1 Images and tables

1.1 Approximations for the TBCs - Constant and linear polynomial

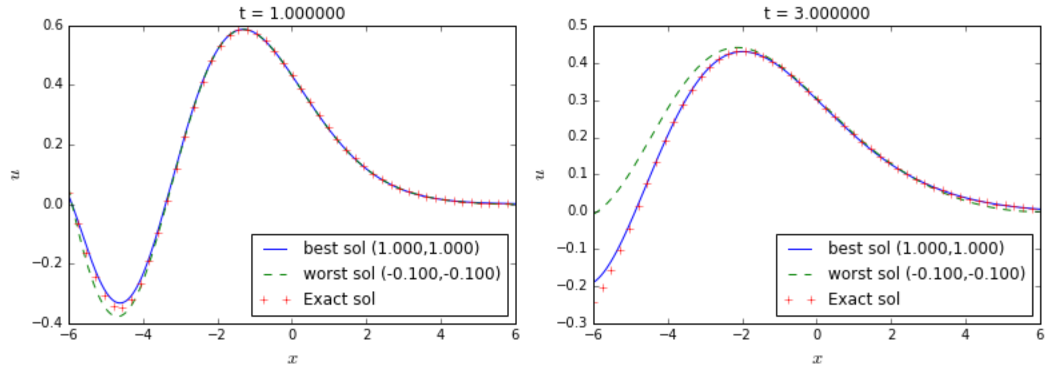


Figure 1: Best and worst solution compared with analytical solution, for the constant polynomial approximation

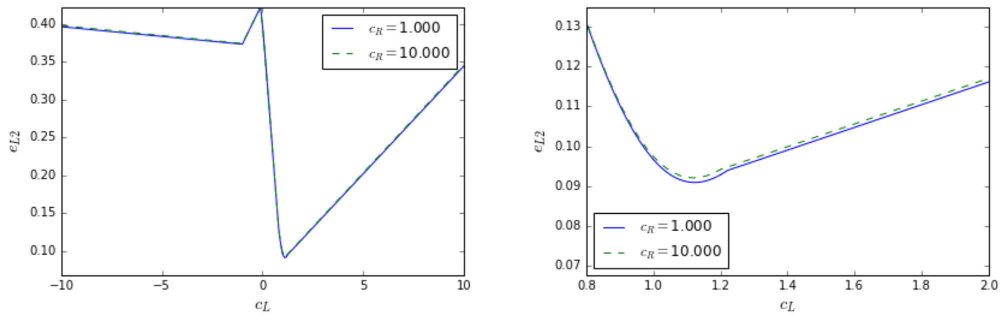
c_L	c_R	e_{L2}
1.0	1.0	0.0947
1.0	10.0	0.0973
1.0	0.1	0.0984
1.0	0.0	0.0992
1.0	-10.0	0.0994
1.0	-0.1	0.1000
1.0	-1.	0.1016
10.0	1.0	0.3470
10.0	0.1	0.3474
10.0	0.0	0.3475

Table 1: Best results (smallest e_{L2}) for the constant polynomial approximation

$d_L = d_R$	$c_L = c_R$	e_{L2}
0.	1.0	0.0947
0.1	1.0	0.1234
1.0	1.0	0.2003
10.0	0.1	0.2204
-10.0	0.1	0.2398
10.0	1.0	0.2716
-10.0	0.0	0.2480
-10.0	1.0	0.3004
10.0	0.0	0.2721
0.0	0.1	0.3674

Table 2: Best results (smallest e_{L2}) for the linear polynomial approximation

1.2 Optimization of the coefficients for the TBC (minimization of the error compared to the analytical solution)



(a) General view of all the tested coefficients

(b) Detail for $c_L \in [0.8, 2.0]$

Figure 2: Error of the numerical solution compared to the analytical solution as function of the constant polynomial approximation for the TBC

1.3 Numerical verification of the error in the DDM

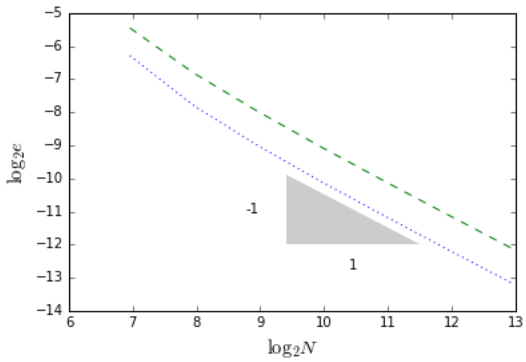
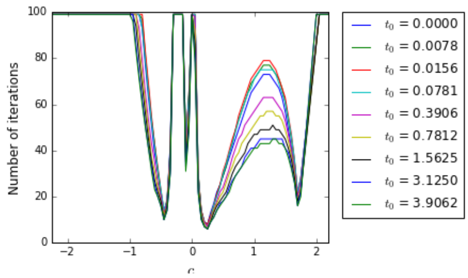
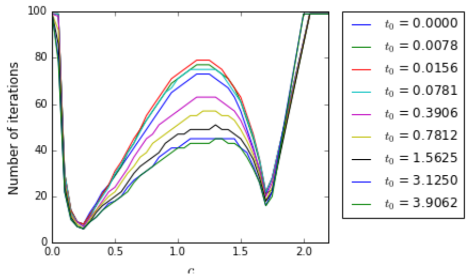


Figure 3: Numerical verification of the order of convergence of the error due to the Domain Decomposition Method

1.4 Optimization of the TBCs for the DDM (speed of convergence)

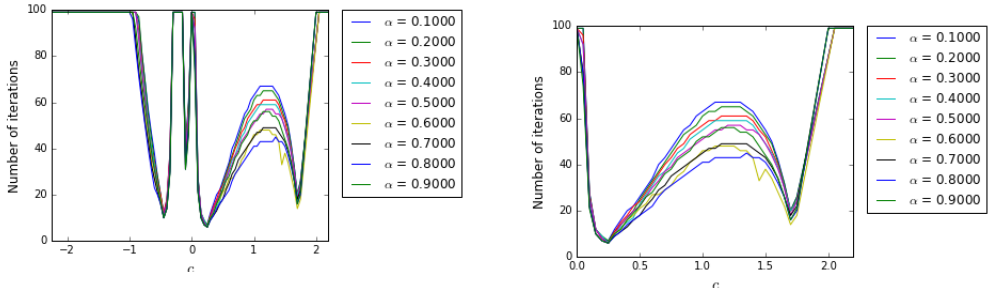


(a) General view for positive and negative coefficients



(b) Detail for $c \geq 0$.

