J. Math. Study
doi: 10.4208/jms.v52n3.19.06

Vol. **52**, No. 3, pp. 320-340
September 2019

A Domain Decomposition Method for Linearized Boussinesq-Type Equations

Joao Guilherme Caldas Steinstraesser[1,3], Gaspard Kemlin[2] and Antoine Rousseau[1,*]

[1] *Inria, IMAG, Univ Montpellier, CNRS, Montpellier, France.*
[2] *Inria Chile, Avenida Apoquindo 2827, Las Condes, Santiago, Chile.*
[3] *MERIC, Avenida Apoquindo 2827, Las Condes, Santiago, Chile.*

Received January 7, 2019; Accepted May 26, 2019

Abstract. In this paper, we derive discrete transparent boundary conditions for a class of linearized Boussinesq equations. These conditions happen to be non-local in time and we test numerically their accuracy with a Crank-Nicolson time-discretization on a staggered grid. We use the derived transparent boundary conditions as interface conditions in a domain decomposition method, where they become local in time. We analyze numerically their efficiency thanks to comparisons made with other interface conditions.

AMS subject classifications: 65M55

Key words: Boussinesq-type equations, finite differences scheme, transparent boundary conditions, domain decomposition, interface conditions, Schwarz alternating method.

1 Introduction

Among the main challenges faced in the mathematical framework of coastal engineering is the study of wave propagation in the nearshore area. One field of research in this topic makes use of the Boussinesq equations for water of varying depth that describe the nonlinear propagation of waves in shallow water. The work of Peregrine [1], Green and Naghdi [2] laid the basis for many Boussinesq-type equations used nowadays. The dispersion properties of these equations have then been improved by Nwogu [3] for practical numerical simulation of ocean wave processes from deep to shallow water. In this paper, we work on the equations derived by Nwogu [3] and that can be recalled as follows. Consider a three-dimensional wave field with surface elevation $\eta(x,y,t)$ over a non-constant water depth $h(x,y)$ and with speed $\boldsymbol{u}(x,y,z,t) = (u,v)$, respectively the

*Corresponding author. *Email addresses:* joao-guilherme.caldas-steinstraesser@inria.fr (J.G. Caldas Steinstraesser), gaspard.kemlin@inria.fr (G. Kemlin), antoine.rousseau@inria.fr (A. Rousseau)

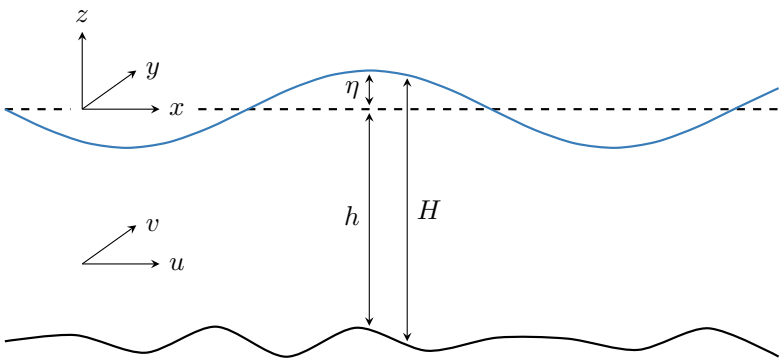

Figure 1: Definition of the quantities η, h, H, $\boldsymbol{u}$.

speeds along the x and y axis, defined at a reference depth $z=z(x,y)$. Figure 1 provides a sketch where we represented these quantities. With $\boldsymbol{\tau}$ being the bottom shear stress, Nwogu [3] obtained (1.1), which consists of a continuity equation and a momentum equation. These equations have been used in a `C/Matlab` program known as the *Boussinesq Ocean and Surf Zone* (BOSZ) model, developed by Roeber and Cheung [4].

$$\begin{cases} \eta_t+\nabla\cdot[(h+\eta)\boldsymbol{u}]+\nabla\cdot\left[\left(\dfrac{z^2}{2}-\dfrac{h^2}{6}\right)h\nabla(\nabla\cdot\boldsymbol{u})+\left(z+\dfrac{h}{2}\right)h\nabla(\nabla\cdot(h\boldsymbol{u}))\right]=0, \\ \boldsymbol{u}_t+(\boldsymbol{u}\cdot\nabla)\boldsymbol{u}+g\nabla\eta+z\left[\dfrac{z}{2}\nabla(\nabla\cdot\boldsymbol{u}_t)+\nabla(\nabla\cdot(h\boldsymbol{u}_t))\right]+\boldsymbol{\tau}=0. \end{cases} \tag{1.1}$$

To simplify the framework, we consider in this paper the 1D equations and ignore the bottom shear stress $\boldsymbol{\tau}$. We also consider a constant flat bottom $h=h_0$. Thus, the total height H can be recovered by the relation $H=h_0+\eta$. Then, we perform a linearization around the equilibrium point $(\bar{\eta},\bar{u})=(0,\bar{u})$, with $\bar{u}\in\mathbb{R}$, to get

$$\begin{cases} \eta_t+\bar{u}\eta_x+h_0u_x+\tilde{h}u_{xxx}=0, \\ u_t+g\eta_x+\bar{u}u_x+\bar{h}u_{xxt}=0, \end{cases} \tag{1.2}$$

to which we now refer as the *linearized Boussinesq equations*. Here, $\tilde{h}$ and $\bar{h}$ are constants defined by

$$\tilde{h}=\left(\frac{z^2}{2}+h_0z+\frac{h_0^2}{3}\right)h_0, \qquad \bar{h}=z\left(\frac{z}{2}+h_0\right). \tag{1.3}$$

Using the value $z=-0.53753\times h_0$ (see [3]) and $h_0=1$, we have that $\bar{h}$ and $\tilde{h}$ are both negative. Setting $\bar{u}=0$, $\tilde{h}=0$, $\bar{h}=-\varepsilon$ a small parameter, $g=1$ and $h_0=1$ leads to the formulation of the linearized Green-Naghdi equations, for which discrete transparent boundary conditions have been derived by Kazakova and Noble [5]. In this paper, we focus on the case where $\bar{u}=0$ for the sake of simplicity, but the conclusions of Section 3

remain the same for $\bar{u} \neq 0$. The equations we are going to focus on in this paper are

$$\begin{cases} \eta_t + h_0 u_x + \tilde{h} u_{xxx} = 0, & \text{(1.4a)} \\ u_t + g\eta_x + \bar{h} u_{xxt} = 0. & \text{(1.4b)} \end{cases}$$

The first objective of this paper is to derive transparent boundary conditions for equations (1.4a)–(1.4b). Indeed, this system is set on the whole space $\mathbb{R}$ and thus we need to restrict the area of computation to a bounded domain for practical applications. This requires to find suitable boundary conditions. We focus on transparent boundary conditions in order to let waves escaping the domain without any reflection (a phenomenon that we observe for instance with Dirichlet conditions). From a mathematical point of view, we set the problem as follows: given a compactly supported initial data, one derives suitable conditions at the boundaries so that the solution on the bounded domain coincides with the restriction to this domain of the solution computed on the whole domain. In practice, this study can be done in either a continuous or discretized framework. A review of these techniques can be found in [6] where the authors build such conditions for the Schrödinger equation. In the linear case, the study in the continuous framework is carried out by applying the Laplace transform in time and adapt boundary conditions to keep solutions bounded. The Laplace-inverse transform of these conditions results into non-local in time operators. The adaptation of this technique to the discretized framework uses the $\mathcal{Z}$-transform, which is the discrete equivalent of the Laplace transform. Again, the numerical inversion of the obtained discrete conditions yields non-local operators.

The second and main objective of this paper is to test the efficiency of the discrete transparent conditions as interface conditions in a domain decomposition method with the alternating Schwarz method. The interest of this method lies in the possibilities, with few modifications in the original code, to split the original computational domain and/or couple different models (for instance farshore and nearshore). Our work in this paper can be seen as a step towards more efficient coupling between different numerical models in coastal engineering. The main difficulty is then to find suitable interface conditions to exchange information between the different subdomains so that the convergence is achieved as fast as possible.

The paper is organized as follows. In Section 2, we apply the study from [5,7,8] to the linearized Boussinesq equations and obtain discrete transparent boundary conditions. Then, we provide some numerical tests to evaluate the accuracy of such conditions before implementing them in a domain decomposition method. In Section 3, we briefly recall the idea of the additive Schwarz method that we are going to use before going on with the adaptation of transparent boundary conditions as interface conditions. We conclude with some numerical experiments to evaluate their efficiency.

