{abs(\phi_{num} - \phi_{an})}{max(abs(\phi_{an}))}. \tag{12}$$

Figs. 8 and 9 give an example of the distribution of absolute errors in entire domain for the ‘original HPC’ and ‘corner only’ method respectively. It is obvious that applying the flux method for the corners not only enhance the accuracy at corner nodes but also in the regions around them. Although the distribution of absolute errors varies from case to case, we can identify some common features. The most important is that high errors occur along the Neumann boundaries. It is fair to say that the difference of error level between the fluid near Neumann boundaries and the fluid in the inner domain is caused by the high numerical error at Neumann boundaries. In another word, if we reduce the error of the nodes at the Neumann boundaries, the error of the nodes nearby will decrease correspondingly.

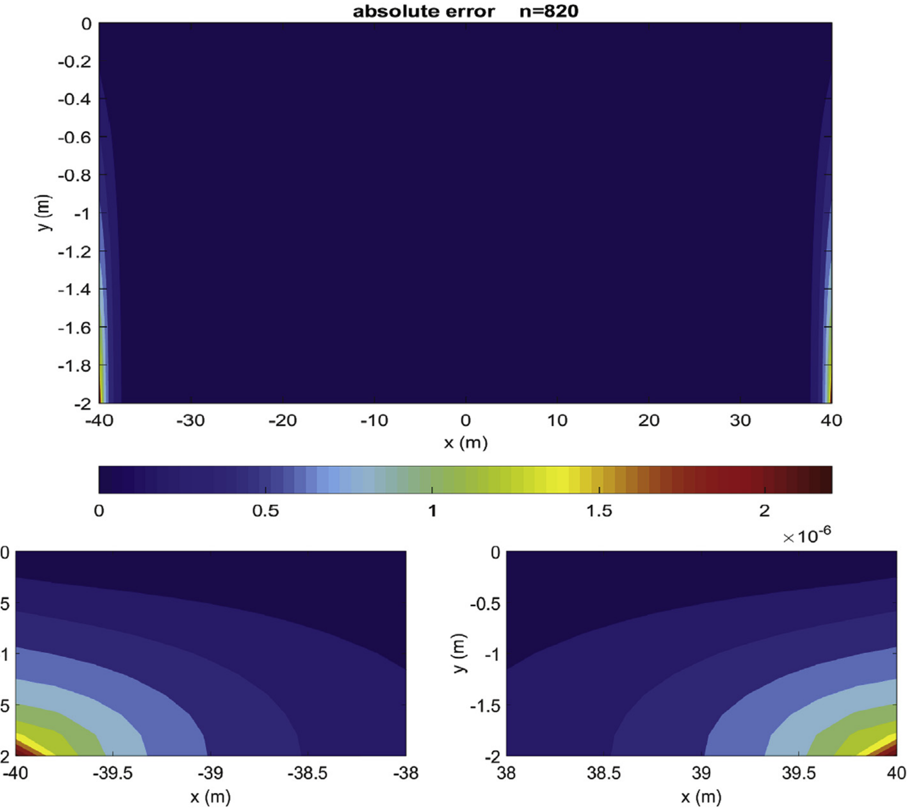


Fig. 8. Distribution of absolute error for the velocity potential from the original HPC method. $kh = 1.0$.

Please cite this article in press as: Zhu, W., et al., Improved HPC method for nonlinear wave tank, International Journal of Naval Architecture and Ocean Engineering (2017), http://dx.doi.org/10.1016/j.ijnaoe.2017.03.009

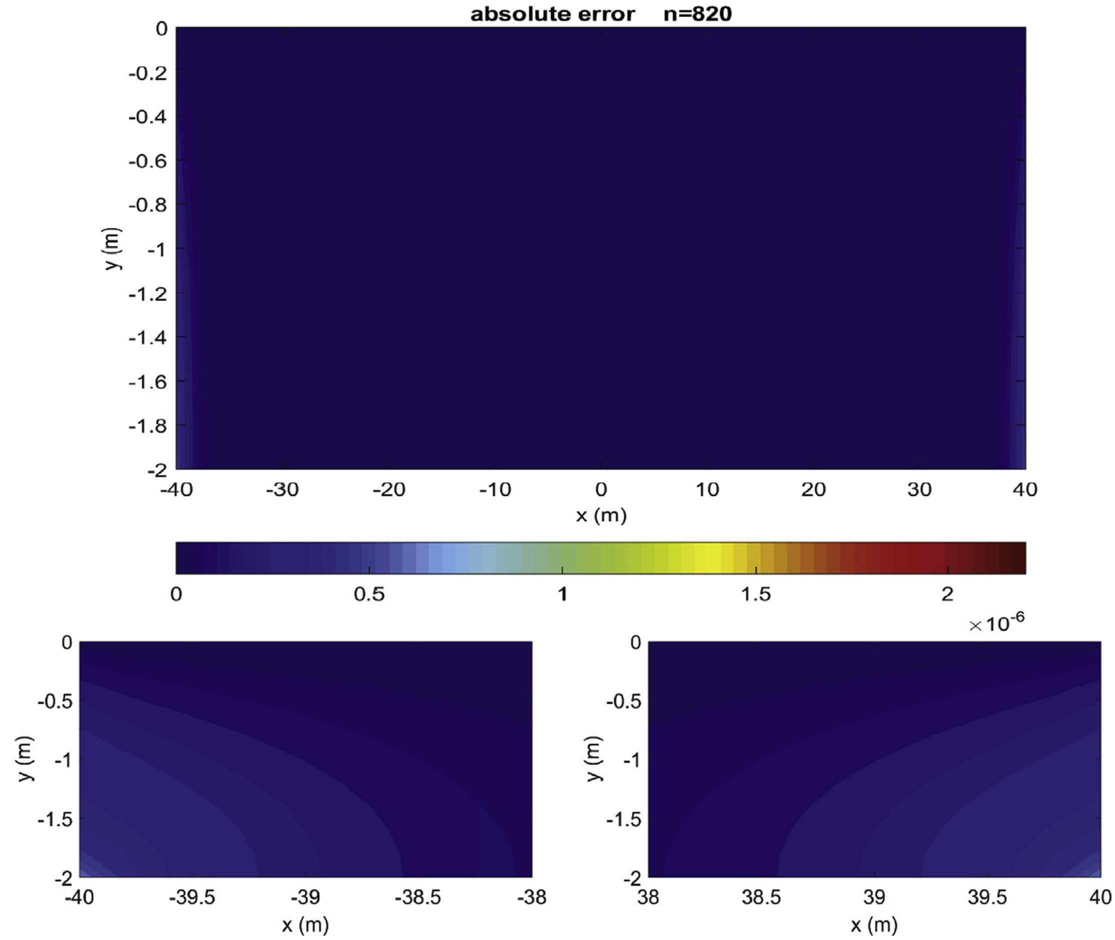


Fig. 9. Distribution of absolute error for the velocity potential from the ‘corner only’ solution. $kh = 1.0$.