Figure 4: Number of iterations until the convergence as function of the coefficient of the TBC (for a fixed interface and different values of t_0)



(a) General view for positive and negative coefficients

(b) Detail for $c \geq 0$

Figure 5: Number of iterations until the convergence as function of the coefficient of the TBC (for a fixed t_0 and different positions of the interface

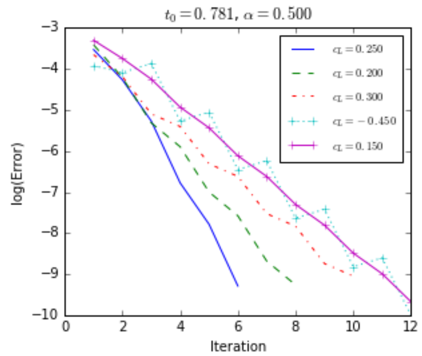


Figure 6: Error evolution with the iterations for the fastest results

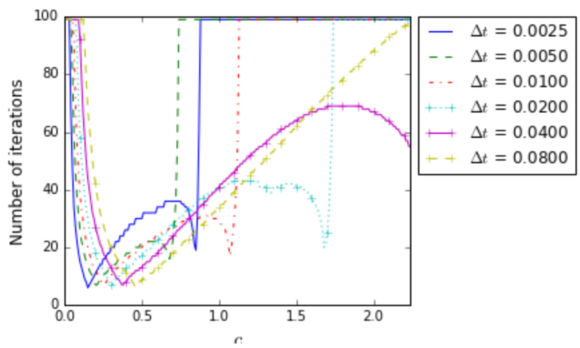


Figure 7: Number of iterations until the convergence as function of the coefficient of the TBC (for a fixed $N = 499$ and different values of dt)

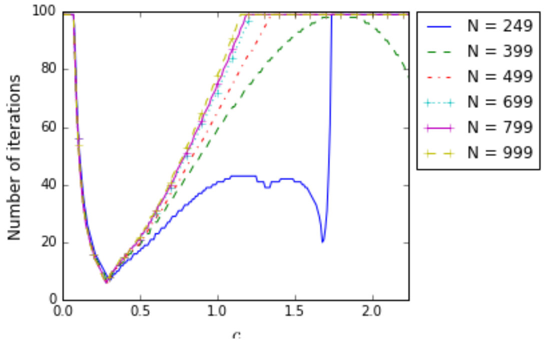


Figure 8: Number of iterations until the convergence as function of the coefficient of the TBC (for a fixed $dt = 0.02$ and different values of dx)

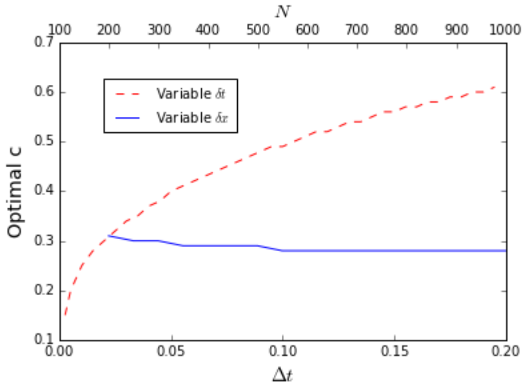


Figure 9: Optimal coefficients as function of the time step and the space step

2 Derivation of Transparent Boundary Conditions for the linearized KdV equation

In [?], Transparent Boundary Conditions (TBCs) are derived for the one-dimensional continuous linearized KdV equation (or Airy equation) :

$$u_t + U_1 u_x + U2u_{xxx} = h(t,x), \quad t \in \mathbb{R}^+, \quad x \in \mathbb{R} \tag{1}$$

where $U_1 \in \mathbb{R}$, $U_2 \in \mathbb{R}_*^+$ and h is a source term.
For an homogeneous initial boundary value problem

$$u_t + U_1 u_x + U2u_{xxx} = h(t,x), \quad t \in \mathbb{R}^+, \quad x \in [a,b] \tag{2}$$

$$u(0,x) = u_0(x), \quad x \in [a,b] \tag{3}$$

$$+\text{boundary conditions}$$

the TBCs are given by

$$u(t,a) - U_2\mathcal{L}^{-1}\left(\frac{\lambda_1(s)^2}{s}\right) * u_x(t,a) - U_2\mathcal{L}^{-1}\left(\frac{\lambda_1(s)}{s}\right) * u_{xx}(t,a) = 0 \quad (4)$$

$$u(t,b) - \mathcal{L}^{-1}\left(\frac{1}{\lambda_1(s)^2}\right) * u_{xx}(t,b) = 0 \quad (5)$$

$$u_x(t,b) - \mathcal{L}^{-1}\left(\frac{1}{\lambda_1(s)}\right) * u_{xx}(t,b) = 0 \quad (6)$$

where $\mathcal{L}^{-1}$ denotes the inverse Laplace transform, $s \in \mathbb{C}$ is the Laplace frequency and λ_1 is, among the three roots of the cubic characteristic equation obtained when solving (**??**) in the Laplace space, the only one with positive real part.
In this paper, we will focus on the special case $U_1 = 0, U_2 = 1$, which results in the KdV with only the dispersive part. We will call it as dispersive KdV equation (DKdV) :

$$u_t + u_{xxx} = 0 \quad (7)$$

Also accordingly to [**?**], in this case the only root with positive real part is

$$\lambda(s) = \lambda_1(s) = -\sqrt[3]{s} \quad (8)$$

3 Approximations for the Transparent Boundary Conditions

The computation of the TBCs (4) - (6) is not simple due to the inverse Laplace transform that must be performed, which makes these conditions to be nonlocal in time. Therefore, we will propose approximations of the root (8) that avoid integrations in time, making the TBCs very simple.
Obviously, as we can see through the results shown in this section, the resulting boundary conditions are not so precise as the ones proposed by [**?**] (who propose TBCs derived for the discrete linearized KdV equation). Nevertheless, the objectives of our work and the work of [**?**] are very different : while they seek to minimize the error of the computed solution due to the boundary conditions, we want to propose here a Domain Decomposition Method ....
We firstly notice that we can rewrite (4)-(6) as

$$\begin{cases} u(t,a) - U_2\mathcal{L}^{-1}\left(\frac{\lambda_1(s)^2}{s}\hat{u}_x(t,s)\right) - U_2\mathcal{L}^{-1}\left(\frac{\lambda_1(s)}{s}\hat{u}_{xx}(t,s)\right) = 0 \\ u(t,b) - \mathcal{L}^{-1}\left(\frac{1}{\lambda_1(s)^2}\hat{u}_{xx}(t,s)\right) = 0 \\ u_x(t,b) - \mathcal{L}^{-1}\left(\frac{1}{\lambda_1(s)}\hat{u}_{xx}(t,s)\right) = 0 \end{cases} \tag{9}$$

where$\hat{u}(s,x) = \mathcal{L}u(t,x)$ the Laplace transform in t of u. Moreover, the inverse Laplace transform satisfies the following properties :