2 Derivation of transparent boundary conditions

First, we introduce the initial value problem that we seek to solve using transparent boundary conditions (TBC):

$$\begin{cases} \eta_t + h_0 u_x + \tilde{h} u_{xxx} = 0, & \forall\, x \in \mathbb{R}, \quad t > 0, \\ u_t + g\eta_x + \bar{h} u_{xxt} = 0, & \forall\, x \in \mathbb{R}, \quad t > 0, \\ u(x,0) = u_0(x), \quad \eta(x,0) = \eta_0(x), & \forall\, x \in \mathbb{R}, \\ u(x,t) \xrightarrow[x\to\pm\infty]{} 0, \qquad \eta(x,t) \xrightarrow[x\to\pm\infty]{} 0, & \forall\, t > 0. \end{cases} \tag{2.1}$$

The goal of TBC is to find boundary conditions for a finite domain (for instance $[0,L]$) such that the solution on this domain coincides with the restriction to this domain of the solution of problem (2.1). Hence, the problem we will work on is

$$\begin{cases} \eta_t + h_0 u_x + \tilde{h} u_{xxx} = 0, & \forall\, x \in [0,L], \quad t > 0, \\ u_t + g\eta_x + \bar{h} u_{xxt} = 0, & \forall\, x \in [0,L], \quad t > 0, \\ u(x,0) = u_0(x), \quad \eta(x,0) = \eta_0(x), & \forall\, x \in [0,L], \\ \qquad + \text{ transparent boundary conditions at } x = 0 \text{ and } x = L. \end{cases} \tag{2.2}$$

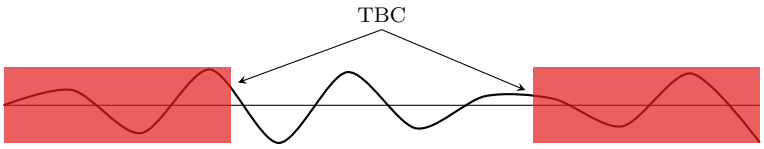

Figure 2: *Transparent* boundary conditions means that the solution we want to compute on $[0,L]$ is a *picture* of the solution on $\mathbb{R}$ restricted to $[0,L]$.

Note that it is possible to decouple equations (1.4a) – (1.4b) to obtain an equation on u only. Taking the cross derivatives of (1.4a) and (1.4b), we get

$$u_{tt} + \bar{h} u_{xxtt} - g h_0 u_{xx} - g\tilde{h} u_{xxxx} = 0. \tag{2.3}$$

Thus, system (1.4a)–(2.3) is equivalent to (1.4a)–(1.4b). We can also note a fourth order space derivative. Therefore, we will need to derive four conditions.

In this section, we follow the steps proposed in [5, 7, 8] to derive discrete TBC that are adapted to the discretized problem. We end this section with some numerical tests to analyze the efficiency of the obtained conditions.

2.1 Deriving discrete transparent boundary conditions

We introduce here the derivation of *discrete* TBC (DTBC) for the discretized problem. These are boundary conditions directly derived from the discretized problem. We adapt here the method used in [7] for the Schrdinger equation, in [8] for the KdV equation and in [5] for the linearized Green-Naghdi equations. It follows four main steps:

1. Discretize the equations (1.4a) and (1.4b).
2. Use the $\mathcal{Z}$-transform on the problem on the complementary set.
3. Find the conditions at the two boundaries in the $\mathcal{Z}$-space.
4. Use the inverse $\mathcal{Z}$-transform to find the transparent boundary conditions.

Discretization

The first step is to find a discretization in time on a staggered grid. Let δx be the spatial-step and δt be the time-step. We build this grid such that $J=L/\delta x$ and $x_j=j\delta x$:

$$0=x_0<x_1<x_2<\cdots<x_{J-1}<x_J=L,$$

$j<0$ and $j>J$ denote nodes that are out of the domain we want to work on. The time t^n then stands for $n\delta t$. Using a Crank-Nicolson scheme, (1.4a) yields

$$\begin{aligned}&\frac{\eta_{j+\frac{1}{2}}^{n+1}-\eta_{j+\frac{1}{2}}^{n}}{\delta t}+\frac{h_0}{2}\left(\frac{u_{j+1}^{n+1}-u_j^{n+1}}{\delta x}+\frac{u_{j+1}^{n}-u_j^{n}}{\delta x}\right)\\&\quad+\frac{\tilde{h}}{2}\left(\frac{u_{j+2}^{n+1}-3u_{j+1}^{n+1}+3u_j^{n+1}-u_{j-1}^{n+1}}{\delta x^3}+\frac{u_{j+2}^{n}-3u_{j+1}^{n}+3u_j^{n}-u_{j-1}^{n}}{\delta x^3}\right)=0.\end{aligned}\tag{2.4}$$

Note that the finite differences operator used for the third spatial derivative is centered around $j+\frac{1}{2}$.

Discretizing (1.4b) is straightforward:

$$\begin{aligned}&\frac{u_j^{n+1}-u_j^{n}}{\delta t}+\frac{g}{2}\left(\frac{\eta_{j+\frac{1}{2}}^{n+1}-\eta_{j-\frac{1}{2}}^{n+1}}{\delta x}+\frac{\eta_{j+\frac{1}{2}}^{n}-\eta_{j-\frac{1}{2}}^{n}}{\delta x}\right)\\&\quad+\frac{\bar{h}}{\delta t}\left(\frac{u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}}{\delta x^2}-\frac{u_{j+1}^{n}-2u_j^{n}+u_{j-1}^{n}}{\delta x^2}\right)=0.\end{aligned}\tag{2.5}$$

Problem on the complementary set

To solve the initial value problem (2.2), we assume that the initial conditions u_0 and η_0 are compactly supported in $[0,L]$. The derivation of the DTBC associated to our problem can be done by studying the problem on the complementary set of $[0,L]$:

$$\begin{cases}\eta_t+h_0u_x+\tilde{h}u_{xxx}=0, & \forall\ x\in\mathbb{R}\backslash[0,L],\quad t>0,\\ u_t+g\eta_x+\bar{h}u_{xxt}=0, & \forall\ x\in\mathbb{R}\backslash[0,L],\quad t>0,\\ u(x,0)=0,\quad \eta(x,0)=0, & \forall\ x\in\mathbb{R}\backslash[0,L],\\ u(x,t)\xrightarrow[x\to\pm\infty]{}0,\quad \eta(x,t)\xrightarrow[x\to\pm\infty]{}0, & \forall\ t>0.\end{cases}\tag{2.6}$$

$\mathcal{Z}$-transform

The second step is to compute the $\mathcal{Z}$-transform of (2.6). The $\mathcal{Z}$-transform of the sequence $(u_n)_{n\geqslant 0}$ is defined as a function of the complex variable z:

$$\forall\ |z|>R>0, \qquad \hat{u}(z)=\mathcal{Z}\{(u_n)\}:=\sum_{n\geqslant 0}u_n z^{-n}, \tag{2.7}$$

where R is the convergence radius of the series. One important property of the $\mathcal{Z}$-transform is that

$$\mathcal{Z}\{(u_{n+1})\}=z\hat{u}(z)-zu_0. \tag{2.8}$$

In our case, as $\forall\ j\leqslant 0,\ j\geqslant J$, $u_j^0=0$ and $\eta_j^0=0$ (by compactness of the initial conditions in $[0,L]$), we have

$$\forall\ j\leqslant 0,\ j\geqslant J, \qquad \mathcal{Z}\left\{\left(u_j^{n+1}\right)\right\}=z\hat{u}_j(z) \ \text{ and } \ \mathcal{Z}\left\{\left(\eta_{j+\frac{1}{2}}^{n+1}\right)\right\}=z\hat{\eta}_{j+\frac{1}{2}}(z). \tag{2.9}$$