As a result, the difference of the L_2 errors between the different solutions studied above is mainly determined by the high local errors. The absolute errors along the left Neumann boundary are plotted in Figs. 10 and 11. ‘corner only’ gives the lowest local errors consistently with the L_2 errors. Compared with the original HPC method, the flux method can reduce the error at the corner between Neumann boundaries remarkably in both shallow and deep water condition. The extremely large error of ‘non-overlapped-1’ at $y = -0.2$m for both $kh = 1.0$ and $kh = 6.28$ confirms that asymmetric mesh will lead to additional inaccuracy.

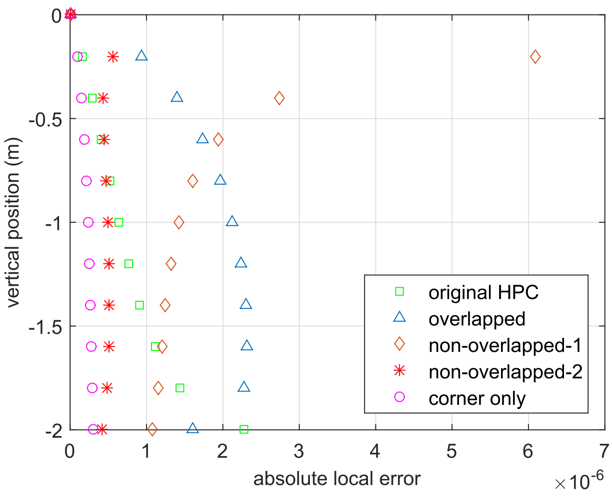


Fig. 10. Distribution of absolute error for the velocity potential at left Neumann boundary where $x = -40$m. $kh = 1.0$.

3.4. Summary

Through the case study, it can be concluded that using the flux as boundary condition can improve profoundly the accuracy at geometrical singularities between Neumann boundaries than applying normal velocity at either side of the Neumann boundaries directly. However, enforcing the normal velocity leads to more accurate results along regular Neumann boundaries. So it is suggested to apply the flux condition only for intersection points between Neumann boundaries with different normal velocity and/or with different slope. This approach would be beneficial for studying wave body interaction problems if the body has sharp corners, because near geometrical singularities potential-flow methods have problems of numerical convergence.

Please cite this article in press as: Zhu, W., et al., Improved HPC method for nonlinear wave tank, International Journal of Naval Architecture and Ocean Engineering (2017), http://dx.doi.org/10.1016/j.ijnaoe.2017.03.009

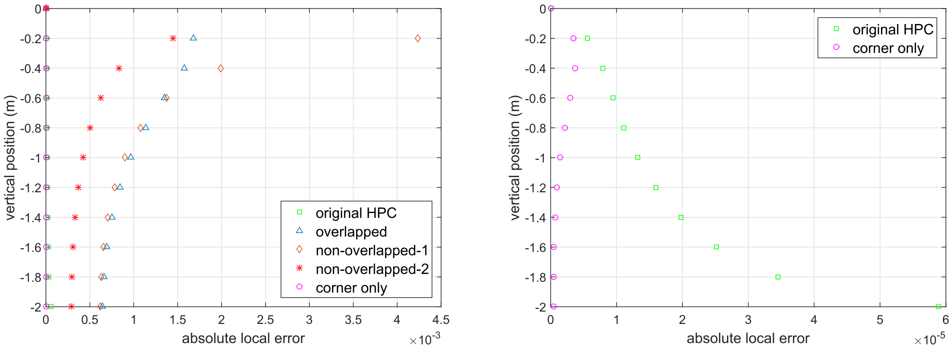


Fig. 11. Distribution of absolute error for the velocity potential at left Neumann boundary where $x = -40$m. $kh = 6.28$. Left: Full view, Right: Enlarged view.

4. Mesh strategies

Two basic mesh strategies are studied in this paper for the HPC solver when implemented in a MEL solution algorithm, and so within a space-time simulation, of numerical wave-tank problems. One is the boundary-fitted mesh as done by Shao and Faltinsen (2012). The other is using a fixed mesh within which the free surface is immersed.

4.1. Boundary-fitted mesh

The boundary-fitted mesh is easy to implement since the velocity potential for all the grid points can be solved directly. An example of the mesh is shown in Fig. 12. It can be easily found that boundary-fitted meshes lead to non-uniform formulation for the cells, in the meaning that the coefficient $c_{i,j}$ are different in each cell and need to be calculated for any cell. Moreover, the coefficients need to be updated at any studied time instant in case of moving boundaries in the examined reference frame. This will also limit the computational efficiency.

4.2. Fixed mesh

In order to improve the efficiency, a boundary immersed or embedded fixed mesh is developed and tested. The horizontal length of computational domain is coincides with the fluid domain extension, while the height should be set larger than the highest wave elevation in order to get the wave surface immersed. An example of this type of mesh is shown in Fig. 13.

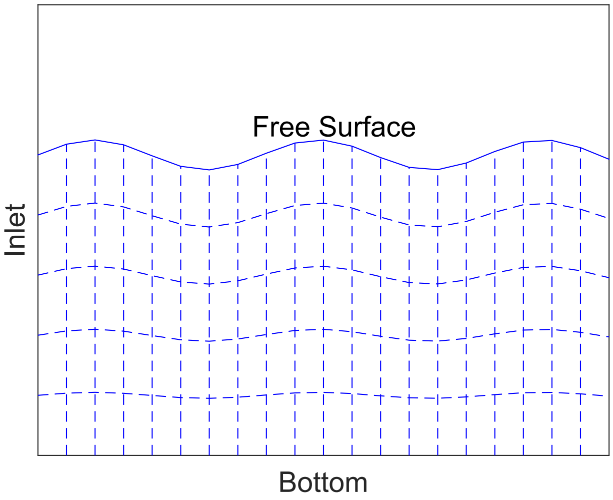


Fig. 12. An example of boundary-fitted mesh.