- Linearity :

$$\mathcal{L}^{-1}\left[a_1\hat{u}_1(s,x) + a_2\hat{u}_2(s,x)\right] = a_1u_1(t,x) + a_2u_2(x,t) \tag{10}$$

- Derivative :

$$\mathcal{L}^{-1}\left[s\hat{u}(s,x)\right] = u_t(s,t) + \mathcal{L}^{-1}\left[u(0,x)\right] = u_t(s,t) + u(0,x)\delta(t) \tag{11}$$

- Convolution :

$$\mathcal{L}^{-1}\left[\hat{u}_1(s,x)\hat{u}_2(s,x)\right] = \mathcal{L}^{-1}\left[\hat{u}_1(s,x)\right] * \mathcal{L}^{-1}\left[\hat{u}_2(s,x)\right] \tag{12}$$

where $\delta(t)$ is the Dirac delta function and $*$ denotes the convolution operator. The properties (10) to (12) motivate us to approximate the operands of the inverse Laplace transforms in (4)-(5) by polynomials in s. Therefore, we use initially a constant polynomial for this approximation. Before that, we remark the following useful relations between the mentioned operands, as a consequence of (8)

$$\frac{\lambda}{s} = -\left(\frac{\lambda^2}{s}\right)^2 \tag{13}$$

$$\frac{1}{\lambda_1(s)^2} = \left(\frac{\lambda^2}{s}\right) \tag{14}$$

$$\frac{1}{\lambda_1(s)} = -\frac{\lambda^2}{s} \tag{15}$$

3.0.1 Approximation of the TBCs using a constant polynomial

Using (13)-(15), we will use the constant polynomial $P_0(s) = c$ for the following approximations :

$$\frac{\lambda^2}{s} = c \tag{16}$$

$$\frac{\lambda}{s} = -c^2 \tag{17}$$

$$\frac{1}{\lambda_1(s)^2} = c^2 \tag{18}$$

$$\frac{1}{\lambda_1(s)} = -c \tag{19}$$

Replacing in (4)-(5) and considering possibly different polynomial approximations for the left and the right boundaries (respectively with the coefficients c_L and c_R), we get the approximate Transparent Boundary Conditions :

$$\begin{cases} u(t,a) - cu_x(t,a) + c^2 u_{xx}(t,s) = 0 \\ u(t,b) - c^2 \hat{u}_{xx}(t,s) = 0 \\ u_x(t,b) + cu_{xx}(t,s) = 0 \end{cases} \tag{20}$$

Using finite difference approximations, (20) is discretized as

$$\begin{cases} u_0 - c_L \frac{u_1 - u_0}{\Delta x} + c_L^2 \frac{u_0 - 2u_1 + u_2}{\Delta x^2} = 0 \\ u_N - c_R^2 \frac{u_N - 2u_{N-1} + u_{N-2}}{\Delta x^2} = 0 \\ \frac{u_N - u_{N-1}}{\Delta x} + c_R^2 \frac{u_N - 2u_{N-1} + u_{N-2}}{\Delta x^2} = 0 \end{cases} \tag{21}$$

3.0.2 Initial numerical experiments

In order to validate our approximation, observe the general its general behavior when varying the constant approximations c_L and c_R and compare our results with the ones obtained by [?], we will solve the same numerical test presented in his paper. This problem was originally treated by [?], in the study of numerical solutions for the KdV equation.

$$u_t + u_{xxx} = 0, \quad x \in \mathbb{R} \tag{22}$$

$$u(0,x) = e^{-x^2}, \quad x \in \mathbb{R} \tag{23}$$

$$u \to 0, \quad |x| \to \infty \tag{24}$$

The fundamental solution of (22) is

$$E(t,x) = \frac{1}{\sqrt[3]{3t}} Ai\left(\frac{x}{\sqrt[3]{3t}}\right) \tag{25}$$

where Ai is the Airy function. The exact solution for the problem (22) - (24) is

$$u_{exact}(t,x) = E(t,x) * e^{-x^2} \tag{26}$$

The problem will be solved in the spatial domain $[-6,-6]$
For a quantitative evaluation of the results, we will calculate the same errors defined in the paper of [?]. For each time step, we compute the relative error

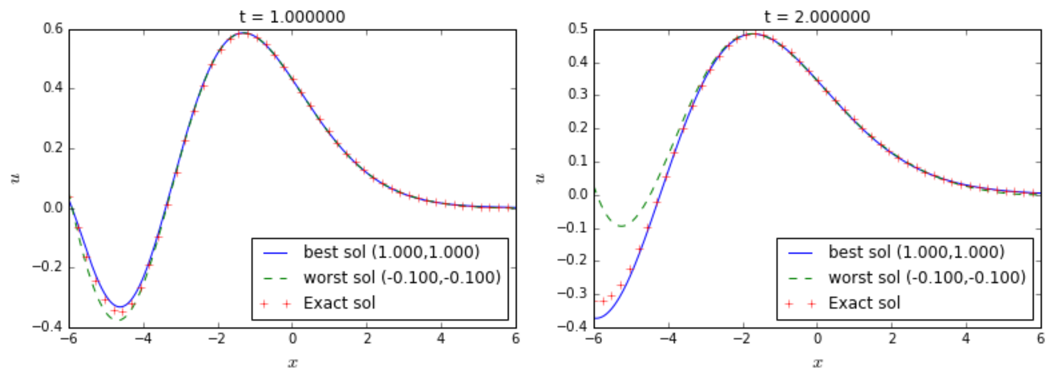
$$e^n = \frac{\left\| u^n_{exact} - u^n_{computed} \right\|_2}{\left\| u^n_{exact} \right\|_2}$$

and, in the whole time interval :

$$e_{Tm} = \max_{0<n<T_{max}} (e^n)$$

$$e_{L2} = \sqrt{\Delta t \sum_{n=1}^{T_{max}} (e^n)^2}$$

In order to verify the influence of c_L and c_R on the computed solutions (and possibly identify a range of values that better approximate the TBCs), we made several tests with all the possible pairs $c_L, c_R \in -10, -1, -0.1, 0, 0.1, 1, 10^2$. The results were classified accordingly to their errors e_{L2} (a criteria based on the error e_{Tm} gives a similar result). The figure 10 shows, for some instants, a comparison between the best, the worst and the exact solution. For naming the worst result, we did not considered the ones in which the numerical solution diverged (following the arbitrary criteria $e_{L2} > 10$).