From now on, $\hat{u}_j$ and $\hat{\eta}_{j+\frac{1}{2}}$ will stand for $\hat{u}_j(z)$ and $\hat{\eta}_{j+\frac{1}{2}}(z)$. Using (2.9) on (2.4) and (2.5), we get

$$\begin{cases} \hat{\eta}_{j+\frac{1}{2}}=-\dfrac{1}{s(z)}\left(\dfrac{h_0}{\delta x}\left(\hat{u}_{j+1}-\hat{u}_j\right)+\dfrac{\tilde{h}}{\delta x^3}\left(\hat{u}_{j+2}-3\hat{u}_{j+1}+3\hat{u}_j-\hat{u}_{j-1}\right)\right), & \text{(2.10a)} \\ \dfrac{g}{\delta x}\left(\hat{\eta}_{j+\frac{1}{2}}-\hat{\eta}_{j-\frac{1}{2}}\right)+\dfrac{\bar{h}}{\delta x^2}s(z)\hat{u}_{j+1}+s(z)\left(1-2\dfrac{\bar{h}}{\delta x^2}\right)\hat{u}_j+\dfrac{\bar{h}}{\delta x^2}s(z)\hat{u}_{j-1}=0, & \text{(2.10b)} \end{cases}$$

where

$$s(z)=\frac{2}{\delta t}\frac{z-1}{z+1}. \tag{2.11}$$

Note that $s(z)$ has a singularity at $z=-1$. For this reason, we will work only on $\mathbb{C}_1:=\{z\in\mathbb{C};\ |z|>1\}$. On this set, $\Re(s(z))>0$. Injecting (2.10a) in (2.10b) gives, in a similar way than the derivation of (2.3), the following recurrence (in space) relation for $\hat{u}$:

$$\begin{aligned} -\frac{g\tilde{h}}{\delta x^4}\hat{u}_{j+2}+\left(\frac{\tilde{h}}{\delta x^2}s(z)^2-\frac{gh_0}{\delta x^2}+4\frac{g\tilde{h}}{\delta x^4}\right)\hat{u}_{j+1}+\left(s(z)^2\left(1-2\frac{\bar{h}}{\delta x^2}\right)+2\frac{gh_0}{\delta x^2}-6\frac{g\tilde{h}}{\delta x^4}\right)\hat{u}_j \\ +\left(\frac{\tilde{h}}{\delta x^2}s(z)^2-\frac{gh_0}{\delta x^2}+4\frac{g\tilde{h}}{\delta x^4}\right)\hat{u}_{j-1}-\frac{g\tilde{h}}{\delta x^4}\hat{u}_{j-2}=0. \end{aligned} \tag{2.12}$$

We look for a solution of the form

$$\hat{u}_j=\sum_{i=1}^{4}\lambda_i r_i(z)^j, \tag{2.13}$$

where the r_i are the roots of the corresponding characteristic polynomial, which reads

$$P(r):=r^4+\left(\alpha-\beta s(z)^2-4\right)r^3+\left(s(z)^2(2\beta-\gamma)-2\alpha+6\right)r^2+\left(\alpha-\beta s(z)^2-4\right)r+1, \tag{2.14}$$

where we used

$$\alpha=\frac{h_0\delta x^2}{\tilde{h}}<0,\qquad \beta=\frac{\bar{h}\delta x^2}{g\tilde{h}}>0,\qquad \gamma=\frac{\delta x^4}{g\tilde{h}}<0. \tag{2.15}$$

Note that α is adimensional whereas β and γ are homogeneous to s^2, resulting into the coefficients of P being adimensional.

Solving the recurrence relation

Now, we are interested in the study of the polynomial P, and especially its roots as the explicit formulation of $\hat{u}_j$ will rely on them.

Proposition 2.1. For all $z\in\mathbb{C}_1$, the roots of the polynomial P given in (2.14) verify (by ordering them with $|r_i|\leqslant|r_{i+1}|$, $i=1,2,3$)

$$|r_4(z)|\geqslant|r_3(z)|>1>|r_2(z)|\geqslant|r_1(z)|.$$

Proof. The proof of this proposition is inspired from [8]. First, let us specify that, as the constant coefficient is equal to 1, we know that

$$|r_1(z)r_2(z)r_3(z)r_4(z)|=1.$$

Moreover, we can see in (2.14) that the coefficients of P are symmetric, in the sense that, as 0 is not a root, $P(r)=0\Leftrightarrow P(1/r)=0$. From these two facts, we can say that

$$|r_4(z)|\geqslant|r_3(z)|\geqslant1\geqslant|r_2(z)|\geqslant|r_1(z)|,$$

where $r_2=1/r_3$ and $r_1=1/r_4$. Next, we show that there is no root of modulus 1. To this end, assume there exists a θ such that $e^{i\theta}$ is a root of P. Then,

$$\begin{aligned}e^{4i\theta}+\left(\alpha-\beta s(z)^2-4\right)e^{3i\theta}+\left(s(z)^2(2\beta-\gamma)-2\alpha+6\right)e^{2i\theta}\\+\left(\alpha-\beta s(z)^2-4\right)e^{i\theta}+1=0.\end{aligned} \tag{2.16}$$

Multiplying by $e^{-2i\theta}$ gives, after using Euler formulas,

$$2\cos2\theta+2\cos\theta\left(\alpha-\beta s(z)^2-4\right)+s(z)^2(2\beta-\gamma)-2\alpha+6=0. \tag{2.17}$$

Finally, we find

$$\begin{aligned}s(z)^2&=\frac{2\alpha(1-\cos\theta)+8\cos\theta-4\cos^2\theta-4}{2\beta(1-\cos\theta)-\gamma}\\&=\frac{2\alpha(1-\cos\theta)-4(\cos\theta-1)^2}{2\beta(1-\cos\theta)-\gamma}\leqslant0,\end{aligned} \tag{2.18}$$

so that $s(z)\in i\mathbb{R}$. Hence, as $z\in\mathbb{C}_1\Rightarrow\Re(s(z))>0$, we reached a contradiction and P has no root of modulus 1, which concludes the proof. □

Now that we know the roots of the polynomial P, we can explicit $\hat{u}$ in both the right and the left regions:

$$\begin{cases} \hat{u}_j = \sum_{i=1}^{4} \lambda_i^l r_i(z)^j, & \forall\, j \leqslant 2, \\ \hat{u}_j = \sum_{i=1}^{4} \lambda_i^r r_i(z)^j, & \forall\, j \geqslant J-2. \end{cases} \tag{2.19}$$

As we are aiming at a bounded solution, we can eliminate two components in each region (and drop the r and l):

$$\begin{cases} \hat{u}_j = \lambda_3 r_3(z)^j + \lambda_4 r_4(z)^j, & \forall\, j \leqslant 2, \\ \hat{u}_j = \lambda_1 r_1(z)^j + \lambda_2 r_2(z)^j, & \forall\, j \geqslant J-2. \end{cases} \tag{2.20}$$

Finding the discrete transparent boundary conditions

Let us recall that we need two relations at each interface, as mentioned at the beginning of this section. It can be easily verified by a straightforward computation using (2.20) that, on the left, $\hat{u}_j$ satisfies the following two conditions (we dropped the z for the sake of clarity):

$$\begin{cases} \dfrac{1}{r_3 r_4}\hat{u}_j - \dfrac{r_3+r_4}{r_3 r_4}\hat{u}_{j-1} + \hat{u}_{j-2} = 0, \\ \dfrac{1}{(r_3 r_4)^2}\hat{u}_{j+2} - 2\dfrac{r_3+r_4}{(r_3 r_4)^2}\hat{u}_{j+1} + \dfrac{(r_3+r_4)^2}{(r_3 r_4)^2}\hat{u}_j - \hat{u}_{j-2} = 0. \end{cases} \quad \forall\, j \leqslant 2, \tag{2.21}$$

For the right interface, the conditions are

$$\begin{cases} \hat{u}_{j+2} - (r_1+r_2)\hat{u}_{j+1} + r_1 r_2 \hat{u}_j = 0, \\ \hat{u}_{j+2} - 2(r_1+r_2)\hat{u}_{j+1} + (r_1+r_2)^2\hat{u}_j - (r_1 r_2)^2 \hat{u}_{j-2} = 0. \end{cases} \quad \forall\, j \geqslant J-2, \tag{2.22}$$

Before applying the inverse $\mathcal{Z}$-transform , we define the nine following kernels:

$$k_1 := r_1 + r_2, \quad k_2 := (r_1+r_2)^2, \quad k_3 := (r_1 r_2), \quad k_4 := (r_1 r_2)^2, \tag{2.23a}$$

$$k_5 := \frac{r_3+r_4}{r_3 r_4}, \quad k_6 := \frac{(r_3+r_4)^2}{(r_3 r_4)^2}, \quad k_7 := \frac{1}{r_3 r_4}, \tag{2.23b}$$

$$k_8 := \frac{1}{(r_3 r_4)^2}, \quad k_9 := \frac{r_3+r_4}{(r_3 r_4)^2}. \tag{2.23c}$$

Hence, Eqs. (2.21) and (2.22) become

$$\begin{cases} \hat{u}_{j-2} - k_5\hat{u}_{j-1} + k_7\hat{u}_j = 0, \\ \hat{u}_{j-2} - k_6\hat{u}_j + 2k_9\hat{u}_{j+1} - k_8\hat{u}_{j+2} = 0, \end{cases} \quad \forall\, j \leqslant 2, \tag{2.24}$$

$$\begin{cases} \hat{u}_{j+2} - k_1\hat{u}_{j+1} + k_3\hat{u}_j = 0, \\ \hat{u}_{j+2} - 2k_1\hat{u}_{j+1} + k_2\hat{u}_j - k_4\hat{u}_{j-2} = 0. \end{cases} \quad \forall\, j \geqslant J-2, \tag{2.25}$$