In this paper, all the grids are of the same size with $dx = dy$. dx and dy are mesh size in horizontal and vertical direction, respectively. After using fixed meshes, the free surface condition is no longer carried by the grid points. So free surface tracking nodes, which are the intersection points between the free surface profile and vertical grid lines as shown in Fig. 13, are introduced additionally to carry the Dirichlet boundary conditions. In time domain analysis they are updated according to later Eqs. (16) and (17) using MEL algorithm. Differently from the boundary-fitted mesh, there are cells containing parts out of the water domain when using a fixed mesh. The velocity potential on the grid points out of water can not be calculated using the basis HPC method for the water domain but they might need to be used for the water evolution and here they are found in an artificial manner preserving the physical water-flow evolution. In particular, the velocity potential in air is extended from the water solution by taking the grid points on or below the free surface as center points which are governed by Eq. (2). More details are given below. To extend the velocity potential into air domain, two treatments are studied and compared.

4.2.1. Treatment 1

In this case, the layer consists of the grid nodes right above the free surface tracking nodes at each vertical grid line is named as artificial layer 1 (see Fig. 13). The Dirichlet boundary conditions are enforced by Eq. (1), which can be seen as constrains for the nodes of artificial node layer 1. Specifically, let us assume j^* as the index in y direction for the node in the artificial node layer 1 at a generic horizontal location with index i. If this node is on the left boundary ($i = 1$), it will be associated to the cell with 9th node ($2, j^* - 1$); if on the right boundary ($i = nx + 1$), to the cell with 9th node ($nx, j^* - 1$); otherwise to the cell with 9th node ($i, j^* - 1$). Constrains for the grid points above the artificial node layer 1 are given by

$$\phi_{i,j} = \phi_{i,j^*}, \quad for \quad j > j^*. \tag{13}$$

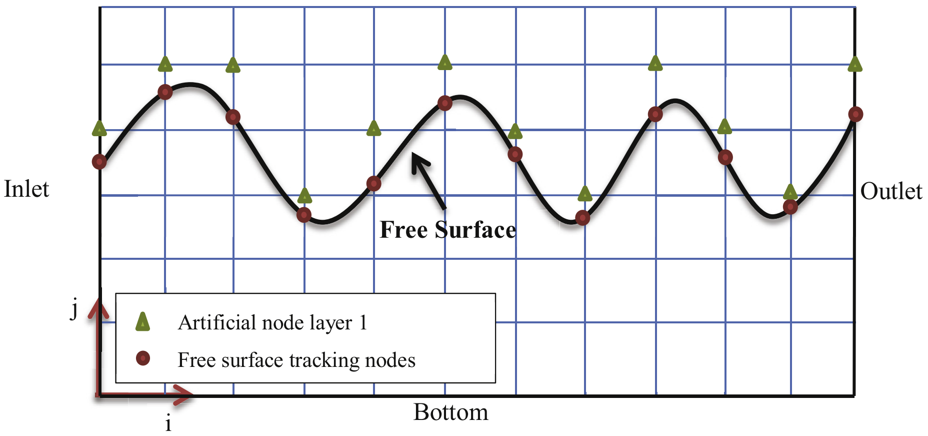


Fig. 13. An example of fixed mesh with free surface immersed. Treatment 1 for the extension of ϕ into the air domain.

A similar approach is proposed by Hanssen et al. (2016) and is further analyzed by Ma et al. (2016). In Eq. (13), i and j represent the column and row number of the grid points in the mesh shown in Fig. 13, respectively. With the information given above, the equations for the local velocity potential are available at all required positions. Once ϕ is available, the velocity can be estimated everywhere. This allows integrating in time the free-surface boundary conditions and also updating the free-surface elevation. As for the classical HPC method, here this is done within a semi-Lagrangian approach (i.e. following just the vertical motion of the free-surface particles).

4.2.2. Treatment 2

In the second treatment, artificial node layer 2 is added based on Treatment 1 (see Fig. 14). The Dirichlet boundary conditions, i.e. the velocity potential at the corresponding free surface tracking nodes are used as the constrains for the nodes of artificial node layer 2. Accordingly, for the left boundary node on the artificial node layer 2, i.e. $(1,j^*+1)$, global index of the 9th node of the associated cell is $(2,j^*)$. For the right boundary node on the artificial node layer 2, i.e. $(nx+1,j^*+1)$, global index of the 9th node of the corresponding cell is (nx,j^*). For the other nodes (i,j^*+1) on the artificial node layer 2, the global index of the 9th node of the corresponding cell is (i,j^*). By doing so, we can preserve the accuracy at free surface when extending the potential into air domain. Constrains for the grid points above the artificial node layer 2 are given by

$$\phi_{i,j}=\phi_{i,j^*+1}, \quad for \quad j>j^*+1. \tag{14}$$

Because the two treatments are associated with the same grid, the global matrices for the two treatments have the same size. However, the use of layer 2 makes the global matrix more complex, i.e. there are more non-zero terms leading to a less efficient solution algorithm.

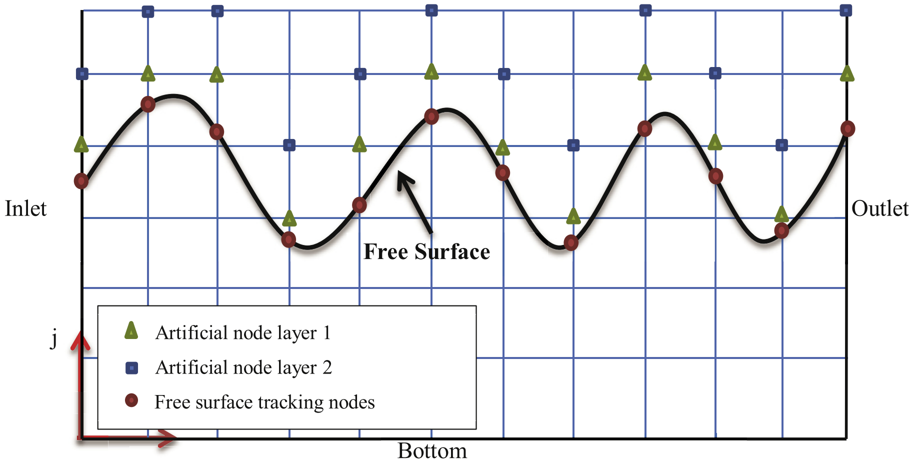


Fig. 14. Treatment 2 for the extension of ϕ into the air domain.