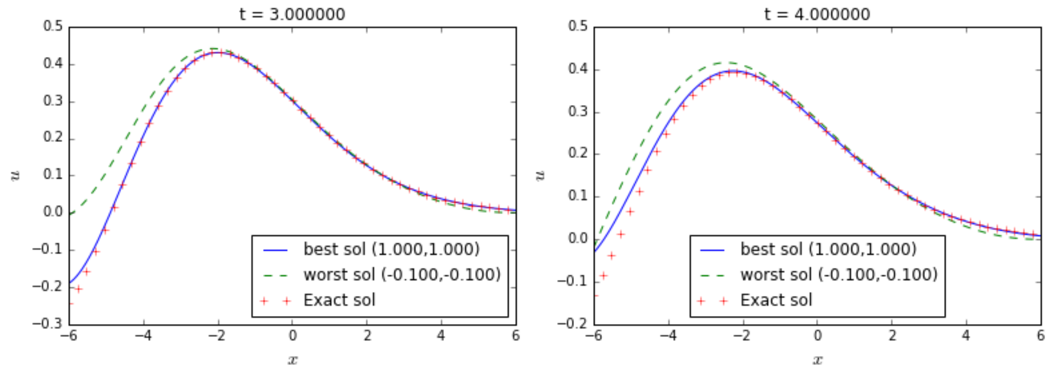


Figure 10: Best and worst solution compared with analytical solution, for the constant polynomial approximation

The table 3 presents the ten smallest e_{L2}.

c_L	c_R	e_{L2}
1.0	1.0	0.0947
1.0	10.0	0.0973
1.0	0.1	0.0984
1.0	0.0	0.0992
1.0	-10.0	0.0994
1.0	-0.1	0.1000
1.0	-1.	0.1016
10.0	1.0	0.3470
10.0	0.1	0.3474
10.0	0.0	0.3475

Table 3: Best results (smallest e_{L2}) for the constant polynomial approximation

We notice that the results are much sensitive to the coefficient on the left boundary : for a fixed c_L, the error is very similar for every c_R. This is a consequence of the fact that the solution of this specific problem is practically constant and equal to zero near $x = 6$, but presents strong variations in time near $x = -6$.

3.1 Approximation of the TBCs using a linear polynomial

In a similar way as done above, we approximate $\frac{\lambda^2}{s}$ by $P_1(s) = ds+c$. Without repeating the details of the derivation, we obtain the following discretized approximate TBCs :

$$u_0^{n+1} - \left(\frac{d_L}{\Delta t} + c_L\right)\left(\frac{u_1^{n+1} - u_0^{n+1}}{\Delta x}\right) + \left(\frac{d_L^2}{\Delta t^2} + \frac{2d_L c_L}{\Delta t} + c_L^2\right)\left(\frac{u_0^{n+1} - 2u_1^{n+1} + u_2^{n+1}}{\Delta x^2}\right) = -\frac{d_L}{\Delta t}\left(\frac{u_1^n - u_0^n}{\Delta x}\right) + \left(2\frac{d_L^2}{\Delta t^2} + \frac{2d_L c_L}{\Delta t}\right)\left(\frac{u_0^n - 2u_1^n + u_2^n}{\Delta x^2}\right) + -\frac{d_L^2}{\Delta t^2}\left(\frac{u_0^{n-1} - 2u_1^{n-1} + u_2^{n-1}}{\Delta x^2}\right) \tag{27}$$

$$u_N^{n+1} - \left(\frac{d_R^2}{\Delta t^2} + \frac{2d_R c_R}{\Delta t} + c_R^2\right)\left(\frac{u_N^{n+1} - 2u_{N-1}^{n+1} + u_{N-2}^{n+1}}{\Delta x^2}\right) = -\left(2\frac{d_R^2}{\Delta t^2} + \frac{2d_R c_R}{\Delta t}\right)\left(\frac{u_N^n - 2u_{N-1}^n + u_{N-2}^n}{\Delta x^2}\right) + \frac{d_R^2}{\Delta t^2}\left(\frac{u_N^{n-1} - 2u_{N-1}^{n-1} + u_{N-2}^{n-1}}{\Delta x^2}\right) \tag{28}$$

$$\frac{u_N^{n+1} - u_{N-1}^{n+1}}{\Delta x} + \left(\frac{d_R}{\Delta t} + c_R\right)\left(\frac{u_N^{n+1} - 2u_{N-1}^{n+1} + u_{N-2}^{n+1}}{\Delta x^2}\right) = \frac{d_R}{\Delta t}\left(\frac{u_N^n - 2u_{N-1}^n + u_{N-2}^n}{\Delta x^2}\right) \tag{29}$$

3.1.1 Initial numerical experiments

We repeated the numerical tests that we made for the approximation with P_0, making the coefficients c_L and d_L assume the values in $-10, -1, -0.1, 0, 0.1, 1, 10$. In order to avoid a too high computation, and also taking in account the remark we made above about the weak dependence of the results on the coefficients in the right boundary, we realized all the tests assuming $c_R = c_L$ and $d_R = d_L$.
We present in the table 4 the ten best results. As we can see, the best result is the one where $d_L = d_R = 0$ and $c_L = c_R = 1.$, which corresponds to the best results of the approximations using constant polynomials.