Finally, by applying the inverse $\mathcal{Z}$-transform at $j=2$ and $j=J-2$, we find two conditions at each interface, with $\{u^n\}=\left(u^0,\dots,u^n\right)$:

$$\textbf{Left} \quad \begin{cases} \Gamma_1^l(\{u^n\}) := u_0^n - (Y_5 * u_1)^n + (Y_7 * u_2)^n = 0, \\ \Gamma_2^l(\{u^n\}) := u_0^n - (Y_6 * u_2)^n + 2(Y_9 * u_3)^n - (Y_8 * u_4)^n = 0, \end{cases} \tag{2.26}$$

$$\textbf{Right} \quad \begin{cases} \Gamma_1^r(\{u^n\}) := u_J^n - (Y_1 * u_{J-1})^n + (Y_3 * u_{J-2})^n = 0, \\ \Gamma_2^r(\{u^n\}) := u_J^n - 2(Y_1 * u_{J-1})^n + (Y_2 * u_{J-2})^n - (Y_4 * u_{J-4})^n = 0, \end{cases} \tag{2.27}$$

where $Y_i^n := \mathcal{Z}^{-1}\{k_i(z)\}$ for $i=1,\dots,9$ and $*$ denotes the discrete convolution:

$$\forall\, i \in [\![1,9]\!], \quad (Y_i * u_j)^n := \sum_{m=0}^{n} Y_i^m u_j^{n-m}. \tag{2.28}$$

It is possible to compute the convolution coefficients Y_i with an inverse Fast Fourier transform, as described in [7,8]. Indeed, the explicit formula for the inversion of $\hat{u}(z)$ is

$$u_n = \frac{1}{2\mathrm{i}\pi} \oint_{C_r} \hat{u}(z) z^{n-1} \mathrm{d}z, \quad r > R, \tag{2.29}$$

where C_r denotes any circle of radius $r > R$. Applying this formula to the kernels k_i and discretizing gives

$$\forall\, i \in [\![1,9]\!],\ n \in [\![0,N-1]\!], \quad Y_i^n = \frac{r^n}{N} \sum_{k=0}^{N-1} k_i\left(r\omega_N^k\right)\omega_N^{nk} = r^n \mathcal{F}^{-1}\left(k_i\left(r\omega_N^k\right)\right), \tag{2.30}$$

where $\mathcal{F}^{-1}$ is the inverse discrete Fourier transform, N is the number of nodes to discretize the circle and $\omega_N := e^{\mathrm{i}\frac{2\pi}{N}}$. This results into non-local in time operators $\Gamma_{1,2}^{l,r}$ but only at some points near the boundaries and [7, 8] give techniques to transform the convolution into recurrence relations that can be updated at each time-step. The goal of this paper being to study the application of transparent boundary conditions to domain decomposition methods, we will use the convolution coefficients as they are.

Remark 2.1. Some Y_i are represented in Figure 3. Instabilities can be observed on the Y_i for large n (they are not represented here). However, we will see when applying the DTBC to domain decomposition techniques in Section 3 that only the first coefficients Y_i^0 are needed. The interested reader can refer to [9], where a better sampling is introduced to avoid these instabilities for large n.

Final numerical scheme

We now have all the tools in our hands to derive a global numerical scheme. To emphasize the fact that the operators at the boundaries $\Gamma_{1,2}^{l,r}$ act on u, we decide to solve numerically the decoupled equation (2.3), that we recall here:

$$u_{tt} + \bar{h} u_{xxtt} - g h_0 u_{xx} - g\tilde{h} u_{xxxx} = 0.$$

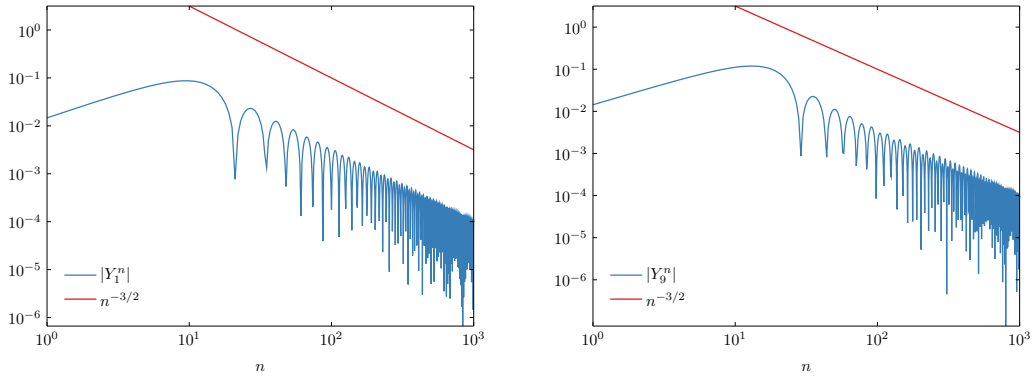

Figure 3: Coefficients Y_1 and Y_9 used in the convolution of the DTBC (2.26) and (2.27), with parameters from (2.41). We can observe a decreasing rate of $O\left(n^{-3/2}\right)$. The behavior of the other coefficients is similar.

Using a Crank-Nicolson scheme, we get the scheme (2.39). Note that we get a 4^{th} order differential equation, which resulted into a 5-points stencil. As it is also a two time-steps scheme, we need to specify how to compute u_j^1 knowing only the initial conditions u_0 and η_0. This can be done with a Taylor extension in the vicinity of $t=0$:

$$\left(u+\bar{h}u_{xx}\right)\big|_{t=\delta t}=\left(u+\bar{h}u_{xx}\right)\big|_{t=0}+\delta t\left(u+\bar{h}u_{xx}\right)_t\big|_{t=0}+\frac{\delta t^2}{2}\left(u+\bar{h}u_{xx}\right)_{tt}\big|_{t=0}+o(\delta t^2). \quad (2.31)$$

As $u(\cdot,0)\equiv 0$, the first term is 0. Then, using equation (1.4b), we find that

$$\left(u+\bar{h}u_{xx}\right)_t\big|_{t=0}=-g\eta_x|_{t=0}. \quad (2.32)$$

Combining (1.4a) and (1.4b), we also have

$$\left(u+\bar{h}u_{xx}\right)_{tt}\big|_{t=0}=-\left(g\eta_{xt}\right)|_{t=0}=\left(gh_0u_{xx}+g\tilde{h}u_{xxxx}\right)\big|_{t=0}=0. \quad (2.33)$$

Finally, we get the approximation

$$\left(u+\bar{h}u_{xx}\right)\big|_{t=\delta t}=-g\eta_x|_{t=0}+o(\delta t^2), \quad (2.34)$$

that we can discretize to compute u^1. One may thus wonder what conditions to use so that we also have TBC at the first time-step? The idea is to apply again the same method and solve the problem on the complementary set (as $\eta_x(\cdot,0)$ is compactly supported in $[0,L]$):

$$\begin{cases} u+\bar{h}u_{xx}=0, & \forall\, x\in\mathbb{R}\backslash[0,L], \\ u(x)\xrightarrow[x\to\pm\infty]{}0. \end{cases} \quad (2.35)$$

Given the conditions at infinity, we have

$$\begin{cases} u(x)=c_1e^{\frac{1}{\sqrt{-\bar{h}}}x}, & \forall\, x<0, \\ u(x)=c_2e^{-\frac{1}{\sqrt{-\bar{h}}}x}, & \forall\, x>L, \end{cases} \quad (2.36)$$

which gives the following boundary conditions:

$$u'(0)=\frac{1}{\sqrt{-\bar{h}}}u(0),\qquad u'(L)=-\frac{1}{\sqrt{-\bar{h}}}u(L). \tag{2.37}$$

Knowing u^0 and η^0, the final numerical scheme is:

- $\boxed{n=0}$

$$\begin{cases} u_0^1-\sqrt{-\bar{h}}\left(\dfrac{-u_0^1+u_1^1}{\delta x}\right)=0, \\ u_j^1+\bar{h}\left(\dfrac{u_{j+1}^1-2u_j^1+u_{j-1}^1}{\delta x^2}\right)=-g\delta t\dfrac{\eta_{j+\frac{1}{2}}^0-\eta_{j-\frac{1}{2}}^0}{\delta x},\quad \forall\, 1\leqslant j\leqslant J-1, \\ u_J^1+\sqrt{-\bar{h}}\left(\dfrac{-u_{J-1}^1+u_J^1}{\delta x}\right)=0. \end{cases} \tag{2.38}$$