4.3. Test case

Here, we examine the effectiveness of the mesh strategies by solving one time instant of a regular sinusoidal propagating wave, which is governed by

$$\phi = \frac{g\zeta_a}{\omega}\frac{\cosh k(h+y)}{\cosh kh}\cos(\omega t - kx), \qquad (15)$$

with waves and tank parameters chosen as $\zeta_a = 0.1$m, $h = 2.0$m, $k = \pi$. The calculation domain is from $x = 0$ to 80m. By taking $\omega t = 0.1k$, we studied a case with a cell having five nodes outside the water domain as shown in Fig. 15. The highlighted cell is used when setting the control equation for the node 6 marked in the cell.

Fig. 16 presents the velocity potential for three columns of nodes which are closest to the left Neumann boundary corresponding to the cell shown in Fig. 15. It is obvious that using the fixed mesh allows extending the velocity potential from water to air smoothly i.e. following the trends of variation in water. It can also be found that all the results fit the analytical value well, except that the velocity potential of node 4 in Fig. 15 has a deviation when using treatment 1 and similarly happens for node 6 when using treatment 2. The latter node has the same velocity potential as node 4 for treatment 1. Since the values of the velocity potential at these two nodes will be used for calculating the evolution of the free surface, it may cause problems which are further studied in Section 5. In treatment 1 the nodes 7 and 8, whose velocity potential is forced to be equal to the nodes just below, will also cause the same problem when calculating evolution of the left free surface point right above node 4.

In conclusion, the results obtained from Treatment 1 have the largest discrepancies with the analytical results. Small errors are even found for nodes in water. Obvious discrepancies between the results acquired from Treatment 2 can only be found on the nodes of artificial layer 2 or above, which merely affect the evolution of wave surface. No obvious discrepancies are found between the results from boundary-fitted mesh algorithm and the analytical solution.

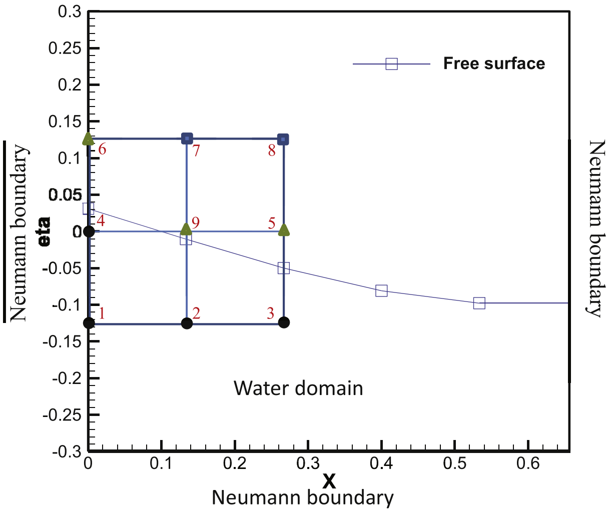


Fig. 15. A cell with five nodes outside the water domain.

5. Nonlinear numerical wave tank

The model of the 2D numerical wave tank is demonstrated in Fig. 17. The tank is 80m long with water depth $h = 2$m. There are several ways to generate a wave in the numerical wave tank (see Kim et al., 1999). Here, feeding numerical velocity on inflow boundary is used for simplicity. The velocity potential is calculated using the improved HPC method.

The fully nonlinear free-surface conditions are given by

$$\frac{d\phi}{dt} = -g\eta - \frac{1}{2}\nabla\phi\cdot\nabla\phi + \boldsymbol{v}\cdot\nabla\phi, \quad on \quad y = \eta \qquad (16)$$

$$\frac{d\eta}{dt} = \frac{\partial\phi}{\partial y} + (\boldsymbol{v} - \nabla\phi)\cdot\nabla\eta, \quad on \quad y = \eta. \qquad (17)$$

When $\boldsymbol{v} = \left(0, \frac{d\eta}{dt}\right)$ is used, the free surface particles are tracked by the so called semi-Lagrangian approach, which was first introduced by Ogilvie (1967). This means that plunging waves cannot be simulated but features of the free-surface treatments 1 and 2 proposed here could be useful also for plunging-wave scenarios. In the latter case, it is more suitable to extend the velocity potential normally to the free-surface instead of vertically.

In Eqs. (16) and (17), the spatial derivatives of the velocity potential are calculated by taking the derivatives of Eq. (1). In order to estimate the spatial derivative of the wave elevation, cubic-order spline fitting technique is used as done by Shao and Faltinsen (2012). Parallel Direct Sparse Solver (PARDISO) is employed for solving the equation system of the BVP for ϕ within the MEL strategy. Details of PARDISO can be found in Intel (2012).

Fourth order explicit Runge-Kutta scheme is adopted as the time integration method for solving Eqs. (16) and (17) and prolonging the free-surface configuration and velocity potential in time.

5.1. Solitary wave

There are several solutions for solitary waves, such as the solution of Boussinesq (1871) and Goring (1978), the 3rd order solution of Grimshaw (1971) and the 9th order solution obtained by Fenton (1972). Here the 3rd order solution is adopted. The analytical wave profile of the 3rd order solitary wave is given by:

$$\eta(x,t) = h\left[\varepsilon s^2 - \frac{3}{4}\varepsilon^2 s^2 q^2 + \varepsilon^3\left(\frac{5}{8}s^2 q^2 - \frac{101}{80}s^4 q^2\right)\right], \qquad (18)$$

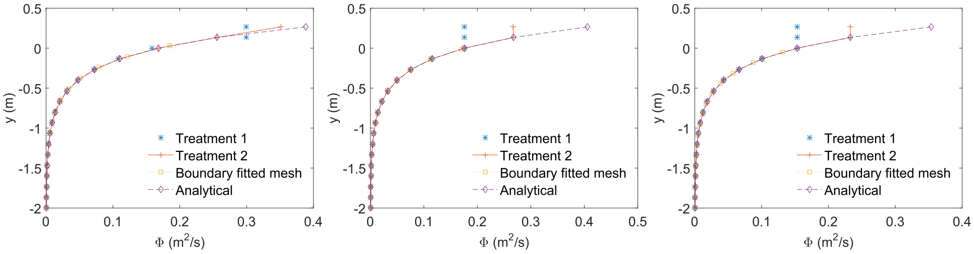


Fig. 16. Velocity potential for different mesh strategies. Left: Nodes at $x = 0.00$m (left Neumann boundary); Middle: Nodes at $x = 0.13$m; Right: Nodes at x = 0.27 m.