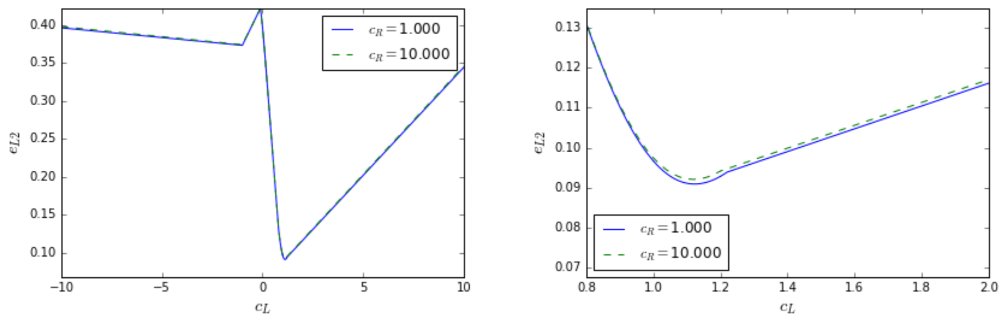
$d_L = d_R$	$c_L = c_R$	e_{L2}
0.	1.0	0.0947
0.1	1.0	0.1234
1.0	1.0	0.2003
10.0	0.1	0.2204
-10.0	0.1	0.2398
10.0	1.0	0.2716
-10.0	0.0	0.2480
-10.0	1.0	0.3004
10.0	0.0	0.2721
0.0	0.1	0.3674

Table 4: Best results (smallest e_{L2}) for the linear polynomial approximation

3.2 Partial conclusion

3.3 Optimization of the approximate TBCs (minimization of the errors compared to the analytical solution)

The initial tests made up to here allowed us to validate the approximation of the TBCs proposed by [?] with a constant polynomial in each boundary, guiding us in the following work. In this subsection, we investigate deeply the influence of this approximation over the error of the computed solution, compared with the analytical one. For this purpose, we repeat the computations, but with a much more refined range of coefficients, always using as criteria of quality the error e_{L2} of each test.
As a consequence of the remarks made in the last subsection, this refinement is higher for the coefficient c_L. By making a gradual study, in which we identified at each step the intervals of c_L which give the best results and studied it even more deeply (up to a step 0.01 for the variation of c_L), we were able to construct the curves presented in the figure **??** and identify the empirical optimum $c_L = 1.16$:



(a) General view of all the tested coefficients

(b) Detail for $c_L \in [0.8, 2.0]$

Figure 11: Error of the numerical solution compared to the analytical solution as function of the constant polynomial approximation for the TBC

4 Application to a Domain Decomposition Method

The constant polynomial approximation of the Transparent Boundary Conditions, expressed by **??**, will be applied to the implementation of a Domain Decomposition Method (DDM). Firstly, we will briefly describe the DDM that we will consider here, and after we will describe and test the incorporation of the proposed TBCs.

4.1 The Schwarz Methods

The following description is based on [**?**]. Domain Decomposition Methods allow to decompose a domain Ω in multiple subdomains Ω_i (that can possibly overlap) and solve the problem in each one of them. Therefore, one must find functions that satisfies the PDE in each subdomain and that match on the interfaces.

The first DDM developed was the Schwarz method, which consists on an iterative method : in the case of a evolution problem, the solution $u_i^{n,\infty}$, in each time step t_n and each subdomain Ω_i, is computed as the convergence of the solution obtained in each iteration, $u_i^{n,k}, \quad k \geq 0$. There are two types of Schwarz methods, depending on the way that the boundary conditions on the interfaces are constructed for computing $u_i^{n,k}$

We will consider her the additive Schwarz method (ASM), in which the boundary conditions are always constructed using the solution $u_j^{n,k-1}, \quad j \neq i$ of the previous iteration in the other partitions of the domain. Therefore, in each interface between the domains Ω_i and Ω_j, the boundary condition for the problem in Ω_i is

$$\mathcal{B}(u_i^{n,k+1}) = \mathcal{B}(u_j^{n,k})$$

where $\mathcal{B}$ denotes the operator of the TBC.

Without loss of generality, in the following we will consider a domain decomposed in two non-overlapping subdomains.

Evidently, the biggest challenge of the Schwarz methods is to define appropriate operators such that :

- The method shows a fast convergence
- The solution u_i in each subdomain Ω_i converges to $u|_{\Omega_1}$, i.e, the solution u of the monodomain Ω restricted to Ω_1

In fact, accordingly to [?], the optimal additive Schwarz method is the one which uses the TBCs **??** as interface boundary conditions: with them, the method converges in two iterations, and no ASM can converge in less than two iterations.
Nevertheless, as discussed previously in this report, the numerical implementation of the exact TBCs **??** are generally impractical, because they are non local in time. Therefore, one should use approximate TBCs, what will be done here in the sequence using the approximations proposed n the previous section.

4.2 ASM with the approximate TBCs for the dispersive equation

The resolution of the dispersive equation with the Additive Schwarz method, using the constant polynomial approximation for the TBCs, is written as

$$\begin{cases} (u_1^{n,k+1})_t + (u_1^{n,k+1})_{xxx} = 0, \quad x \in \Omega_1, \quad t \leq 0 \\ u_1^{n,0} = u_1^{n-1,\infty}, \quad x \in \Omega_1 \\ \Upsilon_1^{c_L^*}(u_1^{n+1,k+1}, -L) = 0, \\ \Theta_2^{c_R^*}(u_1^{n+1,k+1}, 0) = \Theta_2^{c_R^*}(u_2^{n,k}, 0), \\ \Theta_3^{c_R^*}(u_1^{n+1,k+1}, 0) = \Theta_3^{c_R^*}(u_2^{n,k}, 0) \end{cases} \tag{30}$$

$$\begin{cases} (u_2^{n,k+1})_t + (u_2^{n,k+1})_{xxx} = 0, \quad x \in \Omega_2, \quad t \leq 0 \\ u_2^{n,0} = u_2^{n-1,\infty}, \quad x \in \Omega_2 \\ \Theta_1^{c_L^*}(u_2^{n+1,k+1}, 0) = \Theta_1^{c_L^*}(u_1^{n,k}, 0) \\ \Upsilon_2^{c_R^*}(u_2^{n+1,k+1}, L) = 0 \\ \Upsilon_3^{c_R^*}(u_2^{n+1,k+1}, L) = 0 \end{cases} \tag{31}$$