- $\boxed{n\geqslant 1}$

$$\begin{cases} \Gamma_1^l\left(\{u^{n+1}\}\right)=0,\qquad \Gamma_2^l\left(\{u^{n+1}\}\right)=0, \\ \dfrac{u_j^{n+1}-2u_j^n+u_j^{n-1}}{\delta t^2}+\dfrac{\bar{h}}{\delta t^2}\left(\dfrac{u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}}{\delta x^2}-2\dfrac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{\delta x^2}+\dfrac{u_{j+1}^{n-1}-2u_j^{n-1}+u_{j-1}^{n-1}}{\delta x^2}\right) \\ \quad -\dfrac{gh_0}{4}\left(\dfrac{u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}}{\delta x^2}+2\dfrac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{\delta x^2}+\dfrac{u_{j+1}^{n-1}-2u_j^{n-1}+u_{j-1}^{n-1}}{\delta x^2}\right) \\ \quad -\dfrac{g\tilde{h}}{4}\left(\dfrac{u_{j+2}^{n+1}-4u_{j+1}^{n+1}+6u_j^{n+1}-4u_{j-1}^{n+1}+u_{j-2}^{n+1}}{\delta x^4}+2\dfrac{u_{j+2}^n-4u_{j+1}^n+6u_j^n-4u_{j-1}^n+u_{j-2}^n}{\delta x^4}\right. \\ \quad \left.+\dfrac{u_{j+2}^{n-1}-4u_{j+1}^{n-1}+6u_j^{n-1}-4u_{j-1}^{n-1}+u_{j-2}^{n-1}}{\delta x^4}\right)=0,\quad \forall\, 2\leqslant j\leqslant J-2, \\ \Gamma_1^r\left(\{u^{n+1}\}\right)=0,\qquad \Gamma_2^r\left(\{u^{n+1}\}\right)=0. \end{cases} \tag{2.39}$$

Remark 2.2. Notice that (2.39) is equivalent to (2.4)-(2.5). Indeed, by writing (2.4) for two consecutive time steps and two consecutive spatial positions, and (2.5) for two consecutive time steps, we are able to obtain the equation (2.39) only on u. This procedure is the discrete equivalent of taking the cross derivatives of (1.4a) and (1.4b) to obtain (2.3).

Remark 2.3. The numerical scheme (2.39) can be used to find u only, as all our study relied only on equations where we had eliminated the unknown η (except to perform the first time-step). Note that at each iteration in time, we can recover η using (2.4). It is also possible to solve the system (2.4)–(2.5) directly with the same transparent boundary conditions (2.26) – (2.27) and find u and η simultaneously. The results are the same and the main advantage is that, as it is a first order system in time, we don't need to perform the first step independently. Using $\boldsymbol{X}^n=(\eta^n,u^n)\in\mathbb{R}^{2N-1}$, where N is the number of nodes in the discretization, the numerical system reads

$$\begin{bmatrix} \frac{1}{\delta t}\mathrm{Id} & A \\ B & C \end{bmatrix} \boldsymbol{X}^{n+1} = \begin{bmatrix} \frac{1}{\delta t}\mathrm{Id} & -A \\ -B & \mathrm{Id}_0 C \end{bmatrix} \boldsymbol{X}^{n} + \begin{pmatrix} 0_{\mathbb{R}^{N-1}} \\ \boldsymbol{b}^n \end{pmatrix}, \tag{2.40}$$

with

$$A := \begin{bmatrix} -\frac{\tilde{h}}{2\delta x^3}-\frac{h_0}{2\delta x} & \frac{3\tilde{h}}{2\delta x^3}+\frac{h_0}{2\delta x} & -\frac{3\tilde{h}}{2\delta x^3} & \frac{\tilde{h}}{2\delta x^3} & & \\ -\frac{\tilde{h}}{2\delta x^3} & \frac{3\tilde{h}}{2\delta x^3}-\frac{h_0}{2\delta x} & -\frac{3\tilde{h}}{2\delta x^3}+\frac{h_0}{2\delta x} & \frac{\tilde{h}}{2\delta x^3} & & (0) \\ & \ddots & \ddots & \ddots & \ddots & \\ (0) & & -\frac{\tilde{h}}{2\delta x^3} & \frac{3\tilde{h}}{2\delta x^3}-\frac{h_0}{2\delta x} & -\frac{3\tilde{h}}{2\delta x^3}+\frac{h_0}{2\delta x} & \frac{\tilde{h}}{2\delta x^3} \\ & & -\frac{\tilde{h}}{2\delta x^3} & \frac{3\tilde{h}}{2\delta x^3} & -\frac{3\tilde{h}}{2\delta x^3}-\frac{h_0}{2\delta x} & \frac{\tilde{h}}{2\delta x^3}+\frac{h_0}{2\delta x} \end{bmatrix} \in \mathbb{R}^{(N-1)\times N},$$

$$B := \begin{bmatrix} 0 & & & & & \\ 0 & 0 & & & (0) & \\ & -\frac{g}{2\delta x} & \frac{g}{2\delta x} & & & \\ & & \ddots & \ddots & & \\ & & & -\frac{g}{2\delta x} & \frac{g}{2\delta x} & \\ & (0) & & & 0 & 0 \\ & & & & & 0 \end{bmatrix} \in \mathbb{R}^{N\times(N-1)}, \quad \mathrm{Id}_0 := \begin{bmatrix} 0 & & & & & \\ & 0 & & & (0) & \\ & & 1 & & & \\ & & & \ddots & & \\ & & & & 1 & \\ & & (0) & & & 0 \\ & & & & & 0 \end{bmatrix} \in \mathbb{R}^{N\times N},$$

$$C := \begin{bmatrix} 1 & -\Upsilon_5^0 & \Upsilon_7^0 & & & & & \\ 1 & 0 & -\Upsilon_6^0 & 2\Upsilon_9^0 & -\Upsilon_8^0 & & & \\ & \frac{\bar{h}}{\delta t\delta x^3} & \frac{1}{\delta t}-\frac{2\bar{h}}{\delta t\delta x^3} & \frac{\bar{h}}{\delta t\delta x^3} & & (0) & & \\ & & & \ddots & \ddots & \ddots & & \\ & & (0) & & & \frac{\bar{h}}{\delta t\delta x^3} & \frac{1}{\delta t}-\frac{2\bar{h}}{\delta t\delta x^3} & \frac{\bar{h}}{\delta t\delta x^3} \\ & & & -\Upsilon_4^0 & 0 & \Upsilon_2^0 & -2\Upsilon_1^0 & 1 \\ & & & & & \Upsilon_3^0 & -\Upsilon_1^0 & 1 \end{bmatrix} \in \mathbb{R}^{N\times N},$$

$$\boldsymbol{b}^n := \begin{pmatrix} (Y_5 \tilde{*} u_1)^n - (Y_7 \tilde{*} u_2)^n \\ (Y_6 \tilde{*} u_2)^n - 2(Y_9 \tilde{*} u_3)^n + (Y_8 \tilde{*} u_4)^n \\ 0 \\ \vdots \\ 0 \\ 2(Y_1 \tilde{*} u_{J-1})^n - (Y_2 \tilde{*} u_{J-2})^n + (Y_4 \tilde{*} u_{J-4})^n \\ (Y_1 \tilde{*} u_{J-1})^n - (Y_3 \tilde{*} u_{J-2})^n \end{pmatrix} \in \mathbb{R}^N,$$

where

$$\forall\, i \in [\![1,9]\!], \quad (Y_i \tilde{*} u_j)^n := \sum_{m=1}^{n} Y_i^m u_j^{n-m}.$$

2.2 Numerical results

We present here some numerical results to analyze the efficiency of the conditions we just derived. We used the following values for the different parameters we can vary ($\bar{h}$ and $\tilde{h}$ resulted from h_0 and z):

$$\begin{aligned} &L = 1\mathrm{m}, \quad g = 9.81\mathrm{m\cdot s^{-2}}, \quad \delta t = 0.001\mathrm{s}, \quad \delta x = 0.01\mathrm{m}, \\ &h_0 = 1\mathrm{m}, \quad z = -0.53753 \times h_0, \quad \bar{h} = -0.39306\mathrm{m}^2, \quad \tilde{h} = -0.05973\mathrm{m}^3. \end{aligned} \tag{2.41}$$

For the initial conditions, we used

$$u_0 \equiv 0, \qquad \eta_0(x) = \exp\Big(-400 \times (x - L/2)^2\Big). \tag{2.42}$$

Finally, we used $r = 1.001$ for the computation of the Y_i in (2.30). The results are presented in Figure 4 and Table 1. They are quite satisfying as the solution computed with DTBC fits well the reference solution computed on a bigger domain. We used the error e^n as relative error at time t^n and e_T as global l^2 error in time:

$$e^n = \frac{\|u_{\mathrm{ref}}(\cdot, t^n) - u_{\mathrm{num}}(\cdot, t^n)\|_{l^2}}{\|u_{\mathrm{ref}}(\cdot, t^n)\|_{l^2}}, \qquad e_T = \sqrt{\delta t \times \sum_{n=1}^{n_{\max}} (e^n)^2}. \tag{2.43}$$

We used a trapezoidal rule to compute the discrete l^2-norm in space and the reference solution is computed on a bigger domain, with the same transparent conditions. We also encountered some conditioning issues when solving the linear system, but using a Jacobi pre-conditioner is enough to get a correct solution. The error is expected to be small by definition of the *transparency* of the conditions.