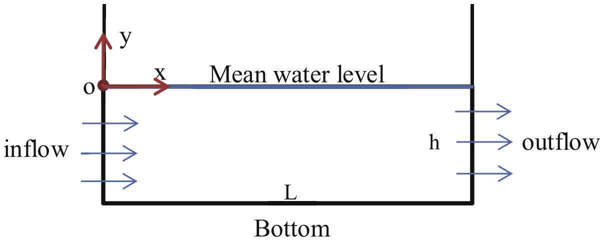


Fig. 17. Sketch of numerical wave tank.

where

$$
\begin{aligned}
s &= \operatorname{sech}\left(\frac{\alpha\left(x-c\left(t-\frac{T}{2}\right)\right)}{h}\right)\\
q &= \tanh\left(\frac{\alpha\left(x-c\left(t-\frac{T}{2}\right)\right)}{h}\right), \\
\alpha &= \sqrt{\frac{3}{4}\varepsilon}\left(1-\frac{5}{8}\varepsilon+\frac{71}{128}\varepsilon^2\right)
\end{aligned}
\tag{19}
$$

and ε is the wave height to depth ratio H/h. The wave velocity c follows the form (Mo, 2010):

$$
c = \sqrt{gh\left(1+\varepsilon-\frac{1}{20}\varepsilon^2-\frac{3}{70}\varepsilon^3\right)}, \tag{20}
$$

where g is the gravity acceleration. $g = 9.81\text{m/s}^2$ is used in this paper. Theoretically, the profile is always positive $(\eta > 0)$ and the period goes to infinite. However, in order to simulate the solitary wave numerically, we have to define the intercepts T. Here we take the precision of four significant digits. Correspondingly, T is given by solving

$$
\eta(0,0) = 0.0001h. \tag{21}
$$

In the numerical wave tank, the wave is generated by applying an inflow velocity at the left Neumann boundary i.e. inlet with $x = 0$. The velocity is constant along the vertical direction and is given by the average velocity

$$
\overline{u}(0,t) = \begin{cases} \dfrac{c\eta(0,t)}{h+\eta(0,t)}, & 0 \le t \le T \\ 0, & t > T \end{cases}. \tag{22}
$$

The wave generation starts from $t = 0$s. Meanwhile, the bottom condition where $y = -h$ is set as vertical velocity $v = 0$. The outlet condition where $x = L$ is given as $u = 0$, i.e. vertical wall condition.

Solitary wave with $\varepsilon = 0.4$ is simulated. Tables 1 and 2 gives the height and horizontal position of wave crests for the different mesh approaches at various time instants, respectively. The corresponding time step for numerical integration is $T/200$ and the mesh size at $t = 0$s is $dx = dy = h/10$. All the three methods have quite similar wave height at various time instants. The wave heights are among 1.0% to 1.3% lower than the analytical one. The loss of wave height is one of the reasons for lag of wave crests found in Table 2. Wave velocities for the three mesh approaches are the same and slightly smaller than the theoretical wave speed. Although the numerical wave crests lag behind the analytical one all the time, the lags do not always increase with time. Except for $t = 1.75T$, the lags are under 0.2m which is the horizontal mesh size. Considering that the numerical wave crests given in Table 2 are found from the discretized points every 0.2m in x direction, the inaccuracy of horizontal position of the wave crests may also caused by numerical discretization. Furthermore, Fig. 18 presents the profile of the solitary waves at $t = 2T$. Generally, the waves generated by all mesh strategies are almost the same and fit the analytical solution well. Trailing waves are observed in all

Table 1
Wave height for different mesh approaches at various time instants.

t	T	1.25 T	1.5 T	1.75 T	2 T	Mean value
Boundary-fitted mesh (m)	0.7899	0.7909	0.7913	0.7910	0.7912	0.7909
Treatment 1 (m)	0.7887	0.7892	0.7889	0.7893	0.7894	0.7891
Treatment 2 (m)	0.7906	0.7919	0.7925	0.7923	0.7926	0.7920
Analytical (m)	0.8	0.8	0.8	0.8	0.8	0.8

Table 2
Horizontal position of wave crest for different mesh approaches at various time instants.

t	T	1.25 T	1.5 T	1.75 T	2 T
Boundary-fitted mesh (m)	20.4	30.8	41	51.2	61.6
Treatment 1 (m)	20.4	30.8	41	51.2	61.6
Treatment 2 (m)	20.4	30.8	41	51.2	61.6
Analytical (m)	20.59	30.88	41.17	51.46	61.76

Please cite this article in press as: Zhu, W., et al., Improved HPC method for nonlinear wave tank, International Journal of Naval Architecture and Ocean Engineering (2017), http://dx.doi.org/10.1016/j.ijnaoe.2017.03.009

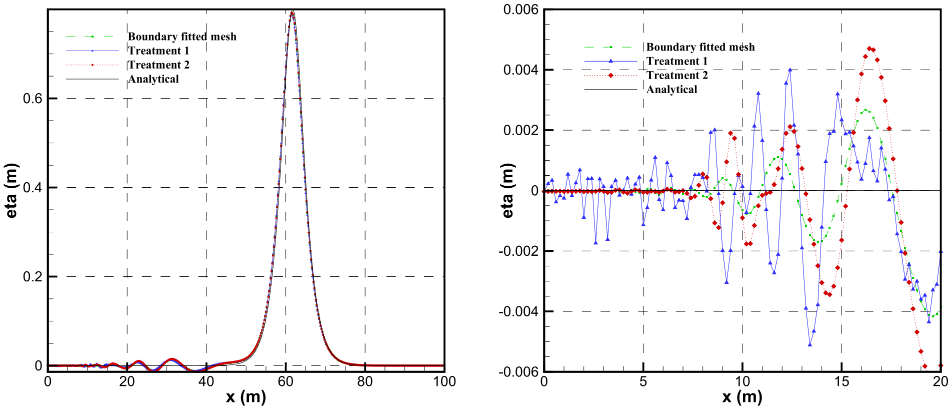


Fig. 18. Comparison of wave elevation at $t = 2T$. $H/h = 0.4$. Left: Full view, Right: Enlarged view.