where $\Upsilon_i, \ \ i = 1, 2, 3$, are the boundary conditions on the external boundaries (i.e, in the intersection of the monodomain boundaries and the subdomain boundaries). These external BCs are independent of the interface BCs. Here, we will consider $\Upsilon_1 = \Theta_1^{1.0}$, $\Upsilon_2 = \Theta_2^{0.0}$ and $\Upsilon_3 = \Theta_3^{0.0}$, which gives

$$\Upsilon_1(u, x) = u - u_x + u_{xx} \tag{32}$$

$$\Upsilon_2(u, x) = 0 \tag{33}$$

$$\Upsilon_3(u, x) = 0 \tag{34}$$

$$\tag{35}$$

This choice was made based on the easy implementation and the good results provided by the coefficients $c_L = 1.0$ and $c_R = 0.0$ in approximating the analytical solution in Ω (as shown in the table 3). Nevertheless,it does not have much importance in the study that we will done in the following paragraphs. In fact, our purpose is to study exclusively the behavior of the DDM implemented here; therefore, all the results must be compared to a referential solution u_{ref}, that can be simply the numerical solution of the monodomain problem. The only restriction for an appropriate study is that the external BCs for computing u_{ref} must be same as $\Upsilon_i, \quad i = 1, 2, 3$.

4.3 Error in the converged solution

When using approximate TBCs in the ASM, one should guarantee that the converged solutions u_1, u_2 satisfies the same equation as the solution u_{ref} of the monodomain problem. In the following paragraphs, we show that this property is not verified by the method (**??**) - (**??**) proposed here. Based on that, we will be able to propose corrections for it.

4.3.1 Study of the error in the DDM method for the dispersion equation

Now we will apply a similar idea to study the error of the Domain Decomposition Method that we proposed for the dispersive equation. As mentioned above, we will focus exclusively on the error produced by DDM, independently of the other possible components of the error compared to the exact solution for the example (external boundary conditions, error accumulation over the time steps). Therefore, all the following study will be made considering the execution of the method over only one time step, allowing us to use a clearer notation for the solution : u_j^i, where i indicates the subdomain Ω_i and j indicates the spatial discrete position. In the cases where the iterative process is take in account, we will add the superscript k to indicate the iteration.
For the interior points of each one of the domains, we will consider a second order spatial discretization of the equation (**??**).

$$\frac{u_j^i - \alpha_j^i}{\Delta t} + \frac{-\frac{1}{2}u_{j-2}^i + u_{j-1}^i - u_{j+1}^i + \frac{1}{2}u_{j+2}^i}{\Delta x^3} = 0 \qquad (36)$$

for $j = 2, ..., N-2$ in the case $i = 1$; for $j = N+2, ..., 2N-2$ in the case $i = 2$; and for $j = 2, ..., 2N-2$ in the case $i = ref$. In the above expression, α_j^i is a given data (for example, the converged solution in the previous time step).

For the points near the boundaries, we use second order uncentered discretizations or the appropriate TBCs. We will focus here on the interface point x_N. Thus, in the resolution of the problem in Ω_1, two interface boundary conditions are imposed (corresponding to Θ_2 and Θ_3), so the solutions u^1_{N-1} and u^1_N are computed using the equations

$$\begin{aligned} &\Theta_2^{c_R}(u^1_N) = \Theta_2^{c_R}(u^2_N) \implies \\ &\implies u^1_N - c_R^2\frac{u^1_N - 2u^1_{N-1} + u^1_{N-2}}{\Delta x^2} = u^2_N - c_R^2\frac{u^2_N - 2u^2_{N+1} + u^2_{N+2}}{\Delta x^2} \end{aligned} \tag{37}$$

$$\begin{aligned} &\Theta_3^{c_R}(u^1_N) = \Theta_3^{c_R}(u^2_N) \implies \\ &\implies \frac{u^1_N - u^1_{N-1}}{\Delta x} + c_R\frac{u^1_N - 2u^1_{N-1} + u^1_{N-2}}{\Delta x^2} = \frac{u^2_{N+1} - u^2_N}{\Delta x} + c_R\frac{u^2_N - 2u^2_{N+1} + u^2_{N+2}}{\Delta x^2} \end{aligned} \tag{38}$$

In the resolution of the problem in Ω_2, only one interface boundary condition is used (corresponding to Θ_1) :

$$\begin{aligned} &\Theta_3^{c_L}(u^2_N) = \Theta_3^{c_L}(u^1_N) \implies \\ &\implies u^2_N - c_L\frac{u^2_{N+1} - u^2_N}{\Delta x} + c_L^2\frac{u^2_N - 2u^2_{N+1} + u^2_{N+2}}{\Delta x^2} = \\ &u^1_N - c_L\frac{u^1_N - u^1_{N-1}}{\Delta x} + c_L^2\frac{u^1_N - 2u^1_{N-1} + u^1_{N-2}}{\Delta x^2} \end{aligned} \tag{39}$$

In the convergence, (**??**) to(**??**) gives respectively

$$\begin{aligned} &u^*_N - c_R^2\frac{u^*_N - 2u^*_{N-1} + u^*_{N-2}}{\Delta x^2} = u^*_N - c_R^2\frac{u^*_N - 2u^*_{N+1} + u^*_{N+2}}{\Delta x^2} \implies \\ &\implies 2c_R^2\frac{-\frac{1}{2}u^*_{N-2} + u^*_{N-1} - u^*_{N+1} + \frac{1}{2}u^*_{N+2}}{\Delta x^2} = 0 \end{aligned} \tag{40}$$

$$\begin{aligned} &\frac{u^*_N - u^*_{N-1}}{\Delta x} + c_R\frac{u^*_N - 2u^*_{N-1} + u^*_{N-2}}{\Delta x^2} = \\ &\frac{u^*_{N+1} - u^*_N}{\Delta x} + c_R\frac{u^*_N - 2u^*_{N+1} + u^*_{N+2}}{\Delta x^2} \implies \\ &\implies -\frac{u^*_{N-1} - 2u^*_N + u^*_{N+1}}{\Delta x} - 2c_R\frac{-\frac{1}{2}u^*_{N-2} + u^*_{N-1} - u^*_{N+1} + \frac{1}{2}u^*_{N+2}}{\Delta x^2} = 0 \end{aligned} \tag{41}$$

$$
\begin{aligned}
&u_N^* - c_L \frac{u_{N+1}^* - u_N^*}{\Delta x} + c_L^2 \frac{u_N^* - 2u_{N+1}^* + u_{N+2}^*}{\Delta x^2} = \\
&u_N^* - c_L \frac{u_N^* - u_{N-1}^*}{\Delta x} + c_L^2 \frac{u_N^* - 2u_{N-1}^* + u_{N-2}^*}{\Delta x^2} \implies \\
\implies &- c_L \frac{u_{N-1}^* - 2u_N^* + u_{N+1}^*}{\Delta x} + 2c_L^2 \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^2} = 0
\end{aligned}
\tag{42}
$$