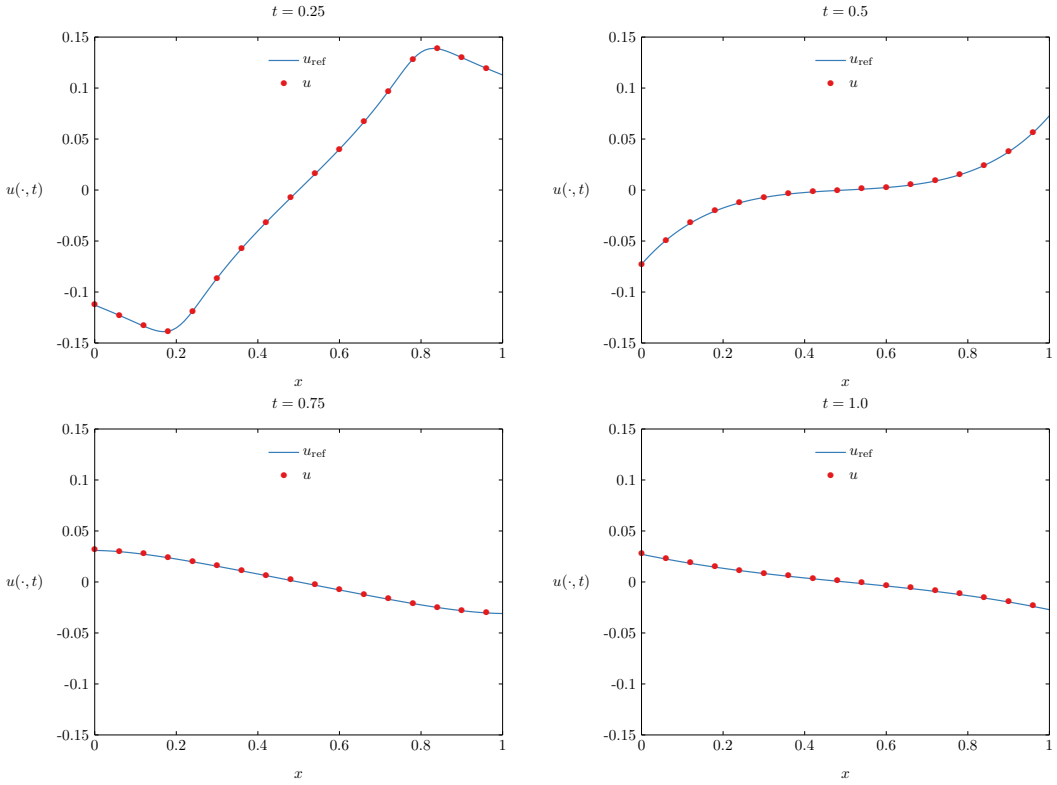

Figure 4: Snapshot at different times of the reference solution u_{ref} (computed on a bigger domain with the transparent conditions at the boundaries) and the solution u computed with the same transparent conditions.

Table 1: e^n for different times and e_T.

t	e^n
0.25	$1.5\cdot10^{-9}$
0.5	$9.5\cdot10^{-7}$
0.75	$5.2\cdot10^{-6}$
1.0	$8.4\cdot10^{-6}$

$e_T = 4.5\cdot10^{-6}$

2.3 Partial conclusion

In this section, we generalized the work of Kazakova and Noble [5] to the case of the linearized Boussinesq equations. The study is similar but what is new here is that the additional dispersive term $\tilde{h}u_{xxx}$ yields a polynomial of higher order and more conditions at the boundaries. The derived TBC revealed themselves, as expected, non-local in time. Then, we performed numerical experiments where we noticed that the conditions are really satisfying. In the next section, we will focus on the application of these conditions to domain decomposition methods.

3 Application to domain decomposition methods

The discrete boundary conditions (2.26) and (2.27) will be used in this section as *interface* boundary conditions (IBC) in a domain decomposition method (DDM). We briefly describe the DDM that we are going to use before presenting analysis and numerical results.

3.1 The Schwarz method

Domain decomposition methods are used to split a domain Ω, on which we want to solve a given problem, in multiple domains Ω_i, that can possibly overlap. Then, we can solve the problem in each domain. Hence, one must find functions that satisfy the PDE in each domain and that match with its neighbours on the interfaces, in a sense that has to be defined. The main difficulty of domain decomposition methods lies in the definition of efficient conditions at the interface between subdomains.

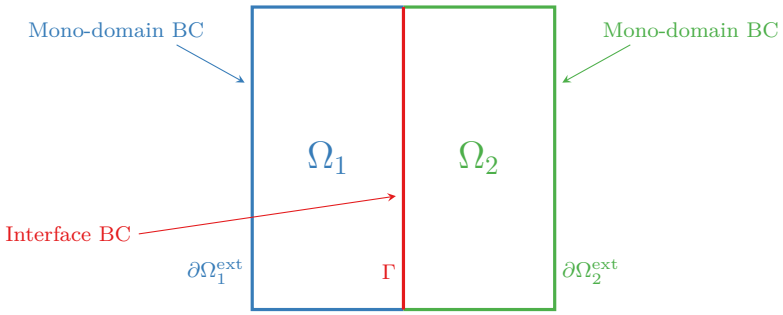

Figure 5: Decomposition of a domain $\Omega = \Omega_1 \cup \Omega_2$ into two subdomains without overlapping. On $\partial\Omega = \partial\Omega_1^{\text{ext}} \cup \partial\Omega_2^{\text{ext}}$, we use the boundary conditions of the mono-domain problem. On $\Gamma = \partial\Omega_1 \cap \partial\Omega_2$, interface boundary conditions have to be defined.

The original DDM was developed by Schwarz in 1870 [10] for a stationary problem and consists in an iterative method: the solution in the i-th subdomain Ω_i is computed as the limit of a sequence u_i^k, $k \geqslant 0$. At each iteration k, we solve the problem in each subdomain with boundary conditions at the interfaces imposed using functions from the other subdomains. We will consider here the *additive* Schwarz method (ASM) in which the interface conditions is always constructed using solutions u_j^{k-1} $(j \neq i)$ from the previous iteration in the neighbour subdomains. Therefore, at each interface between two subdomains Ω_i and Ω_j, the IBC in Ω_i is

$$\mathcal{B}_i\left(u_i^{k+1}\right) = \mathcal{B}_i\left(u_j^k\right). \tag{3.1}$$

Note that it is possible to impose several interface conditions when Ω_i has several neighbours. Initially, the operators $\mathcal{B}_i$ were Dirichlet conditions: $\mathcal{B}_i(u) = u$. For more details on the Schwarz method, the reader can refer to [11–13].

We now look for an operator $\mathcal{B}_i$ inspired from the DTBC we derived in the previous section. Without loss of generality, we consider the domain $\Omega=[0,L]$ divided into two subdomains Ω_1 (left subdomain) and Ω_2 (right subdomain) that can possibly overlap (cf. Figure 6). Ω is discretized into N nodes, while Ω_1 and Ω_2 are discretized into N_1 and N_2 nodes. The nodes in the subdomains coincide with the nodes in Ω. The interface conditions $\mathcal{B}_1$ and $\mathcal{B}_2$ will have to be such that:

- At each Schwarz iteration k, there is a unique discrete solution $u_i^{n,k}$, at each time-step t_n, in each subdomain Ω_i.
- The solution in each subdomain will have to converge (in the sense of Schwarz) to the solution on Ω restricted to this subdomain.

Note that, as problem (2.2) is a time-dependent problem, we will perform the Schwarz method at each time-step. There are other approaches for solving time-dependent PDEs with Schwarz methods, *e.g.* the Schwarz waveform relaxation method [14–16], in which the communications between the subdomains are made after the full simulations in time. Nevertheless, as shown here in the sequel, our local-in-time DDM has a very fast convergence, making it unnecessary to use global-in-time approaches, which would be more costly.

To avoid any confusion between the iteration in the Schwarz method and our time-dependent problem, the word *iteration* and the integer k will refer to the Schwarz algorithm whereas *time-step* and n, m will refer to the evolution in time of the problem we are solving.