the numerical results (see enlarged view in the right plot of the figure). They are due to initial conditions as has already been discussed by Zhou et al. (2016). In our model, we have seen that tail waves are much smaller for $H/h = 0.2$ (indicating that a 3rd order input is good enough for this case). For $H/h = 0.4$, it is obvious that a higher than 3rd order input is needed to get rid of the tail waves, which is not pursued in the present work but will be addressed in the future. The main focuses are on the fixed mesh strategy in Section 4 and the multi-block strategy in Section 6. Treatment 1 of the background mesh will lead to unsmoothed wave surface, which may cause break down of the simulation. Treatment 2 overcomes this problem, while it has a bit more and larger trailing waves compared with boundary-fitted mesh. The simulation is run on a computer with CPU frequency of 2.70GHz. The direct matrix solver of PARDISO runs parallelly with four processors. It is obvious from the comparison in Fig. 19 that using fixed mesh can enhance the efficiency of the numerical wave tank and the improvement increases with the number of nodes as what is expected in Subsection 4.1. Treatment 2 takes a little bit more time than Treatment 1 which can be understood by the more complex global matrix to be solved.

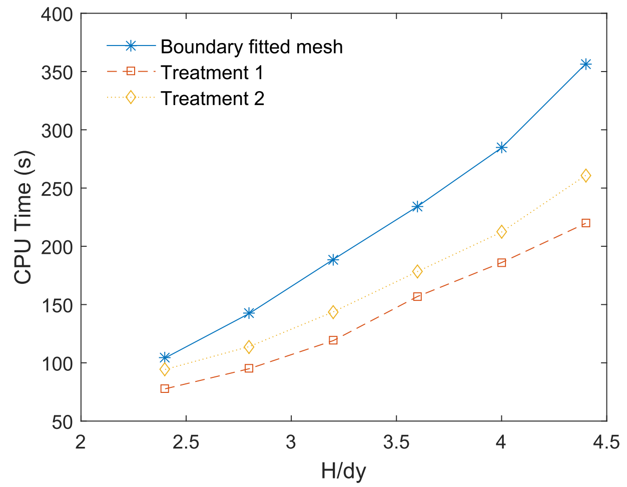


Fig. 19. Comparison of CPU time. $t = 2T$. $H/h = 0.4$.

6. Multi-block strategy

Given the accuracy of boundary-fitted mesh and the efficiency of fixed mesh, we study a multi-block strategy to combine the advantage of these two mesh approaches using the boundary-fitted grid near moving solid boundaries and the fixed grid in the rest of the fluid domain. Within present implementation, the two grids share an overlapping zone to get a smooth transition between two mesh approaches. The shortage of applying the overlapping zone is that it will increase the global matrix as the length of wave tank is increased by the length of overlapping zone.

Fig. 20 gives the sketch of a numerical wave tank using multi-block strategy with overlapped meshes. Zones A and B represent the domain using boundary-fitted mesh and fixed mesh with Treatment 2, respectively. The right boundary for domain A is inside domain B and the corresponding boundary condition is a Dirichlet condition expressing the boundary velocity potential in terms of the velocity potential from nodes in the B domain, using the HPC interpolation. Similarly it is done for the left boundary of domain B, receiving the boundary condition from the solution from domain A. In this way the problem can be solved simultaneously in the whole domain at any time instant. In the overlapping region of the two domains the spacial derivatives for the velocity potential needed in the free-surface boundary conditions (i.e. Eq. (16) and (17)), are estimated as follows:

$$\begin{aligned}
\frac{\partial\phi}{\partial x} &= \left(\frac{\partial\phi}{\partial x}\right)_A \cdot [1 - f(\widehat{x})] + \left(\frac{\partial\phi}{\partial x}\right)_B \cdot f(\widehat{x}) \\
\frac{\partial\phi}{\partial y} &= \left(\frac{\partial\phi}{\partial y}\right)_A \cdot [1 - f(\widehat{x})] + \left(\frac{\partial\phi}{\partial y}\right)_B \cdot f(\widehat{x}), \\
f(\widehat{x}) &= -2\widehat{x}^3 + 3\widehat{x}^2 \\
\widehat{x} &= \frac{x - (L_A - L_{over})}{L_{over}}
\end{aligned} \qquad (23)$$

where L_A is length of domain A, L_{over} is the length of the overlapped domain. In Expressions (23), the subscripts A and B in the derivatives mean that the derivatives are calculated using the solution from domain A and B, respectively.

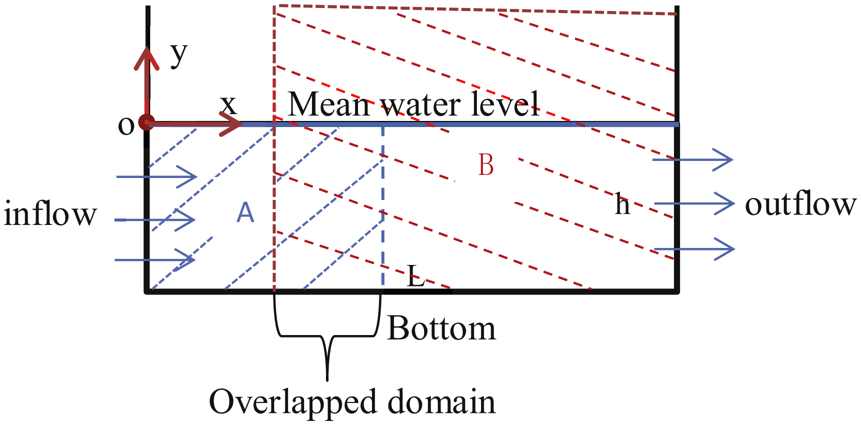


Fig. 20. Sketch of numerical wave tank using multi-block strategy with overlapped mesh.