Therefore, we can see that, the converged solution of the DDM method satisfies the same equation as the reference solution in all the points $x_j \in \Omega_1$, except in x_{N-1} and x_N (for the problem solved in Ω_1) and in x_N (for the problem solved in Ω_2) . For example, it's easy to verify that (**??**) differ from (**??**) by a $O(\Delta x)$ term :

$$
\begin{aligned}
&2\Delta x c_R^2 \left(\frac{u_N^* - \alpha_N^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^3} \right) - \\
&\left(2c_R^2 \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^2} \right) = 2\Delta x c_R^2 \frac{u_N^* - \alpha_N^*}{\Delta t}
\end{aligned}
\tag{43}
$$

Numerical verification of the error The problem (**??**) - (**??**) was solved until the convergence with 5 different uniform spatial discretizations, over one time step (in the interval $[0, \Delta t]$). In each case, the adopted referential solution u^{ref} was the monodomain problem, solved with the same mesh size. Two errors were computed :

$$
e^{N, \Omega_1 \bigcap \Omega_2} = |u_N^{ref} - u_N^1| \tag{44}
$$

$$
e^{N,\Omega} = ||u_N^{ref} - u_N^{1,2}||_2 = \sqrt{dx} \sqrt{\sum_{j=0}^{N} (u_j^{ref} - u_j^1)^2 + \sum_{j=N}^{2N} (u_j^{ref} - u_j^2)^2} \tag{45}
$$

corresponding respectively to the error on the interface and the error on the whole domain. In the notation $e^{N,\bullet}$, N refers to the number of space steps in Ω.

We are interested in the behavior of these error as the mesh size changes. As shown in the table **??**, we verify that the DDM proposed here produces a first order error :

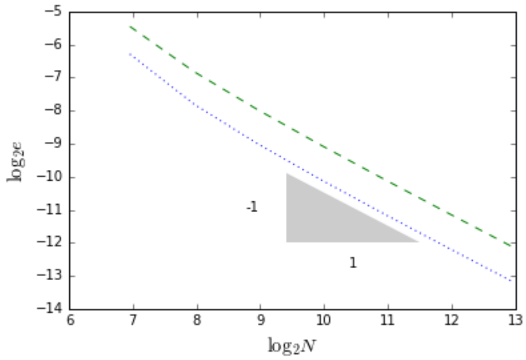


Figure 12: Numerical verification of the order of convergence of the error due to the Domain Decomposition Method

4.3.2 Corrections for the approximate TBCs

Then, we will formulate modified TBCs for the ASM method in order to cancel these errors :

$$\begin{cases} \Theta_1^{c_L^*}(u_2^{n+1,k+1},0) + \theta_1 = \Theta_1^{c_L^*}(u_1^{n,k},0) + \theta_1' \\ \Theta_2^{c_R^*}(u_1^{n+1,k+1},0) + \theta_2 = \Theta_2^{c_R^*}(u_2^{n,k},0) + \theta_2' \\ \Theta_3^{c_R^*}(u_1^{n+1,k+1},0) + \theta_3 = \Theta_3^{c_R^*}(u_2^{n,k},0) + \theta_3' \end{cases} \tag{46}$$

with θ_i, θ_i' derived below, after a numerical verification of the error produced by the DDM.

Determination of θ_1, θ_1' Our objective is to write (**??**) in the point x_N :

$$\frac{u_N^* - \alpha_N^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^3} = 0 \tag{47}$$

Defining

$$\theta_1 = c_L \frac{u_{N+1}^2 - u_N^2}{\Delta x} + c_L^2 \frac{\Delta x}{\Delta t}\left(u_N^2 - \alpha_N^2\right) \tag{48}$$

$$\theta_1' = c_L \frac{u_N^1 - u_{N-1}^1}{\Delta x} - c_L^2 \frac{\Delta x}{\Delta t}\left(u_N^1 - \alpha_N^1\right) \tag{49}$$

we have, in the convergence, that

$$\begin{aligned}
&\Theta_1^{c_L}(u_N^*) + \theta_1 = \Theta_1^{c_L}(u_N^*) + \theta_1' \implies \\
\implies & u_N^* - c_L \frac{u_{N+1}^* - u_N^*}{\Delta x} + c_L^2 \frac{u_N^* - 2u_{N+1}^* + u_{N+2}^*}{\Delta x^2} + c_L \frac{u_{N+1}^* - u_N^*}{\Delta x} + c_L^2 \frac{\Delta x}{\Delta t} (u_N^* - \alpha_N^*) = \\
& u_N^* - c_L \frac{u_N^* - u_{N-1}^*}{\Delta x} + c_L^2 \frac{u_N^* - 2u_{N-1}^* + u_{N-2}^*}{\Delta x^2} + c_L \frac{u_N^* - u_{N-1}^*}{\Delta x} - c_L^2 \frac{\Delta x}{\Delta t} (u_N^* - \alpha_N^*) \implies \\
\implies & 2c_L^2 \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^2} + 2c_L^2 \frac{\Delta x}{\Delta t} (u_N^* - \alpha_N^*) = 0 \implies \\
\implies & \frac{u_N^* - \alpha_N^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^3} = 0
\end{aligned} \tag{50}$$

which corresponds to the discretization (**??**) satisfied in x_N.