3.2 A Schwarz method with transparent conditions at the interface

From [17], we know that transparent boundary conditions are very good candidates for interface conditions in DDM. Thus, the TBC we derived in the previous section might inspire us to set up interface conditions in an additive Schwarz method. We recall that we look for *discrete* interface conditions, so that all the future reasonings will be done with the discrete equation and conditions (2.39). If we focus on the right interface of the left domain, we recall that the transparent boundary conditions is

$$\begin{cases} \Gamma_1^r\left(\left\{u_1^{n+1}\right\}\right)=0, \\ \Gamma_2^r\left(\left\{u_1^{n+1}\right\}\right)=0. \end{cases} \tag{3.2}$$

A first heuristic is to use the left-hand-side with $u_2^{n+1,k}$ to provide a right-hand-side for the computation of $u_1^{n+1,k+1}$. With $\left\{u_1^{n+1}\right\}^k=\left(u_1^0,\ldots,u_1^n,u_1^{n+1,k}\right)$ where all the u_1^m for $0\leqslant m\leqslant n$

have been computed by the Schwarz algorithm at the previous time-steps, we have

$$\begin{cases} \Gamma_1^r\left(\left\{u_1^{n+1}\right\}^{k+1}\right)=\Gamma_1^r\left(\left\{u_2^{n+1}\right\}^{k}\right), \\ \Gamma_2^r\left(\left\{u_1^{n+1}\right\}^{k+1}\right)=\Gamma_2^r\left(\left\{u_2^{n+1}\right\}^{k}\right). \end{cases} \tag{3.3}$$

Assuming we reached convergence at previous time-steps, we have $u_1^m = u_2^m$ at the nodes in the interface zone for $0 \leqslant m \leqslant n$. Hence, the values of u_1^m and u_2^m cancel each other and the IBC becomes (for the right boundary of the left domain)

$$\begin{cases} u_{1,J_1}^{n+1,k+1}-Y_1^0\cdot u_{1,J_1-1}^{n+1,k+1}+Y_3^0\cdot u_{1,J_1-2}^{n+1,k+1}=u_{2,J_{2,1}}^{n+1,k}-Y_1^0\cdot u_{2,J_{2,1}-1}^{n+1,k}+Y_3^0\cdot u_{2,J_{2,1}-2}^{n+1,k}, \\ u_{1,J_1}^{n+1,k+1}-2Y_1^0\cdot u_{1,J_1-1}^{n+1,k+1}+Y_2^0\cdot u_{1,J_1-2}^{n+1,k+1}-Y_4^0\cdot u_{1,J_1-4}^{n+1,k+1} \\ \qquad =u_{2,J_{2,1}}^{n+1,k}-2Y_1^0\cdot u_{2,J_{2,1}-1}^{n+1,k}+Y_2^0\cdot u_{2,J_{2,1}-2}^{n+1,k}-Y_4^0\cdot u_{2,J_{2,1}-4}^{n+1,k}, \end{cases} \tag{3.4}$$

where $J_{2,1} = N_1+N_2-N-1$ is the index such that the node $J_{2,1}$ of the right domain coincides with the node $J_1 = N_1-1$ of the left domain. The operator defined on each side of this system will be denoted as $\mathcal{B}_1$, so that the IBC becomes

$$\mathcal{B}_1\left(u_1^{n+1,k+1}\right)=\mathcal{B}_1\left(u_2^{n+1,k}\right). \tag{3.5}$$

It is worth noting that the operator (3.5) used in (3.4) is local in time, contrarily to the operators $\Gamma_{1,2}^r$. Similarly, the IBC for the left boundary of the right domain is

$$\begin{cases} u_{2,0}^{n+1,k+1}-Y_5^0\cdot u_{2,1}^{n+1,k+1}+Y_7^0\cdot u_{2,2}^{n+1,k+1}=u_{1,O_{1,2}}^{n+1,k}-Y_5^0\cdot u_{1,O_{1,2}+1}^{n+1,k}+Y_7^0\cdot u_{1,O_{1,2}+2}^{n+1,k}, \\ u_{2,0}^{n+1,k+1}-Y_6^0\cdot u_{2,2}^{n+1,k+1}+2Y_9^0\cdot u_{2,3}^{n+1,k+1}-Y_8^0\cdot u_{2,4}^{n+1,k+1} \\ \qquad =u_{1,O_{1,2}}^{n+1,k}-Y_6^0\cdot u_{1,O_{1,2}+2}^{n+1,k}+2Y_9^0\cdot u_{1,O_{1,2}+3}^{n+1,k}-Y_8^0\cdot u_{1,O_{1,2}+4}^{n+1,k}, \end{cases} \tag{3.6}$$

where $J_2 = N_2-1$ and $O_{1,2} = N-N_2$ is the index such that the node $O_{1,2}$ of the left domain coincides with the node 0 of the right domain. Again, we rewrite this system as

$$\mathcal{B}_2\left(u_2^{n+1,k+1}\right)=\mathcal{B}_2\left(u_1^{n+1,k}\right). \tag{3.7}$$

Remark 3.1. Let us notice in (3.4) and (3.6) that the interface conditions need at least 5 nodes in the overlap zone and that it does not allow a non-overlapping ASM. It is not really an issue as the number of nodes required in the overlap zone is small and does not depend on the mesh size.

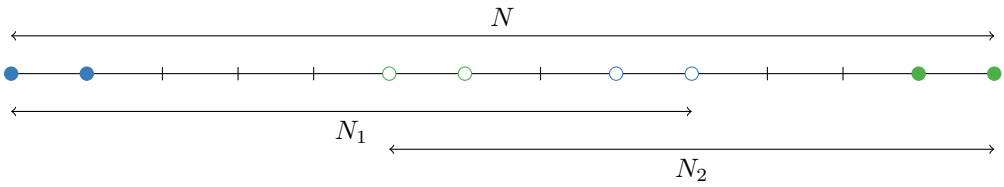

Figure 6: Decomposition of Ω into two domains with overlapping. Blue (green) nodes correspond to the left (right) domain. A filled circle represents the TBC when they are used on the external boundary (non-local in time) whereas an empty circle represents a TBC used at the interface between the two domains (local in time).

Finally, the ASM along with these interface conditions reads

$$\begin{cases} \Gamma_1^l\left(\left\{u_1^{n+1}\right\}\right)=0, \quad \Gamma_2^l\left(\left\{u_1^{n+1}\right\}\right)=0, & \text{left,} \\ u_1^{n+1,k+1}=f_1\left(u_1^n,u_1^{n-1}\right), & \text{interior,} \\ \mathcal{B}_1\left(u_1^{n+1,k+1}\right)=\mathcal{B}_1\left(u_2^{n+1,k}\right), & \text{right,} \end{cases} \tag{3.8}$$

$$\begin{cases} \mathcal{B}_2\left(u_2^{n+1,k+1}\right)=\mathcal{B}_2\left(u_1^{n+1,k}\right), & \text{left,} \\ u_2^{n+1,k+1}=f_2\left(u_2^n,u_2^{n-1}\right), & \text{interior,} \\ \Gamma_1^r\left(\left\{u_2^{n+1}\right\}\right)=0, \quad \Gamma_2^r\left(\left\{u_2^{n+1}\right\}\right)=0, & \text{right,} \end{cases}$$

where f_i stands for the discrete equation from the numerical scheme (2.39) applied to the interior of domain i.

3.3 Numerical results

We end this paper with numerical results obtained with the implementation of an additive Schwarz method to solve the linearized Boussinesq equations. We used the same parameters than in the previous tests (2.41). What we are interested in here is the number of Schwarz iterations required by the DDM (3.8) to converge to the reference solution (given by the solution computed on the domain Ω with discrete transparent conditions at the boundaries). We will use the following stopping criterion:

$$e_{\text{DDM}}^{n,k} \leqslant \varepsilon, \tag{3.9}$$

where $\varepsilon=10^{-12}$ and

$$e_{\text{DDM}}^{n,k}=\sqrt{\delta x\left(\sum_{j=0}^{J_1}\left(u_{\text{ref},j}^n-u_{1,j}^{n,k}\right)^2+\sum_{j=0}^{J_2}\left(u_{\text{ref},O_{1,2}+j}^n-u_{2,j}^{n,k}\right)^2\right)}. \tag{3.10}$$

Again, $O_{1,2}$ is the index on the mono-domain of the node corresponding to the first node of the right domain. To study the efficiency of this method, we analyse its convergence at