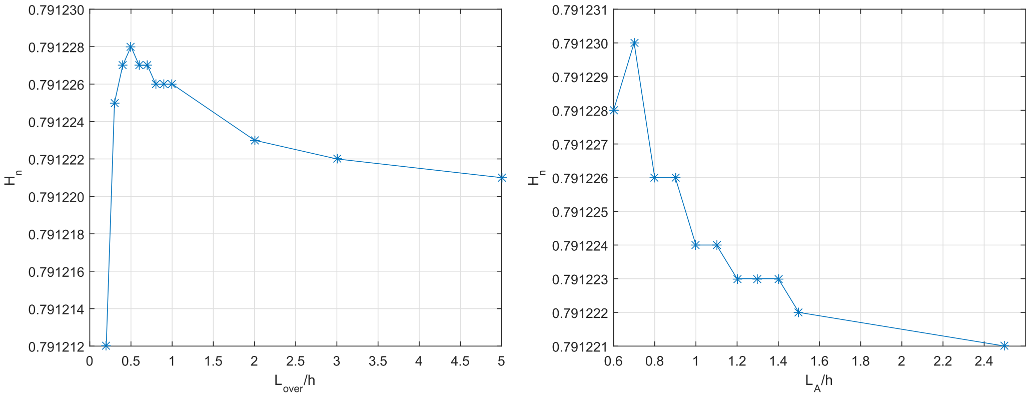


Fig. 21. Wave height for various length of domain. Left: study on the length of overlapped domain. Right: study on the length of domain A.

A preliminary study on the length of domain A and overlapped domain is carried out. It is found that the wave profile is not sensitive to the length of both domains. Fig. 21 gives the wave height for various domain size at $t = 2T$ with $dx = dy = h/10$. Study on the length of overlapped domain is carried out with $L_A - L_{over} = 0.1h$. $L_{over}/h = 0.5$ gives the closest wave height to the analytical value. The trend of the wave height plotted in the left figure of Fig. 21 is probably due to the fact that Treatment 2 leads to relatively higher wave height compared to boundary-fitted mesh approach and a 3rd order weighting function is used for the overlapped domain. Further $L_{over}/h = 0.5$ is adopted in the study on the influence of length of domain A. Since L_{over} is constant, the study on L_A is actually a study on $L_A - L_{over}$, i.e. length of the domain with only boundary-fitted mesh. Here $L_A/h = 0.6$ represents that there is only one grid in x direction for the domain with only boundary-fitted mesh. Since boundary-fitted mesh results in smaller wave height, it is expected that the wave height would decrease with the increase of L_A/h. As shown in the right figure, the wave height decreases with the increase of L_A except for $L_A = 0.6h$. That may be because the left Neumann boundary of domain B is too close to the inflow boundary. More specifically grid nodes on the left Neumann boundary, which have the most inaccurate results as mentioned in Section 3.3, are used expressing the left boundary condition of domain B.

Effectiveness of the overlapping multi-block strategy is verified by making the comparison of the wave elevation and CPU time with the results discussed in Section 5, i.e. from a full boundary-fitted mesh solution and from a full fixed mesh solution. The length of A and overlapped domain are $L_A = 0.7h$ and $L_{over} = h/2$ respectively. Except for the mesh strategy, all the other parameters for the numerical wave tank are the same as in Section 5. The free-surface elevation at $t = 2T$ obtained by multi-block strategy is provided in Fig. 22 together with the analytical solution and the numerical solutions from the full boundary-fitted mesh and the fixed mesh with Treatment 2. From the comparison, globally the multi-block strategy leads to the same accuracy as the two individual mesh strategies. From the enlarged view towards the domain A, multi-block strategy gives the same wave profile as boundary-fitted mesh strategy, in another word it improves the accuracy with respect to using a fixed mesh at Neumann boundaries. When studying the calculation speed, we set $L_A = 0.5h + 2dy$ to ensure the number of grids corresponding to L_A is an integer. It turns out that multi-block approach saves much time compared to the boundary-fitted mesh algorithm when the direct solver PARDISO is applied. An iterative matrix solver may be considered in future studies. The requirement for more CPU time when compared to fixed mesh algorithm is acceptable. The comparison of CPU time for the three cases is presented in Fig. 23. To sum up, the multi-block strategy is a

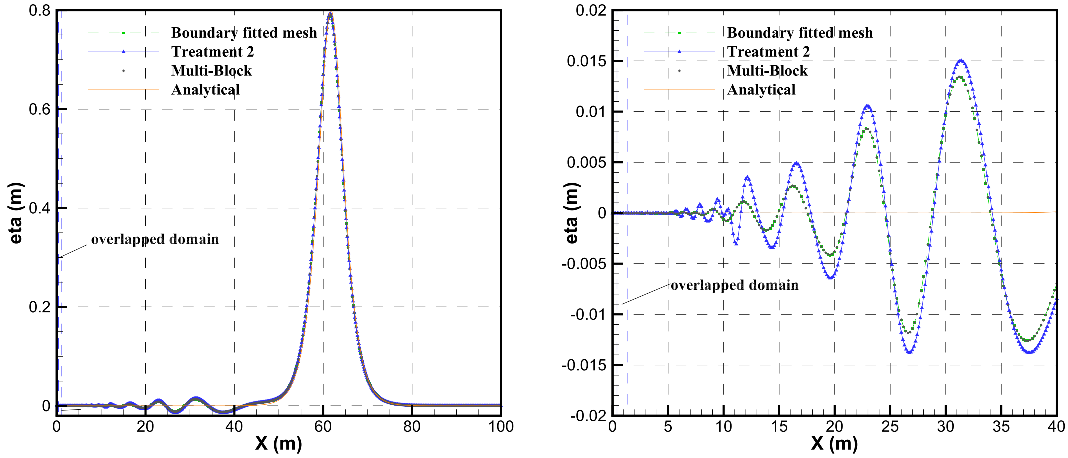


Fig. 22. Wave elevation at $t = 2T$. $H/h = 0.4$. Left: Full view, Right: Enlarged view.