Determination of θ_2, θ_2' Based on (**??**), we define

$$\begin{aligned}
\theta_2 &= \frac{\Delta x}{\Delta t} c_R^2 (u_N^1 - \alpha_N^1) \\
\theta_2' &= -\frac{\Delta x}{\Delta t} c_R^2 (u_N^2 - \alpha_N^2)
\end{aligned} \tag{51}$$

Indeed, we then have, in the convergence

$$\begin{aligned}
&\Theta_2^{c_R}(u_N^*) + \theta_2 = \Theta_2^{c_R}(u_N^*) + \theta_2' \implies \\
\implies & u_N^* - c_R^2 \frac{u_N^* - 2u_{N-1}^* + u_{N-2}^*}{\Delta x^2} + \frac{\Delta x}{\Delta t} c_R^2 (u_N^* - \alpha_N^*) = \\
& u_N^* - c_R^2 \frac{u_N^* - 2u_{N+1}^* + u_{N+2}^*}{\Delta x^2} - \frac{\Delta x}{\Delta t} c_R^2 (u_N^* - \alpha_N^*) \implies \\
\implies & 2\frac{\Delta x}{\Delta t} c_R^2 (u_N^* - \alpha_N^*) + 2c_R^2 \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^2} = 0 \implies \\
\implies & \frac{u_N^* - \alpha_N^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-2}^* + u_{N-1}^* - u_{N+1}^* + \frac{1}{2}u_{N+2}^*}{\Delta x^3} = 0
\end{aligned} \tag{52}$$

which corresponds to the discretization (**??**) satisfied in x_N.

Determination of θ_3, θ_3' Using (**??**) in (**??**), we get

$$-\frac{u_{N-1}^* - 2u_N^* + u_{N+1}^*}{\Delta x} + 2c_R \Delta x \frac{u_N^* - \alpha_N^*}{\Delta t} = 0 \tag{53}$$

Now, our objective is to write (**??**) in the point x_{N-1} :

$$\frac{u_{N-1}^* - \alpha_{N-1}^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-3}^* + u_{N-2}^* - u_N^* + \frac{1}{2}u_{N+1}^*}{\Delta x^3} = 0 \tag{54}$$

what can be achieved by defining

$$\begin{gathered}
\theta_3 = 2\frac{\Delta x}{\Delta t}\left[-\Delta x(u_{N-1}^1 - \alpha_{N-1}^1) - c_R(u_N^1 - \alpha_N^1)\right] + \frac{u_{N-3}^1 - 2u_{N-2}^1 + u_{N-1}^1}{\Delta x} \\
\theta_3' = 0
\end{gathered} \tag{55}$$

In fact, in the convergence,

$$\begin{aligned}
&\Theta_3^{c_R}(u_N^*) + \theta_3 = \Theta_3^{c_R}(u_N^*) + \theta_3' \implies \\
\implies &\frac{u_N^* - u_{N-1}^*}{\Delta x} + c_R\frac{u_N^* - 2u_{N-1}^* + u_{N-2}^*}{\Delta x^2} + 2\frac{\Delta x}{\Delta t}\left[-\Delta x(u_{N-1}^* - \alpha_{N-1}^*) - c_R(u_N^* - \alpha_N^*)\right] + \\
&\frac{u_{N-3}^* - 2u_{N-2}^* + u_{N-1}^*}{\Delta x} = \frac{u_{N+1}^* - u_N^*}{\Delta x} + c_R\frac{u_N^* - 2u_{N+1}^* + u_{N+2}^*}{\Delta x^2} \implies \\
\implies &-\frac{u_{N-1}^* - 2u_N^* + u_{N+1}^*}{\Delta x} + 2c_R\Delta_x\frac{u_N^* - \alpha_N^*}{\Delta t} + \\
&2\frac{\Delta x}{\Delta t}\left[-\Delta x(u_{N-1}^* - \alpha_{N-1}^*) - c_R(u_N^* - \alpha_N^*)\right] + \frac{u_{N-3}^* - 2u_{N-2}^* + u_{N-1}^*}{\Delta x} = 0 \implies \\
\implies &-2\frac{-\frac{1}{2}u_{N-3}^* + u_{N-2}^* - u_N^* + \frac{1}{2}u_{N+1}^*}{\Delta x} - 2\frac{\Delta x^2}{\Delta t}(u_{N-1}^* - \alpha_{N-1}^*) = 0 \implies \\
\implies &\frac{u_{N-1}^* - \alpha_{N-1}^*}{\Delta t} + \frac{-\frac{1}{2}u_{N-3}^* + u_{N-2}^* - u_N^* + \frac{1}{2}u_{N+1}^*}{\Delta x^3} = 0
\end{aligned}$$

Modification of the reference solution The modifications proposed above for the CBCs effectively allows the points $x_{N-1}, x_N \in \Omega_1$ and $x_N \in \Omega_2$ to satisfy the same discrete equation as in the monodomain problem. Nevertheless, the solution of the DDM does not converge exactly to u^{ref}, for a reason that does not depend on the expression of the CBCs, but on the fact that for each domain we write two CBCs in the left boundry and only one on the right. We are using a second order centered discretization for the third spatial derivative (which uses a stencil of two points in each side of the central point), which implies that we must write an uncentered discretization for the point x_{N+1} when solving the problem in ω_2. Therefore, this point does not satisfy the same discrete equation. In order to avoid this problem and allow us to verify that our method is able to correct the error of the DDM, we modify the discretization for the point u_{N+1} in the monodmoain problem, using the same second-order uncentered expression

$$\frac{u_{N+1}^2 - \alpha_{N+1}^2}{\Delta t} + \frac{-\frac{5}{2}u_{N+1}^2 + 9u_{N+2}^2 - 12u_{N+3}^2 + 7\frac{1}{2}u_{N+4}^2 - \frac{3}{2}u_{N+1}^2}{\Delta x^3} = 0 \quad (56)$$

4.4 Optimization of the CBCs (speed of convergence)

Our objective now is to optimize the CBCs in the sense of minimizing the number of iterations of the ASM until the convergence. Therefore, similarly to the optimization of the TBCs made in the section **??**, we will made a very large set of tests in order to find the coefficients c_L and c_R (i.e., the constant polynomial approximation for the TBC) that provides the fastest convergence. In a first moment, we will make this study with fixed time step and space step, in order to analyze exclusively the influence of the coefficient, and after we will introduce these two parameters into the study.
As we are interested in the speed with which the solution of the DDM method converges to the reference solution, the criteria of convergence used is

$$e^k = ||\tilde{u}^{k,DDM} - \tilde{u}^{k,ref}||_2 = \sqrt{\Delta x \left[\sum_{j=0}^{N} \left(u_j^{k,1} - u_j^{k,ref} \right)^2 + \sum_{j=N}^{2N} \left(u_j^{k,2} - u_j^{k,ref} \right)^2 \right]} \quad (57)$$