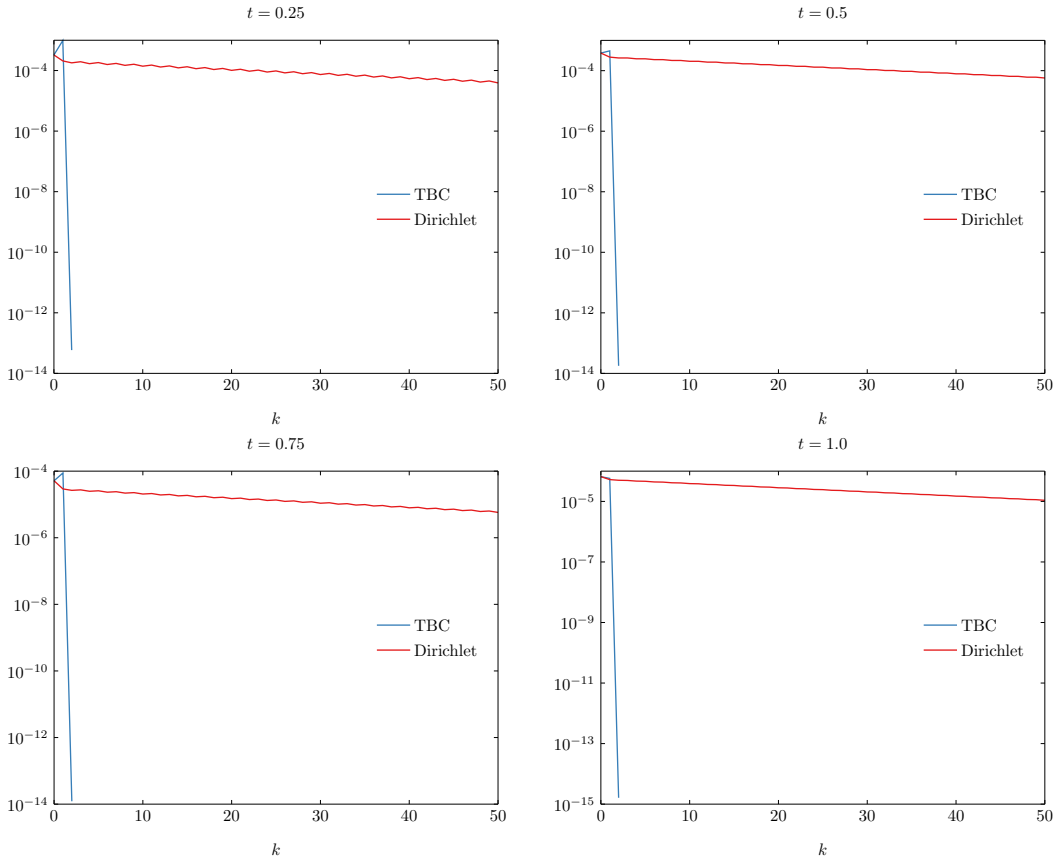

Figure 7: Evolution of $e^{n,k}_{\text{DDM}}$ at different times. We used, for the subdomains, $N_1 = 88$ and $N_2 = 18$ so that the overlap zone is of size 5 (we have $N = 101$ from (2.41)).

fixed points in time and compare our conditions at the interface to the classical Dirichlet conditions.

Results are presented in Figure 7 and Table 2 for the minimum overlap size. They are very satifying as we can see that our interface conditions make the Schwarz algorithm converge in 2 iterations only, where the Dirichlet interface conditions require several hundreds of iterations. As these conditions correspond to the fastest possible convergence (2 iterations), it is not necessary to study the influence of the size of the overlap zone nor the efficiency of a global Schwarz algorithm as it will not converge in less than 2 iterations.

Remark 3.2. We also tested conditions (2.26)–(2.27) (designed for $\bar{u} = 0$) to solve numerically (1.2) when $\bar{u} \neq 0$. If they are not adapted when used as transparent boundary conditions, they are still very efficient when used as interface conditions in a DDM: the convergence of the Schwarz method with (3.4)–(3.6) and $\bar{u} \neq 0$ is reached in 3 iterations.

Table 2: Number of iterations required for convergence in the ASM. We used, for the subdomains, $N_1=88$ and $N_2=18$ so that the overlap zone is of size 5 (we have $N=101$ from (2.41)).

t	TBC	Dirichlet
0.25	2	604
0.5	2	612
0.75	2	546
1.0	2	556

4 Conclusion

In this paper, we have been using discrete transparent boundary conditions for a class of Boussinesq equations. As expected these conditions (non-local in time) provide very satisfying results with respect to the wave reflection at boundaries. When implemented in a domain decomposition framework, the new conditions happen to be both very efficient and local in time, which provides the best possible framework for the simulation in large domains using decomposition techniques.

In future works, we shall both adapt these results to nonlinear and 2D Boussinesq-type equations and study the extension of domain decomposition algorithms to coupling techniques (*e.g.* to simulate a dispersive/non-dispersive transition).

Acknowledgements

The authors thank E. Blayo for fruitful discussions and helpful comments. This work was partially supported by the Marine Energy Research & Innovation Center (MERIC) project CORFO 14CEI2-28228 and the Fundación Inria Chile.

References

[1] D. Peregrine. Long waves on a beach. *J. Fluid Mech.*, 27:815-827, 1967.
[2] A.E. Green and P M. Naghdi. A derivation of equations for wave propagation in water of variable depth. *J. Fluid Mech.*, 78(2):237-246, 1976.
[3] O. Nwogu. Alternative form of Boussinesq equations for Nearshore wave propagation. *J. Waterw. Port Coasta. Ocean Eng.*, 119(6):618-638, 1993.
[4] V. Roeber and K.F. Cheung. Boussinesq-type model for energetic breaking waves in fringing reef environment. *Coast. Eng.*, 70:1-20, 2012.
[5] M. Kazakova and P. Noble. Discrete transparent boundary conditions for the linearized green-naghdi system of equations. submitted 2018.
[6] X. Antoine, A. Arnold, C. Besse and *et al*. A Review of transparent and artificial boundary conditions techniques for linear and nonlinear Schrdinger equations. *Commun. Comput. Phys.*, 4:729-796, 2008.
[7] A. Arnold and M. Ehrhardt. Discrete transparent boundary conditions for the Schrdinger equation. *Rivista di Matematica della Universit di Parma*, 6, 2001.

[8] C. Besse, M. Ehrhardt, and I. Lacroix-Violet. Discreteartificial boundary conditions for the Korteweg-de Vries equation. *Numer. Meth. Part Differ. Equ.*, 2015.
[9] C. Besse, P. Noble and D. Sanchez. Discrete transparent boundary conditions for the mixed KDV-BBM equation. *J. Comput. Phys.*, 345:484-509, May 2017.
[10] H. A. Schwarz. Ueber einen Grenzübergang durch alternirendes Verfahren. *Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich*, 15:272-286, 1870.
[11] M. J. Gander. Schwarz methods over the course of time. *Electron. Trans. Numer. Anal.*, 31:228-255, 2008.
[12] P.-L. Lions. On the Schwarz alternating method I. In *First International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 1-42, Philadelphia, 1988. R. Glowinski, G.H. Golub, G.A. Meurant, J. Priaux (eds.), SIAM.
[13] P.-L. Lions. On the Schwarz alternating method III: A variant for nonoverlapping subdomains. In *Third International Symposium on Domain Decomposition Methods for Partial Differential Equations*, pages 202-223, Houston, 1989. T.F. Chan, R. Glowinski, J. Priaux, O. Widlund (eds.), SIAM.
[14] M. J. Gander and H. Zhao. Overlapping schwarz waveform relaxation for the heat equation in n dimensions. *Bit*, 42(4):779-795, Dec 2002.
[15] L. Halpern, C. Japhet, and J. Szeftel. Optimized schwarz waveform relaxation and discontinuous Galerkin time stepping for heterogeneous problems. *SIAM J. Numer. Anal.*, 50(5):2588-2611, 2012.
[16] L. Halpern and J. Szeftel. Optimized and quasi-optimal Schwarz waveform relaxation for the one-dimensional Schrdinger equation. *Math. Models Meth. Appl. Sci.*, 20(12):2167-2199, 2010.
[17] C. Japhet and F. Nataf. The best interface conditions for domain decomposition methods: absorbing boundary conditions. In *Absorbing Boundaries and Layers, Domain Decomposition Methods: Applications to Large Scale Computations*, pages 348-373, New York, 2001. L. Tourrette and L. Halpern (eds.), Nova Science Publisher, Inc.
[18] A. Arnold. Numerically absorbing boundary conditions for quantum evolution equations. *VLSI Design*, 1998.
[19] A. Arnold, M. Ehrhardt and I. Sofronov. Discrete transparent boundary conditions for the Schrodinger equation: Fast calculation, approximation, and stability. *Commun. Math. Sci.*, 1, 2003.
[20] J. G. Caldas Steinstraesser, R. Cienfuegos and *et al*. A Schwarz-based domain decomposition method for the dispersion equation. *J. Appl. Anal. Comput.*, 2018.
[21] J.-F. Gerbeau and B. Perthame. Derivation of Viscous Saint-Venant system for Laminar shallow water; numerical validation. *Discret. Contin. Dyn. Syst.-Ser. B*, 1(1):89-102, 2000.