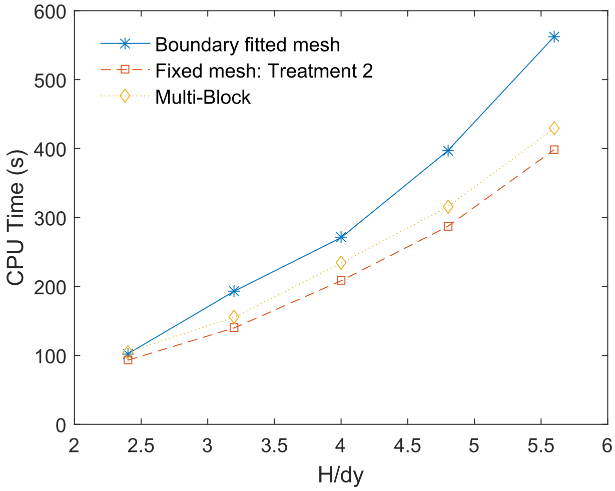


Fig. 23. Comparison of CPU time. $t = 2T$. $H/h = 0.4$.

good candidate for cases with Neumann boundaries like a physical wave-maker or a floating body especially when the water domain is large.

7. Conclusions and future work

Based on the original HPC method, a flux method for Neumann boundary conditions and two fixed-grid treatments near the free surface are proposed. The flux method deals more consistently with the Neumann boundary condition particularly at the sharp corner than the original HPC method. Four strategies of applying the flux algorithm are tested by studying a mixed Dirichlet—Neumann BVP. The comparison of accuracy suggests that it is beneficial to apply the flux method near sharp corners. By using the flux method, the HPC method is more robust. In order to improve the efficiency of the original HPC method, we studied two treatments of fixed mesh with free surface immersed. The fixed mesh approaches are first tested by studying a single time step of a propagating wave. They are further verified by considering a 2D fully nonlinear numerical wave tank. Comparison of the generated wave profile and corresponding CPU time indicates that the proposed Treatment 2 represents a good candidate for the use of a fixed boundary-immersed grid in free-surface problems. A numerical multi-block strategy is developed as a compromise between accuracy and efficiency. It is also verified by a 2D nonlinear wave making case.

The future study will focus on the further development of the multi-block algorithm. Modeling plunging waves by applying Lagrangian formulation of the free surface conditions should also be investigated. Since we limit ourselves in 2D cases for present, the improved HPC method will be extended to 3D.

Acknowledgment

One of the co-authors is connected with the Centre of Excellence NTNU AMOS, supported by the Research Council of Norway through the Centres of Excellence funding scheme AMOS, project number 223254.

References

Atkinson, K., Han, W., 2009. Theoretical Numerical Analysis: a Functional Analysis Framework. Springer Science and Business Media.

Boussinesq, J., 1871. Thorie de lintumescence liquide appele onde solitaire ou de translation, se propageant dans un canal rectangulaire. In: Comptes rendus hebdomadaires des sances de l'Acadmie des Sciences, p. 755.

Cai, X., 2003. Overlapping domain decomposition methods. In: Advanced Topics in Computational Partial Differential Equations. Springer, pp. 57—95.

Colicchio, G., Greco, M., Faltinsen, O., 2006. A bem-level set domain-decomposition strategy for non-linear and fragmented interfacial flows. Int. J. Numer. Methods Eng. 67, 1385—1419.

Colicchio, G., Greco, M., Lugni, C., Faltinsen, O.M., 2010. Towards a fully 3d domain-decomposition strategy for water-on-deck phenomena. J. Hydrodyn. Ser. B 22, 462—467.

Fenton, J., 1972. A ninth-order solution for the solitary wave. J. Fluid Mech. 53, 257—271.

Goring, D.G., 1978. Tsunamis—the Propagation of Long Waves onto a Shelf (Ph.D. thesis). California Institute of Technology.

Grimshaw, R., 1971. The solitary wave in water of variable depth. Part 2. J. Fluid Mech. 46, 611—622.

Hanssen, F.-C., Bardazzi, A., Lugni, C., Greco, M., Faltinsen, O., Free-surface tracking with the harmonic polynomial cell method, under review on Int. J. for Numerical Methods in Engineering, 2017

Intel, R., 2012. Intel Math Kernel Library Reference Manual. Technical Report, Tech. Rep. 630813—051US [Online]. Available: http://software.intel.com/sites/products/documentation/hpc/mkl/mklman/mklman.pdf, 2011.

Kim, C., Clement, A., Tanizawa, K., et al., 1999. Recent research and development of numerical wave tanks-a review. Int. J. Offshore Polar Eng. 9.

Liang, H., Faltinsen, O.M., Shao, Y.L., 2015. Application of a 2d harmonic polynomial cell (hpc) method to singular flows and lifting problems. Appl. Ocean Res. 53, 75—90.

Ma, S., Hanssen, F.-C., Siddiqui, A., Greco, M., Faltinsen, O. Local and global properties of the harmonic polynomial cell method: in-depth analysis in two dimensions,under review on Int. J. for Numerical Methods in Engineering, 2017.

Mo, W., 2010. Numerical Investigation of Solitary Wave Interaction with Group of Cylinders (Ph.D. thesis). Cornell University.

Ogilvie, T.F., 1967. Nonlinear high-froude-number free-surface problems. J. Eng. Math. 1, 215—235.

Quarteroni, A., Valli, A., 1999. Domain Decomposition Methods for Partial Differential Equations. Oxford University Press.

Shao, Y.-L., Faltinsen, O.M., 2012. Towards efficient fully-nonlinear potential-flow solvers in marine hydrodynamics. In: Proceedings of the 31st International Conference on Ocean, Offshore and Arctic Engineering(OMAE), Rio de Janeiro, Brazil, pp. 369—380.

Shao, Y.-L., Faltinsen, O.M., 2014a. A harmonic polynomial cell (hpc) method for 3d laplace equation with application in marine hydrodynamics. J. Comput. Phys. 274, 312—332.

Shao, Y.-L., Faltinsen, O.M., 2014b. Fully-nonlinear wave-current-body interaction analysis by a harmonic polynomial cell method. J. Offshore Mech. Arct. Eng. 136, 031301.

Zhou, B., Wu, G., Meng, Q., 2016. Interactions of fully nonlinear solitary wave with a freely floating vertical cylinder. Eng. Anal. Bound. Elem. 69, 119—131.

Please cite this article in press as: Zhu, W., et al., Improved HPC method for nonlinear wave tank, International Journal of Naval Architecture and Ocean Engineering (2017), http://dx.doi.org/10.1016/j.ijnaoe.2017.03.009